

A computational model for collective
chondrocyte behaviour in osteoarthritis, in
patient-specific biochemical environments

Andreu Pascuet Fontanet



Universitat
Pompeu Fabra
Barcelona

A computational model for collective
chondrocyte behaviour in osteoarthritis, in
patient-specific biochemical
micro-environments

Andreu Pascuet Fontanet

Bachelor's Thesis UPF 2020/2021

Thesis Supervisor(s):

Dr. Jérôme Noailly, (BCN MedTech Principal Investigator)

Maria Segarra Queralt, (BCN MedTech PhD candidate)



Acknowledgments

Behind the Bachelor's Thesis that I am presenting in this report there is a great amount of support and guidance provided by friends and experts.

I would first like to thank my supervisors, Jérôme Noailly and Maria Segarra Queralt. Jérôme, I have to thank you not only because letting me become a member of your team and share your expertise and ideas to bring this project to a reality, but also for your guidance during the whole degree and the implication with us, the students. Maria, your comprehensive and constant feedback has been a basis on the development of my thesis. Besides your expertise on the topic, your continuous interest on the state of the project (and my own state) helped me to calm down even in the most stressful moments.

I would like to thank the URFOA group from IMIM to let me incorporate in their team for a short period of time and make me feel like any other member. Specially, I want to thank Laura Tió, my supervisor at the laboratory, for the valuable information and advises she brought me. I also want to mention Laura Triginer and Anna Ribes, who instructed me in the more technical part of the laboratory.

I also would like to thank the BCN MedTech group, for listening the periodic advances of my work. Concretely, I need to thank Laura Baumgartner, Carlos Ruiz and Simone Tassani because of their questions and contributions, which helped to better define the scope of my work.

Finally, I need to thank my family and friends for supporting and helping me throughout all the development of the process. Although the list is quite large, I want your names to appear here: mom, dad, Aina, Marta, Maria, Olalla, Dani, Ricard, Victoria, Paula, thank you.

Thank you all for all your support.

Moltes gràcies!

Summary/Abstract

Osteoarthritis (OA) is a common condition characterised by the degradation of the articular cartilage that leads to stiff and painful joints. OA is related with the dysregulation of the cartilage chondrocyte's activity, the latter becoming much more catabolic under OA conditions. Both mechanical and biochemical signals are involved in this dysregulation and need to be considered for the understanding of the condition and its treatment. On the one hand, cartilage tissue mechanics and multi-physics have been described through numerical finite element models (FEM) and simulations, pointing out clear relationships among cartilage composition, mechanical properties and OA. On the other hand, network-based modelling (NBM) has proven to be useful to simulate semi-quantitatively the biological activity of the chondrocyte, and there is abundant information about the transduction of mechanical signals by chondrocytes, at the molecular and cell scales. However, the knowledge gap between single-cell regulation and intercellular interactions in a representative tissue volume makes it difficult to readily interpolate tissue mechanics and chondrocyte mechano-transduction in OA, especially when the cell biochemical environment needs to be considered. Then, a method to merge both mechanic and biochemical signals is needed, where cell-cell interactions shall intervene. To address such a need, we hereby hypothesize that it is possible to simulate interacting network effects through agent-based modelling, to grasp relevant dynamics of cell and tissue regulation in OA. Whereas chemo- and mechano-regulation network models could be coupled at the cell and subcellular levels, the agents will simulate cell paracrine activity and the emergence of collective chondrocyte behaviour at the tissue level quantitatively, enabling to merge quantitative mechanics of FEM and semi-quantitative biochemistry from NBM. Both literature data and *in vitro* and *ex vivo* measurements will be used to calibrate and duly assess the different model components and the interactions thereof.

Keywords

Network modelling, knee OA, inflammation, articular cartilage, Agent-based modelling

Prologue

During the Biomedical Engineering Bachelor's degree we receive a lot of information in a wide range of topics, going through cell biology, electronics, mechanics, pathophysiology... And, it is at the end of the degree when you realise that all this information can be integrated for a common goal. In fact, during the different projects that I have been developing during the degree, one of the parts that I liked the most it was when we were able to relate information of different subjects.

Hence, it is from these projects that my interest in studying osteoarthritis arose. It was at the beginning of the third course that we developed the first project on this topic, in which we were studying how osteoarthritis could be tackled by means of synthetic biology. Later on, this same year, we also participated in a project in which we were able to reproduce the regulatory network of a chondrocyte. And, in summer, an email with a project proposal for the bachelor's thesis popped up in my inbox: the possibility to link the regulatory network of the chondrocyte to an intercellular model to later on scale up to mechanical forces was in front of me.

Now, I am presenting you the result of having been working in this project for 7 months. This bachelor's thesis allowed me to deepen my knowledge in cellular biology and multiscale modelling, how they can be merged together, and even get some notions on the laboratory work and how complex is the task of incorporating the obtained data to a mathematical model. Indeed, I have worked with a great amount of *in vitro* data to couple a cell level model to an intercellular one in an *in silico* environment. Besides the previous knowledge, I have also been able to get specific information on how osteoarthritis works and which is its current state from the medical point of view, for which, unfortunately, there are still significant shortcomings.

Then, having the opportunity to take part in a project that seeks to improve diagnostic and treatment tools of an expanded disease such as osteoarthritis is really exciting, and I hope that I am able to translate to you the same feeling in this report.

Index

1	Introduction	1
1.1	Physio-pathological background	1
1.1.1	The knee cartilage	1
1.1.2	Osteoarthritis	1
1.1.3	Aetiology of osteoarthritis	1
1.1.4	Diagnosis of osteoarthritis	2
1.1.5	Treatment of osteoarthritis	3
1.2	Objective of the Bachelor’s Thesis	4
1.2.1	Motivation	4
1.2.2	State-Of-the-Art	4
1.2.2.1	Network-based modelling	4
1.2.2.2	Agent-based modelling	5
1.2.3	Aim of the Bachelor’s Thesis	5
2	Methods	7
2.1	Model environment	8
2.1.1	Boundary conditions	8
2.2	Agents in the model	9
2.2.1	Type of agents	9
2.2.2	Mobility of agents (diffusion)	10
2.2.3	Half-life of the agents	11
2.2.4	Interactions between agents (autocrine & paracrine communication)	11
2.2.5	Normalization parameters	14
2.3	Experimental data	16
2.3.1	mRNA expression profile of the knee cartilage	16
2.3.2	Protein concentration at the synovial liquid	17
2.4	Simulation cases	17
2.4.1	Different environmental volumes	17
2.4.2	Initialization with osteoarthritic phenotypes	19
3	Results	20
3.1	Experimental data	20
3.1.1	mRNA expression profile of the knee cartilage	20
3.1.2	Protein concentration at the synovial liquid	20
3.2	Agent-based model results	20
3.2.1	Different environmental volumes	21
3.2.2	Initialization with osteoarthritic phenotypes	21
4	Discussion	23
4.1	Main findings	23
4.2	Limitations & future work	24
5	Conclusions	27

List of Figures

1	Molecules involved in cartilage metabolism and homeostasis	2
2	The role of pro-inflammatory cytokines in the knee joint	3
3	Chondrocyte regulatory network	5
4	Methodology workflow	7
5	2D representation of a chondrocyte in the model	8
6	Boundary conditions of a volume comprising the total thickness of the cartilage	9
7	Healthy baseline of a chondrocyte	14
8	Punches extracted from the left knee cartilage of one patient	16
9	Initialization environments for the three different simulated volumes	18
10	Number of molecules present in the ABM environment	22
11	Baseline & Node expression level at the steady state	22
S1	Linear regression between literature & experimental protein concentrations at the synovial liquid	A5
S2	Third grade polynomic regression between protein concentration per gram of cartilage and protein concentration per milligram of DNA	A6

List of Tables

1	Kellgren & Lawrence grading scale for radiographic OA	3
2	Mean-squared displacement of the soluble mediators	11
3	Half-life of the soluble mediators	12
4	Normalization values for the soluble mediators	15
5	Normalization parameters of the soluble mediators	18
6	Scaling of the model for the three different cases of simulation	18
7	Initialization values for testing the model under patient-specific conditions	19
8	mRNA expression at the knee cartilage	20
9	Mean concentration at the synovial liquid among all the recruited patients	21
S1	Molecular weights (Da) of the soluble mediators	A1
S2	Minimum radius (nm) of the soluble mediators	A2
S3	Parameters used to solve Stokes-Einstein equation	A2
S4	Boolean regulatory network of a chondrocyte	A3
S5	Values for concentration of proteins per gram of cartilage and concentration of proteins per milligram of DNA	A4
S6	Analysed proteins by multiplexing immunofluorescence assays and used methodology	A7
S7	Mean synovial liquid concentrations of soluble mediators	A8

1 Introduction

1.1 Physio-pathological background

1.1.1 The knee cartilage

The knee is a synovial joint, that is, an articulation that joins long bones, as the femur and the tibia, with a fibrous capsule that is continuous with the periosteum. One main characteristic of these joints is that their bones are covered by a highly specialized tissue, the hyaline cartilage, commonly known as articular cartilage [1].

The basic functions of the articular cartilage are to provide a smooth and lubricated surface in articulated joints and to transfer the loads between the bones that form the joint in a controlled way, preventing a disproportionate loading especially under impact. The resistance to axial forces and elasticity required to cover these basic functions are provided by the extracellular matrix (ECM) that maintained by the chondrocytes, i.e., the cells in charge to synthesize the structural proteins of the ECM, mainly collagen type II and proteoglycans. Along with water that stands for about 80% of the tissue volume, the aforementioned components represent about 95% of the articular cartilage. Chondrocytes need the transport of nutrients and waste through the synovial fluid to fulfil its metabolic functions, given that articular cartilage is an avascular tissue. This avascularity also implies a low repair capacity and self-renewal in case of injury or any other disturbance of the normal biosynthetic activity of chondrocytes, as it occurs in diseases such as osteoarthritis (OA) [2].

1.1.2 Osteoarthritis

OA is the most common disease of joints in adults, affecting more than 200 million people globally. Among the different articulations, the knee seems to be the most affected one, being present in about 22% of men and 31% of women over 55 years of age in 2019 according to the Institute for Health Metrics and Evaluation¹. Moreover, its prevalence is expected to increase. Obesity and ageing are important risk factors that exacerbate the development of the disease, and their global burden is projected to increase in the following years [3–6]. Regarding its effects, OA is usually associated with pain and stiffness in the joint, carrying out an increased morbidity and reduced quality of life which can even lead to cardiovascular diseases [7, 8].

1.1.3 Aetiology of osteoarthritis

The aetiology of the knee OA is multi-factorial, as it involves complex and poorly understood interactions among genetic, biochemical, and mechanical factors [9]. Under healthy conditions, the knee cartilage is in charge of compensating for the molecular damage produced by mechanical stresses and enzymatic reactions thanks to the biosynthetic activity of the embedded chondrocytes. In other words, the chondrocyte phenotype is defined by both mechanical loads and biosynthesis of molecules to preserve the tissue integrity and maintain a balance of catabolic and anabolic events [10, 11]. But when a change in the metabolism occurs, towards a catabolic state in the case of OA, there is a mismatch of anabolic and catabolic processes, and the ECM is not properly maintained anymore to

¹Institute for Health Metrics and Evaluation

accomplish its function of weight bearing the synovial joint [12, 13]. Indeed, chondrocyte basal metabolism is fully dysregulated increasing the production of inflammatory mediators and matrix degrading enzymes, which cause ECM degradation (see **Figure 1**) [14].

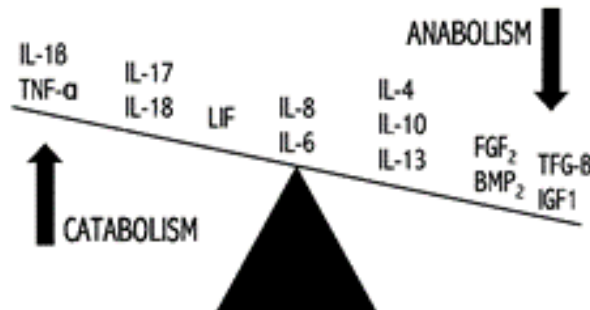


Figure 1: Molecules involved in cartilage metabolism and homeostasis. Source: (Segarra et al., 2019) [15]

Although the articular cartilage of the joint is the main target of the harmful factors promoted by OA, the whole knee and even the systemic level are involved in the change towards a catabolic state (see **Figure 2**) [13]. While for secondary OA the cause of inflammation is usually clear and generally related with exogenous factors, the causes underlying idiopathic OA are less well-known [7]. In fact, inflammatory mediators released by cartilage, bone and synovium have been identified. For example, synovitis is increasingly being associated with the triggering of OA, as it leads to the creation of inflammatory mediators that reach the chondrocytes; induce synovial angiogenesis that increases the production of inflammatory cytokines and degrading enzymes; or even increase the number of synovial macrophages, that might contribute to sustain a catabolic environment. Also, the innate immunity is believed to play a role in the initiation of the disease, being likely involved in synovitis, in interaction with systemic factors such as adipokines, especially in metabolic-related obese patients [16].

1.1.4 Diagnosis of osteoarthritis

OA is commonly defined as “... a heterogeneous group of conditions that leads to joint symptoms and signs which are associated with defective integrity of articular cartilage, in addition to related changes in the underlying bone and at the joint margins” [17]. This means that OA diagnosis is, as its aetiology, polymorphic, and should include different criteria such as patient reported joint pain, radiology findings and proteomics [18, 19].

From the above, medical imaging remains a key tool for the OA diagnosis, although its interpretation is highly dependent on the observer. Then, different grading scales have been proposed to establish a classification scheme of the OA radiographic images, being the Kellgren and Lawrence (KL) classification system the most used [20, 21]. The KL scale is a five-grade classification scheme based on the radiographic features of OA (see **Table 1**) which is used both as a research and as a clinical tool [22].

However, radiology findings alone are not a determinant factor of OA and, moreover, they often appear when the progression of the disease is advanced, especially in X-Ray imaging on which the Kellgren & Lawrence assessment is based [18]. Here, besides other imaging techniques, biomarkers can become a great tool, since inflammatory markers related with OA are being identified and are related with early stages of the disease [23].

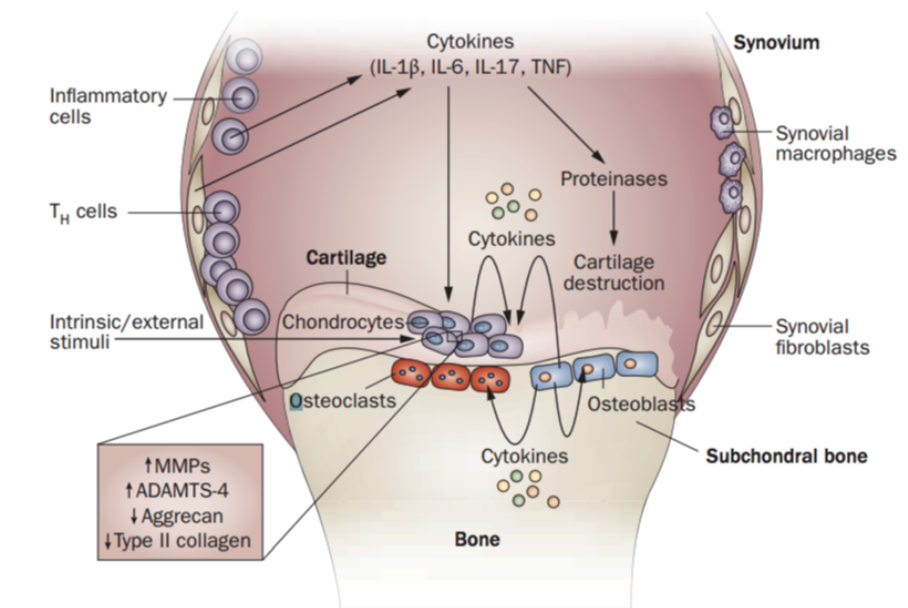


Figure 2: The role of pro-inflammatory cytokines in the knee joint. Pro-inflammatory cytokines are released by several components of the knee joint and up-regulate catabolic events at the cartilage, which debilitate the ECM. Source: Kapoor et al. (2011) [13]

Kellgren & Lawrence grading scale	
Grade	Description
0	No joint space narrowing (JSN) or reactive changes
1	Doubtful JSN, possible osteophytic lipping
2	Definite osteophytes, possible JSN
3	Moderate osteophytes, definite JSN, some sclerosis, possible bone-end deformity
4	Large osteophytes, marked JSN, severe sclerosis, definite bone ends deformity

Table 1: Kellgren & Lawrence grading scale for radiographic OA. Adapted from Kohn et al., 2016 [22].

1.1.5 Treatment of osteoarthritis

Treatments for OA are mainly conservative, aiming to relief pain by losing weight, practicing physiotherapeutic measures or administering drugs. It is in a last stage when the different lines of conservative treatment fail, that total knee arthroplasty (TKA) is proposed as a solution to the patient [7, 8].

Actually, despite the efforts on identifying new targets and drugs for OA, no pharmacological treatments which totally restore the degraded cartilage or decrease the disease progression have been found to date [24]. One of the main challenges in OA drug development is the clinical heterogeneity of the disease and its multifactorial aetiology. Therefore, research is ongoing to define meaningful OA phenotypes that can influence in treatment decisions and identification of biomarkers [25].

1.2 Objective of the Bachelor's Thesis

1.2.1 Motivation

Because of the association with obesity and age, knee OA's prevalence is projected to rise continuously and, combined with the lack of effective treatment strategies, it remains a public health problem of severe importance. It is then evidenced that a better knowledge of the molecular mechanisms involved in its development is needed, so that its diagnosis can be improved and new targets for drug development can be discovered.

1.2.2 State-Of-the-Art

1.2.2.1 Network-based modelling

Many *in vivo* and *in vitro* experiments have been developed to identify the molecular mechanisms involved in OA, but no consensus on its aetiology has been established [9, 12, 13]. Furthermore, the interplay of the different molecules that intervene in the regulation of OA is so complex that making predictions under different conditions escapes from the human abilities. Luckily, computer modelling provides a tool to simulate this intricate interplay from a network perspective, the regulatory network-based models (NBM) [26]. These models are based on the dynamical analysis of network graphs that represent more or less phenomenological maps of knowledge. These maps define interactions among key molecules and the overall regulatory system can be solved mathematically in a semi-quantitative way. Accordingly, static representations of the chondrocyte metabolism could be dynamically analysed through systems of ordinary differential equations (ODEs) or logical statements [27].

Regarding this approach, Segarra et al. (2019) [15] proposed a computational study of the dynamical behaviour of an OA regulatory NBM. The Authors created and optimized a knowledge-based network of the chondrocyte considering the different activations and inhibitions among signals (see **Figure 3**). Comparing the results of Segarra et al. (2019) [15] work with multiplexed proteomics measurements [28] from chondrocyte and cartilage *in vitro* cultures, and with literature-based chondrocyte's behaviours, it was shown that the obtained NBM reproduced well the biological response. Hence, the biochemical signals and interactions proposed could be of great interest for *in silico* investigation of the intricate processes involved in knee OA pathophysiology. Moreover, the regulatory NBM of Segarra et al. (2019) [15] differs from other computational models as it incorporates key soluble and measurable molecules that perturbate the expression of the chondrocytes at the cell level, instead of less easily accessible intracellular transcription factors as proposed by others [26]. Hence, the information from high-level protein model regulations can be directly compared to measurements from chondrocyte cultures, cartilage tissue explants and synovial fluid analyses in patients, making easier the coupling with models of greater scales and patient data.

However, there are still limitations in the application of regulatory networks to study the molecular mechanisms of OA. In fact, these models are sometimes based on experimental results of the literature that may be inconsistent and finally depend on the perspective of the modeller. Furthermore, the results hardly ever are quantitative, which hinders the integration of the cellular models to other quantitative models like Finite Element Models (FEM). That is, the biochemical and biomechanical integration to better define the multi-aetiology of OA is hampered [27].

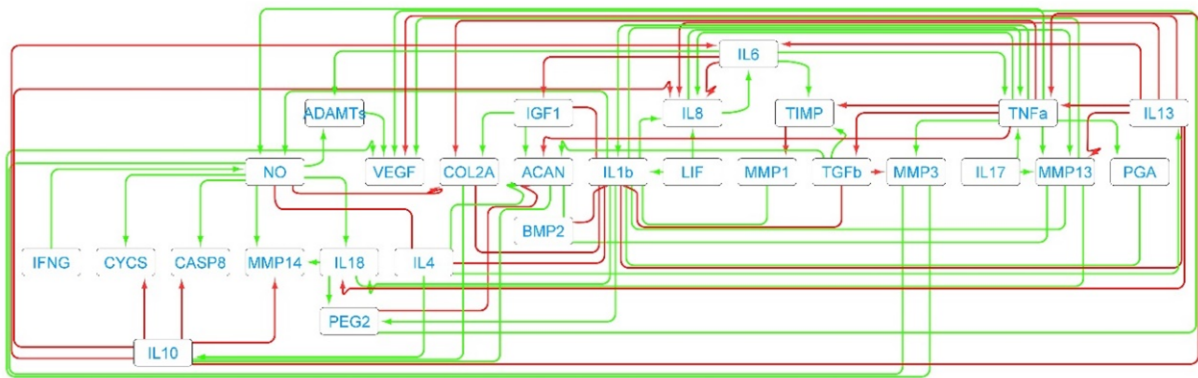


Figure 3: Chondrocyte regulatory network. Directionality of the reactions is indicated by the arrow. Type of reactions are specified by colours (in red inhibiting interactions and in green activating interactions). Image from Segarra et al. (2019) [15]

1.2.2.2 Agent-based modelling

There are several reports on the use of agent-based models (ABMs) to simulate complex systems [29–34]. ABMs are of great interest in the modelling of complex systems that are still not fully understood, as far as there is no optimal behaviour defined that can be used to develop a backward induction of the solution. In other words, ABM is a bottom-up approach based on experimental and deductive data that allows to infer dynamic patterns of behaviour [35].

ABM allows to locate computational objects called agents on space and time and let them interact until a behaviour at the multicellular tissue level is generated. In the area of biomedicine, this becomes helpful to translate the cellular and molecular interactions to specific phenotypes that characterise, for example, a disease [36]. But scaling up from the molecular to the tissue or organ level is not that easy, and often hybrid approaches that describe intracellular networks by means of ODEs and use ABMs for the intercellular level are needed [37].

