

**Universidade do Minho**
Escola de Engenharia

Diogo Lopes

# Numerical Surgery
# Modeling, Simulation and Visualization in real time

Outubro de 2018

**Numerical Surgery**
**Modeling, Simulation and Visualization in real time**
Diogo Lopes
Uminho | 2018



**Universidade do Minho**
Escola de Engenharia

Diogo Lopes

# Numerical Surgery
# Modeling, Simulation and Visualization in real time

PhD dissertation
PhD Degree in Informatics

Dissertation supervised by
**António Ramires Fernandes**
**Stéphane Clain**

Outubro de 2018

# Statement of Integrity

I hereby declare having conductedd my thesis with integrity. I confirm that I have not used plagiarism or any form of falsification of results in the process of the thesis elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the Universidado do Minho.



# ACKNOWLEDGEMENTS

I would like to thank my advisors, Professor António Ramires and Professor Stéphane Clain for their support, motivation, capacity of interpreting my ideas and sharing their knowledge with me. They helped me writing this thesis as well as refining my skills as a researcher.

I also want to give a special thank to my family for their support and patience throughout these years. They have helped me overcome the various obstacles that emerged along the way, specially the ones created by my capacity of over-thinking and over-complicating things. I also want to thank them for all the opportunities they gave since I was born.

I wish to express my gratitude to Dra. Augusta Cardoso and the Hospital de Gaia for providing support and material that was key for this research project. I also want to thank Professor Gueorgui Smirnov and Igor Vasilevskiy for helping me processing the medical information and for sharing with me the knowledge they have gathered in the subject of breast surgeries.

I am grateful for the conditions provided by Departamento de Informática of Universidade do Minho and the assistance that the ALGORITMI and Mathematics research centres gave me. Another special thanks to Professor Rui Pereira and Professor Gaspar Machado for all the fruitful discussions and for their cooperation with this research project.

I also want to thank IPCA for welcoming me and allowing me to teach and share my knowledge with the students. By working with them, I was able to fund my research and complete this project.

Finally, I would like to thank all my friends for their support, with a special reference to Sindel and C-16 for reminding me that there are more things in this life besides the PhD and that sometimes resting can help any research project in the long run.

To everyone, thank you very much.

## ABSTRACT

**Numerical Surgery: Modeling, Simulation and Visualization in Real Time**

Numerical Surgery aims at simulating surgery procedures with a high degree of applicability to real situations. It gathers techniques from several scientific areas such as mathematics, physics, computer science and medicine. The use of computer simulation programs based on biomechanical models enables to better understand the influence of preoperative key parameters on the post-surgical results of, for instance, breast surgeries. Simulation of different preoperative markings of the breast could prevent errors during the intervention planning or correct the existent ones when the desired result was not achieved.

IN the last three decades, many biomechanical models of the breast were presented. They intend to determine the shape, size and behaviour of the tissues when dealing with external perturbations. Most of the models used data from medical equipment like CT scans, MRI and ultrasounds but require expensive and limited equipment to obtain the breast parameters.

The new proposed breast model carries out the numerical simulation by only using pre-surgical measurements done by surgeons instead of data obtained from complex medical equipment. Despite the strong assumptions stated, namely the approximately spherical shape of the breast in a state of free stress and the new knitting conditions, the models discussed provide an efficient and simple breast model that reproduces faithfully the behaviour of the breast tissues.

Parameter identifications such as the mechanical coefficients of the breast turn to be a crucial goal since their values dramatically change from one patient to another. They have to be provided for each patient in order to obtain a suitable simulation. Hence, along with the new breast model, a new iterative method to estimate the breast geometrical and mechanical parameters was developed. Just like the breast model, this method estimates these parameters as a function of the measurements performed by surgeons during consultations.

Numerical methods and more specific parameter identification requires important computational resources and time that ranges between one minute to several hours, depending on the underlying mesh and the accuracy. A neural network approach is presented in order to offer an alternative way to efficiently assess the mechanical and geometrical parameter of the breast model by returning a very good approximation in real time.

Considering the iterative method and neural networks strengths and weaknesses, it was proposed a hybrid model that combines these two methods. This new approach aims at

providing both fast and robust solutions to solve the problem of estimating geometrical and mechanical parameters of the breast.

Simple access and friendly user tools are necessary to enable the surgeon to adopt and effectively use numerical tools. A Web support simulator for breast knitting has been developed and experimented by surgeons that, besides providing real-life cases for testing, also validated the data. Very good predictions of the simulator with real cases are reported which encourage future researches in breast reduction and reconstruction.

# RESUMO

**Cirurgia Numérica: Modelação, Simulação e Visualização em Tempo Real**

Cirurgia Numérica é uma área que reúne técnicas de diversas áreas científicas, como matemática, física, informática e medicina. O uso de programas de simulação baseados em modelos biomecânicos permitem uma melhor compreensão da influência dos parâmetros pré-operatórios nos resultados pós-cirúrgicos, como no caso de cirurgias mamárias. A simulação de diferentes procedimentos pré-operatórios da mama ajudam a prevenir erros durante o planeamento da intervenção ou a corrigir problemas quando o resultado desejado não é alcançado. Ao longo dos anos, muitos modelos biomecânicos da mama foram apresentados que determinam a forma, tamanho e comportamento dos tecidos. A maioria dos modelos usava dados de equipamentos médicos, como tomografia computadorizada, ressonância magnética e ultrassonografias. O problema com a maioria dos métodos usados para obter os parâmetros da mama está no uso deste tipo de equipamentos que são caros e limitados.

É apresentado um novo modelo de mama que usa medidas pré-cirúrgicas feitas pelos cirurgiões para criar a mama em vez de dados provenientes de equipamentos médicos. A simplicidade do modelo e as suposições feitas, ou seja, a forma esférica aproximada da mama num estado livre de stress e as novas condições de sutura fornecem uma boa abordagem para o desenvolvimento de um modelo simples da mama que reproduz o comportamento dos tecidos mamários. A identificação dos parâmetros da mama, torna-se um objetivo crucial, uma vez que seus valores mudam com a pessoa e precisam ser fornecidos para cada paciente, a fim de obter uma simulação mais adequada. Assim, juntamente com o novo modelo mamário, foi desenvolvido um novo método para estimar os parâmetros da mama. Assim como o modelo mamário, este método estima estes parâmetros em função das medidas realizadas pelos médicos durante consultas.

O método apresentado requer bastantes recursos computacionais assim como tempo que pode variar entre um minuto e várias horas, dependendo da qualidade da malha e da precisão requerida. Uma abordagem envolvendo redes neuronais é apresentada para oferecer uma alternativa de estimação dos parâmetros mecânicos e geométricos da mama, obtendo assim uma boa aproximação em tempo real.

Considerando os pontos fortes e fracos do método iterativo assim como das redes neuronais, um modelo híbrido capaz de combiná-los é apresentado. Este novo método tem

como objectivo criar uma solução rápida e robusta de resolver o problema de estimar os parâmetros geométricos e mecânicos da mama.

Uma plataforma Web para simulação de cirurgias de redução mamárias foi desenvolvido e experimentado por cirurgiões que, para além de fornecerem casos da vida real para testes, também validaram os dados. Resultados com casos reais mostram um bom desempenho pelo modelo de mama assim como pelo método de estimação de parâmetros e que por isso encorajam pesquisas futuras.

# CONTENTS


1 INTRODUCTION 1
- 1.1 Motivation 2
- 1.2 Objectives 4
- 1.3 Contributions 4
- 1.4 Document Structure 5

2 NUMERICAL SURGERY 7
- 2.1 Physiology 7
- 2.2 Soft Tissue Modelling 10
  - 2.2.1 Breast geometrical representation 11
  - 2.2.2 Soft Tissue Mechanics 12
  - 2.2.3 Numerical Methods 20
- 2.3 State of The Art 21
  - 2.3.1 Breast Physiology 22
  - 2.3.2 Breast Models 23
- 2.4 Discussion 32

3 BREAST MODEL 34
- 3.1 Numerical Model 34
  - 3.1.1 Geometry 35
  - 3.1.2 Mechanics 37
  - 3.1.3 Boundary Conditions 38
- 3.2 Discretisation of the Model 39
  - 3.2.1 Mesh and Notations 39
  - 3.2.2 Boundary and Suturing Conditions 40
  - 3.2.3 Strain Energy 41
  - 3.2.4 Minimisation algorithm 42
- 3.3 Breast Under Gravity 43
  - 3.3.1 Geometry 43
  - 3.3.2 Mechanics 44
  - 3.3.3 Boundary Conditions 44
- 3.4 Discretisation of the breast under gravity 45
  - 3.4.1 Boundary, Chassaignac and sticking conditions 45
  - 3.4.2 Discretisation of the functional 46
- 3.5 Numerical Tests 46
  - 3.5.1 Symmetric suturing 47

3.5.2 Asymmetric suturing 49
3.5.3 Frontal and rear suturing 50
3.5.4 Comparison with a real surgery case 51
3.6 Discussion 51
4 PARAMETERS IDENTIFICATION 54
4.1 Parameter Identification 54
4.1.1 Manual Methods 54
4.1.2 Medical Imaging 55
4.1.3 Cameras and Scanners 57
4.1.4 Alternative Method 59
4.2 Parameter Estimation 61
4.2.1 Breast Modelling 62
4.2.2 Measurements 64
4.2.3 Parameters identification 65
4.3 Problem Discretisation 68
4.3.1 Numerical approximation to the discrete energy minimiser 68
4.3.2 Cost function discretisation 69
4.4 Proof of Concept 70
4.4.1 Feasibility 71
4.4.2 Sensitivity Analysis 74
4.4.3 Robustness 77
4.5 Performance Evaluation 80
4.5.1 Optimisation experiments 83
4.6 Discussion 84
5 CODE ARCHITECTURE PROFILING 86
5.1 Computational Architecture 86
5.1.1 Simulation Data Structures 86
5.1.2 Code system and map $g$ 88
5.1.3 Simulation Steps 89
5.2 Estimation Method Architecture 92
5.2.1 Estimation Data structures 92
5.2.2 Computational sub-stages 94
5.3 Profiling Analysis 96
5.3.1 Time Consumption and Instructions per Cycle 96
5.3.2 Memory and Cache 98
5.3.3 Arithmetic efficiency and Intensity 100
5.3.4 Arithmetic efficiency and Bandwidth 102
5.3.5 Profiling Analysis 103

5.4 New Code Architecture 105
5.4.1 Code Structure analysis 105
5.4.2 New Structure profiling 107
5.4.3 Improved Algorithms 112
5.5 Parallelism 115
5.5.1 Vectorisation 116
5.5.2 CPU Parallelism 121
5.5.3 GPU Parallelism 123
5.6 Discussion 135
6 MACHINE LEARNING MODEL 136
6.1 Regression Models Review 136
6.1.1 Linear Regression 137
6.1.2 Support Vector Machine 138
6.1.3 Neural networks 139
6.2 Network Architecture and Training 141
6.2.1 Datasets generation 141
6.2.2 Neural Network architecture 143
6.2.3 Hidden layers 144
6.2.4 Activation functions and optimiser 145
6.3 Testing 146
6.3.1 Mesh size impact 146
6.3.2 Error assessments 148
6.3.3 Estimation Robustness 151
6.4 Discussion 152
7 HYBRID MODEL 154
7.1 Neural Network and Iterative Method 154
7.1.1 Performance Comparison 155
7.2 Hybrid Method 156
7.2.1 Parameter Estimation 156
7.2.2 Estimation Analysis 158
7.3 Discussion 161
8 REAL APPLICATIONS WITH A WEB PLATFORM 163
8.1 Surgery Simulation of Real Life Cases 163
8.1.1 Surgery Characteristics 164
8.1.2 Simulation and Evaluation 165
8.2 Web Platform 171
8.2.1 Objectives 172
8.2.2 Platform Structure 173

8.2.3 Workflow 178
8.3 Conclusions 179
9 CONCLUSION 181
9.1 Future Work 183

# LIST OF FIGURES


Figure 1 Viscoelastic response of soft tissues. On the left it shows the shift of the stress-strain response curves (orange) according to the strain-rate (strain of the material with respect to time). On the right the stress-strain curve of soft tissues with 4 regions: toe, heel, linear, plastic. 9

Figure 2 Normal Stress ($\sigma$) and Shear Stress ($\tau$) resulting from the superficial tension $\vec{F}$ 14

Figure 3 Example of a boundary conditions scheme of a model. In this example, the blue line is restricted to not move, the red line is forced to move to the centre grey line, and the green line if free to move according to displacements happening with the neighbour lines. 19

Figure 4 Anatomy of the breast. (Encyclopaedia Britannica) 22

Figure 5 Illustration of how a material can expand in a perpendicular direction $\vec{sd}$ of an applied compression force $\vec{p}$ 26

Figure 6 Breast reduction surgery planning marks on a patient. Photography taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 36

Figure 7 Scheme of breast reduction surgery: the middle panel is the frontal view while the left and right panels report the parameters for the left and right parts of the breast 36

Figure 8 Full breast (left) and half breast (right) in a 3D frontal view (top row) and rear view (bottom row) 37

Figure 9 Incised breast geometry $\Omega_{cut}$ with the incision parameters. On the left is the breast frontal view while on the right is the breast side view. 39

Figure 10 Discretisation of function $g$. For any point $P_n \in \gamma_R$, one determines an approximation $g(P_n) \in \gamma_L$ between two nodes $P_\alpha$, $P_\beta$ of an edge belonging to $\gamma_L$. Barycentric coordinates then derive from the relative position of $g(P_n)$. 41

Figure 11 3D rear views for the gravity problem: full breast (left) and half breast (right). 43

Figure 12 Mesh of the pre-operative breast: frontal view (left) and lateral view (right). 47

Figure 13 Mesh of the breast after a symmetric incision: frontal view (left) and lateral view (right). 47
Figure 14 Shape of the breast after the suturing without gravity field: frontal view (left) and lateral view (right). 48
Figure 15 The breast under the gravity field: frontal view (left) and lateral view (right). 48
Figure 16 Mesh of the breast after an asymmetric incision: frontal view (left) and lateral view (right).. 49
Figure 17 Mesh of the breast after suturing: frontal view (left) and lateral view (right). 49
Figure 18 The breast under the gravity field: frontal view (left) and lateral view (right). 50
Figure 19 Mesh of the breast after a symmetric front incision and a rear incision: frontal view (left) and lateral view (right). 51
Figure 20 Breast shape after the suturing process: frontal view (left) and lateral view (right). 51
Figure 21 The suturing breast in the gravity field: frontal view (left) and lateral view (right). 52
Figure 22 Comparison between a real surgery operation and the simulation. Before the surgery (left) one reproduces the initial breast with an identical volume. After the surgery (right), one carries out the simulation with the cuts made by the surgeon. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 52
Figure 23 Breast self-examination (Wikipedia) 55
Figure 24 Epipolar Geometry (by Arne Nordmann - wikipedia) 58
Figure 25 3D object reconstruction from depth maps and point clouds (by Arsi Warrior - wikipedia) 59
Figure 26 Breast geometry configurations. Radius $R$ characterised the main sphere while $H$ identifies the truncated part following the $y$ direction. 62
Figure 27 Notation and geometry of the gravity-free geometry (left) and configuration in the gravity field (right) 63
Figure 28 Markings on a patient in two positions: standing up on the left; bend over approximately $45^{o}$ on the right. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 64

Figure 29 Size characteristic ($min_y$, $max_y$ and $min_z$) of a breast (stand-up position) 66
Figure 30 Examples of a breast configuration using different granularity: coarse (left), medium (middle) and thin (right) 70
Figure 31 Parameter error evolution of the optimisation process with a coarse mesh (top left), a medium mesh (top right) and a thin mesh (bottom). Evolution is shown in the absolute percentage difference between the current values of the parameters in the iterations (from 0 to 25) and the reference parameters. 82
Figure 32 Breast model code architecture data structures. 87
Figure 33 Flowchart of the simulation 92
Figure 34 Diagram of the data structures used to perform the parameter estimation including a new structure (estimator in blue). 93
Figure 35 Call Graph of the simulation with percentage of, total and self, time consumed (clock cycles spend on each method) 97
Figure 36 New code architecture data structures (new elements in blue). 106
Figure 37 Call graph representing the weight of each major method called during the simulation with a new data structure. 108
Figure 38 New callgraph representing the weight of each major method called during the simulation after the implementation of a new algorithm on the line_search method. 113
Figure 39 New callgraph representing the weight of each major method called during the simulation after implementing SIMD intrinsics. 117
Figure 40 GPU architecture (Owens et al. doi:10.1109/JPROC.2008.917757) 124
Figure 41 GPU data structures types. On the left its is the one dimensional buffer while on the right there is the composite buffer with fixed dimensions and that can have meaningful values (x) or empty values (-) 127
Figure 42 Nau3D simulation diagram 130
Figure 43 Graph which compares the time scalability of the CPU the GPU codes. 134
Figure 44 Image of 3 breast meshes generated by the same parameters: coarse (left), thin (mid) and ultra-thin (right) 142
Figure 45 Breast reduction surgery planning marks on a patient. Usage of rulers to be able to obtain the measures from photographs. Photography taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 164

Figure 46 Scheme of the breast with the incision parameters and the points used to generate the breast mesh. On the left is the breast frontal view while on the right is the breast side view. 165
Figure 47 Picture of the real breast with the markings and the scheme values represented. Photography taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 166
Figure 48 Breast Geometry skeleton made by the points obtained with the markings done by the surgeons on the patient's breasts. 167
Figure 49 Visual Comparison of a patient's breast before the surgery with the digital model. Digital model on top of the patient's right side breast. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 167
Figure 50 Visual Comparison of a patient's breast 18 months after the surgery with the digital model. Digital model on top of the patient's right side breast. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 168
Figure 51 Visual Comparison of a patient's breast 18 months after the surgery with the digital model. Digital model on top of the patient's right side breast. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 169
Figure 52 Visual Comparison of a patient's breast before the surgery with the digital model with the estimated parameters. Digital model on top of the patient's right side breast. Pictures of the patient standing up (left) and bending forward 45 degrees (right). Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 171
Figure 53 Visual Comparison of a patient's breast with the digital model with the estimated parameters. Digital model on top of the patient's right side breast. Pictures of the patient standing up with a frontal (left) and lateral view (right) taken a short period after the surgery. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia. 172
Figure 54 Web platform interface. 173
Figure 55 Web platform breast geometry interface inputs. 175
Figure 56 Web platform breast incision interface inputs. 175
Figure 57 Web platform breast mechanic interface inputs. 176
Figure 58 Web platform breast gravity interface inputs. 176

Figure 59 Web platform rendering of the pre-surgical (left) and incision (right) breast meshes. 177
Figure 60 Web platform rendering of the sutured breast (left) and the post-surgery breast (right). 177
Figure 61 Web platform text helper. 178
Figure 62 Web platform user instructions. 179
Figure 63 Web platform cut diagram. 179
Figure 64 Web platform simulating the surgical procedure. 180

# LIST OF TABLES


Table 1 Comparison between the 3 types of techniques used to generate the geometry of the breast in terms to the complexity of the breast reconstruction (Low, Medium or High), if they are capable of giving information about the internal arrangement of the breast tissues (Yes or No) and if they obtain detail of the breast contours or they obtain an approximation 25

Table 2 Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method with initial guesses which differ the reference parameters up to 10%. Mean values of the error per parameter, the number of iterations necessary to converge and the value of the cost function that measures the difference between the reference measurements and the measurements obtained with the estimated parameters. 72

Table 3 Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method with initial guesses which differ the reference parameters up to 30%. Mean values of the error per parameter, the number of iterations necessary to converge and the value of the cost function that measures the difference between the reference measurements and the measurements obtained with the estimated parameters. 73

Table 4 Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method with initial guesses which differ the reference parameters up to 60%. Mean values of the error per parameter, the number of iterations necessary to converge and the value of the cost function that measures the difference between the reference measurements and the measurements obtained with the estimated parameters. 73

Table 5 Table with the sensibility analysis of the parameters. (Red - unreliable values, yellow - slightly reliable values, green - reliable values) 76

Table 6 Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method using the same initial guess ($\Lambda^{mean}$) which differ the reference parameters up to 10%. 78

Table 7 Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method using the same initial guess ($\Lambda^{mean}$) which differ the reference parameters up to 30%. 79
Table 8 Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method using the same initial guess ($\Lambda^{mean}$) which differ the reference parameters up to 50%. 79
Table 9 Differences in percentage between the measurements obtained with a coarse and thin mesh (T-C) as well as with a medium and thin mesh (T-M). 81
Table 10 Measurement's difference residual between the reference measurements and the measurements obtained with different sized meshes with the same values for the breast parameters $\Lambda$ and the work effort which is spent by the optimisation process. 82
Table 11 Mean value of the cost function 83
Table 12 IPC of the simulation methods with different size meshes 98
Table 13 L3 miss percentage of 3 different sized meshes 98
Table 14 L1 Load miss percentage of 3 different sized meshes 99
Table 15 L1 Store miss percentage of 3 different sized meshes 100
Table 16 Assembly instructions of the face and cell evaluate energy methods. For the face method, the table divides the values by the Chassaignac and skin calculations while the cell method only has one way to calculate the strain-energy. The table is divided in floating operation instructions (addsd, subsd, mulsd, divsd and sqrtsd) and data access instructions (movsd) 101
Table 17 Arithmetic intensity of the energy evaluation methods (cells, Chassaignac and skin faces). Ratio between the number floating point operations (adds, muls, divs and sqrts) and bytes loaded (number of loads times size of double ) 101
Table 18 Arithmetic intensity of the energy evaluation methods (cells, Chassaignac and skin faces) based on number of cycles. Ratio between the number of cycles spent on performing calculations (AM and DS) and loads (L). 102
Table 19 Bandwidth measured for different sizes (32KB, 256KB, 3MB and 500MB), order of access (sequential or random) and bytes loaded at a time (8B and 16B) 102
Table 20 Comparison between the number of doubles loaded in each methods vs the maximum number that could be loaded if the bandwidth was fully utilised. 103

Table 21 Comparison between old and new solution. This table shows the percentage reduction of the measured events from the old code to the new code. 108
Table 22 IPC of the simulation methods with different sized meshes 109
Table 23 L3 cache miss % of the simulation methods with different sized meshes. Cells with '-' mean that the values were smaller than 0.01%109
Table 24 L1 cache load miss % of the simulation methods with different sized meshes 110
Table 25 L1 cache store miss % of the simulation methods with different sized meshes 110
Table 26 Arithmetic intensity of the energy evaluation methods (cells, Chassaignac and skin faces) based on number of cycles. Ratio between the number of cycles spent on performing calculations (AM and DS) and loads (L). 110
Table 27 Comparison between the number of doubles loaded in each methods vs the maximum number that could be loaded if the bandwidth was fully utilised. 111
Table 28 Comparison of the time (in seconds) consumption using the previous code version (base code architecture) and the new version (new code architecture) as well as the speedup obtained for three different mesh sizes. 111
Table 29 Percentage Reduction of cycles and instructions of the code's architecture with the new **line_search** method 113
Table 30 New values of IPC, L3 miss %, L1 load and store miss % of the new line_search method for different sized meshes 114
Table 31 Comparison of the time (in seconds) consumption using the previous code version (new code architecture) and the improved algorithms version (improved line_search) as well as the speedup obtained for three different mesh sizes. 114
Table 32 Reduction in percentage of cycles and instructions after the implementation of SIMD intrinsic functions to replace parts of the **calc_energy_chass**, **calc_energy_skin** and **calc_energy_cell** methods. Negative values represent a percentage increase in comparison to the previous reported values. 117
Table 33 New values of IPC for the simulation methods with different sized meshes 118
Table 34 New values of L3 miss % for the simulation methods with different sized meshes 119

Table 35 New values of L1 Load miss % for the simulation methods with different sized meshes 119
Table 36 New values of L1 Store miss % for the simulation methods with different sized meshes 119
Table 37 Times in the different machines of the simulation method with simd (simd) and without simd (base) instructions 120
Table 38 Simulation time spent for every test machine using multiple treads (1, 2, 4, 8, 16 and 32). Cells with "-" mean that the test was not performed because the number of threads exceed the number of cores (including hyper-threading technology). 123
Table 39 Simulation weight (in ms per frame) for each shader that runs during the simulation. The CPU column shows the time spent in the CPU by the shader. The GPU column shows the time spent in the GPU by the shader. The Total column shows the time spent by the shader during the entire simulation. The bottom rows show the total time spent per frame in CPU and GPU as well as the percentages. 131
Table 40 Simulation weight (in ms per frame) for each shader that runs during the simulation after the first improvements. The CPU column shows the time spent in the CPU by the shader. The GPU column shows the time spent in the GPU by the shader. The Total column shows the time spent by the shader during the entire simulation. The bottom rows show the total time spent per frame in CPU and GPU as well as the percentages. 132
Table 41 Simulation weight for each shader that runs during the simulation for meshes with different number of nodes (coarse, medium and thin). The last row shows the total time in seconds spent by the simulation. 133
Table 42 Times of the simulations with different size meshes for a different number of threads perf thread group. 133
Table 43 Values of loss in percentage for the model training data-set (train) and validation data-set (val). These values represent the mean values of error for 5 runs on training and validating running up to 3000 epochs each. The left columns shows the number of layers, and the number of nodes per layer. For each hidden layer structure, it is shown the difference between the estimated parameters and the reference parameters in relation to the training and validation stages. The last column shows the average difference per parameter. 145

Table 44 Values of error in percentage for the model training data-set (train) and validation data-set (val). These values represent the mean values of error for 5 runs on training and validating up to 3000 epochs each. The left column shows the different optimisers and activation functions. For each combination it is shown the error of the training and validation related to each parameter and for the average total (sum of all values of error and divided by the number of parameters). 146

Table 45 Values of loss for the model training and validating. These values represent the mean values of error for 5 runs on training and validating up to 3000 epochs each. On the left column there are the different size meshes. For each breast mesh, it is shown the loss of the training and validating related to each parameter and for the mean total (sum of all values of loss and divided by the number of parameters). 147

Table 46 Mean value error in % (normalised data) between the reference parameters and the estimated parameters by the neural network model. 149

Table 47 Mean value error in % between the reference and estimated measurements for each mesh. Last lower column shows the mean value of the cost function. 150

Table 48 Mean value of the error in (%) between the reference parameters and the approximation obtained by the neural network model trained with three different data-sets associated to three different meshes. 152

Table 49 Comparison of the two parameter estimations methods: Neural Network (NN) and the Iterative Method (IM). Mean difference in percentage between the reference parameters and the estimated parameters using three mesh sizes: Coarse, Medium and Thin. Time spent by each method in average and the number of iterations (cycles) required by the iterative method to converge and produce a good result. 155

Table 50 Neural Network model parameter estimation deviation groups for each mesh size (Coarse, Medium, Thin): Accurate Estimation; Close Estimation; Medium Far Estimation; Far. Average number of iterations copied from tables 6, 7 and 8 of section 4.4.3 and time in minutes required by the Iterative method to converge and find an accurate solution is calculated from the values of those tables. 157

Table 51 Error (%) between the reference parameters and the calculated parameters for the poor neural network (NN) coarse model estimations. Comparison of the approximation of the NN, with the iterative method (IM) and with the hybrid method (NN+IM). It is shown the average error per parameter (avg total) as well as the difference between the estimated measurements and the reference measurements. The last column show the number of iterations made by IM in both IM model and the Hybrid model. 159

Table 52 Error (%) between the reference parameters and the calculated parameters for the poor neural network (NN) medium model estimations. Comparison of the approximation of the NN, with the iterative method (IM) and with the hybrid method (NN+IM). It is shown the average error per parameter (avg total) as well as the difference between the estimated measurements and the reference measurements. The last column show the number of iterations made by IM in both IM model and the Hybrid model. 160

Table 53 Error (%) between the reference parameters and the calculated parameters for the poor neural network (NN) thin model estimations. Comparison of the approximation of the NN, with the iterative method (IM) and with the hybrid method (NN+IM). It is shown the average error per parameter (avg total) as well as the difference between the estimated measurements and the reference measurements. The last column show the number of iterations made by IM in both IM model and the Hybrid model. 161

Table 54 Estimated parameters by the hybrid model considering three mesh sizes (coarse, medium and thin) and the differences between the estimated measurements and the reference measurements (Cost Function) as well as the number of iterations necessary to produce an accurate result. 170

# GLOSSARY

**AI** Artificial Intelligence.
**ANN** Artificial Neural Network.
**AoS** Array of Structures.
**API** Application Programming Interface.
**APIs** Application Programming Interfaces.

**BEM** Boundary Element Modelling.

**CNN** Convolutional Neural Network.
**CPU** Central Processor Unit.
**CSS** Cascading Style Sheets.
**CT** Computed Tomography.
**CUDA** Compute Unified Device Architecture.

**elu** Exponential Linear Unit.

**FCT** Fundação para a Ciência e Tecnologia.
**FDM** Finite Difference Method.
**FEM** Finite Element Method.

**GLSL** OpenGL Shading Language.
**GPGPU** General Purpose Graphic Processing Units.
**GPU** Graphics Processor Unit.

**HTML** Hypertext Markup Language.

**INF** Infinit.
**INRIA** Inventeurs du monde numérique.
**IPC** Instructions per Cycle.

**ML** Machine Learning.
**MLP** Multi-Layer Perceptron.
**MPa** Mega Pascal.
**MRI** Magnetic resonance imaging.
**MSE** Mean Squared Error.

**NaN** Not a Number.

**NN** Neural Network.

**PCA** Principal Component Analysis.
**PHP** Hypertext Preprocessor.

**RAM** Random Access Memory.
**ReLu** Rectified Linear Unit.
**RNN** Recurrent Neural Network.

**SGD** Stochastic Gradient Descent.
**SIFT** Scale Invariant Feature Transform.
**SIMD** Single Instruction Multiple Data.
**SIMT** Single Instruction Multiple Threads.
**SM** Streaming Multiprocessors.
**SoA** Structure of Arrays.
**SVM** Support Vector Machines.
**SVR** Support Vector Regression.

**XML** Extensible Markup Language.

# 1

# INTRODUCTION

Numerical Surgery is an area that gathers techniques from several scientific fields such as mathematics, physics, computer science and medicine. This knowledge is used to create tools to assist doctors/surgeons in the treatment of their patients, for example, simulating surgical procedures (Delingette and Ayache (2004)). In 2012 breast cancer was the most frequently diagnosed cancer in women worldwide and the second most common cancer overall (Ferlay et al. (2014)). However, it is not the only disease associated with the breast. In fact, the majority of the lesions that occur in the breast (for example fibrocystic changes and macromastia) are benign (Guray and Sahin (2006)). Some other problems which are connected to orthopaedic issues (posture change and lumbar lordosis), as well as psychological issues (eating disorders), can also arise from breasts with large cup sizes and asymmetries (Findikcioglu et al. (2007); Nuzzi et al. (2014)).

The treatment for some of the diseases mentioned previously involves surgical procedures. Some of them, like breast augmentation, focus on aesthetics (Swanson (2017)), others like breast reductions and mastectomies focus on health and comfort (Foreman et al. (2009); Lapid et al. (2013)). In any one of these cases the surgical procedure results depend on several factors such as the patient's age, breast tissue properties (amount of fat and glandular tissues, tendons or muscles for instance), the surgical technique(s) and quality of the surgeon(s) (Oliveira et al. (2013); Scott et al. (2005); Cardoso et al. (2013b)).

Many studies were conducted in order to help surgeons performing these surgeries (Azar et al. (2002); Delingette and Ayache (2004); del Palomar et al. (2008); Han et al. (2012a); Cardoso et al. (2013a); Eder et al. (2014); Eiben et al. (2016)). They lead to the design and creation of breast models with geometrical parameters (shape and size) and mechanical characteristics (viscosity and elasticity). These properties of the breast tissues are crucial to evaluate the behaviour of the breast, namely to predict the aesthetic outcome of a surgery. Some of these studies focused on helping live surgeries (Delingette and Ayache (2004)), others produced numerical models of the breast by gathering and analysing data for the physical and mechanical properties from medical exams like CT scans, MRI and x-rays (Azar et al. (2002); del Palomar et al. (2008); Han et al. (2012a); Eder et al. (2014); Eiben et al. (2016)). With the appearance of new visual technologies like cameras, sensors and Kinect, new methods of creating digital representations and obtaining physical properties

of the breast emerged (Henseler et al. (2014, 2016)). The problem with most of the methods used to obtain the breast parameters lies in the use of expensive equipment whose usage depends on those resources availability which in countries like Portugal can mean waiting up to 6 months (Maia (2016); Carriço (2017)). Another aspect which has been disregarded with the evolution of technology is the medical experience and techniques that surgeons use to plan and perform their operations. It seems that it would be a waste not to use all the information they gather during all stages of a surgery (cuts, sutures and also measurements performed in the planning and post-operative stage).

But how can this information be used to create accurate models of the breast? How can these models be used to simulate surgical procedures with fast and accurate responses?

To address such issues, a study of new methods to estimate breast parameters and simulate real procedures is conducted. It involves the inclusion of algorithms which have the capacity to process a large amount of data in a short amount of time. These algorithms rely on the usage of both accelerating hardware like the GPU or vectorisation units and the training of models using deep learning algorithms.

## 1.1 MOTIVATION

The quest for better treatments is an ongoing effort for any health professional. The technology evolution and the cumulative experiences over time are two keys factors for that goal. While new equipment makes possible large data records about the patient, experience helps the doctors to efficiently process information and decide what is the best treatment to undertake. Surgery simulators and numerical models of the breast are numerical tools that can support surgeons planning surgeries and with that reduce the probability of making mistakes.

Many of the breast models that exist use the data provided by CT scans and MRI to build the breast mesh and assess its behaviour (Azar et al. (2002); Rajagopal et al. (2008)). The methods used to create the digital breast use the collected data and perform a series of steps which include segmentation, classification, mesh generation and material properties assignment in order to create a Finite Element Method (FEM) breast (Azar et al. (2002); Rajagopal et al. (2008)). The digital breast encompasses the major structural elements, (i.e.), the fat, glandular tissue, Cooper ligaments and other tissues that compose the breast are modelled and their properties are set. Such detailed and complexity descriptions require large computational resources but do not necessarily lead to much more accuracy than simpler models (del Palomar et al. (2008)).

For routinely surgical procedures like a breast reduction surgery, there often is no necessity to use expensive imaging machines, and so, when planning such surgical procedure, a surgeon performs a set of measurements and marks the breast to indicate where to perform

the incisions. The more experienced a surgeon is the more confident and accurate will be the final results of that surgery. In any surgery there is the possibility of something go wrong, but simulators can help reduce such eventuality, and therefore it is important to use the information that the surgeons collect in this planning stage, in particular, the measurements they perform on the patients, to create a model of the breast which can simulate relevant surgical procedures. Whereas the common procedures require data from imaging equipment to provide the biomechanical parameters of the breast (Azar et al. (2002); del Palomar et al. (2008); Rajagopal et al. (2008); Han et al. (2012a)), a new method is proposed that only uses the simple measurements that the doctor realises on the patients.

The complexity of the FEM models require a large number of computation resources for achieving representative surgical simulation (Delingette and Ayache (2004)). Depending on the number of unknowns (degree of freedom) of the FEM model and on the hardware resources, simulations could take several days to complete or even weeks (Delingette and Ayache (2004); Hopp et al. (2013); Martínez-Martínez et al. (2017); Liu et al. (2017)). With the new computational power that modern Central Processor Unit (CPU) and Graphics Processor Unit (GPU) have, these computational expensive algorithms now take less time to complete. New software takes advantage of modern hardware like SOFA and NiftySim (Talbot et al. (2015); Johnsesn et al. (2015)). The interest in having faster simulations is that a surgeon can run multiple scenarios and better prepare the surgery. Nevertheless, if simulations last a day or more, the numerical models are not sustainable in a hospital context. A doctor has many patients to take care of and he cannot afford to waste time. Therefore the methods developed to build the simulation architecture responsible for creating the digital breast and perform the breast surgery simulations, need to be optimised. This optimisation depends on the nature of the algorithms as well as the efficient use of modern CPU vectorisation units and/or the GPU capacity. Processing multiple data at the same time help to improve the overall time consumption of the numerical simulations.

The nature of a breast model based on parameters relies on the viscoelastic coefficients of the tissues and on geometric parameters which define the size and shape of a breast (Delingette and Ayache (2004)). The usage of measurements performed in a doctor's office to obtain those parameters will lead to optimisation algorithms which are very time-consuming. However, for problems with sets of inputs (in this case the measurements) that relate to a certain set of outputs (in this case the breast biomechanical parameters), deep learning (neural network) is quite efficient and will be explored in the present work to produce accurate solutions For breast simulation. Notice that deep learning algorithms was already used in the medical context, but essentially connected to the identification and classification of cancers and tumours (Sun et al. (2017); Litjens et al. (2017); Hamidinekoo et al. (2018)). It is intended to apply the deep learning framework (for example a Neural

Network (NN)) to train on simulation data that provide the biomechanical parameters of the breast from a set of measurements.

This kind of approach leads to the elaboration of software capable of quickly simulating, with adequate accuracy, surgical procedures and help in the education of future surgeons. A team of doctors lead by Dra. Augusta Cardoso in Hospital de Gaia collaborates in this project. They validate the data used and the results obtained.

## 1.2 OBJECTIVES

The main goals of this project are to develop new methods and models for decision support instruments that help doctors for the planning and management of surgical procedures. Such supports help reducing the risks and improving the final result of the plastic surgery as well as the life quality of the patient.

Two main issues are addressed: the construction of a numerical model based on nonlinear mechanics for simulating surgery of the breath with suturing, and in an other hand, one considers the parameters' identification problem using simple data deriving from the measures performed by the doctor. Moreover, new technologies based on the deep learning algorithms are introduced to faster provide the necessary biomechanical coefficients and geometrical parameters that are fuelled in to the numerical model. The surgical procedures are limited to breast reduction surgeries but the proposed methodology is extendable to similar situations (soft tissues surgery).

In addition to the models, a study about the implementation efficiency is also done carried out. Indeed, due to the time consuming nature of the numerical methods used in these types of problems, an optimisation of said methods by taking advantage of the modern hardware (CPU vectorisation units and GPU) is done. In the end, there is also an interest of create a simulation architecture with the conducted research with the intent of it being used by doctors to use them in their surgery plannings and also by teachers to help theirs students gain some sensibility and experience before jumping into real life cases.

## 1.3 CONTRIBUTIONS

The main contributions of the manuscript are the following:

- A new breast numerical model capable of simulating with accuracy breast reduction surgeries;
- A proof of concept related to the possibility of using measurements performed on a patient by a doctor in an office to retrieve with success the biomechanical parameters of the patient's breast;

- A new method, based on the proof of concept, to estimate biomechanical parameters of a breast by using measurements performed on a patient;
- Introduction of deep learning algorithms to provide the biomechanical parameters of the breast from a set of measurements of a patient;
- Optimisation and speed-up of the numerical methods used in the breast surgery simulator architecture by using vectorisation and gpu computing;
- The development of an online platform for breast simulation and parameter identification ensuring surgeons to plan breast reduction surgeries and evaluate the properties of a breast. Platform located at "http://surgery.natura2.di.uminho.pt/vaps.html".

In addition to this, other contributions including articles and conferences are:

- D. Lopes and S. Clain and R.M.S. Pereira and G.J. Machado and G. Smirnov and I. Vasilevskiy, Numerical simulation of breast reduction with a new knitting condition, International Journal for Numerical Methods in Biomedical Engineering, Volume 33, Issue 2, 2017, doi: 10.1002/cnm.2796;
- Diogo Lopes and Stéphane Clain and Augusta Cardoso and Igor Vasilevskiy and Georgi Smirnov, Breast Simulation Applications in Real-Life Cases, $7^o$ Congresso Nacional de Biomecânica, January 2017, Guimarães.

Papers in the process of submission:

- Diogo Lopes, Stéphane Clain and António Fernandes, Parameters identification method for breast biomechanical numerical model, Journal of the Mechanical Behavior of Biomedical Materials
- Diogo Lopes, Stéphane Clain and António Fernandes, Estimation of the Breast Mechanical Parameters with a Hybrid Method, IEEE transactions on Biomedical Engineering

## 1.4 DOCUMENT STRUCTURE

The outline of the manuscript structure is the following. Chapter 2 presents an overview of the research scope as well as a review of the state of the art on soft tissue modelling and breast models. In chapter 3 a new breast model with all surgery simulation aspects is introduced. Chapter 4 is dedicated to the proof of concept and the new method to estimate the biomechanical parameters of the breast using measurements of a patient's breast. Chapter 5 presents the optimisation steps taken to improve the computational efficiency

of the breast model. A new paradigm for parameters evaluation with deep learning algorithms is presented in chapter 6. A hybrid method that combines an iterative method and machine learning is described in chapter 7. Chapter 8 propose an evaluation of the impact of the study in real life cases is presented and describe the online platform that has been developed. At last, chapter 9 presents the final conclusions and future work.

# 2

# NUMERICAL SURGERY

The human body is mainly made up of soft tissue and the medical issues on soft tissue modelling are critical for improving treatments ranging from neurosurgery to plastic surgery (Delingette (1998)). In a general way, modelling and simulation of soft tissue turn to be an important toolbox to tackle anatomy behaviour to external perturbations (Delingette and Ayache (2004)). Representative models can then be used to provide a glimpse into the outcome of medical procedures like surgeries.

The breast is a complex soft tissue and there is a considerable literature dedicated to create models and reproduce its shape, size and behaviour (Azar et al. (2002); del Palomar et al. (2008); Han et al. (2012a); Cardoso et al. (2013a); Eder et al. (2014); Eiben et al. (2016)). All of the proposed models consider the breast as a complex structure and evaluate the external perturbations using numerical methods but they differ on which methods are used and on the impact of some of the breast tissues. By analysing these models, one can have a better understanding of how can these models be used in surgical circumstances and how accurate they can be.

## 2.1 PHYSIOLOGY

Knowledge of the human body from what can be seen with the unaided eye down to the molecular level is fundamental to understanding how an organ structure and function are modified, among other external sources, by disease (McCuskey et al. (2005)). For millennia, manual palpation has been used to assist in diagnosis, but it is subjective and restricted to superficial structures (Wells and Liang (2011)). During the last decades, there have been new techniques for imaging anatomy in living patients (McCuskey et al. (2005)). Many protocols and moralities of digital imaging techniques have been adopted to generate medical images, including Magnetic resonance imaging (MRI), Computed Tomography (CT), ultrasound, X-rays and others (Li et al. (2018)). With time, the usage of those technologies and techniques became a normal procedure and play a key role in the treatment and management of diseases like cancer (Filleron (2018)). However, the radiologists and doctors capacity to accurately obtain, communicate and interpret the imaging data without errors

occurring are fundamental to the patient's health (Zhou et al. (2015)). Moreover, there is always a level of uncertainty regarding the outcome of any treatment like a surgery (Ménard et al. (2018)). For that reason, in the middle of the 1990's, the necessity of predicting the outcome of treatments, especially surgical procedures, arose (Delingette and Ayache (2004)). That problem lies in the complexity of the living tissues and the way they respond to external perturbations. Hence the importance of research and development of soft tissue models and surgical simulators (Delingette and Ayache (2004)).

The human body is made up of soft tissues like tendons, skin, ligaments, muscles, internal organs and many others, which are biological tissues made up by cells, collagen, elastin and ground substance (Marino (2018)). The mechanical response of soft tissues is characterised by nonlinear stress-strain relationships associated with an incompressible or quasi-incompressible behaviour due to the high water content (Delingette and Ayache (2004); del Palomar et al. (2008); Chen et al. (2013); Marino (2018)). Besides, the mechanical response has a high impact on the various organs in both health and disease. So the impact of soft tissue models is relevant in many medical fields (Delingette (1998)). To model any kind of soft tissue, one must consider its anatomy. On that regard, medical instruments like MRI, CT and ultrasounds, among others, are very useful tools to help in determining the shape, size, for instance, and mechanical properties of those tissues (Maurel et al. (1998); Delingette and Ayache (2004); Eiben et al. (2016); Martínez-Martínez et al. (2017); Liu et al. (2017)). In the end, this set of data are used to design constitutive models of the tissues which in turn are used to simulate/predict the outcome of surgeries (Maurel et al. (1998); Delingette and Ayache (2004); Marino (2018)). A constitutive model, or computational biomechanical model, of a soft tissue, is a representation of the mechanical properties of the materials through mathematical models together with physical concepts and empirical data (Marino (2018)).

The response behaviour of soft tissues is quite complex and it turns difficult to determine how the various components of a soft tissue like a liver or a breast will position after a surgical procedure is undertaken. The forces applied by the surgeons and the instruments during the surgery, as well as the gravity in a post-surgical context, lead the organs to spatially re-arrange themselves in ways that are hard to predict (Delingette and Ayache (2004)).

Collagen and elastin are the constituents which are mainly responsible for the tissue elasticity, viscoelasticity, stiffness and strength properties while the high water content in ground substance plays an important role in terms of the tissue mobility (Maurel et al. (1998); Marino (2018)). Depending on the tissues, these constituents' proportion and arrangement vary. Elastin is assembled as a continuous network of fibres which allows the tissues to resume their shape after stretching or contracting. Collagen is an abundant protein in the human body and is the main responsible for maintaining the structural integrity

of soft tissues. It can be found in a regular network of fibres (e.g. tendons) or with an irregular pattern (e.g. skin) (Holzapfel (2001)). It provides stiffness to the tissues, (i.e.), the collagen tissues can be rigid like the ones existing in bones or they can be more or less stiff in tissues like the skin, the tendons or even cartilage (Maurel et al. (1998); Marino (2018)). Ground substance is a gelatinous matrix responsible for the viscoelastic behaviour of tissues in the extracellular space and so its amount in dense connective tissues like the tendons is much smaller when comparing to looser connective tissues like the skin (Maurel et al. (1998)).

Unlike hard tissues like bones, soft tissues may undergo large deformations. First, soft tissues behave anisotropically and present viscoelastic behaviour while they are considered incompressible or quasi-incompressible. As an example, the skin is an anisotropic tissue because it has different properties between the planar directions (tangent to the surface) and the normal direction (deep in the skin). This can be seen in the Langer's lines which correspond to the directions of the collagen fibres preferred disposition (Langer (1978)). The skin is a viscoelastic material because it possesses both recoverable elastic deformation as well as viscous deformation. Depending on the amount of strain, (i.e.), for small deformations the skin is capable of recovering to its original state but for larger deformations, the skin can permanently deform. Moreover, the viscoelasticity is also represented by the different responses to the same amount of strain applied in different time intervals (strain-rate), (i.e.), when a certain amount of stress ($\sigma$) is exerted on the skin slowly will lead to a different amount of strain ($\varepsilon$) when that same amount of stress is exerted but more quickly (Marino (2018)). Like any other viscoelastic material, the skin exhibits stress relaxation under constant strain and deforms under constant stress.

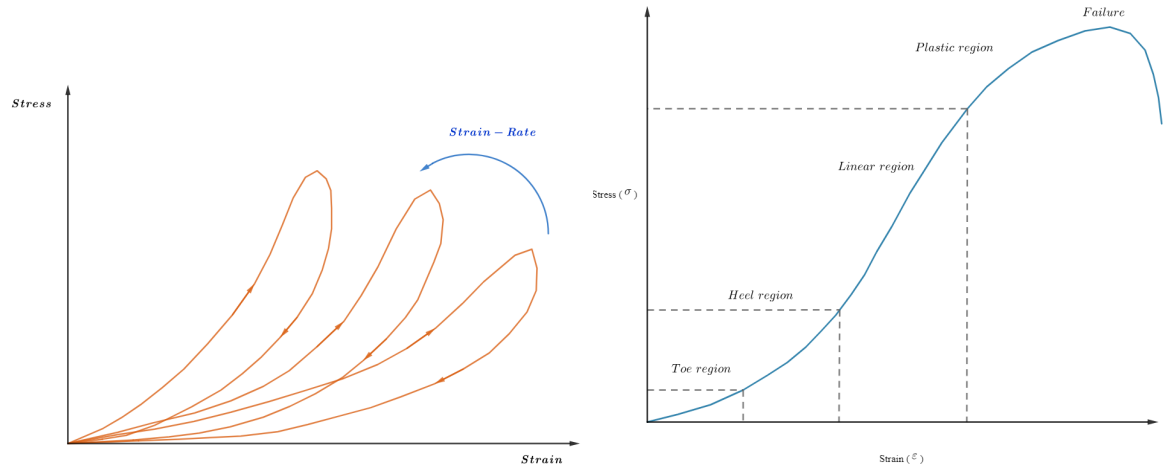



Figure 1: Viscoelastic response of soft tissues. On the left it shows the shift of the stress-strain response curves (orange) according to the strain-rate (strain of the material with respect to time). On the right the stress-strain curve of soft tissues with 4 regions: toe, heel, linear, plastic.

This means that with time if the skin is under constant strain, the levels of stress ($\sigma$) start to decrease while if the skin is under constant stress it will continue and deform (Holzapfel (2001)). If the process of loading and unloading of stress is repeated then a stress-strain phase lag is experienced which can be seen in the figure 1 (left). The mechanical response of soft tissues is highly nonlinear and can be characterised by J-shaped stress-strain curves (Maurel et al. (1998); Holzapfel (2001); Marino (2018)). Figure 1 (right) shows 4 different regions in the stress-strain curve. The toe and heel regions are considered low stiffness regions where the fibres are relaxed. However, in the heel region, the stiffening response by the tissue starts. The linear region strain range is characterised by higher stiffness of the tissue where the fibres become straighter. Moreover, the region is considered linear due to the shape of the curve in that area as seen in figure 1. When the values of strain are too high the fibres start to show plastic behaviour, (i.e.), it leads to permanent deformations (Holzapfel (2001); Marino (2018)). A material compressibility is defined by the change in volume per change in pressure/stress (Maurel et al. (1998)). For very large changes in the pressure applied to the skin, there are very small changes in volume, which means that the skin is very hard to compress and therefore it is considered to be incompressible or quasi-incompressible (Maurel et al. (1998)).

## 2.2 SOFT TISSUE MODELLING

The behaviour of many soft tissues was tested and studied with the observance of its microscopic composition as well as the response to deformations (Maurel et al. (1998); Marino (2018)). The purpose of this evaluation was to obtain general constitutive laws describing the behaviour of the materials. What was observed is that a soft tissue behaviour depends on the structure and organisation of the elastin, collagen and ground substance over the tissue. However, the mechanical properties of a tissue can hardly be completely assessed (Maurel et al. (1998)). That leads to assumptions and necessary simplifications when developing any soft tissue model (Delingette and Ayache (2004)).

To recreate the behaviour of these tissues, from the biomechanical point of view, one has to formulate a constitutive equation relating the tension in the material to the deformation it undergoes. In other words, a biomechanical model is a mathematical description of the tissues deformation with respect to the stress by linking the strain to the applied forces assuming elasticity and viscoelasticity properties of the material (Maurel et al. (1998)). Constraints and deformation in real tissues can accurately be represented via the design of numerical schemes (Delingette (1998)).

### 2.2.1 *Breast geometrical representation*

There are several aspects to consider when analysing a problem including soft tissue modelling. A first issue concerns the identification of the elastic and viscoelastic parameters of the tissues as well as the methods to simulate the behaviour of those tissues. For that purpose, soft tissue models are elaborated regarding the tissue's geometry, deformation mechanics, boundary conditions and equilibrium equations (Maurel et al. (1998)).

In numerical surgery, a piece of the body can be as simple as a single soft tissue like the skin or tendon or can be made by several soft tissues like an organ (e.g. liver) or system (e.g. circulatory system). In real life, every tissue is intrinsically volumetric, but different tissues can be represented only having one dimension (e.g. vein), two dimensions (e.g. skin) or three dimensions (e.g. hearth) with respect to the characteristic size of the domain. From a numerical point of view, these dimensions are represented by geometric shapes like points/vertexes, lines/edges, triangles/quads, tetrahedrons/hexahedrons or other polyhedrons(Delingette and Ayache (2004)). These shapes are then assembled together to form meshes which retrieve the shape and size of the tissue(s) like what happens with FEM, or they can be used to represent the surface of a volumetric structure (Delingette and Ayache (2004)) or they can be a mix, (i.e.), these shapes handle very few elements but whose size is large (Costa and Balaniuk (2001)). The detail degree of the geometry representation depends on the goals of the model and the accuracy requirements. On one hand, the usage of a simpler volumetric structure leads to perform fewer calculations and therefore consume less time when compared with more complex geometries with large amounts of elements. On the other hand, the existence of more elements leads to more flexibility and better approximations of the real behaviour of living tissues (Delingette and Ayache (2004); Rajagopal et al. (2007)). The same happens when simplifying a volumetric structure like the vein with an asymptotic one-dimensional model instead of considering the full three-dimensional object.

The free-constraint domain is the reference domain, *i.e.* the body shape when no constraint is prescribed. External forces will be applied to the reference domain in which the resulting deformations will be evaluated. In numerical terms, that is achieved by using discrete models and numerical methods which can determine the behaviour of the tissues (Maurel et al. (1998); Marino (2018)). Tissues parameters that characterise the material are used together with numerical expressions to compute the stress and strain (e.g. tensors) which in turn determine the displacement of the geometry. Soft tissue models usually have specific and distinct purposes (Delingette and Ayache (2004)). To define those purposes they use what is commonly known as boundary conditions, which are constraints necessary for conditioning the problem (Maurel et al. (1998); Delingette and Ayache (2004); Rajagopal et al. (2007)). As an example, when considering breast surgery, one of the boundary con-

ditions is to ensure that the zone of the breast attached to the torso can only move inside a specific zone (Cardoso et al. (2013a)). If such a condition was not considered, then the breast tissues would behave in a less natural manner like falling of the body. Another aspect that arises when analysing soft tissue models is the existence of equations that ensures the conservation of some properties. For example, a breast under gravity should maintain the same mass (Rajagopal et al. (2007)). For that purpose, soft tissue models have to derive from conservation laws that provide equilibrium equations (Rajagopal et al. (2007)). These equations determine the behaviour of a tissue by balancing the stress and strain over the elements of a particular geometry.

### 2.2.2 *Soft Tissue Mechanics*

According to Delingette and Ayache, the different mechanics that represent the deformation of soft tissue are extremely diversified, varying from continuum mechanics to mass-spring systems, chain-mail and long element models (Delingette and Ayache (2004)):

- Continuum Mechanics: simulate deformations on continuous volumetric objects (e.g. meshes) by considering the tissue as a continuum medium. Such methods can evaluate major deformations and operations of a tissue as cuts and sutures by solving an optimisation problem where the values of strain tension and energy are minimised in order to find the best mesh configuration (Donea et al. (2004));

- Mass-Spring: simulate deformations for elastic elements by simulating the elastic deformation of a directional force (e.g. gravity on a breast). In this algorithm the elements of the tissues behave like a spring pointed in a certain direction where they are displaced according to its mass and elasticity by using Hooke's Law (Delingette and Ayache (2004));

- Chain-mail: they are mainly used to simulate deformations on tissues with high values of stiffness (e.g. cartilage). It simulates deformations on tissues that behave like a set of elements linked in a chain, (i.e.), the force propagates from element to element and displaces them. However, according to Gibson et al. (1997), this scheme is only valid for small values of displacement;

- Long element method: based on Pascal's principle and volume conservation, it implements a static solution for elastic global deformations of objects filled with fluid. The volumes are discretised in long elements that handle meshes with one order of magnitude smaller than tetrahedral meshes while the body is modelled using bulk variables like pressure, density, volume and stress (Costa and Balaniuk (2001)).

The choice of the method to evaluate the deformations depends on the tissue characteristics and on the simulation context. For soft tissues, the usage of chain-mail is not appropriate due to the small deformations restriction with large values for the stiffness. Mass-spring system is considered too simplistic for complex deformations such as the ones which occur during a major surgery although it is simple to implement and provides low computational costs (Delingette and Ayache (2004)). Long element methods are faster due to the use of one less magnitude order element but the static solution yields that the final result suffers from a rough approximation of the deformation which results into a more unpredictable result. Continuum mechanics are capable of evaluating large deformations using a dynamic approach but the main drawback is the expensive cost in terms of computational resources. Nevertheless, the majority of the soft tissue models in the literature use continuum mechanics to describe the behaviour due to their versatility and accuracy (Delingette (1998); Maurel et al. (1998); Azar et al. (2002); del Palomar et al. (2008); Rajagopal et al. (2007); Cardoso et al. (2013a); Eiben et al. (2016)). The main reasons behind this choice by the models lie in the validated framework for modelling the deformation of volumetric objects as well as the scalability (the behaviour of a mesh is guaranteed independently of the mesh granularity) and the possibility of identifying the tissues physical parameters (Delingette and Ayache (2004)).