It is interesting how, during the last years, the use of these hybrid models to link biochemical interactions at the molecular level with biomechanics at the tissue level is flourishing [29, 30, 38, 39]. For example, Baumgartner et al. (2020) [29] propose an ABM to simulate intervertebral disc cell behaviour in a multifactorial environment that indirectly considers biomechanics. However, what is interesting about this work towards the development of a model for knee OA is that the influences of stimulus for mRNA expression were governed by a regulatory NBM, as it happens with the profile of protein expression of a chondrocyte developed by Segarra et al. (2019) [15].

1.2.3 Aim of the Bachelor’s Thesis

Considering that computational models are proving to be really helpful in the comprehension of the molecular mechanisms of OA, but that they are not focused on its integration to multiscale modelling, a new model to understand the multi-aetiology of knee OA is presented. NBM to simulate chondrocyte activity has proven to be a useful tool, although it has only considered single-cell level communication and has given semi-quantitative results [15, 27]. Then, a method to integrate intercellular interactions and give a quantitative solution is needed, and here is where cell-cell interactions shall intervene. Using

this paracrine regulation, it is possible to develop a secondary NBM that it is able to activate and deactivate the nodes of the first model, as it happens in Baumgartner's et al. (2020) [29] work. Indeed, the present work seeks to develop a multiscale ABM which will simulate a volume of the knee cartilage where its agents will be the chondrocytes, the only cell type of this tissue. Cell information is then passed up the spatial scales by diffusion of soluble mediators which will be used for spatial characterization of cartilage inflammation as Shim et al. (2011) [40] did. This way, the model should be able to translate the steady states calculated in a chondrocyte as the expression levels of the molecules that need to be released to the ECM and communicate with other cells. Compared to previous work in chondrocyte NBM and ABM, the present project proposes a unique integration of the dynamic biochemical regulation of single chondrocytes and the effect thereof on collective cell regulation in a representative volume of tissue. Such an approach uniquely allows the further coupling of model inputs and outputs with patient molecular profiles.

All in all, a two-layers model to simulate the cartilage under OA at the multicellular level will be presented and may be helpful to give quantitative results and link them to multiscale modelling of the knee OA. Also, new experimental data regarding mRNA expression at the knee cartilage and protein concentrations at the synovial liquid is reported to test the model.

2 Methods

The open-source Java-based modelling system Repast Symphony version 2.8.0² was used to create the model and run the simulations.

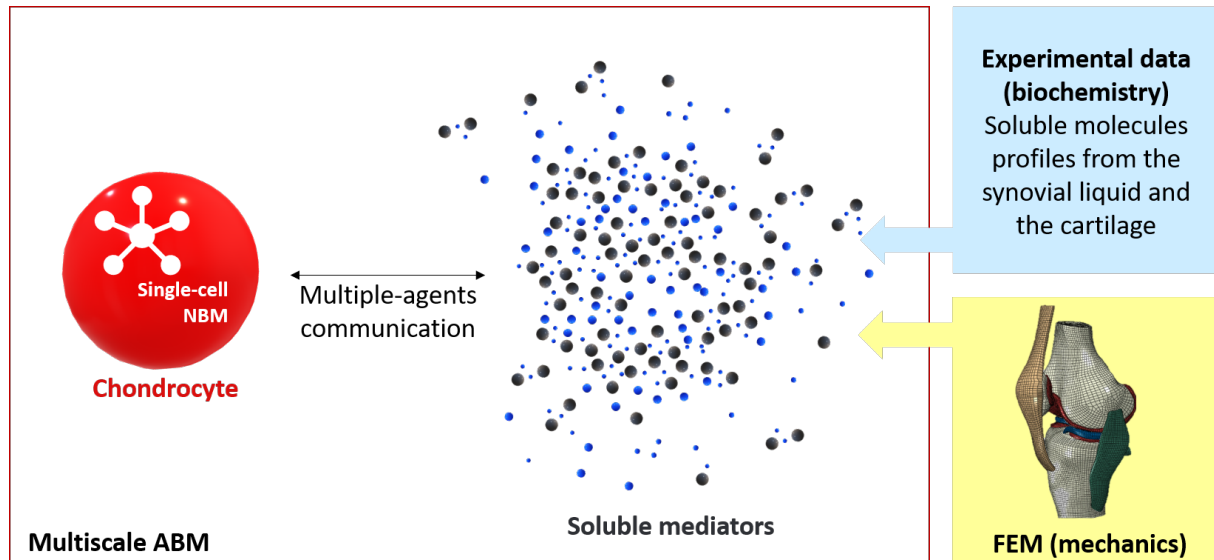


Figure 4: Methodology workflow. An ABM incorporating chondrocytes and soluble mediators was created to represent a region of the knee cartilage. Chondrocytes are characterised by a NBM that defines expression levels of the soluble molecules and that incorporates environmental soluble mediators as an input. Thanks to interactions among soluble mediators in the environment and the chondrocytes, external information such as biomechanics or biochemistry profiles can be easily incorporated in the form of an interacting agent in the model.

As shown in **Figure 4**, a multiscale ABM was created by means of agents representing the chondrocytes and molecules found at the knee cartilage. Each chondrocyte was characterised by a NBM that determined the expression levels of the different molecules and released them accordingly. Molecules were determined to be able to exhibit autocrine and paracrine communication by interacting with the chondrocyte’s membrane.

Besides that, experimental data was collected to validate the model. On the one hand, mRNA expression at the knee cartilage was determined by reverse transcription real-time quantitative polymerase chain reaction (RT-qPCR). On the other hand, protein expression levels at the synovial liquid were quantified by both multiplexed immunofluorescence assays and Enzyme-Linked ImmunoSorbent Assays (ELISAs).

In the following subsections, more details on the creation of the model, the simulation cases and the recollection of experimental data can be found.

²Repast Symphony Suite

2.1 Model environment

Three different cartilage volumes ($100.000 \mu\text{m}^3$, $300.000 \mu\text{m}^3$ and $1.000.000 \mu\text{m}^3$) were represented as a cube of 1 million patches. Depending on the represented volume V_{total} (μm^3), each patch had a volume V_{patch} (μm^3) defined by equation (1) and an edge size e_{patch} (μm) defined by equation (2) (see **Figure 5** for an example of how a chondrocyte is created in the environment depending on the scaled sizes).

$$V_{patch} = \frac{V_{total}}{Numberofpatches} \quad (1)$$

$$e_{patch} = \sqrt[3]{V_{patch}} = \sqrt[3]{\frac{V_{total}}{Numberofpatches}} \quad (2)$$

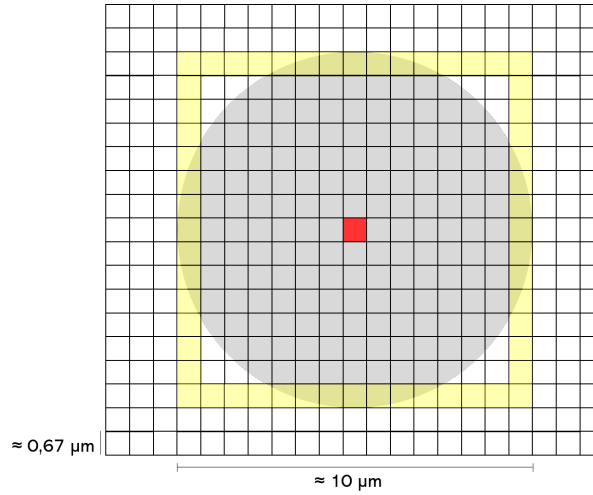


Figure 5: 2D representation of a chondrocyte in the model. In this case, the model is representing a cartilage volume of $300.000 \mu\text{m}^3$ (see **Table 6** for scaling values). As explained in **Section 2.2.4**, chondrocytes in the ABM are represented as cubes of edge length equal to the diameter of the cell. In the figure, the actual chondrocyte is represented by a grey circle. Its representation in the ABM is settled by the red patch, which represents its location by the central point, and the yellow out-layer, which stands for the faces of the cube, representing the cell membrane. To define the patches that the cube should occupy to represent a chondrocyte, the $10 \mu\text{m}$ diameter is translated to the number of patches by the scale (1:0,67 in this case) and the obtained number of patches are settled to be the edges of the volume.

Each time-step of the model, known as a “tick” in Java Repast environment, corresponded to 1 minute in time, since this is the minimum unit to take into account molecules half-life, as it is shown in the following subsections (see **Section 2.2.3**).

2.1.1 Boundary conditions

Considering the three represented volumes, the maximum edge is the one corresponding to the $1.000.000 \mu\text{m}^3$ cube, which is of $100 \mu\text{m}$ edge, while the thickness of the knee cartilage ranges between 1.69 to 2.55 mm [41]. Therefore, continuous boundary conditions were assumed for the whole volume, considering that a central region of the cartilage is simulated. In other words, any agent that exits the model by one of its faces, it will enter by the opposite one keeping its properties.

However, boundary conditions of a volume with edge length equal to the thickness of the knee cartilage were also defined, although not used in the simulations. In this case, in the area in contact with the bone agents would be bouncing, as they are not able to diffuse through bony tissues. Otherwise, in the area in contact with the synovial liquid agents would diffuse out of the cartilage. The four remaining faces would still be represented by continuous boundary conditions, as the longitudinal area of the cartilage is much larger than its thickness (see **Figure 6**).

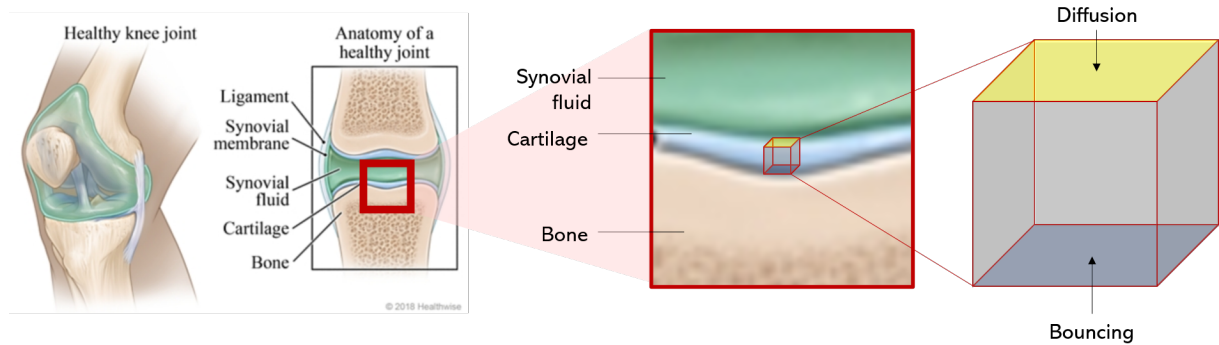


Figure 6: Boundary conditions of a volume comprising the total thickness of the cartilage. The upper face, in contact with the synovial liquid, allows diffusion of the molecules; while the bottom face, in contact with the bone, makes the molecules bounce. The rest of the faces are defined by continuous boundary conditions. Images of the knee anatomy are adapted from [Healthwise staff](#).

2.2 Agents in the model

2.2.1 Type of agents

The ABM consists of two different types of agents: the chondrocytes and the soluble mediators, which differ in its attributes and methods, that is, in its properties and functions. The chondrocytes are static cells with $\approx 10 \mu\text{m}$ of diameter seeded homogeneously, while soluble mediators are mobile proteins with $\approx 0.005 \mu\text{m}$ of diameter secreted by the chondrocytes [2]. The soluble mediators include the cytokines; interferon gamma ($\text{IFN-}\gamma$), interleukin 1-beta ($\text{IL-1}\beta$), interleukin 4 (IL-4), interleukin 6 (IL-6), interleukin 8 (IL-8), interleukin 10 (IL-10), interleukin 13 (IL-13), interleukin 17 (IL-17), interleukin 18 (IL-18), leukaemia inhibitory factor (LIF), tumour necrosis factor alpha ($\text{TNF-}\alpha$); growth factors: bone morphogenic protein 2 (BMP2), insulin growth factor 1 (IGF-1), transforming growth factor beta ($\text{TGF-}\beta$), vascular endothelial growth factor (VEGF); and other molecules: nitric oxide (NO) and prostaglandin E2 (PGE2), from the regulatory network developed by Segarra et al. (2019) [15] able to exit the cell and produce auto- and paracrine communication [42, 43].

Note that, while the size of the chondrocytes has a well established reference, soluble mediators are not under the same condition. Ideally, the size of the proteins is determined by its tertiary structure, but not many cases have its tertiary structure fully defined because of the difficulty of establishing them. Then, soluble mediators were approximated to be spheres of uniform density with a minimum radius R_{min} (m) proportional to its molecular weight³ M (Da=g/mol), as Erickson (2009) [44] work endorses (see equation

³Molecular weight for each protein was found at UniProt. See **Table S1**

(3)). Due to the similarity of the radius for the different soluble mediators (see **Table S2**) and because of the reduced scale that soluble mediators suppose when compared to chondrocytes, its diameter was generalized to be $0.005 \mu\text{m}$ for simplicity of the model.

$$R_{min} = 0.066M^{1/3} \quad (3)$$

2.2.2 Mobility of agents (diffusion)

As mentioned in the previous section, chondrocytes are static agents, as they barely move through the cartilage due to its pericellular matrix that retain and protect them from mechanical loads [45]. However, soluble mediators move randomly through the cartilage and are in charge for the autocrine and paracrine communication. To resemble this random movement to the physiological behaviour, the diffusion of these molecules through the cartilage was defined as the distance that a soluble mediator could move in one time-step.

To determine this diffusion, an analysis at the nanometric and intercellular levels was used. At the nanometric level, the knee cartilage constitution can be considered similar to the layered water [46]. Then, soluble mediators can be approximated as spherical particles in water with low Reynolds number, which is perfect to determine the diffusion coefficient D_{water} by the Stokes-Einstein equation (see equation (4) and **Table S3** for the parameters).

$$D_{water} = \frac{k_B T}{6\pi\eta r} \quad (4)$$

So far, the diffusion at the nanometric level is determined. However, the diffusion at the intercellular level is needed. As cartilage at the tissular level can be considered a porous material with a fluid volume fraction $n_f = 0.8$, the effective diffusivity of the particles could be determined by the empirical relation of Mackie & Meares (1955) [47], that considers the porosity and tortuosity of the media by means of its fluid volume fraction n_f to determine the diffusion coefficient at the cartilage ($D_{cartilage}$) related to the diffusion in water (D_{water}) (see equation (5)) [48].

$$D_{cartilage} = \left(\frac{n_f}{1 - n_f} \right)^2 D_{water} \quad (5)$$

With that, the diffusion coefficients were determined for each soluble mediator and the distance that they were able to travel in one time-step could be determined by the mean-squared displacement (MSD). The MSD considers the dimensions n in which the movement happens, which in this model are 3, the diffusion coefficient $D_{cartilage}$ specific for each molecule and the time the molecule is moving t , which in the model are 60 seconds corresponding to the time that represents each time-step (see equation (6)) (see **Table 2**).

$$MSD = 2nD_{cartilage}t \quad (6)$$

Finally, to implement the random movement in the model, the total distance that each soluble mediator could move in one time-step was split in small size-steps of length equal to the generalized $0.005 \mu\text{m}$ diameter of the soluble mediator. This way, directionality of the soluble mediator was randomly changed at each size-step and the movement of the soluble mediators had a defined distance but in a random trajectory for each time-step.

Mobility of the soluble mediators	
Molecule	Mean-squared displacement (MSD) [μ m]
BMP2	4740
IFN γ	5450
IGF-1	5340
IL-1 β	5040
IL-4	5540
IL-6	5270
IL-8	5980
IL-10	5390
IL-13	5630
IL-17	5540
IL-18	5320
LIF	5330
NO	16000
PGE2	10600
TGF- β	4740
TNF- α	5200
VEGF	5150

Table 2: MSD of the soluble mediators calculated by equation (6).

2.2.3 Half-life of the agents

Chondrocytes in adult cartilage are postmitotic stable cells for which this model does not need to consider its natural half-life, as it will not be reached [49]. However, soluble mediators have a much smaller half-life than mature chondrocytes, so they need the determination of this parameter. Definition of these values was based on an extensive search on the literature (see **Table 3**). To implement the half-life of the soluble mediators, as introduced at the beginning of this section, the minimal value of the soluble mediators' half-life determined the duration that each time-step of the model represents, so that all soluble mediators survive, at least, one time-step.

2.2.4 Interactions between agents (autocrine & paracrine communication)

The presented model exhibits, in fact, two levels of regulation: the cellular and the intercellular. Cell level information is regulated by the regulatory NBM of a chondrocyte developed by Segarra et al. (2019) [15] which, at its turn, is passed to the intercellular level thanks to the cell-cell communication of the chondrocytes, represented by the ABM.

Segarra et al. (2019) [15] NBM has proven to work correctly, and it seems to be a good candidate to represent the behaviour at the single-cell level in an intercellular model, mainly because the proposed regulatory network accounts for interactions at the cell level instead of the intracellular level, which makes easier the coupling with environmental signals. However, a better understanding on how the NBM is constructed needs to be defined so that the ABM can implement it.

Half-life of the soluble mediators	
Molecule	Half-life [minutes]
BMP2 [50] [*]	7
IFN γ [51]	36
IGF-1 [52]	660
IL-1 β [53] ^{*2}	20
IL-4 [54]	19
IL-6 [55]	10
IL-8 [56]	240
IL-10 [57]	52
IL-13 [58]	240
IL-17 [59] ^{*3}	480
IL-18 [60] [*]	1560
LIF [61]	180
NO [62, 63] ^{*4}	70
PGE2 [64]	10
TGF- β [65] ^{*5}	50
TNF- α [66] ^{*6}	76
VEGF [67]	60

Table 3: Half-life of the soluble mediators in minutes.

(^{*} in non-human primates; ^{*2} in rats; ^{*3} estimated from IL-12 subfamily; ^{*4} dependant on the concentration; ^{*5} in human epidermal keratinocytes; ^{*6} estimated from mRNA half-life)

The NBM is based on a Boolean regulatory network representing a chondrocyte-specific protein interactome, which does not take into account intermediate metabolites for sake of simplicity when comparing with cell culture experimental data (see **Figure 3** and **Table S4**). Modelling this network as a dynamical system should drive to clearly identify its stable cellular states but, as for many regulatory networks, the underlying biochemical reactions of the interactions in the network are not known, but only its connectivity and possible directionality (see, for example, Mendoza et al. (1999) [68] or Sánchez & Thieffry (2003) [69]). Luckily, the connectivity of the signalling is sufficient for developing a semi-quantitative model of the regulatory network as proposed by Mendoza & Xenarios (2006) [70]. Briefly, the Mendoza & Xenarios' model proposes an ODE that continuously regulates the expression of each node of the network ($\frac{dx_i}{dt}$) by an activation function that integrates the total input to a node with a given gain and a term of decay (see equation (7)).

To determine the nodal inputs (ω_i), the influence of the activating (x_n^a) and inhibiting (x_m^i) effects is integrated by means of intrinsic activation (α_n) and inhibition (β_m) weights. Concretely, for the chondrocyte's NBM, α_n and β_m were determined to be 0.1 so that the response of the cell to external stimulus is a sigmoid, as it happens with the value of gain h , which was established to be 10 [71].

$$\frac{dx_i}{dt} = \frac{-e^{0.5h} + e^{-h(\omega_i-0.5)}}{(1 - e^{0.5h})(1 + e^{-h(\omega_i-0.5)})} - \gamma_i x_i \quad (7)$$

with,

$$\omega_i = \begin{cases} \left(\frac{1+\sum \alpha_n}{\sum \alpha_n} \right) \left(\frac{\sum \alpha_n x_n^a}{1+\sum \alpha_n x_n^a} \right) \left(1 - \left(\frac{1+\sum \beta_m}{\sum \beta_m} \right) \left(\frac{\sum \beta_m x_m^i}{1+\sum \beta_m x_m^i} \right) \right) & \S \\ \left(\frac{1+\sum \alpha_n}{\sum \alpha_n} \right) \left(\frac{\sum \alpha_n x_n^a}{1+\sum \alpha_n x_n^a} \right) & \S\S \\ \left(1 - \left(\frac{1+\sum \beta_m}{\sum \beta_m} \right) \left(\frac{\sum \beta_m x_m^i}{1+\sum \beta_m x_m^i} \right) \right) & \S\S\S \end{cases}$$

where,

$$0 \leq x_i \leq 1$$

$$0 \leq \omega_i \leq 1$$

$$h, \alpha_n, \beta_m, \gamma_i > 0$$

x_n^a is the set of activators of x_i

x_m^i is the set of inhibitors of x_i

\S is used if x_i has activators and inhibitors

$\S\S$ is used if x_i has only activators

$\S\S\S$ is used if x_i has only inhibitors

Finally, the set of differential equations representing the network (one for each node) are numerically solved by the Runge-Kutta (RK4) integration method, assuming that initial conditions are known. The RK4 method is a fourth-order method based on the well-known routine Euler method. The Euler method is able to solve the differential equations as it follows: assuming that a starting point for the solution is known, the slope of the derivative at this point is calculated. From here, the tangent line is drawn and, taking a small step along this line, the slope is calculated again, and the process is repeated iteratively until the differential equation is solved. Then, the smaller the step-size and the larger the iterations, the more accurate the solution will be. The RK4 only differs from this method in the calculation of the slope of the derivative, which is calculated in four different steps that are finally averaged to give a unique integral solution, making this way the truncation error of the process lower. In the present model, RK4 method was used with a step size of 1 and 775 iterations, which was confirmed to be enough to converge to the steady states. Default initial conditions were set according to the baseline (healthy steady state) determined by Segarra et al. (2019) (see **Figure 7**) [15].

Although expression at the single-cell level can be described by the NBM, collective cell behaviour must be taken into account to scale up to the tissue and organ levels. The ABM is helpful to locate in temporal and spatial scales the chondrocytes, which integrate the single-cell NBM previously described, solving it at each time-step; the soluble mediators released by the chondrocytes, the quantity of which will depend on the expression level that the NBM defines; and its interactions.

To simulate the communication, chondrocytes need to sense its environment. In the ABM, interactions occur at the discretized space, that is, between the patches that conform the cubic volume representing a region of the cartilage. For coding simplification, instead of spherical volumes, chondrocytes are represented as cubes with their edge size

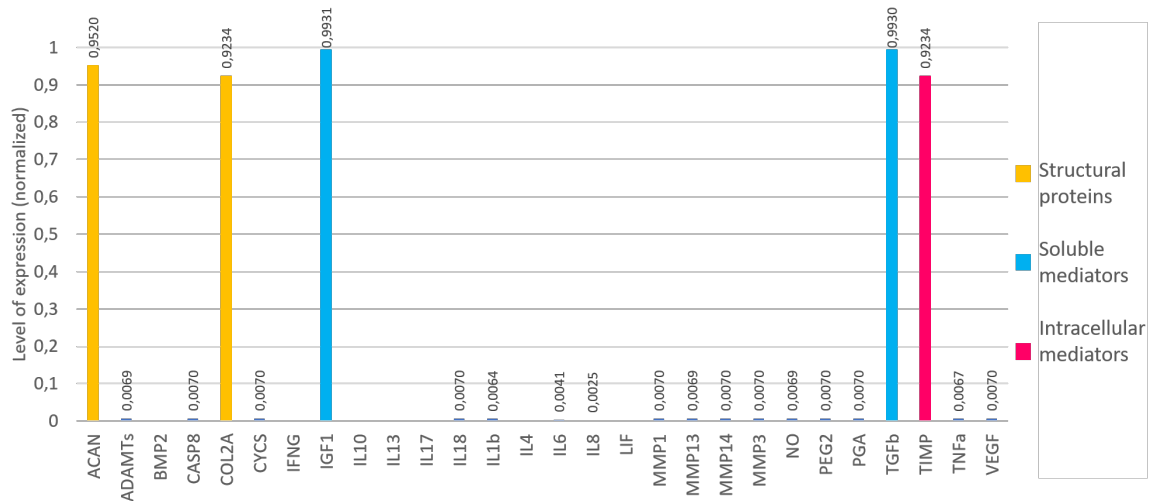


Figure 7: Healthy baseline of a chondrocyte [15].

equal to the diameter of the cell. Since cell interactions occur at the cell membrane, chondrocytes will be able to sense the soluble mediators that are present only at its outer layer (see **Figure 5**). When a patch of the outer layer of a chondrocyte has a soluble mediator inside, the chondrocyte identifies which soluble mediator is in its membrane and communicates with the NBM by clamping to the maximum expression the corresponding node during the calculation of the expression levels of each node, instead of using the default value assigned by the baseline. This way, the phenotype of released molecules may be changed and other nodes may be clamped, entering this way in a feedback between the ABM and the NBM.

2.2.5 Normalization parameters

In order to be consistent with the cell density of the cartilage, which is of 10.000 chondrocytes /mm³, the number of chondrocytes for each simulated volume was calculated. Similarly, the maximum concentration of soluble mediators in the cartilage is needed. Soluble mediators releasing from chondrocytes is not a straightforward process. In fact, there are many checkpoints that regulate, first, gene transcription from DNA to mRNA and, second, its translation into proteins. Covering all of them is out of the scope of this work, but an important one may be the shutdown of expression of soluble mediators from chondrocytes when the ECM is saturated by the end-product of expression. To mimic this behaviour, maximum concentrations of soluble mediators are described, so that releasing of molecules stop when they are reached based on experimental data from Tsuchida et al. (2014) [72] (see **Table 4**).

The maximum number of soluble mediators that a chondrocyte can release should also relate to the experimental data from Tsuchida et al. (2014) [72] and the normalized expression value from the NBM (the result of solving numerically Mendoza et al. (2006) [70] equation (7)). In other words, the values of expression from the NBM, which are normalized as a factor ranging from 0 to 1, need to be translated to the number of released soluble mediators per time-step. For that, Tsuchida et al. (2014) [72] offers data related to the mass of soluble mediators per milligram of DNA in the cartilage, which was considered to be the number of released molecules if node expression was maximum [72]. Then, the data was converted to the number of molecules released per chondrocyte and

it was related proportionally to the expression of each node of the NBM, as equation (8) shows (see **Table 4**). Here, N_i relates with the number of molecules expressed, X_i with the expression level found by solving the Mendoza & Xenarios' equation and N_i^{MAX} with the number of molecules released at maximum node expression (that is, when $X_i = 1$):

$$N_i = X_i \cdot N_i^{MAX} \quad (8)$$

However, experimental data offered by Tsuchida et al. (2014) [72] do not cover all the soluble mediators that are represented in the model. Moreover, due to the high inter-variability of experimental data offered by different experimental sets, inferring the missing values from other sources is not always possible. In fact, only values for the IL-17 and IL-18 could be inferred from the experimental data obtained at the laboratory to validate this model. For that, regression of soluble mediators concentrations at the synovial liquid between Tsuchida et al. (2014) [72] results and lab-work results from the present study was constructed (see **Figure S1**). The concentrations obtained for IL-17 and IL-18 were assumed to be comparable to the maximum concentration at the cartilage by means of the cartilage density. Then, regression between the maximum concentration of molecules at the cartilage and the number of molecules released by a chondrocyte was constructed too (see **Figure S2**). With that, the number of IL-17 and IL-18 released by a chondrocyte for 1 minute were obtained. The rest of soluble mediators, for which no normalization parameters could be obtained, assumed the median of the values that were available (see **Table S5**).

Normalization values for the soluble mediators		
Molecule	Max. number of molecules/cartilage μm^3	Max. number of released molecules/chondrocyte/min
BMP2	0,016930223	10,917946071
IFN γ	0,007737591	0,011217848
IGF-1	0,034651108	0,372429548
IL-1 β	0,000495501	0,033294523
IL-4	0,000075740	0,002868931
IL-6	0,041139416	0,101454205
IL-8	0,895323482	120,95106746
IL-10	0,026216554	1,010723201
IL-13	0,033464440	257,722638467
IL-17	0,001259264	1,618233479
IL-18	0,009027895	0,958202729
LIF	0,106219565	11,059122440
NO	25,218755415	271,050774896
PGE2	2,147601731	23,082388634
TGF- β	0,017068060	0,183447233
TNF- α	0,79276516	0,020841230
VEGF	5,193145477	38,414133570

Table 4: Normalization values for the soluble mediators. Conversion factors for its calculation for the simulations can be seen at supplementary equations (S1) and (S2).

2.3 Experimental data

Experimental data presented in this project was obtained in collaboration with the Inflammation and Cartilage Research Group from the Hospital del Mar Medical Research Institute (IMIM) Centre⁴, under the context of the HOLOA project⁵. The clinical study design included a total of 87 patients with OA, 51 women (27 under conservative treatment and 26 suggested for TKA) and 37 men (23 under conservative treatment and 13 suggested for TKA). Mean age (SD) was 67.71 (4.87) and 67.69 (3.95) in the conservative and TKA group respectively, and mean body mass index (BMI) (SD) was 30.45 (4.43) and 31.93 (6.18), showing no significant differences between them. The evaluation of the knee OA degree by the KL scale exerted by 3 different professionals and averaged did not show significant differences between the two different treatment groups.

Although more factors were analysed in the clinical study, only the mRNA expression profile of the knee cartilage and the protein concentration in the synovial liquid will be presented in this report, since these measurements are the most informative to test and validate the NBM and the coupled ABM.

2.3.1 mRNA expression profile of the knee cartilage

For patients from the TKA group, analysis of the mRNA expression profile at the cartilage was available. To consider how the mechanical forces applied to knee could modify the expression phenotype of the chondrocytes, 11 different punches of 6 mm diameter from each knee cartilage were extracted, paying attention in its location so that it corresponded to zones associated with high or low mechanical loads (see **Figure 8**). These zones were determined accordingly to the FE model results from Adouni et al. (2014) [73], which shows the distribution of forces in the meniscal and tibial cartilage during gait analysis.

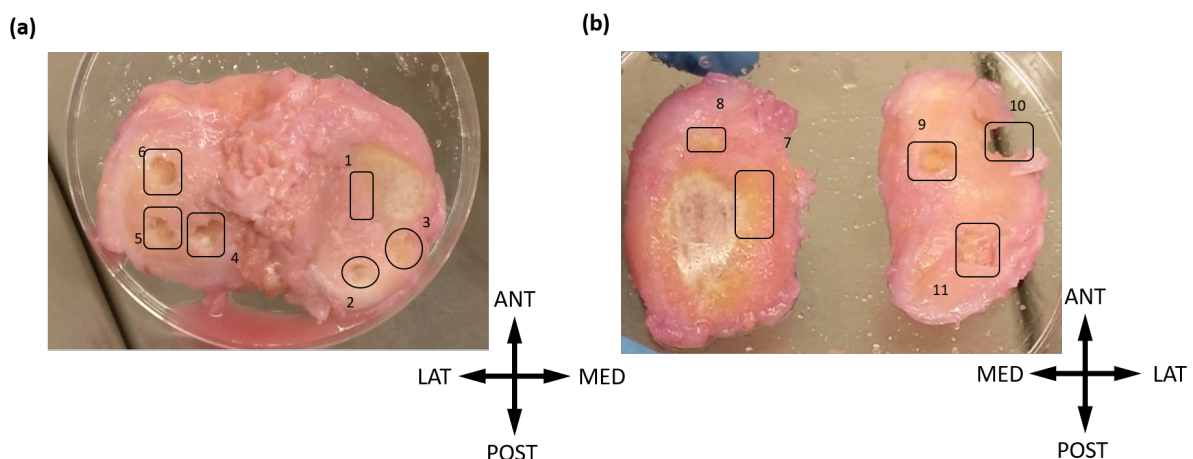


Figure 8: Punches extracted from the left knee cartilage of one patient. (a) Shows the punches extracted from the tibial plateau and (b) The punches extracted from the condyle. For each figure, the antero-posterior (ANT-POST) and medial-lateral (MED-LAT) axis are indicated.

Knee cartilage punches were immediately treated with RNAase later after extraction to prevent mRNA degradation and then cryopreserved at -80°C until total RNA isolation

⁴Inflammation and Cartilage Research Group from IMIM

⁵HOLOA project

was performed. The tissue was manually minced using a mortar and pestle frozen with liquid nitrogen until powder-like texture was achieved. Thereafter, a commercial extraction kit (RNAqueou-Micro Kit, Ambion) was used to isolate total RNA and DNase digestion was performed according to the manufacturer's instructions of the same kit. The resulting total RNA of each sample was used to synthesize cDNA using reverse transcription protocol for SuperScript IV VILO master mix (Invitrogen) with cycling conditions as follows: annealing at 25°C during 10 min, reverse transcription at 50°C during 10 min and finally, enzyme inactivation at 80°C for 5 minutes. Prior to RT-qPCR, preamplification of cDNA targets was carried out according to TaqMan® PreAmp Master Mix instructions using TaqMan® PreAmp Pools (Thermo Fisher Scientific). Finally, diluted preamplification products (1:20) were mixed with TaqMan® OpenArray® Real-Time PCR Master mix and RT-qPCR was performed using QuantStudio™ 12K Flex Real-Time PCR System. Relative expression levels were calculated by the $\Delta\Delta C_T$ method using GADPH, Ppia, RPL4, Tbp and YWHAZ as internal controls, and sample genes of the whole cartilage as reference controls.

Depending on their expression level respect to the reference controls, genes were classified into three different groups: over-expressed, normal-expressed and under-expressed. Moreover, mean values of the $2^{-\Delta\Delta C_T}$ parameter, standing for relative fold gene expression of the samples, were calculated.

2.3.2 Protein concentration at the synovial liquid

Protein concentration levels at the synovial liquid were determined for the 87 patients included in the study. Total synovial effusion volume was extracted, prior to TKA in the intervened patients, and cryopreserved until its analysis. Dilutions to establish the range of concentrations that needed to be considered were performed prior to multiplexing immunofluorescence assays performed with 2 custom Procartaplex (Invitrogen), a Milliplex TIMP Magnetic (Millipore) or performed with ADAMTS-4 and ADAMTs-5 ELISAs kits (FineTest) following manufacturer indications. The list of analysed proteins and used method can be found in **Table S6**. A replicate for each protein was measured in the assays, so mean values between the replicates were calculated, as well as mean of the total samples for each protein.

2.4 Simulation cases

2.4.1 Different environmental volumes

As detailed in **Section 2.1**, three different environmental volumes were simulated: 100.000 μm^3 , 300.000 μm^3 and 1.000.000 μm^3 . Then, parameters previously described depending on the size of the modelled region were calculated, as shown in **Table 5**. **Figure 9** shows the initialization at the 3D space of the 3 models.

Besides scaling the concentrations of agents in the model to the different volumes, sizes also need to be scaled, since the number of patches composing the model are always kept in 1.000.000 cubes. The size scale is given by the previously defined equations (1) and (2). **Table 6** describes the correlation between one patch of the model and one edge of a path in the model and its simulated volume and length, respectively. **Figure 5** shows how a chondrocyte is represented by the patches of the model.

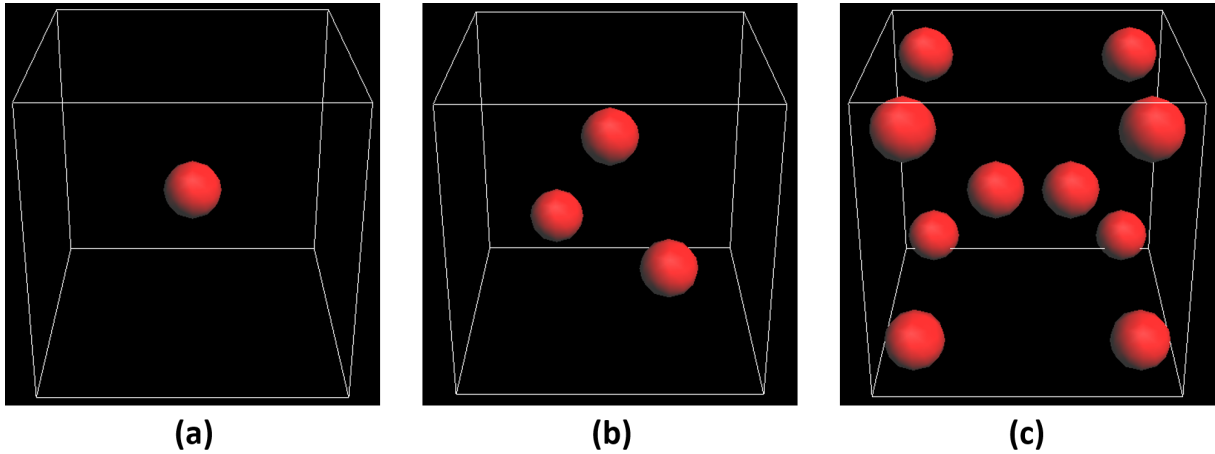


Figure 9: Initialization environments for the three different simulated volumes: (a) 100.000 μm^3 , (b) 300.000 μm^3 , (c) 1.000.000 μm^3

Max. number of soluble mediators			
Simulated volume	100.000 μm^3	300.000 μm^3	1.000.000 μm^3
Number of chondrocytes	1	3	10
BMP2	1.693	5.079	16.930
IFN- γ	774	2.321	7.738
IGF-1	3.465	10.395	34.651
IL-1 β	50	149	496
IL-4	8	23	76
IL-6	4.114	12.342	41.139
IL-8	89.532	268.597	895.323
IL-10	2.622	7.865	26.217
IL-13	3.346	10.039	33.464
IL-17	126	378	1.259
IL-18	903	2.708	9.028
LIF	10.622	31.866	106.220
NO	2.521.876	7.565.627	25.218.755
PGE2	214.760	644.281	2.147.602
TGF- β	1.707	5.120	17.068
TNF- α	7.928	23.783	79.277
VEGF	519.315	1.557.944	5.193.145

Table 5: Normalization parameters of the soluble mediators: number of agents for different cartilage volumes. Conversion for its calculation can be seen at equation (S3).

Scaling of the model			
Environmental volume [μm^3]	100.000	300.000	1.000.000
Volume/patch [μm^3]	0.1	0.3	1
Length/edge path [μm]	0,464158834	0,6694329501	1

Table 6: Scaling of the model for the three different cases of simulation.

2.4.2 Initialization with osteoarthritic phenotypes

Moreover, to test the model, four different cases derived from clinical data were proposed. Synovial liquid protein levels were subdivided in 4 groups: KL=2 under conservative treatment, KL=3 under conservative treatment, KL=2 under TKA and KL=3 TKA. Mean values of the concentrations were obtained and are presented in **Table S7**. Therefore, four different simulations were run, one for each group, on a cartilage volume of $300.000 \mu\text{m}^3$, initialized with three chondrocytes and the initial protein concentrations translated to the corresponding number of molecules in the volume (see **Table 7**).

Initial number of soluble mediators to simulate patient-specific conditions				
Molecule	KL=2 & Conservative	KL=3 & Conservative	KL=2 & TKA	KL=3 & TKA
IFN γ	129	176	148	197
IL-1 β	56	56	56	56
IL-4	166	177	277	177
IL-6	1.064	1.623	789	2.933
IL-8	161	453	810	1.468
IL-10	48	39	38	49
IL-13	137	137	137	137
IL-17	288	342	136	212
IL-18	716	1.523	1.555	2.002
TNF- α	706	982	864	1.069
VEGF	7.387	16.595	21.291	16.255

Table 7: Initialization values for testing the model under patient-specific conditions. Conversion factors for its calculation can be seen in equations (S2) and (S3).

3 Results

3.1 Experimental data

3.1.1 mRNA expression profile of the knee cartilage

Although 56 genes were analysed, only data for those present in the NBM is shown (see **Table 8**). From the 16 presented genes, 7 were classified as under-expressed and 7 as over-expressed in comparison to the intrinsic controls. Moreover, mean expression of each gene in terms of the $2^{-\Delta\Delta C_t}$ value are presented with respect to the reference controls.

mRNA expression at the knee cartilage		
Gene	Level of expression	Mean $2^{-\Delta\Delta C_t}$
ACAN	Under-expressed	2,679291783
ADAMTs-4	Over-expressed	0,723075162
ADAMTs-5	Normal-expressed	2,898388793
BMP-2	Normal-expressed	34,94801298
COL-2 α 1	Under-expressed	24,92923533
IGF- β	Over-expressed	2,476735956
IL-1 β	Over-expressed	0,126100789
IL-6	Over-expressed	0,36742785
MMP-1	Over-expressed	778,7424531
MMP-13	Over-expressed	1,681927178
MMP-3	Under-expressed	11,43694783
TGF- β	Under-expressed	17,50251118
TIMP-2	Under-expressed	4,172397134
TIMP-3	Under-expressed	3,936279266
TNF- α	Over-expressed	0,262645614
VEGF	Under-expressed	3,273484135

Table 8: mRNA expression at the knee cartilage. Only genes found in the chondrocyte regulatory network are presented (see **Figure 3**).

3.1.2 Protein concentration at the synovial liquid

As it happens for the mRNA, data for 29 different proteins was collected, but only the ones present in the NBM are presented.

Table 9 shows the mean values among all the patients for the concentration of proteins at the synovial liquid.

Table 7 shows also mean values, but in this case, patients are subdivided according to their level of radiologic OA and prescribed treatment. Then, mean values are shown for patients with KL = 2 and under conservative treatment, KL = 3 and under conservative treatment, KL = 2 and under TKA, and KL = 3 and under TKA.

3.2 Agent-based model results

During the simulations, the ABM could display the number of the different soluble mediators present in the environment, as well as its 2D and 3D location in a spatiotemporal

Mean concentration at the synovial liquid among all patients	
Protein	Mean concentration [pg/ μ L]
IFN- γ	28,16202904
IL-1 β	14,27
IL-4	27,12550123
IL-6	387,5767973
IL-8	77,78461569
IL-10	7,641840324
IL-13	17,97
IL-17 α	38,71117281
IL-18	298,4629449
MMP-1	12.225,00159
MMP-3	3.382,844153
MMP-13	49,25687716
TNF- α	206,4087705
VEGF- α	3.585,937254

Table 9: Mean concentration at the synovial liquid among all the recruited patients. Only proteins found in the chondrocyte regulatory network are presented (see **Figure 3**).

domain for each time-step. Text and CSV files were created during the simulations offering information about node expression of the NBM and number of soluble molecules at the ABM for each time-step.

3.2.1 Different environmental volumes

For the three different represented volumes, calculations were performed for more than 7 hours to achieve more than 100 time-steps in all cases, running with a computer with 12GB RAM, Intel(R) Core™ i7-7500U CPU @ 2.70GHz (dual core). The evolution of the number of soluble mediators present in the environment for the 3 simulated cases is shown in **Figure 10**. Although in the three cases the released agents are the same, differences in the number of total agents are found. Concretely, the larger the volume represented, the greater the number of chondrocytes and the greater the number of total molecules released during the same time.

Figure 11 shows the level of activation of each node of the NBM against the baseline by which the simulations were initialized. Steady states converge to the baseline itself and are the same in the three simulations, for all the chondrocytes and for each time-step.

3.2.2 Initialization with osteoarthritic phenotypes

Although code and initial conditions were already set up for these simulations, results cannot be shown because of the large computational time they supposed. In this case, for the four different initial conditions, calculations lasted more than 4 hours to advance only 1 time-step in the same computer as before (12GB RAM, Intel(R) Core™ i7-7500U CPU @ 2.70GHz (dual core)).