*Continuum mechanics: fundamentals*

In continuum mechanics, one describes the three-dimensional geometry of a soft tissue in the free-constraint reference configuration/state as $\Omega_0$ where the material point is located by the position vector $X$. Applying forces and constraints, $x$ represents the positions of the same material point in the deformed configuration $\Omega$. Various deformation mechanic expressions can be formulated in terms of the reference configuration (Lagrangian description) or in terms of the deformed configuration (Eulerian description). Usually, the Lagrangian description is used in elastic models (Rajagopal et al. (2007)). This is important when analysing the behaviour of soft tissues which can be represented by analysing their strain and stress values. While stress expresses the forces that are exerted over the elements of the tissue's geometry elements (e.g. gravity), strain represents the deformation of the material (Maurel et al. (1998)). This means that for strain, one can evaluate its amount by evaluating the difference between the tissue in a reference state and in a deformed state.

To this end, one introduces the transformation that maps the reference configuration to the deformed one

$$X \in \Omega_0 \rightarrow x = x(X) \ni \Omega, \tag{1}$$

and the deformation gradient tensor $F = F(X) = \nabla_X x(X)$ which relates the geometry reference state to the deformed state (Maurel et al. (1998); Rajagopal et al. (2007)). From the deformation tensor, the strain tensors (Maurel et al. (1998); Rajagopal et al. (2007))

measures the relative displacement of the material points (Kaye et al. (1998)). The most common strain tensors are the Cauchy-Green deformation tensors (right and left), as well as the Green-Lagrange tensor, the Cauchy tensor (also known as Piola tensor) and the small-strain tensor (Maurel et al. (1998); Kaye et al. (1998)). These tensors derive from the deformation gradient tensor and are used to evaluate the amount of strain in the tissue(s). For example, the Cauchy-Green right tensor is usually defined as $C = F^T F$ and is used as the strain tensor for evaluating the strain in viscoelastic materials (Kaye et al. (1998); Maurel et al. (1998)). The Cauchy-Green left tensor is usually defined as $B = FF^T$ and is used to measure strain in elastic materials (Kaye et al. (1998)). The strain tensors used to measure the strain in soft tissues depends on the material's nature, (i.e.), a linear elastic material may use different tensors than a hyperelastic material (Maurel et al. (1998); Delingette and Ayache (2004)).

The stress is assumed to act passively under external loads from the initial unstressed and reference state $\Omega_0$ (Maurel et al. (1998)). Stress is usually denoted by $T$ and can be described as the application of a force over an object (Delingette and Ayache (2004)). Those forces can be either volume forces that affect the entire object (e.g. gravity) or superficial forces applied on the boundary and propagated inside the domain (e.g. pressure). Let $S$ be an elementary piece of the plane passing through a material point $X$. The application of a force $\vec{F}$ over $S$ can generate different types of stress: normal and shear stress. While normal stress results from the force vector component perpendicular to the material cross-section, shear stress results from the parallel component (Figure 2). The stress vector $T$ can be regarded as the sum of these two components.

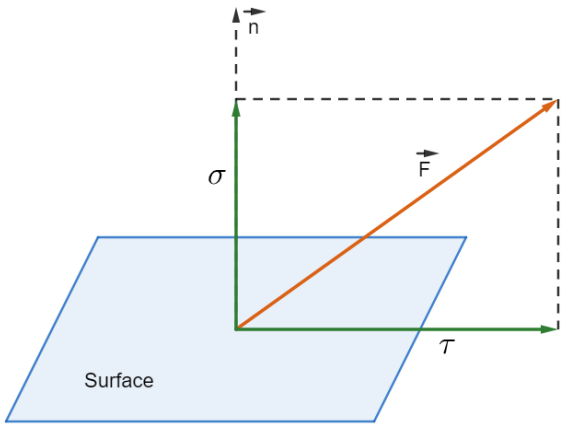



Figure 2: Normal Stress ($\sigma$) and Shear Stress ($\tau$) resulting from the superficial tension $\vec{F}$

A stress tensor is able to measure the amount of stress that is exerted at each element of a tissue and to do that it relates the forces to the areas in the object. Normal stress (denoted by $\sigma$) is the orthogonal component of the tensor with respect to the plane, while the shear stress (denoted by $\tau$) is the component lying in the plane (Maurel et al. (1998);

Kaye et al. (1998)). The most common stress tensors are the Cauchy stress tensor and the first and second Piola-Kirchhoff stress tensors (Maurel et al. (1998)). While the Cauchy stress tensor is used to analyse the stress on materials experiencing small deformations, the Piola-Kirchhoff stress tensors can be used to measure stress for large deformations (Maurel et al. (1998)). Indeed, the Cauchy tensor measures the stress on the deformed state ($\Omega$) while the Piola-Kirchhoff tensors measure stress with respect to the reference state ($\Omega_0$) together with the gradient deformation tensor $F$ (Maurel et al. (1998); Delingette and Ayache (2004); Marino (2018)). The usage of the reference state is the reason why the Piola-Kirchhoff tensors are able to evaluate stress considering large deformations (Maurel et al. (1998)).

*Stress and Strain analysis*

As mentioned before, most soft tissues enjoy elastic and viscous behaviour (viscoelasticity). The elastic component of a soft tissue is the one that allows that tissue to resist the stress loading and go back to its initial state when the load is removed. The viscous component exhibits time-dependent behaviour when the stress is applied and a residual deformation is observed after the load release (Papanicolaou and Zaoutsos (2001)). Most of the viscoelastic soft tissues possess a linear elastic behaviour for small amounts of strain and a nonlinear behaviour for large deformations (Maurel et al. (1998); Delingette and Ayache (2004)). When studying elasticity in the $17^{th}$ century, Robert Hooke noticed that, for many materials, the relation between stress and strain has a linear region. For displacements lower than a characteristic size depending on the material, the force required to stretch an elastic object such as a metal spring is directly proportional to the extension of the spring. This is known as Hooke's law and commonly written

$$F = -kx \tag{2}$$

where $F$ is the force, $x$ is the length of extension/compression and k is a constant of proportionality known as the spring constant (Khan Academy: Hooke's Law). Elasticity can be described by a simple relation between stress ($\sigma$) and strain ($\varepsilon$) which can be measured with tensors (Maurel et al. (1998)). The deformation gradient tensor ($F$) is necessary for evaluating the strain. The reason is, that this tensor describes the motion from the three-dimensional object from its reference state ($X \in \Omega_0$) to the deformed state ($x \in \Omega$), (i.e.), it relates any elementary segment $dX$ of $\Omega_0$ into a segment of $dx$ of $\Omega$ giving $F = \frac{dx}{dX}$ (Maurel et al. (1998)). The deformation of a tissue can be characterised and quantified through the analysis of the Cauchy-Green right tensor given by $C = F^T F$ (Delingette and Ayache (2004)). To measure the strain, one considers the steady-state configuration of the deformed state,

and deriving the right Cauchy-Green deformation tensor, the Green-Lagrange strain tensor $E$ is obtained with

$$E = \frac{1}{2}(C - I) \tag{3}$$

where $I$ is the identity matrix.

The displacement is denoted by $u = x - X$ while the tensors are expressed in terms of the displacement gradient tensor with respect to the coordinates of the reference domain $\Omega_0$

$$\nabla_X u = \frac{du}{dX}. \tag{4}$$

The tensors then read

$$F = I + \nabla u \tag{5}$$

$$C = F^T F = I + \nabla u + \nabla u^T + \nabla u^T \nabla u \tag{6}$$

$$E = \frac{1}{2}(\nabla u + \nabla u^T + \nabla u^T \nabla u). \tag{7}$$

For small deformations, simplification are performed by keeping the linear first order term (*i.e.* neglecting the non-linear components providing the so-called infinitesimal strain tensor $\epsilon$

$$E \approx \epsilon = \frac{1}{2}(\nabla u + \nabla u^T) \tag{8}$$

(Delingette and Ayache (2004)).

In a three-dimensional body, the stress at a material point is the sum of all the forces that are being applied upon it (Delingette and Ayache (2004)). According to Cauchy stress theorem, the force long direction $n$ are represented as a linear combination between a tensor and the vector, namely

$$T = \sigma \cdot n \tag{9}$$

where $\sigma$ is the $3\times3$ Cauchy's stress tensor, evaluated at the material point $x$ associated to the deformed domain $\Omega$ which stores the values of stress (compression, traction and shear).

Alternative stress tensors are the Piola-Kirchhoff $1^{st}$ and $2^{nd}$ stress tensors. The $1^{st}$ Piola-Kirchhoff $P$ relates the forces in the deformed configuration for direction given in the reference configuration and is expressed by

$$P = J\sigma F^{-T} \tag{10}$$

where $\sigma$ is the Cauchy's stress tensor, $F$ is the gradient deformation tensor necessary to relate the reference and deformed configurations and $J$ is the Jacobian determinant of $F$ ($J = det(F)$) which expresses how much the area of the element in the reference state was changed (e.g. stretched or shrunk) in comparison to the deformed state. The usage of this tensor is not practical because it is not symmetric and that leads to the non-preservation

of the stress in the reference configuration (Maurel et al. (1998)). For that purpose, an additional element is necessary to preserve said condition which leads to the $2^{nd}$ Piola-Kirchhoff stress tensor which is given by

$$P = JF^{-1}\sigma F^{-T} \tag{11}$$

where $F^{-1}$ is the element that guaranties the symmetry of the tensor and therefore corresponds the stress tensor in the reference configuration to the deformed configuration (Maurel et al. (1998); Marino (2018)). Consequently, the $2^{nd}$ Piola-Kirchhoff tensor is the representation of the stress with respect to the initial configuration $\Omega_0$.

*Virtual work*

To model the displacements of the tissues, one applies the virtual work principle which states that for any virtual displacement applied on a domain, the total internal virtual work is equal to the total external work (Delingette and Ayache (2004)). The total internal work is given by the integral of elastic energy ($W(X)$) over the body volume whereas the external work derives from the application of all the internal and external forces (Delingette and Ayache (2004)), leading to

$$\int_{\Omega_0} W(X)d\Omega_0 = \int_{\Omega_0} u^T f^{\Omega_0} d\Omega_0 + \int_{\partial\Omega_0} u^T f^{\partial\Omega_0} d\partial\Omega_0 \tag{12}$$

where $\Omega_0$ is the volume, $\partial\Omega_0$ are the external surface areas, $u$ is the displacement vector, $f^i$ represents the internal forces and $f^e$ represents the external forces.

The internal elastic energy of a tissue is given as a function of the strain tensor (Maurel et al. (1998)), called strain energy function and it relates the amount of energy stored in a tissue undergoing deformation and the deformation gradient (Delingette and Ayache (2004)). There is a wide spectrum of models depending on the level of non-linear and non-isotropic behaviour. As a first approximation, the material is assumed to be linear and isotropic such that the strain energy function is a function of the tensor invariant given by:

$$W_i = \frac{\lambda}{2}(tr(\varepsilon))^2 + \mu\ tr(\varepsilon^2) \tag{13}$$

with $\lambda$ and $\mu$ the Lamé parameters, $tr(\varepsilon)$ the trace function of the strain tensor $\varepsilon$. The trace function is the sum of the eigenvalues of the strain tensor, i.e, the values of the main diagonal. The Lamé parameters are quantities that express the material stiffness (Delingette

and Ayache (2004)). The matrix invariant ($I_i,\ i = \{1,2,3\}$) are defined by the eigenvalues ($k_i,\ i = \{1,2,3\}$) of the strain tensor as

$$\begin{aligned} I_1 &= tr(\varepsilon) = k_1 + k_2 + k_3, \\ I_2 &= \frac{1}{2}(tr(\varepsilon)^2 - tr(\varepsilon^2)) = k_1k_2 + k_1k_3 + k_2k_3, \\ I_3 &= det(\varepsilon) = k_1k_2k_3. \end{aligned}$$

An anisotropic elastic material requires a corrective term ($\Delta W_{ani}$) to evaluate the energy (Picinbono et al. (2002)). The anisotropic contribution is composed of the additional stretch and shear (Picinbono et al. (2002)) given by

$$\Delta W_{ani} = \Delta\lambda I_4 tr(\varepsilon) + 2\Delta\mu I_5 - (\frac{\Delta\lambda}{2} + \Delta\mu) I_4^2 \tag{14}$$

where $I_4 = a^T\varepsilon a$ and $I_5 = a^T\varepsilon^2$ are the invariant which characterise the strain tensor (stretch and strain respectively) in a direction $a$, $\Delta\lambda$ and $\Delta\mu$ are the Lamé parameters of the material in the direction $a$. At last, the strain energy is given by:

$$W = W_i + \Delta W_{ani}. \tag{15}$$

In most of situations, soft tissues are assumed to be hyper-elastic material (Maurel et al. (1998); Delingette and Ayache (2004); Rajagopal et al. (2007); Marino (2018)) with nonlinear behaviour (Delingette and Ayache (2004)). As the linear situation, nonlinear stress-strain relationship derives from a strain energy function where the strain energy function ($W_{nl}$) is given by the St Venant-Kirchhoff elasticity (Delingette and Ayache (2004)). The function is similar to equation 13 but the infinitesimal strain tensor is replaced by the Green-Lagrange strain tensor $E$

$$W_{nl} = \frac{\lambda}{2}(tr(E))^2 + \mu\ tr(E^2). \tag{16}$$

For non isotropic material, one employs the same corrective term as in the linear case.

An elastic material is characterised by the capacity return to the reference configuration after deformation when releasing the external forces (no residual deformation). On the contrary, a viscoelastic material is characterised by a permanent deformation of the material and better correspond to the tissue response (Maurel et al. (1998); Azar et al. (2002)). This leads to consider a general linear viscoelastic relationship between stress and strain with respect to time (Maurel et al. (1998)). The resolution of this problem consists in evaluating the stress relaxation, i.e, by measuring the stress over time

$$\sigma(t) = E\varepsilon(t) + \eta\frac{d\varepsilon(t)}{dt} \tag{17}$$

where $\sigma$ is the stress tensor, $\varepsilon$ is the strain tensor, $E$ is the elastic modulus, $\eta$ is the viscous modulus (Maurel et al. (1998)). For isotropic materials the Kelvin-Voigt model is one of the most common methods to solve this problem and reads

$$\sigma = E(\varepsilon) + \eta(\varepsilon) \tag{18}$$

$$E(\varepsilon) = \lambda tr(\varepsilon) I + 2\mu\varepsilon \tag{19}$$

$$\eta(\varepsilon) = \lambda\theta_\lambda tr(\varepsilon) I + 2\mu\theta_\mu\varepsilon \tag{20}$$

with $E(\varepsilon)$ the elastic component and $\eta(\varepsilon)$ the viscous contribution with respect to the strain tensor $\varepsilon$ (Maurel et al. (1998)).

*Boundary Conditions*

The boundary conditions of a soft tissue model are constraints located at the boundaries of the tissues and are necessary to solve problems involving the application of superficial forces (like pressure or contact) and fixed displacements (Delingette and Ayache (2004)).

For example, let's consider a bariatric surgery where part of the stomach is removed and then the remaining part is sutured into the duodenum. The incised part of the stomach and the duodenum are considered boundaries of the problem. To ensure that the simulation complies with the surgical procedure, a boundary condition is prescribed in order to force a union between the incised part of the stomach and the duodenum. If this type of condition is not applied, the simulation can produce results that might not be considered valid in real life cases (for example, the stomach connects with the lateral part of the duodenum). So, boundary conditions can be interpreted in terms of constrained displacements and other similar boundary conditions, like fixation, are displayed in figure 3.

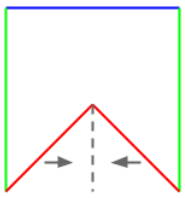


Figure 3: Example of a boundary conditions scheme of a model. In this example, the blue line is restricted to not move, the red line is forced to move to the centre grey line, and the green line if free to move according to displacements happening with the neighbour lines.

In order to correctly simulate any action, the strain energy functions need to satisfy the boundary conditions (Rajagopal et al. (2007)). The total energy of the system is composed of the volume force contribution and the superficial force acting on the boundary, namely

$$W = W_i,\ x \in \Omega$$
$$W = W_{bc}(Bc),\ x \in \Gamma_{bc}$$

where $\Gamma_{bc}$ is the boundary of $\Omega$ and $W_{bc}$ is the strain energy on the boundary.

*Equilibrium Equations*

The equilibrium equations derive from conservation laws for mass, momentum, energy, Rajagopal et al. (2007). Let's consider a case like the displacement of the breast due to the action of gravity. In this case, the mass that the breast has in the reference state (considering it was in a free-stress state) is the same when the breast is in the deformed state (after applying gravity). If the soft tissue model does not ensure the conservation of mass, then it is possible that during the simulating the values of mass could change, which is clearly non-physical.

Extending this idea to the balancing of forces, one gets the conservation of momentum (Rajagopal et al. (2007)), *i.e.*, in the reference state the breast tissue is at rest since no constraints are applied, but when an external force is applied, internal forces are created with displacement to balance the forces, and those displacements lead to a new shape of the breast.

As Rajagopal et al. (2007) mentions, the conservation laws must be satisfied subject to the particular set of boundary conditions o the problem.

### 2.2.3 *Numerical Methods*

Theoretical mechanics provide the models that describe the properties and behaviour of soft tissues. To compute displacement approximations and carry out simulations, numerical methods will be designed. Most common methods used to compute the deformation based on the elasticity theory are FEM, Boundary Element Modelling (BEM) and Finite Difference Method (FDM) (Maurel et al. (1998); Delingette and Ayache (2004)). Linear models lead to less costly solutions since no fixed point iterative procedure is required. Nevertheless, in the case of the breast, physically realist models involving non-linear terms have to be taken into account (Maurel et al. (1998)).

The idea of FEM is to approximate a continuous function with a simple regular shape function defined piece-wise on a mesh (Maurel et al. (1998)). A mesh is defined by a set of nodes which are connected between themselves creating elementary volumes named cell

(simplex convex subset). A cell contains edges (represented with lines) and faces (represented with areas) that it shares with its neighbour cells. The most common mesh topology is the tetrahedral mesh along with hexahedral (Delingette and Ayache (2004)). The evaluation of stress and strain energy is carried out with the volume energy computed in the cells, while the energy deriving from forces applied on the boundary is calculated on the faces (Maurel et al. (1998); Delingette and Ayache (2004)). This method allows setting a different kind of energy, in function of the constraints localisation, allowing to describe complex configurations (Delingette and Ayache (2004)).

BEM is a numerical method that only handle the mesh of the domain surface (Delingette and Ayache (2004)), (i.e.), the solution of the problem is calculated for the surfaces (boundaries) and the stress and strain of the internal volume is interpolated (Delingette and Ayache (2004)). The advantages of this method are that it does not require a full three-dimensional mesh and turns to be very efficient for linear elastic isotropic materials when the mesh topology is not modified (Delingette and Ayache (2004)). However, when compared with the other methods, BEM does not present accurate approximations for nonlinear materials and the interpolation of the internal values is quite rough to accurately represent displacements of the internal elements of the materials like tumours (Delingette and Ayache (2004)).

FDM is a numerical method that consists in partitioning the tissue in space and time where an approximation of the solution is calculated for each space and time elements (Maurel et al. (1998)). This method estimates the values of stress and strain of the next time step based on the present time step using finite difference formulation such as central differences (Maurel et al. (1998)). They sometimes lead to the same discrete system as FEM when specific finite elements based on linear interpolation are employed and they can both solve problems with nonlinear materials (Delingette and Ayache (2004)). The main difference between FEM and FDM is that while FDM is easier to implement, FEM is more capable of handling more complex geometry and boundary conditions (Delingette and Ayache (2004)).

## 2.3 STATE OF THE ART

The breast is a complex soft tissue which contains, among others, fat and glandular tissue, ligaments and skin. Moreover, the arrangement of these materials is not the same in every woman (Jesinger (2014)) leading to a case by case approach. The behaviour of the breast was studied by several researchers over the years which lead to the creation of several breast models (Azar et al. (2002); del Palomar et al. (2008); Eder et al. (2014)). The development of any soft tissue model poses restrictions, namely accuracy with admissible time computation. The more detail a model possesses, the higher is the accuracy of such a model in reproducing the real tissue's behaviour. At the same time, more detail leads to more cal-

culations which means more time necessary to perform said calculations (Delingette and Ayache (2004)). This poses the question:

Do one create a realistic model but needing large amounts of time to run any simulation? or Do one simplify some of the model aspects in order to perform simulations more quickly?

The answer to these questions depends on the accuracy restriction to obtaining a suitable approximation for real-life applications. To do that an insight on the breast physiology as well as a review about previous studies is presented.

### 2.3.1 *Breast Physiology*

The breast is a soft tissue composed with a conical disk of glandular tissue which is encased in variable quantities of fat that give it its characteristic shape (Encyclopaedia Britannica: mammary gland). The glandular tissue itself is made up of 15 to 20 lobes constituted of solid cords of ductal cells. In turn, each lobe is subdivided into many smaller lobules, separated by broad fibrous suspensory bands also called Cooper's ligaments, connecting the skin with the fascia or sheet of connective tissue that covers the pectoral muscles beneath the breast (figure 4 - Encyclopaedia Britannica: mammary gland). Each lobe, ligament and

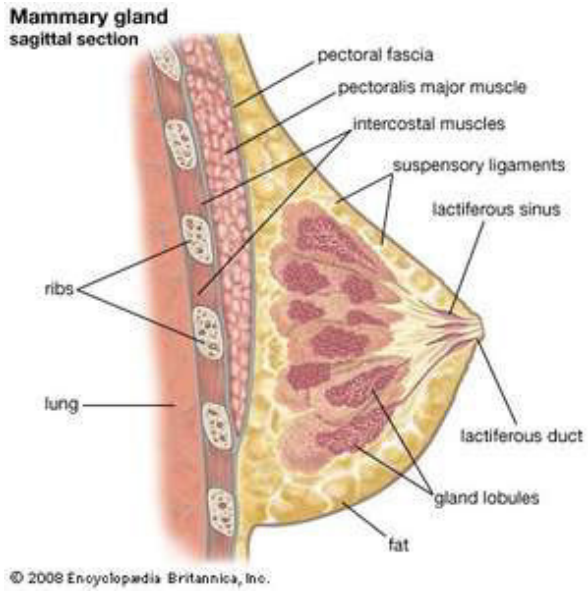



Figure 4: Anatomy of the breast. (Encyclopaedia Britannica)

portion of fat tissue has different mechanical properties that vary with genetic factors and age (Poplack et al. (2004)). Moreover, the amount of internal glandular tissue increases during pregnancy and lactation and regresses after a certain period (Gefen and Dilmoney (2007); Jesinger (2014)).

A good understanding of the breast tissues mechanical properties is essential in any attempt to recreate their behaviour using a numerical model (Azar et al. (2002); Delingette and Ayache (2004); Gefen and Dilmoney (2007)). These tissues exhibit viscoelastic behaviour and so properties such as the elastic modulus are crucial parameters to be plugged in the constitutive equations (Holzapfel (2001); Gefen and Dilmoney (2007)). The range of the property that measures the stiffness of a material (elastic modulus) varies from person to person. Higher values mean more resistance to deformation, more stiffness (Gefen and Dilmoney (2007)). The pectoralis muscle is in the chest wall and is separated of the breast by the pectoralis fascia characterised by a thin lamina of tissue (Gefen and Dilmoney (2007)). Such tissues are usually not used in breast models assumed as a base support for the breast, (i.e.), the connection between the breast and the rest of the torso (Vavourakis et al. (2016)). On the other hand, the fat and glandular tissues, as well as the skin and Cooper's ligaments modelling, are found in published works (Azar et al. (2002); Gefen and Dilmoney (2007); del Palomar et al. (2008)).

The Cooper's ligaments are thin ligaments that cross the subcutaneous fat tissue and help to maintain structural integrity (Gefen and Dilmoney (2007)). They are very stiff and, like other ligaments, the value of elastic modulus ranges between 80 $MPa$ and 400 $MPa$ (Gefen and Dilmoney (2007)). The breast glandular tissue houses the lobes and ducts (where the milk is produced and conducted) which converge in a network towards the nipple (Encyclopaedia Britannica: mammary gland). They are less stiff than the Cooper's ligaments with the range of the elastic modulus ranging from 7.5 $kPa$ to 66 $kPa$ (Gefen and Dilmoney (2007)). The fat tissue of the breast (which is white adipose tissue) stores energy and it surrounds the glandular tissue (Encyclopaedia Britannica: mammary gland). It is characterised by its low stiffness with the values of the elastic modulus ranging from 0.2 $kPa$ to 25 $kPa$ (Gefen and Dilmoney (2007)). The skin helps support the internal breast tissue and exhibits high nonlinear and viscoelastic behaviour (Maurel et al. (1998)). It is made of various layers (epidermis, dermis, and hypodermis) and its elasticity widely varies with age (Gefen and Dilmoney (2007)). The values of the elastic modulus may vary from 200 $kPa$ to 3000 $kPa$ (Gefen and Dilmoney (2007)).

### 2.3.2 *Breast Models*

Physically realistic modelling of breasts has important issues for medical applications such as cancer diagnosis, image registration, surgical planning and image-guided surgery (Han et al. (2012a)). Since the work of Azar et al. (2001), several breast biomechanical models were proposed (Azar et al. (2002); Rajagopal et al. (2007); del Palomar et al. (2008); Han et al. (2012a); Dufaye et al. (2013); Eder et al. (2014); Eiben et al. (2016)). When developing a constitutive model of the breast, two cornerstones have to be considered: The identification

of the breast parameters and the discretisation of the breast mechanics (Azar et al. (2002); Rajagopal et al. (2007); del Palomar et al. (2008); Eder et al. (2014)). The identification of the breast parameters determines the behaviour of the breast tissues, (i.e.), how it responds to the external stress and how it deforms (strain). The discretisation of the breast mechanics includes the breast geometry and the constitutive equations that describe the boundary conditions, the analysis of stress and strain and the equilibrium equations. An additional aspect concerns the assumptions/simplifications we made. As an example, some models take into account the breast constituents like the fat, glandular tissue, skin and the Cooper's ligaments (Azar et al. (2001)) and other use a more simplified approach to the problems considering glandular tissue and fat as a same homogeneous material del Palomar et al. (2008)).

*Geometry and Parameter Identification*

The description of the deformations depends on the geometry and properties of the materials (Azar et al. (2002); Rajagopal et al. (2007)). The shape and size of a breast vary with people and can change according to posture (e.g. a woman's breast shape differs when she is standing up and when she's lying down (Rajagopal et al. (2007))). Most of these models rely on FEM to compute said deformations (Azar et al. (2002); del Palomar et al. (2008)). Therefore, the most common geometry of breasts consists in digital representations where the breast materials are modelled with meshes, either hexahedral (Azar et al. (2001)) or tetrahedral (del Palomar et al. (2008)). Each element of these meshes has a certain property which tells the type of material it represents, in other words, it tells if an element represents fat, glandular tissue, skin or even a tumour (Azar et al. (2002)).

Palpation was (and still is) used by doctors to find if there is any indication of potential disease (Gennisson et al. (2013)). The main problem with this approach is the lack of information about the internal organs (Gennisson et al. (2013)). Nowadays, technology helps the doctors to formulate the diagnosis, namely MRI, CT, ultrasounds and X-rays. The techniques allowed physicians to visualise in real time the organs' disposition, if there is anything out of the ordinary and even how they behave when perturbations are made upon them (Azar et al. (2002)). Another type of technology concerns the usage of cameras with laser sensors (e.g. Kinect) and scanners to generate 3D models of the breast (Cardoso et al. (2007); Eder et al. (2012); Henseler et al. (2016); Pöhlmann et al. (2017)).

To represent the breast's shape and size (geometry), some models consider simple geometrical primitives like a spherical cap and assumptions are stated on the breast tissues arrangement (Cardoso et al. (2013a)). Other models used images from cameras (stereo or RGB-D) or 3D laser scanners that they processed to reconstructing the breast geometry (Cardoso et al. (2007); Henseler et al. (2011); Eder et al. (2012); Henseler et al. (2014); Wheat et al. (2014); Henseler et al. (2016); Pöhlmann et al. (2017)). However, most models use tools

like MRI and CT scans to obtain images of the breast following with segmentation and classification techniques to map the various materials of the breast (Azar et al. (2002); del Palomar et al. (2008); Eder et al. (2014)).

For the models which use geometrical primitives, the mesh of the breast is generated considering the breast shape in a free stress state, (i.e.), without any forces acting on it including gravity (Cardoso et al. (2013a)). Unfortunately, most of the measured configurations (MRI and ultrasounds which use segmentation and classification techniques as well as the digital reconstructions based on images from cameras and scanners), cannot rely upon the free-stress configuration (Cardoso et al. (2007); Henseler et al. (2016)). Therefore they usually use primitives to generate the mesh adding the effect of gravity on the breast mesh to obtain the final shape and size of the breast (Cardoso et al. (2013a)). Despite removing the complexity of using segmentation and classification to generate the breast mesh by using geometrical primitives, by itself, it is not sufficient to provide information about how the internal tissues are arranged. The same can be said for the models based on images from cameras and 3D scanners. Table 1 shows a comparison between the different types of techniques used to obtain the geometry of the breast.

| Techniques | Processing | Internal | Surface |
|---|---|---|---|
| **Primitives** | Low | No | Approximation |
| **Cameras** | Medium | No | Yes |
| **Medical Imaging** | High | Yes | Yes |

Table 1: Comparison between the 3 types of techniques used to generate the geometry of the breast in terms to the complexity of the breast reconstruction (Low, Medium or High), if they are capable of giving information about the internal arrangement of the breast tissues (Yes or No) and if they obtain detail of the breast contours or they obtain an approximation

The numerical model we consider aims at determining the breast behaviour by minimising the system energy (Rajagopal et al. (2007)). It includes a constitutive relation between stress and strain of the breast tissues according to the established viscoelastic parameters. These parameters vary with the type of material we consider. In Eder et al. (2014) and Krouskop et al. (1998), researchers only rely on the elasticity and stiffness of the materials to describe the elastic behaviour of the tissues. Most sophisticated relations were considered in del Palomar et al. (2008) by assessing how soft tissue material can stretch and compress, namely the elastic moduli which characterises how the material responds to normal stress (Young's modulus), the way it responds to shear stress (shear modulus) and the way it responds to pressure which leads to changes of volume (bulk modulus) (Sigrist et al. (2017)). These three types of elastic moduli are not independent but they are related, for example, when a material stretches in a perpendicular direction from the one exerted upon the material (figure 5), the ratio between compression and stretch is designated as Poisson's ratio and its referenced by some breast models (Azar et al. (2002); del Palomar et al. (2008)). The

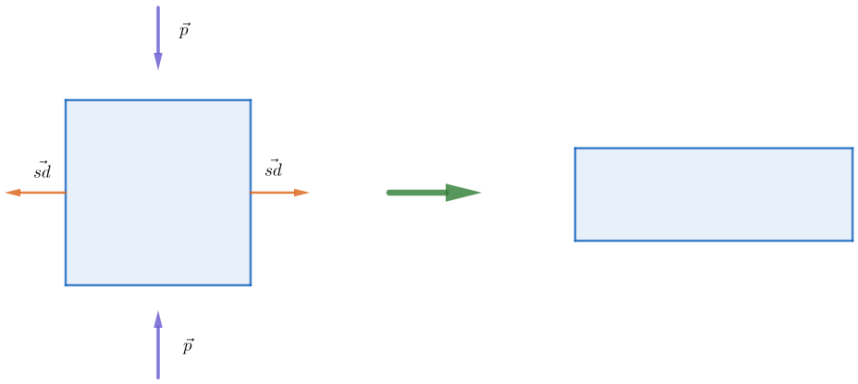



Figure 5: Illustration of how a material can expand in a perpendicular direction $\vec{sd}$ of an applied compression force $\vec{p}$

values of elastic moduli and Poisson's ratio used in most breast models were obtained from studies of the tissues behaviour under different amounts of stress and strain (Azar et al. (2002); Sigrist et al. (2017)). In breast models, these values are used in stress-strain curves together with empirical parameters (e.g. Lamé parameters) which are in turn used to better approximate the tissues behaviour to that of the real tissue (Rajagopal et al. (2007); del Palomar et al. (2008)). The literature proposed two main approaches to assess the moduli values:

- study of a tissue where different amounts of stress are applied and the amount of strain is registered (e.g. Azar et al. (2002))
- analysis of the breast deformation of a patient in different positions (e.g. standing up and in lying prone position as in del Palomar et al. (2008)) or of the breast in different states (e.g. breast undeformed or deformed under plate compression Han et al. (2012a))

In both cases, the great majority of models use the data from MRI, CT, X-rays or ultrasounds and use elastography and other imaging techniques to obtain said parameters (Azar et al. (2002); del Palomar et al. (2008); Sigrist et al. (2017)) although in the case of deformation analysis some models use experimental measurements (Cardoso et al. (2013a)). Elastography is a set of imaging techniques that map the elastic properties of tissues using mainly MRI and ultrasounds (Mariappan et al. (2010); Gennisson et al. (2013); Sigrist et al. (2017)). These techniques map the stiffness of a tissue when responding to external perturbations. This mapping is done by estimating the strain in the tissue under a quantifiable amount of stress or by the imaging of waves (shear and mechanical) whose propagation depends on the tissue stiffness (Gennisson et al. (2013)). When analysing a breast deformation, most models use an iterative approach where they use a set of initial parameters deriving from

literature and then try to refine those parameters until the digital breast is a good approximation to the real breast (del Palomar et al. (2008); Han et al. (2012a)). The comparison can be done through the use of markers (del Palomar et al. (2008)) or the usage of images from either CT scans and MRI (Han et al. (2012a)) as well as images from RGB-D cameras like the Kinect and other 3D scanners (Pöhlmann et al. (2017)). There are some works that apply Machine Learning (ML) in the estimation of these parameters but there are few studies on this particular subject (Martínez-Martínez et al. (2017)).

*Breast Mechanics*

Biomechanical breast models have been employed in several medical and computer science fields for simulating mechanical deformations of the female breast including image registration, biopsies, compression in mammograms and breast surgery planning (Eder et al. (2014)). Most models opt to consider mechanical transformations employing continuum mechanics with FEM equipped with constitutive equations (Rajagopal et al. (2007); Eder et al. (2014)). Regarding the volumetric structures, no proposed model considers every aspect of the breast anatomy, (i.e.), they do not take into account every single element of the breast like number and size of glandular lobes and ducts because it greatly increases the complexity of the model. So due to this great difficulty in evaluating and simulating the complexity of the breast biomechanics, most researchers assumed simplified models of the breast (Azar et al. (2002); Rajagopal et al. (2007); del Palomar et al. (2008)). The degree of simplifications varies from models with greater complexity Azar et al. (2001), where fat and glandular tissue are considered distinct materials, to simplify del Palomar et al. (2008), where the breast is assumed as a homogeneous mass with elasticity coefficients being an average value from the amount of fat and glandular tissue.

One of the first studies on the viscoelastic properties of the breast was presented by Krouskop et al. (1998) where they analysed the stress-strain curve of *ex vivo* fat and glandular tissues. With the cases they analysed they concluded that the breast tissues behaviour depended mainly on their elastic properties, (i.e.), the viscous properties were negligible (Krouskop et al. (1998)) and the relationship between stress and strain could be considered linear. Wellman et al. (1999) also reached a similar conclusion, where the breast constituents (fat and glandular tissue) behave as piecewise linear elastic materials. Both models were used to analyse the behaviour of the breast when under compression (Krouskop et al. (1998); Wellman et al. (1999)).

One of the first models to assist in a surgical procedure was presented in Azar et al. (2001) with a continuation in Azar et al. (2002). Their model aims at helping the surgeon to guide a needle in live biopsy of tumour cells. The biopsy started with the patient lying prone and placing the breast between 2 plates that would compress the breast an make the process of inserting the needle easier. After the needle is inserted, it pushes on until it reaches

the suspect tumour zone and retrieves a portion of that tissue for analysis. To reproduce these steps digitally, they considered the breast as a deformable domain with nonlinear material properties. Moreover, they used exponential curves to describe the stress-strain properties of the breast tissue (Azar et al. (2001)). The model assumes that the viscous component could be considered negligible and the breast as an isotropic, homogeneous and incompressible (Azar et al. (2001, 2002)) material. The model's density is higher on the zones of interest (closest to the needle) and decreases away from that point. The biopsy has 2 different moments of deformation. A big deformation that occurs when the breast is compressed between 2 plates, and smaller deformations when the needle punctures the skin and travels through the breast. To this end, a specific elastic model was proposed where the big deformations were approximated with a set of small deformations steps satisfying the standard linear mechanics equations. They argue that this option has the advantage of using small strain formulation which is simpler, easier to implement and produces for fast calculations (Azar et al. (2001, 2002)). This model employs FEM and the generated mesh is made of hexahedrons for the volumetric cells and triangular shapes for the skin. The model differentiates fat and glandular tissues which are represented by cells, and they consider the effect of the Cooper's Ligaments in the behaviour of the fat tissues (Azar et al. (2002)). The surgical procedure is modelled by the boundary conditions that set the degrees of freedom of the nodes when in contact with the compression plates and the zone of the breast attached to the torso (Azar et al. (2001, 2002)). To mimic the surgical procedure, the model simulates the forces that will cause the meshes nodes displacement and by a sequence of external forces to update the displacements vector by successive iterations. The displacements are restricted by the boundary conditions and the internal strain is evaluated for each cell ($e$) by solving a triple integral energy density over its volume given by

$$K_e = \int\int\int_{V_e} B^T D B dV \tag{21}$$

where $D$ is a matrix that contains the elastic properties of an element $e$ and $B$ is the tensor that relates nodal strains to displacements and incorporates the Cauchy infinitesimal strain tensor (Azar et al. (2001)). In other words, this strain energy function relates the nodal strain and stress resulting from the disturbance of the internal forces by the external actions of compressing the breast and the action of the needle (Azar et al. (2001, 2002)). After all displacement iterations being calculated, the final configuration of the breast is achieved (Azar et al. (2002)). At the time and with all simplifications and optimisation's, they were able to perform simulations that took around 30 minutes (Azar et al. (2001)).

More recently, a general model was conducted by del Palomar et al. (2008). Their purpose was to carry out simulations in any kind of medical procedure without major changes apart from the elastic coefficients of the tissues (del Palomar et al. (2008)). For that purpose, they considered a simple model where the fat and glandular tissues are represented with an

average value of the mechanical properties depending on their respective proportion, (i.e.), a patient with more fat tissue is less stiff and a patient with more glandular tissue (del Palomar et al. (2008)). Moreover, the effect of Cooper's ligaments was neglected and the skin considered as a Neo-Hooken material. They used CT scans data to create a volumetric structure with tetrahedrons as cells and triangles that represent the skin (del Palomar et al. (2008)). Instead of using an exponential curve to describe the stress-strain of the tissues like in Azar et al. (2002), they opted to use a quasi-incompressible, isotropic and hyperelastic material known as Neo-Hookean material (del Palomar et al. (2008)). To test this model they evaluated the deformation of the breast tissues *in vivo* with the patient in two positions: lying in a prone position (reference) and standing up (final position). To fit the material properties of the breast tissues they use the CT images of the patient in a vaulted position considering the breast in a free gravity state, moving after to a vertical position and then applying the gravity field (del Palomar et al. (2008)). To evaluate the displacements they solve the strain energy function $\psi$ based on the Neo-Hookean material which was given by:

$$\psi(C) = \psi_V(J) + \psi_{iso}(C) = \frac{1}{D} + C_1(I_1 - 3) \tag{22}$$

where $\psi_{vol}(J)$ and $\psi_{iso}(C)$ are given scalar-valued functions of the Jacobian $J = det(F)$ which describe the volumetric and distortional responses of the material, with $C$ being the right Cauchy deformation tensor ($C = J^{2/3}F^TF$) and $F$ the deformation gradient tensor (del Palomar et al. (2008)). $D$ and $C_1$ are empirical constants that are adjusted to the tissues properties and $I_1$ is the first invariant of $C$ which is the trace of the Jacobian matrix (del Palomar et al. (2008)). The simulation of the tissue ends when the breast displacements lead to a configuration where the strain energy value goes below a certain threshold (del Palomar et al. (2008)).

In Rajagopal et al. (2008) a similar approach is proposed by assuming hyperelastic breast tissues but they considered that the breast of a patient in a prone position is affected by stress (which was neglected by del Palomar et al. (2008)). They have estimated the breast is in a free stress state by acquiring images with MRI of the patient's breast when immersed in water (Rajagopal et al. (2008)). To generate the breast geometry they used cubic-Hermit basis functions to fit the breast surfaces and for the fat and glandular tissues, they used segmentation techniques on the MRI images to obtain the contours of the tissues and then map the values to the finite element mesh (Rajagopal et al. (2008)). In order to produce a more accurate representation of the breast mechanics, they considered the area of the breast attached to the torso as a boundary condition by prescribing a no displacement condition for the nodes of that array (Rajagopal et al. (2008)). The simulation starts with the breast

in a free stress state, and then, gravity field is applied. The simulation solved the strain energy function $W$ based on the Neo-Hookean hyperelastic material is given by:

$$W = C_1(I_1 - 3) \tag{23}$$

where the properties $C_1$ and the invariant $I_1 = trace(F^T F)$ have the same purpose as in equation 22, (i.e.), the $I_1$ is a measure of strain and $C_1$ is the user-fixed parameter of elasticity (Rajagopal et al. (2008)). The strain energy equation satisfies the equilibrium equations deriving from the mass, linear and angular momentum conservation. The simulation ends when the strain energy functional is minimised below a certain threshold value (Rajagopal et al. (2008)). Similar models using hyperelastic materials were also derived to detecting lesions and tumours (Zhong and Peters (2007); Bounaïm et al. (2007); Stewart et al. (2011)).

A complete model for simulating surgical procedures of breast reduction was proposed by Cardoso et al. (2013a). The main goal was to propose a solution to handle the general problem of plastic surgery (Cardoso et al. (2013a)). They considered the breast as a hyperelastic and compressible Neo-Hookean material. They argued that the elastic properties of the breast cannot be readily deduced from the elastic properties of its tissues. They differ from the most common notion that the breast tissues are incompressible due to the fact that in some breast reduction surgeries where only part of the skin was removed, the breast's volume was reduced (Cardoso et al. (2013a)). Just like del Palomar et al. (2008) they also consider the fat and glandular tissue as one mass but they consider the skin as having different mechanical properties and that for that they need to be modelled separately (Cardoso et al. (2013a)). Moreover, they take into account the Chassaignac space which is a physiological area situated between the breast and the torso which plays a special role in breast mobility. The geometry of the breast is created using a spherical cap where the surface is subdivided into triangles and the internal volume is subdivided into tetrahedrons. The cap section represents the Chassaignac space, the spherical surface represents the skin, and the internal volume represents the fat and glandular tissue (Cardoso et al. (2013a)). The breast reduction surgery was solved with specific boundary conditions for the Chassaignac surface whose elements are connected to the torso via a spring-like force with restrictive movement. In the same way, the cut surfaces are linked via an additional constraint in order to simulate the suturing (Cardoso et al. (2013a)).

To perform the breast reduction simulation, the mesh is first generating according to elastic parameters $\lambda$ and $\mu$ and the geometrical parameters that characterise the cuts to be performed (Cardoso et al. (2013a)). Under the null gravity condition, superficial forces corresponding to the suturing process are applied and the energy $I$ is minimising in function

of the displacements $f$ of the body $(B)$ and including the effect of the Chassaignac space $(chass)$, namely

$$I(F) = \int_B W_B + W_{chass} \tag{24}$$

where $W_B$ represents the strain energy function of the breast tissues and $W_{chass}$ is the mass-spring system positioned at the base of the breast (Cardoso et al. (2013a)). The strain energy function $(W)$ is based on the Neo-Hookean material for the breast tissues and is given by

$$W_B(F) = \frac{\mu}{2}(tr(FF^T) - 3 - 2ln(det(F))) + \frac{\lambda}{2}(det(F) - 1)^2 \tag{25}$$

where $F$ is the deformation gradient tensor (or Jacobi Matrix), $\mu$ and $\lambda$ are Lamé parameters and are different for the skin and for the fat+glandular tissue (Cardoso et al. (2013a)). The effect of the Chassaignac space is modelled by the spring system ernergy

$$W_{chass}(p) = \int_{chass} c(p)(f(p) - p)^2 dp \tag{26}$$

where $c(p)$ is a given function (Cardoso et al. (2013a)) that represents the non-linear spring stiffness.

Eder et al. (2014) tested several breast models and simulations (some which were already presented above) in order to assess their adequacy to real-life cases. Different mechanical behaviour the breast tissues were considered: linear elastic (Krouskop et al. (1998)), exponential elastic (Azar et al. (2001, 2002)) and hyperelastic (del Palomar et al. (2008); Rajagopal et al. (2008)). Regarded to hyperelastic materials, they presented comparisons with the Neo-Hookean model and more general Mooney-Rivlin model (Maurel et al. (1998)). They used the elastic parameters estimated by those models (the empirical parameters like $C_1$ in the Neo-Hookean model) and showed that the Neo-Hookean models provide better approximations, especially with the model proposed by Rajagopal et al. (2008). It is also reported in Eder et al. (2014) the lack of deformation accuracy and heavy computation time, (i.e.), a trade-off between time and quality is achieved by using meshes with fewer nodes (Eder et al. (2014)).

Eiben et al. (2016) presented an extension of the previous model where they take into account the additional deformations when the breast is in contact with the MRI coil during the MRI image acquisition. They also argued that defining appropriate prescribed displacements usually requires manual pre-processing (as mentioned in Azar et al. (2002); del Palomar et al. (2008); Rajagopal et al. (2008)) which is undesirable in a clinical context (Eiben et al. (2016)). So they acquire images of the breast when the patient is in prone and supine positions and they use FDM to analyse the displacements which do not require an explicit mesh generation step just like what happens for the more common FEM models (Eiben et al. (2016)). With FDM they split the displacements into small time steps which they solve using

the strain-energy functions based on the Neo-Hookean material. The boundary conditions include a technique they called slave nodes that simulate the effect of what Cardoso et al. (2013a) point as the Chassaignac space (Eiben et al. (2016)). The registration process starts with an initial breast configuration which is obtained with the MRI images. Then they remove the gravity field and use FDM to estimate the breast under a free stress state for both supine and prone positions (Eiben et al. (2016)). Then they start a refinement process to obtain the breast elastic coefficients which use the MRI images to apply a correction (Eiben et al. (2016)). They used the NiftySim GPU based package to solve nonlinear finite element problems including the simulation of soft tissue biomechanics (Johnsesn et al. (2015); Eiben et al. (2016)).

Recently, new models and algorithms are developed in order to accelerate the breast mechanical parameters estimation taking advantage of the GPU and of ML algorithms, but are still in an experimental stage (Martínez-Martínez et al. (2017)).

## 2.4 DISCUSSION

The breast mechanical behaviour is a complex system to reproduce numerically. The various studies point out that modelling every breast tissue and mechanics is still intractable (Maurel et al. (1998); Azar et al. (2002); del Palomar et al. (2008); Rajagopal et al. (2008); Eder et al. (2014)). Every model in literature is based on a set of simplification. Curiously, some simpler models were able to provide better results than models with a higher level of detail (Eder et al. (2014)). That means that such assumptions are relevant and provide representative solutions while reducing the computational effort. To put in a nutshell, the different breast models proposed over the years have some elements in common such as:

- FEM - most models use FEM to evaluate the tissues deformations (with some examples using FDM instead like in Eiben et al. (2016))
- free stress state - more recent breast models showed the importance of considering the shape and size of the breast and its tissues arrangement in a free stress state
- viscoelasticity - most models neglect the viscous behaviour of the tissues and consider only their elastic behaviour
- Imaging - most models acquire images from MRI and CT scans and use those images to generate the breast's geometry

These models showed that they can reproduce the behaviour of breasts with some accuracy. However, most of them are very complex and their application is sometimes very restrictive. For example, to identify internal lesions and tumours, medical imaging equipment like MRI is necessary to allow surgeons to locate them and then remove them. But the same

cannot be said for breast reduction surgeries which are usually planned by marking the patient, although the use of such equipment could be used (Cardoso et al. (2013b)). Another problem concerns the usage of this type of equipment which in many places is limited and can lead to long wait periods that in some cases can reach 90 days (Maia (2016); Carriço (2017)).

The study conducted by Eder et al. (2014) showed that the distinction between fat and glandular tissue isn't a key factor in terms of a models accuracy but the right choice of the elastic parameters of the materials is. Some models like the one proposed by Azar et al. (2002); del Palomar et al. (2008) used elastic parameters from the literature, but the model proposed by Cardoso et al. (2013a) offered a new way to look into this problem which passed by estimating these parameters for each patient because every person is different from each other and the elastic properties of the breast can change with time (Cardoso et al. (2013a); Poplack et al. (2004)). The breast in a free stress state used as the reference state helps very much to generate the geometry of the breast using primitives because what has usually considered the natural shape and size of the breast is a representation of the breast under the Earth's gravitational field.

Due to the increasing complexity of the models and the breast shape, a trade-off between accuracy and computation effort has to be reached. To obtain better results additional computational resources are required and the relation is not linear (the computational effort increased more rapidly than the accuracy). There are software solutions that use the GPU to solve problems with FEM (Johnsesn et al. (2015); Talbot et al. (2015)) recently use in Eiben et al. (2016). The usage of accelerating hardware and software can contribute to minimise the time constraint of these models. The idea proposed by Martínez-Martínez et al. (2017) to use ML algorithms to create a breast model can also have the potential to minimise the computational costs and is something to take into consideration in the development of future models.

# 3

# BREAST MODEL

To support surgeons take better informed decisions, numerical models enable the simulation of virtual surgery operations in which one can visualise the post-operated breast and therefore analyse several scenarios. A new breast model is presented along with a numerical tool capable of simulating breast reduction surgeries. The model considers performing sutures with asymmetric geometrical configurations as well as viscoelastic parameters based on the Neo-Hookean mechanical model. The research was published in Lopes et al. (2017a). The parameters are obtained solving an inverse problem presented in chapter 4. One also carried out the post-surgical breast simulation to analyse its behaviour when under gravity. Moreover, the discretisation of the mathematical problem, as well as a series of benchmarks are presented in this chapter to assess the ability to reproduce the surgeries.

## 3.1 NUMERICAL MODEL

The breast is a combination of various soft tissues whose behaviour presents high nonlinear aspect under stresses. Due to the physiological complexity of the breast, assumptions such as homogeneous and isotropic material are stated to simplify the model at a macroscopic scale. Moreover, following (del Palomar et al. (2008); Rajagopal et al. (2008); Cardoso et al. (2013a); Eiben et al. (2016)) the reference geometry (coordinate space) is the breast in the free stress state. The breast model is composed of the following ingredients:

- Geometry
- Mechanics
- Boundary Conditions

It is first assumed that the reference geometry in a free stress state is a spherical cap. Notice that the breast shape obtained with equipment like MRI is affected by the gravity and submitted to the compression of plates. It yields an additional deformation of the tissues with respect to the stress-free domain (Eiben et al. (2016)). Therefore, the spherical cap

assumption is a good approximation to the breast's shape and size without gravity and pressure.

The biomechanical model is assumed to be a nonlinear hyperelastic Neo-Hookean material. The good results presented in literature Eder et al. (2014) support the idea that the Neo-Hookean model is capable of describing with accuracy the mechanical behaviour of the breast. Cardoso et al. (2013a) argued that despite the incompressible or quasi-incompressible nature of the breast tissues, the reduction of the breast's volume happens in breast lift's where only a portion of skin was removed. This suggests that, in practice, the mammary gland can compress. Therefore the model has to consider the breast tissues as compressible materials. From a computational point of view, a FEM discretisation will be carried out to evaluate the tissues deformation using tetrahedron elements meshes.

Boundary conditions are prescribed to fix the breast that to the chest wall and a suturing condition is given on the sections of the breast that will be joined during the surgical procedure (also known as knitting conditions). However, the model the breast to be in a free gravity state because the effect of gravity does not affect the suturing process. Nonetheless, to determine the breast shape after the surgery, Cardoso et al. (2013a) and Eiben et al. (2016) showed, the importance to consider the mobility of the breast and the effect of the Chassaignac space (Cardoso et al. (2013a)) when applying the gravity.

### 3.1.1 *Geometry*

The domain definition of the breast, *i.e.* its geometry, is the first step in the modelling. Particularly, breast reduction surgery is a procedure used to remove excess of fat, tissue, and skin from the breasts in order to achieve a breast size in proportion with the body and to alleviate the discomfort associated with overly weight breast. Therefore, the suturing process is very complex from a geometrical point of view, involving contact areas that are joined into a unknown final surface. One has to carefully define the interfaces of the breast and its parametrisation to enable the description of the future sutured interface.

There exist several techniques to perform breast reduction like the inverted-T and vertical scar techniques (Zoumaras and Lawrence (2008); Cardoso et al. (2013b)). While the vertical scar technique can be simple to implement, in practice they cannot be performed on very large breasts (Cardoso et al. (2013b)) and so, this model allows the implementation of both. In both techniques the surgeon starts by planning the new position of the nipple and the frontal cuts marking them on the breast like what is shown in figure 6. The inverted-T technique includes rear cuts that will align in the infra-mammary fold (Zoumaras and Lawrence (2008); Cardoso et al. (2013b)). At last, the skin is closed with stitches to initiate the natural suturing process.

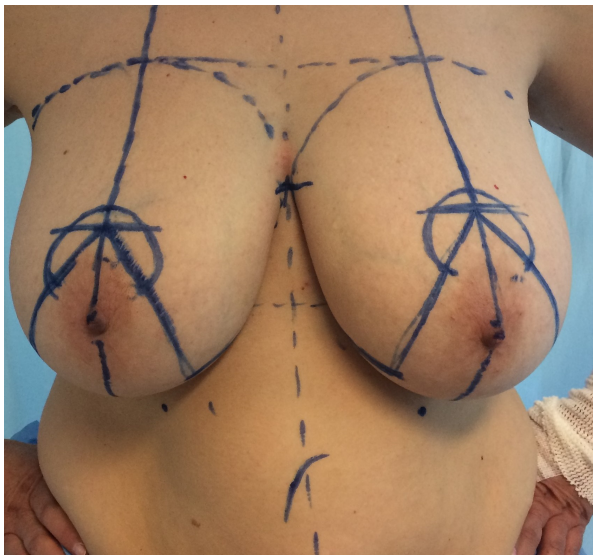


Figure 6: Breast reduction surgery planning marks on a patient. Photography taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

From a mathematical point of view, $\Omega_{cut}$ represents the domain corresponding to the incised breast and where one prescribes the conditions to perform the suturing (Figure 7), *i.e.* the breast after the cut but before exerting the forces to join the two parts.