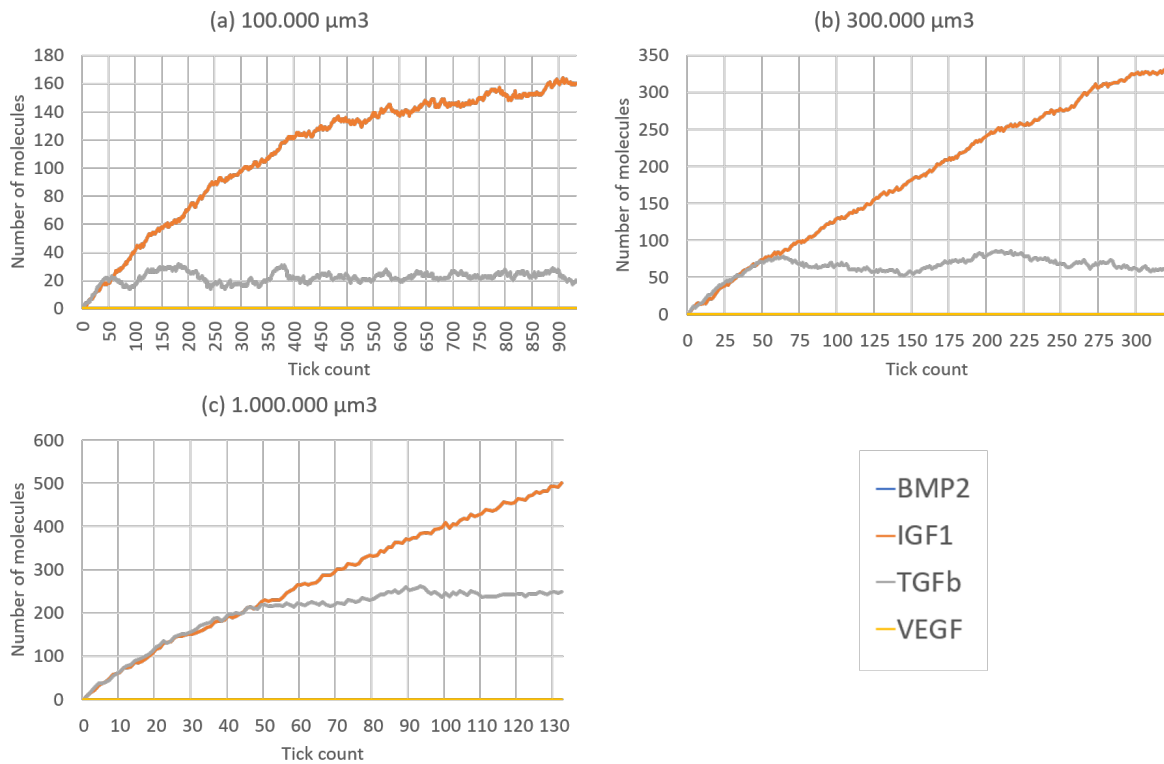


Figure 10: Number of molecules present in the ABM environment for volumes of (a) 100.000 μm^3 , (b) 300.00 μm^3 and (c) 1.000.000 μm^3 . Simulations run for 7 hours, reaching different number of time-steps. Only growth factors are shown, as other anti-inflammatory and pro-inflammatory mediators were not present in the environment (not released).

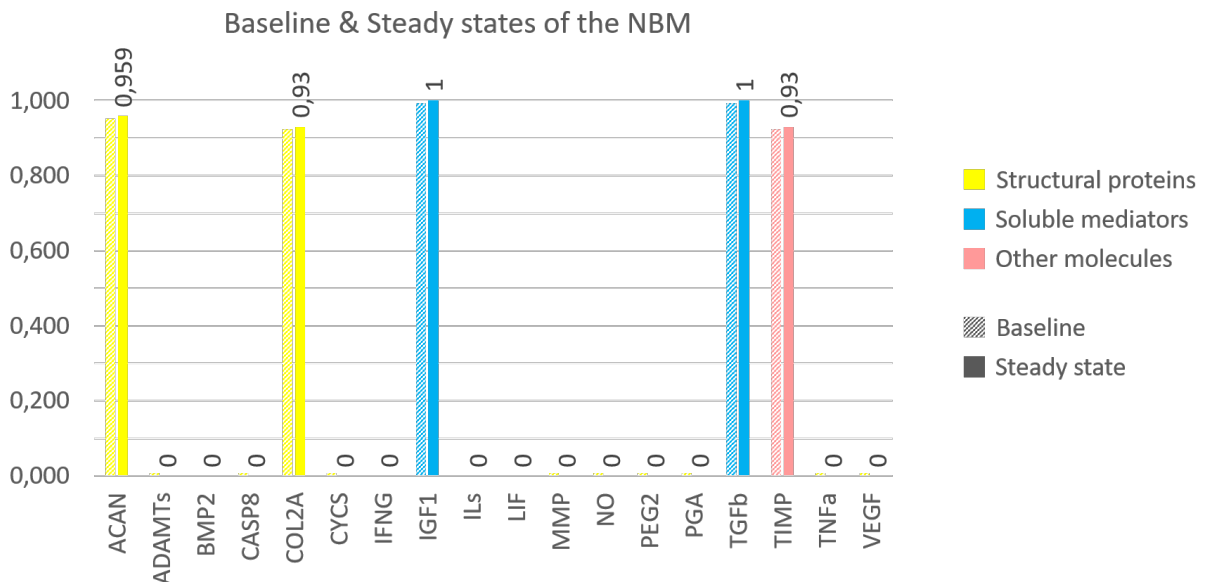


Figure 11: Steady state of the nodal expression achieved in the 3 different simulations (full color bars) compared with the baseline by which simulations were initialized (dashed lines bars). Interleukins (ILs) and metalloproteinases (MMP) are averaged to a single node, as they all had values near zero.

4 Discussion

4.1 Main findings

Results of the ABM show correlation with the results of the single-cell NBM. The baseline defined as the initial conditions to solve the NBM is kept throughout the whole simulation in the three volumes represented (see **Figure 7**). This is not surprising since the baseline corresponds to the healthy steady state of a chondrocyte. In fact, the only nodes corresponding to soluble mediators considered to be active are the TGF- β and IGF-1 (node expression ≥ 0.01). Then, the state of the chondrocytes cannot be perturbed, as long as the nodes that could be clamped by the released soluble mediators are already at its maximum expression at the stable state (see **Figure 11**). Hence, the only regulations that can be appreciated in the model are the number of chondrocytes, as the higher their number the greater the protein concentration; and the cytokine half-life, which effect is especially clear for IGF-1 in the 100.000 μm^3 volume: as early as 50 time-steps are reached (IGF-1's half-life), its concentration is stabilized because, for each new molecule released in a time-step, an old one dies (see **Figure 10**). However, some fluctuations can still be appreciated due to the assumption that, when a soluble mediator enters a cell, it dies.

Although only results for a healthy baseline are presented, the model is able to also integrate pro-inflammatory inputs as initial conditions. With that, a completely catabolic response is obtained, and the model seems to do nothing to counteract it. However, due to the higher expression per time-step of catabolic soluble mediators, the number of agents rapidly increase, and the simulation turns to advance really slow, and that is why data is not shown, as it happens in the simulations with OA phenotypes (see **Section 3.2.2**). In any case, the lack of a balancing response can be justified by looking at the interactions in the regulatory network of the NBM (see **Figure 3** ad **Table S4**). The only anti-inflammatory nodes that could exit the cell and counteract the catabolic state are IL-4, IL-10 and IL-13. However, IL-10 and IL-13 will only be activated if IL-4 is expressed, and the latter one has no biochemical activators nor inhibitors, so a purely biochemical network cannot activate its expression. Despite of that, it is known that IL-4 is produced by chondrocytes when its mechanotransduction pathways are considered [74]. Therefore, to easily prove that IL-4 could play a role in balancing the chondrocyte metabolism, the biochemical phenotype characterised by the osmophysiological properties of the cartilage could be integrated in the ABM, probably leading to the release of IL-4 which would exhibit its anti-inflammatory autocrine and paracrine response.

Based on the state-of-the-art research, despite the increasing number of works using ABM for multi-scaling, the present work appears to be the first approach in using the combination of NBM and ABM to represent the intercellular communication at the knee cartilage at both the single-cell and multicellular levels, to later scale up to larger levels. In fact, the only study found with the same aim is the one of Shim et al. (2011) [40], which integrates organ, tissue and cell levels to explain a possible mechanism of initiation of OA. However, Shim et al. (2011) [40] work only considers mechanical inputs to the three levels of the model. Therefore, given the current directories on the multi-aetiology of OA, and the great importance of pro-inflammatory mediators as triggering factors of the disease, a model able to integrate both biochemical and mechanical information seems crucial [9, 12, 13, 16].

For that, although the presented methodology is focused on the biochemical inter-

actions at the knee cartilage, it should not be difficult to integrate the homogenized mechanics present at the tissue level thanks to the ABM at the multicellular level. On the one hand, chondrocytes could easily integrate mechanoreceptors as new nodes that regulate the chondrocyte's phenotype, and even incorporate transcription factors that refine the behaviour of the model [75]. On the other hand, having a representation of a tissue volume allows to determine the concentrations of structural proteins and determine mechanic parameters of the tissue [40].

Moreover, synovitis is pointing out as an important factor in the development of idiopathic OA [16]. Since the interactions in the present model are based at the cell level, and not at the intracellular level as other studies of the biochemical regulation of OA, the incorporation of external signals as the ones prominent from the synovium diffusion are readily incorporated.

Regarding the experimental results, from the overall mRNA expression, it is interesting to point out the over-expression of catabolic and pro-inflammatory mediators and the under-expression of anabolic mediators and structural proteins, which could be expected since only patients proposed for TKA, that is, with an advanced stage of OA, could be analysed, as taking samples of the cartilage can only be done *ex vivo*. Then, the deviation of the results from mRNA expression towards patients in need for TKA and the lack of data for healthy controls highlights again the need of *in silico* models to study the multi-aetiology of OA.

A question that may arise is why studying mRNA expression instead of directly protein concentrations at the cartilage, as these results could better fit in the model. That is because, firstly, mRNA identification is easier and cheaper than protein identification, as it is present in larger concentrations and it is only encoded by 4 bases, instead of the 20 aminoacids that encode proteins. But secondly, and more importantly, gene expression is a highly regulated process at different levels, and post-transcriptional mRNA is the first gate of this regulation, so it is a great marker to know if special findings such as low concentrations come from the gene expression or if other processes such as protein degradation need to be considered.

4.2 Limitations & future work

The presented work is not only complex because it seeks to model a disease with multi-aetiology at different levels, but also only at the single-cell level interactions are complex and involve a lot of regulatory steps. Therefore, several assumptions and simplifications were stated during the development of the bachelor's thesis. In this section, different simplifications and problems that came up while advancing with the project are presented and focused to the refinement of a better model for future work.

In a first instance, the size of the simulated cartilage volumes needed to be defined. Initially, a volume of one million μm^3 was defined so that 10 chondrocytes could be represented in the ABM. However, the huge computational times required to solve each time-step of the simulation led to compare the results with a model of three chondrocytes, so that less agents were created, and computational time was reduced. Despite that, results regarding the chondrocyte's phenotype under a healthy baseline do not differ among the different volumes represented, although other simulation cases with different initial conditions should be considered to assess the dependency on the number of chondrocytes represented (see **Figure 10**).

Moreover, defining a cube of one million μm^3 as the portion of the cartilage, leads to a thickness of 0.1 mm, which is not enough to consider the boundary conditions exerted by the synovium and calcified zone, that is, the layers that compress the cartilage (see **Figure 6**). Therefore, bigger volumes need to be simulated to study the effect of these boundary conditions.

On top of that, the environmental volume was discretized into one million patches in which interactions of the ABM took place. However, the real size of the soluble mediators ($\approx 0.005 \mu\text{m}$ diameter) could not be considered during interactions, as working in the discretized space limits the minimum size of an agent to the volume of a patch. From the cases simulated, the edges of the patches represented the shortest length in the discretized volume of $100.000 \mu\text{m}^3$ ($\approx 0.464 \mu\text{m}$ edge length), which still was two orders of magnitude greater than it should to represent a single soluble mediator. This could have non-desired influences in the probability of interacting with a chondrocyte. From here, one may think that the solution is as simple as discretizing the volume into more patches of less volume. But the problem comes due to computational power limitations, as no more than one million patches could be used given the memory of the computer. Another possible solution could be the representation of soluble mediators into clusters, although this is not straightforward. The defined maximum concentrations of soluble mediators were lower than the number of molecules that should conform a cluster in a patch. So, if these maximums want to be kept, clustering is not an applicable option either.

From the points explained above, it can be drawn that computational power is a limiting factor in the modelling approach. This is clearly identified by the times that the different simulations lasted, being of hours to only calculate some hundreds of time-steps in the comparison of environmental volumes, and not even calculating one time-step in 4 hours in the simulations with initial OA environments. This is probably because of the computational workload that supposes creating an agent in the world and making it move for each time-step, as simulations show an increase in the duration of a time-step while more agents are present in the environment. Furthermore, significant results regarding the phenotype of the disease should appear at least in months, due to the commonly slow progression of OA. Then, it is evidenced that alternatives for the model to calculate more time-steps in shorter times are needed, from which two are further developed in the following paragraph as future work for the model enhancement.

On the one hand, the open-source toolkit Repast Symphony 2.8.0 was used because of its widespread and capacity for large-scale agent modelling, as well as its Java-based modelling system and libraries, which allows easily creating ABMs in a fully personalized program by the user [33]. Interestingly, the toolkit has its equivalent in a C++-based modelling system designed for use on high-performance computing (Repast for HPC⁶). Translating the model into this new modelling environment should be really helpful in the sense that the behaviour of the agents could be parallelized, largely reducing the computational time a time-step of the model lasts. In fact, the process of translating the code into this new environment was started and it is currently being carried out. However, such an extensive task requires of a non-available huge amount of time because of the timings for the presentation of this report and the modelling and experimental workload. On the other hand, a homogenized response can be defined after the computation of, for example, 60 time-steps, which would be the equivalent to 1 hour. Then, this homogenized

⁶Repast Suite Documentation - HPC

response can be used to rescale the temporal domain of the model, calculating the evolution per hour instead of per minute and reducing this way the time to reach simulations representing larger periods.

Aside from computational power issues, other factors need to be considered. For example, in the ABM, only chondrocytes and soluble mediators were represented as agents, since the model was aimed at testing the effect of cell-cell communication. Nevertheless, the inclusion of structural proteins as agents may be of great interest for future purposes in which mechanics of the tissue need to be considered, so that their concentrations and distribution can be used to determine mechanical parameters of the ECM. However, due to the intricate distribution of structural proteins in the cartilage, their inclusion may not be straightforward as for the soluble mediators. Moreover, chondrocytes were set in the environment in a homogenized way, although altered distribution of the chondrocytes among the cartilage is another factor characteristic of OA states. Therefore, a study under different locations of the chondrocytes could determine relations between the state of OA and chondrocyte distribution in the cartilage.

Regarding the parameters of the agents, half-life of the soluble mediators was considered, but not for the chondrocytes. Chondrocyte half-life does not need to be considered because of its large lifetime, but apoptotic behaviour under the stresses produced by the pro-inflammatory state of the cartilage may occur, although here it was not considered because the lack of apoptotic soluble mediators [76–78]. Hence, incorporation of intracellular transcription factors could help defining the apoptotic state of the cells which probably would carry several effects at later stages of the disease, as one of its characteristics is the lack of chondrocytes at the cartilage.

To establish the diffusion of soluble mediators through the cartilage, Einstein diffusion was assumed to be applicable at the ECM to calculate the diffusion coefficient. This means that multiple-particle occupancy and uncorrelated random walks are considered. Despite that, the high values obtained for the diffusion coefficients and, in consequence, the large distance that soluble mediators diffuse in a minute –in the order of thousands of μm^3 – indicate that these assumptions may be incorrect. While there is nothing that appears to be wrong in considering uncorrelated random walks; multiple-particle occupancy may be a source of errors. Depending on the molecular charge of the molecules, short-range repulsive interactions may occur between adjacent molecules, so multiple-particle occupancy cannot be considered. Then, occupancy in terms of diffusivity and mobility of the particles should be taken into account to calculate the diffusion coefficient [79].

Finally, the values that characterise the number of released soluble mediators in the model need to be commented. The NBM that characterises each chondrocyte is based on a semi-quantitative dynamical study of its regulatory network. As far as there are no rate constants for the expression of each node that allow to feed the Mendoza & Xenarios' (2006) [68] equations up to the quantitative level, weaker formulations like the number of molecules released per cell per minute from experimental data need to be considered, so that the model can be scaled up to higher levels by means of the quantitative results [27, 70]. Moreover, besides not having values for all the soluble mediators and approximating them by regression or the median –something that should be corrected–, the data was extracted from *in vitro* experiments. Although these procedures are useful to get rid of possible influences from the systemic level, the chondrocytes are submitted a series of stresses that lead to a change in their phenotype, so inter-variability between experiments and, of course, between physiological values appear [72].

5 Conclusions

In this final bachelor's thesis, a two-layer model is presented to better define the multi-aetiology of OA. A dynamical system of the biochemical regulation of OA at the single-cell level constitutes a NBM that defines the behaviour of a chondrocyte. Then, to scale-up to the multi-cellular level, an ABM is proposed, so that cell-cell communication is performed by the soluble mediators that chondrocytes release.

Therefore, this model offers a first step towards the multiscale modelling of knee OA, showing a link between biochemical and biomechanical signals thanks to the representation of a specific volume of the cartilage and to the location of the agents in space and time.

The use of experimental data is clearly necessary to test and validate the behaviour of the model, and also to better define its parameters and assumptions. For that, mRNA expression at the cartilage and protein concentrations at the synovial liquid were analysed, but its incorporation to the model could not be fulfilled because of the lack of computational power.

In fact, there is some evidence that the model could better define catabolic states of the cartilage, in which cell-cell communication may play a higher role according to the interaction of the NBM. In any case, more simulations with a larger number of time-steps are needed and, for that, it is evident that the model requires of a higher computational power to achieve significant results. Then, future steps in the development of the model should focus on translating it to an HPC environment, for which a Repast option is already available. With that, significant results are expected to be achieved, in larger time-scales which account from months to years. Moreover, once the model's functionality is confirmed, its space and time definitions make it an interesting starting point to relate biochemical and biomechanical effects at the cellular level with mechanical forces at the organ level, offering a new integrated point of view to reveal the multi-aetiology of OA.

Bibliography

- [1] Alexander Y. Hui et al. “A systems biology approach to synovial joint lubrication in health, injury, and disease”. In: *Wiley interdisciplinary reviews. Systems biology and medicine* 4.1 (2012), pp. 15–37. DOI: [10.1002/wsbm.157](https://doi.org/10.1002/wsbm.157).
- [2] Mehmet Selcuk Şenol and Hamza Özer. “Chapter 7 - Architecture of cartilage tissue and its adaptation to pathological conditions”. In: *Comparative Kinesiology of the Human Body*. Ed. by Salih Angin and Ibrahim Engin Şimşek. Academic Press, 2020, pp. 91–100. ISBN: 978-0-12-812162-7. DOI: [10.1016/B978-0-12-812162-7.00007-2](https://doi.org/10.1016/B978-0-12-812162-7.00007-2).
- [3] A. Cui et al. “Global, regional prevalence, incidence and risk factors of knee osteoarthritis in population-based studies”. In: *EClinicalMedicine* 29-30.100587 (2020). DOI: [10.1016/j.eclim.2020.100587](https://doi.org/10.1016/j.eclim.2020.100587).
- [4] A. Turkiewicz et al. “Current and future impact of osteoarthritis on health care: a population-based study with projections to year 2032”. In: *Osteoarthritis and Cartilage* 22.11 (2014), pp. 1826–1832. DOI: [10.1016/j.joca.2014.07.015](https://doi.org/10.1016/j.joca.2014.07.015).
- [5] M. Blagojevic et al. “Risk factors for onset of osteoarthritis of the knee in older adults: a systematic review and meta-analysis”. In: *Osteoarthritis and cartilage* 18.1 (2010), pp. 22–43. DOI: [10.1016/j.joca.2009.08.010](https://doi.org/10.1016/j.joca.2009.08.010).
- [6] T. Kelly et al. “Global burden of obesity in 2005 and projections to 2030”. In: *International journal of obesity* 32.9 (2008), pp. 1431–1437. DOI: [10.1038/ijo.2008.102](https://doi.org/10.1038/ijo.2008.102).
- [7] J. W. Michael, K. U. Schlüter-Brust, and P. Eysel. “The epidemiology, etiology, diagnosis, and treatment of osteoarthritis of the knee”. In: *Deutsches Arzteblatt international* 107.9 (2010), pp. 152–162. DOI: [10.3238/arztebl.2010.0152](https://doi.org/10.3238/arztebl.2010.0152).
- [8] A. E. Nelson. “Osteoarthritis year in review 2017: clinical”. In: *Osteoarthritis and cartilage* 26.3 (2018), pp. 319–325. DOI: [10.1016/j.joca.2017.11.014](https://doi.org/10.1016/j.joca.2017.11.014).
- [9] M. B. Mueller and R. S. Tuan. “Anabolic/Catabolic balance in pathogenesis of osteoarthritis: identifying molecular targets”. In: *PM R : the journal of injury, function, and rehabilitation* 3.6 Suppl 1 (2011), S3–S11. DOI: [10.1016/j.pmrj.2011.05.009](https://doi.org/10.1016/j.pmrj.2011.05.009).
- [10] Farshid Guilak. “Biomechanical factors in osteoarthritis”. In: *Best Practice Research Clinical Rheumatology* 25.6 (2011). Musculoskeletal Science, pp. 815–823. ISSN: 1521-6942. DOI: [10.1016/j.berh.2011.11.013](https://doi.org/10.1016/j.berh.2011.11.013).
- [11] R. Ruhlen and K. Marberry. “The chondrocyte primary cilium”. In: *Osteoarthritis and cartilage* 22.8 (2014), pp. 1071–1076. DOI: [10.1016/j.joca.2014.05.011](https://doi.org/10.1016/j.joca.2014.05.011).
- [12] D. Chen et al. “Osteoarthritis: toward a comprehensive understanding of pathological mechanism”. In: *Bone research* 5.16044 (2017). DOI: [10.1038/boneres.2016.44](https://doi.org/10.1038/boneres.2016.44).
- [13] M. Kapoor et al. “Role of proinflammatory cytokines in the pathophysiology of osteoarthritis”. In: *Nat Rev Rheumatol* 7 (2011), pp. 33–42. DOI: [10.1038/nrrheum.2010.196](https://doi.org/10.1038/nrrheum.2010.196).
- [14] T. G. Benedek. “A history of the understanding of cartilage”. In: *Osteoarthritis and Cartilage* 14.3 (2006), pp. 203–209. ISSN: 1063-4584. DOI: [10.1016/j.joca.2005.08.014](https://doi.org/10.1016/j.joca.2005.08.014).

- [15] M. Segarra, M. Neidlin, and J. Noailly. “Osteoarthritis regulatory network. A computational study about its dynamical behaviour”. In: *Master’s Thesis* (2019). Not published.
- [16] F. Berenbaum. “Osteoarthritis as an inflammatory disease (osteoarthritis is not osteoarthrosis!)” In: *Osteoarthritis and Cartilage* 21.1 (2013), pp. 16–21. ISSN: 1063-4584. DOI: [10.1016/j.joca.2012.11.012](https://doi.org/10.1016/j.joca.2012.11.012).
- [17] R. Altman et al. “Development of criteria for the classification and reporting of osteoarthritis. Classification of osteoarthritis of the knee. Diagnostic and Therapeutic Criteria Committee of the American Rheumatism Association”. In: *Arthritis and rheumatism* 29.8 (1986), pp. 1039–1049. DOI: [10.1002/art.1780290816](https://doi.org/10.1002/art.1780290816).
- [18] M.G. Lequesne and E. Maheu. “Clinical and radiological evaluation of hip, knee and hand osteoarthritis”. In: *Aging Clin Exp Res* 15 (2003), pp. 380–390. DOI: [10.1007/BF03327359](https://doi.org/10.1007/BF03327359).
- [19] H. N. Daghestani and V. B. Kraus. “Inflammatory biomarkers in osteoarthritis”. In: *Osteoarthritis and cartilage* 23.11 (2015), pp. 1890–1896. DOI: [10.1016/j.joca.2015.02.009](https://doi.org/10.1016/j.joca.2015.02.009).
- [20] J. H. Kellgren and J. S. Lawrence. “Radiological Assessment of Osteo-Arthrosis”. In: *Annals of the Rheumatic Diseases* 16.4 (1957), pp. 494–502. ISSN: 0003-4967. DOI: [10.1136/ard.16.4.494](https://doi.org/10.1136/ard.16.4.494).
- [21] H. J. Braun and G. E. Gold. “Diagnosis of osteoarthritis: Imaging”. In: *Bone* 51.2 (2012). Osteoarthritis, pp. 278–288. ISSN: 8756-3282. DOI: [10.1016/j.bone.2011.11.019](https://doi.org/10.1016/j.bone.2011.11.019).
- [22] M. D. Kohn, A. A. Sassoon, and N. D. Fernando. “Classifications in Brief: Kellgren-Lawrence Classification of Osteoarthritis”. In: *Clinical orthopaedics and related research* 474.8 (2016), pp. 1886–1893. DOI: [10.1007/s11999-016-4732-4](https://doi.org/10.1007/s11999-016-4732-4).
- [23] R. Kumavat et al. “Biomarkers of Joint Damage in Osteoarthritis: Current Status and Future Directions”. In: *Mediators of Inflammation* 2021 (Mar. 2021), pp. 1–15. DOI: [10.1155/2021/5574582](https://doi.org/10.1155/2021/5574582).
- [24] X. Cai et al. “New Trends in Pharmacological Treatments for Osteoarthritis”. In: *Frontiers in Pharmacology* 12 (2021), p. 701. ISSN: 1663-9812. DOI: [10.3389/fphar.2021.645842](https://doi.org/10.3389/fphar.2021.645842).
- [25] A. Latourte, M. Kloppenburg, and P. Richette. “Emerging pharmaceutical therapies for osteoarthritis”. In: *Nat Rev Rheumatol* 16 (2020), pp. 673–688. DOI: [10.1038/s41584-020-00518-6](https://doi.org/10.1038/s41584-020-00518-6).
- [26] R. Lesage, J. Kerkhofs, and L. Geris. “Computational Modeling and Reverse Engineering to Reveal Dominant Regulatory Interactions Controlling Osteochondral Differentiation: Potential for Regenerative Medicine”. In: *Frontiers in Bioengineering and Biotechnology* 6 (2018), p. 165. DOI: [10.3389/fbioe.2018.00165](https://doi.org/10.3389/fbioe.2018.00165).
- [27] S. Mukherjee et al. “Use of Computational Modeling to Study Joint Degeneration: A Review”. In: *Frontiers in Bioengineering and Biotechnology* 8 (2020), p. 93. ISSN: 2296-4185. DOI: [10.3389/fbioe.2020.00093](https://doi.org/10.3389/fbioe.2020.00093).
- [28] I. N. Melas et al. “Modeling of signaling pathways in chondrocytes based on phosphoproteomic and cytokine release data”. In: *Osteoarthritis and cartilage* 22.3 (2014). DOI: [10.1016/j.joca.2014.01.001](https://doi.org/10.1016/j.joca.2014.01.001).

- [29] L Baumgartner et al. “Simulating intervertebral disc cell behaviour within 3D multifactorial environments”. In: *Bioinformatics* 37.9 (Dec. 2020), pp. 1246–1253. ISSN: 1367-4803. DOI: [10.1093/bioinformatics/btaa939](https://doi.org/10.1093/bioinformatics/btaa939).
- [30] M. Ceresa et al. “Multi-scale immunological and biomechanical model of emphysema progression”. In: *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (2017), pp. 2712–2715. DOI: [10.1109/EMBC.2017.8037417](https://doi.org/10.1109/EMBC.2017.8037417).
- [31] V.A. Folcik, G.C. An, and C.G. Orosz. “The Basic Immune Simulator: An agent-based model to study the interactions between innate and adaptive immunity”. In: *Theor Biol Med Model* 4.39 (2007). DOI: [10.1186/1742-4682-4-39](https://doi.org/10.1186/1742-4682-4-39).
- [32] Eric Bonabeau. “Agent-based modeling: Methods and techniques for simulating human systems”. In: *Proceedings of the National Academy of Sciences* 99.suppl 3 (2002), pp. 7280–7287. ISSN: 0027-8424. DOI: [10.1073/pnas.082080899](https://doi.org/10.1073/pnas.082080899).
- [33] D. A. Samuelson and C. M. Macal. *Agent-Based Simulation Comes of Age*. 2006. URL: <https://www.informs.org/ORMS-Today/Archived-Issues/2006/orms-8-06/Agent-Based-Simulation-Comes-of-Age> (visited on 05/27/2021).
- [34] Gary An. “Introduction of an agent-based multi-scale modular architecture for dynamic knowledge representation of acute inflammation”. In: *Theor Biol Med Model* 5.11 (2008). DOI: [10.1186/1742-4682-5-11](https://doi.org/10.1186/1742-4682-5-11).
- [35] Scott de Marchi and Scott E. Page. “Agent-Based Models”. In: *Annual Review of Political Science* 17.1 (2014), pp. 1–20. DOI: [10.1146/annurev-polisci-080812-191558](https://doi.org/10.1146/annurev-polisci-080812-191558).
- [36] Gary An. “Integrating physiology across scales and formalizing hypothesis exploration with agent-based modeling”. In: *Journal of Applied Physiology* 118.10 (2015), pp. 1191–1192. DOI: [10.1152/jappphysiol.00243.2015](https://doi.org/10.1152/jappphysiol.00243.2015).
- [37] Ashlee N. Ford Versypt. “Multiscale modeling in disease”. In: *Current Opinion in Systems Biology* (2021). ISSN: 2452-3100. DOI: [10.1016/j.coisb.2021.05.001](https://doi.org/10.1016/j.coisb.2021.05.001).
- [38] M. Ceresa et al. “Coupled Immunological and Biomechanical Model of Emphysema Progression”. In: *Frontiers in Physiology* 9 (2018), p. 388. ISSN: 1664-042X. DOI: [10.3389/fphys.2018.00388](https://doi.org/10.3389/fphys.2018.00388).
- [39] A. Shuaib et al. “Heterogeneity in The Mechanical Properties of Integrins Determines Mechanotransduction Dynamics in Bone Osteoblasts”. In: *Sci Rep* 9.13113 (2019). DOI: [10.1038/s41598-019-47958-z](https://doi.org/10.1038/s41598-019-47958-z).
- [40] V. B. Shim et al. “A Multiscale Framework Based on the Physiome Markup Languages for Exploring the Initiation of Osteoarthritis at the Bone–Cartilage Interface”. In: *IEEE Transactions on Biomedical Engineering* 58.12 (2011), pp. 3532–3536. DOI: [10.1109/TBME.2011.2165955](https://doi.org/10.1109/TBME.2011.2165955).
- [41] D. E. Shepherd and B. B. Seedhom. “Thickness of human articular cartilage in joints of the lower limb”. In: *Annals of the rheumatic diseases* 58.1 (1999), pp. 27–34. DOI: [10.1136/ard.58.1.27](https://doi.org/10.1136/ard.58.1.27).
- [42] Michael M. Cox and L. Nel David. *Lehninger principles of biochemistry*. Ed. by L. Schultz et al. Winslow, Susan, 2008. ISBN: 071677108X.
- [43] B. Alberts et al. *Essential Cell Biology*. Ed. by M. Morales et al. 4th edition. Garland Science, Taylor Francis Group, LLC, 2014. ISBN: 978-0-8153-4454-4.

- [44] H.P. Erickson. “Size and Shape of Protein Molecules at the Nanometer Level Determined by Sedimentation, Gel Filtration, and Electron Microscopy”. In: *Biol Proced Online* 11.32 (2009). DOI: [10.1007/s12575-009-9008-x](https://doi.org/10.1007/s12575-009-9008-x).
- [45] Zijun Zhang. “Chondrons and the Pericellular Matrix of Chondrocytes”. In: *Tissue Engineering Part B: Reviews* 21.3 (2015), pp. 267–277. DOI: [10.1089/ten.teb.2014.0286](https://doi.org/10.1089/ten.teb.2014.0286).
- [46] L. Eberhardsteiner, C. Hellmich, and S. Scheiner. “Layered water in crystal interfaces as source for bone viscoelasticity: arguments from a multiscale approach”. In: *Computer Methods in Biomechanics and Biomedical Engineering* 17.1 (2014), pp. 48–63. DOI: [10.1080/10255842.2012.670227](https://doi.org/10.1080/10255842.2012.670227).
- [47] J. S. Mackie, P. Meares, and Eric Keightley Rideal. “The diffusion of electrolytes in a cation-exchange resin membrane I. Theoretical”. In: *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 232.1191 (1955), pp. 498–509. DOI: [10.1098/rspa.1955.0234](https://doi.org/10.1098/rspa.1955.0234).
- [48] W. Wilson, J.M. Huyghe, and C.C. van Donkelaar. “Depth-dependent Compressive Equilibrium Properties of Articular Cartilage Explained by its Composition”. In: *Biomech Model Mechanobiol* 6 (2007), pp. 43–53. DOI: [10.1007/s10237-006-0044-z](https://doi.org/10.1007/s10237-006-0044-z).
- [49] T. Aigner, N. Schmitz, and D. M. Salter. “175 - Pathogenesis and pathology of osteoarthritis”. In: *Rheumatology (Sixth Edition)*. Ed. by Marc C. Hochberg et al. Mosby, 2015, pp. 1462–1476. ISBN: 978-0-323-09138-1. DOI: [10.1016/B978-0-323-09138-1.00175-3](https://doi.org/10.1016/B978-0-323-09138-1.00175-3).
- [50] Kevin W.-H. Lo et al. “Studies of bone morphogenetic protein-based surgical repair”. In: *Advanced Drug Delivery Reviews* 64.12 (2012). Targeted delivery of therapeutics to bone and connective tissues, pp. 1277–1291. DOI: [10.1016/j.addr.2012.03.014](https://doi.org/10.1016/j.addr.2012.03.014).
- [51] P.A. Todd and K.L. Goa. “Interferon Gamma-1b”. In: *Drugs* 43 (1992), pp. 111–122. DOI: [10.2165/00003495-199243010-00008](https://doi.org/10.2165/00003495-199243010-00008).
- [52] D. Fouque, S. C. Peng, and J. D. Kopple. “Pharmacokinetics of recombinant human insulin-like growth factor-1 in dialysis patients”. In: *Kidney international* 47.3 (1995), pp. 869–875. DOI: [10.1038/ki.1995.130](https://doi.org/10.1038/ki.1995.130).
- [53] S. Kudo et al. “Clearance and Tissue Distribution of Recombinant Human Interleukin 1 in Rats”. In: *Cancer research* 50 (1990), pp. 5751–5755.
- [54] P. J. Conlon et al. “Interleukin-4 (B-cell stimulatory factor-1) augments the in vivo generation of cytotoxic cells in immunosuppressed animals”. In: *Biotechnology therapeutics* 1.1 (1989), pp. 31–41.
- [55] G. Endler et al. “The Interleukin-6 G(174)C Promoter Polymorphism Does Not Determine Plasma Interleukin-6 Concentrations in Experimental Endotoxemia in Humans”. In: *Clinical Chemistry* 50.1 (2004), pp. 195–200. DOI: [10.1373/clinchem.2003.022459](https://doi.org/10.1373/clinchem.2003.022459).
- [56] T. Orlikowsky et al. “Evaluation of IL-8-Concentrations in Plasma and Lysed EDTA-Blood in Healthy Neonates and Those with Suspected Early Onset Bacterial Infection”. In: *Pediatr Res* 56 (2004), pp. 804–809. DOI: [10.1203/01.PDR.0000141523.68664.4A](https://doi.org/10.1203/01.PDR.0000141523.68664.4A).

- [57] H. Rachmawati et al. “Intravenous Administration of Recombinant Human IL-10 Suppresses the Development of Anti-Thy 1–Induced Glomerulosclerosis in Rats”. In: *PDA Journal of Pharmaceutical Science and Technology* 65.2 (2011), pp. 116–130.
- [58] D. J. Wodsedalek et al. “IL-13 promotes in vivo neonatal cardiomyocyte cell cycle activity and heart regeneration”. In: *American Journal of Physiology-Heart and Circulatory Physiology* 316.1 (2019), H24–H34. DOI: [10.1152/ajpheart.00521.2018](https://doi.org/10.1152/ajpheart.00521.2018).
- [59] M. Balestrino. “Cytokine Imbalances in Multiple Sclerosis: A Computer Simulation”. In: *Masters of Engineering Projects* (2009). URL: <https://hdl.handle.net/1813/11726>.
- [60] D. J. Herzyk et al. “Preclinical Safety of Recombinant Human Interleukin-18”. In: *Toxicologic Pathology* 31.5 (2003), pp. 554–561. DOI: [10.1080/01926230390226681](https://doi.org/10.1080/01926230390226681).
- [61] D. H. Gunawardana et al. “A Phase I Study of Recombinant Human Leukemia Inhibitory Factor in Patients with Advanced Cancer”. In: *Clinical Cancer Research* 9.6 (2003), pp. 2056–2065. ISSN: 1078-0432. URL: <https://clincancerres.aacrjournals.org/content/9/6/2056>.
- [62] J. Salvatierra et al. “Cartilage and serum levels of nitric oxide in patients with hip osteoarthritis”. In: *The Journal of rheumatology* 26.9 (1999), pp. 2015–2017. URL: <https://pubmed.ncbi.nlm.nih.gov/10493684/>.
- [63] J. S. Beckman and W. H. Koppenol. “Nitric oxide, superoxide, and peroxynitrite: the good, the bad, and ugly”. In: *The American journal of physiology* 271.5 Pt 1 (1996), pp. C1424–C1437. DOI: [10.1152/ajpcell.1996.271.5.C1424](https://doi.org/10.1152/ajpcell.1996.271.5.C1424).
- [64] C. Brochhausen et al. “Immobilization and controlled release of prostaglandin E2 from poly-L-lactide-co-glycolide microspheres”. In: *Journal of Biomedical Materials Research Part A* 91A.2 (2009), pp. 454–462. DOI: [10.1002/jbm.a.32215](https://doi.org/10.1002/jbm.a.32215).
- [65] B. J. Rollins et al. “Environment-dependent growth inhibition of human epidermal keratinocytes by recombinant human transforming growth factor-beta”. In: *Journal of Cellular Physiology* 139.3 (1989), pp. 455–462. DOI: [10.1002/jcp.1041390302](https://doi.org/10.1002/jcp.1041390302).
- [66] A. Neininger et al. “MK2 targets AU-rich elements and regulates biosynthesis of tumor necrosis factor and interleukin-6 independently at different post-transcriptional levels”. In: *The Journal of biological chemistry* 277.5 (2002), pp. 3065–3068. DOI: [10.1074/jbc.C100685200](https://doi.org/10.1074/jbc.C100685200).
- [67] S. D. Finley et al. “Pharmacokinetics and pharmacodynamics of VEGF-neutralizing antibodies”. In: *BMC Syst Biol* 5.193 (2011). DOI: [10.1186/1752-0509-5-193](https://doi.org/10.1186/1752-0509-5-193).
- [68] L. Mendoza, D. Thieffry, and E. R. Alvarez-Buylla. “Genetic control of flower morphogenesis in *Arabidopsis thaliana*: a logical analysis.” In: *Bioinformatics* 15.7 (1999), pp. 593–606. ISSN: 1367-4803. DOI: [10.1093/bioinformatics/15.7.593](https://doi.org/10.1093/bioinformatics/15.7.593).
- [69] L. Sánchez and D. Thieffry. “Segmenting the fly embryo:: a logical analysis of the pair-rule cross-regulatory module”. In: *Journal of Theoretical Biology* 224.4 (2003), pp. 517–537. ISSN: 0022-5193. DOI: [10.1016/S0022-5193\(03\)00201-7](https://doi.org/10.1016/S0022-5193(03)00201-7).
- [70] L. Mendoza and I. Xenarios. “A method for the generation of standardized qualitative dynamical systems of regulatory networks”. In: *Theor Biol Med Model* 3.13 (2006). DOI: [10.1186/1742-4682-3-13](https://doi.org/10.1186/1742-4682-3-13).

- [71] J. Feher. “9.1 - General Principles of Endocrinology”. In: *Quantitative Human Physiology (Second Edition)*. Ed. by Joseph Feher. Second Edition. Academic Press, 2017, pp. 853–869. ISBN: 978-0-12-800883-6. DOI: [10.1016/B978-0-12-800883-6.00084-7](https://doi.org/10.1016/B978-0-12-800883-6.00084-7).
- [72] A.I. Tsuchida et al. “Cytokine profiles in the joint depend on pathology, but are different between synovial fluid, cartilage tissue and cultured chondrocytes”. In: *Arthritis Res Ther* 16.441 (2014). DOI: [10.1186/s13075-014-0441-0](https://doi.org/10.1186/s13075-014-0441-0).
- [73] M. Adouni and A. Shirazi-Adl. “Evaluation of knee joint muscle forces and tissue stresses-strains during gait in severe OA versus normal subjects”. In: *Journal of Orthopaedic Research* 32.1 (2014), pp. 69–78. DOI: [10.1002/jor.22472](https://doi.org/10.1002/jor.22472).
- [74] D. M. Salter et al. “Integrin–Interleukin-4 Mechanotransduction Pathways in Human Chondrocytes”. In: *Clinical Orthopaedics and Related Research* 391.Suppl (2001). DOI: [10.1097/00003086-200110001-00006](https://doi.org/10.1097/00003086-200110001-00006).
- [75] Z. Zhao et al. “Mechanotransduction pathways in the regulation of cartilage chondrocyte homeostasis”. In: *Journal of Cellular and Molecular Medicine* 24.10 (2020), pp. 5408–5419. DOI: [10.1111/jcmm.15204](https://doi.org/10.1111/jcmm.15204).
- [76] S. B. Abramson. “Osteoarthritis and nitric oxide”. In: *Osteoarthritis and cartilage* 16.Suppl 2 (2008), S15–S20. DOI: [10.1016/S1063-4584\(08\)60008-4](https://doi.org/10.1016/S1063-4584(08)60008-4).
- [77] T. Olee et al. “IL-18 Is Produced by Articular Chondrocytes and Induces Proinflammatory and Catabolic Responses”. In: *The Journal of Immunology* 162.2 (1999), pp. 1096–1100. URL: <https://www.jimmunol.org/content/162/2/1096>.
- [78] R. S. McCulloch et al. “Progression of Gene Expression Changes following a Mechanical Injury to Articular Cartilage as a Model of Early Stage Osteoarthritis”. In: *Arthritis* (2014). DOI: [10.1155/2014/371426](https://doi.org/10.1155/2014/371426).
- [79] S. E. Guidoni and C. M. Aldao. “On diffusion, drift and the Einstein relation”. In: *European Journal of Physics* 23.4 (2002), pp. 395–402. DOI: [10.1088/0143-0807/23/4/302](https://doi.org/10.1088/0143-0807/23/4/302).

SUPPLEMENTARY INFORMATION

Molecular weights of the soluble mediators	
Molecule	Molecular mass [Da = g/mol]
BMP2	44702
IFN γ	19348
IGF-1	21841
IL-1 β	30748
IL-4	17492
IL-6	23718
IL-8	11098
IL-10	20517
IL-13	15816
IL-17	17504
IL-18	22326
LIF	22008
NO	30,01
PGE2	352,4
TGF- β	44341
TNF- α	25644
VEGF	27042

Table S1: Molecular weights (Da) of the soluble mediators extracted from UniProt^I, except for NO^{II} and PGE2^{III} (g/mol).