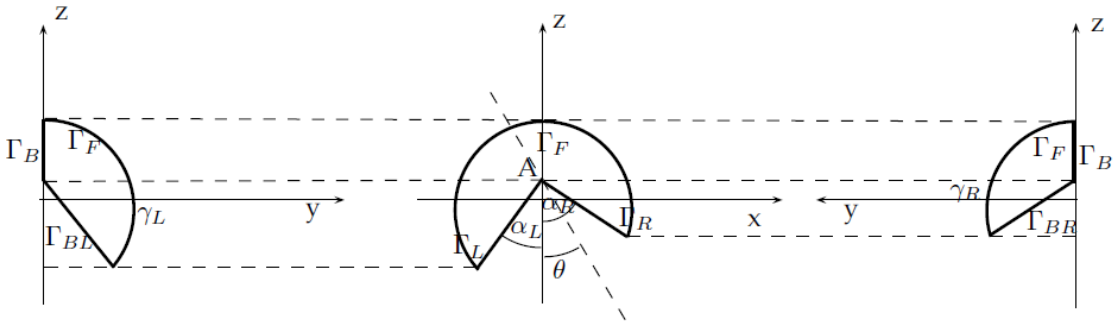



Figure 7: Scheme of breast reduction surgery: the middle panel is the frontal view while the left and right panels report the parameters for the left and right parts of the breast

Figure 7 displays three technical views of the incised breast with the boundaries being described as follows:

- $\Gamma_B \rightarrow$ back side of the breast that is fixed to the chest
- $\Gamma_F \rightarrow$ surface associated to the skin of the breast
- $\Gamma_L, \Gamma_R \rightarrow$ left and right cuts performed during the frontal incision
- $\Gamma_{BL}, \Gamma_{BR} \rightarrow$ left and right cuts performed during the rear incision
- $A \rightarrow$ point of the front of the breast where the first incision starts and which corresponds to the new nipple position

- $\gamma_0 = \Gamma_L \cap \Gamma_R \rightarrow$ line between point $A$ and the chest
- $\gamma_L = \Gamma_F \cap \Gamma_L \rightarrow$ frontal left suturing line
- $\gamma_R = \Gamma_F \cap \Gamma_R \rightarrow$ frontal right suturing line
- $\gamma_{BL} = \Gamma_F \cap \Gamma_{BL} \rightarrow$ backward left suturing line
- $\gamma_{BR} = \Gamma_F \cap \Gamma_{BR} \rightarrow$ backward right suturing line
- $\theta \rightarrow$ angle of the suturing plane
- $\alpha_L, \alpha_R \rightarrow$ left and right angles of the frontal incision

The breast is partially covered with skin characterised by the boundary surface $\Gamma_F$. The frontal incision is performed between two interfaces denoted by $\Gamma_L$ and $\Gamma_R$ respectively sharing the line $\gamma_0 = \Gamma_L \cap \Gamma_R$ passing at point A, orthogonal to the trunk body. A second rear incision is performed between the breast and the trunk such that a portion of the breast remains in contact with the chest wall via the interface $\Gamma_B$ while $\Gamma_{BL}$ and $\Gamma_{BR}$ are the left and right surfaces respectively deriving from the rear incisions. Boundary lines $\gamma_L$ and $\gamma_R$ in red in figure 8 correspond to the suturing line that the surgeon will join while $\gamma_{BL}$ and $\gamma_{BR}$ are the rear incision lines.

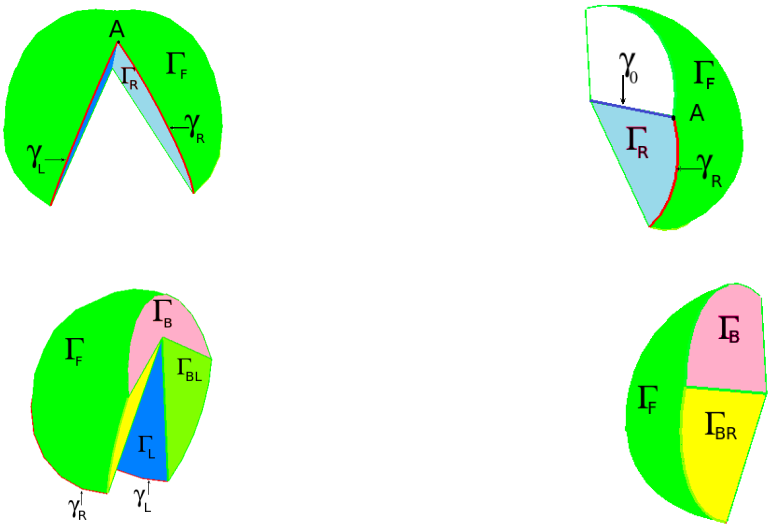



Figure 8: Full breast (left) and half breast (right) in a 3D frontal view (top row) and rear view (bottom row)

### 3.1.2 *Mechanics*

The breast behaviour is described with a nonlinear hyperelastic Neo-Hookean compressible model. The surgery simulation neglects the gravity force assuming that the suturing

process is independent of the external field. The breast is in a "at rest" state and the suturing process will lead to the breast deformation. The problem turns to minimise the associated energy functional under restrictions: one determines the new position vector field $f : p \to f(p)$ which minimises

$$\mathcal{J}_{cut}(f) = \int_{\Omega_{cut}} W_{br}(\nabla f(p))dp + \int_{\Gamma_F} W_{sk}(\nabla_{||} f(p))dS_p \tag{27}$$

where $W_{br}$ and $W_{sk}$ are the expressions for the volume (mammary gland) and surface (skin) strain energy densities respectively. As the tissues behaviour employ the Neo-Hookean model the expressions $W_{br}$ and $W_{sk}$ are given by

$$W_{br}(F) = \frac{\mu_{br}}{2}\left(tr(FF^T) - 3 - 2ln(det(F))\right) + \frac{\lambda_{br}}{2}(det(F) - 1)^2, \tag{28}$$

where $F = \nabla f$ is the Jacobian matrix of $f$ and $(\lambda_{br}, \mu_{br})$ are the Lamé parameters for the breast and

$$W_{sk}(F_{||}) = \frac{\mu_{sk}}{2}\left(tr(F_{||}F_{||}^T) - 2 - 2ln(det(F_{||}))\right) + \frac{\lambda_{sk}}{2}(det(F_{||}) - 1)^2, \tag{29}$$

where $F_{||} = \nabla_{||} f$ is the Jacobian matrix of the superficial (skin) displacement $f_{||}$ and $(\lambda_{sk}, \mu_{sk})$ are the Lamé parameters for the skin. $f(\Omega_{cut})$ will be the new breast configuration after the suturing that one has to determine. In chapter 4 one details a technique to determine the breast and skin mechanical parameters.

### 3.1.3 *Boundary Conditions*

The breast reduction surgery boundary conditions (also known as knitting conditions) are given by the equation 27 representing the constraints displacement for the suturing process. These conditions read:

$$f(p) = p, \qquad \text{in } \Gamma_B, \tag{30}$$

$$b(f(p)) = 0, \qquad \text{in } \Gamma_{BL} \cup \Gamma_{BR}, \tag{31}$$

$$\Pi_\theta(f(p)) = 0 \qquad \text{in } \Gamma_R \cup \Gamma_L, \tag{32}$$

$$f(p) - f(g(p)) = 0, \qquad \text{in } \gamma_R, \tag{33}$$

where $\Pi_\theta$ is the equation of the plane where the suture of the left and right frontal cuts will take place. Indeed, this model assumes that for a small asymmetry, (i.e.), $\alpha_L \approx \alpha_R$, the suturing interface may be approximated by the medium plane. Of course, for large asymmetries, this assumption is no longer valid and the problem of the location of the final interface turns to be more complicated.

Condition 30 guarantees that the back interface remains in contact with the trunk while condition 31 states that the rear cut interfaces ($\Gamma_{BL}$ and $\Gamma_{BR}$) belongs to the trunk plane to model the suturing of the breast with the body. To achieve the suturing process with the frontal cut, two boundary conditions are prescribed. Condition 32, states that the junction of the left ad right incised sections ($\Gamma_L$ and $\Gamma_R$ respectively) occurs in a plane $\Pi_\theta$ characterised by the angle $\theta$ which is prescribed as the mean value of the left and right angles. Condition 33 introduces the mapping $g : \gamma_R \rightarrow \gamma_L$ associated to the suturing which provides the point-to-point correspondence between the left and right side of the cut.

## 3.2 DISCRETISATION OF THE MODEL

### 3.2.1 *Mesh and Notations*

It is assumed that the breast geometry reference domain, *i.e.* at rest without any stress, is obtained from a spherical cap with the performed cuts which are made from geometrical parameters as the ones in Figure 9.

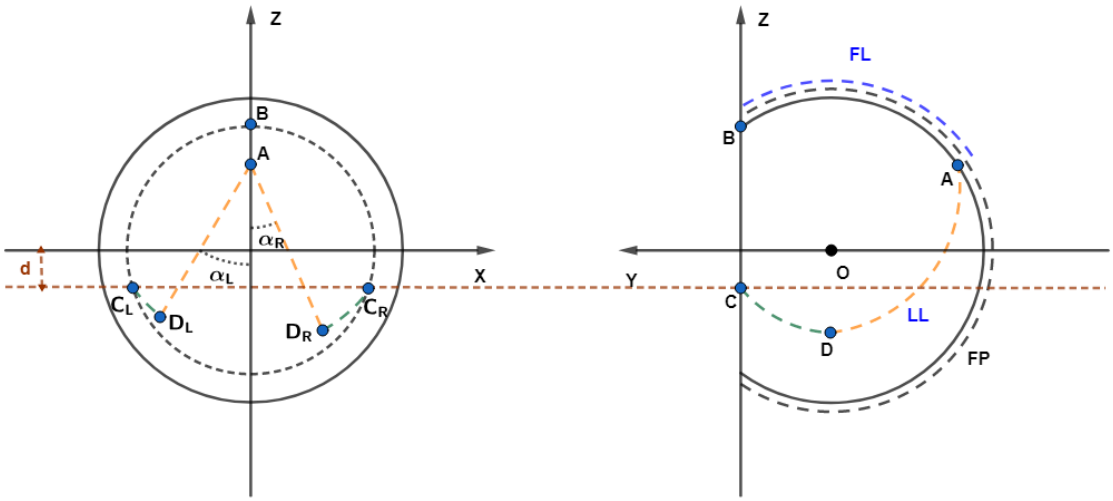



Figure 9: Incised breast geometry $\Omega_{cut}$ with the incision parameters. On the left is the breast frontal view while on the right is the breast side view.

$FL$ represents the distance between the top of the breast to the beginning of the incision (point $A$) and its value can vary from the breast frontal perimeter ($FP$) to a value where the height of point $A$ would be inferior to the height of point $B$ ($A_z < B_z$). $LL$ is the length of the lateral incisions marks that ends at point $D$ which can be in any part of that mark from the point $A$ to $C$. Point $C$ is a point in the base of the breast which is determined by the value distance $d$ from the spherical cap middle section ($xOy$ plane in figure 9). The

incision marks depend on the frontal cut angles $\alpha_L$ for the left angle and $\alpha_R$ for the right angle which will lead to find the positions of the $D_k$ and $C_k$ points where $k \in \{L, R\}$.

The breast is discretised by a tetrahedral mesh with the displacements being represented by a piece-wise continuous representation of the new position $f$. To this end, $\mathcal{T}$ denotes a mesh of $\Omega_{cut}$ constituted of $I$ non-overlapping tetrahedron cells $\tau_i$, $i = 1, \ldots, I$, and $N$ vertices $P_n = (P_{nx}, P_{ny}, P_{nz}) \in \mathbb{R}^3$, $n = 1, \ldots, N$. Moreover, $T_k$, $k = 1, \ldots, K$, represents the faces of the tetrahedrons of the mesh that belong to $\Gamma_{\mathrm{F}}$. Quantities $|\tau_i|$ and $|T_k|$ represent the volume and the area of the cell and the triangle respectively. In local indexation, the vertices of $\tau_i$ are denoted by $P_{ij} = (P_{ijx}, P_{ijy}, P_{ijz}) \in \mathbb{R}^3$, $j = 1, 2, 3, 4$, while the vertices of $T_k$ are denoted by $P_{kj} = (P_{kjx}, P_{kjy}, P_{kjz}) \in \mathbb{R}^3$, $j = 1, 2, 3$.

To discretise function $f$, one associates to each node $P_n$ an approximation $f_n \approx f(P_n)$ and denotes by $f_h$ the usual continuous piece-wise linear function while vector $\Phi_h = (f_{nx}, f_{ny}, f_{nz})_p$ collects the $3N$ components of the new positions. Vector $\Phi_h$ corresponds to the new configuration but the entries are not the unknowns of the problem since some of them are characterised by the boundary or the suturing conditions. Therefore, one extracts a sub-vector $X_h$ of $\Phi_h$ which only contains the unknown values that are used in the minimisation process while the boundary conditions define an operator

$$X_h \to \Phi_h = \mathcal{B}(X_h) \tag{34}$$

which provides the other entries to complete vector $\Phi_h$.

### 3.2.2 *Boundary and Suturing Conditions*

Condition 30 yields that $f_n = P_n$ for any $P_n$ on $\Gamma_{\mathrm{B}}$ while relation 31 implies that for any $P_n$ on $\Gamma_{\mathrm{BL}} \cup \Gamma_{\mathrm{BR}}$ one sets $f_{ky} = 0$ to enforce the suturing of the rear cut on the trunk plane $y = 0$. To apply constraint 32, for any $P_n \in \Gamma_{\mathrm{R}} \cup \Gamma_{\mathrm{L}}$, $f_{ny}$ and $f_{nz}$ are used as unknowns, (i.e.), $f_{ny}$ and $f_{nz}$ are entries of $X_h$. Therefore, one computes the last component with the relation $f_{nx} = \tan(\theta) f_{nz}$.

Constraint 33 is the main condition since it describes the surgical procedure. For a node $P_n \in \gamma_{\mathrm{L}}$, $f_{ny}$ and $f_{nz}$ are entries of $X_h$ and one computes the last component with $f_{nx} = \tan(\theta) f_{nz}$ thanks to condition 32. Assuming now that function $g : \gamma_{\mathrm{R}} \to \gamma_{\mathrm{L}}$ is given, one proceeds in the following way at the discrete level:

- For a node $P_n \in \gamma_{\mathrm{R}}$, the edge $e = [P_\alpha P_\beta] \in \gamma_{\mathrm{L}}$ is determined such that $g(P_n) \in e$;
- One then defines the unique barycentric coordinates such that $\lambda_{n\alpha} P_\alpha + \lambda_{n\beta} P_\beta = g(P_n)$ (see Figure 10).

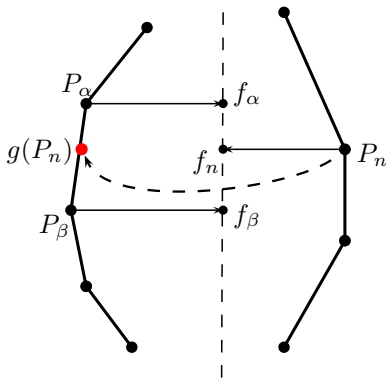



Figure 10: Discretisation of function $g$. For any point $P_n \in \gamma_R$, one determines an approximation $g(P_n) \in \gamma_L$ between two nodes $P_\alpha$, $P_\beta$ of an edge belonging to $\gamma_L$. Barycentric coordinates then derive from the relative position of $g(P_n)$.

To preserve the relative position of $f_n$ with respect to $f_\alpha$ and $f_\beta$ in the final configuration, one sets

$$f_n = \lambda_{n\alpha} f_\alpha + \lambda_{n\beta} f_\beta.$$

The barycentric coordinates are computed once as a pre-processing step and none of the coordinates of $P_n$ is an entry of $X_h$. For any $P_n$, the choice of the associated edge $e$ and the barycentric coordinate is equivalent, at the discrete level, to completely define function $g$.

### 3.2.3 *Strain Energy*

The energy functional given by equation 27 is approximated with the following discrete representation

$$J_h(X_h) = \widehat{J}_h(\mathcal{B}(X_h)) = \widehat{J}_h(\Phi_h) = J_h^1 + J_h^2 \tag{35}$$

constituted of the volume energy $Jh^1$ and the surface energy $Jh^2$.

For a new configuration characterised by the approximation $p \in \Omega_{cut} \to f_h(p) \in \mathbb{R}^3$ and stored in vector $\Phi_h$, the internal energy on tetrahedron $\tau_i$ is given by

$$W_{\tau_i} = |\tau_i| \left( \frac{\mu}{2} \Big[ tr(F_i F_i^T) - 3 - 2ln(det(F_i)) \Big] + \frac{\lambda}{2} \Big[ det(F_i) - 1 \Big]^2 \right), \tag{36}$$

where $F_i$ is the 3 $x$ 3 matrix solution of the linear system

$$f_{i2} - f_{i1} = F_i(P_{i2} - P_{i1}), \quad f_{i3} - f_{i1} = F_i(P_{i3} - P_{i1}), \quad f_{i4} - f_{i1} = F_i(P_{i4} - P_{i1}).$$

The total internal energy is then approximated by

$$J_h^1 = \sum_{\tau_i \in \Omega_{cut}} W_{\tau_i}. \tag{37}$$

The surface energy is given by the surface integrals of the functional. The discrete piece-wise linear function $f_h$ transforms a triangle $T_k$ with vertices $OAB$ into a triangle $T_k'$ with vertices $O'A'B'$. Since the translation and the rotation do not change the stress due to the deformation, one assumes that $O'B'$ is collinear to $OB$ and $A'$ belongs to the same plane than triangle $OAB$. Function $f_{||}$ is a two-dimensional function locally given by $f_{||}(O) = O$, $f_{||}(A) = A'$, $f_{||}(B) = B'$. The Jacobian matrix of $f_{||}$ is the constant matrix

$$Jf_{||} = \mathcal{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

To determine the matrix, one writes

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} \|OB\| \\ 0 \end{bmatrix} = \begin{bmatrix} \|O'B'\| \\ 0 \end{bmatrix}, \quad \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} \|OA\| \cos(\alpha) \\ \|OA\| \sin(\alpha) \end{bmatrix} = \begin{bmatrix} \|O'A'\| \cos(\alpha') \\ \|O'A'\| \sin(\alpha') \end{bmatrix},$$

where $\alpha = \angle(OA, OB)$ and $\alpha' = \angle(O'A', O'B')$. The first linear system gives $c = 0$ and $a = \frac{\|O'B'\|}{\|OB\|}$. Substituting these expressions in the second linear system one obtains

$$d = \frac{\|O'A'\| \sin(\alpha')}{\|OA\| \sin(\alpha)}, \quad b = \frac{\|O'A'\| \cos(\alpha') - \frac{\|O'B'\|}{\|OB\|} \|OA\| \cos(\alpha)}{\|OA\| \sin(\alpha)}.$$

The superficial energy on triangle $T_k$ for the skin is then given by

$$W_{T_k} = |T_k| \left( \frac{\mu_{\text{sk}}}{2} \Big[ \text{tr}(Jf_{||} Jf_{||}^t) - 2 - 2 \ln(\det(Jf_{||})) \Big] + \frac{\lambda_{\text{sk}}}{2} \Big[ \det(Jf_{||}) - 1 \Big]^2 \right) \tag{38}$$

and the whole superficial energy is approximated by

$$J_h^2 = \sum_{T_k \in \Gamma_F} W_{T_k}. \tag{39}$$

### 3.2.4 *Minimisation algorithm*

For a given vector $X_h$ and the boundary conditions, one deduces vector $\Phi_h = \mathcal{B}(X_h)$, hence the continuous linear piece-wise function $f_h$ used to compute the discretised energy functional. In order to determine the minimiser $\bar{X}_h$ of the discrete functional $J_h(X_h)$, the conjugate gradients method was employed. To this end, one has to compute an approximation

of the derivative of the discrete functional in directions $\ell = 1, \ldots, \#X_h$. Since the function is nonlinear and has complex structure, numerical derivatives are computed in a very simple way. For example, an approximation of derivative with respect to direction $\ell$ at point $X$ is given by

$$\partial_\ell J_h(X) \approx \frac{J_h(X + \epsilon_\ell) - J_h(X)}{\epsilon}, \tag{40}$$

where $\epsilon_\ell$ is a vector of zeros except value $\epsilon$ for the $\ell-$th entry.

## 3.3 BREAST UNDER GRAVITY

After the suturing, the two faces $\Gamma_L$ and $\Gamma_R$ are in contact and the question arises of the breast shape in the gravitational field in function of the trunk position.

### 3.3.1 *Geometry*

The sutured breast, now denoted by $\Omega_{grav}$ is modelled with the previous geometry, *i.e.* so the incised sections ($\Gamma_L$ and $\Gamma_R$) are still considered as being distinct parts of the boundary with a non-matching mesh. Because the gravity problem is different from the suturing problem, new boundaries are considered as seen in figure 11.

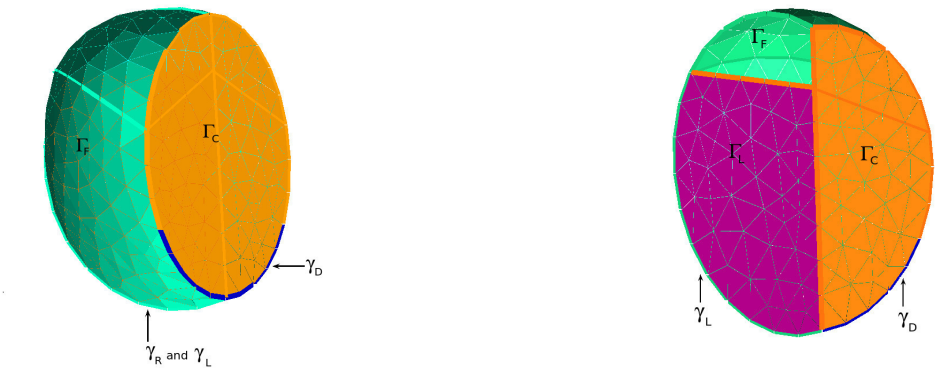



Figure 11: 3D rear views for the gravity problem: full breast (left) and half breast (right).

The breast geometry domain $\Omega_{\text{grav}}$ deriving from the suturing process still maintains the skin surface $\Gamma_F$, the frontal suture sections ($\Gamma_L$ and $\Gamma_R$) as well as the frontal suture lines ($\gamma_L$ and $\gamma_R$). The breast rear interfaces $\Gamma_B$, $\Gamma_{BL}$ and $\Gamma_{BR}$ were merged into a unique interface $\Gamma_C$ where the Chassaignac condition is prescribed. Moreover, a new curve $\gamma_D$ on the infra-mammary fold is defined where the breast elements are fixed.

### 3.3.2 *Mechanics*

Besides the evaluation of the strain energy from the skin and the breast fat and glandular tissue, two new contributions are considered for the energy functional: the gravity force and the Chassaignac space. The latter corresponds to a mobile zone located between the breast and the trunk which acts as a spring to maintain the breast close to the trunk ($\Gamma_C$). The gravity force will deform the mesh according to the alignment of the breast, *i.e.* depending on the position of the patient. Because the geometry of $\Gamma_{grav}$ results from the suturing process without physical alterations, the two interfaces $\Gamma_{\mathrm{L}}$ and $\Gamma_{\mathrm{R}}$, despite being in contact, are distinct. Therefore, to determine the new shape, one assumes that the surfaces are stick together in such a way that the relative position of the nodes of $\Gamma_{\mathrm{R}}$ (resp. $\gamma_{\mathrm{R}}$) with respect to the nodes of $\Gamma_{\mathrm{L}}$ (resp. $\gamma_{\mathrm{L}}$) will be preserved when the gravity field is applied.

The problem turns to the minimisation of the associated energy functional where one determines the new position vector field $f : p \to f(p)$ which minimises

$$\mathcal{J}_{\mathrm{gra}}(f) = \int_{\Omega_{\mathrm{gra}}} W_{\mathrm{br}}(\nabla f(p))\mathrm{d}p + \int_{\Gamma_{\mathrm{F}}} W_{\mathrm{sk}}(\nabla_{||} f(p))\mathrm{d}S_p - \int_{\Omega_{\mathrm{gra}}} \rho\, a_{\mathrm{g}} \cdot f(p)\mathrm{d}p + \int_{\Gamma_{\mathrm{C}}} c\|f(p) - p\|\mathrm{d}S_p \tag{41}$$

where $a_g$ is the gravity acceleration, $c$ the Chassaignac coefficient, and $\rho$ the density. So the new energy functional evaluates the contributions of the skin and the fat plus glandular tissue just like in the suturing procedure (equation 28) but now adds the contribution of the potential energy due to the application of gravity over every element of the breast that will lead to displacements of the mesh elements and the contribution of the Chassaignac space for the mobility of the nodes along $\Gamma_C$.

### 3.3.3 *Boundary Conditions*

The boundary conditions of the breast under gravity are used to ensure that the additional contributions to the energy functional are properly conditioned. This means that the left and right interfaces ($\Gamma_L$ and $\Gamma_R$) along with the suturing lines ($\gamma_L$ and $\gamma_R$) need a condition to keep them together, because when the force of gravity is applied, if a sticking condition is not prescribed, the two sections will be displaced independently and the suturing process is undone. So, the energy functional is restricted by the following boundary conditions

$$f(p) = p, \qquad \text{in } \gamma_{\mathrm{D}}, \tag{42}$$

$$b(f(p)) = 0, \qquad \text{in } \Gamma_{\mathrm{C}}, \tag{43}$$

$$f(p) - f(G(p)) = 0, \qquad \text{in } \Gamma_{\mathrm{R}} \cup \gamma_{\mathrm{R}}, \tag{44}$$

where $G$ is a functions which results from the suturing process which defined the mapping from $\Gamma_L$ (resp. $\gamma_L$) onto $\Gamma_R$ (resp. $\gamma_R$).

Condition (42) fix the position of the elements along curve $\gamma_D$ which correspond to the infra-mammary fold. Condition (43) maintains the breast on the trunk despite the mobility evaluation with the Chassaignac space contribution to the energy functional. Condition (44) is the stick condition that preserves the junction between the left and right interfaces previously joined in the suturing process.

## 3.4 DISCRETISATION OF THE BREAST UNDER GRAVITY

A discrete version of problem 41-44 is introduced based on a mesh of tetrahedrons of the sutured breast and a continuous piecewise representation of the displacement $f$. Due to the new boundary and constraint conditions, one has to deal with a new subset of unknowns $Y_h$ from $\Phi_h$ and a new operator $\mathcal{C}$ such that $Y_h \to \Phi_h = \mathcal{C}(Y_h)$ models all the constraints 42-44.

### 3.4.1 *Boundary, Chassaignac and sticking conditions*

Condition 42 yields that $f_n = P_n$ for any $P_n \in \gamma_D$ while relation 43 implies that for any $P_n \in \Gamma_C$, one sets $f_{ny} = 0$ to maintain interface $\Gamma_C$ on the trunk plane $y = 0$. Constraint 44 means that the two interfaces previously joined and stick will preserve their relative position and function $G$ is defined in the following way. At the initial stage, after reading the mesh, the barycentric coordinates are calculated for points $P_n$ on $\gamma_R$ and $\Gamma_R$ corresponding to the suturing process:

- If $P_n \in \gamma_R$, one determines the edge $e = [P_\alpha, P_\beta] \subset \gamma_L$ such that $P_n$ also belongs to $e$ since the two curves are joined. Then the unique barycentric coordinates are defined such that $G(P_n) = \lambda_{n\alpha} P_\alpha + \lambda_{n\beta} P_\beta = P_n$. Hence the constraint on the new location $f_n$ associated to point $P_n$ writes $f_n = \lambda_{n\alpha} f_\alpha + \lambda_{n\beta} f_\beta$ where $f_\alpha$ and $f_\beta$ are the new positions of $P_\alpha$ and $P_\beta$, respectively. $f_n$ does not belong to the entries of $Y_h$ since it is deduced from the linear combination of $P_\alpha$, $P_\beta$.

- If $P_n \in \Gamma_R$, one determines the triangle $f = [P_i P_j P_k] \subset \Gamma_L$ such that $P_n$ also belongs to $f$ since the two surfaces are sutured. Then the unique barycentric coordinates are defined such that $G(P_n) = \lambda_{ni} P_i + \lambda_{nj} P_j + \lambda_{nk} P_k = P_n$. Hence the constraint on the new location $f_n$ associated to point $P_n$ writes $f_n = \lambda_{ni} f_i + \lambda_{nj} f_j + \lambda_{nk} f_k$ where $f_i$, $f_j$, and $f_k$ are the new positions of $P_i$, $P_j$ and $P_k$, respectively. $f_n$ does not belong to the entries of $Y_h$ since it is deduced from the linear combination of $P_i$, $P_j$, and $P_k$.

### 3.4.2 *Discretisation of the functional*

A discrete version of the energy functional setting is introduced as

$$J_h(Y_h) = \widetilde{J}_h(\mathcal{C}(Y_h)) = \widetilde{J}_h(\Phi_h) = J_h^1 + J_h^2 + J_h^3 + J_h^4 \tag{45}$$

with $J_h^1$ being the volume strain energy and $J_h^2$ being the surface strain energy. $J_h^3$ represents the potential energy of gravity force exerted over the breast mass (whose density $\rho$ is constant) and $J_h^4$ represents the energy obtained with the spring-mass model of the Chassaignac space like as the one presented by Cardoso et al. (2013a) where $c$ is a constant. $J_h^1$ and $J_h^2$ are given in section 3.2.3. $J_h^3$ and $J_h^4$ are given by

$$J_h^3 = -\sum_{\tau_i} Wg_{\tau_i} \tag{46}$$

and

$$J_h^4 = \sum_{T_j \subset \Gamma_B} Wc_{T_j} \tag{47}$$

where $Wg_{\tau_i}$ and $Wc_{T_j}$ are given by

$$Wg_{\tau_i} = \frac{|\tau_i|}{4}\rho a_g \left(f(P_{i1}) + f(P_{i2}) + f(P_{i3}) + f(P_{i4})\right) \tag{48}$$

and

$$Wc_{T_j} = \frac{|T_j|}{3}c \left(\|f(P_{i1}) - P_{i1}\| + \|f(P_{i2}) - P_{i2}\| + \|f(P_{i3}-)P_{i3}\|\right). \tag{49}$$

The minimisation of functional $J_h$ is performed in the similar way as presented in section 3.2.4.

## 3.5 NUMERICAL TESTS

To evaluate the performance of the numerical model in terms of breast shape and size, some numerical tests were done. Tests concerning real-life cases are shown in chapter 8 where the results were validated by surgeons with the exception of just one example to give practical context. Apart from that example, this section only considers numerical results.

As mentioned previously, the original breast is modelled as a truncated sphere characterised by its principal radius and the length of the truncated portion. Figure 12 displays the initial configuration of the breast, *i.e.* before the incisions. It is worth noticing that more complex geometries can easily be considered and the suturing method is not limited to that particular configuration.

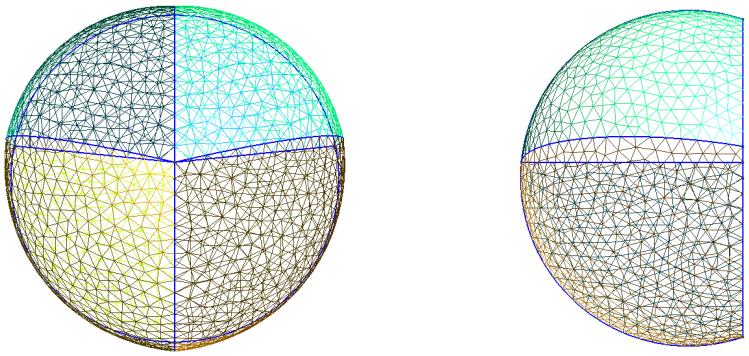


Figure 12: Mesh of the pre-operative breast: frontal view (left) and lateral view (right).

To this extent, the tests aimed to evaluate in visual terms how the model behaves for breast surgeries with frontal symmetric incisions, frontal asymmetric incisions as well as breast reduction surgeries with frontal and rear incisions. Every test started by generating the geometry of the breast with cuts, followed by the suturing process and ending with the simulation of the sutured breast under gravity. The figures will show the breast mesh configuration in two views: frontal and lateral. This way it is possible to observe the differences between the various tests more clearly. The breast examples showed in this section have a value of radius $R = 10cm$ and $H = 5cm$.

### 3.5.1 *Symmetric suturing*

In the first numerical tests, symmetric frontal incisions were done with various values for $\alpha_L = \alpha_R$. Figure 13 shows an example of a breast with frontal cuts performed with $\alpha_L = \alpha_R = 30^\circ$ and without rear cuts. The value of $FL$ chosen to decide where the incision starts (point $A$) was of $130mm$.

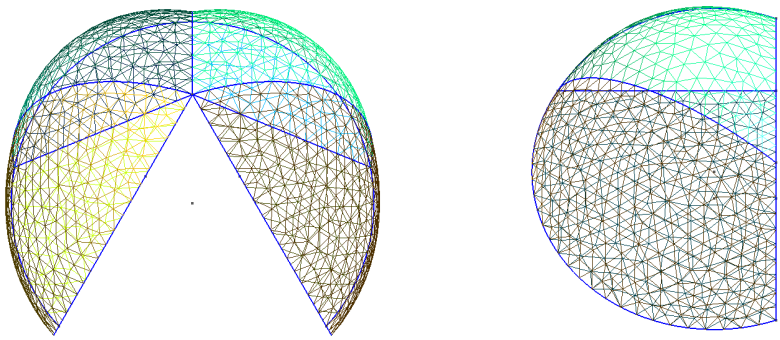


Figure 13: Mesh of the breast after a symmetric incision: frontal view (left) and lateral view (right).

This mesh is used during the breast suturing simulation and the result can be seen in figure 14 with the junction plane $\Pi_\theta$ being obtained with $\theta = 0^\circ$ in order to preserve the symmetry of the junction.

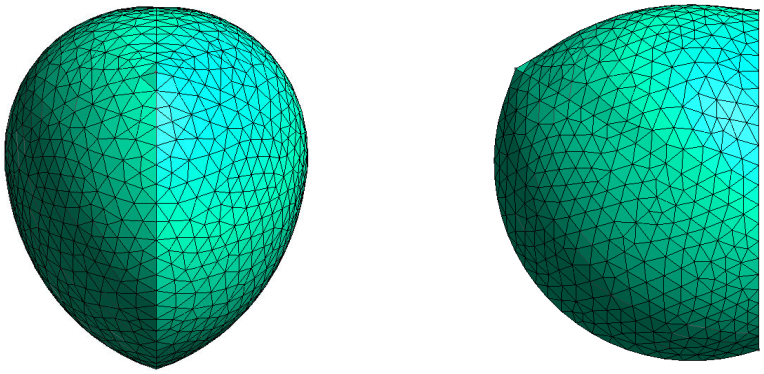


Figure 14: Shape of the breast after the suturing without gravity field: frontal view (left) and lateral view (right).

The symmetry of the shape is preserved and the convergence is checked to a limit configuration using thinner meshes. One then evaluates the breast shape, after being sutured, subject to the gravity field for a patient in a stand up position. Figure 15 displays the frontal and side view of the breast under the gravity field.

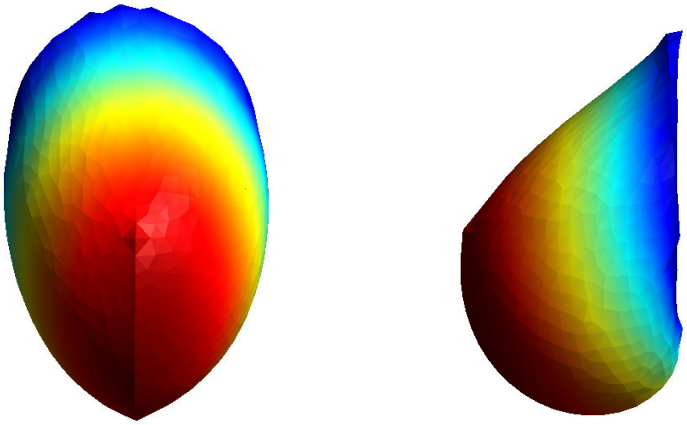


Figure 15: The breast under the gravity field: frontal view (left) and lateral view (right).

Notice that the symmetry is also preserved and the junction obtained is a straight line. Computations have been carried out with thinner meshes to check that the solution converges to a limit one. The presented solution involves a mesh of 1246 tetrahedrons and is very closed to the one obtain with thinner meshes (less that 1% of relative errors).

### 3.5.2 *Asymmetric suturing*

In some situations, the surgeon needs to perform asymmetric incisions. Therefore additional experiments were conducted to assess the code architecture capacity to handle a case with such higher complexity. Figure 16 presents an example of the breast with cuts before the suturing process.

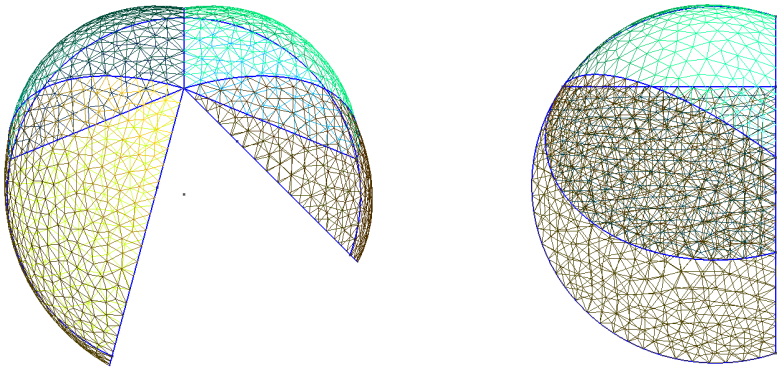


Figure 16: Mesh of the breast after an asymmetric incision: frontal view (left) and lateral view (right)..

For the frontal incision angles, $\alpha_L = 15°$ and $\alpha_R = 45°$ were chosen as well as $FL = 130mm$ for the initial incision point. Clearly the breast volume of the right side is strongly reduced in comparison with the left side. The junction plane is characterised by taking the angle $\theta_0 = (-15 + 45)/2$ of the suturing plan as the (algebraic) mean values of the two frontal incision angles and the result of the suturing can be seen in figure 17.

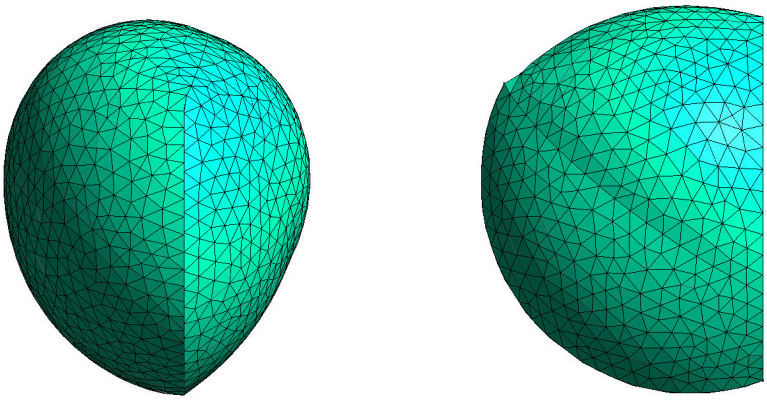


Figure 17: Mesh of the breast after suturing: frontal view (left) and lateral view (right).

Computations were carried out with a 1376 tetrahedrons mesh and the shape of the breast after the suturing process is clearly asymmetric. As in the former case, one obtains shape convergence using thinner meshes which guarantee the stability of the method.

As in the previous example, the behaviour of the sutured breast under the gravity field was also tested. Figure 18 shows the final configuration of the breast after all iterations of the simulation of the breast behaviour under gravity assuming a patient standing up.

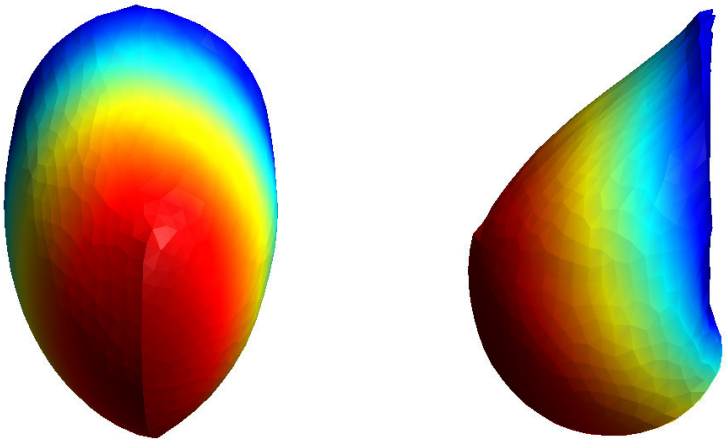


Figure 18: The breast under the gravity field: frontal view (left) and lateral view (right).

Notice that the suture location is no longer a straight line as displayed in figure 17. The breast nodes re-arranged in a form that seems more natural even though it shows the area where the incised sections were sutured. Convergence to a limit configuration has been also checked using thinner meshes.

### 3.5.3 *Frontal and rear suturing*

The last set of tests were done considering frontal and rear cuts. Figure 19 shows an example of a test where the mesh corresponding to two frontal symmetric incisions with $\alpha_L = \alpha_R = 30^\circ$ and the rear cuts being defined by a value for $d = 10mm$, $FL = 130mm$ and the lateral cuts being given by $LL = 115mm$.

Figure 20 shows the mesh configuration after the numerical suturing process where the chosen suturing plane $\Pi_\theta$ was obtained with $\theta = 0^\circ$.

A 1539 tetrahedrons mesh was used and the results show a symmetric breast preserved by the choice of suturing plane. Several finer meshes were also experimented to check the convergence. No significance differences (lower than 1%) have been reported with respect to the solution. Finally the sutured breast behaviour was simulated considering that the patient was standing up. It is displayed in figure 21 the resulting shape and observe a symmetric final breast with a nipple position being in a central position of the breast.

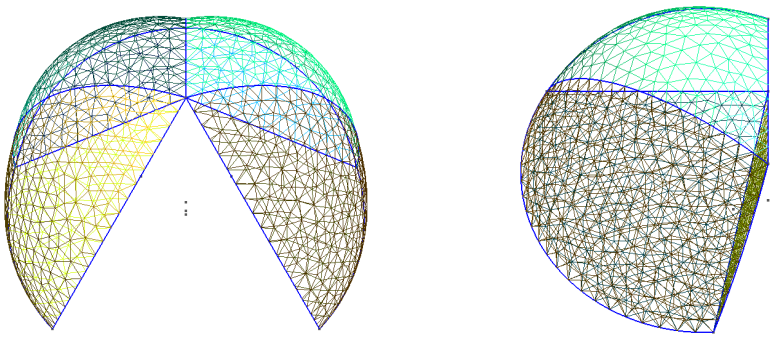


Figure 19: Mesh of the breast after a symmetric front incision and a rear incision: frontal view (left) and lateral view (right).

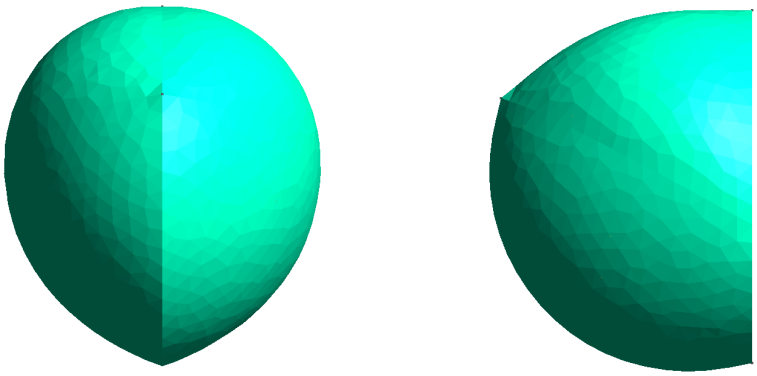


Figure 20: Breast shape after the suturing process: frontal view (left) and lateral view (right).

### 3.5.4 *Comparison with a real surgery case*

To validate the code and assess the capacity of the numerical method to provide realistic results, a comparison between a real surgery and the simulation has been carried out. As a first stage, a real surgery took place, and pictures of the breast were taken.

Then, a numerical simulation was performed using a set of parameters inferred from the original breast. Figure 22 displays a comparison before and after the surgery between the real breast and the corresponding simulation. Qualitative results are very good since one obtains the same shape and a good approximation of the volume. There is also a nice matching of the nipple location which show that the simulation is in accordance with the reality.

## 3.6 DISCUSSION

The breast model developed in the scope of this project enable to accurately predict the behaviour of the breast, given the surgical parameters (in this particular case breast reduc-

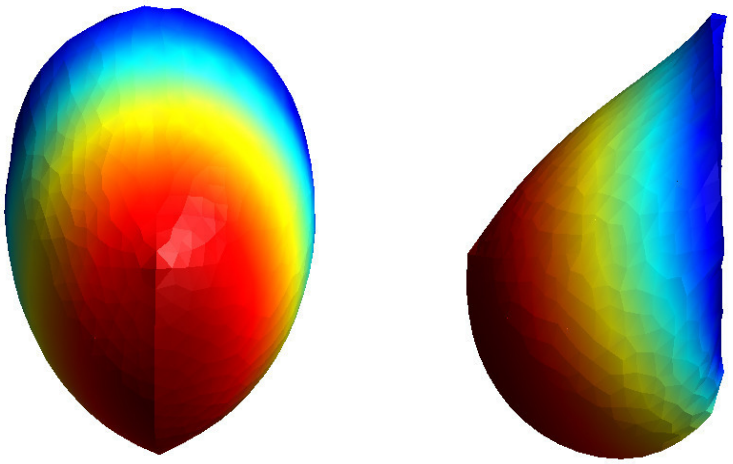


Figure 21: The suturing breast in the gravity field: frontal view (left) and lateral view (right).

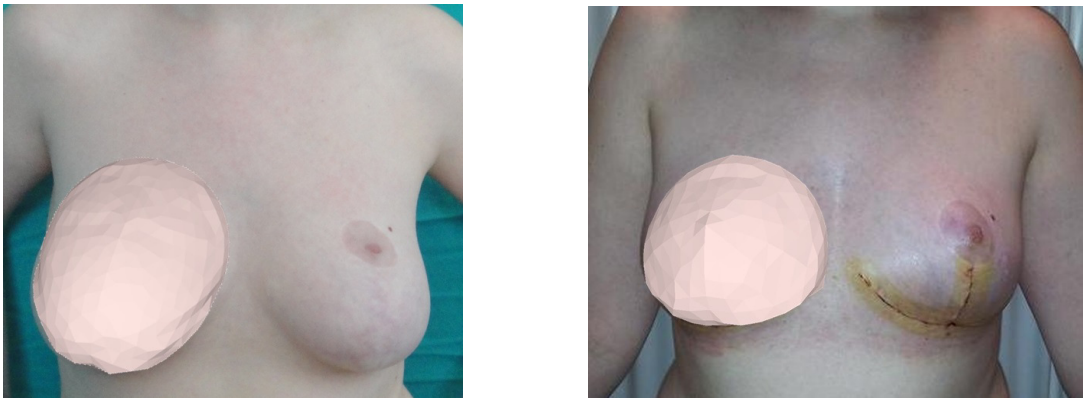


Figure 22: Comparison between a real surgery operation and the simulation. Before the surgery (left) one reproduces the initial breast with an identical volume. After the surgery (right), one carries out the simulation with the cuts made by the surgeon. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

tion surgery parameters). The simplicity of the model and the assumptions made, namely the approximate spherical shape of the breast in a state of free stress and the new knitting conditions provide a good approach to the development of a simple breast model that reproduces the behaviour of the breast tissues.

The overall performance of the simulation method is good although in the first versions of the code it could took a few minutes to process data for meshes with just a 1000 nodes. Improvements were made afterwards to improving the simulation performance detailed in chapter 5. The real life case presented in this chapter shows the ability of the model to guide surgeons for planing surgeries. It also provides an helpful medical assistant for students to gain some experience and develop a certain sensitivity when studying surgical cases. Accuracy depends on a correct estimation of the breast's biomechanical parameters presented in chapter 4.

Despite the good results, there is still works to make the model more robust, namely additional tests to consolidate the work already done in this project. There were some considerations made regarding the breast model which concerns its evolution in a posterior phase which involve the study of other breast structures like Cooper's ligaments or think on exploring other primitives like splines or simply ellipsoidal shapes.

# 4

# PARAMETERS IDENTIFICATION

The parameters associated to any breast material are key ingredients in the constitutive model's performance. The study by Eder et al. (2014) presents a clear example that the biomechanical parameters of breast tissues play a major role in describing the breast behaviour. In this chapter a new method to estimate biomechanical parameters of a breast model is presented along with its proof of concept using the constitutive model described in chapter 3.

## 4.1 PARAMETER IDENTIFICATION

The correct identification and estimation of the geometric and mechanics parameters is a crucial issue for achieving representative purposes. From *ex vivo* tissue samples to elastography there exist a large spectrum of techniques that are used to estimate the elastic properties of the female breast. The identification of soft tissue parameters quite depends on the underlying model (Delingette and Ayache (2004)) and the material characterisation, namely, the linear elastic model, the Neo-Hookean model and even the Mooney-Rivlin model (Eder et al. (2014)).

### 4.1.1 *Manual Methods*

Palpation is a very simple and basic procedure used by doctors and encouraged in people by self-examination (see figure 23) to detect possible lumps or other tumours out of the ordinary on the breast (Wells and Liang (2011); Gennisson et al. (2013)). Palpation does not offer, by itself, any kind of reliable information about the elasticity of the breast internal tissues (Wells and Liang (2011)). It can, however, provide hints and comparisons that together with experience can lead to fair estimations. For example, when holding two breasts one can determine if the difference in weight is significant. Two breasts with approximately the

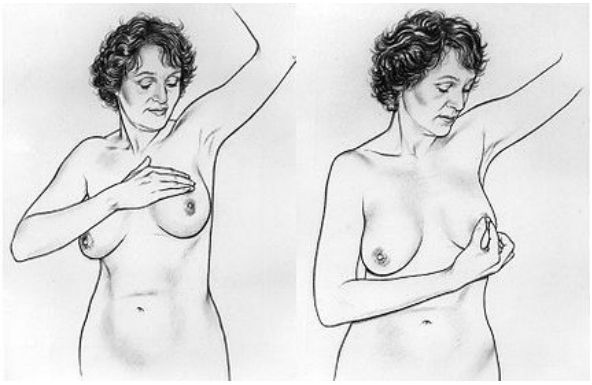


Figure 23: Breast self-examination (Wikipedia)

same size and shape may have a substantial weight difference leading to the definition of density ($\rho$)

$$\rho = \frac{m}{V} \tag{50}$$

with the body's mass ($m$) and the associated volume ($V$). Commonly, the heavier breast has more fat and glandular tissue is more compressed by the skin. However, the breast is constituted of a lot of different internal tissues and the volume information is not enough to characterise the breast (Gefen and Dilmoney (2007)). With experience, the doctor feels the difference for well distinct cases without using additional equipment. However, the method is almost qualitative, quite limited and depends on the doctor's skill. As highlighted in Wells and Liang (2011), the interpretation of palpation is completely subjective, *i.e.* it does not provide any quantitative data and manual estimation of the breast elasticity parameters using palpation is not eligible any longer.

### 4.1.2 *Medical Imaging*

As mentioned in chapter 2, medical imaging technology like MRI, CT, ultrasounds and x-rays allowed doctors and medical practitioners to collect information about the internal tissues of the human body (Gennisson et al. (2013)). More importantly, a new imaging technique known as elastography leads to the rise of various imaging processing used to measure and map tissue's stiffness. A potential tumour detected with palpation could be confirmed and localized (Wells and Liang (2011)). In terms of soft tissue constitutive models, the technique determines the elastic properties of the tissues (Wells and Liang (2011); Gennisson et al. (2013); Sigrist et al. (2017)) prescribed in the biomechanical model (Azar et al. (2002); del Palomar et al. (2008)). linear model or hyperelastic relations such as Neo-Hookean and Mooney-Rivlin (Eder et al. (2014)) are the most common model used in a health context.

Original methods use *ex vivo* samples of breast tissues to analyse the deformation under compression using medical imaging techniques (Krouskop et al. (1998); Wellman et al. (1999); Azar et al. (2001)). More recent techniques apply non-invasive procedures by displacement differentials in the breast when the patients were standing up, in a supine or prone position (del Palomar et al. (2008); Eiben et al. (2016)).

The *ex vivo* sample analysis provides average values for the elastic modulus of the breast tissues, especially for fat, glandular tissue, skin and ligaments (Krouskop et al. (1998); Wellman et al. (1999)). As mentioned by Gefen and Dilmoney (2007), these values vary with the studies, for example, fat tissue can have values for the elastic modulus ranging from 0.5 to 25 kPa while the skin values vary from 200 to 3000 kPa. The coefficients were evaluated for different rates of stress and strain in order to provide a specific stress-strain curve for the breast models. In Yin et al. (2004); Stewart et al. (2011); M.Kuhlmann et al. (2013), the authors suggest that averaging the different tissue component, regard to the elastic parameters, might not deliver good enough approximations for complex cases as they highlighted in Eder et al. (2014). Moreover, the deformation of the tissues when being compressed (in MRI or CT scan devices) introduces error in the measurements of the elastic properties (Rajagopal et al. (2008)).

Other breast models proposed by del Palomar et al. (2008); Han et al. (2012a); Dufaye et al. (2013) use the coefficient values from literature as a starting point to generate an initial mesh and they adjust the parameters for a specific breast by an iterative procedure until the difference between the digital model and the real-life model provides a small enough error. They obtain images from specialised equipment like MRI machines to compare with the refined breast.

In a similar approach, the technique proposed by Rajagopal et al. (2008); Cardoso et al. (2013a); Eiben et al. (2016) enables to evaluate the tissues parameters by considering the breast in a free stress state from which they analyse the breast's displacement deriving from the gravity force. Those techniques partially overcome the potential inaccuracy of inadequate values issuing from the literature. However, they are time consuming and their efficiency depends on some additional factors such that the initial set of parameters considered for the iterative procedure (del Palomar et al. (2008)). This latter aspect is also referenced in Eiben et al. (2016) which use an extreme set of initial values (meaning that the breast is either very soft or stiff) as a starting point for optimising the solution. They mention that this approach does not give optimal parameters but accurate enough provide rough approximations (Eiben et al. (2016)).

The model proposed by Dufaye et al. (2013) picks up again the previous approaches for parameter estimation. They use a predetermined set of patients (in the paper 10 cases are considered) to produce MRI images and then solve the inverse problem where the parameters are refined by comparing with keys elements of the breast, namely the nipple position.

After obtaining the parameters, they then consider the average values for predicting the parameters for every other future case (Dufaye et al. (2013)). They compute the estimates of the elastic parameters by using significant samples, *i.e.* patients with breasts that can cover most of the female universe. Although the study only considers patients with C cup or higher breasts volume, the method can be easily extended to every patient. Nevertheless, average values of properties are not adequate for every kind of breast (Eder et al. (2014)).

### 4.1.3 *Cameras and Scanners*

Medical imaging techniques and technology, mainly MRI, CT and ultrasounds provide the raw data they used to estimate the breast parameters by comparing the respective positions of specific elements of the breast like the nipple (Dufaye et al. (2013)) or boundaries (Cardoso et al. (2013a)) or even zones assigned with markers (del Palomar et al. (2008)). To accurately determine the internal tissues location, medical imaging equipment like MRI is necessary since they reveal the entire volume of the breast (more specifically multiple slices that transverse the breast). That is particularly important for tumorous/cancerous cells identification and localisation (Azar et al. (2002)). However, in mechanical models as in del Palomar et al. (2008), the difference between the fat and glandular tissue is neglected and information deriving from visual evidence is enough to estimate the breast parameters and images or films to determine the external aspects of objects was widely studied over the years (Bregler et al. (2000); Kimura and Saito (2000); Viola and Jones (2001); Lowe (2004); Salzmann and Urtasun (2012); Eder et al. (2012); Zollhöfer et al. (2014); Henseler et al. (2016); Pöhlmann et al. (2017); Guo et al. (2017)).

More recently, new algorithms were proposed to capture and reconstruct the surface of a three-dimensional object (Oliveira et al. (2013)). One uses regular cameras to seek features like corners, edges and zones that are detectable by the images. These features have characteristics that make them observable and easily detectable. According to Oliveira et al. (2013), such features take into account the following:

- Reliability - features that can be seen and recognised in different perspectives of the object
- Distinctiveness - features should be different/unique in order to ease the process of recognition
- Locality - the features should be placed in distinctive areas of the object
- Quantity - The more distinctive features, the easier is to recognise the object

In the case of the breast, features pointed in literature are the nipple, the infra-mammary fold, the clavicular notch, the armpit and the contrast made between the breast and the

torso (del Palomar et al. (2008); Dufaye et al. (2013); Oliveira et al. (2013); Henseler et al. (2016)). These features fit the criteria mentioned above and so they are used to create a representative mesh. The identification of features and object surface is performed with regular cameras or cameras and scanners with the capacity of evaluating depth (Oliveira et al. (2013)).

Regular cameras do not directly provide information about the three-dimensional of the breast that is required to reconstruct a mesh. Most algorithms use multiple cameras as in reference Kimura and Saito (2000) to reproduce the eyes perception of the three dimensions. Two cameras using epipolar geometry (figure 24) are used to simulate the geometry of stereo vision (Kimura and Saito (2000)).

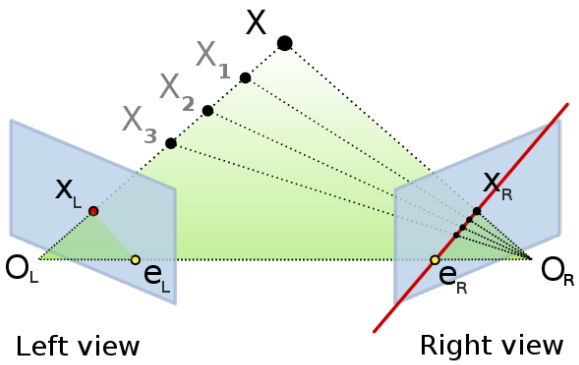



Figure 24: Epipolar Geometry (by Arne Nordmann - wikipedia)

Henseler et al. (2011) presented a method using a capturing system using 8 cameras which are aimed at both breasts of the patient. The problem of a method like this is the potential of errors that can occur from the location of the cameras, calibration and even lighting (Lowe (2004); Henseler et al. (2011)). Those errors could also arise from the quality of the algorithm used (e.g. Scale Invariant Feature Transform (SIFT)) and the tweaks made before capturing (Lowe (2004))

Recently, low-cost cameras and scanners have been massively produced and enable to measure depth (Eder et al. (2012); Zollhöfer et al. (2014); Henseler et al. (2016); Pöhlmann et al. (2017); Guo et al. (2017)). For instance, the Kinect camera developed by Microsoft turns to become famous in the video game industry (Izadi et al. (2011)) but also reveals to be a precious tool for collecting 3D picture. Similarly, the Konica-Minolta V910 scanner used by Eder et al. (2012) also provide 3D views for breast reconstruction. The usage of the Kinect camera was widely studied mainly due to its low cost compared to other cameras and solutions (Oliveira et al. (2013)). Independently of the depth camera used, the complexity of the problem is reduced with the additional depth information. KinectFusion is a software solution developed by Microsoft that displays real-time 3D reconstruction of objects by moving the Kinect camera (Izadi et al. (2011)). One also mentions Zollhöfer et al.

(2014) who manages to compute the depth map that generate a point cloud on the object's surface. The points are then tracked even when the camera moves (Izadi et al. (2011); Zollhöfer et al. (2014); Henseler et al. (2014, 2016); Pöhlmann et al. (2017)).

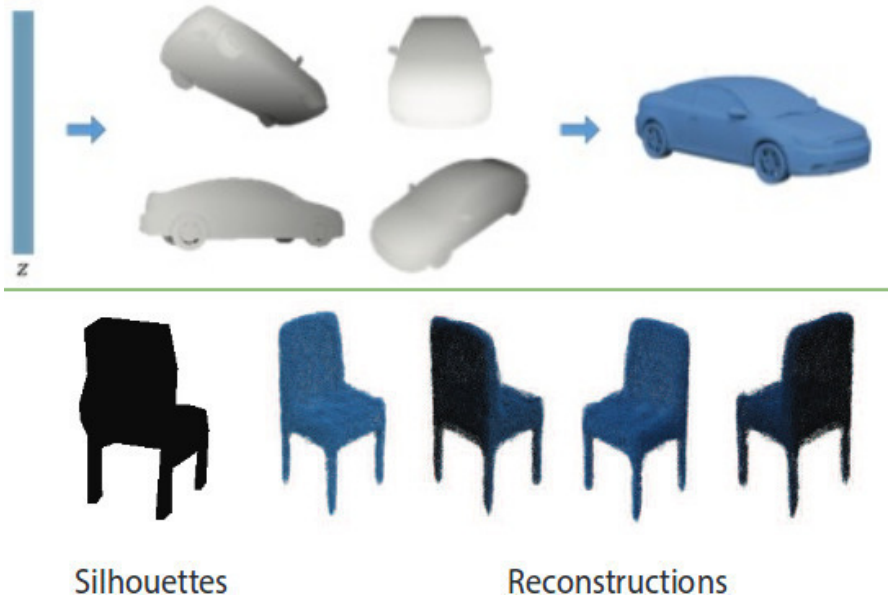



Figure 25: 3D object reconstruction from depth maps and point clouds (by Arsi Warrior - wikipedia)

From this point cloud on the objects, a surface is then possible to reconstruct the object (figure 25) by connecting these points using algorithms that integrate 3D vertices (points) into voxels like signed distance functions which allow determining how far are some points from a certain objects surface (Izadi et al. (2011)). The usage of the Kinect in breast reconstruction was explored in Pöhlmann et al. (2017) and the results show that these type of cameras designed for speed have the potential of estimating the breast surface and volume. There are however issues concerning these types of cameras, namely the error that can come from depth noise, the resolution of the camera (which is better with the new versions of the Kinect) and in terms of capturing people the error that can come from their natural movement (Henseler et al. (2014); Pöhlmann et al. (2017)). Besides, with the exception of the Kinect, other 3D scanners are expensive and for that matter may not be accessible to everyone (Pöhlmann et al. (2017)).

### 4.1.4 *Alternative Method*

The above methods (palpation, medical imaging, 3D reconstruction using cameras) have their pros and cons. Palpation is easily done by a doctor and quite practical but does not give quantifiable information about the breast properties. Regarding medical imaging technologies and techniques, the pros and cons are:

- Pros
  - Provide information about the internal organisation of the tissues;
  - Enable the analysis of the tissues elastic properties with great accuracy (Krouskop et al. (1998); Azar et al. (2002)).
- Cons
  - The equipment is very expensive and is limited (Carriço (2017); Pöhlmann et al. (2017));
  - The patient spends a great deal of time during the exams (MRI average can go from 20 to 60 minutes Edelstein et al. (2010));
  - The techniques used are very complex and time consuming (Azar et al. (2001); Eder et al. (2014); Eiben et al. (2016)).

The list of pros and cons for 3D reconstruction using cameras are:

- Pros
  - Cheap cameras like the Kinect can be used to generate the surface of the object with a certain amount of accuracy (Izadi et al. (2011); Pöhlmann et al. (2017));
  - Enables the analysis of the breast surface with a certain accuracy (Henseler et al. (2014));
  - The techniques used are easier to implement when compared to the medical imaging (Oliveira et al. (2013); Pöhlmann et al. (2017)).
- Cons
  - Do not provide information about the internal organisation of the tissues;
  - Quality of the results depend on the quality of the camera, lighting and the behaviour of the patient when capturing images or scanning (Henseler et al. (2014); Pöhlmann et al. (2017));
  - High definition cameras which can lead to less errors are expensive (Pöhlmann et al. (2017)).

Unfortunately, doctors may not have access to any of those resources, either by the MRI machine being scheduled for other patients or by not having cameras or not having calibrated cameras. Moreover, the cost and time required to achieve the measures is also an obstacle. It would be desirable that the pre-surgical planning data could be used to estimate a breast's elastic parameters.

Without data from any medical imaging equipment, it is not possible to create a breast model with an adequate discretisation of tumours and other types of internal lesions. However these different approaches have something in common: for breast models where the

internal arrangement of the tissues can be neglected, they can use features of the breast surface with the patient in different positions and analyse the deformations in order to obtain the breast parameters. So, knowing the way a breast reduction surgery is planned, it might be possible to use the marks made on the breast (like the ones in figure 6) as measurements that one compares with the patient in different positions. The approach to estimate parameters presented in Cardoso et al. (2013a) reflects such a perspective but they have only estimated the mechanical parameters of the fat plus glandular tissue.

Another aspect to take into consideration is that, without medical imaging techniques and the usage of cameras, the only way to generate a mesh passes by the usage of splines and other primitives which use parameters to be generated and fit the breast. So, in the end, the method to estimate the breast mechanical parameters could possible also estimate geometrical parameters. This approach has the benefits of being flexible and easy to implement because the method can not only be used by a surgeon using only pre-surgical planning marks but also use the other technologies and techniques mentioned previously (MRI and RGB-D cameras). With this, the method to estimate the breast model parameters, which is presented in this chapter, will try to answer the question mentioned in the introduction "How can measurements be used to create accurate models for the breast?"

## 4.2 PARAMETER ESTIMATION

Determining the material parameters of soft tissues for simulation is the very critical point since it almost drives the simulations. To provide the reference configurations, optimisation procedures are usually carried out on the basis of MRI and other radiological acquisitions (Azar et al. (2002); del Palomar et al. (2008); Rajagopal et al. (2008); Eiben et al. (2016)). The methods proposed in literature are based on very complex reverse elasticity problems which have a high computational cost (Han et al. (2012a); Eiben et al. (2016)), that would be unnecessary or too expensive for routinely or low-cost practices. From a practical point of view, the objective of these methods is to recreate a digital model of the breast that presents a very similar behaviour to the real breast (del Palomar et al. (2008); Rajagopal et al. (2008)). However, as Eder et al. (2014) proved, the use of complex models with many details don't necessarily mean that the results will be good, hence a new and simpler method is proposed.

The proposed method wants to provide an effective way of retrieving the breast parameters to be, in turn, used to accurately predict breast deformations. Moreover, another aim of this model is to enable it to run on a mid-range laptop and produce results within a few minutes. The key ideas are:

- introduction of a simple set of *in vivo* breast measurements;

- usage of a four-parameter non-linear mechanical model defining on a simplified two-parameters reference configuration (breast in a free stress state).

The measurements provide the data to fit the model parameters (both mechanical and geometrical) that best approximate the breast size, shape and behaviour.

### 4.2.1 *Breast Modelling*

The breast model is presented in section 3.1. Deformation evaluation of the breast over external actions is achieved by considering a simplified stress-free geometrical domain of the breast equipped with the Neo-Hookean mechanical model where the breast core (fat and glandular tissue) and skin are differentiated and with the Chassaignac space which affects the breast mobility. The gravity-free reference breast domain $\Omega_g$ is a piece of spherical cap where the plane section is attached to the torso. The domain is parameterised with the radius $R$ of the sphere while $H < R$ represents the non-truncated length as displayed in Figure 26. The cap which is placed in the torso plane corresponds to a circle of radius $r$ which is smaller than $R$.

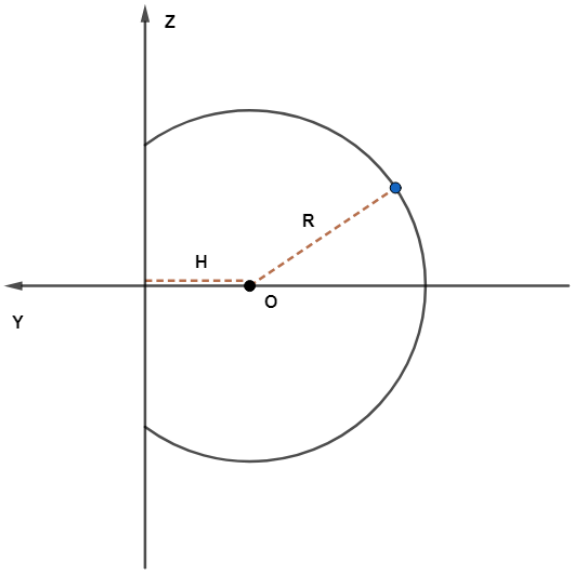



Figure 26: Breast geometry configurations. Radius $R$ characterised the main sphere while $H$ identifies the truncated part following the $y$ direction.

The difference between the present model and the one presented in chapter 3 is its goal. While the other model's focus relied on the surgical procedure, in here one analyses the breast deformation from a state where the breast is in a free stress state and under a gravity field (figure 27). To prescribe the boundary condition and compute the energy associated to the skin, one denotes $\Gamma_B$ as the back side plane of the breast which attaches to the torso, $\Gamma_F$

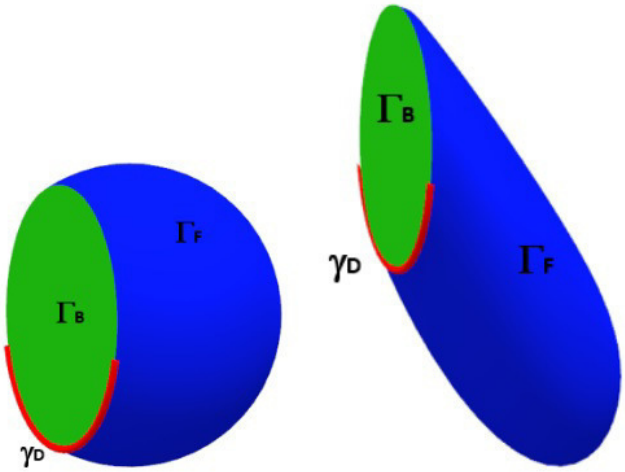



Figure 27: Notation and geometry of the gravity-free geometry (left) and configuration in the gravity field (right)

that represents the surface associated to the skin of the breast and $\gamma_D$ which is the arc of the infra-mammary fold. $\Gamma_B$ also corresponds to the Chassaignac space. The main difference is then reflected on the boundary conditions, because in this problem, one evaluates the breast parameters before the surgery, which means that there are no cuts performed in the breast and so the sticking conditions are not applicable.

Under gravity, the breast corresponds to a minimisation of the energy functional associated with the Neo-Hookean model. Such a functional aggregates the internal energy due to the bulk, the energy deriving from the skin displacement, the gravitational energy characterised by the gravitational field $a_g \in \mathbb{R}^3$ and the spring energy associated to Chassaignac space. For any generic point $p \in \Omega_g$, one seeks the new position vector field $f : p \in \Omega_g \to f(p) \in \mathbb{R}^3$ which minimises the energy functional given by

$$\mathcal{J}(f) = \int_{\Omega_g} W_{\text{br}}(\nabla f(p))\mathrm{d}p + \int_{\Gamma_{\text{F}}} W_{\text{sk}}(\nabla_{||} f(p))\mathrm{d}S_p - \int_{\Omega_g} \rho\, a_{\text{g}} \cdot f(p)\mathrm{d}p + \int_{\Gamma_{\text{B}}} c\|f(p) - p\|\mathrm{d}S_p \tag{51}$$

subject to the constraints

$$f(p) = p, \qquad \text{in } \gamma_{\text{D}}, \tag{52}$$

$$f(p).\vec{N}^B = 0, \qquad \text{in } \Gamma_{\text{B}} \tag{53}$$

where $c$ is the Chassaignac coefficient, and $\rho$ stands for the bulk density. Constraint 52 represents the fixation of the infra-mammary fold on the torso, *i.e.* any point $p$ in $\gamma_{\text{D}}$ maintain their position while constraint 53 states that any point $p$ on the breast plane attached to

the torso (Chassaignac space) only moves inside that trunk plane. The expressions for the volume strain-energy and the skin strain-energy densities, respectively represented by $W_{\text{br}}$ and $W_{\text{sk}}$, are given in equations 28 and 29.

### 4.2.2 *Measurements*

To estimate the geometrical and mechanical parameters of a breast, one checks that the measurements correlate with the parameters, *i.e.* a particular set of measures should relate to a specific set of measurements. In literature, the influence of the volume of the breast and the area of the skin surface are one of the most mentioned (Rajagopal et al. (2007); del Palomar et al. (2008); Henseler et al. (2016)). Others criteria as the position of the nipple (Dufaye et al. (2013)) and even the frontal and horizontal length of the breast (Cardoso et al. (2013a)) are underlined as potential measurements. Most of these measurements derive from medical imaging techniques or cameras but the intention of the method proposed is to enable gathering measurements during consultations. In Liu (2009), an optical method capable of accurately measuring the breast volume and surface area was presented. The work that leads to the method development proves that it is possible to estimate breast characteristics that some considered difficult without using special equipment (Liu (2009)). Anthropometric measurements techniques were presented in Kayar et al. (2011). When surgeons plan surgeries, for example, breast reductions, they perform some markings that allow them to plan the course of action (figure 28).

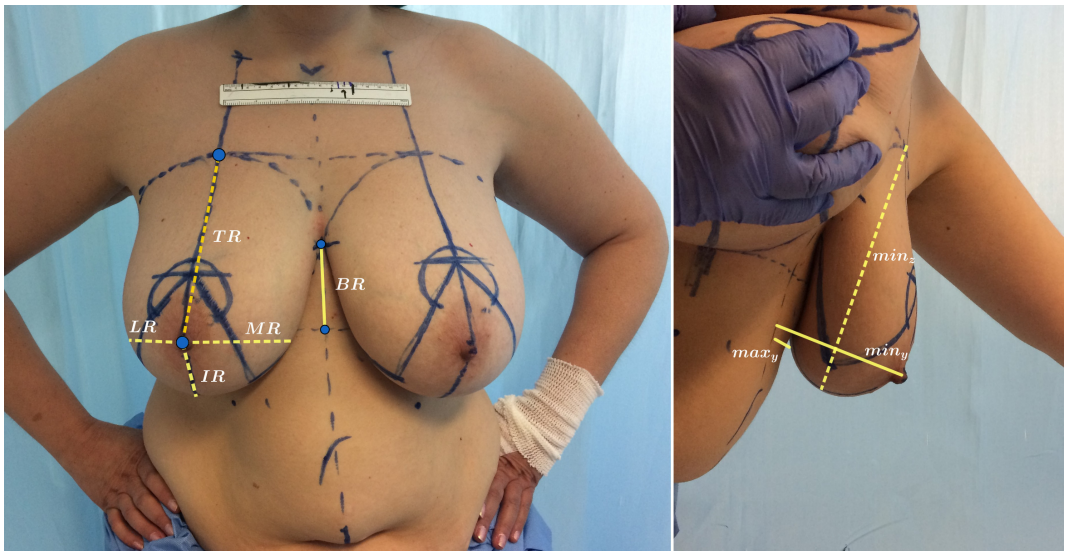



Figure 28: Markings on a patient in two positions: standing up on the left; bend over approximately 45° on the right. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

The surgeon marks the breast with lines that represent the incisions, the new position of the nipple as well as the limits of the breast. Based on the markings and additional information provided by the literature, it was decided to determine the mechanical and geometrical parameters with the help of the following measurements:

- Breast Volume (*volume*): calculated with the breast inferior radius $IR$, the lateral radius $LR$, the medial radius $MR$ and the breast depth is given by $min_y$ using a formula presented in Kayar et al. (2011);
- Skin surface area (*area*): calculated with the same elements as the volume but adding the breast base radius $BR$ and the distance between the top of the breast and the nipple ($TR$);
- Height of the breast ($min_z$): difference between the highest point and the lowest point of the breast;
- Frontal Depth ($min_y$): distance between the torso and the most frontal point of the breast (can be obtained with a normal or squared ruler);
- Back Depth ($max_y$): distance between the torso and the back of the breast;

Other measurements could be considered, but our choice is mainly driven by the practical capacity of a doctor to achieve the measures on every patient. The breast volume and skin surface area can be obtained using a similar strategy to the one presented in Liu (2009), while the others can be obtained using a measuring tape. Using other equipment it is also possible by using the reconstructed mesh elements (cells for volume and the area of the faces located in the breast's skin) where the height of the breast, frontal depth and back depth can be obtained with a bounding box. Considering the coordinate system of figure 26, the height of the breast, the frontal depth and back depth become the minimum in $z$, the minimum in $y$ and the maximum in $y$ respectively. Notice that, from a mathematical point of view, the number of measurements has to be greater than the number of parameters. One performs five measures for each position, totalling 15 data with the three positions, the patient bending forward approximately $45^{o}$ (just like is shown in the right view of figure 26) and the patient on all fours. Note that some of these measurements might not play a significant role in estimating the breast parameters.

### 4.2.3 *Parameters identification*

Lets denote $\Lambda = (\Lambda_g, \Lambda_m)$ as the six parameters which represents the geometrical and the mechanical degrees of freedom respectively. First, for the two geometrical parameters

$\Lambda_g = (R, H)$, one creates the stress-free configuration $\Omega_g$, as shown in Figure 26, which defines the operator

$$\Lambda_g \to \Omega_g = \Omega_g(\Lambda_g). \tag{54}$$

Secondly, for a set of mechanical parameters $\Lambda_m = (\lambda_{br}, \mu_{br}, \lambda_{sk}, \mu_{sk})$, minimisation of the energy $\mathcal{J}(f) = \mathcal{J}(f; \Omega_g, \Lambda_m) = \mathcal{J}(f; \Lambda)$ over the set of regular functions $f$ defined on $\Omega_g$ provides the solution $f(p) = f(p; \Lambda)$. At last, applying the displacement over the reference domain provides the deformed domain $\Omega(\Lambda) = f(\Omega_g)$ which, at the end of the day, defines the operator

$$\Lambda \to (\Lambda_m, \Omega_g) \to \Omega(\Lambda) = f(\Omega_g) = \left\{ f(p, \Lambda);\ p \in \Omega_g \right\}. \tag{55}$$

Third, to perform the parameters identification, one considers the measurements mentioned previously: the volume, the surface area of the breast's skin and the three sizes, namely $\min_y$, $\min_z$ and $\max_y$. Figure 29 shows the last three measurements in a digital context. The measures are performed for three patient positions, stand up, inclined (45 degrees)

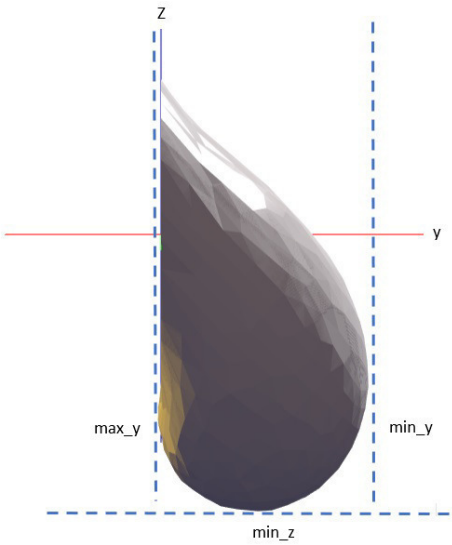



Figure 29: Size characteristic ($min_y$, $max_y$ and $min_z$) of a breast (stand-up position)

and horizontal, providing 15 real values denoted $\overline{M} = (\overline{M}_1, \cdots, \overline{M}_{15})^T$. On the other hand, given a set of parameters and solving three times the direct problem corresponding to the three positions provides the vectors $M(\Lambda) = M(\Omega(\Lambda)) = (Mk_1, \cdots, M_{15})^T$. The problem

consists in seeking the set of parameters such that $M(\Lambda)$ is equal to $\overline{M}$ in the least square sense. Introducing the cost function

$$E(\Lambda) = \sum_{i=1}^{n} (M_i(\Lambda) - \overline{M_i})^2, \tag{56}$$

one seeks for $\overline{\Lambda} = \arg\min_{\Lambda} E(\Lambda)$ which minimises the errors between the experimental data and the theoretical solution values.

To provide the optimal set of parameters, an iterative process is considered by evaluating the sequences $\Lambda^{k+1} = \Lambda^k + \Delta\Lambda^k$ such that $E(\Lambda^{k+1}) < E(\Lambda^k)$. The conjugate gradient was experimented at first but it proved to be very slow and have problems converging. Knowing that the problem consists in minimising a sum of square values (in this case the square differences) the Gauss-Newton method was experimented. This method is mentioned in literature as being able to efficiently solve these types of problems that are not only from breast parameter estimation as in Rajagopal et al. (2007) but also other problems like heat transfer coefficients as in Marques et al. (2016). So, the Gauss-Newton method was used to take advantage that function $E$ casts in the specific form

$$E(\Lambda) = (M_i(\Lambda) - \overline{M_i})^T (M_i(\Lambda) - \overline{M_i}). \tag{57}$$

To obtain the optimal parameters, the variation of the measurements is evaluated by changing the values of $\Lambda$. For that purpose one considers the first-order Taylor series expansion

$$M(\Lambda + \Delta\Lambda) \approx M(\Lambda) + \frac{\partial M}{\partial \Lambda}(\Lambda)\,\Delta\Lambda = M + \frac{\partial M}{\partial \Lambda}\,\Delta\Lambda \tag{58}$$

where $\dfrac{\partial M}{\partial \Lambda}$ is the Jacobian matrix. From relations 57 and 58 one deduces the first term of error for a perturbation of the parameters $\Delta E = E(\Lambda + \Delta\Lambda) - E(\Lambda)$

$$\Delta E = -\Delta\Lambda^T \Big[ 2 \Big( \frac{\partial M}{\partial \Lambda} \Big)^T (\overline{M} - M(\Lambda)) - \Big( \frac{\partial M}{\partial \Lambda} \Big)^T \frac{\partial M}{\partial \Lambda}\,\Delta\Lambda \Big].$$

Thus, by identification, one deduces

$$\frac{\partial E}{\partial \Lambda}(\Lambda) \approx -2 \Big( \frac{\partial M}{\partial \Lambda} \Big)^T (\overline{M} - M(\Lambda)) + \Big( \frac{\partial M}{\partial \Lambda} \Big)^T \frac{\partial M}{\partial \Lambda}\,\Delta\Lambda \tag{59}$$

One seeks a perturbation $\Delta\Lambda$ that minimises $E$, i.e., $\frac{\partial E}{\partial \Lambda} = 0$. Therefore Equation 59 provides the expression

$$\Delta\Lambda = 2 \Big[ \Big( \frac{\partial M}{\partial \Lambda} \Big)^T \frac{\partial M}{\partial \Lambda} \Big]^{-1} \Big( \frac{\partial M}{\partial \Lambda} \Big)^T (\overline{M} - M(\Lambda)) \tag{60}$$

Since a direct computation of the Jacobian matrix is not possible, it becomes necessary to numerically evaluate each partial derivative setting

$$\partial_i M(\Lambda) \approx \frac{M(\Lambda + \varepsilon e_i) - M(\Lambda)}{\varepsilon} \in \mathbb{R}^n \tag{61}$$

where $\varepsilon$ is fixed by the user in function of the minimisation problem and $e_i = (\delta_{ij})_{j=1,\cdots,n}$.

## 4.3 PROBLEM DISCRETISATION

To carry out the numerical simulations a discretisation of the model and method are introduced. The model discretisation is similar to the one presented in chapter 3. $\mathcal{T}_{h,g}$ denotes a mesh of the gravity free domain $\Omega_g = \Omega_g(\Lambda_g)$ constituted of $I$ non-overlapping tetrahedron cells $\tau_i$, $i = 1, \ldots, I$, $N$ vertices $P_n = (P_{nx}, P_{ny}, P_{nz}) \in \mathbb{R}^3$, $n = 1, \ldots, N$, $T_k$, $k = 1, \ldots, K$ which represents the faces of the tetrahedrons of the mesh that belong to $\Gamma_F$, and $T_j$, $j = 1, \ldots, J$ which represents the faces of the tetrahedrons of the mesh that belong to $\Gamma_B$. The discrete domain is described by

$$\Omega_{g,h} = \bigcup_{\tau_i \in \mathcal{T}_{h,g}} \tau_i.$$

Vector $\Phi_h$ corresponds to the new configuration. Not all the entries of $\Phi_h$ are unknowns of the problem since some of them are characterised by the boundary conditions. So, again, a sub-vector $X_h$ of $\Phi_h$ is obtained with only the unknown values that shall be used in the minimisation process. The boundary conditions operator is denoted in the same way as in equation 34 but with a distinct set of boundary conditions: condition 52 yields that $f_n = P_n$ for any $P_n \in \gamma_{\mathrm{D}}$ while relation 53 implies that for any $P_n \in \Gamma_{\mathrm{B}}$, one sets $f_{ny} = 0$ to maintain interface $\Gamma_{\mathrm{B}}$ on the trunk plane $y = 0$.

### 4.3.1 *Numerical approximation to the discrete energy minimiser*

The discrete version of the energy functional ($J_h(X_h) = J_h^1 + J_h^2 + J_h^3 + J_h^4$) is the same as the one presented in sections 3.2.3 and 3.4.2. For a given vector $X_h$ and the boundary conditions, one deduces vector $\Phi_h = \mathcal{B}(X_h)$, hence the continuous linear piecewise function $f_h$ that is used to compute the discrete energy functional. This leads to build the operator

$$X_h \to \Phi_h = \mathcal{B}(X_h) \to J_h(X_h; \Lambda) = \widehat{J}_h(\Phi_h; \Lambda) \in \mathbb{R}$$

where $\Lambda$ is a given set of parameters. The numerical solution that is sought provides the vector $\Phi_h = \Phi_h(\Lambda)$ which minimises the energy of the discrete mechanical system.

Conjugate gradients method is employed to determine the minimiser $\bar{X}_h$ of the discrete functional $J_h(X_h;\Lambda)$ in a similar way as shown in equation 40.

### 4.3.2 *Cost function discretisation*

Let $\Lambda = (\Lambda_m, \Lambda_g)$ be a set of parameters. One deduces the discrete gravity free configuration $\Omega_{g,h}$ which provides the operator $\Lambda_g \rightarrow \Omega_{g,h}$. Then one computes the solution $\Phi_h(\Lambda)$ minimising the discrete energy functional $\mathcal{J}_h(f_h) = \mathcal{J}_h(f_h;\Omega_{g,h},\Lambda_m) = \mathcal{J}_h(f_h;\Lambda)$. Then the final discrete breast $\Omega_h(\Lambda)$ is deduced after applying the deformation

$$\Omega_h(\Lambda) = \{f_h(p,\Lambda); p \in \Omega_{g,h}\}$$

with $f_h$ being the function that provides the new position. The measures on $\Omega_h(\Lambda)$ are then assessed and so one gets vector $M_h(\Lambda)$, *i.e.* one defines an operator that from a set of parameters $\Lambda$ obtains the correspondent measurements $M_h(\Lambda)$. This operator is denoted as $\Lambda \rightarrow M_h(\Lambda)$. The objective of the estimation method is to obtain the best possible approximation between the experimental measurements $\overline{M_i}$ and the measurements $M_h$ obtained with the refined parameters $\Lambda$ ($M_h(\Lambda)$). For that purpose the method employs the discrete cost function which reads

$$E_h(\Lambda) = \sum_{i=1}^{n} k_i \Big( M_{h,i}(\Lambda) - \overline{M_i} \Big)^2 \tag{62}$$

where $k_i$ is a pondering factor that depends of the type of measure and its purpose is to balance the cost function so that every element as a similar importance. This happens because, the volume is measured in cubic meters, the surface area is evaluated in squared meters and the other 3 different measure types are evaluated in meters. While some measurements are evaluated around 0.1 like $min_y$, others may be evaluated around 0.001 (volume expressed in cubic meters). An alternative cost function which uses a normalisation of the measurements could also be employed. The usage of such function could also solve the measurement range value difference problem. This function would be denoted as

$$EN_h(\Lambda) = \sum_{i=1}^{n} \Big( \frac{M_{h,i}(\Lambda) - \overline{M_i}}{\overline{M_i}} \Big).$$

The iterative process is applied to the discrete versions to get the best parameter set $\Lambda$ by calculating successive variation of the parameters ($\Delta\Lambda^n$) until $E_h(\Lambda^n) - E_h(\Lambda^{n+1}) < \epsilon_m$ for a given threshold $\epsilon_m$ as well as when $E_h(\Lambda) < \epsilon_e$ for a given threshold $\epsilon_e$ and for a predetermined number of max iterations.

## 4.4 PROOF OF CONCEPT

To assess the model quality and robustness, several numerical experiments were carried out using a manufactured solution. The main reason to use synthetic solution is the limitation of obtaining real-life cases (waiting for the doctors to obtain patients to perform breast surgeries, as well as consents) while on the other hand synthetic cases can be generated with ease using the validated breast model presented in chapter 3 and published in Lopes et al. (2017a) (the results with real-life cases are shown in chapter 8). In any case, the estimation method receives the measurements from the doctor/surgeon (or from the digital model in terms of synthetic cases) and tries to estimate the set of geometrical and mechanical parameters that can provide an approximation such that the measurements obtained with those parameters are very close to the reference values. The main objectives of these experiments (according to the hypothesis mentioned in subsection 4.2.2) are:

1. Check if the method is capable of recovering the reference values for the parameters that match the reference measures;
2. Analyse the impact that each measurement has on the breast model's parameters;
3. Evaluate the method's robustness.

To this end, one evaluates the initial guess accuracy, the impact of the breast mesh granularity and the relationship between the measurements and the parameters. In theory, initial guesses that are closer to the reference values have an easier time converging to the reference values rather than initial guesses which are far.

In the conducted experiments, three different mesh sizes ($\bar{h}$) were used: coarse with 226 nodes and 1116 elements; medium with 1014 nodes and 5030 elements; thin with 2498 nodes and 12986 elements. Figure 30 shows an example of the visual aspect of the

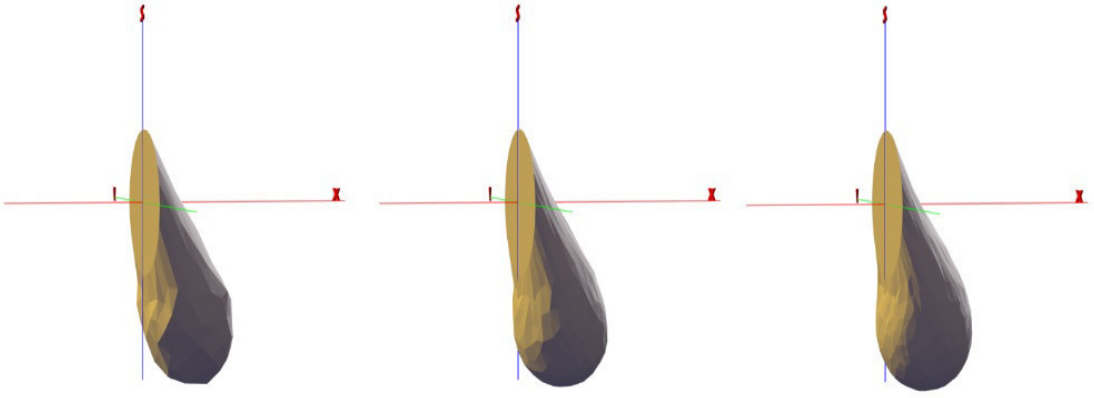


Figure 30: Examples of a breast configuration using different granularity: coarse (left), medium (middle) and thin (right)

breast configuration for the patient standing up with the reference parameters $\Lambda^{ref} = (R = 0.12, H = 0.05, \lambda_{br} = 1100, \mu_{br} = 150, \lambda_{sk} = 7000, \mu_{sk} = 1400)^T$. The thin mesh (right configuration) is a better approximation to a real breast than the coarse mesh (left configuration) and will result in better approximations to the breast parameters.

There exist some relationships between the parameters and the measurements. For example, larger volumes might mean that the values of the geometrical parameters are higher. However, the same can happen for larger values of the mechanical parameters which can mean that the breast tissues can undergo bigger deformations (Maurel et al. (1998)). So the relations are not linear and a careful analysis of them is done using specific tests.

### 4.4.1 *Feasibility*

To check that the method is functional, a synthetic breast is generated using the reference parameters ($\Lambda^{ref}$). With $\overline{\Lambda} = \Lambda^{ref}$, one applies the operators 54 and 55 that provide the correspondent measurements $\overline{M}$. One now considers the reverse problem taking as an initial guess ($\Lambda_0$) a perturbation of the real parameter ($\Lambda^{ref}$) controlled by a maximum variation index in %. One runs the parameter identification method to find a set of parameters $\Lambda_h$ that approximates $\overline{\Lambda}$. The process is repeated $N$ times in order to statistically analyse the convergence to the reference parameters. At last, one computes the relative difference between each parameter $i$ of $\overline{\Lambda}$ and $\Lambda_h$ using equation 63.

$$E_{param}(i) = \frac{|\overline{\Lambda}_i - \Lambda_{h,i}|}{\overline{\Lambda}_i}. \tag{63}$$

Besides the difference between the parameters, one also registers the number of iterations required by the method to converge, hence allowing to evaluate if the method is fast or slow (the smaller number of iterations the faster the method). The method converges when the criteria mentioned in section 4.3 is met, *i.e.* the value obtained with the cost function 62 ($E_h(\Lambda)$) is inferior to a given $\varepsilon_e = 5e^{-6}$ and the variation of the error between iterations is smaller than a given $\varepsilon_m = 1e^{-9}$. These criteria ensure that the error obtained is very small (for example the difference between the reference and the estimated value for $min_y$ is inferior to the millimetre). To prevent the method to run indefinitely, the maximum number of iterations was set to 25. The tests are carried out for different initial guesses as perturbations of the reference solution, starting with small variations up to 10%, going up to 30% and, finally, for large variations up to 60%.

More precisely, the initial guesses are gathered in a set of $N$ vectors $s^n \in \mathbb{R}^6$, $n = 1, \cdots, N$ generated such that each component $s_i^n$ is chosen randomly with the normal law $\mathcal{N}(0, g_{init})$, where $g_{init}$ is the value of the variation of the parameters $\Lambda^{ref}$ (0.1 representing the variation of 10%, 0.3 representing the variation of 30% and 0.6 representing the variation of 60%).

Then one sets $\Lambda_i^n = (1 + s_i^n)\Lambda_i^{ref}$, $i = 1, \cdots, 6$, $n = 1, \cdots, N$, eliminating the non-physical situations (negative values). A total of 20 tests were performed for each variation of the initial guess and for different meshes granularity, namely coarse, medium and thin. A statistical analysis of the results in tables 2, 3 and 4 is presented.

| **Estimation with Initial guess of 10%** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mesh Granularity** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **error per parameter** | **number of iterations** | **cost function** |
| ***Coarse (N=226)*** | 0.06% | 0.73% | 2.13% | 1.53% | 2.11% | 3.77% | 1.72% | 3 | $9.37e^{-7}$ |
| ***Medium (N=1014)*** | 0.12% | 0.87% | 2.00% | 0.96% | 0.80% | 3.43% | 1.36% | 3 | $8.59e^{-7}$ |
| ***Thin (N=2498)*** | 0.13% | 0.86% | 1.93% | 0.91% | 1.04% | 3.16% | 1.34% | 3 | $8.62e^{-7}$ |

Table 2: Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method with initial guesses which differ the reference parameters up to 10%. Mean values of the error per parameter, the number of iterations necessary to converge and the value of the cost function that measures the difference between the reference measurements and the measurements obtained with the estimated parameters.

When the initial guess ranges in intervals up to 10% of the reference parameters, the identification parameter method provides approximation $\Lambda^n$ with good accuracy and converged with 3 iterations on average (table 2). The average relative error per parameter is around 1.5% while the average value obtained with the cost function is inferior to $1.0e^{-6}$. Notice that the relative error obtained for the geometrical parameters ($R$ and $H$) is smaller when compared with the error obtained for the mechanical parameters ($\lambda_{br}$, $\lambda_{sk}$, $\mu_{br}$ and $\mu_{sk}$). Using a 10%-variation initial guess corresponds to small perturbations and it is not enough for asserting the efficiency of the inverse method. Concretely, it is quite difficult for a practitioner to give an estimate within the 10% range from the correct set of parameter (as an initial guess). To this end, the variation of the initial guess parameters was increased such that the difference between the initial guess parameters would differ up to 30% of the reference parameters. Using the same conditions as in the previous test, for 20 different cases, table 3 reports the relative error average.

The 30%-variation parameters use as a guess still provide good performances. The method is still converging with an average of 5 iterations. The major difference between these and the previous tests concerns the increase of the relative error obtained with the mechanical parameters of the breast skin $\lambda_{sk}$ and $\mu_{sk}$. They range from approximately $0.91 - 1.53\%$ and $3.16 - 3.77\%$ respectively to $2.32 - 2.86\%$ for $\lambda_{sk}$ and $8.97 - 9.55\%$ for $\mu_{sk}$. The other parameters error also increases but in a smaller magnitude. The main discrepancy is associated to $\mu_{sk}$ and leads to the increase of the relative error average of approximately 3%. Despite such drawback, the average value of the cost function is still below the $5e^{-6}$

| Estimation with Initial guess of 30% | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mesh Granularity** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **error per parameter** | **number of iterations** | **cost function** |
| ***Coarse (N=226)*** | 0.28% | 0.90% | 2.50% | 2.86% | 2.56% | 9.55% | 3.11% | 5 | $1.36e^{-6}$ |
| ***Medium (N=1014)*** | 0.39% | 1.01% | 2.38% | 2.51% | 2.21% | 9.47% | 3.00% | 5 | $1.19e^{-6}$ |
| ***Thin (N=2498)*** | 0.31% | 1.28% | 2.32% | 2.32% | 2.14% | 8.97% | 2.89% | 5 | $1.13e^{-6}$ |

Table 3: Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method with initial guesses which differ the reference parameters up to 30%. Mean values of the error per parameter, the number of iterations necessary to converge and the value of the cost function that measures the difference between the reference measurements and the measurements obtained with the estimated parameters.

threshold which ensures that the estimation is still good. The larger value of error associated with the $\mu_{sk}$ parameter suggest that this parameter impact of the breast shape and size might not be significant, *i.e.* a poor estimation of this parameter might not result in significant changes in the measurements. The increase in the number of iterations necessary for the method to converge is understandable knowing that the initial guess is further away from the reference values.

Next steps consists in checking how well the method works with initial guesses that are even further away and a new set of tests is carried out where the initial guess parameters differ from the reference values up to 60%. Once again an additional set of 20 different cases are evaluated, whose results can be seen in table 4.

| Estimation with Initial guess of 60% | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mesh Granularity** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **error per parameter** | **number of iterations** | **cost function** |
| ***Coarse (N=226)*** | 0.26% | 1.67% | 2.84% | 3.34% | 2.32% | 11.44% | 3.64% | 8 | $2.10e^{-6}$ |
| ***Medium (N=1014)*** | 0.46% | 1.62% | 2.90% | 3.48% | 2.10% | 11.05% | 3.60% | 8 | $1.97e^{-6}$ |
| ***Thin (N=2498)*** | 0.44% | 1.83% | 2.87% | 3.35% | 2.06% | 10.70% | 3.54% | 8 | $1.89e^{-6}$ |

Table 4: Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method with initial guesses which differ the reference parameters up to 60%. Mean values of the error per parameter, the number of iterations necessary to converge and the value of the cost function that measures the difference between the reference measurements and the measurements obtained with the estimated parameters.

For 60%-variation initial parameters, the method still converges and good approximations for the reference parameters are reported. Once again the relative error for $\mu_{sk}$ and $\lambda_{sk}$ increases in average to approximately 11% and 3.5% respectively. Such errors do not, however, deteriorate the value of the cost function significantly (the difference between the reference measurements and the estimated values approximating $2e^{-6}$) and so that value remains smaller than the $5e^{-6}$ threshold. The increased relative error of $\mu_{sk}$ supports the idea that this parameter might not be very significant or perhaps the considered measurements are not suited to recover this particular parameter. As expected the number of iterations necessary to converge increase to approximately 8.

Overall, these results prove that this method succeeds in finding a good approximation to the real parameters of a breast when given a set of measurements with a large spectrum of the initial condition. In relation to the impact of the mesh granularity, it is difficult to assert any significant finding. However, the results suggest that coarse meshes can provide better approximations to the geometrical parameters while thinner meshes can provide better approximations to the mechanical parameters. The worse performance of the method when estimating the mechanical parameters of the skin, mainly $\mu_{sk}$, point out to a lack of sensitivity of this parameter in relation to the measures considered by the estimation method. As expected, worse initial guesses lead to more iterations to find a good approximation to the reference parameters (3 iterations for an initial guess difference up to 10%, 5 iterations for an initial guess difference up to 30% and approximately 8 iterations for an initial guess difference up to 60%).

### 4.4.2 *Sensitivity Analysis*

Sensitivity analysis is an important issue for assessing the robustness of the method and evaluating the measures that preponderantly influence the parameters. Moreover, potential errors might derive from the performed measurements in a practitioner's office and one has to assess the consequences of such errors on the parameters. At last, it is a good indicator of the ill-posed inverse problem. Minimising the cost function $\Lambda \rightarrow E_h(\Lambda)$ provides the relation $\overline{M} \rightarrow \Lambda(\overline{M})$, *i.e.* the best parameters that fit the measurements. The sensitivity analysis aims at assessing the impact of the measure variations on the parameters evaluation. Low sensitivity of parameter $R$ with respect to measure $m_1$ means that the relative partial derivative

$$d_{R,m_1} = \frac{1}{R(\overline{M})} \frac{\partial R}{\partial m_1}(\overline{M}) << 1$$

while an high sensitivity is obtained if $d_{R,m_1} >> 1$.

It is assumed that practitioners provide measurements $m_i$ with a relative error up to 10%. To simulate the error impact, two new sets of parameters are defined, namely $\Lambda(m_i+)$ and

$\Lambda(m_i-)$ which correspond to a positive and a negative variation of 10% of parameter $\overline{m}_i$, while the others are fixed. To calculate the sensibility value one approximates the relative derivative with the expression

$$\frac{\partial \Lambda_j}{\partial \overline{m_i}} = \frac{\Lambda_j(m_i+) - \Lambda_j(m_i-)}{0.2 * m_i * \Lambda_j(\overline{M})}. \tag{64}$$

It results a $6 \times 15$ matrix that approximate the Jacobi matrix. Computations are achieved with the thin mesh and the results are reported in Table 5 with the parameters in row and the measures in columns.

Such data needs to be handled and interpreted with caution since the variation derives from the $\pm 10\%$ prescribed and computational errors due to the discretisation may degrade the approximation accuracy. Therefore one has two variations to take into consideration

$$\delta\Lambda_j(m_i) = \Lambda_j(m_i+) - \Lambda_j(m_i-)$$

which corresponds to the error of the variation of the measure $m_i$ and

$$\Delta\Lambda_j = \Lambda_j(\overline{M}) - \overline{\Lambda_j}$$

which corresponds to the error of the discretisation with respect to the exact parameters $\overline{\Lambda}$.The data can be classified into three groups. The red groups are the data where the discretisation error is preponderant, namely $\delta\Lambda_j(m_i) < 2\Delta\Lambda_j$. It results that the derivative approximation error is mainly due to the discretisation and not the variation of the measurement. The yellow corresponds to the data where $2\Delta\Lambda_j < \delta\Lambda_j(m_i) < 5\Delta\Lambda_j$, *i.e.* the discretisation errors and the variation errors are of same order. At last, the green group provides the data such that $5\Delta\Lambda_j < \delta\Lambda_j(m_i)$ which guarantee that the derivative is the consequence of the 10% variation of the parameter $m_i$ and can be considered as a correct approximation of the partial derivative. The values in the table are mapped according to the colours of reliability.

The impact of measurements $m_1$, $m_6$ and $m_{11}$ corresponding to the volume for the three positions cannot be considered as significant since the error deriving from the discretisation is larger than the error of variation, hence no significant information can be deduced. The same happens with $max_y$ given by the measures $m_{10}$ and $m_{15}$. It suggests that a potential misevaluation by a doctor (error in the measurement of 10%) of this measurements might not have a great impact on the estimated parameters. The other measures evaluations have an impact since they provide an admissible approximation of the partial derivatives. The main conclusions are:

- Parameter $R$ does not present a great sensitivity to the measurements except with measurement $m_9$ corresponding to the $min_z$ in the lying down position. This means

| | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ |
|---|---|---|---|---|---|---|
| $m_1$ | 0.43 | 3.05 | 12.89 | 1.55 | 7.34 | 0.58 |
| $m_2$ | 1.92 | 2.05 | 39.71 | 7.72 | 60.54 | 10.60 |
| $m_3$ | -0.66 | -6.65 | -18.53 | -51.29 | -14.83 | -144.72 |
| $m_4$ | -0.93 | -2.54 | -49.88 | -8.67 | -15.66 | -30.36 |
| $m_5$ | 1.15 | 8.62 | 19.62 | 35.48 | 2.31 | 106.97 |
| $m_6$ | 0.40 | 0.87 | 15.49 | 3.23 | 3.97 | 6.37 |
| $m_7$ | 3.80 | 5.67 | 59.84 | 38.28 | 53.72 | 137.28 |
| $m_8$ | -0.62 | -0.37 | -53.02 | -16.30 | -8.70 | -77.01 |
| $m_9$ | -10.51 | -2.34 | -52.03 | -1.78 | -8.12 | -12.00 |
| $m_{10}$ | 2.99 | 25.17 | 67.18 | 43.56 | 1.95 | 136.82 |
| $m_{11}$ | 0.59 | 1.68 | 9.57 | 3.85 | 0.03 | 11.09 |
| $m_{12}$ | 0.51 | 5.75 | 6.72 | 4.49 | 21.24 | 4.14 |
| $m_{13}$ | -1.58 | -8.58 | -28.47 | -12.19 | -1.05 | -39.19 |
| $m_{14}$ | -4.71 | -0.68 | -20.33 | -21.35 | -2.87 | -52.51 |
| $m_{15}$ | 2.15 | 13.20 | 114.59 | 8.43 | 20.90 | 38.76 |

Table 5: Table with the sensibility analysis of the parameters. (Red - unreliable values, yellow - slightly reliable values, green - reliable values)

that an error in a particular measurement in the order of 10% will not cause a large variation in the estimated value.

- The same conclusion is drawn for parameter $H$ although one observes a slightly larger sensitivity (in average) than the $R$ parameters.

- It is possible to observe a great sensitivity for $\lambda_{br}$ with respect to the measurements, especially for the skin area of the patient lying down $m_7$ with a value of 59. The error of the practitioner for this particular measure will have a high impact on the $\lambda_{br}$ parameter. It is also noticeable that the measurements may act in opposite way since there is a positive derivative for $m_7$ and a negative derivative for $m_8$ and $m_9$ ($min_y$ and $min_z$) of about $-50$. Such a situation leads to a more complex analysis since several parameters have antagonist actions.

- The $\mu_{br}$ parameter present a similar behaviour to $\lambda_{br}$ but with a smaller impact, except for the $min_y$ patient standing up measurement $m_3$ with a value of $-51$. Positive derivative associated to measurement $m_5$ with value 35 denote the antagonist action between the two measures.

- Parameter $\lambda_{sk}$ has a high sensitivity with respect to the $min_y$ measurements over the 3 positions with values of 60, 53, 21 respectively. Other parameters provide negative derivative and have an antagonist effect.

- The coefficient $\mu_{sk}$ is the most sensitive of the six parameters considered in this study which explains the high volatility when performing optimisation with meshes of different granularity. High positive values 137 and 106 were recorded for the derivatives with respect to $m_5$ and $m_7$ whereas negative values of $-144$ and $-77$ were obtained for the derivatives with $m_3$ and $m_8$. It is safe to say that accurate value for $\mu_{sk}$ requires a careful evaluation of the measures. This high volatility can also mean that the measurements used are not ideal to recover this parameter.

Overall, these results point out to a future necessity of considering different measurements in order to estimate the breast model parameters, mainly for $\mu_{sk}$ whose sensitivity to some measurements is very high. The fact that the discretisation error for 5 different measures ($m_1$, $m_6$, $m_{10}$, $m_{11}$ and $m_{15}$) is larger than the errors that come from the evaluation of the partial derivatives suggests that maybe the other measurements play a more significant role in the estimation process.

### 4.4.3 *Robustness*

The previous examples concerned the evaluation of the parameters for a specific breast with the set of reference parameters $\Lambda^{ref}$. It is important to evaluate how the method performs for other types of breasts, hence, additional tests were done. To this end, a pool of 75 cases is generated where each case consists of a set of parameters that are generated using a similar configuration as the one present in subsection 4.4.1.

A Gaussian distribution of the parameters centred on $\Lambda^{mean} = (R = 0.1, H = 0.04, \lambda_{br} = 1000, \mu_{br} = 400, \lambda_{sk} = 8000, mu_{sk} = 1600)^T$ is used. The choice for the values of $\Lambda^{mean}$ comes from multiple experiences and from an optical assessment of the generated breast which was not too big or too small (this process could have been done with a different set of parameters). $N$ vectors $s^n \in \mathbb{R}^6$, $n = 1, \cdots, N$ are generated such that each component $s_i^n$ is chosen randomly with the normal law $\mathcal{N}(0, g_{init})$, where $g_{init}$ is the value of the variation of the parameters $\Lambda^{mean}$. Then one sets $\Lambda_i^n = (1 + s_i^n)\Lambda_i^{mean}$, $i = 1, \cdots, 6$, $n = 1, \cdots, N$, eliminating the non-physical situations (negative values). This process is done for three different values of $g_{init} = \{0.1; 0.3; 0.5\}$. For a value of $g_{init} = 0.5$, each parameter $\Lambda_i^n$ will vary up to 50% from $\Lambda_i^{mean}$, for a value of $g_{init} = 0.3$ it means that the parameters will vary up to 30% and finally for a value of $g_{init} = 0.1$ it means that the parameters will vary up to 10%. The 75 cases are split between the three values of $g_{init}$ and are performed for each mesh size (which results in a total of 225 tests).

For any set of reference parameters that comes from the pool $\overline{\Lambda}$, one computes the three configurations of the breast under a gravitational field (up, bend forward and all fours) that provide the reference set of measurements

$$M(\overline{\Lambda}) = \overline{M}_{\bar{h}} = (\overline{M}_{\bar{h},1}, \cdots, \overline{M}_{\bar{h},15})^T$$

where $\bar{h}$ characterises the mesh granularity. Then one uses $\overline{M}_{\bar{h}}$ as the reference measurements and uses $\Lambda^{mean}$ as the initial guess. A statistical analysis of the results in tables 6, 7 and 8 is presented.

| **Estimation with Parameter derivation up to 10%** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mesh Granularity** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **error per parameter** | **number of iterations** | **cost function** |
| ***Coarse (N=226)*** | 1.98% | 2.56% | 4.51% | 6.22% | 1.66% | 8.62% | 4.26% | 4 | $2.73e^{-06}$ |
| ***Medium (N=1014)*** | 2.29% | 3.44% | 4.24% | 6.13% | 1.63% | 8.10% | 4.31% | 4 | $2.53e^{-06}$ |
| ***Thin (N=2498)*** | 2.36% | 3.66% | 4.17% | 5.96% | 1.60% | 7.92% | 4.28% | 4 | $2.46e^{-06}$ |

Table 6: Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method using the same initial guess ($\Lambda^{mean}$) which differ the reference parameters up to 10%.

The identification parameter is still efficient and provides a good approximation to the reference parameters. Despite the increase of the error per parameter to approximately 4.3%, the difference between the reference measurements and the estimated measurements is still inferior to the accuracy threshold of $5e^{-6}$ (cost function value around $2.5e^{-6}$). The mechanical parameters of the breast's skin are the ones that present higher values for relative error ($\approx$ 6% and $\approx$ 8% respectively). The number of iterations as also augmented to 4 iterations. When comparing the values of table 6 to table 2, it is possible to observe that there is a substantial error growth. A possible explanation is that the data curve associated to the synthetic breast used to obtain the results of table 2 (curve obtained with the values of the cost function depending on the breast model parameters), lye in the convergence basin or, on contrary, are far from the minimum point.

Table 7 reports similar results regarded to table 6. The main change is reflected on the breast skin mechanical parameters $\lambda_{br}$ and $\mu_{br}$ which increase from approximately $6 - 6.2\%$ and $8 - 8.6\%$ to $7.9 - 8.7\%$ and $13 - 14\%$ respectively. This rise of the relative error leads to the increase of the error per parameter to approximately 5.7%. Nevertheless, the cost function value is still around $3.3e^{-6}$ which means that it remains under the threshold value that ensures a good approximation. The number of iterations also increase to 7 and 8

| Estimation with Parameter derivation up to 30% | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mesh Granularity** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **error per parameter** | **number of iterations** | **cost function** |
| ***Coarse (N=226)*** | 2.12% | 2.48% | 4.74% | 8.77% | 2.46% | 14.01% | 5.76% | 8 | $3.37e^{-06}$ |
| ***Medium (N=1014)*** | 2.31% | 3.62% | 4.62% | 8.02% | 2.38% | 13.28% | 5.70% | 7 | $3.35e^{-06}$ |
| ***Thin (N=2498)*** | 2.52% | 3.78% | 4.61% | 7.85% | 2.21% | 12.85% | 5.64% | 7 | $3.31e^{-06}$ |

Table 7: Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method using the same initial guess ($\Lambda^{mean}$) which differ the reference parameters up to 30%.

iterations which is expected taking into account the highest difference between the initial guess parameters.

| Estimation with Parameter derivation up to 50% | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mesh Granularity** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **error per parameter** | **number of iterations** | **cost function** |
| ***Coarse (N=226)*** | 2.30% | 2.71% | 5.28% | 9.41% | 3.07% | 15.72% | 6.42% | 14 | $4.03e^{-06}$ |
| ***Medium (N=1014)*** | 2.42% | 3.78% | 5.11% | 9.24% | 2.68% | 15.19% | 6.40% | 13 | $3.86e^{-06}$ |
| ***Thin (N=2498)*** | 2.56% | 3.83% | 5.07% | 9.13% | 2.48% | 14.43% | 6.25% | 12 | $3.58e^{-06}$ |

Table 8: Difference between the reference parameters ($\overline{\Lambda}$) and the parameters obtained with the estimation method using the same initial guess ($\Lambda^{mean}$) which differ the reference parameters up to 50%.