^IUniProt; ^{II}NO molecular weight; ^{III}PGE2 molecular weight

Minimum radius of the soluble mediators	
Molecule	R_{min} [nm]
BMP2	2,342
IFN γ	1,772
IGF-1	1,845
IL-1 β	2,068
IL-4	1,713
IL-6	1,896
IL-8	1,472
IL-10	1,807
IL-13	1,657
IL-17	1,714
IL-18	1,858
LIF	1,850
NO	0,205
PGE2	0,466
TGF- β	2,336
TNF- α	1,946
VEGF	1,981

Table S2: Minimum radius [nm] of the soluble mediators calculated by equation (3)

Parameters of the Stokes-Einstein equation		
Parameter	Units	Value
k_B (Boltzmann's constant)	Joules/Kelvin (J/K)	$1,380649 \cdot 10^{-23}$
T (Absolute temperature)	Kelvins (K)	310,15
η (Viscosity)	Pascals·Second (Pas)	$6,990993 \cdot 10^{-07}$
r (Radius)	Meters (m)	See Table S2

Table S3: Parameters used to solve Stokes-Einstein equation (4)

Chondrocyte's regulatory network		
Nodes	Activators	Inhibitors
ACAN	BMP-2, IGF-1, IL-4, IL-10, TGF- β	PGE2, TNF- α
ADAMTs	IL-6, NO	-
BMP-2	-	-
CASP-8	NO	IL-10
COL2 α	IGF-1, IL-10	IL-1 β , NO, TNF- α
CYCS	NO	IL-10
IFN- γ	-	-
IGF-1	-	IL-1 β , IL-6
IL1- β	LIF, TNF- α	BMP-2, IL-4, IL-13, TGF- β
IL-4	-	-
IL-6	IL-8	IL-10, IL-13
IL-8	IL-1 β , LIF, TNF- α	IL-6, IL-10, IL-13
IL-10	IL-4	-
IL-13	IL-4	-
IL-17	-	-
IL-18	IL-1 β , NO	IL-13
LIF	-	-
MMP-1	IL-1 β	TIMP
MMP-3	NO, TNF- α	TGF- β
MMP-13	BMP-2, IL-1 β , IL-8, IL-17, IL-18, TNF- α	IL-13
MMP-14	IL-18, NO	IL-10
NO	IFN- γ , IL-1 β , TNF- α	IL-4
PGE2	IL-1 β , IL-18	-
PGA	IL-1 β , TNF- α	-
TGF- β	-	TNF- α
TIMP	IL-6, TGF- β	TNF- α
TNF- α	IL-1 β , IL-6, IL-17	IL-10, IL-13
VEGF	ADAMTs, MMP-3, MMP-13, PGE2	IL-13

Table S4: Boolean regulatory network representing a chondrocyte-specific protein interactome, expressed as a table (see **Figure 3**)[15].

Normalization parameters

Tsuchida et al. (2014) [72] experimental data		
Molecule	Molecules <i>pg</i> /Cartilage <i>g</i>	Molecules <i>pg</i> /DNA <i>mg</i>
BMP2	1.142,5*	6.483,613753*
IFN γ	226	173
IGF-1	1.142,5*	6.483,613753*
IL-1 β	23	816
IL-4	2	40
IL-6	1.473	1.918
IL-8	15.000	1.070.000
IL-10	812	16.529
IL-13	799	3.249.000
IL-17	33,27519333**	22.577,662295**
IL-18	304,2733906**	17.051,744247**
LIF	3.529	194.000
NO	1.142,5*	6.483,813753*
PGE2	1.142,5*	6.483,813753*
TGF- β	1.142,5*	6.483,813753*
TNF- α	3.069	426
VEGF	212.000	828.000

Table S5: Values for concentration of proteins per gram of cartilage and concentration of proteins per milligram of DNA, extracted from Tsuchida et al. (2014) [72].

*Values approximated to the median.

Values obtained by inferring through regression (see **Figure S1 & S2).

From concentration per mass to number of molecules per volume

The concentration of molecules per gram of cartilage was used to establish the maximum number of molecules per μm^3 of cartilage. For that, the following conversion factor was used:

$$\frac{\text{number of protein (P) molecules}}{\mu\text{m}^3 \text{ cartilage (C)}} = \frac{X \text{ pg P}}{1 \text{ g C}} \cdot \frac{10^{-12} \text{ g P}}{1 \text{ pg P}} \cdot \frac{B \text{ moles P}}{A \text{ g P}} \cdot \frac{6,022 \cdot 10^{23} \text{ molecules P}}{1 \text{ mole P}} \cdot \frac{1.1 \text{ g C}}{1 \text{ cm}^3 \text{ C}} \cdot \frac{1 \text{ cm}^3 \text{ C}}{10^{12} \mu\text{m}^3 \text{ C}} \quad (\text{S1})$$

From concentration per milligram of DNA to number of molecules released per chondrocyte

The concentration of molecules per milligram of DNA was used to establish the maximum number of molecules released by a chondrocyte. For that, the following conversion factor

was used:

$$\frac{\text{number of released protein (P) molecules}}{1 \text{ chondrocyte (Chon.)}} = \frac{X \text{ pg P}}{1 \text{ mg DNA}} \cdot \frac{1 \text{ mg DNA}}{10^9 \text{ pg DNA}} \cdot \frac{6 \text{ pg DNA}}{1 \text{ Chon}} \cdot \frac{10^{-12} \text{ g P}}{1 \text{ pg P}} \cdot \frac{B \text{ moles P}}{A \text{ g P}} \cdot \frac{6,022 \cdot 10^{23} \text{ molecules P}}{1 \text{ mole P}} \quad (\text{S2})$$

Inferring the missing data

IL-17 and IL-18 maximum concentrations were determined inferring them by a regression between Tsuchida et al. (2014)[72] data and experimental data of the presented clinical study (see **Figure S1**). Releasing of IL-17 and IL-18 was also inferred by the regression between concentration of protein per gram of cartilage and concentration of protein per milligram of DNA (see **Figure S2**).

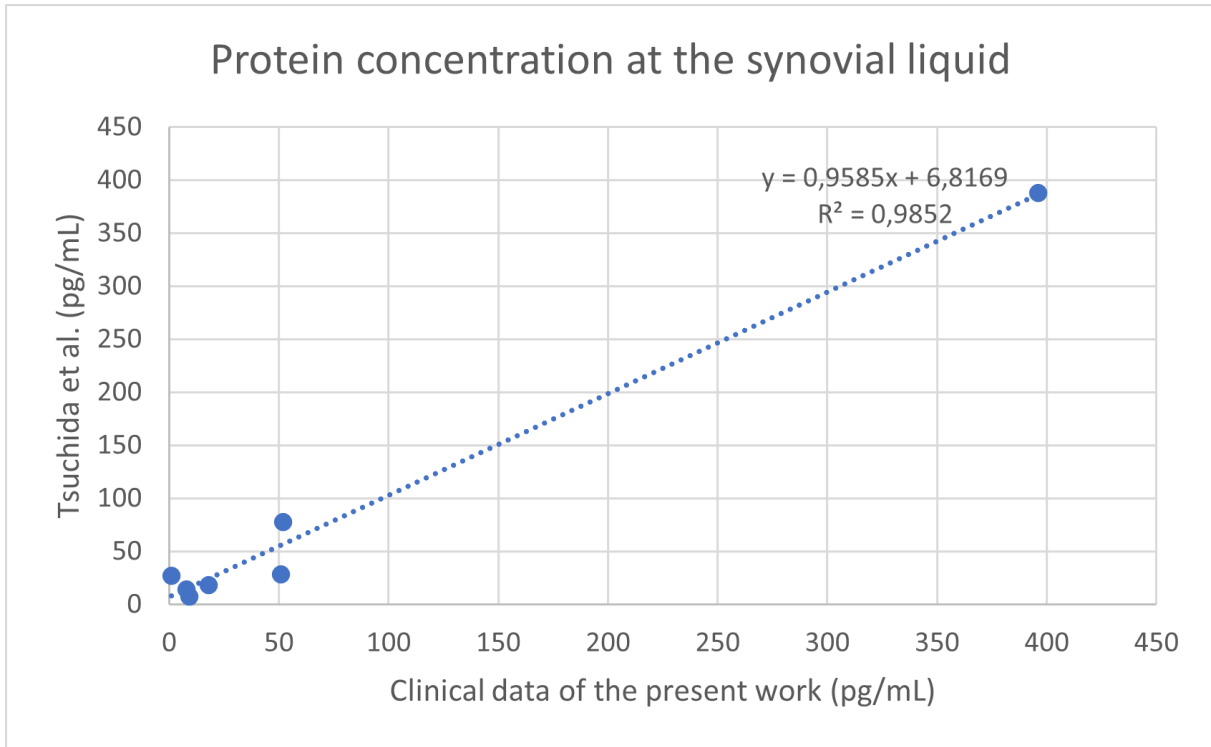


Figure S1: Linear regression between Tsuchida et al. (2014) [72] and present clinical study protein concentrations at the synovial liquid.

However, the maximum number of molecules in the model depend on the volume of cartilage represented. For that, the number of molecules per μm^3 of cartilage was related with the number of molecules for the whole volume by the following relation:

$$\text{number of molecules at the model} = \frac{\text{number of molecules}}{\mu\text{m}^3 \text{ cartilage}} \cdot \text{volume represented} (\mu\text{m}^3) \quad (\text{S3})$$

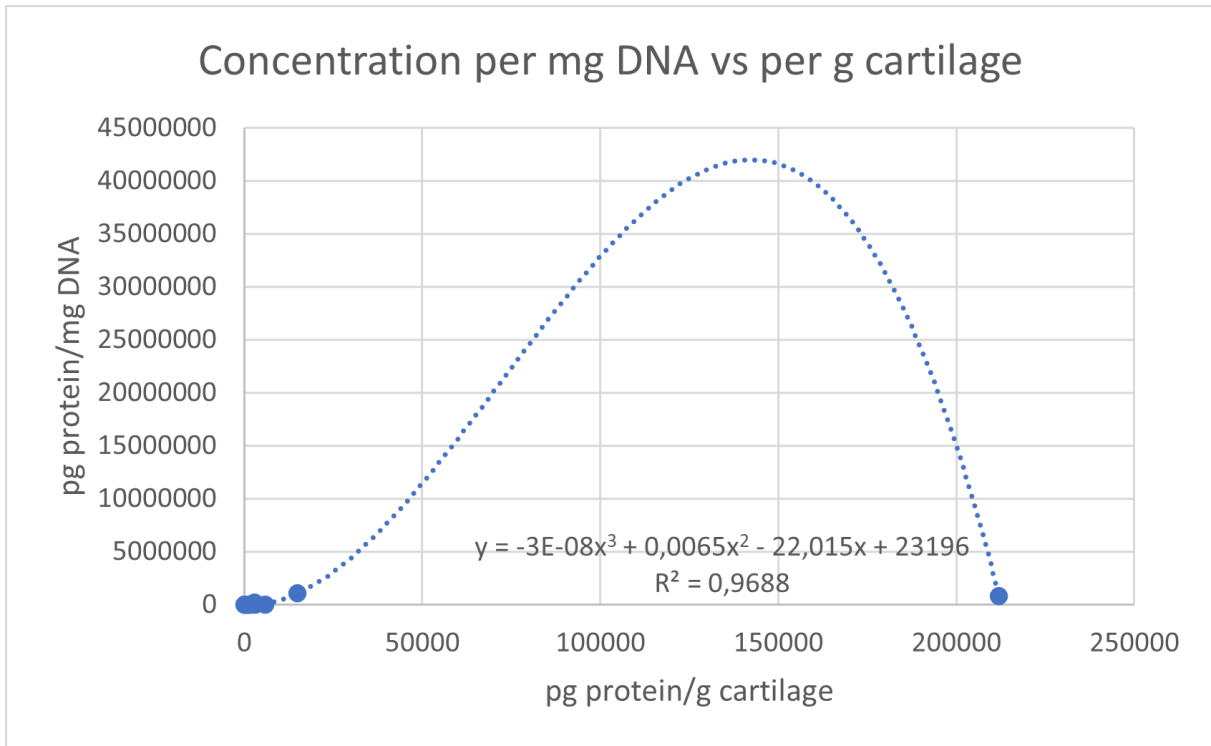


Figure S2: Third grade polynomic regression between protein concentration per gram of cartilage and protein concentration per milligram of DNA from Tsuchida et al. (2014) [72].

Experimental study supporting information

The synovial liquid concentration of soluble mediators was translated to the corresponding number of molecules per μm^3 of cartilage by the following conversion factor:

$$\frac{\text{number of protein (P) molecules}}{\text{mL synovial liquid (SL)}} = \frac{X \text{ pg P}}{1 \text{ mL SL}} \cdot \frac{10^{-12} \text{ g P}}{1 \text{ pg P}} \cdot \frac{B \text{ moles P}}{A \text{ g P}} \cdot \frac{6,022 \cdot 10^{23} \text{ molecules P}}{1 \text{ mole P}} \cdot \frac{1 \text{ mL SL}}{10^{12} \mu\text{m}^3 \text{ SL}} \quad (\text{S4})$$

Multiplexing immunofluorescence assays methodology	
Protein	Method for multiplexing immunofluorescence assay
TIM-1	Milliplex TIMP Magnetic (Millipore)
TIM-2	Milliplex TIMP Magnetic (Millipore)
TIM-3	Milliplex TIMP Magnetic (Millipore)
TIM-4	Milliplex TIMP Magnetic (Millipore)
EOTAX	Procartaplex (Invitrogen) ¹
IFN- γ	Procartaplex (Invitrogen) ¹
IL-17 α	Procartaplex (Invitrogen) ¹
IL-18	Procartaplex (Invitrogen) ¹
IL-4	Procartaplex (Invitrogen) ¹
MIP-1 α	Procartaplex (Invitrogen) ¹
MMP-1	Procartaplex (Invitrogen) ¹
MMP-13	Procartaplex (Invitrogen) ¹
MMP-2	Procartaplex (Invitrogen) ¹
MMP-3	Procartaplex (Invitrogen) ¹
TNF- α	Procartaplex (Invitrogen) ¹
VEGF	Procartaplex (Invitrogen) ¹
IL-10	Procartaplex (Invitrogen) ²
IL-12(P70)	Procartaplex (Invitrogen) ²
IL-13	Procartaplex (Invitrogen) ²
IL-1 β	Procartaplex (Invitrogen) ²
IL-1R α	Procartaplex (Invitrogen) ²
IL-2	Procartaplex (Invitrogen) ²
IL-5	Procartaplex (Invitrogen) ²
IL-6	Procartaplex (Invitrogen) ²
IL-8	Procartaplex (Invitrogen) ²
LEPTIN	Procartaplex (Invitrogen) ²
MCP-1	Procartaplex (Invitrogen) ²
MIP-1 β	Procartaplex (Invitrogen) ²
RANTES	Procartaplex (Invitrogen) ²
ADAMTs-4	ADAMTs-4 ELISA kit (FineTest)
ADAMTs-5	ADAMTs-5 ELISA kit (FineTest)

Table S6: Analysed proteins by multiplexing immunofluorescence assays and used methodology.

¹First customization of Procartaplex (Invitrogen);

²Second customization of Procartaplex (Invitrogen)

Mean synovial liquid concentrations [pg/mL]				
Molecule	KL=2 & Conservative	KL=3 & Conservative	KL=2 & TKA	KL=3 & TKA
IFN γ	20,79487326	28,30996022	23,74170015	31,61825758
IL-1 β	14,27	14,27	14,27	14,27
IL-4	24,0443567	25,63579627	40,17124717	25,77836040
IL-6	209,4413261	319,670289	155,2782592	577,6006303
IL-8	14,82548841	41,77418384	74,6181959	135,2602347
IL-10	8,1088838	6,7065592	8,23601666	8,34072158
IL-13	17,97	17,97	17,97	17,97
IL-17	41,84378078	49,71391376	19,79590207	30,8719064
IL-18	132,7609359	282,3393098	288,2489445	371,1951421
TNF- α	150,260714	282,3393098	183,9001691	227,5800616
VEGF	1.658,674783	3.726,101877	4.780,425546	3.649,67782

Table S7: Mean synovial liquid concentrations in pg/mL of soluble mediators obtained from the clinical study.

Code for the model

Chondrocyte agents

```
package kneeOA_V1;

import kneeOA_V1.Cytokine.BMP2;
import kneeOA_V1.Cytokine.IFNG;
import kneeOA_V1.Cytokine.IGF1;
import kneeOA_V1.Cytokine.IL10;
import kneeOA_V1.Cytokine.IL13;
import kneeOA_V1.Cytokine.IL17;
import kneeOA_V1.Cytokine.IL18;
import kneeOA_V1.Cytokine.IL1b;
import kneeOA_V1.Cytokine.IL4;
import kneeOA_V1.Cytokine.IL6;
import kneeOA_V1.Cytokine.IL8;
import kneeOA_V1.Cytokine.LIF;
import kneeOA_V1.Cytokine.NO;
import kneeOA_V1.Cytokine.PEG2;
import kneeOA_V1.Cytokine.TGFb;
import kneeOA_V1.Cytokine.TNFa;
import kneeOA_V1.Cytokine.VEGF;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintWriter;
import java.math.RoundingMode;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.concurrent.ThreadLocalRandom;

import repast.simphony.engine.environment.RunEnvironment;
import repast.simphony.engine.schedule.ScheduledMethod;
import repast.simphony.parameter.Parameters;
import repast.simphony.space.continuous.ContinuousSpace;
import repast.simphony.space.continuous.NdPoint;
import repast.simphony.space.grid.Grid;
import repast.simphony.space.grid.GridPoint;
import repast.simphony.util.ContextUtils;
import repast.simphony.util.SimUtilities;
import repast.simphony.context.Context;
import repast.simphony.query.space.grid.GridCell;
import repast.simphony.query.space.grid.GridCellNgh;
import repast.simphony.random.RandomHelper;

public class Chondrocyte {
    private ContinuousSpace<Object> space;
    private Grid<Object> grid;
    static int id;

    // Get the interactions of the network from a .csv table.
```

```

    static String location =
"C:\\\\Users\\\\andre\\\\Desktop\\\\ABM_TFG\\\\";
    static String filename = "Interactions.csv";
    static String filenameBaseline = "Baseline.csv";
    static String path = location + filename; // Directory of the network.
    static String pathBaseline = location + filenameBaseline;

// Initialize variables to solve Mendoza & Xenario's equation.
    static final double DELTA = 1;
    static final int h = 10;

// Declare global class variables.
    static HashMap<String,Double> FinalSteadyStates = new
HashMap<String,Double>();
    static ArrayList<String> NodeNames = new ArrayList<String>();

    static int NumOfNodes;
    static double Mact [][];
    static double Minh [][];
    static double X[];
    static double baseline[];
    static int perturbedNodes[];
    static double k1[];
    static double k2[];
    static double k3[];
    static double k4[];

    static StringBuilder sb;

    static Parameters params = null;

// Initialize global class variables

    public Chondrocyte (ContinuousSpace<Object> space, Grid<Object> grid)
throws IOException {
        this.space = space;
        this.grid = grid;

        params = RunEnvironment.getInstance().getParameters();
        NumOfNodes = extractNumOfNodes(path);
        NodeNames = readNodeNames(path);
        baseline = readBaseline(pathBaseline, NumOfNodes);
        Mact = readActivators(path, NumOfNodes);
        Minh = readInhibitors(path, NumOfNodes);
        X = new double[NumOfNodes];

        sb = new StringBuilder();

        perturbedNodes = new int[NumOfNodes];
    }

/*
 * @throws IOException
*****
*****
 * Solve the system ODE (Mendoza & Xenario's equation) and updates the
environment by adding cytokines in answer to the *

```



```

// Create the initial state (1 if the node is perturbated,
baseline if not)
for (int i = 0; i < NumOfNodes; i++) {
    if(perturbatedNodes[i] == 1) {
        X[i] = 1;
    } else {
        X[i] = baseline[i];
    }
}

/*
*****
* SOLIVING DIFFERENTIAL EQUATIONS ACCORDING TO MENDOZA ET AL.
by RK4 method *
*****
*/

for (int time = 0; time < 750; time ++) {

// k1 calculus
k1 = f_i(X, Mact, Minh, h, NumOfNodes);

// k2 calculus
double[] k2factor = new double[NumOfNodes];
for (int i = 0; i < NumOfNodes; i++) {
    k2factor[i] = X[i] + k1[i]*0.5;
}
k2 = f_i(k2factor, Mact, Minh, h, NumOfNodes);

// k3 calculus
double[] k3factor = new double[NumOfNodes];
for (int i = 0; i < NumOfNodes; i++) {
    k3factor[i] = X[i] + k2[i]*0.5;
}
k3 = f_i(k3factor, Mact, Minh, h, NumOfNodes);

// k4 calculus
double[] k4factor = new double[NumOfNodes];
for (int i = 0; i < NumOfNodes; i++) {
    k4factor[i] = X[i] + k3[i];
}
k4 = f_i(k4factor, Mact, Minh, h, NumOfNodes);

// X[i] for each time step adding the possible
perturbations

for (int i = 0 ; i < NumOfNodes; i++) {
    X[i] = X[i] + (k1[i] + 2.0*k2[i] + 2.0*k3[i]
+ k4[i])/6.0;

    if(perturbatedNodes[i] == 1) {
        X[i] = 1;
    }
}

```

```

    }
}

/*
*****
* ADDING AGENTS TO THE SPACE ACCORDING TO THE CALCULATED STEADY
STATES *
*****
*/

// Counting the number of cytokines in the environment to
determine if new ones need to be created

int numberOfBMP2 = 0;
int numberOfIFNg = 0;
int numberOfIGF1 = 0;
int numberOfIL10 = 0;
int numberOfIL13 = 0;
int numberOfIL17 = 0;
int numberOfIL18 = 0;
int numberOfIL1b = 0;
int numberOfIL4 = 0;
int numberOfIL6 = 0;
int numberOfIL8 = 0;
int numberOfLIF = 0;
int numberOfNO = 0;
int numberOfPEG2 = 0;
int numberOfTGFb = 0;
int numberOfTNFa = 0;
int numberOfVEGF = 0;

for(Object obj : context) {
    if(obj instanceof Cytokine.BMP2) {
        numberOfBMP2 = numberOfBMP2 + 1;
    } else if(obj instanceof Cytokine.IFNG) {
        numberOfIFNg = numberOfIFNg + 1;
    } else if(obj instanceof Cytokine.IGF1) {
        numberOfIGF1 = numberOfIGF1 + 1;
    } else if(obj instanceof Cytokine.IL10) {
        numberOfIL10 = numberOfIL10 + 1;
    } else if(obj instanceof Cytokine.IL13) {
        numberOfIL13 = numberOfIL13 + 1;
    } else if(obj instanceof Cytokine.IL17) {
        numberOfIL17 = numberOfIL17 + 1;
    } else if(obj instanceof Cytokine.IL18) {
        numberOfIL18 = numberOfIL18 + 1;
    } else if(obj instanceof Cytokine.IL1b) {
        numberOfIL1b = numberOfIL1b + 1;
    } else if(obj instanceof Cytokine.IL4) {
        numberOfIL4 = numberOfIL4 + 1;
    } else if(obj instanceof Cytokine.IL6) {
        numberOfIL6 = numberOfIL6 + 1;
    } else if(obj instanceof Cytokine.IL8) {
        numberOfIL8 = numberOfIL8 + 1;
    } else if(obj instanceof Cytokine.LIF) {
        numberOfLIF = numberOfLIF + 1;
    }
}