Table 8 report substantial growth of the error per parameter ($\approx 6.4\%$) but the associated cost function ranges within the acceptable interval ($4e^{-6} < 5e^{-6}$). Once again, on average $\lambda_{br}$ and $\mu_{br}$ present the worst results. The number of iterations also increases to a large number of iterations (12 to 14).

The values reported in tables 6, 7 and 8 proves that the proposed estimation method works well. The method succeeds in providing good approximations to the reference parameters, although some case is considered quite difficult to solve (many iterations). It happens with a data curve obtained with parameters variations around $\Lambda^{mean}$. These curves shape might be non regular or quadratic when comparing them with the curve using the breast with the $\Lambda^{ref}$ parameters (results in tables 2, 3 and 4). Therefore, by refining the parameters, the iterative process slowly reaches a point where the value of the cost function goes below the set threshold of $5e^{-6}$. Such a situation is supported by the behaviour of

the parameters reported in subsection 4.4.2, namely the error compensation. For example, when the parameter $R$ is smaller than the reference value, that means (assuming that the other parameters are equal to the reference values) that the volume and the surface area is smaller than what it should be. So, in order to compensate that, the value of the $H$ increases. A better approximation could be obtained but would require a more restrictive criterion to provide a better approximation. However, as the value of the cost function ranges in an acceptable interval, further numerical improvements are not necessary.

## 4.5 PERFORMANCE EVALUATION

One aims at evaluating the implementation code performance since the computational cost of this method relies heavily on the cost of the simulation method. For instance, the parameter estimation method consumes huge amounts of time which can reach to hours of computation using a coarse and medium mesh and a day with a thin mesh. Even after code improvements, the simulation code (chapter 5) the process of estimating the breast model parameters using a thin mesh could take hours.

The thin mesh provides the best approximates of a real breast (figure 30). Therefore to achieve good approximation regarded of real cases, the thin mesh is the first choice. Due to the iterative procedure by successive approximations, the method performance depends on the quality of the initial guess. Roughly speaking, a poor initial guess leads to around 12 iterations to converge (Table 8) whereas a nice initial candidate may converge with just 4 iterations (tables 7 and 6 respectively). Moreover, recall that the number of iterations with a thin mesh would be respectively of 3, 5 and 8 iterations with initial guesses range in intervals 10%-variation, 30%-variation and 60%-variation around the reference parameter. To reduce the number of iteration with the thin mesh and improve the coding efficiency, the following is proposed:

1. Use a coarse mesh to run a few iterations and obtain an approximation to the reference parameters;

2. Use the results coming from the estimation with the coarse mesh and run additional iterations with the medium mesh refining the parameters even more;

3. Run the estimation method with the thin mesh and obtain the final set of parameters.

The time spent with a coarse and medium mesh is inferior to the time spent using a thin mesh. So, first candidate is created with the coarser mesh which is afterwards refined by successively thinner meshes until the last set of parameters become accurate.

For the same set of parameters, the measurements obtained with a coarse, medium and thin mesh are different due to the mesh quality. 25 different set of parameters were used

to generate the 25 sets of measures for each mesh size. The differences between the coarse and thin mesh as well as the difference between the medium and thin mesh are reported in table 9.

| **Diff Mesh** | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ | $m_6$ | $m_7$ | $m_8$ | $m_9$ | $m_{10}$ | $m_{11}$ | $m_{12}$ | $m_{13}$ | $m_{14}$ | $m_{15}$ | ***avg*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **T-C** | 4.98 | 5.37 | 11.63 | 6.92 | 79.35 | 7.26 | 18.33 | 11.91 | 6.83 | 32.56 | 4.09 | 3.37 | 3.81 | 3.38 | 13.82 | 14.24 |
| **T-M** | 1.23 | 1.75 | 5.27 | 3.97 | 21.49 | 4.95 | 14.95 | 5.99 | 4.60 | 25.90 | 0.96 | 0.61 | 0.96 | 1.06 | 4.45 | 6.54 |

Table 9: Differences in percentage between the measurements obtained with a coarse and thin mesh (T-C) as well as with a medium and thin mesh (T-M).

It is possible to observe that the relative difference of the measurements with respect to the thin and a coarse mesh is around of 14% (on average) for each measurement. The comparison between the measurements obtained with a thin and a medium mesh substantially reduces the difference with only 6.5%. As seen in subsection 4.4.2, a variation of the measures can affect greatly some parameters while not affecting others. With this in mind, and in order to test the hypothesis that the estimation process with a thin mesh could be optimised using coarser meshes, additional tests were done, based on the set of parameters $\Lambda^{ref}$ as in subsections 4.4.1 and 4.4.2. Using 10 different initial guesses, whose parameter values can differ the ones of $\Lambda^{ref}$ up to 60%, the average evolution of the parameters is recorded along with the current iteration number and displayed in figure 31.

The graphics present the parameters evolution in absolute percentage difference for the first 25 iterations. Note that most parameters, with the exception of $\mu_{sk}$, stabilise/converge within iterations 5 and 7. Drawing comparisons with the parameters evolution for the coarse and medium mesh, it is possible to notice that the parameters converge to limit values closer to the reference with the medium mesh in comparison with the coarse mesh. Moreover, differences between the measurements follow the same pattern, namely 6.5% for the medium mesh and 14% for the coarse mesh. Nonetheless, the approximation of the geometrical parameters $R$ and $H$ are well approximated as well as parameters $\lambda_{sk}$ and $\mu_{br}$ both for the coarse and medium mesh (below 10%).

The figure highlights the high sensitivity of the $\lambda_{br}$ errors to the measurements. With a coarse mesh, the value blows up to almost 50% while the discrepancy reduces to 20% with a medium mesh. At last, with the thin mesh, $\lambda_{br}$ stabilises close to the reference values. $\mu_{sk}$ presents a more volatile nature, especially using a coarse mesh. With the usage of thinner meshes that issue is less visible but even using a thin mesh, it is possible to see the volatility of the $\mu_{sk}$ parameter. This results support the idea that the measurements used to estimate $\mu_{sk}$ are not ideal.

Overall these results show that it can be possible to optimise the breast parameter estimation process when the measurements come from a thin mesh by using coarser meshes to successively approximate the parameters to the reference values. In practice, these results

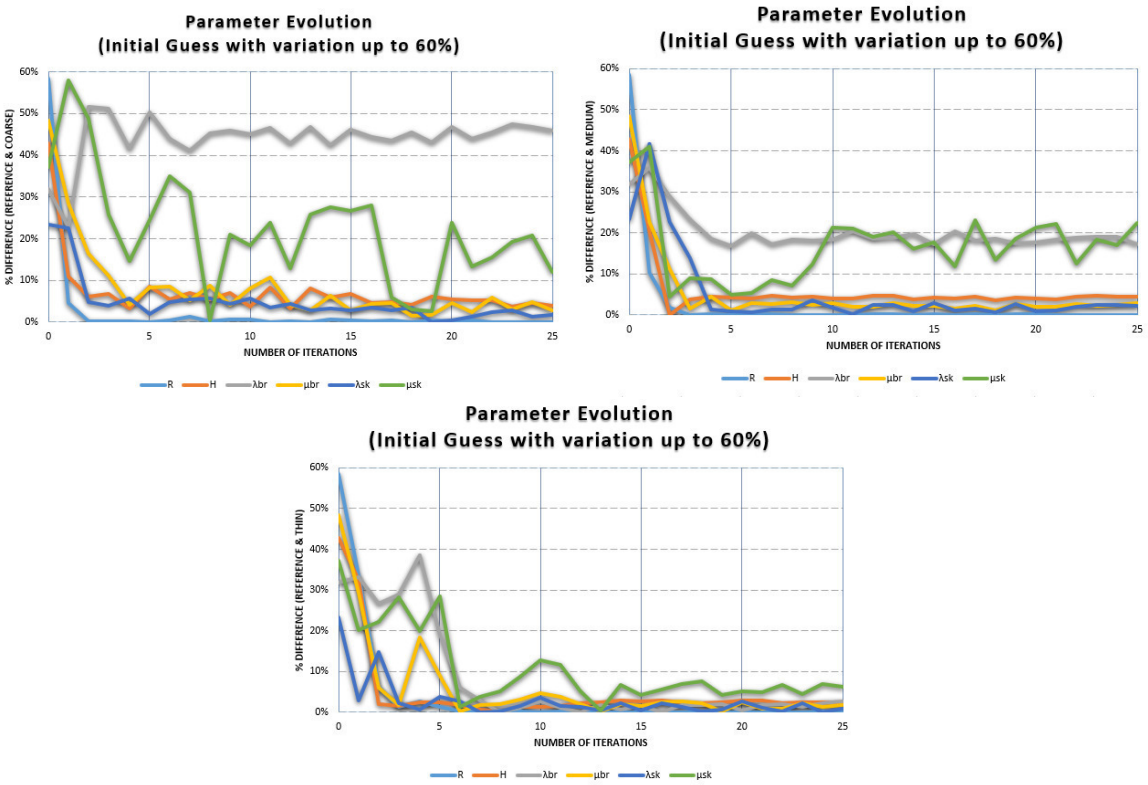



Figure 31: Parameter error evolution of the optimisation process with a coarse mesh (top left), a medium mesh (top right) and a thin mesh (bottom). Evolution is shown in the absolute percentage difference between the current values of the parameters in the iterations (from 0 to 25) and the reference parameters.

show that using a coarse mesh one can obtain a better approximation of the $R$, $H$, $\lambda_{sk}$ and $\mu_{br}$ parameters. Then using a medium mesh, one can refine $R$, $H$, $\lambda_{sk}$ and $\mu_{br}$ even further but more importantly, it is possible to approximate $\lambda_{br}$ and also $\mu_{sk}$ closer to the reference values (lower than 5%). In conclusion, the thin mesh only helps to refine $\lambda_{br}$ and $\mu_{sk}$ but does not bring any improvement with the other variables.

To measure the efficiency of the new method, an evaluation of the computational cost is done together with the value of the cost function. The computational cost is evaluated in terms of running time for achieving a full iteration with the coarse mesh and presented in table 10. This table reports that the computational cost increases with the mesh granularity

| **Mesh Size** | **Coarse** | **Medium** | **Thin** |
|---|---|---|---|
| **Cost Function** | $5.52e^{-4}$ | $7.26e^{-5}$ | $1.12e^{-6}$ |
| **Time** | 1 | 4.26 | 12.36 |

Table 10: Measurement's difference residual between the reference measurements and the measurements obtained with different sized meshes with the same values for the breast parameters $\Lambda$ and the work effort which is spent by the optimisation process.

since one parameter estimation iteration with the thin mesh required 12 more times for a coarse mesh iteration. Comparing the thin and medium mesh, that ratio is of approximately 3. The evolution of the cost function with respect to iteration indicates that the difference between the reference measurements and the estimated measurements is only acceptable when one uses a thin mesh. An average error of 6.5% is achieved for the measurements using a thin or a medium mesh.

### 4.5.1 *Optimisation experiments*

A mesh cascade by computing successive approximations using three levels of mesh quality is now tested.

1. The reference measurements $\overline{M}$ are processed at first by the coarse mesh together with the initial guess of the parameters ($\Lambda_0$). With this mesh one runs up to 7 iterations and obtains a new set of parameters $\Lambda_{coarse}$ which are closer to the reference parameters $\overline{\Lambda}$;

2. Then one uses the medium mesh which receives $\Lambda_{coarse}$ and refines them up to 5 iterations obtaining then $\Lambda_{medium}$;

3. In the end the thin mesh is used as it normally would and runs the estimation method until it converges. The difference is that, instead of using the initial guess $\Lambda_0$, it uses $\Lambda_{medium}$.

To test our technique, new benchmarks were performed with the reference parameters and measurements as in subsections 4.4.1 and 4.4.2. It is expected a faster convergence for the final approximation on thin mesh than the one only computed with the thin mesh as 4.4.3.

10 runs have been carried out with different initial guesses ranging in the 60%-variation intervals with respect to the reference solution. The results are presented in table 11. As

| | **Coarse** | **Medium** | **Thin** |
|---|---|---|---|
| **Cost Function** | $4.86e^{-4}$ | $4.12e^{-5}$ | $1.50e^{-6}$ |
| **Iterations** | 5 | 3 | 3 |

Table 11: Mean value of the cost function

a first stage, one obtains a first approximation using a coarse mesh with 5 iterations in average to stabilise the parameters. Then the second stage with the medium mesh requires 3 iterations. An additional 3 iterations are needed for the last stage with thin mesh in order to reach the target accuracy, *i.e.* where the value of the cost function would be under $5e^{-6}$. Since 1 iteration with a thin mesh corresponds to 3 iterations with a medium mesh and

12 iterations with a coarse mesh, one estimates that the method cut half the running time from 8 iterations with the full thin mesh (table 4) to approximately 4.5 thin-mesh iterations ($5/12 + 3/3 + 3 \approx 4.5$). This represents a speedup of approximately 1.8 times which is really better. Even by cutting the computational time by two, the method still requires almost a full day to complete. Therefore additional improvements with respect to the implementation, memory management and hardware efficiency (vectorisation, multi-core) has to be developed to strongly reduce the computational cost.

## 4.6 DISCUSSION

A new parameter identification method for estimating the geometrical and mechanical parameters of a breast model in function of measurements performed by doctors during consultations was presented and analysed. Such parameters are involved in the numerical model to accurately evaluate the breast deformation after suturing. The simulation then helps the practitioner for planning breast surgeries. The accuracy of the model in representing not only the breast shape and size but the behaviour as well enable the surgeon to evaluate complex situations and adapt procedures. It reduces the probability of making errors and gives a visual expectation of the final shape after the surgery. The method allows doctors to handle virtual breast without needing to wait for long (mainly in terms of waiting periods) and expensive exams like MRI and CT scans.

The results obtained with this method are very positive and show that its potential usage in real life cases. There are some aspects of the method that can be improved, namely the measurements used to estimate the breast parameters which some are not ideal, especially relating to $\mu_{sk}$ and the high time consumption. In terms of the lack of influence that some measurements have in some parameters, the issue can be solved by considering and testing other measurements like the frontal perimeter of the breast or even the lateral contour of the breast (more or less convex) as well as removing the measures that have small to no effect. In terms of the time consumption, despite the improvement obtained for estimations using thin breast meshes, other improvements will be considered for code optimisation: change the data structures, improve the memory management of the data, implementation of parallel versions that can take advantage of vectorisation units or the GPU. Another possibility includes the introduction of machine learning algorithms to train a model capable of providing a set of parameters when given a set of measurements. Besides these issues, there are other aspects that were considered but that are out of the scope of this project which include adding additional parameters to the breast model and finding the least amount of measures in order to obtain a good estimation.

Overall the method's performance is good and the results with synthetic tests are promising. Additional tests with real-life cases are the next step in terms of evaluating the true potential of the proposed method.

# 5

# CODE ARCHITECTURE PROFILING

The modelling and implementation of the numerical surgery as well as the parameter estimation iterative method has been performed successfully and provides good results. Nevertheless, the question of the computational efficiency arises since the simulation stage with just 1000 elements counting cells and faces is computationally very expensive taking approximately 50 seconds for the problem of the breast under gravity. Furthermore, the necessity to improve the code architecture increases even more with the parameter evaluation method which runs several hundred stages per iteration. So, a surgery simulation that can take around 50 seconds to complete leads to a parameter identification process that can take approximately 7 hours. For meshes with more nodes, the required time can reach days of calculations. There are several was to improve the code efficiency which include reconsidering the implementation, use new data structures, use acceleration hardware like the GPU or taking advantage of modern CPU with Single Instruction Multiple Data (SIMD) technology. To this end, the present chapter is dedicated to the assessment of the main bottlenecks and performance issues. This assessment will lead to a more efficient design and robust code architecture.

## 5.1 COMPUTATIONAL ARCHITECTURE

To simulate a breast reduction surgery in both suturing and gravity field stages, a code architecture was developed. For the mesh generation, GMSH software was used while the rest of the code was developed in C++. The mesh is generated from a set of geometrical parameters as described in section 3.2.

### 5.1.1 *Simulation Data Structures*

The data structures used in the architecture represent the mesh elements (vertex/nodes, edges/lines, faces/triangles and cells/tetrahedrons) as well as auxiliary structures used to solve the minimisation of the energy functional of equation 35 in section 3.2.3.

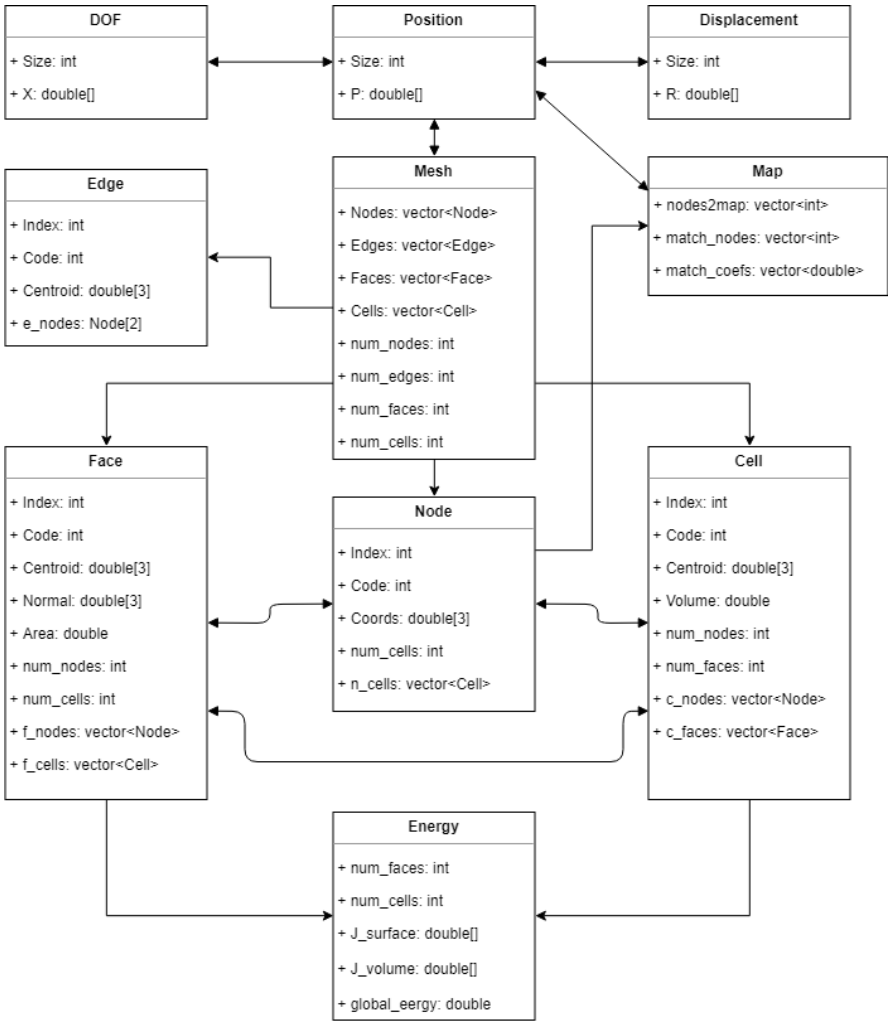



Figure 32: Breast model code architecture data structures.

Figure 32 shows the diagrams of the data structures used to solve the breast reduction surgery problem. The mesh is composed of a list of all the nodes, edges, faces and cells.

For the mesh (figure 13 in section 3.5), each vertex/node has an index to map it inside the faces and cells, a code that identifies the type of node, *i.e.* it determines if a node belongs to a fat or glandular cell or if it belongs to one of the boundaries (for example the incised sections). The nodes also have the information about their coordinates (Coords $P_n = (P_{nx}, P_{ny}, P_{nz})$) and the list of cells to which the node belongs to. Each edge has a code and index and the information about the two nodes that it belongs. Each face has an index and code just like the nodes and cells, but it also has information about its centroid, normal, area and the list of nodes and cells that are linked to it. Besides the index and code, each cell has a centroid, volume and the list of nodes and faces that are linked to it.

For the auxiliary structures, the position structure represents the components of the new configuration ($\Phi_h$) while DOF represents the unknown variables of the problem which are

the degrees of freedom. The degrees of freedom are the nodes coordinates that can be displaced according to the Neo-Hookean model and the boundary conditions ($X_h$ for the suturing problem and $Y_h$ for the breast under gravity problem). The displacement structure contains the variation of strain energy that is used to re-arrange the nodes positions at each iteration of the simulation. The energy structure stores the information about the volume strain energy of each cell and the surface strain energy of each face. The structure also stores the value of strain energy of the current mesh configuration which is the sum of the strain energy over the cells and skin faces. There is also another structure that is used to map the frontal incisions ($\Gamma_L$ and $\Gamma_R$). To this end, the map structure stores in $nodes2map$ the node's indexes of a frontal incision line (left or right). In the *match_nodes* it stores two nodes from the opposite front incision along with the coefficients which are stored in $match_coefs$.

### 5.1.2 *Code system and map g*

There is a system of codes which is used to identify the behaviour of each mesh element. Those behaviour are the following:

- Fixed position (($\Gamma_B$) and $\gamma_D$ )
- Chassaignac Space ($\Gamma_C$)
- Skin ($\Gamma_F$)
- Left Frontal Incision/Suture ($\Gamma_L$)
- Right Frontal Incision/Suture ($\Gamma_R$)
- Back Incision/Suture ($\Gamma_{BL}, \Gamma_{BR}$)
- Backward left suturing line ($\gamma_{BL}$)
- Backward right suturing line ($\gamma_{BR}$)
- Frontal left suturing line ($\gamma_L$)
- Frontal right suturing line ($\gamma_R$)

These behaviours are connected to its respective code which are used inside the methods to know how to evaluate the strain energy values for each mesh element. They are stored inside the mesh file generated with GMSH.

Function $g$ is the fundamental information to describe the suturing process by defining the mapping from $\gamma_R$ onto $\gamma_L$ which physically represents the skin junction performed by the surgeon. To this end, a new method to provide such a mapping is proposed based on the *principle of equivalent relative distance*. More precisely, since $A$ is a fixed point, for

any $P \in \gamma_R$, one computes the distance of arc $\widehat{AP}$ denoted by $|\widehat{AP}|$ and defines the relative distance

$$r(P) = \frac{|\widehat{AP}|}{|\gamma_R|}$$

where $|\gamma_R|$ is the length of the curve $\gamma_R$. Then one takes the associated $g(P)$ such that $|\widehat{Ag(P)}| = r(P)|\gamma_L|$. At the numerical level, for a node $P_n \in \gamma_R$, one seeks the edge $e = [v_{\ell 1}, v_{\ell 2}] \subset \gamma_L$ such that $g(P_n) \in e$ and computes the barycentric coordinates by solving equation

$$g(P_n) = \lambda_1 v_{\ell 1} + \lambda_2 v_{\ell 2}, \quad \lambda_1 + \lambda_2 = 1.$$

The correspondence between vertices and edges and the associated barycentric coefficients are store in the Map data structure.

### 5.1.3 *Simulation Steps*

Computation is divided into two main parts: initialisation and the minimisation loop. In the initialisation process, the mesh structure is built from the mesh file as well as pre-computed data structures used later on. The minimisation main loop search the node configuration that minimises the value of strain energy. In practice, two successive total energies are compared and the loop stops when the difference value is lower than a given threshold $\epsilon$. The different stages within the loop are:

- **pos2dof_global** - method which copies the current coordinates of the mesh ($\Phi_h$) which belong to the unknown variables of the DOF structure ($K_h$, $K \in \{X, Y\}$). In practice, the method copies the coordinates which in the problem are free to move. The coordinates that can be displaced are controlled by the codes associated to each node;
- **dof2pos_local** - method which copies an element from vector $K_h$ to $\Phi_h$ updating its position. This methods leads to a change in the node configuration. The way these changes are done depends on the codes associated to each node. Their specifications are described in sections 3.2.2 and 3.4.1;
- **calc_gradient** - method which evaluates for each degree of freedom in $K_h$, the local variations of strain energy ($\partial_\ell J_h(K)$, with $K \in \{X, Y\}$) and evaluates its impact on the global strain energy value ($J_h$). This method creates a vector which contains the displacement direction ($R$) for each degree of freedom $K_h$;
- **local_gradient** - method that displaces one degree of freedom ($K_h$) using **dof2pos_local** and evaluates the effect of that displacement on the global value of strain energy ($J_h$);
- **calc_energy_global** - method used to evaluate the strain energy of the mesh ($J_h$). It iterates over all faces and all cells and calls **calc_energy_face** and **calc_energy_cell** to

calculate their energy according to equation 35 in section 3.2.3 for the breast suturing problem or equation 45 in section 3.4.2 for the problem with the breast under gravity;

- **line_search** - method which evaluates a displacement factor $\beta$ to apply on the mesh nodes along with the displacement vector ($R$) calculated with the calc_gradient method. This method uses a convergence cycle that temporarily displaces the nodes of the mesh by $\beta R$ and evaluates the strain energy ($J_h$) of those configurations. The cycle then converges when it finds a value for $\beta$ which can minimise the global energy value. This method also calls the **calc_energy_face** and **calc_energy_cell** methods to compute the values of strain-energy;

- **calc_residual** - method which sums the values of $R$ such that $nor = \sum_{r=0}^{N} R_r^2$ where $N$ is the size of the unknown variables in $K_h$;

- **calc_energy_face** - method which calculates the strain energy of a face ($W_{T_k}$ for the skin and $Wc_{T_j}$ for the Chassaignac space);

- **calc_energy_cell** - method which calculates the strain-energy of a cell ($W_{\tau_i}$ for the breast suturing problem and $W_{\tau_i} - Wg_{\tau_i}$ for the problem with the breast under gravity).

A pseudo-code version of the breast surgery simulation can be seen in algorithm 1.

**Algorithm 1:** Simulation process pseudo-code

```
1  mesh τ ← load(file);
2  {position Φ_h; dof K_h; energy J; displacement R} ← init(τ);
3  map g ← mapping(τ);
4  double nor, β;
5  K_h ← pos2dof_global(Φ_h);
6  R ← calc_gradient(Φ_h, K_h, J);
7  β ← line_search(Φ_h, K_h, J, R);
8  nor ← calc_residual(R);
9  while nor > ε do
10 |  K_h ← pos2dof_global(Φ_h);
11 |  R ← calc_gradient(Φ_h, K_h, J);
12 |  β ← line_search(Φ_h, K_h, J, R);
13 |  nor ← calc_residual(R);
14 end
```

The initialisation stage is represented in the pseudo-code from line 1 to line 8. In line 1 the mesh is loaded and the data structures related to the mesh are filled with the links between the nodes, edges, faces and cells as well as the codes corresponding to the problem. In the case of the breast suturing problem, the file contains the geometry obtained with the

GMSH software and for the problem with the breast under gravity, the geometry is the one obtained after the breast suturing process ended. Line 2 presents the pre-computation step where values like the faces area, the cells volume as well as their centroid are calculated and stored. Besides that, the mesh nodes current position are also copied to $\Phi_h$ and the other structures ($K_h$, $J$ and $R$) are initialised with zeros. Line 3 represents the elaboration of the mapping between the right and left side for the interfaces suturing as well as their suturing lines. In line 5 the unknown variables of the problem are evaluated using the codes associated to the mesh elements and so one obtains the vector with the degrees of freedom for the simulation problem ($X_h$ for the suturing and $Y_h$ for the gravity). Line 6 corresponds to a first guess of the solution, *i.e.* the degrees of freedom of vector $K_h$ are displaced using **dof2pos_local** which modifies the node configuration. This step leads the frontal incision interface nodes to move towards the suturing plane ($\theta$) as well as the rear interfaces to join the zone attached to the torso ($y = 0$). The first candidate displacement vector is filled thanks to the local strain-energy variations per degree of freedom (gradient method). Line 7 calculates the best displacement factor following the gradient while line 8 calculates the current energy residual value.

From line 9 to 14 it is represented the energy minimisation cycle. In line 9 the simulation evaluates if the local variation of energy is smaller than a threshold value $\epsilon$ which is very close to zero ($< 1e^{-6}$). If the value is not small enough then it proceeds to lines 10-13 which are an iteration of lines 5-8. This means that the nodes are re-arranging themselves until the energy variation is below the threshold, meaning that the final configuration is obtained.

*Breast under gravity*

The evaluation of the breast configuration under gravity works in a similar fashion to the workflow presented above. However, in this cases the mapping is not necessary and so, a workflow diagram of the breast under gravity simulation is presented in figure 33.

The simulation starts by loading the mesh $\tau$ from a file which is then used by the method **init** to fill the data structures with the information about the mesh current configuration $\Phi_h$ (node, face and cell data), the Energy structure $J$ (values for the volume and area) and initialising with zeros the unknown variables $X_h$ and the displacement vector $R$. Then $X_h$ is updated with the values in $\Phi_h$ which, depending on the code, can be displaced using the strain energy functionals. Then, a first evaluation of the strain energy is done by calculating the gradient which returns the vector of displacement ($R$). Afterwards, all these structures are used to obtain the optimal displacement factor $\beta$ that will then be used together with $R$ to change the mesh configuration. This leads to the evaluation of the residual which is the squared sum of the values of $R$. The simulation enters in a state where if the value of the residual ($nor$) is larger than a given $\varepsilon$, then it continues with the **pos2dof_global**,

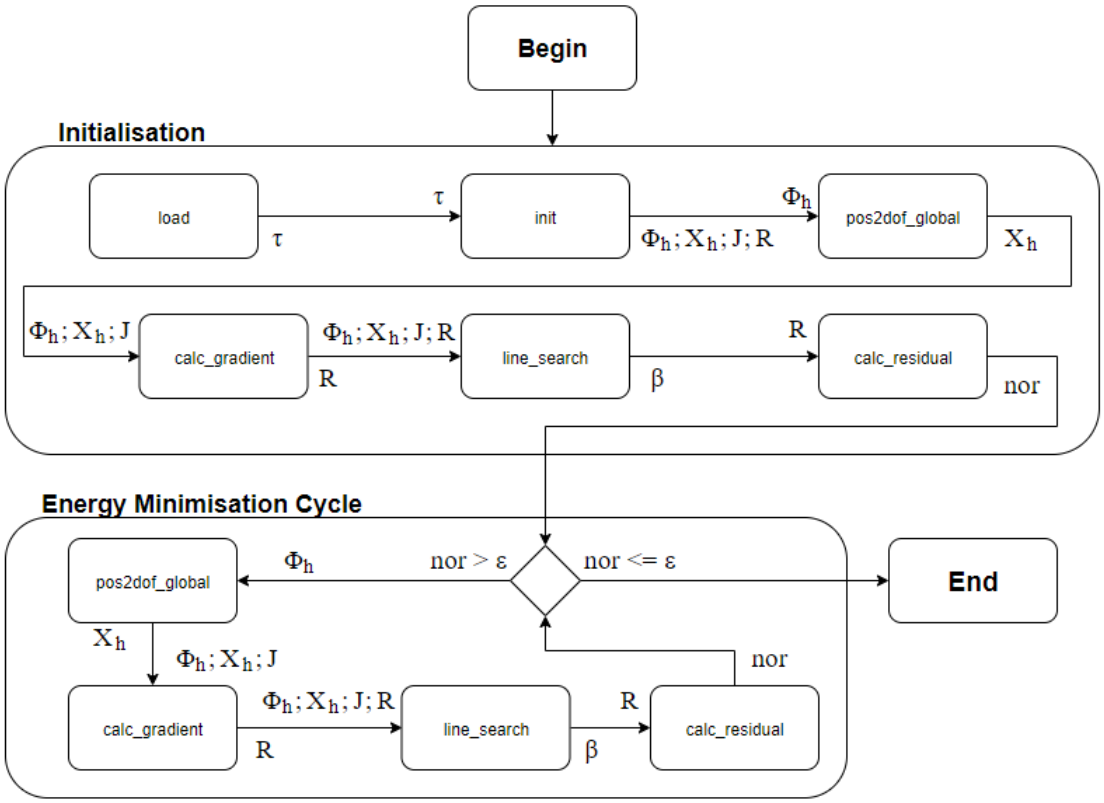



Figure 33: Flowchart of the simulation

**calc_gradient**, **line_search** and **calc_residual** methods until the value of *nor* becomes smaller than $\varepsilon$.

## 5.2 ESTIMATION METHOD ARCHITECTURE

The section is dedicated to the effective implementation of the breast model parameters estimation method, using the code architecture presented in section 5.1.

### 5.2.1 *Estimation Data structures*

The data structures are quite similar to the ones use in the knitting code although some of code blocks are not used. For instance, the map structure that links two interfaces is not required in parameters identification. On the other hand, one additional structure that holds the estimation parameter variables (figure 34) is needed.

The estimator structure holds the input parameters such as the initial guess (*input_p*), the reference measurements (*input_m*) that the optimisation method will attain by computing successive candidate parameters. For each iteration, the parameters are modified and

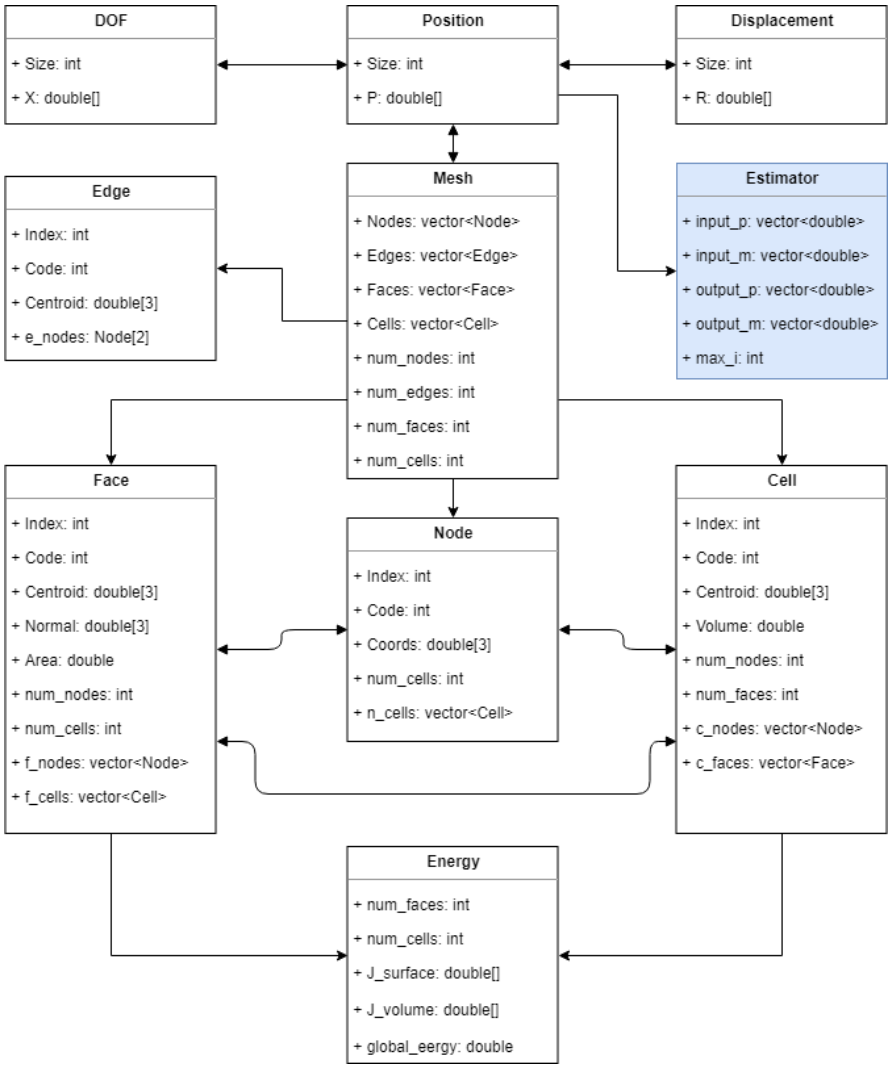



Figure 34: Diagram of the data structures used to perform the parameter estimation including a new structure (estimator in blue).

stored in $output_p$ while the associated computed measurements (from the mesh current configuration in the Position structure) are stored in $output_m$. At the end of the iterative process when the solution converges up to a given error, the final parameters estimation is stored in ($output_p$). Notice that a maximum number of iterations $max_i$ is prescribed in case that no convergence is achieved. To compute the vector of the unknown variables ($X_h$) at the node, specific label are attributed together with the boundary conditions:

- $1 \rightarrow$ Fixed position ($\gamma_D$)
- $2 \rightarrow$ Chassaignac Space ($\Gamma_B$)
- $3 \rightarrow$ Skin ($\Gamma_F$)

The nodes and edges on the infra-mammary fold are fixed, while the elements in the Chassaignac space can only move along the torso (not forward). The labels are also used, to identify which equation hat to be used to calculate the values of strain energy (equation 49 in section 3.4.2 for the elements in $\Gamma_B$ and equation 29 in section 3.1.2 for the elements in $\Gamma_F$).

### 5.2.2 *Computational sub-stages*

The estimation of the parameters can be divided into the initialisation stage and the optimisation loop. Inside the loop, a subroutine is dedicated to calculate the partial derivatives and an other one employs the Gauss-Newton method. To main sub-stages in the estimation process are:

- **fit_parameter** - method which uses the geometrical parameters $R$ and $H$ to generate the stress-free configuration ($\Omega_g$);
- **simulation** - method which takes the stress free breast and a gravity field (according to a specific patient position) and solves the minimisation of the energy functional (equation 51);
- **simulate_3pos** - method which uses the method **simulation** for the three patient positions;
- **get_outputs** - method which obtains the measurements of a breast configuration;
- **get_derivative** - method which calculates the derivative of the measurements by varying a parameter (equation 61);
- **solve_partial** - method which solves evaluates the partial derivatives of the measurements by varying the value of one parameter at a time for all parameters. The return value is a vector which possess the variation of measurements which is then used in the Gauss-Newton method;
- **solve_newton** - method which receives the results of the partial derivatives and calculates the optimal variation of the parameters seeking to minimise the difference between the reference measurements ($\overline{M}$) and the current measurements ($M_h(\Lambda)$);
- **evaluate_error** - method which evaluates the value of the cost function (equation 62);
- **calc_step** - method which evaluates the change of the values of the parameters and chose a value of step that most fit the solution.

The estimation process starts with an initial guess $\Lambda_0$ and with the reference measurements $\overline{M}$. A pseudo-code version of the parameter estimation method can be seen in algorithm 2.

**Algorithm 2:** Parameter estimation method pseudo-code

```
1  mesh τ ← load(file);
2  estimator E ← input(Λ_0, M̄);
3  position[] (Φ_h) ← fit_parameter(E,τ);
4  Φ_h ← simulate_3pos(Φ_h, E);
5  E ← get_outputs(τ);
6  double E_m ← 1e6;
7  double E_e ← evaluate_error(E);
8  int i ← 0;
9  double step ← 2.0e^-1;
10 while E_m > ε_m and E_e > ε_e and i < E.max_i do
11 |  ΔM ← solve_partial(E, Φ_h);
12 |  ΔΛ ← solve_newton(ΔM);
13 |  step ← calc_step;
14 |  E.output_p ← E.output_p + ((ΔΛ) * step);
15 |  Φ_h ← fit_parameter(E,τ);
16 |  Φ_h ← simulate_3pos(Φ_h, E);
17 |  E ← get_outputs(τ);
18 |  E_m, E_e ← evaluate_error;
19 end
```

The initialisation stage goes from line 1 to line 9. The first line shows the loading of the mesh from a file. In line 2 the estimator receives the values of input of the problem, namely the parameters initial guess ($\Lambda_0$) and the reference measurements ($\overline{M}$). Line 3 shows the function **fit_parameter** which takes the mesh nodes positions, and changes it in order to fit the new geometrical parameters that come from the initial guess. The result is stored in the three element array that memorises the mesh current configuration for all three positions that are simulated (patient standing up, bend 45 degrees forward and on all fours). The method sets the same value in each element of the array. Line 4 solves equation 51 for each of the patients position and stores the results in the three element array $\Phi_h$. Line 5 shows the initial evaluation of the differences between the reference measurements and the measurements obtained with an initial parameter guess. Lines 6-9 show only the initialisation of variables like the energy variation $E_m$, the measures squared difference $E_e$, the iteration number and the amount of step which by default takes the value $2e^{-1}$ (after testing with different values this value was found to provide a better starting point).

The optimisation loop is described from line 10 to line 18. While the stop conditions in line 10 are not satisfied, the loop starts by evaluating the partial derivatives of the measurements variation by varying the values of one parameter at a time for all 6 parameters

in *E.output_p* (line 11). In line 12 the Gauss-Newton method is used to obtain the best variation of the parameters. In line 13 the new step is calculated and is used in line 14 to obtain and the new set of parameters and update the estimator data structure. With the new parameters, the values of the new positions are once again update according to the mechanical parameters (line 15) and are followed by the solution of the minimisation problem with the breast model mechanical parameters (line 16). In line 17 the new measurements are assessed and in line 18 the difference between the reference measurements and the current measurements is evaluated returning the values of the variation of error and the squared error.

## 5.3 PROFILING ANALYSIS

The previously implemented solution in C++ presents some issues in terms of performance. These issues affect the overall time consumption of the simulation process even for simulations with meshes with few nodes. The objective of this type of procedures is to identify and understand the possible bottlenecks of the code (Chang et al. (1991); Chen et al. (1993); Hennessy and Patterson (2017); Melo and Hat (2010); Bryant and O'Hallaron (2016); Khan (2011)). For that purpose a series of profiling tests were conducted using a laptop with the following specifications:

- Memory : 4GB 1333 MHz DDR3
- Processor : intel i5 2450m 2.3GHz
- Cache : 3MB L3, 256KB L2, 2*16 load + 16 store L1 with 32KB
- Micro-architecture : sandy bridge

For a matter of consistency, the results presented in the next sections consist in the mean values of groups of 5 tests/runs were the number of iterations run by the **simulation** method (main loop) is fixed on 1500.

When evaluating the weights of the methods used in the estimation of the breast parameters, GProf (Graham et al. (1982)) showed that 99% of the time was being spent solving the minimisation of the energy functional, *i.e.* the time was being spent with the **simulation** method which was in turn being called by the **simulate_3pos** mentioned in section 5.2.2. Because of this, the focus of the profiling was in this method (**simulation**).

### 5.3.1 *Time Consumption and Instructions per Cycle*

An initial test was made to better understand which routines were consuming more CPU time. Figure 35 presents the first call graph using GProf (Graham et al. (1982)) to profile

the **simulation** method. This graph shows that the solution is spending more time in the

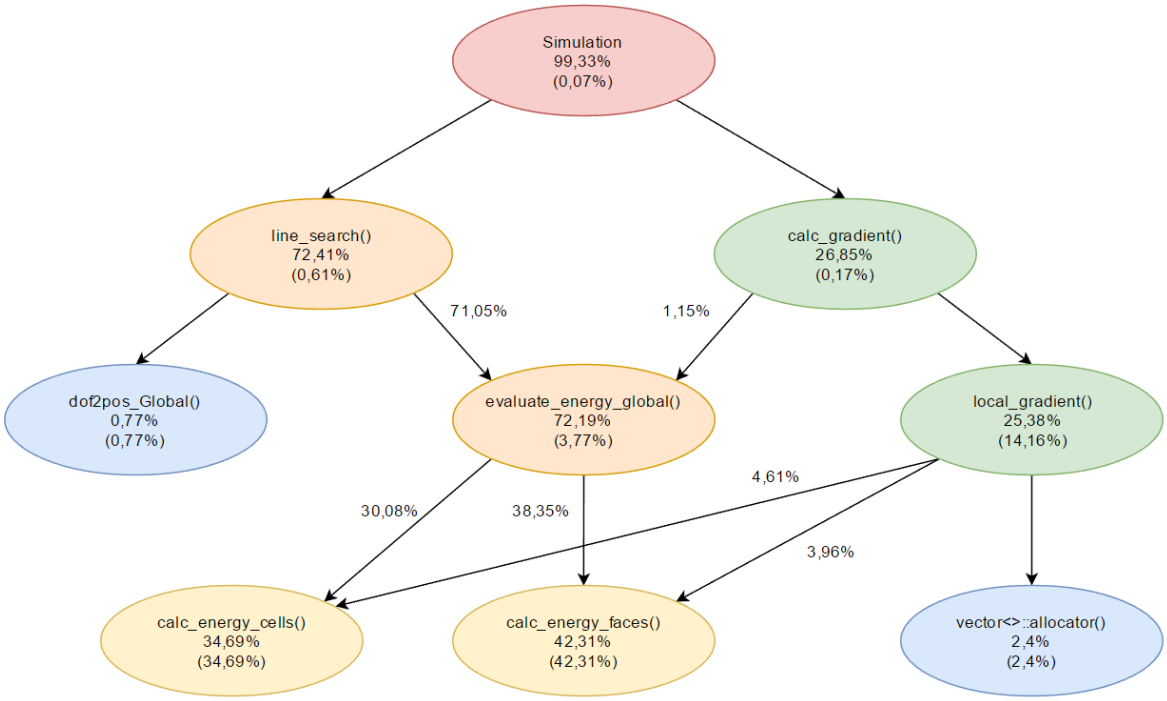



Figure 35: Call Graph of the simulation with percentage of, total and self, time consumed (clock cycles spend on each method)

**line_search** method than any other method ($\approx$ 72%). However, this time is spent on child functions like the method that evaluates the energy of the breast (**evaluate_energy_global** with 72%) which in turn call the methods that evaluate the energy for the cells (**calc_energy_cells**) and for the faces (**calc_energy_faces**) with 34% and 42% respectively. Other method that has a significant impact is the **local_gradient** method with 25%. This means that the **line_search**, **calc_gradient** and the energy evaluation methods for the faces and cells need improvements. **line_search** and **calc_gradient** are methods with low self time consumption, while, on the other hand, the **calc_energy_cells** and **calc_energy_faces** (and even the **local_gradient** method) have a great portion of self time. The **pos2dof_global** method which was mentioned above, does not show in this graph which means its time consumption is inferior to 1%.

To better understand the efficiency of these methods, linux's perf (Melo and Hat (2010)) was used to evaluate their Instructions per Cycle (IPC). The number of IPC, indicates the efficiency of the method, *i.e.* higher values of IPC indicate that the processor can perform more instructions per unit of time (in this case, clock cycles) and vise-versa (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). Therefore, low values for this indicator suggest a need for improvements.

| IPC | Coarse (4,172MB) | | Medium (8,644MB) | | Thin (17,068MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 1.05 | 1.76 | 1.10 | 1.83 | 1.10 | 0.81 |
| **calc_energy_global** | 0.95 | 2.06 | 0.99 | 1.86 | 0.99 | 1.94 |
| **line_search** | 0.97 | 1.84 | 1.01 | 1.16 | 1.00 | 0.78 |
| **calc_gradient** | 1.31 | 1.32 | 1.38 | 1.22 | 1.39 | 1.46 |
| **local_gradient** | 1.25 | 2.37 | 1.26 | 1.71 | 1.30 | 1.65 |
| **calc_energy_faces** | 0.57 | 0.57 | 0.57 | 0.57 | 0.56 | 0.56 |
| **calc_energy_cells** | 1.24 | 1.24 | 1.20 | 1.20 | 1.20 | 1.20 |

Table 12: IPC of the simulation methods with different size meshes

Table 12 shows that the methods IPC do not vary much with the granularity of the mesh used in the simulation. Almost all methods show an IPC close or higher than 1, except for the **calc_energy_faces** with values of 0.56 and 0.57. This shows that this particular method is far from optimised. In the code, this method performs several division operations, which are very expensive for any CPU. Nevertheless, these values, by themselves, cannot explain why the bottleneck is there (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). So, an analysis to memory access patterns was also performed.

### 5.3.2 *Memory and Cache*

It was necessary to know if the bottlenecks were due to lack of bandwidth, lack of processing power or maybe both. So, the linux's **perf** was also used to analyse the memory usage and cache access (Melo and Hat (2010)). On top of this, the amount of memory consumption was also measured. The following tables (13, 14 and 15) show the results of perf profiling. Table 13 shows the miss % when the simulation methods try to access L3 cache as

| L3 miss % | Coarse (4,172MB) | | Medium (8,644MB) | | Thin (17,068MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 0.01 | 0.06 | 26.41 | 28.92 | 67.72 | 30.36 |
| **calc_energy_global** | 0.01 | 0.01 | 32.57 | 70.21 | 84.87 | 95.23 |
| **line_search** | 0.01 | 0.01 | 29.55 | 15.01 | 76.88 | 27.81 |
| **calc_gradient** | 0.04 | 0.01 | 14.10 | 0.93 | 26.00 | 3.17 |
| **local_gradient** | 0.05 | 0.01 | 13.40 | 9.05 | 23.29 | 12.80 |
| **calc_energy_faces** | 0.02 | 0.02 | 35.00 | 35.00 | 78.17 | 78.17 |
| **calc_energy_cells** | 0.01 | 0.01 | 9.36 | 9.36 | 32.21 | 32.21 |

Table 13: L3 miss percentage of 3 different sized meshes

well as the memory used during the simulation. As it can be seen, the size of memory used

increases with the number of nodes of the mesh (thinner meshes have more nodes). This information helps understanding why the amount of cache misses also increases with the higher number of mesh nodes. As mentioned above, the size of the L3 cache used for the tests is 3MB, and the size of the data structures for thinner meshes are bigger than all L3 cache. That means there is a higher probability for a miss. The high miss % for a medium mesh, and specially, for a thin mesh suggests that the data being used during the calculations might be replacing cache lines on a constant basis, therefore, hurting the simulation performance (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). One aspect worth mentioning is the big difference for thinner meshes between the two methods which evaluate energy (faces and cells). While the method which evaluates the strain-energy of cells have a substantial miss rate (9.36% and 32.21%) the method to calculate the faces strain-energy has an enormous miss rate, specially for the thin mesh with 78.17%. This means, the structures used for this method might have very poor locality and so, the data that is being requested for the method requires entire L3 cache lines to be constantly overwritten which is inefficient (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). Table

| **L1 Load miss %** | **Coarse** | | **Medium** | | **Thin** | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 2.78 | 31.53 | 3.27 | 53.00 | 3.68 | 21.35 |
| **calc_energy_global** | 3.35 | 0.91 | 3.88 | 1.04 | 4.52 | 2.58 |
| **line_search** | 3.52 | 40.89 | 4.07 | 40.18 | 4.65 | 34.90 |
| **calc_gradient** | 1.05 | 2.91 | 1.41 | 2.94 | 1.53 | 3.28 |
| **local_gradient** | 0.95 | 0.55 | 1.32 | 1.05 | 1.38 | 1.13 |
| **calc_energy_faces** | 4.12 | 4.12 | 4.09 | 4.09 | 4.19 | 4.19 |
| **calc_energy_cells** | 3.23 | 3.23 | 4.93 | 4.93 | 6.43 | 6.43 |

Table 14: L1 Load miss percentage of 3 different sized meshes

14 presents the miss % for the L1 cache loads. The number of L1 load accesses is much higher than the L3 cache accesses. In fact, the number of L1 loads is approximately a third of the number of instructions of the simulation. Overall, the number of L1 load misses don't follow a particular pattern like the one seen in the L3 miss % case, where it increases with the number of mesh nodes. However, it is possible to see that for thinner meshes, the methods which have the highest self time consumption (**calc_energy_cells** and **calc_energy_faces**) don't follow the tendency of L3 miss. In this case, the method for cell energy evaluation has higher miss % than the face method (4.93% and 6.43% opposed to 4.09% and 4.19% respectively). In terms of store cache accesses, the number is approximately half when comparing to L1 load accesses. Table 15 shows that the miss % for data storage in the L1 cache is close or inferior to 1%, which means that there are no major bottleneck for the simulation when storing data.

| L1 Store miss % | Coarse | | Medium | | Thin | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 0.41 | 4.61 | 0.36 | 11.32 | 0.34 | 16.63 |
| **calc_energy_global** | 0.56 | 0.10 | 0.40 | 0.14 | 0.36 | 0.30 |
| **line_search** | 0.60 | 6.83 | 0.51 | 13.54 | 0.47 | 12.53 |
| **calc_gradient** | 0.07 | 0.42 | 0.08 | 0.78 | 0.07 | 1.50 |
| **local_gradient** | 0.05 | 0.06 | 0.06 | 0.08 | 0.06 | 0.08 |
| **calc_energy_faces** | 1.63 | 1.63 | 1.42 | 1.42 | 1.33 | 1.33 |
| **calc_energy_cells** | 1.36 | 1.36 | 0.72 | 0.72 | 0.71 | 0.71 |

Table 15: L1 Store miss percentage of 3 different sized meshes

The misses on L3 cache leads to memory accesses which are expensive for the **simulation** method overall performance. Although every significant method was analysed, 2 of those methods possess an higher self percentage of time consumption (namely **calc_energy_cells** and **calc_energy_faces**). So the main focus on this cache and memory analysis was on these particular methods. As mentioned above, when comparing the miss rate values between these methods, **calc_energy_cells** have a much lower L3 miss rate and a slightly higher L1 load miss rate than **calc_energy_faces**. The profiling analysis showed that **calc_energy_cells** have a much higher number of L1 load accesses than **calc_energy_faces** while **calc_energy_faces** has more L3 accesses than **calc_energy_cells**. This suggests that, for **calc_energy_cells**, the size of the cell structures do not fit in L1 or might have redundant information that is pulled but that is not used and therefore occupies cache lines that could be used more efficiently, hence an higher miss % on L1 cache. On the other hand, the lower miss rate on L3 suggests a better organisation of the entire cell's data, i.e., the cells arrangement might be more contiguous and therefore enjoy better locality which leads to reduced numbers of misses on L3. **calc_energy_faces** face structure might have less redundant data and therefore the number of L1 cache misses might reflect that, but the arrangement of the data is clearly poorer and result on high miss rate.

### 5.3.3 *Arithmetic efficiency and Intensity*

As mentioned above, the nature of a bottleneck relies on lack of power to process calculations or lack of bandwidth to load necessary data (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). So, in order to better understand the nature of these bottlenecks a peek to the assembly code was made. This closer look allows to see the instructions called by the processor during these methods and evaluate what kind of calculations and loads are being performed. Table 16 shows the number of times and number of cycles the float operating instructions (subsd, mulsd, addsd, sqrtsd and divsd) and memory calls (movsd) were performed. The **calc_energy_faces** method has an if which performs different calcula-

| *floating ops* | addsd | subsd | mulsd | divsd | sqrtsd |
|---|---|---|---|---|---|
| **Faces (chass ; skin)** | 8 ; 20 | 3 ; 23 | 5 ; 51 | 3 ; 13 | 0 ; 6 |
| **Cells** | 41 | 16 | 58 | 0 | 0 |

| memory | movsd |
|---|---|
| **Faces (chass ; skin)** | 14 ; 22 |
| **Cells** | 26 |

Table 16: Assembly instructions of the face and cell evaluate energy methods. For the face method, the table divides the values by the Chassaignac and skin calculations while the cell method only has one way to calculate the strain-energy. The table is divided in floating operation instructions (addsd, subsd, mulsd, divsd and sqrtsd) and data access instructions (movsd)

tions depending on the code of the face, which can be Chassaignac (face in the zone of the torso which is mentioned in the table 16 by "chass") and skin (face associated to the breast's skin). The instructions show for the case of the **calc_energy_cells** method a high number of floating point operations (115) while having less data access operations (26), just like the **calc_energy_faces** method for the skin face with 113 floating point operations and 22 data accesses. On the other hand, the **calc_energy_faces** method for the Chassaignac faces have only 19 floating point operations for the 14 data accesses.