```

```

    } else if(obj instanceof Cytokine.NO) {
        numberOfNO = numberOfNO + 1;
    } else if(obj instanceof Cytokine.PEG2) {
        numberOfPEG2 = numberOfPEG2 + 1;
    } else if(obj instanceof Cytokine.TGFb) {
        numberOfTGFb = numberOfTGFb + 1;
    } else if(obj instanceof Cytokine.TNFa) {
        numberOfTNFa = numberOfTNFa + 1;
    } else if(obj instanceof Cytokine.VEGF) {
        numberOfVEGF = numberOfVEGF + 1;
    }
}

// Adding cytokines to the environment if the maximum is
not reached

    int cartilageVolume = 1000000; // defines the cartilage
volume in um3 depending on the cell density (10000 cells/mm3)
    double startExpressing = 0.01; // defines the expression
threshold for which a node starts releasing soluble mediators

    for (int i = 0 ; i < NumOfNodes; i++) {

        if (i == 2 && X[i] > startExpressing &&
numberOfBMP2 <= (int) Math.round(0.0169302*cartilageVolume)){

            int halfLifeBMP2 = 2;
            int diffusionBMP2 = 4730;
            int numberOfMolecules = (int)
(Math.ceil(X[i]*10.917946));

            for(int a = 0; a < numberOfMolecules;
a++) {
                BMP2 bmp2 = new BMP2(space,
grid, halfLifeBMP2, diffusionBMP2);

                context.add(bmp2);
                double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                double yDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                double zDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                double randomNumber1 =
Math.random();
                double randomNumber2 =
Math.random();

                if(randomNumber1 < 1/2) {
                    if(randomNumber2 < 1/3) {
                        xDisplacement = -
(auxRadius + 1);
                    } else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
                        yDisplacement = -
(auxRadius + 1);
                    } else {
                        zDisplacement = -
(auxRadius + 1);
                    }
                } else {
                    if(randomNumber2 < 1/3) {

```



```

                                yDisplacement =
auxRadius + 1;                                } else {
                                                zDisplacement =
auxRadius + 1;                                }
                                                }
                                                space.moveTo(iffng, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
NdPoint newpt1 =
space.getLocation(iffng);
                                grid.moveTo(iffng, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
                                }

                                } else if (i == 7 && X[i] > startExpressing &&
numberOfIGF1 <= (int) Math.round(0.0346511*cartilageVolume)){

                                int halfLifeIGF1 = 660;
                                int diffusionIGF1 = 5330;
                                int numberOfMolecules = (int)
(Math.ceil(X[i]*0.3724295));
                                for(int a = 0; a < numberOfMolecules;
a++) {
                                IGF1 igf1 = new IGF1(space,
grid, halfLifeIGF1, diffusionIGF1);
                                context.add(igf1);
                                double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                                double yDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                                double zDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                                double randomNumber1 =
Math.random();
                                double randomNumber2 =
Math.random();

                                if(randomNumber1 < 1/2) {
                                    if(randomNumber2 < 1/3) {
                                        xDisplacement = -
(auxRadius + 1);
                                    } else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
                                        yDisplacement = -
(auxRadius + 1);
                                    } else {
                                        zDisplacement = -
(auxRadius + 1);
                                    }
                                } else {
                                    if(randomNumber2 < 1/3) {
                                        xDisplacement =
(auxRadius + 1);
                                    } else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
                                        yDisplacement =
(auxRadius + 1);
                                    } else {

```



```

        space.moveTo(il10, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
        NdPoint newpt1 =
space.getLocation(il10);
        grid.moveTo(il10, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
    }
        } else if (i == 9 && X[i] > startExpressing &&
numberOfIL13 < (int) Math.round(0.0334644*cartilageVolume)){
        int halfLifeIL13 = 240;
        int diffusionIL13 = 5630;
        int numberOfMolecules = (int)
(Math.ceil(X[i]*257.7226385));
        for(int a = 0; a < numberOfMolecules;
a++) {
            IL13 il13 = new IL13(space,
grid, halfLifeIL13, diffusionIL13);
            context.add(il13);
            double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            double yDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            double zDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            double randomNumber1 =
Math.random();
            double randomNumber2 =
Math.random();
            if(randomNumber1 < 1/2) {
                if(randomNumber2 < 1/3) {
                    xDisplacement = -
(auxRadius + 1);
                } else if(randomNumber2
                >= 1/3 && randomNumber2 < 2/3) {
                    yDisplacement = -
(auxRadius + 1);
                } else {
                    zDisplacement = -
(auxRadius + 1);
                }
            } else {
                if(randomNumber2 < 1/3) {
                    xDisplacement =
auxRadius + 1;
                } else if(randomNumber2
                >= 1/3 && randomNumber2 < 2/3) {
                    yDisplacement =
auxRadius + 1;
                } else {
                    zDisplacement =
auxRadius + 1;
                }
            }
            space.moveTo(il13, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
            NdPoint newpt1 =
space.getLocation(il13);

```

```

        grid.moveTo(il13, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
    }

    } else if (i == 10 && X[i] > startExpressing &&
numberOfIL17 <= (int) Math.round(0.0012593*cartilageVolume)){

        int halfLifeIL17 = 480;
        int diffusionIL17 = 5530;
        int numberOfMolecules = (int)
(Math.ceil(X[i]*1.6182335));
        for(int a = 0; a < numberOfMolecules;
a++) {
            IL17 il17 = new IL17(space,
grid, halfLifeIL17, diffusionIL17);
            context.add(il17);
            double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            double yDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            double zDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            double randomNumber1 =
Math.random();
            double randomNumber2 =
Math.random();
            if(randomNumber1 < 1/2) {
                if(randomNumber2 < 1/3) {
                    xDisplacement = -
(auxRadius + 1);
                } else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
                    yDisplacement = -
(auxRadius + 1);
                } else {
                    zDisplacement = -
(auxRadius + 1);
                }
            } else {
                if(randomNumber2 < 1/3) {
                    xDisplacement =
auxRadius + 1;
                } else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
                    yDisplacement =
auxRadius + 1;
                } else {
                    zDisplacement =
auxRadius + 1;
                }
            }
            space.moveTo(il17, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
            NdPoint newpt1 =
space.getLocation(il17);
            grid.moveTo(il17, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
        }
    }
}

```

```

        } else if (i == 11 && X[i] > startExpressing &&
numberOfIL18 <= (int) Math.round(0.0090279*cartilageVolume)){

        int halfLifeIL18 = 1560;
        int diffusionIL18 = 5310;
        int numberOfMolecules = (int)

(Math.ceil(X[i]*0.9582027));

        for(int a = 0; a < numberOfMolecules;
a++) {
            IL18 il18 = new IL18(space,
            context.add(il18);
            double xDisplacement =
            double yDisplacement =
            double zDisplacement =
            double randomNumber1 =
            double randomNumber2 =

            if(randomNumber1 < 1/2) {
                if(randomNumber2 < 1/3) {
                    xDisplacement = -
                } else if(randomNumber2
                    yDisplacement = -
                } else {
                    zDisplacement = -
                }
            } else {
                if(randomNumber2 < 1/3) {
                    xDisplacement =
                } else if(randomNumber2
                    yDisplacement =
                } else {
                    zDisplacement =
                }
            }
            space.moveTo(il18, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
            NdPoint newpt1 =
            grid.moveTo(il18, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
        }

        } else if (i == 12 && X[i] > startExpressing &&
numberOfIL1b < (int) Math.round(0.0004955*cartilageVolume)){

        int halfLifeIL1b = 20;

```

```

(Math.ceil(X[i]*0.033294523));
a++) {
grid, halfLifeIL1b, diffusionIL1b);
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
Math.random();
Math.random();

(auxRadius + 1);
>= 1/3 && randomNumber2 < 2/3) {
(auxRadius + 1);

(auxRadius + 1);

auxRadius + 1;
>= 1/3 && randomNumber2 < 2/3) {
auxRadius + 1;

auxRadius + 1;

xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
space.getLocation(il1b);
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
}

} else if (i == 13 && X[i] > startExpressing &&
numberOfIL4 < (int) Math.round(0.0000757398*cartilageVolume)){

(int) halfLifeIL4 = 19;
(int) diffusionIL4 = 5530;
(int) numberOfMolecules = (int)
(Math.ceil(X[i]*0.002868931));
int diffusionIL1b = 5040;
int numberOfMolecules = (int)
for(int a = 0; a < numberOfMolecules;

IL1b il1b = new IL1b(space,
context.add(il1b);
double xDisplacement =
double yDisplacement =
double zDisplacement =
double randomNumber1 =

double randomNumber2 =

if(randomNumber1 < 1/2) {
if(randomNumber2 < 1/3) {
xDisplacement = -
} else if(randomNumber2
yDisplacement = -
} else {
zDisplacement = -
}
} else {
if(randomNumber2 < 1/3) {
xDisplacement =
} else if(randomNumber2
yDisplacement =
} else {
zDisplacement =
}
}
space.moveTo(il1b, pt1.getX() +
NdPoint newpt1 =
grid.moveTo(il1b, (int)
}
}

```

```

a++) {
    halfLifeIL4, diffusionIL4);

    ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
    ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
    ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
    Math.random();
    Math.random();

    (auxRadius + 1);
    >= 1/3 && randomNumber2 < 2/3) {
    (auxRadius + 1);
    (auxRadius + 1);

    auxRadius + 1;
    >= 1/3 && randomNumber2 < 2/3) {
    auxRadius + 1;
    auxRadius + 1;

    auxRadius + 1;
    xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
    space.getLocation(il4);
    newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
    }

    } else if (i == 14 && X[i] > startExpressing &&
    numberOfIL6 < (int) Math.round(0.0411394*cartilageVolume)){

    int halfLifeIL6 = 10;
    int diffusionIL6 = 5260;
    int numberOfMolecules = (int)

    (Math.ceil(X[i]*0.101454205));
    a++) {
    halfLifeIL6, diffusionIL6);

```

```

for(int a = 0; a < numberOfMolecules;

```

```

    IL4 il4 = new IL4(space, grid,

```

```

    context.add(il4);

```

```

    double xDisplacement =

```

```

    double yDisplacement =

```

```

    double zDisplacement =

```

```

    double randomNumber1 =

```

```

    double randomNumber2 =

```

```

    if(randomNumber1 < 1/2) {

```

```

        if(randomNumber2 < 1/3) {

```

```

            xDisplacement = -

```

```

        } else if(randomNumber2

```

```

            yDisplacement = -

```

```

        } else {

```

```

            zDisplacement = -

```

```

        }

```

```

    } else {

```

```

        if(randomNumber2 < 1/3) {

```

```

            xDisplacement =

```

```

        } else if(randomNumber2

```

```

            yDisplacement =

```

```

        } else {

```

```

            zDisplacement =

```

```

        }

```

```

    } else {

```

```

        if(randomNumber2 < 1/3) {

```

```

            xDisplacement =

```

```

        } else if(randomNumber2

```

```

            yDisplacement =

```

```

        } else {

```

```

            zDisplacement =

```

```

        }

```

```

    }

```

```

    space.moveTo(il4, pt1.getX() +

```

```

    xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);

```

```

    NdPoint newpt1 =

```

```

    grid.moveTo(il4, (int)

```

```

    newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());

```

```

    }

```

```

    } else if (i == 14 && X[i] > startExpressing &&

```

```

    numberOfIL6 < (int) Math.round(0.0411394*cartilageVolume)){

```

```

    int halfLifeIL6 = 10;

```

```

    int diffusionIL6 = 5260;

```

```

    int numberOfMolecules = (int)

```

```

    (Math.ceil(X[i]*0.101454205));

```

```

    a++) {

```

```

    halfLifeIL6, diffusionIL6);

```

```


```

```


```

```


```

```


```

```


```

```


```

```

        context.add(il6);
        double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
        double yDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
        double zDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
        double randomNumber1 =
Math.random();
        double randomNumber2 =
Math.random();
        if(randomNumber1 < 1/2) {
            if(randomNumber2 < 1/3) {
                xDisplacement = -
            } else if(randomNumber2
                yDisplacement = -
            } else {
                zDisplacement = -
            }
        } else {
            if(randomNumber2 < 1/3) {
                xDisplacement =
            } else if(randomNumber2
                yDisplacement =
            } else {
                zDisplacement =
            }
        }
        space.moveTo(il6, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
        NdPoint newpt1 =
space.getLocation(il6);
        grid.moveTo(il6, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
    }
    } else if (i == 15 && X[i] > startExpressing &&
numberOfIL8 < (int) Math.round(0.8953235*cartilageVolume)){
        int halfLifeIL8 = 240;
        int diffusionIL8 = 5970;
        int numberOfMolecules = (int)
(Math.ceil(X[i]*120.9591067));
        for(int a = 0; a < numberOfMolecules;
a++) {
            IL8 il8 = new IL8(space, grid,
halfLifeIL8, diffusionIL8);
            context.add(il8);
            double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);

```

```

ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
Math.random();
Math.random();

(double) yDisplacement =
(double) zDisplacement =
(double) randomNumber1 =

(double) randomNumber2 =

if(randomNumber1 < 1/2) {
    if(randomNumber2 < 1/3) {
        xDisplacement = -
    } else if(randomNumber2
        yDisplacement = -
    } else {
        zDisplacement = -
    }
} else {
    if(randomNumber2 < 1/3) {
        xDisplacement =
    } else if(randomNumber2
        yDisplacement =
    } else {
        zDisplacement =
    }
}

space.moveTo(il8, pt1.getX() +
NdPoint newpt1 =
grid.moveTo(il8, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
}

} else if (i == 16 && X[i] > startExpressing &&
numberOfLIF < (int) Math.round(0.1062196*cartilageVolume)){

int halfLifeLIF = 180;
int diffusionLIF = 5330;
int numberOfMolecules = (int)

for(int a = 0; a < numberOfMolecules;

LIF lif = new LIF(space, grid,

context.add(lif);
(double) xDisplacement =
(double) yDisplacement =
(double) zDisplacement =

ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);

```



```

    (auxRadius + 1);
    >= 1/3 && randomNumber2 < 2/3) {
    (auxRadius + 1);
    (auxRadius + 1);

    auxRadius + 1;
    >= 1/3 && randomNumber2 < 2/3) {
    auxRadius + 1;
    auxRadius + 1;
    auxRadius + 1;

    xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
    NdPoint newpt1 =
    space.getLocation(no);
    newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
    }

    } else if (i == 22 && X[i] > startExpressing &&
    numberOfPEG2 <= (int) Math.round(2.1476017*cartilageVolume)){

    int halfLifePEG2 = 10;
    int diffusionPEG2 = 10600;
    int numberOfMolecules = (int)
    (Math.ceil(X[i]*23.082389));
    a++) {
    grid, halfLifePEG2, diffusionPEG2);

    ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
    ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
    ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
    Math.random();
    Math.random();

    (auxRadius + 1);

    if(randomNumber1 < 1/2) {
        if(randomNumber2 < 1/3) {
            xDisplacement = -
        } else if(randomNumber2
            yDisplacement = -
        } else {
            zDisplacement = -
        }
    } else {
        if(randomNumber2 < 1/3) {
            xDisplacement =
        } else if(randomNumber2
            yDisplacement =
        } else {
            zDisplacement =
        }
    }
    space.moveTo(no, pt1.getX() +
    NdPoint newpt1 =
    grid.moveTo(no, (int)
    newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
    }
    } else if (i == 22 && X[i] > startExpressing &&
    numberOfPEG2 <= (int) Math.round(2.1476017*cartilageVolume)){
        int halfLifePEG2 = 10;
        int diffusionPEG2 = 10600;
        int numberOfMolecules = (int)
        (Math.ceil(X[i]*23.082389));
        a++) {
            grid, halfLifePEG2, diffusionPEG2);

            ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
            Math.random();
            Math.random();

            (auxRadius + 1);

            if(randomNumber1 < 1/2) {
                if(randomNumber2 < 1/3) {
                    xDisplacement = -
                } else if(randomNumber2
                    yDisplacement = -
                } else {
                    zDisplacement = -
                }
            } else {
                if(randomNumber2 < 1/3) {
                    xDisplacement =
                } else if(randomNumber2
                    yDisplacement =
                } else {
                    zDisplacement =
                }
            }
            space.moveTo(no, pt1.getX() +
            NdPoint newpt1 =
            grid.moveTo(no, (int)
            newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
            }
        }
    }

```

```

} else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
    yDisplacement = -
(auxRadius + 1);
} else {
    zDisplacement = -
(auxRadius + 1);
}
} else {
    if(randomNumber2 < 1/3) {
        xDisplacement =
(auxRadius + 1);
    } else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
        yDisplacement =
(auxRadius + 1);
    } else {
        zDisplacement =
(auxRadius + 1);
    }
}
space.moveTo(peg2, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
NdPoint newpt1 =
space.getLocation(peg2);
grid.moveTo(peg2, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
}
} else if (i == 24 && X[i] > startExpressing &&
numberOfTGFb <= (int) Math.round(0.0170681*cartilageVolume)){
    int halfLifeTGFb = 50;
    int diffusionTGFb = 4740;
    int numberOfMolecules = (int)
(Math.ceil(X[i]*0.1834472));
    for(int a = 0; a < numberOfMolecules;
a++) {
        TGFb tgfb = new TGFb(space,
grid, halfLifeTGFb, diffusionTGFb);
        context.add(tgfb);
        double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
        double yDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
        double zDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
        double randomNumber1 =
Math.random();
        double randomNumber2 =
Math.random();
        if(randomNumber1 < 1/2) {
            if(randomNumber2 < 1/3) {
                xDisplacement = -
(auxRadius + 1);
            } else if(randomNumber2
>= 1/3 && randomNumber2 < 2/3) {
                yDisplacement = -
(auxRadius + 1);
            }
        }
    }
}
}

```



```

    } else {
        if(randomNumber2 < 1/3) {
            xDisplacement =
auxRadius + 1;
        } else if(randomNumber2
            yDisplacement =
auxRadius + 1;
        } else {
            zDisplacement =
auxRadius + 1;
        }
    }
    space.moveTo(tnfa, pt1.getX() +
xDisplacement, pt1.getY() + yDisplacement, pt1.getZ() + zDisplacement);
    NdPoint newpt1 =
    space.getLocation(tnfa);
    grid.moveTo(tnfa, (int)
newpt1.getX(), (int) newpt1.getY(), (int) newpt1.getZ());
    }
        } else if (i == 27 && X[i] > startExpressing &&
numberOfVEGF < (int) Math.round(5.1931455*cartilageVolume)){
            int halfLifeVEGF = 60;
            int diffusionVEGF = 5150;
            int numberOfMolecules = (int)
(Math.ceil(X[i]*38.41413357));
            a++) {
                VEGF vegf = new VEGF(space,
grid, halfLifeVEGF, diffusionVEGF);
                context.add(vegf);
                double xDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                double yDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                double zDisplacement =
ThreadLocalRandom.current().nextDouble(-(auxRadius + 1), auxRadius + 1);
                double randomNumber1 =
Math.random();
                double randomNumber2 =
Math.random();
                if(randomNumber1 < 1/2) {
                    if(randomNumber2 < 1/3) {
                        xDisplacement = -
                    } else if(randomNumber2
                        yDisplacement = -
                    } else {
                        zDisplacement = -
                    }
                } else {
                    if(randomNumber2 < 1/3) {
                        xDisplacement =
auxRadius + 1;
                    }
                }
            }
        }
    }
}

```



```

*****
*****
    */

    public static double[] f_i (double[] X_i, double[][] Mact, double[][]
Minh, int h, int NumOfNodes) {

        double[] w = new double[NumOfNodes]; // total input of a node,
one per node = vector.
        double[] f = new double[NumOfNodes]; // the derivative (one
derivative for each node in the network).
        double[] Ract = new double[NumOfNodes]; // It will change for
each node, the combination of activating links for a node.
        double[] RinH = new double[NumOfNodes]; // It will change for
each node, the combination of inhibiting links for a node.
        int[] gamma = new int[NumOfNodes];

        Arrays.fill(gamma,1);

        /*

*****
        *** CALCULATING THE WEIGHTS EQUIVALENT TO THE TOTAL INPUT FOR
EACH NODE ***
*****
        */

        for (int i = 0; i < NumOfNodes; i++) {
            // Summing the interactions that arrive at each
node (1),
            // extracting row by row (Ract ant RinH) (2) and
// summing the activators at sumRact/RinH (3).
            Ract = Mact[i];
            double sumAlfa = 0;
            for (int j = 0 ; j <
Ract.length; j++) { // sum Ract
                sumAlfa = sumAlfa +
Ract[j];
            }
            double sumAlphaX = 0;

            RinH=Minh[i];
            double sumBeta = 0;
            for (int j = 0 ; j <
RinH.length; j++) { // sum RinH
                sumBeta = sumBeta +
RinH[j];
            }
            double sumBetaX = 0;

            // Calculate the weight (depending on having
activators, inhibitors, both or none).
            // Node i has no inhibitors and at least 1
activator
            if (sumBeta == 0 && sumAlfa > 0) {
                // Calculate the sum
alpha(i)*x(i) by looping through each element of Ract