To know if the method is compute or memory bound one has to evaluate the ratio between floating operations and memory loads (floating operations per byte). As the methods operate on doubles, that means each movsd loads a 8 byte double, so, one gets the following results (table 17): These ratios show that for each byte there is a low amount of floating

| *Method* | Adds and Muls (AM) | Divs and Sqrts (DS) | Loads (L) | Arithmetic Intensity (AM+DS)/(L*8) |
|---|---|---|---|---|
| **Cells** | 115 | 0 | 26 | 0.55 |
| **Chassaignac** | 16 | 3 | 14 | 0.17 |
| **Skin** | 94 | 19 | 22 | 0.64 |

Table 17: Arithmetic intensity of the energy evaluation methods (cells, Chassaignac and skin faces). Ratio between the number floating point operations (adds, muls, divs and sqrts) and bytes loaded (number of loads times size of double )

point operations. Therefore, it suggests that the memory bandwidth isn't enough to feed the calculations performed in the methods ( specially the energy evaluation method of the Chassaignac faces). However, this simple analysis does not evaluate a key aspect of the methods that is the time the processor spends between loading data and performing calculations.

### 5.3.4 *Arithmetic efficiency and Bandwidth*

According to Intel (2016), the sandy bridge micro-architecture has 1 units for multiplications (muls) and additions (adds) and 1 unit for divisions (divs) and square roots (sqrts). However, due to its particular specifications, the sandy bridge has only 2 ports to schedule the instructions, and in one of these ports both muls, divs and sqrts are scheduled while on the other port only the adds are scheduled. This means that when a division is being processed, no multiplications can be done. On the other hand as the adds are on an independent port they can be done at the same time as divisions and multiplications. Nevertheless, the modern processors with the skylake micro-architecture work with independent units which allows to perform muls and adds simultaneously (Intel (2016)). With this in mind the study of the time spent takes into account the modern processes. In sum, both muls, adds and divisions can be done simultaneously but with a maximum of two at a time. The same Intel manual states that the processor has the capacity of performing 2 loads and 1 write of 16 bytes at the same time.

In terms of time consumption, i.e., the instructions throughput, adds and muls have only 1 cycle of throughput, while divs and sqrts have approximately 20 cycles. For the load instructions, they can have latency of 1 and 2 cycles, so, one considers its throughput to be 1.5. The alternative arithmetic intensity measurement by time spent gives the following results (table 18): In addition to these values one tested the bandwidth of the machine in

| *Method* | **Adds and Muls (AM)** | **Divs and Sqrts (DS)** | **Loads (L)** | **Arithmetic Intensity** $\frac{max(AM,DS)}{L}$ |
|---|---|---|---|---|
| **Cells** | 57.5 | 0 | 39 | 1.49 |
| **Chassaignac** | 8 | 60 | 21 | 3.10 |
| **Skin** | 47 | 380 | 33 | 13.06 |

Table 18: Arithmetic intensity of the energy evaluation methods (cells, Chassaignac and skin faces) based on number of cycles. Ratio between the number of cycles spent on performing calculations (AM and DS) and loads (L).

order to know how much data can be sent to the processor and obtained the following results: These values of bandwidth can show the maximum number of doubles that can be

| *Bandwidth (GB/s)* | **32KB** | **256KB** | **3MB** | **500MB** |
|---|---|---|---|---|
| **Sequential 8B** | 24.3 | 16.8 | 11.5 | 9.1 |
| **Random 8B** | 16.8 | 11 | 5.2 | 2.5 |
| **Sequential 16B** | 47.6 | 23.9 | 15.5 | 10.6 |

Table 19: Bandwidth measured for different sizes (32KB, 256KB, 3MB and 500MB), order of access (sequential or random) and bytes loaded at a time (8B and 16B)

loaded in a certain time. For the test it was chosen memory sizes that coincide with the sizes of the L1, L2 and L3 cache. So, the amount of data that is being loaded varies with the amount of doubles that can be feed to the processor. As the structure sizes used in the simulations round the 3MB and the access is most of it sequential, lets consider 11.5 GB/s as the memory bandwidth. This bandwidth means that, for the 2.3GHz processor used, for each second 1.437.500.000 doubles per second which in cycles turns into 0.625. So, one can build a table with the maximum amount of doubles that can be loaded during the time of calculations (table 20) These values contradict the ones presented before and suggest that

| *Methods* | Calculating cycles | Loading operations | Max Operations |
|---|---|---|---|
| Cells | 57.5 | 26 | 36 |
| Chassaignac | 60 | 14 | 37 |
| Skin | 380 | 22 | 237 |

Table 20: Comparison between the number of doubles loaded in each methods vs the maximum number that could be loaded if the bandwidth was fully utilised.

these methods are not memory bound but actually compute bound (specially the method to evaluate the skin energy). Either way, both tests point out the performance bottlenecks which are the high number of divisions and square roots.

### 5.3.5 *Profiling Analysis*

The profiling tests that were performed over the simulation with different mesh sizes have shown that the main bottlenecks are located in the calc_energy_cells and calc_energy_faces methods. So any improvement on these methods may lead to high gains in terms of performance. The tests suggest that the calc_energy_cells and calc_energy_faces methods are compute bound and also memory bound. So, the strategy for calculating the energy of the mesh elements could pass by packing the data and operations and take advantage of the vectorisation units (SSE, AVX, ...), while also by restructuring the data structures (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). Another problem was the high L3 miss percentage, which can be solved by improving the data locality which in turn can be made by restructuring the code structures.

Another way of improving the code performance would be to take advantage of the multiple cores on a single machine to perform the several operations (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). The parallelism and vectorisation of data would be able to reduce the amount of time spent on performing multiplications and divisions, i.e., if a single division takes 20 cycles to perform, and the method has 12 divisions, if all divisions are done in sequence, they will take 240 cycles to complete, while if this number could be

spread by an AVX unit or by 4 different cores, the time could be reduced to only 60 cycles to perform all 12 divisions.

One another aspect worth mentioning is the improvement on these methods may not only be done by changing the code structures, but also by changing the methods that call them (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). On this particular note, the line_search method, which uses a convergence algorithm, was tested to evaluate the number of iterations necessary to obtain a good value for the displacement factor. The tests showed that the number of iterations were around 40-50. This high number of iterations is far from optimal, and therefore, a more convergent algorithm should be implemented.

In terms of improving the data locality and accesses, one aspect that was verified during this profiling stage was the way calc_energy_faces was working inside the methods that call it, namely, local_gradient and evaluation_energy_global. The majority of these calls resulted in choosing which method to calculate the value of energy depending if the face was part of the skin, or the torso/Chassaignac space. Besides these faces there are others with no code attached because their contributions are already calculated through the cells. In fact these are the majority of the faces. This result in thousands of calls for this method that only perform an if statement to then simply return. This aspect can also be optimised by changing the data structures so that the faces are organised in terms of calculations, i.e., the faces of the Chassaignac space would be on one list, the faces of the skin would be on another list and the rest would be in a third list.

With all these considerations the following strategy for the new code architecture was devised:

- The data structures will be reorganised to a Structure of Arrays (SoA) or Array of Structures (AoS) in order to improve the data locality (Faria et al. (2013));
- The line_search method will be changed to use a more convergent algorithm in order to reduce the number of calls to heavy functions and methods (Graham et al. (1982));
- The strain-energy evaluation of the cells will be changed to take advantage of the SIMD units;
- The new structures and methods will be able to take advantage of parallelism and therefore reduce the time by performing more calculations simultaneously (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)).

After these changes to the code architecture, a new profiling set of tests will be made and they will evaluate what type of improvements were accomplished with this change.

## 5.4 NEW CODE ARCHITECTURE

After profiling the code and identifying the bottlenecks, the next step is to perform the necessary changes in order to improve the code's overall performance. Knowing that the main performance bottleneck concerns memory access, the first upgrade was made in the data structures. Next, different algorithms were implemented in order to improve some method's efficiency (*e.g.* the **line_search** method). After that, an attempt to improve the code's performance is made by taking advantage of its parallel nature as well as the modern hardware capabilities to perform several tasks simultaneously.

At every step of this optimisation phase, new profiling tests were performed. Most of these tests were made in the same machine as the previous ones. New machines were also used to establish comparisons and to evaluate the new code's efficiency with different processors and computational environments (used in a multi-thread context).

### 5.4.1 *Code Structure analysis*

The profiling of the breast simulation code has shown memory issues. A high L3 miss percentage and a high rate of loading instructions suggests a data structure with poor locality and with too much information. Therefore, the new data structures were designed to solve these problems. To improve data locality the information was arranged in a SoA type (Faria et al. (2013)). To reduce the size of the structures, some redundant data was removed. The new data structures are presented in figure 36 with the new elements (comparing to the the previous structures of figure 34) marked in blue.

The new Mesh structure is a SoA which holds the current value of energy of the mesh during the simulation and the arrays with the information about the nodes, faces and cells. The simulation methods use this structure to access the necessary data to perform the calculations. The main reason behind the choice to arrange the data in a SoA instead of AoS, was the usage of the elements of the structure, *i.e.*, when calculating the strain energy values, single elements from many structures are used (volume of a cell, the current positions of the nodes) and so, according to Faria et al. (2013), this behaviour benefits the usage of SoA.

The nodes, faces and cells suffered a change in terms of organisation. The main change concerns the way the relevant elements were organised. By relevant elements, one means, the faces and cells which were used to compute the strain energy value during the simulation (when evaluating the strain energy of faces, the ones which didn't belong to the Chassaignac space or to the skin were not doing anything and so were spending time with if statements unnecessarily). The strain energy of elements with different codes is evaluated differently, in this particular case, the energy of the faces of the skin and of the faces

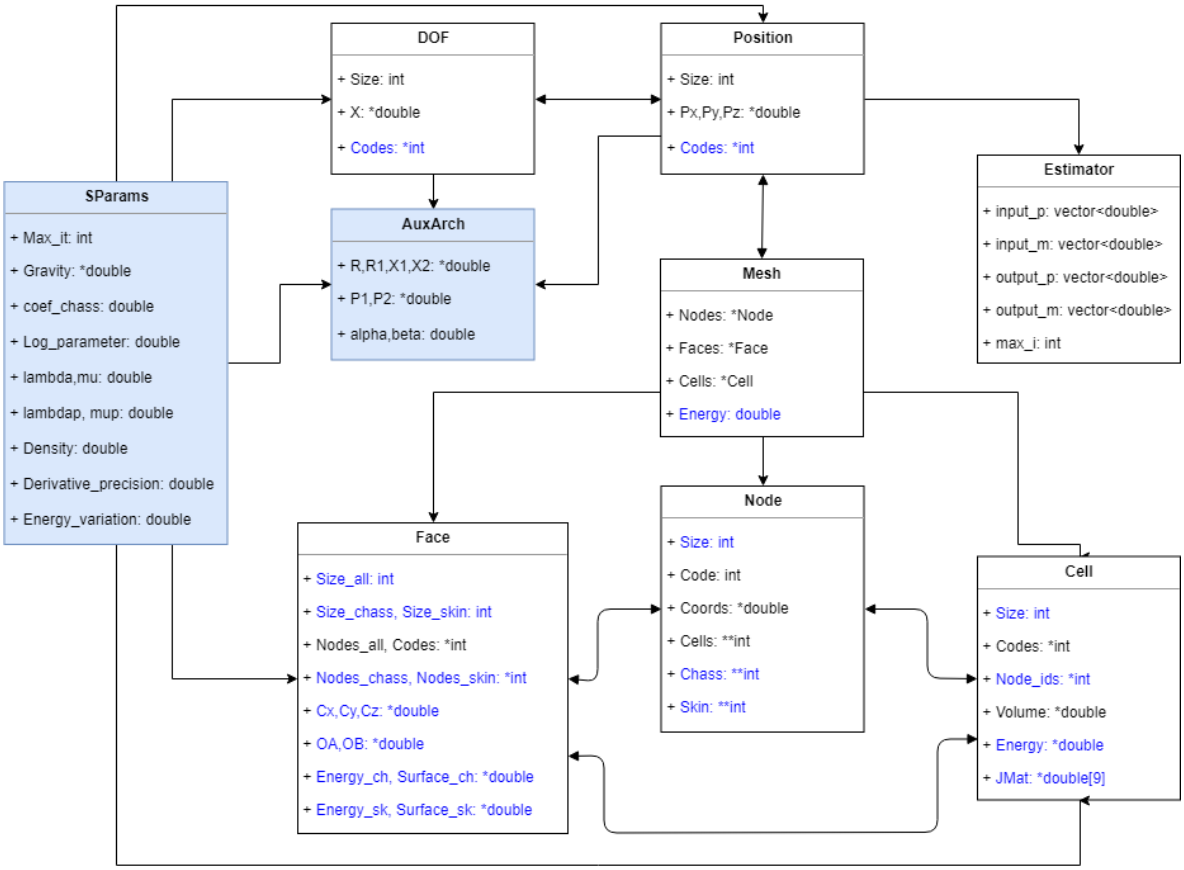



Figure 36: New code architecture data structures (new elements in blue).

in the Chassaignac space are computed with different numerical methods. So, to improve the cache accesses, the data is organised in arrays for each relevant type of element. Due to its redundancy, the arrays that linked both faces and the cells (figure 32 f_cells and c_faces) were removed.

The Node structure, stores the number of nodes of a mesh, the codes of those nodes, the coordinates represented by x, y and z. It also stores, in an arrays of arrays, each cell, Chassaignac face and skin face belonging to each node. In these structures, the first element of the first array (column [k][o]) indicates the number of those elements.

The Face structure, stores the data of each face, namely the size, codes and nodes. Then it stores the necessary data to compute dividing it by relevant elements. An array to hold the nodes of the Chassaignac faces (Nodes_chass) and another for the skin faces (Nodes_skin). It also stores for each of those element types, the energy, area (surface) and auxiliary arrays (Cx, Cy and Cz for Chassaignac faces and OA and OB for skin faces). As mentioned before, each of the relevant elements strain energy is calculated differently, and so, each one of them has a different function to perform the energy evaluation (**calc_energy_chass()** and

**calc_energy_skin()**). The Cell structure holds every information about the nodes of each cell, the codes, energy, volume and also other auxiliary computational elements like JMat.

However the Mesh and its core elements are not sufficient to perform the simulations and like before, auxiliary structures were also implemented. The Position and DOF structures are very similar to the previous ones: the Position structure stores the current positions of the nodes while the simulation is occurring and the DOF structure holds the degrees of freedom of the system. The difference in these structures is the inclusion of the codes of the nodes for energy calculations purposes.

On the previous version, there was a structure Energy which held the energy data of the simulation. Now that data is spread on other structures. Moreover, two new structures with different functions: the AuxArch structure which has arrays used in the simulation to store temporary values like R and R1 (which are used to store the displacement vector in each simulation cycle) and P1 and P2 (which are copies of the current mesh configuration [px,py,pz]) and the SParams structure which holds the values of the simulation, namely the mechanical parameters of the breast, the max number of iterations, the density of the cells and the Chassaignac coefficient used in the calculation of the strain energy for the elements in that boundary. This structure also holds other parameters like derivative_precision which represents the threshold value $\varepsilon$ in figure 33 and log_parameter which is used in some methods like **line_search** to determine when the optimal displacement factor was found.

### 5.4.2 *New Structure profiling*

After implementing the new structure, a new set of tests was conducted in order to assess its performance. These tests check the speedup obtained, the IPC, cache miss percentages and also the number of cycles, instructions and cache related events. To better understand the difference in terms of code's performance, these results are compared with the ones obtained before (done with the same machine).

Figure 37 presents the weight of each method of the simulation process using GProf. There are not significant changes when comparing this call graph with the one present in the section 5.3.1 (**line_search** continues to be the most expensive method). However, big changes can be seen in the measured values with linux's perf.

Table 21 shows the difference between the old and new solution. In terms of cycles there is a reduction that goes from 28% for a coarse mesh to 39% to a thin mesh. This value ultimately represents a significant reduction in time consumption by the code architecture. In terms of instructions the reduction rounds the 50% for every mesh size. In terms of L3 cache misses the reduction is even more significant reaching 99% for thinner meshes having only 80% reduction in L3 cache references. For the L1 cache there is also a significant re-

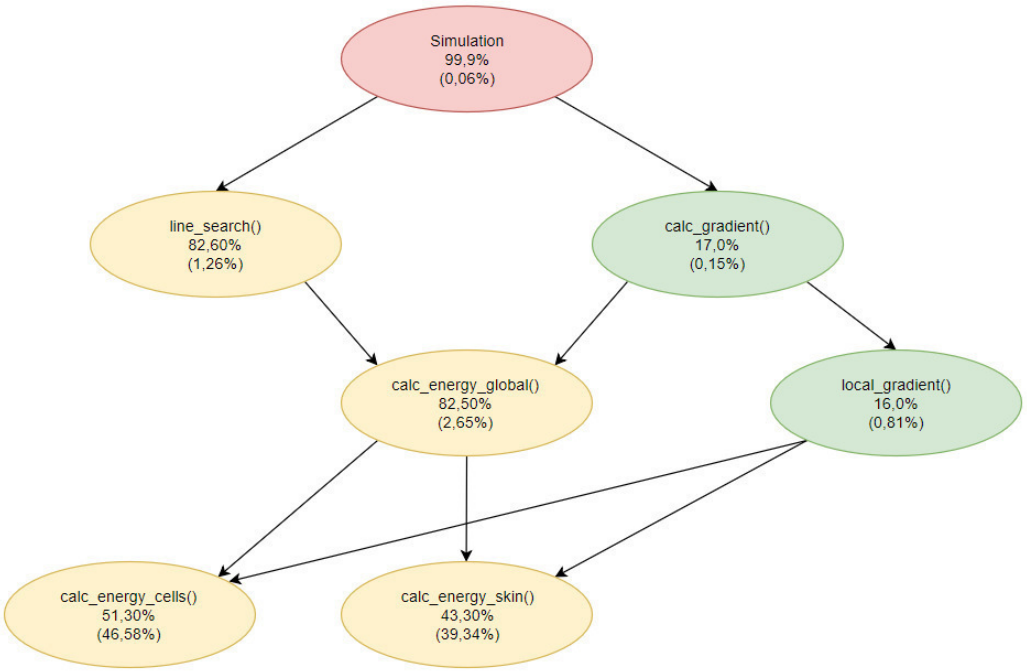



Figure 37: Call graph representing the weight of each major method called during the simulation with a new data structure.

| *Events* | Coarse (3,212 MB) | Medium (4,848 MB) | Thin (6,452MB) |
|---|---|---|---|
| Cycles | 28,4 | 34,0 | 39,3 |
| Instructions | 51,5 | 55,4 | 56,6 |
| L3 cache misses | 62,5 | 99,3 | 99,7 |
| L3 cache references | 94,4 | 70,4 | 79,4 |
| L1 Load cache misses | 61,0 | 51,1 | 36,6 |
| L1 Load cache references | 51,3 | 55,2 | 56,6 |
| L1 Store cache misses | 40,5 | 36,7 | 40,5 |
| L1 Store cache references | 67,4 | 70,8 | 71,8 |

Table 21: Comparison between old and new solution. This table shows the percentage reduction of the measured events from the old code to the new code.

duction in both misses and references (load and store). In this particular case the reduction is more expressive in the L1 store cache misses.

Table 22 shows the IPC of the simulation methods of the new code architecture. The table shows that the IPC of the methods, excepting the **calc_energy_cell**, is bellow 1. When

| IPC | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 0.80 | 1.39 | 0.84 | 1.86 | 0.88 | 1.52 |
| **calc_energy_global** | 0.79 | 0.66 | 0.83 | 0.79 | 0.86 | 0.87 |
| **line_search** | 0.80 | 2.16 | 0.84 | 2.50 | 0.89 | 2.19 |
| **calc_gradient** | 0.80 | 0.79 | 0.82 | 0.83 | 0.87 | 0.87 |
| **local_gradient** | 0.79 | 0.92 | 0.82 | 0.79 | 0.87 | 0.83 |
| **calc_energy_skin** | 0.48 | 0.48 | 0.47 | 0.47 | 0.50 | 0.50 |
| **calc_energy_cell** | 1.35 | 1.35 | 1.37 | 1.37 | 1.37 | 1.37 |

Table 22: IPC of the simulation methods with different sized meshes

comparing these values, with the previous version one can see that they are in fact lower. However, the number of instructions and cycles is much lower in this version of the code. This means that despite performing less instructions per cycle, this code is more efficient and also that can be improved in terms of computation. One other aspect worth mentioning is the non appearance of the **calc_energy_chass** when profiling the code with perf. This particular method consists on a very simple calculation with low instructions and it represents less than 1% of all time spent during the simulation, and therefore its deep evaluation was neglected.

| L3 miss % | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 0.14 | - | 0.06 | 0.18 | 0.32 | 1.52 |
| **calc_energy_global** | 0.24 | - | 0.06 | 0.03 | 0.26 | 0.58 |
| **line_search** | 0.13 | - | 0.03 | 0.01 | 0.26 | 0.27 |
| **calc_gradient** | 0.19 | 1.45 | 0.12 | 0.61 | 0.40 | 0.90 |
| **local_gradient** | 0.20 | - | 0.12 | 0.28 | 0.40 | 0.58 |
| **calc_energy_skin** | 0.35 | 0.35 | 0.09 | 0.09 | 0.36 | 0.36 |
| **calc_energy_cell** | 0.42 | 0.42 | 0.04 | 0.04 | 0.16 | 0.16 |

Table 23: L3 cache miss % of the simulation methods with different sized meshes. Cells with '-' mean that the values were smaller than 0.01%

Table 23 shows a large improvement on the L3 cache miss %. Almost all values show a L3 miss % inferior to 1. The reduction of this miss % is significant and show the improvement in locality of the data. Another factor that contributes to the miss % improvement is the reduced size of the data structure (from 4MB, 8MB and 17MB to only 3MB, 4MB and 6MB).

In terms of L1 load miss %, the new values are very similar to the previous version as shown in table 24. The same can be said of the L1 store miss % (Table 25).

These new results show an architecture with less cycles and even less instructions. Moreover, there is a huge reduction in terms of data accesses which are reflected by the 50%

| L1 Load miss % | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 1.98 | 20.80 | 3.19 | 13.47 | 4.77 | 14.93 |
| **calc_energy_global** | 1.80 | 3.10 | 3.28 | 2.43 | 4.98 | 2.65 |
| **line_search** | 2.01 | 8.63 | 3.35 | 5.23 | 5.23 | 13.46 |
| **calc_gradient** | 1.79 | 1.01 | 2.42 | 3.22 | 2.64 | 4.80 |
| **local_gradient** | 1.77 | 2.24 | 2.38 | 2.24 | 2.46 | 2.98 |
| **calc_energy_skin** | 1.18 | 1.18 | 2.88 | 2.88 | 1.57 | 1.57 |
| **calc_energy_cell** | 1.65 | 1.65 | 3.00 | 3.00 | 5.41 | 5.41 |

Table 24: L1 cache load miss % of the simulation methods with different sized meshes

| L1 Store miss % | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 0.61 | 4.11 | 0.63 | 17.28 | 0.60 | 20.63 |
| **calc_energy_global** | 0.63 | 4.63 | 0.47 | 19.32 | 0.40 | 23.08 |
| **line_search** | 0.68 | 2.18 | 0.68 | 7.69 | 0.64 | 11.45 |
| **calc_gradient** | 0.24 | 0.09 | 0.26 | 0.68 | 0.28 | 0.64 |
| **local_gradient** | 0.18 | 2.39 | 0.21 | 1.76 | 0.27 | 2.60 |
| **calc_energy_skin** | 1.75 | 1.75 | 1.13 | 1.13 | 0.89 | 0.89 |
| **calc_energy_cell** | 1.00 | 1.00 | 0.98 | 0.98 | 1.04 | 1.04 |

Table 25: L1 cache store miss % of the simulation methods with different sized meshes

reduction of L1 Load cache references and by approximately 80% of L3 cache references (thin mesh). This means that the memory bottleneck of the previous solution does not have the same impact in this new version of the code architecture. To check this hypothesis the arithmetic intensity of this new code's version was measured (Table 26).

| *Method* | Adds and Muls (AM) | Divs and Sqrts (DS) | Loads (L) | Arithmetic Intensity $\frac{max(AM,DS)}{L}$ |
|---|---|---|---|---|
| **Cells** | 57 | 0 | 12 | 4.75 |
| **Chassaignac** | 11.5 | 60 | 7.5 | 3.64 |
| **Skin** | 44 | 360 | 16.5 | 48.00 |

Table 26: Arithmetic intensity of the energy evaluation methods (cells, Chassaignac and skin faces) based on number of cycles. Ratio between the number of cycles spent on performing calculations (AM and DS) and loads (L).

The values of bandwidth are the same as the ones seen in Table 19. Just as before, one considers 11.5 GB/s as the memory bandwidth. So, for the 2.3GHz processor used, for each second 1.437.500.000 doubles per second which in cycles turns into 0.625. So, the new

maximum amount of doubles that can be loaded during the time of calculations ca be seen in table 27.

| *Methods* | Calculating cycles | Loading operations | Max Operations |
|---|---|---|---|
| Cells | 57 | 8 | 36 |
| Chassaignac | 60 | 5 | 38 |
| Skin | 360 | 11 | 225 |

Table 27: Comparison between the number of doubles loaded in each methods vs the maximum number that could be loaded if the bandwidth was fully utilised.

These values of arithmetic intensity and loading operations are very different from the ones present in section 5.1. In fact, the arithmetic intensity of the methods improved from 1.49, 3.10 and 13.06 to 4.75, 3.64 and 48.00 (**calc_energy_cells**, **calc_energy_chass** and **calc_energy_skin** respectively). It is also observable the reduction in loading operations from 26, 14 and 22 to just 8, 11 and 5 (**calc_energy_cells**, **calc_energy_chass** and **calc_energy_skin** respectively). This reduction in loading operations in these methods reinforces the idea that the new data structures have improved the code's performance by improving the data locality and accesses.

*Speedup*

These improvements on the code in terms of time can be seen in table 28.

| Time (s) | Base Code Architecture | New Code Architecture | Speedup |
|---|---|---|---|
| **Coarse** | 49.02 | 35.59 | 1.38 |
| **Medium** | 244.54 | 166.52 | 1.47 |
| **Thin** | 601.03 | 404.32 | 1.49 |

Table 28: Comparison of the time (in seconds) consumption using the previous code version (base code architecture) and the new version (new code architecture) as well as the speedup obtained for three different mesh sizes.

By changing the data structures, the time required to complete a simulation for three different mesh sizes, namely coarse, medium and thin, reduced from 49, 245 and 601 seconds to just 36, 167 and 404 seconds respectively. This represents an approximate speedup of 1.45 times which is very positive. So, these results prove that by reducing the memory dependency, one was able to improve the code's efficiency and therefore improve the overall performance.

The next step concerns the improvement of the algorithms and in the increase of the amount of floating point operations that can be done at the same time.

### 5.4.3 *Improved Algorithms*

With a new data structure small modifications of the methods were made. These modifications were necessary to accommodate this new organisation. However, some parts of the code, specially the **line_search** method, still showed poor performance. The simulation as well as the **line_search** function are convergence methods. The speed of these methods depend not only on how optimised the code is, but also on the minimum number of iterations necessary to achieve an accurate result. With **line_search** being the method were more time is consumed, the focus is on improving its convergence by reducing the number of necessary iterations to find the optimal displacement factor ($\beta$).

The tests showed that each time the **line_search** method was called, it was necessary to perform, on average, 10 to 15 iterations to find an optimal $\beta$. The same tests also showed that the conversion method was fast at reaching an approximate value (4 to 5 iterations) but it was slow at the refining step (6 to 10 iterations). With this in mind, one way to improve this method consisted in using an algorithm with a quadratic nature of convergence which is faster in this type of cases like the Newton method (Bonnans et al. (2006)).

So, a new algorithm was implemented based on the Newton method (Bonnans et al. (2006)) which calculates the values of strain energy of the mesh configuration after being displaced with by three different factors ($\beta$, $\beta+\epsilon$ and $\beta-\epsilon$) at each iteration and then calculates the new $\beta$ from the strain energy values obtained with each factor (algorithm 3).

**Algorithm 3:** New Line Search pseudo-code

```
1  Input: mesh τ, position Φ_h, DOF X_h, SParams p, AuxArch aa, double[] R;
2  double E, Em, Ep, ϵ, β;
3  while (Ep−Em)/(Ep+Em) > ϵ do
4      Φ_h ← dof2pos_global(Φ_h, X_h, β, R);
5      E ← calc_energy_global(m, Φ_h, p);
6      Φ_h ← dof2pos_global(Φ_h, X_h, β+ϵ, R);
7      Ep ← calc_energy_global(m, Φ_h, p);
8      Φ_h ← dof2pos_global(Φ_h, X_h, β-ϵ, R);
9      Em ← calc_energy_global(m, Φ_h, p);
10     β ← β − ((Ep−Em)*ϵ)/(−2.0*E+Ep+Em) * 0.5 ;
11 end
```

This algorithm, for each iteration, displaces the mesh's degree's of freedom using the displacement vector $R$ with different values of $\beta$ ($\beta$, $\beta+\epsilon$ and $\beta-\epsilon$) and evaluates for each one of them the systems strain-energy value. At the end of the cycle, it evaluates the new $\beta$ by subtracting to it, the difference factor between the strain-energy values of the $\beta+\epsilon$ and $\beta-\epsilon$ configurations until the difference of these values become smaller than a certain

threshold ($\epsilon$). This new method showed a fast approximation to the final value (1 to 2 iterations) and even faster refinement (1 to 2 iterations as well).

This new method was able to reduce the average amount of iterations to just 2,3 or 4 iterations, which results in a significant reduction in terms of cycles and instructions.

| Events | Coarse (3,212 MB) | Medium (4,848 MB) | Thin (6,452 MB) |
|---|---|---|---|
| **cycles** | 78.73 | 78.10 | 77.33 |
| **instructions** | 78.66 | 77.68 | 77.53 |

Table 29: Percentage Reduction of cycles and instructions of the code's architecture with the new **line_search** method

In fact, table 29 shows a reduction of approximately 77% of both cycles and iterations of the solution with a new **line_search** method. This major reduction results in an overall improvement of the simulation's time consumption. Because this method was responsible for about 70% of the time consumed by the simulation, this reduction changes the weight of the methods used in the simulation. To better understand the changes that this improvement brought to the code architecture, a new call graph was created.

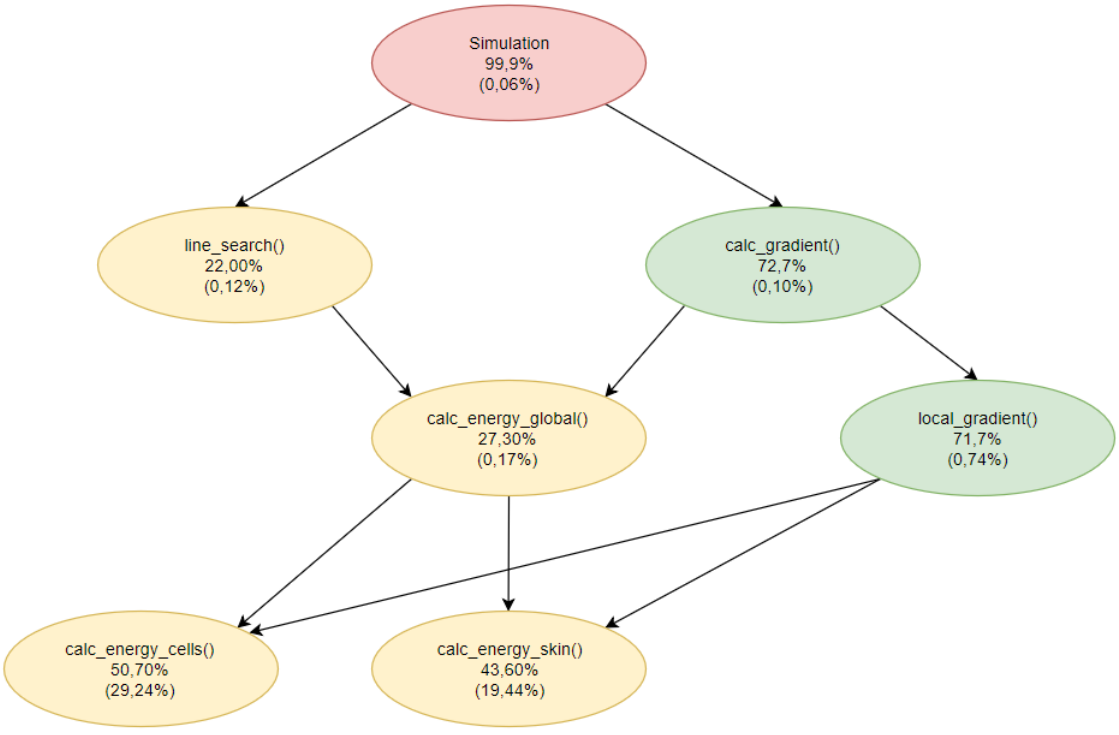



Figure 38: New callgraph representing the weight of each major method called during the simulation after the implementation of a new algorithm on the line_search method.

Figure 38 shows the new weight configuration after implementing the new **line_search** method. As it can be seen, **line_search** is no longer the heaviest method in terms of time consumption (now it just has approximately 22%). The **calc_gradient** method is now the one which consumes more cpu time with approximately 73% followed by the **local_gradient** method with 72%. Its is possible to see that the strain energy evaluation methods (**calc_energy_cells** and **calc_energy_skin**) have not a self time (values inside parenthesis) equal to the total time spent (values outside parenthesis). That happens due to the large amount of time spent in a single instruction which perf labelled as **_ieee754_log_avx** which represents approximately 35% of all time spent by the simulation. Both **calc_energy_cells** and **calc_energy_skin** require the calculation of the natural logarithm due to the Neo-Hookean model (equations 28 and 29).

One thing that is also worth to look into, is the impact of this new method on the memory, L3 and L1 cache accesses.

| **line_search** | **Coarse (3,212 MB)** | | **Medium (4,848 MB)** | | **Thin (6,452 MB)** | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **IPC** | 0,826 | 2,394 | 0,847 | 3,022 | 0,883 | 2,222 |
| **L3 miss %** | 0,258 | 0,093 | 0,031 | 0,027 | 1,120 | 1,247 |
| **L1 load miss %** | 1,931 | 9,048 | 3,379 | 11,883 | 5,315 | 12,383 |
| **L1 store miss %** | 0,572 | 1,431 | 0,616 | 13,681 | 0,524 | 8,959 |

Table 30: New values of IPC, L3 miss %, L1 load and store miss % of the new line_search method for different sized meshes

Table 30 shows what changed in memory/cache accesses on this new version of the code. In terms of IPC, the improvement is small and it is better seen in the self IPC which goes from 2.16, 2.50 and 2.19 to 2.39, 3.02 and 2.22 (for coarse, thin and ultra medium meshes respectively). In terms of L3 miss % there is a slight increase from 0.13, 0.03 and 0.26 to 0.26, 0.03 and 1.12. For the L1 load and store miss % the values remain very similar. These values point to a small impact of this new method in terms of IPC, L3 and L1 miss % rates. This means that this new method's main impact concerns the overall time consumption which was drastically reduced (table 31).

| **Time (s)** | **New Code Architecture** | **Improved Line_Search** | **Speedup** |
|---|---|---|---|
| **Coarse** | 35.59 | 7.34 | 4.85 |
| **Medium** | 166.52 | 39.99 | 4.16 |
| **Thin** | 404.32 | 96.51 | 4.19 |

Table 31: Comparison of the time (in seconds) consumption using the previous code version (new code architecture) and the improved algorithms version (improved line_search) as well as the speedup obtained for three different mesh sizes.

The values presented in table 31 show that the improved algorithms were responsible for a large performance improvement of the **simulation** method. For all three different mesh sizes, the simulation speedup was higher than 4 times.

As the impact of the **line_search** method reduces, the impact of the **calc_gradient** increases. So, the focus on optimising the code would necessary pass by optimising the **calc_gradient** method. **line_search** is a convergence method and by studying its nature it was possible to find that it took too many iterations (around 10-14) and it was found a different algorithm that was able to converge in less iterations. The **calc_gradient** method is not a convergence method which means that the techniques used to improve **line_search** are not useful. Moreover, it was difficult to find other ways to improve the code without considering a parallel approach.

Another aspect worth mention is the weight of the calculation of the natural logarithm. Alternative methods to calculate the logarithm were implemented and tested using pre-calculated tables, using Taylor expansion series with simplifications, but no improvements were observed. The main problem with this calculation is the precision required for the calculations: values with small precision lead to accumulation of errors which reflect in the final result and high precision requires time to achieve. Any solution found that could speed up the code has lead to potential problems in the solution, for example, using pre-calculated tables lead to cases were the simulation crashed because it would end up making calculations with Infinit (INF) values or Not a Number (NaN).

A large portion of the time is spent in the methods that calculate the values of strain energy, namely **calc_energy_cells** and **calc_energy_skin** and both these methods have multiple operations that can be done simultaneously, *i.e.* it is possible to calculate the values of strain energy of multiple cells and faces at the same time because the calculations are independent. Therefore, in the next section, a new approach was taken that takes advantage of the code's parallel nature.

## 5.5 PARALLELISM

To perform multiple operations on a single computer there are three main methods: Vectorisation, CPU multi-threading parallelisation and GPU parallelisation. Intel's and AMD's SIMD instructions allow to perform operations like additions and multiplications simultaneously (AMD (2015); Intel (2016)). The increase in the number of cores in modern processors has increased the potential of using multiple threads to perform multiple tasks at the same time (Bryant and O'Hallaron (2016)). The GPU is a hardware capable of processing large amounts of data at the same time using the multiple processing units it contains (Owens et al. (2008); Blythe (2008)).

Besides the machine used in the previous tests (**CPU A** - Intel i5-2450m), others were used for the tests, with one of them being used for the tests with the GPU version of the code:

- **CPU B** - Intel Xeon CPU ES-2650 v2 2.60Ghz with 32 cores
- **CPU C** - AMD FX-8350 4.0Ghz with 8 cores
- **CPU D** - Intel i7-6700HQ 2.60GHz with 4 cores and a NVIDIA GeForce GTX 1060 with 1280 Compute Unified Device Architecture (CUDA) cores and 1404MHz

The tests presented in this section aim to evaluate the scalability of the solution as well as how much can be gained from implementing algorithms to take advantage of parallelism.

### 5.5.1 *Vectorisation*

In order to take advantage of the vectorial units in modern CPU the calculations were packed into groups of 2 and 4 elements (Huang et al. (2013); Zhou et al. (2018)). Also, the packing was done with operations of the same type ,*i.e.* vector operations do not work when there is one addition and one multiplication. It performs better when multiple of one of these types of instructions is used (Huang et al. (2013); Zhou et al. (2018)).

#### *Implementation of Intrinsics*

In order to improve the code's performance further, SIMD instructions were implemented. The first approach consisted in the usage of Intel intrinsic functions (Intel (2016)). As mentioned before, for these functions to work properly, the multiple variables that were used in the computations were grouped into small arrays of 2 and 4 units. Then the intrinsic functions would use these small arrays and perform the vector operations.

Due to the necessity of having multiple instructions of the same type, of all the methods in the code architecture, only three methods fill the requirements: **calc_energy_chass**, **calc_energy_skin** and **calc_energy_cell**. So, only in these methods the variables were modified to fit the vector units of the CPU and new tests were made to evaluate their impact (using the same machine as the other tests). Table 32, shows the reduction in terms of cycles and instructions after the change of the methods to use SIMD instructions.

The implementation of vectorisation has managed to reduce the amount of instructions by approximately 10-14%, and the amount of cycles by approximately half. It also shows reductions in terms of L3 cache accesses and misses. In terms of L1 cache accesses and misses, the number increased after the inclusion of SIMD instructions. The most significant aspect of these results is the reduction in half of the time consumed by the simulation. This shows the potential that the vector units have in improving code's with multiple calculations.

| SIMD | Coarse (3,212 MB) | Medium (4,848 MB) | Thin (6,452 MB) |
|---|---|---|---|
| **Cycles** | 50 | 50 | 47 |
| **Instructions** | 14 | 14 | 10 |
| **L3 cache misses** | 23 | 22 | 19 |
| **L3 cache references** | 27 | 7 | 6 |
| **L1 Load cache misses** | 1 | 1 | -1 |
| **L1 Load cache references** | 0 | 0 | -1 |
| **L1 Store cache misses** | -3 | 0 | -3 |
| **L1 Store cache references** | -10 | -14 | -16 |

Table 32: Reduction in percentage of cycles and instructions after the implementation of SIMD intrinsic functions to replace parts of the **calc_energy_chass**, **calc_energy_skin** and **calc_energy_cell** methods. Negative values represent a percentage increase in comparison to the previous reported values.

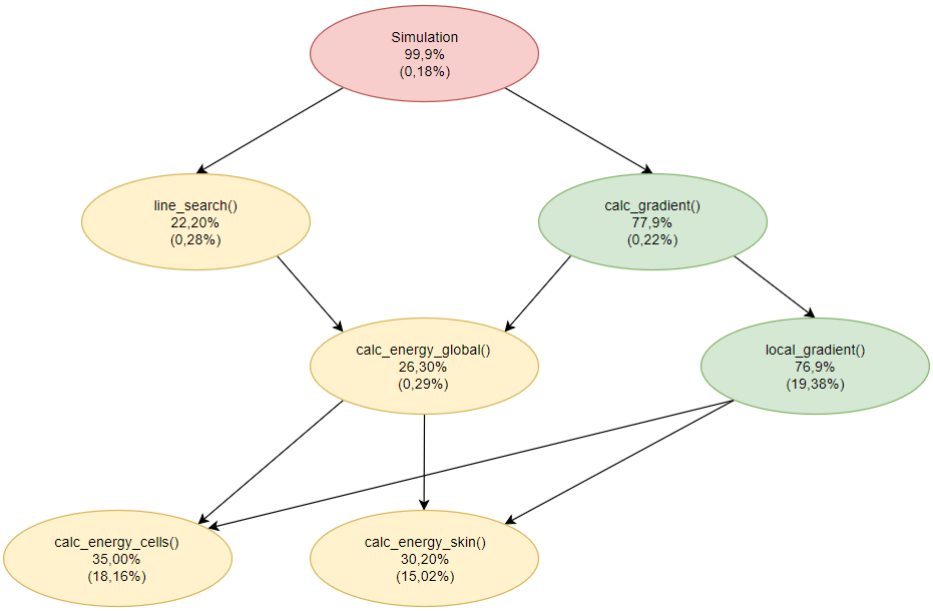



Figure 39: New callgraph representing the weight of each major method called during the simulation after implementing SIMD intrinsics.

The reduction of cycles changed once again the relative weight of the main methods of the code architecture after the new improvements (figure 39). It is possible to see that the relative weight of each method changes slightly with the self time consumption of the

local_gradient method increasing from about 1% to almost 20% and a reduction of the total time spent by the **calc_energy_cells** and **calc_energy_skin** methods to 35% and 30% respectively. Again around 30% of the total time spent is with the **_iee754_log_avx** method (only visible using linux's perf).

*Profiling SIMD instructions*

The reductions in terms of cycles due to the introduction of SIMD instructions is significant. However, one must check how these instructions affect the code's performance not only in terms of cycles and instructions, but also in terms of memory access. The different reduction in terms of cycles spent and number of instructions leaves to different values of IPC. These values can be seen in table 33.

| IPC | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 1,418 | 1,182 | 1,461 | 1,168 | 1,470 | 1,268 |
| **calc_energy_global** | 1,450 | 1,839 | 1,514 | 1,829 | 1,538 | 2,331 |
| **line_search** | 1,534 | 6,992 | 1,523 | 1,829 | 1,539 | 1,881 |
| **calc_gradient** | 1,541 | 1,385 | 1,600 | 1,437 | 1,606 | 1,450 |
| **local_gradient** | 1,631 | 1,631 | 1,648 | 1,648 | 1,647 | 0,081 |
| **calc_energy_skin** | 0,923 | 0,923 | 1,140 | 1,140 | 1,114 | 1,114 |
| **calc_energy_cell** | 2,869 | 2,869 | 3,109 | 3,109 | 2,821 | 2,821 |

Table 33: New values of IPC for the simulation methods with different sized meshes

The improvement of the methods IPC is substantial in relation to the values in table 30. Previously, apart from the **calc_energy_cell**, every other method's IPC was bellow 1. Now every method (except for the **calc_energy_skin** for a coarse mesh) presents a number of instructions per cycle above 1, being **calc_energy_cell** the method with the highest IPC (around 3).

As mentioned previously, when performing multiple instructions simultaneously, the data needs to be packed. This includes not only the packing of instructions like additions or multiplications but also the of instructions that are responsible for storing and loading data. By packing loads and stores, the way the code accesses the cache also changes.

Table 34 shows an increase of the miss % in the L3 cache. Before those values were inferior to 1 in every instance. The inclusion of vectorisation increased the number of L3 misses per access. However the increase is very small. Except for the **calc_energy_global**, every method presents a L3 miss % inferior to 2 which does not cause major penalties in the overall performance of the code (in fact the reduction of cycles by half proves this).

In terms of L1 cache access, Tables 35 and 36 show the impact of the implementation of SIMD intrinsic functions. There is a small improvement when one compares the

| L3 miss % | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 0,994 | 0,151 | 0,055 | 0,175 | 1,208 | 3,630 |
| **calc_energy_global** | 2,059 | 0,082 | 0,110 | 0,001 | 1,798 | 1,286 |
| **line_search** | 0,453 | 0,035 | 0,037 | 0,053 | 1,707 | 1,806 |
| **calc_gradient** | 1,073 | 9,035 | 0,041 | 0,515 | 1,074 | 2,186 |
| **local_gradient** | 1,067 | 1,067 | 0,028 | 0,028 | 0,975 | 0,975 |
| **calc_energy_skin** | 0,068 | 0,068 | 0,101 | 0,101 | 1,918 | 1,918 |
| **calc_energy_cell** | 0,084 | 0,084 | 0,038 | 0,038 | 1,035 | 1,035 |

Table 34: New values of L3 miss % for the simulation methods with different sized meshes

| L1 Load miss % | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 1,807 | 10,885 | 2,616 | 28,657 | 3,183 | 22,159 |
| **calc_energy_global** | 1,548 | 1,357 | 2,622 | 0,547 | 4,116 | 0,617 |
| **line_search** | 1,693 | 8,175 | 2,797 | 9,357 | 4,205 | 11,707 |
| **calc_gradient** | 1,757 | 1,843 | 2,589 | 2,632 | 2,964 | 3,188 |
| **local_gradient** | 1,751 | 1,751 | 2,572 | 2,573 | 2,776 | 2,776 |
| **calc_energy_skin** | 0,739 | 0,739 | 1,006 | 1,006 | 0,694 | 0,694 |
| **calc_energy_cell** | 1,723 | 1,723 | 3,289 | 3,289 | 4,965 | 4,965 |

Table 35: New values of L1 Load miss % for the simulation methods with different sized meshes

**calc_energy_skin** and **calc_energy_cell** methods L1 load miss %, one can see a reduction from 1.57 and 5.41 to 0.69 and 4.97 (for the case of the thin mesh).

| L1 Store miss % | Coarse (3,212 MB) | | Medium (4,848 MB) | | Thin (6,452 MB) | |
|---|---|---|---|---|---|---|
| | total | self | total | self | total | self |
| **simulation** | 0,278 | 8,371 | 0,099 | 3,949 | 0,300 | 13,725 |
| **calc_energy_global** | 0,438 | 9,866 | 0,091 | 12,177 | 0,263 | 17,009 |
| **line_search** | 0,399 | 7,043 | 0,142 | 3,145 | 0,419 | 11,962 |
| **calc_gradient** | 0,216 | 0,335 | 0,061 | 0,106 | 0,177 | 0,317 |
| **local_gradient** | 0,137 | 0,137 | 0,055 | 0,055 | 0,156 | 0,016 |
| **calc_energy_skin** | 0,797 | 0,797 | 0,111 | 0,111 | 0,204 | 0,204 |
| **calc_energy_cell** | 0,312 | 0,312 | 0,095 | 0,095 | 0,291 | 0,291 |

Table 36: New values of L1 Store miss % for the simulation methods with different sized meshes

The changes on the L1 store miss chances evolves from 0.89 and 1.04 to 0.20 and 0.29 (thin mesh). However, when one considers these values with the ones in table 32, the improvement in miss % is mainly a result of the greater increase in L1 store cache references than misses. So, in general, the L1 store cache has worsen with this new version of the code.

However, as the impact of these instructions is still very small, this increase does not cause performance issues.

In general, the inclusion of SIMD instructions on the methods responsible for calculating the values of strain-energy of the cells and faces have improved the overall performance of the code.

*Comparison between different machines*

Because the performance of these type of instructions may depend on the processor micro architecture (Intel (2016)) performance tests were done. The objective was to compare the improvement between the version with and without SIMD instructions.

| **Time (s)** | | **CPU A** | **CPU B** | **CPU C** | **CPU D** |
|---|---|---|---|---|---|
| **Coarse** | ***base*** | 7.34 | 3.11 | 1.90 | 2.34 |
| | ***simd*** | 3.74 | 1.84 | 1.83 | 1.22 |
| **Medium** | ***base*** | 39.99 | 17.89 | 9.79 | 16.91 |
| | ***simd*** | 19.25 | 9.63 | 9.40 | 6.46 |
| **Thin** | ***base*** | 96.51 | 48.42 | 25.81 | 46.41 |
| | ***simd*** | 53.02 | 26.35 | 25.68 | 16.23 |

Table 37: Times in the different machines of the simulation method with simd (simd) and without simd (base) instructions

Table 37 shows the different results with and without the application of SIMD instructions on the code. The results show that the implementation of these instructions improved the overall performance approximately by 2 times in most machines (CPU A, B and D). The solution worked differently for the machine with an AMD micro architectures (CPU C) where there is not as significant difference between the code version with and without intrinsics (only a slight improvement). Looking at the times of the other machines, namely CPU B and CPU D, it seems that there are two possible explanations for such results: one is that the higher clock frequency of the CPU (4GHz of CPU C vs 2.6GHz of CPU B and D) is such that it can perform more calculations per second and therefore obtain better timings; the other is that, somehow the compiler in the AMD was capable of performing some of these optimisations using the SIMD units of the processor without the explicit implementation of the intrinsic functions. In fact, concerning this last possibility, it was possible to find that the code version that run on the AMD without the explicit implementation of the intrinsic functions showed the execution of vectorial instructions. The same did not happen with the Intel processors.

### 5.5.2 *CPU Parallelism*

The inclusion of SIMD instructions increased the overall performance of the code architecture. However, this performance was obtained from using a single core. Nowadays, every CPU possesses 2 or more cores which can be used simultaneously to process data (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016)). So, in order to further reduce the amount of time necessary to complete a simulation, a solution which could use multiple threads was implemented.

When evaluating the potential speedup of parallelism in the code, one has to take into account Amdahl's law given by equation 65 (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016))

$$speedup(s, p) = \frac{1}{(1-p) + \frac{p}{s}} \tag{65}$$

where, $s$ represents the speedup of the parallel section of the code, and $p$ represents the portion of the total code which has a parallel nature. This law states that the potential to speedup the code depends on the portion of the code which is parallel. If only half the code is parallel, then the maximum theoretical speedup would be 2.

Of the simulation's methods, **calc_energy_cells**, **calc_energy_skin** and **local_gradient** have a parallel nature. However, the **local_gradient** method is not 100% parallelisable due to some dependencies (calculation of the partial derivatives forces the change of one degree of freedom at a time). So, a possible estimate of the parallelisable code would be around $70 - 75\%$. This means that the maximum speedup one can achieve in a situation with a large number of cores is about 3.3 to 4 times.

To implement multiple threads running at the same time, and considering that the code is written in C++ there are several alternatives like using Pthreads (Narlikar and Blelloch (1998)) and other Application Programming Interface (API) like OpenMP (Board (2015)).

This API allows to create a localised pool of threads that can be used to perform different instructions at the same time using different physical processor units. It is important to mention that the setup, and usage of the OpenMP thread generate an overhead that must be taken into account when evaluating the speedup obtained. The implementation of the multiple threading was done majorly in for loops, where the work would be split evenly by the threads in the pool. An example of this can be seen in the **calc_energy_global** method (algorithm 4).

Because each face in the Chassaignac space and the skin as well as each cell are independent, each thread picks an element to calculate the value of strain energy and when it finishes picks other until every elements was processed. This procedure was easy but in the case of the **local_gradient** method which has dependencies a different approach was required. In this method, to calculate the local variation of energy, a degree of freedom is temporarily changed (*e.g.* the x value of a node is moved) and the effect of this change is

**Algorithm 4:** Global Energy evaluation pseudo-code

```
1  Input: mesh τ, position Φ_h, DOF X_h, SParams p;
2  double sum = 0;
3  for each chass ∈ τ do
4  |  sum += calc_energy_chass(chass);
5  end
6  for each skin ∈ τ do
7  |  sum += calc_energy_skin(skin);
8  end
9  for each cell ∈ τ do
10 |  sum += calc_energy_cell(cell);
11 end
```

evaluated. When this evaluation ends the value of that degree of freedom returns to its original value and a new degree of freedom is in turn changed and the process repeats. This causes dependency on the data structure which holds the temporary mesh configuration. To overcome this problem, a copy of that structure was made for each thread. Then the number of degrees of freedom was split by the number of threads requested for the simulation and each one of those threads would calculate the local gradient for their elements. Despite the overhead of copying these structures for every thread, the work split benefits the overall performance by the methods and in turn the simulation process.

Once again, the parallel solution was tested in several machines and the results can be seen in table 38. The results show the average of 5 tests where the stop criteria was set to be 1500 iterations independently of the energy residual value. With this one ensures that the simulation method performs the same amount of calculations.

The results show that the usage of multiple threads to perform calculations simultaneously is capable of reducing the time of the simulations. It is also possible to see by the results that the speed up obtained with thinner meshes is higher than the speed up obtained with the coarse mesh. For example, for CPU A, the improvement of the simulation with the coarse mesh is of approximately 1.5 times while the improvement with the thin mesh is of approximately 1.75 times. For CPU B the improvement with a coarse mesh is of approximately 1.8 while for a thin mesh the improvement reaches 5 times (26.25 seconds with 1 thread and only 5.25 seconds with 16 threads). This difference between a coarse and thin mesh can be explained by the overhead of initialisation of the OpenMP threads (Board (2015)), *i.e.* a coarse mesh as less elements and therefore the processor needs less time to calculate when compared with the thin mesh which has many more elements. So, the overhead has a greater impact in the coarse simulations than with thinner meshes.

Concerning the Intel processors (with the exception of CPU B), namely CPU A and D, which despite having only two and four cores respectively, are capable of processing four

| Time (s) | | 1 | 2 | 4 | 8 | 16 | 32 |
|---|---|---|---|---|---|---|---|
| CPU A | *Coarse* | 3.74 | 2.71 | 2.44 | - | - | - |
| | *Medium* | 19.25 | 13.04 | 11.62 | - | - | - |
| | *Thin* | 53.02 | 33.59 | 30.15 | - | - | - |
| CPU B | *Coarse* | 1.84 | 1.75 | 1.24 | 1.02 | 1.09 | 1.31 |
| | *Medium* | 9.63 | 7.88 | 5.53 | 3.15 | 2.93 | 3.55 |
| | *Thin* | 26.35 | 20.48 | 13.71 | 7.40 | 5.25 | 5.95 |
| CPU C | *Coarse* | 1.83 | 1.36 | 0.98 | 0.88 | - | - |
| | *Medium* | 9.40 | 6.89 | 4.66 | 3.21 | - | - |
| | *Thin* | 25.68 | 18.07 | 11.49 | 7.77 | - | - |
| CPU D | *Coarse* | 1.22 | 0.84 | 0.67 | 0.67 | - | - |
| | *Medium* | 6.40 | 4.26 | 2.75 | 2.55 | - | - |
| | *Thin* | 16.23 | 10.75 | 6.62 | 6.3 | - | - |

Table 38: Simulation time spent for every test machine using multiple treads (1, 2, 4, 8, 16 and 32). Cells with "-" mean that the test was not performed because the number of threads exceed the number of cores (including hyper-threading technology).

and eight threads at the same time due to the Hyper-Threading technology (Intel (2016)). However, the improvements from the number of physical cores (2 for the CPU A and 4 for the CPU D) to the number of logical cores (4 for the CPU A and 8 for the CPU D) is not great. A possible explanation is the limited number of arithmetic logical units which are being used at the same time and therefore even with two threads trying to work at the same time, in a specific arithmetic unit it is only possible to process one instruction at a time (Intel (2016)).

The improvements obtained with parallel calculations for the coarse mesh were of approximately 1.8 times, for the medium mesh the improvements reached the 2.6 times and for the thin mesh the improvements round the 3.17 times. The results obtained with the thin mesh are not very far from the expected values of speedup taking into account the parallel portion of the code. Nonetheless, for coarser meshes the speedup value does not match the expected. This can be explained by the OpenMP thread overhead that for a large number of elements becomes very small, but for meshes with few nodes can be significant. In summary, the implementation of multiple threads has proven successful in improving the simulation overall performance.

### 5.5.3 *GPU Parallelism*

A different approach of taking advantage of the code's parallel nature is to use the GPU. This hardware had a great acceleration in terms of performance over the last years, mainly due to the effort of the gaming industry in following the gamers expectations and of the

scientific community Owens et al. (2008); Nickolls and Dally (2010); Thomas and Daruwala (2018); Lisjak et al. (2018). This increase in performance came also from the evolution and increase necessity in processing massive amounts of data (Owens et al. (2008); Taylor et al. (2008); Thomas and Daruwala (2018)). There are two main reasons behind this:

1. The amount of data to process at real-time frame rates is growing with the increasing complexity of the applications (Owens et al. (2008); Taylor et al. (2008); Nickolls and Dally (2010));

2. Many applications and algorithms are parallel and the GPU architecture is well suited to deal with parallelism (Owens et al. (2008); Nickolls and Dally (2010); Thomas and Daruwala (2018)).

General Purpose Graphic Processing Units (GPGPU) are emerging as a preferred high performance computing platform for various general purpose parallel applications over conventional multi-core CPU (Thomas and Daruwala (2018)). The architecture of modern GPU consists in several Streaming Multiprocessors (SM) which contain multiple cores (figure 40) that are capable of performing integer, single-precision and some double precision operations (Blythe (2008); NVIDIA (2017)).

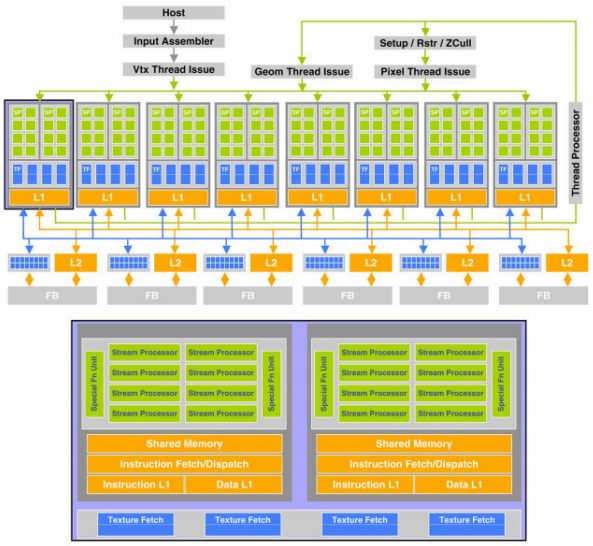



Figure 40: GPU architecture (Owens et al. doi:10.1109/JPROC.2008.917757)

These cores are much simpler than the ones of a CPU, *i.e.* they do not have some of the CPU features like branch handling/prediction (Hennessy and Patterson (2017); Bryant and O'Hallaron (2016); NVIDIA (2017)). So, instead of muscling through branches like a CPU, whenever a branch happens, the GPU launches another thread that will deal with the other possible path. For example, in an if else statement, the original thread would deal with the code assuming a true condition while the another thread would deal with the

code assuming a false condition converging the results at the end (NVIDIA (2017)). The existence of multiple cores to handle multiple threads is the great advantage that modern GPU offer to parallel algorithms, including FEM (Blythe (2008); Lisjak et al. (2018)).

There are some tools that use the GPU resources for science computing, namely OpenGL, OpenCL and CUDA (Owens et al. (2008); Taylor et al. (2008); Thomas and Daruwala (2018)). In fact there is a great volume of research reported in the literature about the potential and impact of the usage of GPGPU to solve problems like finite element analysis and soft tissue simulations (Taylor et al. (2007, 2008, 2010); Han et al. (2012b); Strbac et al. (2015)), gene sequencing (Rahman and Muniyandi (2018)), wildfires (Gregorio et al. (2013)) and many others (Zhu et al. (2012); J.Gibson et al. (2015); Badawi et al. (2018)). Although some of the projects used programming languages like C/C++ with CUDA and OpenCL, in terms of soft tissue modelling, the most prevalent usage of the GPU capabilities was due to Application Programming Interfaces (APIs) like SOFA and NiftySim (Comas et al. (2008); Eiben et al. (2016)). The advantages of using these API are the simplification in programming for researchers which are not comfortable generating optimal code for GPU (Johnsesn et al. (2015); Talbot et al. (2015)). SOFA was designed with the specific intention to help generate models for medical simulation Allard et al. (2007). This engine was created by some of the elements of Project Epidaure which were among the pioneers in soft tissue modelling. The work done by Comas et al. (2008) with the SOFA engine, allowed to demonstrate the potential of using graphic units to generate and treat meshes based on soft tissues theory. Also in terms of breast modelling, Han et al. (2012b) and Eiben et al. (2016) presented models which used NiftySim to improve the model performance.

Nau3D is a graphics engine which was developed in Universidade do Minho and provides an environment for developing projects whose code can run in the GPU using OpenGL, namely compute shaders (OpenGL: Compute Shader). Unlike the other API, Nau3D allows to program at a low level for the GPU and therefore one can have more control when developing the code. There are not many works which use compute shaders outside game development (Dharma et al. (2017)). A compute shader is made of code that will run for a certain number of threads which are predetermined by the number of thread groups and the number of threads within a group. The total number of threads is the product between the thread groups and the number of threads per group (OpenGL: Compute Shader). In terms of the code running inside the GPU, each thread group is scheduled to run at a certain point and inside each group, each thread will run its part of the code. Only when all threads of a group finish, is then considered that the thread group finished(OpenGL: Compute Shader).

The main problems in programming for the GPU concerns thread synchronisation (ensure that an error does not occur when a thread is writing into a place where another thread is reading) and the debugging of the code, but Nau3D has debug tools that allow to read

the values of the buffers in the memory of the GPU and it has tools that measure which parts of the code are consuming more or less resources (Liu and Fernandes (2014)). So, the code responsible for simulating the behaviour of the breast under gravity was modified to fit the GPU specifications.

*Nau3D project*

Programming for a CPU is different than programming for a GPU. While in CPU one can program code with a sequential view of the problem, in a GPU environment thinking and programming in parallel is almost obligatory because it is the hardware's purpose (Blythe (2008); Owens et al. (2008)). Moreover, the GPU possess some limitations in terms of data structures . That is why, in Nau3D, compute shaders can be used to launch multiple threads simultaneously to perform the calculations (for example the strain energy values of the breast mesh cells), and the data structures are reduced to simple arrays/buffers (buffers of doubles or integers in this particular case). Therefore, the breast simulation data structures were modified to fit the new style of programming. The Nau3D breast simulation project structure consists of:

- Scripting Files: scripts in Lua which are used to control the execution of the code and modify global variables that are used by the breast simulation;
- Data Files: documents which store the values of the buffers like the current value of strain energy of the cells, the x,y,z coordinates of the nodes, and so on;
- Shaders: files which contain the code that will run in the GPU;
- Material File: A different document which registers the buffers that are associated to the shaders, as well as registers the buffers initialisation (with values at zero or by loading values from a file);
- Project File: A Extensible Markup Language (XML) document registers the pipeline/-workflow of the simulation ( describes when a shader will run and if the shader execution depends on a script), setup global variables and rendering options.

The scripts are written in Lua and they are responsible for controlling global variables and the different simulation moments like the strain energy minimisation loop (the scripts check if the number of maximum number of iterations was reached and if the energy residual *nor* became lower than the pre-determine threshold value $\varepsilon$).

The data files are created with a function from the C++ breast simulation code architecture that writes in different files the values for the nodes (ids, codes, coordinates, connected cells and faces), cells (energy, volume, ids, codes, node ids), Chassaignac and skin faces (energy, surface area, codes, node ids) as well as other auxiliary data like a map between

the nodes current configuration and the unknown variables (degrees of freedom) and some pre-calculated expressions used to calculate the values of strain energy of the cells and faces.

The compute shader files contain the code that performs the calculations of strain energy of the cells, the faces in the Chassaignac space and of the skin as well as the code that modifies the breast configurations. These shaders were designed in terms of the type of mesh element, *i.e.* some shaders were coded around the mesh nodes (GPU versions of the **pos2dof**, **dof2pos** methods as well as **local_gradient** and **calc_residual**), the cells (GPU versions of **calc_energy_cells**) and the faces (GPU versions of **calc_energy_chass** and **calc_energy_skin**). In each shader it is defined the number of threads per groups that will run in the GPU.

The material file determines the buffers that are used in the shaders and the way they are organised. The buffers used in the breast simulation are very simple and made of only integers (ids of nodes, cells and faces, map between degrees of freedom and the breast node configuration) and doubles (values of energy, volume, area, coordinates and displacement vector). Essentially, the data structures reported in section 5.1 were split into individual arrays which can be one dimensional like the node ids and codes or can have multiple dimensions like the arrays that store the cells to which a node belongs (figure 41). The

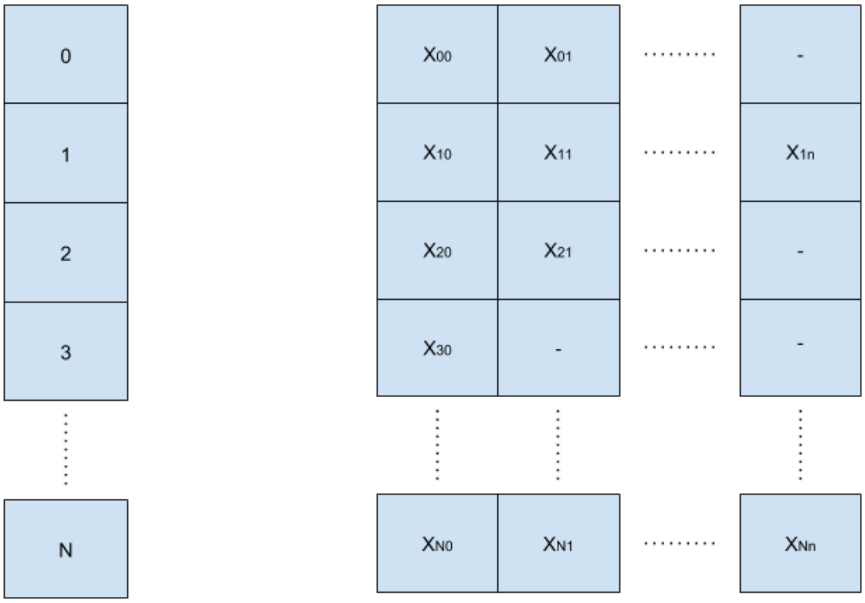



Figure 41: GPU data structures types. On the left its is the one dimensional buffer while on the right there is the composite buffer with fixed dimensions and that can have meaningful values (x) or empty values (-)

composite buffers were used mainly to associate the nodes and the cells, Chassaignac and skin faces they belonged to. For example, node with $id = 2$ could have 7 cells, 2 Chassaignac faces and 0 skin faces while a node with $id = 15$ could have 15 cells, 0 Chassaignac faces

and 5 skin faces. This means that the number of cells (and faces) between nodes is different. Having dynamic arrays in GPU may not be very efficient and so a choice was made to create these composite arrays with fixed dimensions ($n$) where the elements which are meant to be blank will have a value to mark that statement (for example $-1$). The material file also contains the buffers that are used to help in the calculations and contain global values necessary for the calculations.

The project file describes the simulation workflow. It describes the order in which the shaders will run and the scripts associated to each shader that control how they will run. It is in this file that its is determined the number of thread groups that are launched for each compute shader. This file also describes the global variables of the simulation like the global value of strain energy, the number of nodes, cells, Chassaignac and skin faces as well as the mechanical parameters of the breast ($\Lambda_m$).

*Nau3D Simulation Workflow*

The simulation of the breast behaviour under gravity is controlled by the shader order inside the project file and by the Lua scripts attached to each shader call in that same file. The shaders that are used in the simulation are similar to the methods of section 5.1. The shaders used during the simulation are:

- **pos2dof**($N_{coords}$, $N_{codes}$, $D_{coords}$) - copies the nodes coordinates from the mesh current configuration array ($N_{coords}$) to the array with the degrees of freedom ($D_{coords}$) according to the node codes ($N_{codes}$);

- **calc_energy_chass**($N_{coords}$, $Chass_{nid}$, $Chass_{energy}$, $Chass_{aux}$) - shader which calculates the strain energy for each face that belongs to the Chassaignac space. For that purpose the shader uses the information about the face's nodes ($Chass_{nid}$), namely their current coordinates ($N_{coords}$) as well as auxiliary information like the face's area ($Chass_{aux}$). In the end the value of energy is stored in $Chass_{energy}$;

- **calc_energy_skin**($N_{coords}$, $Skin_{nid}$, $Skin_{energy}$, $Skin_{aux}$) - shader which calculates the strain energy for each face that belongs to the skin. For that purpose the shader uses the information about the face's nodes ($Skin_{nid}$), namely their current coordinates ($N_{coords}$) as well as auxiliary information like the face's area ($Skin_{aux}$). In the end the value of energy is stored in $Skin_{energy}$;

- **calc_energy_cells**($N_{coords}$, $Cell_{nid}$, $Cell_{energy}$, $Cell_{aux}$) - shader which calculates the strain energy for each cell. For that purpose the shader uses the information about the cell's nodes ($Cell_{nid}$), namely their current coordinates ($N_{coords}$) as well as auxiliary information like the volume ($Cell_{aux}$). In the end the value of energy is stored in $Cell_{energy}$;

- **sum_energy**($Chass_{energy}$, $Skin_{energy}$, $Cell_{energy}$, $Global_{vars}$) - shader which sums the values of energy of the Chassaignac faces ($Chass_{energy}$), the skin ($Skin_{energy}$) and the cells ($Cell_{energy}$) and stores the value in the global variable buffer ($Global_{vars}$);

- **local_gradient**($N_{coords}$, $Global_{vars}$, $D_{map}$, $N_{chass}$, $N_{skin}$, $N_{cell}$, $R$, $Chass_{energy}$, $Skin_{energy}$, $Cell_{energy}$, $Chass_{aux}$, $Skin_{aux}$, $Cell_{aux}$) - shader which calculates the local gradient by moving one node coordinate ($N_{coords}$) if it belongs to the unknown variables ($D_{map}$) and evaluating the change in the values of strain energy for the local faces and cells using their auxiliary structures; ($Chass_{energy}$, $Skin_{energy}$, $Chass_{aux}$, $Skin_{aux}$ for the faces and $Cell_{energy}$, $Cell_{aux}$ for the cells). The variations on the values of local energy are stored in the displacement array $R$;

- **displace_nodes**($D_{coords}$, $R$) - shader which displaces the degrees of freedom ($D_{coords}$) according to the displacement vector ($R$);

- **dof2pos**($N_{coords}$, $N_{codes}$, $D_{coords}$, $D_{map}$) - shader which copies the degrees of freedom from ($D_{coords}$) to the nodes current configuration ($N_{coords}$) according to the node codes ($N_{codes}$);

- **calc_residual**($R$) - shader which calculates the value of the residual energy by summing the square values of $R$.

Some of these shaders only run once like **pos2dof** while others can run several times like **calc_energy_cells**. The workflow of the Nau3D simulation can be seen in figure 42.

The workflow of the simulation can be divided in 2 major sections: the initialisation and the energy minimisation loop. In turn each one of these sections is split into two main methods: calculation of the gradient and the calculation of the displacement factor. As mentioned previously, Lua scripts are attached to each call of a shader and the different scripts are used to modify global variables as well as control the workflow of the simulation. As the diagram in figure 42 shows, the initialisation the global value of energy is evaluated as well as the local gradient. After that the simulation, using the Lua scripts, calculates the best displacement factor $\beta$ until a condition is met ($\beta$ inferior to a given threshold value $\varepsilon_b$ which is similar to what happens with the **line_search** method of section 5.1).

In the main loop, the above shader execution orders are repeated but the main difference lies with the scripts that are used. One of the scripts run after the **local_gradient** shader to reset the variables for the optimal displacement factor evaluation and another scripts runs after the **calc_residual** shader not only to test if the optimal displacement factor was found but also to evaluate if the stop conditions of the strain energy minimisation loop were met, *i.e.*, the script check if the maximum number of pre-determined iterations (1500) was reached.

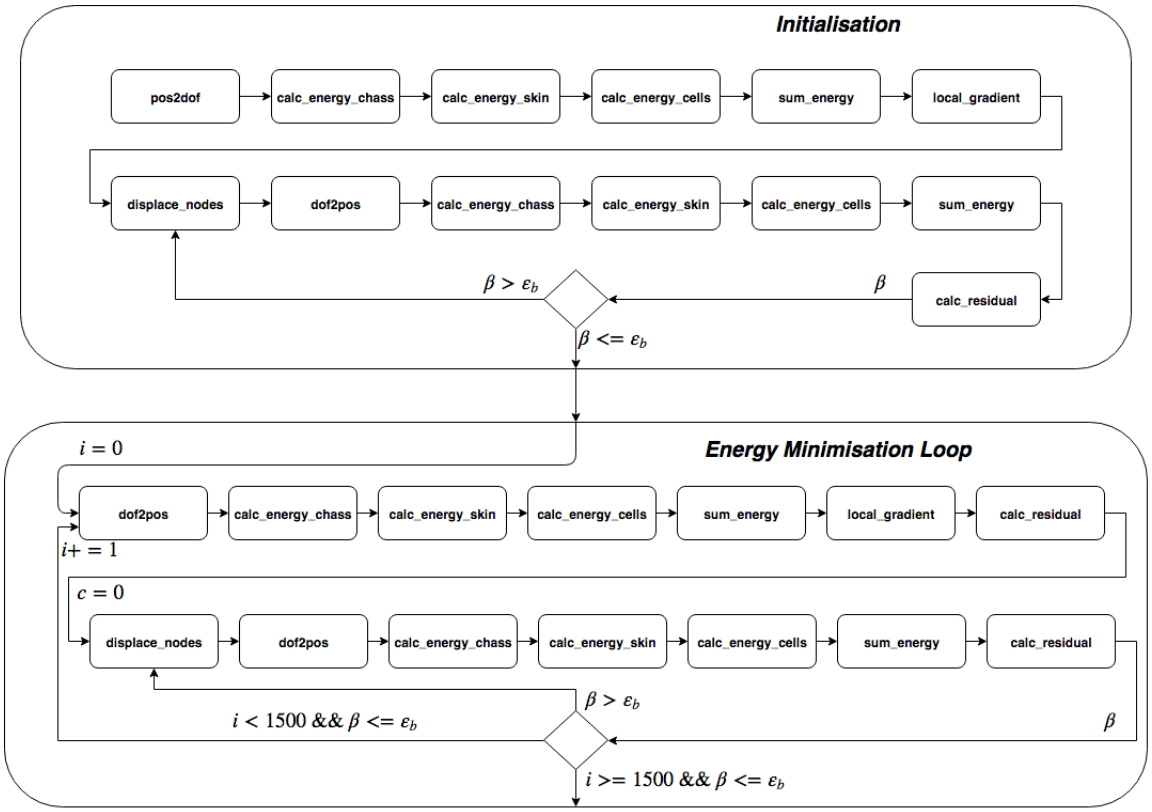



Figure 42: Nau3D simulation diagram

During the development of the GPU version of the breast simulation code, greater focus was given to one key aspect: ensuring that the values obtained with this version matched the values obtained with the CPU code architecture. One of the problems that appeared concerned synchronisation. The values that were being written in the buffers that store energy values ($Chass_{energy}$, $Skin_{energy}$ and $Cell_{energy}$) were incorrect and accumulating errors. To overcome this problem, the number of threads per thread group was set to only one, hence reducing the probability of occurring synchronisation errors. However, there was another problem in the code which was the lack of support by GLSL on some double precision operations which included the calculation of the natural logarithm of a number. The OpenGL Shading Language (GLSL) libraries have only functions that calculate the values of the natural logarithm for floats and not doubles. So, a function that was able to calculate the natural logarithm was implemented. This function used the Taylor series expansion to calculate the natural logarithm of a number $x$ between 0 and 2 and used a pre-calculated value for the natural logarithm of 2 which was then multiplied by the number of times that $x$ was higher than 2. After this solution was found the code was providing the same results as the CPU version of the code.

*Nau3D Solution Analysis*

A performance comparison between the GPU code and the CPU code was done. For that purpose a coarse mesh with only 226 nodes and the energy minimisation loop stop criteria was set to run 1500 iterations (just like in subsection 5.5.2). The time obtained with the first version of the code was of 93 seconds which is much superior to the any of the values reported in table 38 which includes simulations with meshes with 10 times more nodes. To find a reason for the poor performance of the code, an analysis of the CPU and GPU weight was done. The tests performed with the GPU code were done with CPU D.

| **Shader Weight (ms)** | **CPU** | **GPU** | **Total** |
|---|---|---|---|
| *calc_energy_chass* | 0,10 | 0,01 | 0,11 |
| *calc_energy_skin* | 0,16 | 1,06 | 1,22 |
| *calc_energy_cell* | 0,09 | 0,95 | 1,04 |
| *sum_energy* | 5,53 | 0,23 | 5,76 |
| *local_grad* | 0,02 | 9,31 | 9,33 |
| *dof2pos* | 0,02 | 0,01 | 0,03 |
| *displace_nodes* | 0,06 | 0,01 | 0,07 |
| *swap* | 0,30 | 0,00 | 0,30 |
| **Total** | 6.28 | 11.58 | 17.86 |
| **Weight (%)** | 35.16 | 64.84 | 100 |

Table 39: Simulation weight (in ms per frame) for each shader that runs during the simulation. The CPU column shows the time spent in the CPU by the shader. The GPU column shows the time spent in the GPU by the shader. The Total column shows the time spent by the shader during the entire simulation. The bottom rows show the total time spent per frame in CPU and GPU as well as the percentages.

Table 39 shows the relative weights of each shader in the CPU and GPU. The **swap** refers to the time spent by the pipeline execution of Nau3D to go from one frame to the next (in this case from the **calc_residual** shader to **pos2dof** during the strain energy minimisation loop). The heavier methods are the **sum_energy** which spends more time in the CPU and **local_grad** shaders which spends more time in the GPU. The **local_grad** shader is heavy and it is understandable that a large amount of time is spent there but the large amount of time spent by the **sum_energy** in the CPU is very strange because that time was spend in the Lua scripts attached to each shader. In fact, as the results show, approximately 35% of the time is spent in the CPU. One possible explanation is thread synchronisation, *i.e.* the Nau3D engine was registering the time the script was waiting after the first threads finished their work. To check if this was true, some of the value registration in global variables that were set in the the Nau3D project file were removed and passed to the shader $Global_{vars}$ array. The new time obtained after this modification was of 82 seconds which is still high but is better than the previous value. The value of the weights can be seen in table 40.

| Shader Weight (ms) | CPU | GPU | Total |
|---|---|---|---|
| *calc_energy_chass* | 0.05 | 0.01 | 0.06 |
| *calc_energy_skin* | 0.09 | 1.06 | 1.15 |
| *calc_energy_cell* | 0.07 | 0.95 | 1.02 |
| *sum_energy* | 0.11 | 5.63 | 5.74 |
| *local_grad* | 0.02 | 9.31 | 9.33 |
| *dof2pos* | 0.02 | 0.01 | 0.03 |
| *displace_nodes* | 0.03 | 0.01 | 0.04 |
| *swap* | 0.18 | 0,00 | 0.18 |
| **Total** | 0.57 | 16.98 | 17.55 |
| **Weight (%)** | 3.25 | 96.75 | 100 |

Table 40: Simulation weight (in ms per frame) for each shader that runs during the simulation after the first improvements. The CPU column shows the time spent in the CPU by the shader. The GPU column shows the time spent in the GPU by the shader. The Total column shows the time spent by the shader during the entire simulation. The bottom rows show the total time spent per frame in CPU and GPU as well as the percentages.

This table continues to show that **local_grad** is still the shader were most of the time is spent by the simulation. However, the CPU time has reduced significantly (only 3% when compared to the previous 35% presented in table 39) and the energy calculation methods (**calc_energy_skin** and **calc_energy_cell**) became more prevalent. These changes in the weight lead to look carefully into the three shaders and the common element between them is the calculation of the natural logarithm which is done by the Taylor series based function. Two considerations were done: or the implementation as flaws or the function is too heavy. After implementing the same function in the CPU and comparing the results in terms of time with the C++ math library, it was found that the function was working perfectly fine. However, this method is a convergence method and by a matter of curiosity the number of iterations necessary to find a precise result (with at least 15 significant numbers) differed in the CPU and in the GPU. In an attempt to solve this problem, the values of the natural logarithm between 0 and 2 were pre-computed. This originated a large array with 100 000 elements which occupied 800KB of memory. The problem with this amount of elements is that the final solution was producing inaccurate results. This inaccuracy only faded for 10 million pre-computed values of the natural logarithm. But, despite the 80MB of memory that the array occupied, the time spent by the simulation went from 82 seconds to only 11.51 seconds which despite being far from the best time of 0.67 seconds in the CPU is much better that 82 seconds.

With these improvements of the GPU code performance, the next step passed by analysing the codes performance with meshes of different sizes (namely the medium and thin mesh).

| Shader Weight (%) | Coarse (226 nodes) | Medium (1014 nodes) | Thin (2498 nodes) |
|---|---|---|---|
| *calc_energy_chass* | 13.11 | 5.22 | 2.16 |
| *calc_energy_skin* | 17.21 | 20.15 | 16.97 |
| *calc_energy_cell* | 14.75 | 16.79 | 20.47 |
| *sum_energy* | 12.30 | 20.15 | 21.80 |
| *local_grad* | 19.67 | 26.12 | 31.61 |
| *dof2pos* | 2.46 | 1.12 | 1.33 |
| *displace_nodes* | 2.46 | 2.99 | 1.33 |
| *swap* | 14.75 | 5.97 | 3.33 |
| **Time (s)** | 11.51 | 19.81 | 48.69 |

Table 41: Simulation weight for each shader that runs during the simulation for meshes with different number of nodes (coarse, medium and thin). The last row shows the total time in seconds spent by the simulation.

Table 41 shows the weight of each method for the different size meshes. Across every mesh size, the shaders that are more time consuming are the **calc_energy_skin**, **calc_energy_cell**, **sum_energy** and **local_grad**. The impact of the synchronisation of the **swap** is higher for the coarse mesh than for the thinner meshes because, for each mesh size, **swap** takes 20ms but the total time spent (per frame cycle) by the thin mesh is higher than the coarse mesh. The time values observed with the GPU code is closer to the values observed with the CPU code but can still improve. As mentioned previously the threads per thread group of each shader was set to only 1 for synchronisation purposes. This meant that for each shader, only 1 thread would process data which is inefficient. However with the code providing stable results a new attempt to boost the multi-threading capabilities of the GPU was made. New tests were made with the same conditions as the previous tests and the average of 5 tests is presented in table 42.

| Simulation Time (s) | Coarse (226 nodes) | Medium (1014 nodes) | Thin (2498 nodes) |
|---|---|---|---|
| ***1*** | 11.51 | 19.81 | 49.69 |
| ***2*** | 10.57 | 14.93 | 30.27 |
| ***4*** | 10.22 | 12.96 | 21.92 |
| ***8*** | 10.05 | 11.44 | 16.93 |
| ***16*** | 10.02 | 10.75 | 14.59 |
| ***32*** | 9.98 | 10.39 | 12.14 |
| ***64*** | 10.01 | 10.51 | 12.15 |

Table 42: Times of the simulations with different size meshes for a different number of threads perf thread group.

It is possible to see the decrease in time with the increase in the number of threads per group. The improvement is better seen in the thin mesh. In fact while the speedup obtained with the coarse and medium mesh are only of 1.15 and 1.91 times respectively, the speedup obtained with the thin mesh reaches 4.09 times which is very good. The difference in terms of time spent by the simulation for the different sized meshes is very close. Nonetheless, the times obtained with this version of the code are still worse than the ones obtained for the same machine (values of CPU D in table 38).

However a comparison between the evolution of the time spent by the simulation between the CPU code and the GPU code suggest that the scalability of the GPU version of the code is potentially better than the CPU version. With that in mind, new meshes with more nodes were generated in order to verify if at any point the GPU code version would outperform the CPU code version. Figure 43 shows a graph of 4 cases where the first three relate the coarse, medium and thin meshes mentioned previously and an additional mesh with 8449 nodes.

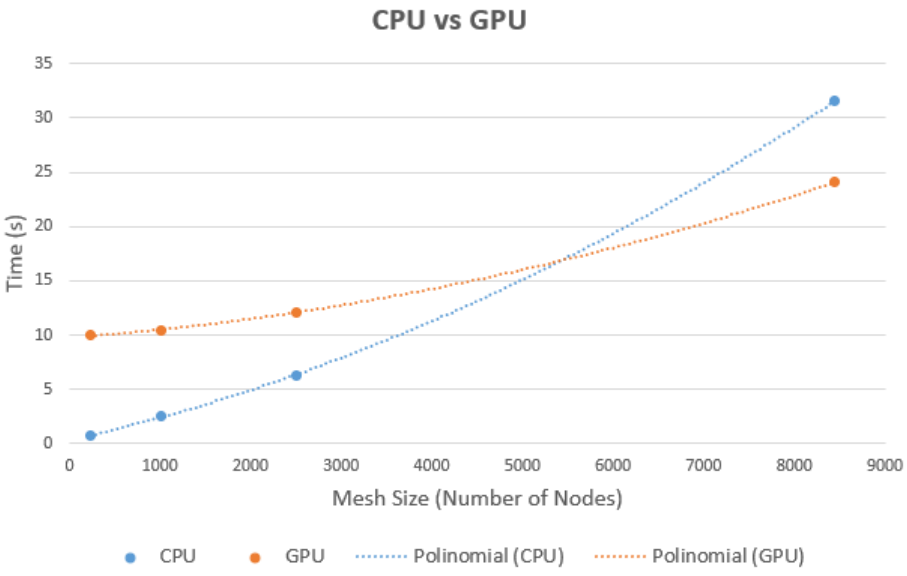



Figure 43: Graph which compares the time scalability of the CPU the GPU codes.

The graph of figure 43 shows, the tendency curves of the CPU (in blue) and GPU codes show that although the CPU code outperform the GPU code for the coarse, medium and thin meshes, it does not do so for the mesh with 8449 nodes were it takes approximately 32 seconds while the GPU code only takes 24 seconds. This means that for meshes with high number of nodes, the GPU is a better solution. However, for the case of breast simulations, the meshes used so far in the tests (coarse, medium and thin) are sufficient, hence, the best solution is the one provided by the CPU code architecture.

## 5.6 DISCUSSION

The optimisation of the code architecture was very successful. With the times of the new simulation it is possible to perform the estimation of parameters more quickly (10 minutes instead of 7 hours for a coarse mesh). The change of the data structures proved to have a great impact in the code. The exploit of the vectorisation units of the modern CPU showed improvements as well as the parallelisation of the code, with the latter having an higher impact in terms of overall performance. The implementation of a code in the GPU showed an alternative method whose results show a potential usage with more refined meshes in the future.

However, despite the great speedup obtained, the breast simulation code architecture is not good enough to provide a real-time response, specially for the process of estimating the breast parameters. In terms of the simulation performance, potential improvements could pass by studying a more efficient data structure or finding a more convergent algorithm for the strain energy minimisation loop. In terms of the breast parameter estimation, a different possibility could involve studying the Newton method that estimates the breast parameters and modify it so that the convergence improves. Another possibility that could be considered concerns using ML algorithms. This latter option is very interesting since the main goal of the breast parameter estimation method is to obtain a set of parameters from a pre-determined set of measurements. So, it is possible to train a model by giving it a large number of samples/cases which were made of the breast measurements and the matching parameters.

# 6

# MACHINE LEARNING MODEL

As presented in chapter 3, a numerical model of the breast can be a powerful tool to analyse surgical scenarios and support the medical staff in their decisions. Each patient has specific mechanical and geometrical parameters and their identification is of fundamental importance in order to provide relevant subject specific simulations. To this end, an inverse non-linear elasticity problem solver capable of estimating the bio-mechanical parameters of the breast was proposed in chapter 4. This method is iterative and uses measurements of the breast (that a doctor can perform during consultations) to estimate those parameters.

The main issue with this method is that it it can take several minutes to hours to find a good solution depending on the required accuracy and the detail of the mesh. In chapter 5 a strategy was presented to optimise the method's performance by altering the data structures, implementing more efficient algorithms and taking advantage of the code's parallel nature. Nevertheless, the performance gain, although very significant, was not enough to reduce the estimation time to be considered feasible in a consultation scenario. So, considering the nature of the problem, a different approach is required to obtain the breast parameters in a closer to real time solution.

ML algorithms and models proved over the years that they can solve complex problems (Mitra and Balaji (2010); Huber et al. (2011); Biswas and Acharyya (2014); Saniei et al. (2016); Arbabi et al. (2016); Vidaki et al. (2017); Tonutti et al. (2017); Thirumalai and Manzoor (2017)) and despite needing large amounts of time to train, they can provide results almost instantaneously (Goodfellow et al. (2016)). Taking into account the nature of the breast parameter estimation problem, the focus is in training a nonlinear regression model that given a set of measurements is able to estimate the breast parameters.

## 6.1 REGRESSION MODELS REVIEW

The breast parameter estimation problem can be characterised by non-linear relations between the measurements (inputs) and the parameters (outputs). The breast model presented in chapter 3 can be used to generate many cases which can be used to train a ML

model. The focus is in exploring the best choice to develop a regression model that can perform the same task as the iterative method for parameter estimation. Among regression models the most common include linear models, Support Vector Machines (SVM) and NN (Ayodele (2010); Shalev-Shwartz and Ben-David (2014); Dey (2016); Goodfellow et al. (2016); Bonaccorso (2017)).

### 6.1.1 *Linear Regression*

Linear regression model is one of the most common models and the training process is done by minimising the value of the cost function which is based on the Mean Squared Error (MSE) of the outputs (Goodfellow et al. (2016); Dey (2016); Bonaccorso (2017)). The more recent works on linear regression models include Mohammadi et al. (2017); van der Zijden et al. (2017); Seligman et al. (2018).

van der Zijden et al. (2017) presented a multi linear regression model that used kinematic input variables to estimate the severity of sideways fall. To do that, they have used a system of cameras that captured the movements of the subjects falling. The inputs are given by the analysis of the many frames like the peak hip impact velocity and the shoulder angle on impact as well as other measurable inputs like the body mass. They use these inputs to estimate the hip impact forces (output). The results presented, according to van der Zijden et al. (2017), showed an acceptable accuracy in estimating the fall severity (hip impact forces) but the results also expose variances from 46% to 63% for the validation and training datasets respectively. They explained that the inputs used depend on the motions of persons which might not act equally every time, hence, such variances were to be expected.

In Seligman et al. (2018), a study is presented were the authors analyse the potential uses of ML models (including linear regression models) to estimate biological markers of chronic disease risk: systolic blood pressure, body mass index, waist circumference and telomere length. To estimate these values (output) they use as input social values regarding gender, age, income, education, health behaviours, and many others. They compared the results of the trained linear regression models with the ones obtained with minimal and theory-driven models. They have found that using the linear regression models were no better than theory-driven models. They suggested that the social causation of health is more complex than linear models can capture (Seligman et al. (2018)).

But these models are not only used in medical contexts, or as a stand alone solution. The work presented by Mohammadi et al. (2017), shows a combined approach of dictionary learning and linear regression models for estimating the intensity of facial action units (AU). These AU (33 according to Mohammadi et al. (2017)) can have different intensities (from 0 to 5) and can be used to represent or evaluate basic expressions like joy or fear. So, this type of work can have many applications in fields like psychology. Their approach is to use

the dictionary learning model to create a sparse representation of the feature vectors that contain the AU and then use the linear regression model to estimate the values of intensity of each one of the AU present in the feature vectors (Mohammadi et al. (2017)). They tested the efficiency of their proposed method with others basic approaches. The results showed that the usage of the dictionary learning model together with the linear regression model were capable of outperforming the other AU intensity estimation methods.

### 6.1.2 *Support Vector Machine*

SVM are algorithms that can be used for regression and classification tasks working with linear and non-linear scenarios (Goodfellow et al. (2016); Bonaccorso (2017)). The way this type of algorithms work is by finding a hyper-plane which separates the elements per class (Dey (2016); Goodfellow et al. (2016); Bonaccorso (2017)). These hyper-planes have associated margins in order to reduce the potential of errors (Goodfellow et al. (2016)). The SVM algorithm adapts the weights of the model in a way that it finds the better option for the hyper-planes that separate the elements of different classes. Some examples of the usage of SVM methods, more concretely Support Vector Regression (SVR) models, in practical applications were presented in Huber et al. (2011); Mahmoodian and Ebrahimian (2016); Tonutti et al. (2017).

Huber et al. (2011) presented a new method that employs a SVR model that is used to predict the yield stress (biomechanical strength) of trabecular bone from features extracted of MRI data. The evaluation of this property can be important to access the risk of fractures because the variations in bone density (small values increase the risk of fracture) are related to variations in bone strength (Huber et al. (2011)). The data was gathered from MRI images of the proximal tibia and the scaling index method was used to extract the features of that image (local geometric properties) that are in turn the input values of the SVR model. The model uses the features to estimate the value of yield stress. The performance of this method is compared with a multi-regression analysis model as well as more common model used in these types of problems called bone volume fraction. The usage of the SVR model together with the feature extraction method lead to a reduction of the prediction error by approximately 5% when compared to the other methods.

Tonutti et al. (2017) presented a new method to predict the recurrence of breast cancer based on gene expression profile. The proposed method uses SVR together with fuzzy systems to analyse the problem's features (the gene expression values of the gene expression profiles) and predict the time that will take for the cancer to relapse. The problem's data consists in multiple cases of cancer relapse and they were divided into different groups according to the relapse risk (very high risk group, high risk group, low risk group and very low risk group - Tonutti et al. (2017)). For each group they train a SVR model that uses

the gene expression values to evaluate the relapse time. According to Tonutti et al. (2017), the results show improvements when compared to methods proposed in other works.

The usage of SVR in dealing with complex problems was also presented in Mahmoodian and Ebrahimian (2016). They propose a new model that uses a SVR to determine the deformation of tissues during neurosurgical procedures in real-time. For that purpose, they model a brain from MRI data and insert a tumour using geometrical operators. Then, they perform multiple simulations of surgeries considering the forces (magnitude and direction) that are applied, during such type of procedure, on the mesh nodes and they register the deformations that occur. These forces are analysed with the prescription of load nodes on the surface of the brain. They then proceeded to analyse the results of the deformations and extracted the important features to analyse the tissue deformation like the components of the force vector and the distance between the load nodes and the tumour nodes. This data is then used to train the SVR model which can then estimate the deformation of the tissue, *i.e.* the displacement of the nodes of the brain mesh due to the application of forces (Mahmoodian and Ebrahimian (2016)). The results presented by the trained SVR model show an average error of 0.1 to 0.2 mm for the magnitude of the deformations which was considered very good by the authors (Mahmoodian and Ebrahimian (2016)).

### 6.1.3 *Neural networks*

A NN is a deep learning method that operates in a manner inspired by the natural neural network in the brain (Shalev-Shwartz and Ben-David (2014); LeCun et al. (2015); Goodfellow et al. (2016); Bonaccorso (2017)). A NN is composed by a large number of processing elements, also known as neurons, that are connected to each other and that are generally organised in multiple layers. The network learns through experience of many cases where it adjusts the values of the weights (associated to the neurons and the connections) in order to improve its performance (Goodfellow et al. (2016); Bonaccorso (2017)). The information is processed and communicated between different nodes, *i.e.* a neuron input comes from other neuron outputs with each one of them being characterised by a connection weight. These values are then combined and the result is filtered by a function (a.k.a. activation function) which produces the neurons output (Shalev-Shwartz and Ben-David (2014); Goodfellow et al. (2016); Bonaccorso (2017)). There are different types of NN that can be organised in multiple layers, like Multi-Layer Perceptron (MLP) where the neurons from one layer receive the output from the previous layer (Goodfellow et al. (2016)). The many layers of a NN have different purposes: the input layer receives the raw data, the hidden layer(s) (which can be one or more) can be used to extract features from the problem and the output layer combines the values of the last hidden layer and returns the final results (Dey (2016); Bonaccorso (2017)). The performance of a NN depends of its architecture, *i.e.*, the

number of hidden layers, nodes, type of activation functions used and the optimiser (Duchi et al. (2011); Tieleman and Hinton (2012); Kingma and Ba (2014); Goodfellow et al. (2016)). There is a vast amount of work concerning the usage of NN, including the ones presented by Mitra and Balaji (2010); Biswas and Acharyya (2014); Saniei et al. (2016); Arbabi et al. (2016); Vidaki et al. (2017)).

Just like linear models and SVR models, there is also a volume of work where the usage of NN is analysed. In Mitra and Balaji (2010), a NN model is used to estimate the tumour parameters from a thermal image of the breast. In case of a cancerous breast, the heat generation inside the tumour alters the normal temperature profile of the breast (Mitra and Balaji (2010)). To train a model using the NN, the authors generated the thermal images artificially using a method that solves bio-heat transfer equations together with a simple model of the breast shaped as an hemisphere. They then insert a tumour shaped as a sphere in a a location which is considered as a variable. They perform several simulations and provide the NN with the thermal images. A NN with two hidden layers (15-4) is used to train a model that receiving a thermal image of the breast with the objective of determining the position and radius of the spherical tumour. The results show that the NN model was able to estimate the tumours position with an error close to 10% and its radius with an error close to 5%.

Within the same subject, Saniei et al. (2016) also presented a model to estimate the parameters of tumours (position and size) using thermal images of the breast. Just like in Mitra and Balaji (2010), the authors generate a model of the breast solving the bio-heat transfer equations and use the values to train the NN. The main difference of this approach is in the usage of a dynamic NN where they delay the outputs of a layer to the other (Saniei et al. (2016)). The results obtained with this approach show error of the position and size of the tumour in the order of 0.31cm and 0.2cm respectively which is considered good. Both presented models (Mitra and Balaji (2010) and Saniei et al. (2016)) showed promising results in estimating parameters from thermal images.

The usage of NN to estimate the properties of tissues was also presented in Arbabi et al. (2016). The work presented in this article shows the usage of NN models to estimate the diffusion coefficients of cartilage. In the considered problem, the cartilage is modelled using a neo-Hookean material with the definition of different diffusion coefficients on the different zones of the cartilage model and by simulating the transport of Iodixonal solutions within a certain time frame (Arbabi et al. (2016)). Different diffusion coefficients can lead to a different transport pattern of the Iodixonal solutions. So, they create many variations of the cartilage with different diffusion coefficients and simulate the process. The product of these simulations are, what they called, concentration-time curves. These curves are the input to the NN model which trains with the objective of determining the different diffusion coefficients (Arbabi et al. (2016)). The results of the estimation of the diffusion

coefficients from the concentration-time curves show a fit of the results with a correlation coefficients of $R = 0.99992$.

## 6.2 NETWORK ARCHITECTURE AND TRAINING

Considering the problem of estimating the breast bio-mechanical parameters, the inputs consists in the measurements of the breast and the outputs are the associated parameters. Hence, one can use this association between the measurements and the parameters to create a model with the purpose of performing the same task as the iterative method proposed in chapter 4.

Considering the nature of the problem, the non-linear relations between the measurements and the parameters and the literature review on ML algorithms, the choice was to use a NN to train a regression model. The reason behind this choice is the higher number of works with NN concerning the problem of estimating the parameters of tissues which indicate that this type of structure is better suited to solve the breast parameter estimation problem. Moreover, this is supported by the fact that, according to Goodfellow et al. (2016), NN possess a high capacity of extracting complex patterns from non-linear data. So, NN is better suited than linear models (too simple) and SVM (can be seen as a MLP (a type of NN) with only one hidden layer - Goodfellow et al. (2016)).

### 6.2.1 *Datasets generation*

To train, validate and test the Neural Network (NN), a large dataset is created with the numerical breast model proposed in chapter 3 carrying out several thousands of numerical simulations with a wide range of variation of coefficients $\Lambda$.

From the entire dataset 500 cases are taken for $\mathcal{D}^{test}$. From the remaining 40000 cases, 90% of the dataset is used to train ($\mathcal{D}^{train}$) and 10% for validating ($\mathcal{D}^{val}$).

In the numerical experiments, one takes $\Lambda^{user} = \{R = 5,\ H = 5.64,\ \lambda_{br} = 1000,\ \lambda_{sk} = 8000,\ \mu_{br} = 400$ and $\mu_{sk} = 1600\}$ and assume these values to represent the mean values for each parameter. At the numerical level, the data-set is generated for a given mesh with a certain granularity. Consequently, as figure 44 shows, the number of vertices in a breast mesh can affect the quality of the result, *i.e.* breast meshes with more vertices (thin mesh) present a better visual match to a real breast than meshes with less vertices (coarse mesh).

To assess the impact of the mesh size, three data-sets were generated ($\mathcal{D}_{coarse}$, $\mathcal{D}_{medium}$ and $\mathcal{D}_{thin}$) with $N = 40500$ elements for each one of the meshes with 226 nodes, 1014 and 2498 vertices (respectively).

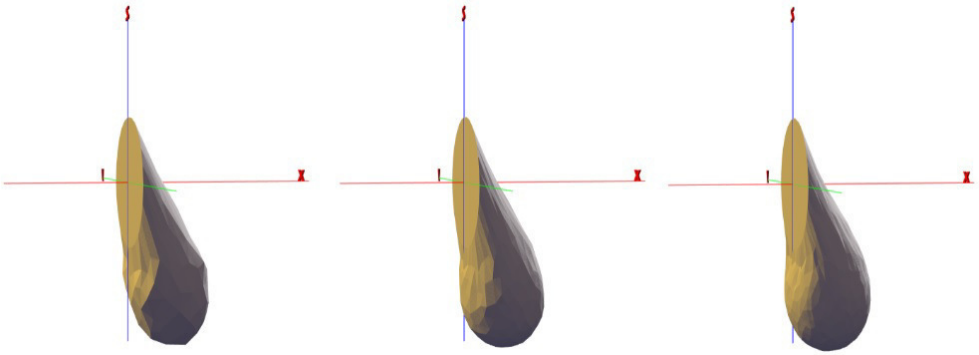


Figure 44: Image of 3 breast meshes generated by the same parameters: coarse (left), thin (mid) and ultra-thin (right)

The data-set of the coarse mesh $\mathcal{D}_{coarse}$ was achieved within a week, whereas the data-sets for the medium breast mesh $\mathcal{D}_{medium}$ and thin breast mesh $\mathcal{D}_{thin}$ required around three and six weeks respectively.

To generate the datasets, a Gaussian distribution of the parameters centred on the user parameters $\Lambda^{user}$ is used. A data-set is created as follows.

1. One generates $N$ vectors $s^n \in \mathbb{R}^6$, $n = 1, \cdots, N$ such that each component $s_i^n$ is chosen randomly with the Normal law $\mathcal{N}(0, 1/2)$. Then one sets $\Lambda_i^n = (1 + s_i^n)\Lambda_i^{user}$, $i = 1, \cdots, 6$, $n = 1, \cdots, N$, eliminating the non-physical situations (negative values).

2. One uses the geometric parameters $\Lambda_g^n = (\Lambda_1^n, \Lambda_2^n)$ to define the free stress domain $\Omega_g^n$ and solves the nonlinear problem with the mechanical parameters $\Lambda_m^n = (\Lambda_3^n, \Lambda_4^n, \Lambda_5^n, \Lambda_6^n)$ that provides the displacement field $f^n$ for the 3 positions and the resulting shape $\Omega(\Lambda^n) = f^n(\Omega_g^n)$ as detailed in chapter 4.

3. One computes the volume, surface and size measures $M^n \in \mathbb{R}^{15}$ corresponding to the parameters $\Lambda^n$ and creates the element $D^n = \{M^n, \Lambda^n\}$. The dataset $\mathcal{D}$ gathers all the elements that are normalised by setting

$$M_j^n := \frac{M_j^n}{\overline{M}_j}, \ j = 1, \cdots, 15,$$
$$\Lambda_i^n := \frac{\Lambda_i^n}{\overline{\Lambda}_i}, \ i = 1, \cdots, 6,$$

   where $\overline{M}_j$ and $\overline{\Lambda}_i$ stand for the mean values over the data-set.

4. The data-set is split in three parts, namely $\mathcal{D}^{train}$ and $\mathcal{D}^{val}$ for the out-samples error used during the training stage as a criterion to assess the NN quality and $\mathcal{D}^{test}$ for testing after the training.

### 6.2.2 *Neural Network architecture*

A NN model is trained to emulate the inverse problem solver, *i.e.* to provide an approximation of $\Lambda$ in function of measurements $M$. The NN architecture depends on the nodes structure, the activation functions and on the solver that fits the NN parameters with respect to the training data set. An input layer $h^0$ (measurements), an output layer $h^K$ (parameters) and several hidden layers $h^k$,$k = 1, \cdots, K-1$ constitute the network $h$. Knowing that the model receives 15 measures as input and returns 6 parameters as outputs, one builds a parametric operator

$$h^0 = M \rightarrow \varphi(M, w) = h^K = \Lambda$$

and, for a given set $\mathcal{D} = (M^n, \Lambda^n)_{n=1,\cdots,N}$, one defines the cost function $\varepsilon_i$ for each parameter $\Lambda_i$ as

$$\varepsilon_i(w) = \frac{1}{N} \sum_{n=1}^{N} |\Lambda_i^n - \varphi_i(M^n, w)|, \; i = 1, \cdots, 6. \tag{66}$$

One then determines the weights tensor $w$ that minimise the total loss function given by

$$w \rightarrow \varepsilon(w) = \sum_{i=1}^{6} \varepsilon_i(w)$$

using the training set $\mathcal{D}_{train}$. After a successful training, one gets the training loss $\varepsilon_i^{train}$ (in-samples error) and the validation loss $\varepsilon_i^{val}$ (out-samples error) for each parameter $\Lambda_i$. Notice that the loss functions are relative since the data-set is normalised and the error values shall be reported in terms of %. Lesser amounts of loss means that the results are more accurate.

The objective of the training process is to determine matrix $w$ that minimises the value of loss function $\varepsilon$ over a training data-set. After training one obtains a model capable of, given a set of reference measurements $\overline{M}$, estimate the set of parameters $\Lambda$ that best correlates with said measurements. The most close to the reference parameters the better the result.

Google's Tensorflow with Keras is used to implement the solution of training, evaluating and testing (Abadi et al. (2015); Chollet et al. (2015)). The training ends whether a prescribed minimum error over the validation set is reached, or the training as performed up to 3000 epochs, *i.e.* running over, at least, 3000 times the whole training set. After the training, one obtains the tensor $w$ that can be employed in the simulations. For the sake of statistical validation, one starts again the leaning process with the same training data-set to provide a second guess for tensor $w$. This operation is repeated in order to provide five tensors $w$ and then one performs all the tests independently with each calculated tensor. At last, the average values of the errors obtained with the five independent simulations are report in tables.

The NN set-up depends on the hidden layers, their size, and the activation function choice. On the other hand, the optimiser *i.e.*, the routine that provides an approximation of weights tensor $w$, is also an important issue for the sake of efficiency. Several experiences were carried out to determine an efficient configuration with respect to the different data-sets. Different degrees of freedom have been explored, namely:

- the hidden layers configuration where multiple hidden layers have been considered together with a number of nodes per layer from 16 to 1024 nodes;
- the activation functions such as exponential linear unit (elu) and the rectified linear unit (relu) functions;
- the optimisers provided by the Keras library were the RMSprop, Adagrad and Adam optimiser.

### 6.2.3 *Hidden layers*

The first step consisted in evaluating the impact of the hidden layer configuration. For that purpose the elu activation function and the RMSProp optimiser were used with multiple hidden layer configurations (different number of layers and different number of nodes per layer) and observed the training and validation results. Table 43 gives the percentages of the in-sample and out-sample errors for five of the configurations performed in the experiments, varying the number of layers and nodes.

The results show that the neural network benefits from having a larger number of hidden layers. Using only one hidden layer it was observed that the best result was obtained with 64 nodes and the average error per parameter round the 10%. But with more hidden layers it is possible to observe that the average error per parameter improved to 5% using 5 hidden layer and 3.6%, 3.5% and 3.55% for the cases with 11, 12 and 13 hidden layers respectively. For configurations with multiple hidden layers it was found that the optimal number of nodes was of 128. The NN is capable of learning accurately the values of the geometrical parameters ($R$ and $H$) as well as the mechanical parameters of the glandular and fat tissue ($\lambda_{br}$ and $\mu_{br}$). The mechanical parameters of the breast's skin are more difficult, specially $\mu_{sk}$ with errors rounding 11% for configurations with 11, 12 and 13 layers. Learning the $\lambda_{sk}$ parameter on the other hand becomes easier with the increase of hidden layers going from an approximate value of error of 16.5% with one layer to 3.8% with 12 hidden layers. It was found that the best configuration (in the tests performed) with an average error per parameter of 3.5% using the validation dataset is the one with 12 layers with 128 nodes in each layer.

| **Hidden Layers** | **Nodes per Layer** | **Stage** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **Avg Total** |
|---|---|---|---|---|---|---|---|---|---|
| **1** | **64** | ***train*** | 3.95 | 6.28 | 8.81 | 16.78 | 2.99 | 24.01 | 10.47 |
| | | ***val*** | 3.92 | 6.20 | 8.56 | 16.59 | 3.03 | 24.28 | 10.43 |
| **5** | **128** | ***train*** | 1.05 | 1.87 | 2.89 | 5.66 | 1.76 | 16.61 | 4.97 |
| | | ***val*** | 1.05 | 1.89 | 2.96 | 5.99 | 1.74 | 16.78 | 5.07 |
| **11** | **128** | ***train*** | 0.78 | 1.10 | 2.24 | 3.60 | 1.35 | 11.46 | 3.42 |
| | | ***val*** | 0.78 | 1.14 | 2.37 | 4.00 | 1.40 | 12.05 | 3.63 |
| **12** | **128** | ***train*** | 0.73 | 1.08 | 2.19 | 3.53 | 1.28 | 11.02 | 3.31 |
| | | ***val*** | 0.76 | 1.09 | 2.31 | 3.81 | 1.35 | 11.70 | 3.50 |
| **13** | **128** | ***train*** | 0.75 | 1.08 | 2.21 | 3.51 | 1.29 | 11.02 | 3.31 |
| | | ***val*** | 0.76 | 1.20 | 2.32 | 3.80 | 1.37 | 11.86 | 3.55 |

Table 43: Values of loss in percentage for the model training data-set (train) and validation data-set (val). These values represent the mean values of error for 5 runs on training and validating running up to 3000 epochs each. The left columns shows the number of layers, and the number of nodes per layer. For each hidden layer structure, it is shown the difference between the estimated parameters and the reference parameters in relation to the training and validation stages. The last column shows the average difference per parameter.

### 6.2.4 *Activation functions and optimiser*

Additional tests were performed to determine the activation function and optimiser that provide an efficient fitting with the breast parameter estimation problem. Five tests were performed for each different combination of activation function and optimiser with 3000 epochs. The average errors of the different estimated parameters are reported in table 44.