```

```

                                for (int k = 0 ; k <
Ract.length; k++) {
                                sumAlphaX = sumAlphaX +
Ract[k]*(X_i[k]);
                                }
                                // Calculate the weight
                                w[i] =
((1+sumAlfa)/(sumAlfa))*((sumAlphaX)/(1+sumAlphaX));
                                // Node i has no activators and at least 1
inhibitor
                                } else if (sumBeta > 0 && sumAlfa==0)
{
                                // Calculate the sum
beta(i)*x(i) by looping through each element of RinH
                                for (int k = 0 ; k <
Rinh.length; k++) {
                                sumBetaX =
sumBetaX + Rinh[k]*(X_i[k]);
                                }
                                // Calculate the weight
                                w[i]=1-
((1+sumBeta)/(sumBeta))*((sumBetaX)/(1+sumBetaX));
                                // Node i has at least one activator and one
inhibitor
                                } else if (sumBeta > 0 && sumAlfa > 0)
{
                                // Calculate the sum
alpha(i)*X(i) and beta(i)*X(i) by looping through each element of Ract and
Rinh
                                for (int k = 0 ; k <
Ract.length; k++) {
                                sumAlphaX=sumAlphaX
+ Ract[k]*(X_i[k]);
                                }
                                for (int k = 0 ; k <
Rinh.length; k++) {
                                sumBetaX=sumBetaX + Rinh[k]*(X_i[k]);
                                }
                                // Calculate the weight
                                w[i]=(((1+sumAlfa)/sumAlfa)*(sumAlphaX/(1+sumAlphaX)))*(1-
((1+sumBeta)/(sumBeta))*((sumBetaX)/(1+sumBetaX)));
                                // Node i has no activators nor inhibitors
                                } else {
                                w[i]=0;
                                }
                                // Compute the derivative for node i
                                f[i]=DELTA*((-Math.exp(0.5*h)+Math.exp(-
h*(w[i]-0.5)))/((1-Math.exp(0.5*h))*(1+Math.exp(-h*(w[i]-0.5))))-
gamma[i]*(X_i[i]));

```

```

        }
        return f; // a vector with the computed derivatives of each
node
    }

/**
 * *****
 *
 * Reading the baseline of the .csv file and putting it into a string
 *
 * *****
 */

public static double[] readBaseline(String pathBaseline, int
NumOfNodes) throws IOException {

    double[] baseline = new double[NumOfNodes];
    String line = "";
    BufferedReader br = new BufferedReader(new
FileReader(pathBaseline));
    ArrayList<String> baselineList = new ArrayList<String>();

    while((line = br.readLine())!=null) {

        // Read each line of the network.
        baselineList.add(line);

    }

    baselineList.remove(0);

    for(int i = 0; i < NumOfNodes; i++) {
        baseline[i] = Double.parseDouble(baselineList.get(i));
    }

    br.close();
    return baseline;
}

/**
 * *****
 *
 * Reading the nodes of the .csv file and putting them in a string *
 * *****
 */

public static ArrayList<String> readNodeNames(String path) throws
IOException {

    String line = "";
    BufferedReader br = new BufferedReader(new FileReader(path));
    ArrayList<String> dataList = new ArrayList<String>();
    ArrayList<String> NodeNames = new ArrayList<String>();

    while((line = br.readLine())!=null) {

```

```

        // Read each line of the network.
        dataList.add(line);

        // Extract information of each column.
        String[] columns = line.split(";");

        // In the first column there are the nodes (the names).
        NodeNames.add(columns[0]);
    }

    br.close();

    // Removing the first line, where there are the categories of
each column.
        NodeNames.remove(0);

    return NodeNames;
}

/**
*****
*****
* Transform the activating links into a binary matrix of dimensions
NumOfNodes x NumOfNodes *
*****
*****
*/

    public static double[][] readActivators(String path, int NumOfNodes)
throws IOException {
        double [][] activatorsMatrix = new double
[NumOfNodes][NumOfNodes];
        String line= "";
        BufferedReader br = new BufferedReader(new FileReader(path));
        ArrayList<String> dataList = new ArrayList<String>();
        ArrayList<String> NodeNames = new ArrayList<String>();
        ArrayList<String> Activators = new ArrayList<String>();

        while((line = br.readLine())!=null) {
            // Read each line of the network.
            dataList.add(line);

            // Extract information of each column.
            String[] columns = line.split(";");
            // In the first column there are the nodes.
            NodeNames.add(columns[0]);
            // In the second there are the activating interactions for
the node
                Activators.add(columns[1]);
        }

        br.close();

```

```

        // Removing the first line, where there are the categories of
each column.
        NodeNames.remove(0);
        Activators.remove(0);

        // Transform the network into a binary system, where there will
be two matrices NumOfNodes dimensional containing the activating and
inhibiting links.

        for (int i = 0; i < NumOfNodes; i++) {

            // For activators
            String node = Activators.get(i);
            String[] arr = node.split(",");
            double[] PresentInNode = new double[NumOfNodes];

            for (int j = 0; j < arr.length; j++) {
                for (int k = 0; k < NodeNames.size();
k++){
                    if
(arr[j].equals(NodeNames.get(k))) {
                        PresentInNode[k] = 1;
                    }
                }
            }
            activatorsMatrix[i] = PresentInNode;
        }
        return activatorsMatrix;
    }

    /**
    *****
    *****
    * Transform the inhibiting links into a binary matrix of dimensions
NumOfNodes x NumOfNodes *
    *****
    *****
    **/

    public static double[][] readInhibitors(String path, int NumOfNodes)
throws IOException {
        double [][] inhibitorsMatrix = new double
[NumOfNodes][NumOfNodes];
        String line= "";
        BufferedReader br = new BufferedReader(new FileReader(path));
        ArrayList<String> dataList = new ArrayList<String>();
        ArrayList<String> NodeNames = new ArrayList<String>();
        ArrayList<String> Inhibitors = new ArrayList<String>();

        while((line = br.readLine())!=null) {
            // Read each line of the network.
            dataList.add(line);
            // Extract information of each column.
            String[] columns = line.split(";");
            // In the first column there are the nodes.
            NodeNames.add(columns[0]);
            // In the second there are the activating interactions for
the node

```

```

        Inhibitors.add(columns[2]);
    }

    br.close();

    // Removing the first line, where there are the categories of
each column.
        NodeNames.remove(0);
        Inhibitors.remove(0);

    // Transform the network into a binary system, where there will
be two matrices NumOfNodes dimensional containing the activating and
inhibiting links.

        for (int i = 0; i < NumOfNodes; i++) {

            // For inhibitors
            String nodeInh = Inhibitors.get(i);
            String[] arrInh = nodeInh.split(",");
            double[] PresentInNodeInh = new double[NumOfNodes];

                for (int j = 0; j < arrInh.length; j++) {
                    for (int k = 0; k < NodeNames.size();
k++){
                        if
(arrInh[j].equals(NodeNames.get(k))) {
                            PresentInNodeInh[k] = 1;
                        }
                    }
                }
            inhibitorsMatrix[i] = PresentInNodeInh;
        }
    return inhibitorsMatrix;
}

/**
*****
* Extract the number of nodes of from the csv file. *
*****
**/

public static int extractNumOfNodes (String path) throws IOException {
    int numNodes = 0;
    String line= "";
    BufferedReader br = new BufferedReader(new FileReader(path));
    ArrayList<String> dataList = new ArrayList<String>();

    while((line = br.readLine())!=null) {
        // Read each line of the network.
        dataList.add(line);
        numNodes++;
    }
    br.close();

    // Removing the first line, where there are the categories of
each column.
        numNodes = numNodes - 1;

```

```

        return numNodes;
    }

    /**
     * Get the final steady states by name
     */
    public static HashMap<String,Double>
    getFinalSteadyStates(ArrayList<String> NodeNames, double[] X){

        HashMap<String,Double> SteadyStates = new
        HashMap<String,Double>();

        for (int i = 0 ; i<NumOfNodes; i++) {
            SteadyStates.put(NodeNames.get(i), X[i]);
        }

        return SteadyStates;
    }
}

```

Soluble mediator agent

```
package kneeOA_V1;

import repast.simphony.context.Context;
import repast.simphony.engine.schedule.ScheduledMethod;
import repast.simphony.random.RandomHelper;
import repast.simphony.space.continuous.ContinuousSpace;
import repast.simphony.space.continuous.NdPoint;
import repast.simphony.space.grid.Grid;
import repast.simphony.util.ContextUtils;

public class Cytokine {

    // Declare global class variables
    private ContinuousSpace<Object> space;
    private Grid<Object> grid;
    private int halfLife;
    private int diffusion;

    // Initialize global class variables
    public Cytokine(ContinuousSpace<Object> space, Grid<Object> grid, int
halfLife, int diffusion) {
        this.space = space;
        this.grid = grid;
        this.halfLife = halfLife;
        this.diffusion = diffusion;
    }

    // Get the kind of class of the cytokine
    public String whatIsTheClass() {
        String typeOfClass = this.getClass().getSimpleName();
        return typeOfClass;
    }

    /*
    *****
    *   Make the cytokines move through the space with a given half-life
    *
    *                               @param pt
    *
    *****
    */

    // Random movement of the cytokines through the environment
    public void randomWalk() {
        space.moveByDisplacement(this, RandomHelper.nextDoubleFromTo(-
0.005,0.005), RandomHelper.nextDoubleFromTo(-0.005, 0.005),
RandomHelper.nextDoubleFromTo(-0.005, 0.005));
        NdPoint myPoint = space.getLocation(this);
        grid.moveTo(this, (int)myPoint.getX(), (int)myPoint.getY(),
(int) myPoint.getZ());
    }

    // Death of a cytokine (it removes the cytokine from the environment
when its half-life is reached)
```

```

public void die() {
    Context<?> context = ContextUtils.getContext(this);
    context.remove(this);
}

// Move through the space and die in a given time
@ScheduledMethod(start = 1, interval = 1, shuffle = true)
public void step() {
//    NdPoint pt = space.getLocation(this);
    if (halfLife > 0) {
        for(int a = 0; a < (int) Math.round(diffusion/0.005); a++)
        {
            randomWalk();
        }
        halfLife--;

//        // Uncomment the following lines to activate physiological
//        boundary conditions of the total thickness of the cartilage
//        if((int)pt.getY() < 1) {
//            die();
//        } else if((int)pt.getY() >= 99) {
//            space.moveByDisplacement(this, 0, -0.5, 0);
//            NdPoint newPoint = space.getLocation(this);
//            grid.moveTo(this, (int)newPoint.getX(),
//            (int)newPoint.getY(), (int)newPoint.getZ());
//        }
        } else {
            die();
        }
    }
}

/*****
*
*           Create subclasses for each cytokine
*
*****/

public static class BMP2 extends Cytokine {
    public BMP2(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
        super(space, grid, halfLife, diffusion);
    }
}

public static class IFNG extends Cytokine {
    public IFNG(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
        super(space, grid, halfLife, diffusion);
    }
}

public static class IGF1 extends Cytokine {
    public IGF1(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
        super(space, grid, halfLife, diffusion);
    }
}

```

```

    public static class IL10 extends Cytokine {
        public IL10(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class IL13 extends Cytokine {
        public IL13(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class IL17 extends Cytokine {
        public IL17(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class IL18 extends Cytokine {
        public IL18(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class IL1b extends Cytokine {
        public IL1b(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class IL4 extends Cytokine {
        public IL4(ContinuousSpace<Object> space, Grid<Object> grid, int
halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class IL6 extends Cytokine {
        public IL6(ContinuousSpace<Object> space, Grid<Object> grid, int
halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class IL8 extends Cytokine {
        public IL8(ContinuousSpace<Object> space, Grid<Object> grid, int
halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class LIF extends Cytokine {

```

```

        public LIF(ContinuousSpace<Object> space, Grid<Object> grid, int
halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class NO extends Cytokine {
        public NO(ContinuousSpace<Object> space, Grid<Object> grid, int
halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class PEG2 extends Cytokine {
        public PEG2(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class TGFb extends Cytokine {
        public TGFb(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class TNFa extends Cytokine {
        public TNFa(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }

    public static class VEGF extends Cytokine {
        public VEGF(ContinuousSpace<Object> space, Grid<Object> grid,
int halfLife, int diffusion) {
            super(space, grid, halfLife, diffusion);
        }
    }
}

```

Builder

```
import java.io.IOException;

import repast.simphony.context.Context;
import repast.simphony.context.space.continuous.ContinuousSpaceFactory;
import repast.simphony.context.space.continuous.ContinuousSpaceFactoryFinder;
import repast.simphony.context.space.grid.GridFactory;
import repast.simphony.context.space.grid.GridFactoryFinder;
import repast.simphony.dataLoader.ContextBuilder;
import repast.simphony.engine.environment.RunEnvironment;
import repast.simphony.parameter.Parameters;
import repast.simphony.space.continuous.ContinuousSpace;
import repast.simphony.space.continuous.NdPoint;
import repast.simphony.space.continuous.RandomCartesianAdder;
import repast.simphony.space.grid.Grid;
import repast.simphony.space.grid.GridBuilderParameters;
import repast.simphony.space.grid.SimpleGridAdder;
import repast.simphony.space.grid.WrapAroundBorders;

import kneeOA_V1.Cytokine.BMP2;
import kneeOA_V1.Cytokine.IFNG;
import kneeOA_V1.Cytokine.IGF1;
import kneeOA_V1.Cytokine.IL1b;
import kneeOA_V1.Cytokine.IL4;
import kneeOA_V1.Cytokine.IL6;
import kneeOA_V1.Cytokine.IL8;
import kneeOA_V1.Cytokine.IL10;
import kneeOA_V1.Cytokine.IL13;
import kneeOA_V1.Cytokine.IL17;
import kneeOA_V1.Cytokine.IL18;
import kneeOA_V1.Cytokine.LIF;
import kneeOA_V1.Cytokine.NO;
import kneeOA_V1.Cytokine.PEG2;
import kneeOA_V1.Cytokine.TGFb;
import kneeOA_V1.Cytokine.TNFa;
import kneeOA_V1.Cytokine.VEGF;

public class kneeOABuilder implements ContextBuilder<Object> {

    static Parameters params = null;

    @Override
    public Context<Object> build(Context<Object> context) {

        context.setId("kneeOA_V1");

        params = RunEnvironment.getInstance().getParameters();

        ContinuousSpaceFactory spaceFactory =
ContinuousSpaceFactoryFinder.createContinuousSpaceFactory(null);
        ContinuousSpace<Object> space =
spaceFactory.createContinuousSpace("space", context, new
RandomCartesianAdder<Object>(), new
repast.simphony.space.continuous.WrapAroundBorders(), 100, 100, 100);

        GridFactory gridFactory =
GridFactoryFinder.createGridFactory(null);
```

```

Grid<Object> grid = gridFactory.createGrid("grid", context, new
GridBuilderParameters<Object>(new WrapAroundBorders(), new
SimpleGridAdder<Object>(), true, 100, 100, 100));

```

```

    int chondrocyteCount = (Integer)
params.getValue("chondrocyteCount");
    int BMP2Count = (Integer) params.getValue("initalBMP2Count");
    int IFNGCount = (Integer) params.getValue("initalIFNGCount");
    int IGF1Count = (Integer) params.getValue("initalIGF1Count");
    int IL1BCount = (Integer) params.getValue("initalIL1BCount");
    int IL4Count = (Integer) params.getValue("initalIL4Count");
    int IL6Count = (Integer) params.getValue("initalIL6Count");
    int IL8Count = (Integer) params.getValue("initalIL8Count");
    int IL10Count = (Integer) params.getValue("initalIL10Count");
    int IL13Count = (Integer) params.getValue("initalIL13Count");
    int IL17Count = (Integer) params.getValue("initalIL17Count");
    int IL18Count = (Integer) params.getValue("initalIL18Count");
    int LIFCount = (Integer) params.getValue("initalLIFCount");
    int NOCount = (Integer) params.getValue("initalNOCount");
    int PEG2Count = (Integer) params.getValue("initalPEG2Count");
    int TGFbCount = (Integer) params.getValue("initalTGFbCount");
    int TNFaCount = (Integer) params.getValue("initalTNFaCount");
    int VEGFCount = (Integer) params.getValue("initalVEGFCount");

    for(int i = 0; i < chondrocyteCount; i++) {
        try {
            context.add(new Chondrocyte(space, grid));
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

    for(int i = 0; i < BMP2Count; i++) {
        context.add(new BMP2(space, grid, 7, 4730));
    }

    for(int i = 0; i < IFNGCount; i++) {
        context.add(new IFNG(space, grid, 36, 5440));
    }

    for(int i = 0; i < IGF1Count; i++) {
        context.add(new IGF1(space, grid, 660, 5330));
    }

    for(int i = 0; i < IL1BCount; i++) {
        context.add(new IL1b(space, grid, 20, 5040));
    }

    for(int i = 0; i < IL4Count; i++) {
        context.add(new IL4(space, grid, 19, 5530));
    }

    for(int i = 0; i < IL6Count; i++) {
        context.add(new IL6(space, grid, 10, 5260));
    }

    for(int i = 0; i < IL8Count; i++) {
        context.add(new IL8(space, grid, 240, 5970));
    }

```

```

    }

    for(int i = 0; i < IL10Count; i++) {
        context.add(new IL10(space, grid, 52, 5390));
    }

    for(int i = 0; i < IL13Count; i++) {
        context.add(new IL13(space, grid, 240, 5630));
    }

    for(int i = 0; i < IL17Count; i++) {
        context.add(new IL17(space, grid, 480, 5530));
    }

    for(int i = 0; i < IL18Count; i++) {
        context.add(new IL18(space, grid, 1560, 5310));
    }

    for(int i = 0; i < LIFCount; i++) {
        context.add(new LIF(space, grid, 180, 5330));
    }

    for(int i = 0; i < NOCount; i++) {
        context.add(new NO(space, grid, 70, 16000));
    }

    for(int i = 0; i < PEG2Count; i++) {
        context.add(new PEG2(space, grid, 10, 10600));
    }

    for(int i = 0; i < TGFbCount; i++) {
        context.add(new TGFb(space, grid, 50, 4740));
    }

    for(int i = 0; i < TNFaCount; i++) {
        context.add(new TNFa(space, grid, 76, 5190));
    }

    for(int i = 0; i < VEGFCount; i++) {
        context.add(new VEGF(space, grid, 60, 5150));
    }

    /* ***** *
    * Randomly distributed chondrocytes *
    * ***** */
    // for(Object obj : context.getObjects(Object.class)) {
    //     NdPoint pt = space.getLocation(obj);
    //     grid.moveTo(obj, (int)pt.getX(), (int)pt.getY(),
    (int)pt.getZ());
    // }

    /*
    ***** *
    * Uniformly distributed chondrocytes and randomly added
    cytokines *
    *
    ***** */

```

```

// For a 100.000 um3 volume (1 chondrocyte)
// for(Object obj : context) {
//     if (obj instanceof Chondrocyte) {
//         space.moveTo(obj, 50, 50, 50);
//     }
//     NdPoint pt = space.getLocation(obj);
//     grid.moveTo(obj, (int)pt.getX(), (int)pt.getY(),
// (int)pt.getZ());
//     }
//

// For a 300.000 um3 volume (3 chondrocytes)
// int placedChondrocytes = 0;
// for(Object obj : context) {
//     if (obj instanceof Chondrocyte) {
//         if (placedChondrocytes == 0) {
//             space.moveTo(obj, 30, 30, 30);
//         } else if (placedChondrocytes == 1) {
//             space.moveTo(obj, 50, 70, 50);
//         } else if (placedChondrocytes == 2) {
//             space.moveTo(obj, 70, 30, 70);
//         }
//         placedChondrocytes++;
//     }
//     NdPoint pt = space.getLocation(obj);
//     grid.moveTo(obj, (int)pt.getX(), (int)pt.getY(),
// (int)pt.getZ());
//     }
//

// For a 1.000.000 um3 volume (10 chondrocytes)
int placedChondrocytes = 0;
for(Object obj : context) {
    if (obj instanceof Chondrocyte) {
        if (placedChondrocytes == 0) {
            space.moveTo(obj, 13, 13, 13);
        } else if (placedChondrocytes == 1) {
            space.moveTo(obj, 13, 13, 87);
        } else if (placedChondrocytes == 2) {
            space.moveTo(obj, 87, 13, 87);
        } else if (placedChondrocytes == 3) {
            space.moveTo(obj, 87, 13, 13);
        } else if (placedChondrocytes == 4) {
            space.moveTo(obj, 13, 87, 13);
        } else if (placedChondrocytes == 5) {
            space.moveTo(obj, 13, 87, 87);
        } else if (placedChondrocytes == 6) {
            space.moveTo(obj, 87, 87, 87);
        } else if (placedChondrocytes == 7) {
            space.moveTo(obj, 87, 87, 13);
        } else if (placedChondrocytes == 8) {
            space.moveTo(obj, 39, 50, 50);
        } else if (placedChondrocytes == 9) {
            space.moveTo(obj, 65, 50, 50);
        }
        placedChondrocytes++;
    }
}

```

```
        NdPoint pt = space.getLocation(obj);
        grid.moveTo(obj, (int)pt.getX(), (int)pt.getY(),
(int)pt.getZ());
    }
    return context;
}
```

Parameters

```
<?xml version="1.0" encoding="UTF-8" ?>
<parameters>

  <parameter name="randomSeed" displayName="Default Random Seed"
type="int" defaultValue="__NULL__" />

  <parameter name = "chondrocyteCount"
displayName = "Number of chondrocytes"
type = "int"
defaultValue = "1"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

  <parameter name = "initalBMP2Count"
displayName = "Initial number of BMP2"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

  <parameter name = "initalIFNGCount"
displayName = "Initial number of IFNg"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

  <parameter name = "initalIGF1Count"
displayName = "Initial number of IGF1"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

  <parameter name = "initalIL1BCount"
displayName = "Initial number of IL1b"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

  <parameter name = "initalIL4Count"
displayName = "Initial number of IL4"
type = "int"
defaultValue = "0"
isReadOnly = "false"
```

```
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLIL6Count"
displayName = "Initial number of IL6"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLIL8Count"
displayName = "Initial number of IL8"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLIL10Count"
displayName = "Initial number of IL10"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLIL13Count"
displayName = "Initial number of IL13"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLIL17Count"
displayName = "Initial number of IL17"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLIL18Count"
displayName = "Initial number of IL18"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>
```

```

<parameter name = "initaLLIFCount"
displayName = "Initial number of LIF"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLNOCount"
displayName = "Initial number of NO"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLPEG2Count"
displayName = "Initial number of PEG2"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLTGFbCount"
displayName = "Initial number of TGFb"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLTNFaCount"
displayName = "Initial number of TNFa"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

<parameter name = "initaLVEGFCount"
displayName = "Initial number of VEGF"
type = "int"
defaultValue = "0"
isReadOnly = "false"
converter =
"repast.simphony.parameter.StringConverterFactory$IntConverter"
/>

</parameters>

```

Context

```
<context id="knee0A_V1"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  xsi:noNamespaceSchemaLocation="http://repast.org/scenario/context">  
  
  <projection type = "continuous space" id = "space"/>  
  <projection type = "grid" id = "grid"/>  
</context>
```