Training the neural network with the RMSProp optimiser and the elu activation function gives the lowest errors per parameter. RMSProp solver combining with the elu activation function achieves a 3.50% average error per parameter with the validation data-set whereas the other combinations have values larger than 4.64%. The main difference relies on a better evaluation of the $\mu_{sk}$ parameter which significantly improves the accuracy. Due to these results the remaining experiments were done using a NN architecture constituted of twelve hidden layers with 128 nodes each using activation function elu for all layers and the RMSProp optimiser.

| Error per Parameter (%) | | | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | Avg Total |
|---|---|---|---|---|---|---|---|---|---|
| **RMSProp** | **elu** | ***train*** | 0.73 | 1.08 | 2.19 | 3.53 | 1.28 | 11.02 | 3.31 |
| | | ***val*** | 0.76 | 1.09 | 2.31 | 3.81 | 1.35 | 11.70 | 3.50 |
| | **relu** | ***train*** | 1.12 | 1.31 | 2.71 | 4.94 | 2.39 | 23.36 | 5.97 |
| | | ***val*** | 1.16 | 1.41 | 3.01 | 5.56 | 2.48 | 23.95 | 6.26 |
| **Adam** | **elu** | ***train*** | 1.60 | 1.70 | 3.52 | 4.72 | 1.89 | 13.42 | 4.47 |
| | | ***val*** | 1.34 | 1.49 | 4.27 | 5.93 | 1.43 | 15.18 | 4.94 |
| | **relu** | ***train*** | 1.26 | 1.04 | 2.48 | 3.22 | 1.87 | 16.36 | 4.37 |
| | | ***val*** | 2.11 | 1.38 | 4.93 | 3.99 | 2.03 | 18.66 | 5.52 |
| **Adagrad** | **elu** | ***train*** | 1.05 | 2.54 | 2.78 | 5.36 | 1.63 | 14.50 | 4.64 |
| | | ***val*** | 1.06 | 2.60 | 2.79 | 5.51 | 1.53 | 14.31 | 4.64 |
| | **relu** | ***train*** | 0.94 | 1.08 | 2.18 | 3.70 | 1.87 | 22.80 | 5.43 |
| | | ***val*** | 1.02 | 1.32 | 2.31 | 4.53 | 1.95 | 23.35 | 5.75 |

Table 44: Values of error in percentage for the model training data-set (train) and validation data-set (val). These values represent the mean values of error for 5 runs on training and validating up to 3000 epochs each. The left column shows the different optimisers and activation functions. For each combination it is shown the error of the training and validation related to each parameter and for the average total (sum of all values of error and divided by the number of parameters).

## 6.3 TESTING

After finding a NN configuration, the next step consists in testing its performance. To do that, one evaluates the impact of the mesh size on the accuracy of the estimations as well as performs an assessment of the errors obtained.

### 6.3.1 *Mesh size impact*

Since an exact solution is not possible due to the usage of meshes instead of the real breast measurements, the issue of accuracy impact on the NN is addressed. In other words, how the mesh granularity influence the training and the quality of the prediction. To this end, three additional series of tests were carried out using three different sized meshes: the coarse (226 nodes), the medium (1014 nodes) and the thin (2498 nodes). Three data-sets ($D_{coarse}$, $D_{medium}$ and $D_{thin}$) were built and for each one, a total of 5 training runs (under

the same conditions mentioned previously in this section) were performed. In table 45, the mean values of loss functions for the 3 size meshes are reported.

| **Error per Parameter (%)** | | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **Avg Total** |
|---|---|---|---|---|---|---|---|---|
| **Coarse** | ***train*** | 0.73 | 1.08 | 2.19 | 3.53 | 1.28 | 11.02 | 3.31 |
| | ***val*** | 0.76 | 1.09 | 2.31 | 3.81 | 1.35 | 11.70 | 3.50 |
| **Medium** | ***train*** | 0.82 | 1.35 | 2.15 | 2.98 | 0.98 | 8.38 | 2.78 |
| | ***val*** | 0.86 | 1.40 | 2.20 | 3.04 | 0.99 | 8.61 | 2.85 |
| **Thin** | ***train*** | 1.24 | 3.00 | 3.97 | 3.80 | 1.18 | 9.40 | 3.77 |
| | ***val*** | 1.23 | 3.11 | 3.94 | 4.11 | 1.22 | 9.54 | 3.86 |

Table 45: Values of loss for the model training and validating. These values represent the mean values of error for 5 runs on training and validating up to 3000 epochs each. On the left column there are the different size meshes. For each breast mesh, it is shown the loss of the training and validating related to each parameter and for the mean total (sum of all values of loss and divided by the number of parameters).

The error from the training and validating sets vary with the granularity of the mesh. It is possible to observe that the validation error obtained with the medium mesh (2.85% per parameter) is smaller than the coarse and thin meshes (with 3.5% and 3.86% error per parameter respectively). Overall, the results show a good approximation to the real results for most of the parameters with the exception of $\mu_{sk}$. The error obtained with most of the parameters with any type of mesh size, is found to be equal or lower than 3%, which taking into account the results obtained with the iterative method are very good. The higher value of error of $\mu_{sk}$ being, in the best case, above 8.5% suggests that such parameter is not well-controlled by the measurements that were chosen to solve the inverse problem. Indeed, even at the numerical level, *i.e.*, using the iterative inverse solver, it is hard to obtain a good approximation value (relative error larger than 5%). Hence the data-set is actually not enough accurate for this last parameter and would hardly deliver correct estimates.

It could be expected that the thinner the mesh, the better the approximation, but as the values in table 45 show, the number of vertices of the mesh can have different impacts on certain parameters. The coarse mesh presents smaller values of error for the geometrical parameters than the other meshes while, the medium and thin meshes present better approximations to the mechanical parameters, specially the medium mesh. This can be explained by the nature of the parameters. A coarser mesh, as figure 44 shows, is more geometrical and therefore the impact of the geometrical parameters can have a larger impact on the final results. On the other hand the mechanical parameters need more definition and therefore an increase of the mesh vertices can help determining their values. A possible

explanation for the global poorer performance of the thin mesh in relation to the others, is that due to the increase in the number of vertices of the geometrical parameters are harder to obtain and that can also affect the determination of the mechanical parameters, because in the end the measurements are obtained by the combination of all the parameters. In other words, with meshes with small number of vertices, one does not have enough detail to obtain a good approximation to the mechanical parameters, and with too many details one can have too much detail which in turn can lead to the existence of local minimums and therefore lead to worse approximations. Therefore, a possible optimal mesh size might depend on a number of vertices that balance the geometrical aspects of the breast and the detail for the mechanical parameters.

### 6.3.2 *Error assessments*

For the same $\overline{\Lambda^n}$, three different evaluations of $\overline{M^n}$ were produced depending of the mesh refinement. Hence, one gets three different data-sets for test purpose, namely $\mathcal{D}^{test}_{coarse}$, $\mathcal{D}^{test}_{medium}$ and $\mathcal{D}^{test}_{thin}$.

From $(\overline{\Lambda^n}, \overline{M^n}) \in \mathcal{D}^{test}_{coarse}$, one computes an approximation $\overline{M^n} \rightarrow \Lambda^n$ using the NN training with the data-set of $\mathcal{D}^{train}_{coarse}$. The procedure is repeated for the medium mesh and the thin mesh for evaluating the error deriving from the discretisation procedure. It is important to remember that the components $\Lambda$ and $M$ have been normalised with respect to the mean value in order to provide fair balanced between very different measures or parameters.

In conclusion, one has to assess the error given by $\overline{\Lambda^n} - \Lambda^n$ using a dataset that was not used during the training process. In table 46 the mean values of the error for parameters $\Lambda_i$ are reported, namely

$$e_i = \frac{1}{P} \sum_{p=1}^{P} |\overline{\Lambda^p}_i - \Lambda^p_i|, \; i = 1, \cdots, 6$$

that are indicated in % while the total mean error is obtained by summing the $e_j$. The experience is reproduced with the two other meshes corresponding to the second (medium) and third (thin) row of the table.

It is possible to observe an overall good performance in terms of accuracy indicated by the average error per parameter (3.51% with a coarse mesh, 2.86% error with a medium mesh and 3.89% with a thin mesh). The values of table 46 show similar values to the ones obtained with the validation sets in table 45 which confirms that the dataset used is representative. The values of error for the geometrical parameters with the exception of $\mu_{sk}$ and the values of $\lambda_{br}$ and $\lambda_{sk}$ with the thin mesh, continue to present good approximations (majority of the values under 3% error). These results reinforce the idea that the measurements used

| **Error per Parameter (%)** | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | **Avg Total** |
|---|---|---|---|---|---|---|---|
| **Coarse** | 0.74 | 1.13 | 2.36 | 3.76 | 1.33 | 11.72 | 3.51 |
| **Medium** | 0.83 | 1.39 | 2.23 | 3.04 | 1.01 | 8.68 | 2.86 |
| **Thin** | 1.27 | 3.11 | 4.06 | 4.11 | 1.15 | 9.65 | 3.89 |

Table 46: Mean value error in % (normalised data) between the reference parameters and the estimated parameters by the neural network model.

to estimate the parameters are not well suited for $\mu_{sk}$ (values of error above 8.5% for any size mesh) and that there is a relation between the number of vertices of a mesh and the accuracy of the result in terms of the geometrical and mechanical parameters.

When one compares these values to the ones reported in tables 6, 7 and 8 of section 4.4.3, it is possible to observe that, in average, the approximations obtained by the NN models are slightly better.

There is however, some cases were the estimated parameters are significantly different from the reference values. But it is difficult to assess, just by looking into these values, that a set of parameters is inaccurate. Because in practical cases, one receives the measurements of the breast from a surgeon but not the parameters. So, a different way of evaluating the accuracy of the predicted parameters is needed. To do that one compares the reference measurements $\overline{M}$ with the measurements obtained with the estimated parameters $M(\Lambda)$ (obtained with the breast model presented in chapter 3) using

$$E(\overline{M}, \Lambda) = \sum_{i=1}^{15} k_i \Big( \overline{M}_i - M_i(\Lambda) \Big)^2. \tag{67}$$

where $k_i$ represents the relative weight of each measurement (the values of volume are much smaller than the values of $min_y$ and so they are balanced using weights). It is considered that a set of parameters is accurate when the value returned by the cost function in equation 67 is lower than a certain threshold $\varepsilon_e = 5e^{-6}$ which ensures that, for example, the difference between the $min_y$ of $\overline{M}$ and $M(\Lambda)$ is inferior to one millimetre. In the tests made with the iterative method it was found that $\varepsilon_e$ can be met with a set of parameters $\Lambda$ which differs from the reference values $\overline{\Lambda}$ about $4-5\%$ per parameter or even more depending on the error accumulation per parameter (in the case of the $\mu_{sk}$ parameter that difference can even be higher than 12%).

To assess the impact of the parameter errors on the breast shape, one evaluates the mean values of the error for parameters $M_j$, namely

$$e_j = \frac{1}{P}\sum_{p=1}^{P} |\overline{m^p}_j - m_j^p|,\ j = 1, \cdots, 15,$$

that are reported in table 47 in % (normalised data) for the three different meshes. It is also presented the value of error using the cost function (equation 67) to evaluate if the error in practical terms is significant or not (*i.e.* if the error is inferior to a millimetre or not).

| **Mesh** | $e_{m_1}$ | $e_{m_2}$ | $e_{m_3}$ | $e_{m_4}$ | $e_{m_5}$ | $e_{m_6}$ | $e_{m_7}$ | $e_{m_8}$ | $e_{m_9}$ |
|---|---|---|---|---|---|---|---|---|---|
| **Coarse** | 0.33 | 0.33 | 0.79 | 0.63 | 4.63 | 0.37 | 0.33 | 0.76 | 0.60 |
| **Medium** | 0.22 | 0.34 | 0.56 | 0.48 | 3.73 | 0.45 | 0.52 | 0.83 | 0.88 |
| **Thin** | 0.34 | 0.36 | 0.93 | 0.71 | 4.43 | 0.43 | 0.71 | 0.87 | 0.92 |
| **Mesh** | $e_{m_{10}}$ | $e_{m_{11}}$ | $e_{m_{12}}$ | $e_{m_{13}}$ | $e_{m_{14}}$ | $e_{m_{15}}$ | ***avg total*** | ***Cost Function*** | |
| **Coarse** | 7.49 | 0.33 | 0.28 | 0.52 | 0.52 | 6.08 | 3.10 | $1.65e^{-6}$ | |
| **Medium** | 5.16 | 0.32 | 0.44 | 0.55 | 0.63 | 5.31 | 2.78 | $1.31e^{-6}$ | |
| **Thin** | 7.23 | 0.29 | 0.53 | 0.54 | 0.66 | 5.37 | 3.30 | $1.84e^{-6}$ | |

Table 47: Mean value error in % between the reference and estimated measurements for each mesh. Last lower column shows the mean value of the cost function.

The average values of error per measure presented in table 47 show a good accuracy obtained in line with the error estimation of the parameters. The average error per measurement round the 3% which can be considered very good because it means that the estimated measurements are very close to the real values (differences inferior to millimetres). It is possible to distinguish two groups of measurements, the first one whose error value is under 1% whereas measures $m_5$, $m_{10}$ and $m_{15}$, corresponding to the same characteristic but in three different positions, present a larger dispersion (error above 3.7%). The average value of the cost function per mesh size is smaller than the $\varepsilon_e = 5e^{-6}$ ($1.65e^{-6}$, $1.31e^{-6}$ and $1.84e^{-6}$ for the coarse, medium and thin mesh respectively) which means that the vast majority of the test cases presented very good approximations.

The great majority of the parameters $\Lambda_i$ provided by the NN are excellent approximations that can be used with the breast model to determine the shape and behaviour of the breast. However, the NN model estimation can sometimes predict a set of parameters that are not accurate, *i.e.* the error in the measurements given by equation 67 was above the accuracy threshold ($5e^{-6}$) which in practical terms, represent errors in the order of centimetres. These

situations can happen with very particular cases or parameter combinations. In fact, in the 500 test cases considered for each mesh size (1500 cases total), it was found that this happened only 13 times. Nonetheless the existence of these cases introduce a necessity to ensure the quality of the parameter estimation by the NN model at every case.

### 6.3.3 *Estimation Robustness*

The performance results of the NN model presented in section 6.3.2 show a good accuracy with errors of 3.51%, 2.86% and 3.89% for every mesh size (coarse, medium and thin respectively). This means that this model can provide a good approximation to the breast parameters that determine the breast shape, size and mechanical properties. The iterative method uses a pre-determined set of parameters as the initial guess and then proceeds to refine them until the measurements generated from those parameters match the input measurements (generally given by the doctor).

A pre-determined initial guess has its limits because it only works for a certain range of parameters. So, whenever the reference parameters differs from this pre-determined initial guess by a substantial amount, then the iterative method suffers in terms of performance as it was observed in section 4.4.3. A better method consists in obtaining an initial guess that is close to the reference values. This is exactly what the NN trained models do.

However, it is important to mention that these tests were done with the NN models receiving the exact values of the measurements. In real cases, the doctors might not give the exact measurements of the breast, *i.e.* they give an approximation of the real measurements and so, it is important to test the robustness of the model. For that purpose, one introduces errors on the measurements and evaluate the effect of that error in the estimation of the parameters.

From doctors experience, it is assumed a potential error up to 10% for each measurement. With that in mind, one uses the test datasets ($\mathcal{D}_{\alpha}^{test}$, where $\alpha = coarse, medium, thin$) and apply a variation that in the end, resulted in an average error per parameter of approximately 7%. The sensitivity analysis performed on the estimation method in section 4.4.2 showed that while the difference on some measurements might not be very relevant, some can have a great impact. So, it is important to evaluate how the NN models deal with potential errors in the measurements. In other words, if an average of 7% error in the measurements causes about 12% to 14% error on the estimated parameters, then one can conclude that the error propagation is within the expected and the NN can be considered as able to provide a reliable and good approximation. Otherwise, if the error caused on the estimated parameters goes up to 20% or more, than the NN is not reliable enough to provide a good approximation to be used by the iterative method.

The impact of the measurements error in the determination of the parameters is assessed and the results are presented in table 48.

| **NN training set mesh** | $e_R$ | $e_H$ | $e_{\lambda_{br}}$ | $e_{\lambda_{sk}}$ | $e_{\mu_{br}}$ | $e_{\mu_{sk}}$ | ***avg total*** |
|---|---|---|---|---|---|---|---|
| **Coarse** | 7.71 | 9.21 | 12.68 | 11.97 | 4.87 | 35.17 | 13.60 |
| **Medium** | 6.68 | 9.09 | 12.29 | 10.58 | 4.67 | 32.36 | 12.61 |
| **Thin** | 7.64 | 11.73 | 12.08 | 11.75 | 4.88 | 32.88 | 13.49 |

Table 48: Mean value of the error in (%) between the reference parameters and the approximation obtained by the neural network model trained with three different data-sets associated to three different meshes.

The difference between the estimated parameters and the reference values increased as expected due to the errors in the measurements. That difference grows from 3.51%, 2.86% and 3.89% (table 46) to 13.6%, 12.61% and 13.33% for the coarse, medium and thin mesh sizes respectively. An average 7% error in the measurements caused the error of some parameters to grow in some cases about 4 times (case of $\mu_{br}$) or even 8 times (case of parameter $H$). This difference can be explained by the non-linear relations between the measurements and the parameters. In fact, despite the increase of the error per measurement of 7% (in average) and the non linearity of the data, the error per parameter only increased around 10%. This means that the error propagation using a trained NN model is not very high and therefore, it can be used reliably as a first estimator of the breast model parameters.

## 6.4 DISCUSSION

Traditional methods for computing bio-mechanical parameters of a complex organ like the breast lye on numerical method as the Finite Element Method for solving an Inverse Problem (parameters identification). Such popular techniques require important computational resources and time response that ranges between one minutes to several hours, depending on the underlying mesh and the required accuracy. The neural network approach offers an alternative way to efficiently assess the mechanical and geometrical parameter of the breast model by returning a very good approximation in real time.

Such a conclusion does not mean that classical bio-mechanical models should be discarded. On the contrary, they are a fundamental piece of the game since the data sets for training and testing are built on the basis of a large number of simulations. The modelling and numerical simulations can be seen as the back-office part of the Neural Network which, in return, assimilate the data and produce a relevant response. Accuracy and efficiency depends on the underlying model, the meshes, the NN architecture and the training stage

(amount of data, training method). For instance, the choice of the best configuration was an important factor in the final results reported in this paper. The number of nodes of a mesh and the number of cases of the training, evaluation and testing data sets affects the learning process by the neural network. In the tests the data sets size was of 40500 elements for all three mesh granularities (coarse, medium and thin mesh). To obtain a similar number of real-life cases could take several years so the usage of validated numerical breast models assures the reliability of the results.

As future work, it would be interesting to consider other measurement method to assess the values of the bio-mechanical parameters such as the Inverse Problem become less sensitive to the volatility of some parameters such as $\mu_{sk}$ (improve the solvability and conditioning). One other aspect, is the impact assessment of these models for real-life cases as well as the elaboration of a data set with real life cases.

# 7

# HYBRID MODEL

Two different approaches to estimate the breast geometrical and mechanical parameters were presented: An iterative method that solves the inverse elasticity problem and a NN based model that learned from synthetic cases. The iterative method presented in chapter 4 is capable of estimating with good accuracy the parameters of the breast model but it can take from several minutes to hours to produce results. The NN model presented in chapter 6 is capable of estimating the parameters almost instantaneously but their results can, in very few occasions, be inaccurate. The optimal solution would consist in the strengths of both approaches, *i.e.* the guarantee of accuracy of the iterative method and the speed of the NN model.

On this regard, a hybrid model is presented in this chapter, where the NN model provides a first estimation of the parameters which is very likely to be close to the reference values, and use the iterative method to validate the accuracy of NN output parameters. In case the estimation is poor, refine those values until a satisfying solution is found. An analysis on the efficiency of this hybrid approach is also presented.

## 7.1 NEURAL NETWORK AND ITERATIVE METHOD

When discussing and analysing the importance of the breast parameter estimation methods in chapters 4 and 6, it was mentioned that their main goal is to determine a set of parameters that allows to create a digital model of the real breast, *i.e.* a model which accurately reproduces the real breast mechanical behaviour as well as its aesthetics. Such model can then be used to support doctors and surgeons making decisions regarding surgical procedures and hence reduce the risk of errors and potentially help correct them. However, if the estimated parameters lead to a breast model whose behaviour differs from the real breast, then, instead of helping, the model can contribute to create problems. Therefore, the method that estimates the parameters needs to be robust and provide reliable results.

Another goal of these methods (which was discussed in chapter 4), concerned the time consumption, *i.e.*, a slow method would not only be unappealing to doctors but also incon-

venient in cases where the doctor might need help within a small time frame (*e.g.* facing a potential problem during a surgical procedure where time is crucial).

### 7.1.1 *Performance Comparison*

To evaluate the overall quality of each method one determines the difference between the estimated parameters and the reference values as well as the time spent by the method. The iterative method performance depends on the initial guess of parameters, and in this case a scenario where the initial guess parameters could differ from the reference values up to 50% was considered. So, the values for the iterative method considered for comparison are the ones already reported in table 8 in section 4.4.3.

The values for the NN method are the same as the ones presented in table 46 in section 6.3.2. These values show the performance by the NN method for the three different mesh sizes. The results of both method are presented in table 49 showing the difference of the approximations as well as the time spent by each method to obtain the parameters.

| **Mesh / Method** | | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | ***avg total*** | ***cycles*** | ***time (s)*** |
|---|---|---|---|---|---|---|---|---|---|---|
| **Coarse** | ***IM*** | 2.30 | 2.71 | 5.28 | 9.41 | 3.07 | 15.72 | 6.42 | 14 | 196 |
| | ***NN*** | 0.74 | 1.13 | 2.36 | 3.76 | 1.33 | 11.72 | 3.51 | - | 0.02 |
| **Medium** | ***IM*** | 2.42 | 3.78 | 5.11 | 9.24 | 2.68 | 15.19 | 6.40 | 13 | 696 |
| | ***NN*** | 0.83 | 1.39 | 2.23 | 3.04 | 1.01 | 8.68 | 2.86 | - | 0.02 |
| **Thin** | ***IM*** | 2.56 | 3.83 | 5.07 | 9.13 | 2.48 | 14.43 | 6.25 | 12 | 1587 |
| | ***NN*** | 1.27 | 3.11 | 4.06 | 4.11 | 1.15 | 9.65 | 3.89 | - | 0.02 |

Table 49: Comparison of the two parameter estimations methods: Neural Network (NN) and the Iterative Method (IM). Mean difference in percentage between the reference parameters and the estimated parameters using three mesh sizes: Coarse, Medium and Thin. Time spent by each method in average and the number of iterations (cycles) required by the iterative method to converge and produce a good result.

The results show that the NN method is capable of estimating the breast model parameters with higher accuracy and in less time. For all meshes the NN method is clearly more efficient since it provides more accurate approximations in a very short time. The error is reduced by 50% and the computational time reduction, reported in the last column, is very significant. Clearly, such results highlight the interest to substitute classical mechanical problem solvers with a Neural Network at the production level. However, as mentioned in section 6.3.2, the NN model estimation can sometimes predict a set of parameters that is not accurate while the iterative method always provides an estimate within the a guaranteed

error maximum level. Therefore the values that are obtained from the NN model need to be used with caution.

In summary, none of these methods should be used by themselves, because despite being very fast, the NN models can provide poor estimations, and on the other hand, despite being reliable, the iterative method can take hours to estimate the breast parameters. However, if one uses the NN first to provide an instantaneous estimation of the parameters and then uses the iterative method to refine and ensure the quality of the estimation, then it is possible to achieve the two goals mentioned previously.

## 7.2 HYBRID METHOD

The hybrid method that estimates the breast geometrical and mechanical parameters consists in performing a first estimation of said parameters using a NN and then use the iterative method to confirm if the estimation is accurate or not. In the case the parameters provided by the NN are not evaluated as accurate, then the iterative method refines the values of those estimated parameters until they are assessed as accurate.

### 7.2.1 *Parameter Estimation*

The NN trained model that estimates the breast parameters ($\Lambda$) obtains a first set of parameters ($\Lambda_{NN}$) after receiving the measurements of the breast ($\overline{M}$). In real scenarios one can only measure the accuracy of the estimation by comparing the estimated measurements ($M_{NN} = M(\Lambda_{NN})$) with the reference measurements ($\overline{M}$) using the cost function (equation 67 in section 6.3.2). In the vast majority of the times, the estimated parameters $\Lambda_{NN}$ already provide a set of measurements $M_{NN}$ which are very close to the reference values $\overline{M}$ (cost function difference below the accuracy threshold value of $5e^{-6}$). But sometimes that estimation is not so good and the value of the cost function are higher than the threshold value ($5e^{-6}$). When that happens, the iterative method can take $\Lambda_{NN}$ as the initial guess and refine the parameters values until that difference goes below the accuracy threshold value obtaining $\Lambda = \Lambda_{IM+NN}$.

As already observed in tables 6, 7 and 8 of section 4.4.3, closer initial guesses, or in this case $\Lambda_{NN}$, leads to lesser iterations by the iterative method to obtain an accurate set of parameters. In table 50, it is presented an analysis of the set of parameters estimated by the NN model for each mesh size with a total number of $N = 500$ cases per mesh. It is also presented the average values of iterations required by the iterative method to converge towards an good solution depending on how close the parameter estimation is from the reference values (these values are taken directly from the tables already mentioned above, namely 6, 7 and 8).

| NN Estimation Performance | | Number of Estimations | Percentage | IM Average Iterations | IM Average Time (min) |
|---|---|---|---|---|---|
| **Coarse (N=500)** | ***Accurate*** | 496 | 99.20% | 0 | 0 |
| | ***Close*** | 1 | 0.20% | 4 | 1.4 |
| | ***Medium Far*** | 2 | 0.40% | 8 | 2.8 |
| | ***Far*** | 1 | 0.20% | 14 | 4.9 |
| **Medium (N=500)** | ***Accurate*** | 497 | 99.40% | 0 | 0 |
| | ***Close*** | 2 | 0.40% | 4 | 4.2 |
| | ***Medium Far*** | 1 | 0.20% | 7 | 7.35 |
| | ***Far*** | 0 | 0.00% | 13 | 13.65 |
| **Thin (N=500)** | ***Accurate*** | 494 | 98.80% | 0 | 0 |
| | ***Close*** | 2 | 0.40% | 4 | 9.8 |
| | ***Medium Far*** | 3 | 0.60% | 7 | 17.15 |
| | ***Far*** | 1 | 0.20% | 12 | 29.4 |

Table 50: Neural Network model parameter estimation deviation groups for each mesh size (Coarse, Medium, Thin): Accurate Estimation; Close Estimation; Medium Far Estimation; Far. Average number of iterations copied from tables 6, 7 and 8 of section 4.4.3 and time in minutes required by the Iterative method to converge and find an accurate solution is calculated from the values of those tables.

The values presented in table 50 show, for each mesh size, the 4 different types of estimations performed by the NN trained models: Accurate; Close (difference of parameters up to 10%); Medium Far (difference of parameters up to 30%) and Far (difference of parameters up to 50%). The NN models are accurate approximately 99% of the times (99.2%, 99.4% and 98.8% with the NN coarse, medium and thin models respectively). This means that in 99% of the cases, the hybrid model will be able to estimate the breast parameters instantaneously.

From the total of 1500 cases (500 cases per mesh size) only 13 $\Lambda_{NN}$ were not accurate: 5 $\Lambda_{NN}$ (0.33% of the cases) were close to the reference values; 6 were medium close (0.4% of the cases) and 2 cases were poor (far) estimations (0.13% of the cases). Considering the number of iterations and the average time required, its is possible to see that in the

worst case scenario (poor estimation obtained with the thin NN model), the most lengthy estimation would require in average approximately 29 minutes.

This analysis show that the hybrid model can not only ensure an accurate estimation of the geometrical and mechanical parameters of the breast, but also ensure to obtain them (in average) very quickly.

### 7.2.2 *Estimation Analysis*

One tests the performance of the hybrid model for the 13 cases whose estimations were not accurate and compares the results with the ones obtained from just using the iterative method (one already knows that the NN model performance was not so good on these cases). The iterative method uses a random initial guess (in this particular case $\Lambda^0 = \{R = 5, H = 5.64, \lambda_{br} = 1000, \lambda_{sk} = 8000, \mu_{br} = 400$ and $\mu_{sk} = 1600\}$).

Tables 51, 52 and 53 present the values of the estimated parameters using the NN model, the iterative method (IM) and the hybrid model considering the three different mesh sizes (coarse, medium and thin). These tables include information about the average error per parameter, the number of iterations taken to find an accurate set of parameters as well as the value of the cost function. Note that this cost function value is done by comparing the reference measurements ($\overline{M}$) and the measurements obtained with the estimated parameters ($M(\Lambda)$).

Table 51 shows a reduction of the number of iterations required by the hybrid model to obtain a good set of parameters when compared to the iterative method (IM) with the exception of case 1 (the iterative method needed 1 less iteration than the hybrid method). This means that the approximation given by the coarse NN model can help the iterative method in reducing the number of iterations. In fact case 2 showed an improvement of 4 times by taking only 2 iterations instead of 8.

In terms of the parameter approximation, comparing the iterative method results with the hybrid solution, it is possible to see that in the majority of the cases the hybrid model have slightly better approximations in average. Notice that the hybrid method does not try to obtain a good estimation for every parameter but rather the entire solution. For example, in case 3, $\lambda_{sk}$ is closer to the reference value in the first estimation with the NN model than after running 3 iterations with the iterative method (1.82% difference before and 6.52% after).

Just like what was seen in section 4.4, the iterative stage of the estimation method using a coarse mesh generally obtains a good estimation of the geometrical parameters and an average approximation of the mechanical parameters. The only exception to this was case 3. The average error per parameter is smaller with the NN estimation than the average error per parameter obtained with the iterative method. This case shows that the average

| Coarse Cases | Method | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | *Avg Total* | *Cost Function* | *Iterations* |
|---|---|---|---|---|---|---|---|---|---|---|
| **1-Close** | **NN** | 1.06 | 8.73 | 0.67 | 2.74 | 1.80 | 17.10 | 5.35 | $8.79e^{-6}$ | - |
| | **IM** | 1.83 | 8.95 | 2.79 | 5.19 | 1.31 | 27.26 | 7.89 | $4.35e^{-6}$ | 5 |
| | **Hybrid** | 0.05 | 1.80 | 3.01 | 0.84 | 2.34 | 15.10 | 3.86 | $3.12e^{-6}$ | 6 |
| **2-Medium Far** | **NN** | 1.35 | 3.94 | 6.02 | 20.79 | 4.09 | 22.78 | 9.83 | $8.94e^{-6}$ | - |
| | **IM** | 1.02 | 0.52 | 2.61 | 4.95 | 4.19 | 14.80 | 4.68 | $4.01e^{-6}$ | 8 |
| | **Hybrid** | 0.20 | 0.67 | 4.36 | 1.67 | 0.61 | 16.18 | 3.95 | $1.02e^{-6}$ | 2 |
| **3-Medium Far** | **NN** | 2.89 | 9.96 | 20.38 | 1.82 | 10.60 | 28.18 | 12.31 | $1.42e^{-5}$ | - |
| | **IM** | 0.72 | 1.34 | 5.21 | 5.40 | 4.20 | 26.32 | 7.20 | $3.5e^{-6}$ | 6 |
| | **Hybrid** | 0.80 | 2.08 | 4.41 | 6.52 | 5.61 | 23.01 | 7.07 | $3.76e^{-6}$ | 3 |
| **4-Far** | **NN** | 15.44 | 26.02 | 40.83 | 11.99 | 8.48 | 47.03 | 24.96 | $6.14e^{-5}$ | - |
| | **IM** | 1.92 | 0.67 | 3.88 | 11.55 | 5.02 | 20.65 | 7.28 | $4.21e^{-6}$ | 8 |
| | **Hybrid** | 1.98 | 0.67 | 5.82 | 8.89 | 2.21 | 21.90 | 6.91 | $2.31e^{-6}$ | 7 |

Table 51: Error (%) between the reference parameters and the calculated parameters for the poor neural network (NN) coarse model estimations. Comparison of the approximation of the NN, with the iterative method (IM) and with the hybrid method (NN+IM). It is shown the average error per parameter (avg total) as well as the difference between the estimated measurements and the reference measurements. The last column show the number of iterations made by IM in both IM model and the Hybrid model.

error per parameter does not necessarily mean that the value of the cost function will be smaller, because as mentioned above, the cost function evaluates the difference between measurements and not parameters.

So, the smaller value of the cost function even with an higher error per parameter presented by the iterative method, can be explained by the compensation phenomena mentioned during the sensitivity analysis in section 4.4.2. The sensitivity analysis showed that an error of a parameter in a certain direction (+ if larger than the reference value of - otherwise) generates a certain error in the measurements which in turn can be countered (in a limited range) by errors in other parameters in the opposite direction. Considering this case, the error caused in the estimated measurements by the $-9\%$ error of $H$ (considering the signal) leads to an error of the $min_z$ measurement that would make it smaller than the reference value. Ignoring momentarily the error of $H$, the errors of parameters $R$, $\lambda_{br}$ and $\lambda_{sk}$ ($+1.83\%$, $+2.79\%$ and $+5.19$ respectively) would lead the value of $min_z$ to be higher than the reference value. So, combining them, one gets a value of $min_z$ that is close to

the reference value but that was obtained with parameters that are not exactly close to the reference parameters.

| Medium Cases | Method | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | *Avg Total* | *Cost Function* | *Iterations* |
|---|---|---|---|---|---|---|---|---|---|---|
| **1-Close** | **NN** | 0.88 | 14.48 | 24.92 | 2.37 | 0.46 | 4.99 | 8.02 | $5.60e^{-6}$ | - |
| | **IM** | 1.52 | 2.72 | 9.43 | 4.30 | 1.23 | 14.32 | 5.59 | $3.87e^{-6}$ | 7 |
| | **Hybrid** | 1.41 | 1.61 | 1.54 | 5.42 | 1.45 | 20.96 | 5.40 | $3.45e^{-6}$ | 2 |
| **2-Close** | **NN** | 9.21 | 1.02 | 12.02 | 7.82 | 8.22 | 14.84 | 8.85 | $7.51e^{-6}$ | - |
| | **IM** | 0.62 | 0.30 | 0.57 | 0.76 | 3.02 | 13.73 | 3.17 | $1.14e^{-6}$ | 5 |
| | **Hybrid** | 0.28 | 0.60 | 3.09 | 0.76 | 2.12 | 11.43 | 3.05 | $1.12e^{-6}$ | 2 |
| **3-Medium Far** | **NN** | 25.33 | 24.40 | 24.48 | 3.52 | 8.02 | 27.48 | 18.87 | $4.00e^{-5}$ | - |
| | **IM** | 5.31 | 4.94 | 9.24 | 8.99 | 3.41 | 15.21 | 7.85 | $3.56e^{-6}$ | 9 |
| | **Hybrid** | 2.08 | 0.25 | 4.74 | 13.51 | 1.87 | 12.71 | 5.86 | $3.02e^{-6}$ | 5 |

Table 52: Error (%) between the reference parameters and the calculated parameters for the poor neural network (NN) medium model estimations. Comparison of the approximation of the NN, with the iterative method (IM) and with the hybrid method (NN+IM). It is shown the average error per parameter (avg total) as well as the difference between the estimated measurements and the reference measurements. The last column show the number of iterations made by IM in both IM model and the Hybrid model.

Table 52 continues to show a reduction of the number of iterations required by the hybrid model to obtain an accurate set of parameters when compared to the iterative method. There are only 3 cases where the medium NN model was not accurate, but 2 of them (cases 1 and 2) show values of the cost function close to the accuracy threshold value and with that, the hybrid model only took 2 iterations to refine the NN model results. When comparing these two cases with the bottom case (medium far estimation by the NN model) it is possible to see the impact a close estimation can have in terms of the performance of the iterative method, *i.e.* closer estimations by the NN model can lead to a significant reduction in time consumption. Concerning the error per parameter, the results continue to show that the iterative method aim is not to approximate each parameter to the reference value but rather the entire set of parameters as a whole.

Table 53 shows the 6 cases were the NN model estimated a set of inaccurate set of parameters. The results show a similar pattern as the one seen in the previous tables (51 and 52) which is a reduction of the number of iterations of the hybrid model which runs the iterative method with the initial guess provided by the NN model compared to the iterative method with a pre-determined initial guess value. These results show once again the impact of the importance of the approximation. In the cases where the NN model estimation

| Thin Cases | Method | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | *Avg Total* | *Cost Function* | *Iterations* |
|---|---|---|---|---|---|---|---|---|---|---|
| **1-Close** | **NN** | 0.18 | 3.02 | 2.88 | 12.84 | 0.14 | 12.68 | 5.29 | $5.09e^{-6}$ | - |
| | **IM** | 1.00 | 4.60 | 6.68 | 0.76 | 0.18 | 9.06 | 3.71 | $4.89e^{-6}$ | 14 |
| | **Hybrid** | 0.09 | 2.94 | 0.42 | 0.56 | 0.10 | 5.39 | 1.58 | $3.11e^{-6}$ | 1 |
| **2-Close** | **NN** | 4.74 | 0.85 | 8.89 | 4.56 | 2.48 | 24.20 | 7.62 | $1.24e^{-5}$ | - |
| | **IM** | 0.16 | 3.15 | 3.94 | 5.56 | 0.09 | 21.43 | 5.72 | $2.45e^{-6}$ | 8 |
| | **Hybrid** | 2.83 | 0.51 | 3.09 | 5.28 | 0.03 | 18.04 | 4.96 | $1.87e^{-6}$ | 2 |
| **3-Medium Far** | **NN** | 1.37 | 8.84 | 46.36 | 3.19 | 0.22 | 3.73 | 10.62 | $6.89e^{-6}$ | - |
| | **IM** | 1.02 | 4.24 | 5.41 | 2.56 | 0.67 | 13.98 | 4.65 | $3.22e^{-6}$ | 15 |
| | **Hybrid** | 0.67 | 3.16 | 1.58 | 2.12 | 0.88 | 17.48 | 4.32 | $3.07e^{-6}$ | 5 |
| **4-Medium Far** | **NN** | 0.78 | 11.07 | 17.26 | 10.34 | 2.29 | 28.62 | 11.73 | $9.40e^{-6}$ | - |
| | **IM** | 3.93 | 3.07 | 5.96 | 0.60 | 6.21 | 23.93 | 7.28 | $4.63e^{-6}$ | 7 |
| | **Hybrid** | 4.52 | 0.89 | 7.02 | 6.46 | 5.21 | 28.62 | 8.79 | $4.71e^{-6}$ | 2 |
| **5-Medium Far** | **NN** | 1.11 | 19.59 | 19.27 | 15.52 | 4.15 | 11.97 | 11.94 | $2.52e^{-5}$ | - |
| | **IM** | 0.03 | 9.46 | 3.76 | 8.70 | 1.99 | 7.58 | 5.26 | $4.16e^{-6}$ | 9 |
| | **Hybrid** | 1.20 | 7.53 | 5.83 | 12.28 | 0.59 | 14.70 | 7.02 | $4.37e^{-6}$ | 6 |
| **6-Far** | **NN** | 2.20 | 42.69 | 9.49 | 33.04 | 6.82 | 40.38 | 22.44 | $1.18e^{-5}$ | - |
| | **IM** | 0.07 | 1.44 | 6.34 | 3.23 | 7.65 | 2.89 | 3.60 | $1.16e^{-6}$ | 7 |
| | **Hybrid** | 0.99 | 2.17 | 2.91 | 4.53 | 7.45 | 17.73 | 5.97 | $2.95e^{-6}$ | 8 |

Table 53: Error (%) between the reference parameters and the calculated parameters for the poor neural network (NN) thin model estimations. Comparison of the approximation of the NN, with the iterative method (IM) and with the hybrid method (NN+IM). It is shown the average error per parameter (avg total) as well as the difference between the estimated measurements and the reference measurements. The last column show the number of iterations made by IM in both IM model and the Hybrid model.

was close to the reference parameters (cases 1 and 2), the hybrid model just needed 1 or 2 iterations to find a good result. On the other cases where that approximation was not so good, namely cases 3,5 and 6 the number of iterations increased to 5, 6 and 8 iterations respectively. Nonetheless it was still a smaller number of iterations when compared with the standalone iterative method.

## 7.3 DISCUSSION

The iterative method and NN models are capable of providing good estimations of the breast geometrical and mechanical parameters. The iterative method is robust and reliable

but is slow. The NN models can provide results very quickly but they do not guaranty that the estimated parameters are accurate.

The hybrid model presented in this chapter combines these methods. It uses the NN models to obtain a fast estimation of the parameters which is accurate in 99% of the cases, and on the other 1% it reduces the number of iterations required by the iterative method to find an accurate solution. Moreover, the iterative method serves as a validation for the estimations performed by the NN model.

This new approach of estimating the breast parameters show the potential of combining ML methods and models, like NN, together with other numerical methods, as the iterative inverse problem solver, to create more efficient estimators/predictors.

The tests performed with this new model are very good and show great potential of its usage in real situations. Future work would involve testing this new approach with real breast as well as look into other possibilities that were not explored (for example, other ML algorithms or other inverse problem solvers).

# 8

# REAL APPLICATIONS WITH A WEB PLATFORM

A large spectrum of numerical tools to simulate the breast reduction was developed and the final goal is to carry out simulations in relation with concrete situations, namely real surgical acts on the breast involving knitting and suturing with patient. For instance, the model presented in chapter 3 was confronted to real cases and the results were reported in Lopes et al. (2017a,b). One highlights that the comparison with real cases was only possible thanks to the pictures provided by Hospital de Gaia, and more specifically the data collected by Dra. Augusta Cardoso among her patients that consent to pose for pictures. Dra. Augusta Cardoso has made an important contribution to this research project enabling all the validations with concrete cases.

Along with this goal, an ecosystem was designed and implemented around a friendly-user web platform to provide simulations of the breast reduction. Such an ecosystem is dedicated to doctors and surgeon to easily plane the operation and experiment different options to assess scenarios and prevent possible aesthetic and functional problems with the surgery. This platform allows doctors from Hospital de Gaia to perform tests and report on the validity of the obtained results. The results obtained with real life cases and the testing of the web platform is reported in this chapter.

## 8.1 SURGERY SIMULATION OF REAL LIFE CASES

Breast reduction is a common procedure in plastic surgery to achieve medical and/or aesthetic weight diminution. It consists in removing excessive fat, glandular tissue and skin from the breasts to relieve the tension from the vertebral column and reduce large deformation effect on the breast shape. The surgeon makes a set of precise cuts and removes excess tissue. After that, the suturing process allows to achieve the desired shape and a new position for the nipple-areola complex (Cardoso et al. (2013b)). A breast model and the boundary conditions of the breast reduction surgery problem were already presented in chapter 3. This section presents the simulation of breast reduction surgeries and the comparison of the results with the real life cases.

### 8.1.1 *Surgery Characteristics*

Breast reductions surgeries are planned during consultations were the surgeon performs calculations which are reflected by markings on the breast. These lines show the cuts that will be performed as well as the location of the new nipple position (figure 45). A breast

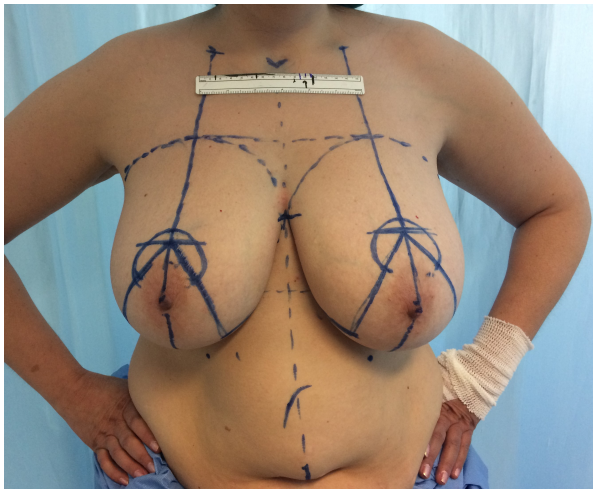


Figure 45: Breast reduction surgery planning marks on a patient. Usage of rulers to be able to obtain the measures from photographs. Photography taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

reduction surgery can be divided into three different stages:

1. Pre-surgical stage: At this stage, the surgeon measures the breast and performs some calculations regarding to the new position of the nipple which is generally above the previous nipple position (Cardoso et al. (2013b)). From that position the surgeon draws lines to the right and left following a trajectory that makes an angle with the medium line (line between the nipple and the sternal notch) which, in the analysed cases, are situated between 25 and 50 degrees. In most cases, the right and left angle of the drawn lines are very similar, with some exceptions where the difference may reach a maximum of 5 degrees. After the lateral lines drawing, cuts from the base onto the breast frontal are printed;

2. Surgical stage: The surgeon performs cuts on the frontal and lateral incision lines and removes tissue (glandular tissue and fat). Then it sutures the frontal lines together with special caution to guarantee the end of each lateral line matches each other.At last, the practitioner sutures the lateral lines onto the torso which will be the new inframammary fold;

3. Post-surgical stage: After the surgery the breast is swollen and along several weeks, the doctor accompanies the patient, evaluating the breast tissues rearrangement. After

a period that can be of several months, if the surgery was not successful (for example, the nipple remained in a bad position in aesthetic terms) then the surgeons plans an additional surgery to correct the problem.

### 8.1.2 *Simulation and Evaluation*

The numerical method presented in chapter 3 simulates the second and third stage of the surgical procedure by receiving the data from the first stage, *i.e.* the information about the breast biomechanical parameters and the cuts to perform. It carries out the suturing of the incised surfaces to obtain the new breast and then applies the gravity force to compute the shape of the breast after the surgery. Figures 46 and 47 display an example of the markings drawn by the surgeon on the patient's breasts during the pre-surgical consultations.

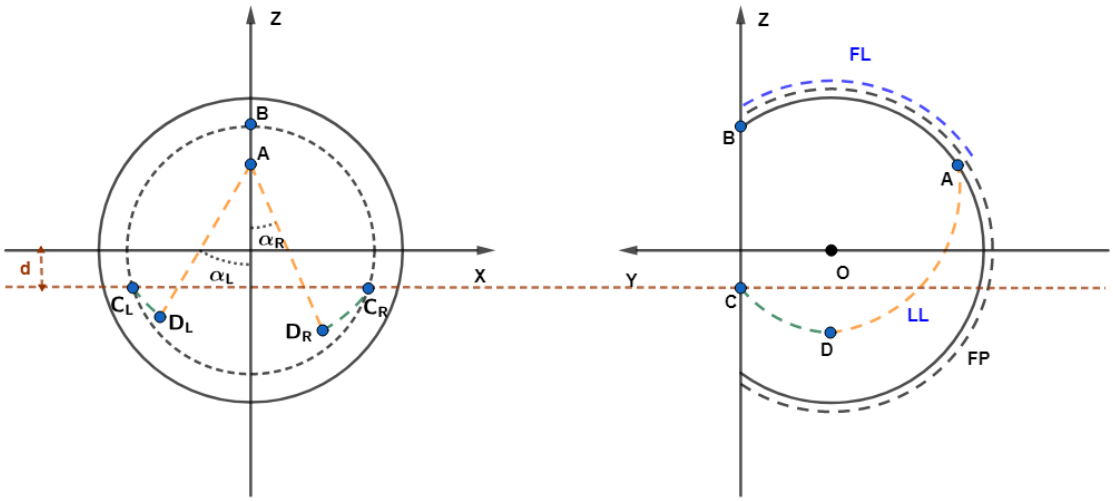



Figure 46: Scheme of the breast with the incision parameters and the points used to generate the breast mesh. On the left is the breast frontal view while on the right is the breast side view.

Anthropological information with respect to the size of the breast are key elements for the software to generate the mesh. $FP$ and $FL$ assessments enable the localisation of the new nipple position, while angles $\alpha_L$ and $\alpha_R$ are used to obtain the angle of the frontal incisions of length $LL$. The lateral incision that will be sutured to the torso during the surgical procedure, goes from the end of the frontal incision and extends to an incision point situated on the breast's base and characterise with coefficient $d$. Figure 48 displays the markings made by the surgeons that enable to define the reference points used to generate the breast mesh.

- Point $A$: new position of the nipple;
- Point $B$: top middle point of the breast that is located on the torso;

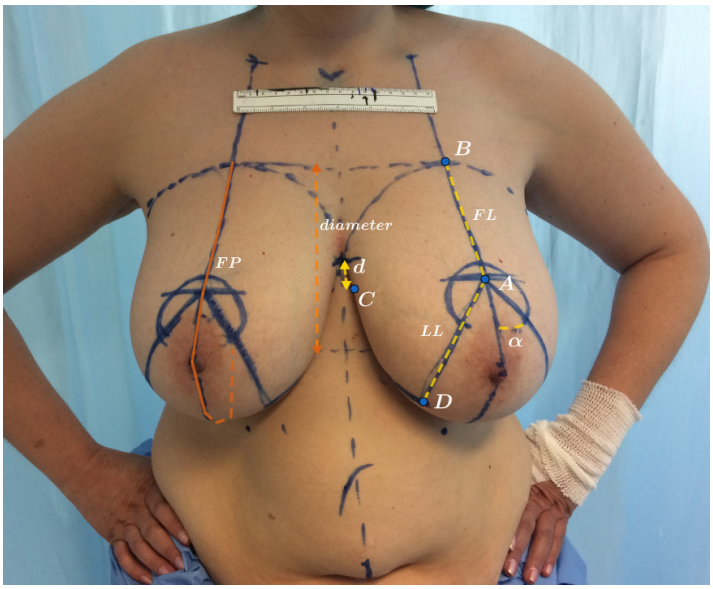



Figure 47: Picture of the real breast with the markings and the scheme values represented. Photography taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

- Point $C_L$: point on the base of the breast that marks the start of the left lateral incision;
- Point $C_R$: point on the base of the breast that marks the start of the right lateral incision;
- Point $D_L$: point on the breast that marks the intersection between the left lateral and frontal incisions;
- Point $D_R$: point on the breast that marks the intersection between the right lateral and frontal incisions;
- Point $E$: projection of the position of the nipple on the torso;
- Point $O$: centre of the sphere used to generate the round shape of the breast.

In the mesh generator software GMSH, one uses parametric expressions to create lines and arcs as well as the surface. At last, the listing the surfaces provide the three dimensional shape. One generates a mesh constituted of tetrahedral elements of the breast and triangles for the faces. At the end of this process one obtain a complete mesh of the incised breast.

To simulate the suturing process, the breast model presented in chapter 3 uses the mechanical parameters of the breast fat and glandular tissue ($\lambda_{br}$, $\mu_{br}$) as well as the mechanical parameters of the skin ($\lambda_{sk}$ and $\mu_{sk}$) to model the behaviour of the mesh elements. These parameters are used not only for the suturing process but also to obtain the breast configuration when under gravity.

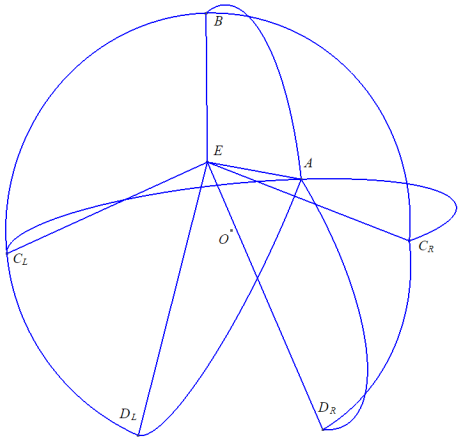



Figure 48: Breast Geometry skeleton made by the points obtained with the markings done by the surgeons on the patient's breasts.

*Manual Parameter Tuning*

As a first guess, the parameters are deduced from observation and tune with successive optimisation, *i.e.* the values of the parameters are adjusted until the visual aspect of the digital model matches the visual aspect of the real breast. To generate the digital breast, one uses the breast model circumscribed to the boundary conditions of section 3.3. Such a breast mesh is defined in a free-stress state and undergoes a transformation by applying the gravity field which displaces the nodes until the strain energy is minimal. Different values of the parameters lead to visually different breasts that are tuned by successive approximations. One example of the estimation of the parameters using a manual optimisation can be seen in figure 49.

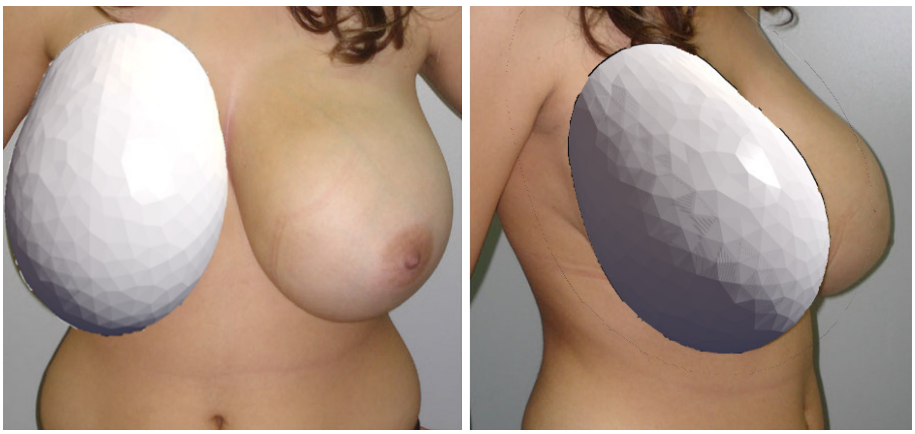


Figure 49: Visual Comparison of a patient's breast before the surgery with the digital model. Digital model on top of the patient's right side breast. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

The pictures show a frontal and lateral view of the patient's breast before the surgery. To obtain this digital breast, the six parameters used were $\Lambda^{real} = (R = 0.1, H = 0.04, \lambda_{br} = 8000, \mu_{br} = 800, \lambda_{sk} = 8000, \mu_{sk} = 1600)$. One obtains a digital breast that fit with the real breast, specially with respect to the volume. However, the lateral view indicates that the real breast is not as smooth as the digital breast. The patient was submitted to a breast reduction procedure with nipple repositioning 3cm above its original position while frontal cuts are originated by a 45 degrees angle with a length of 8cm. After 18 months the patient went back for a checking and the doctor took another set of pictures to record the final shape of the breast (see figure 50).

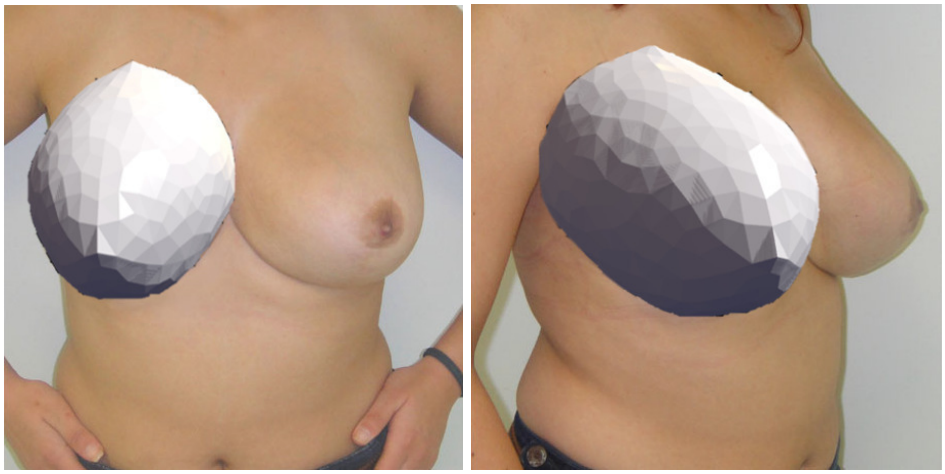


Figure 50: Visual Comparison of a patient's breast 18 months after the surgery with the digital model. Digital model on top of the patient's right side breast. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

Comparing the digital model with the real breast after the surgery, it is possible to observe a good fitting between the two shapes. The nipple is well-positioned and the lower part (infra-breast zone) is also correctly positioned. However, the volume is slightly larger than the one measured on the breast along with a small difference on the superior part of the breast. Such errors may be derived from the finite element approximation errors of the breast parameters as well as the approximation of incision parameters. Different skin mechanical parameters may also produce better visual aspect and underline the difficulty for well tuning the breast model parameters using only visual rule of thumb.

*Parameter identification*

In chapters 4, 6 and 7, numerical procedures have been developed to determinate the breast parameters with higher accuracy. They require more information for achieving the estimates and rough anthropological data are unsatisfactory. The parameter identification requires measures taken in different positions (standing up, bend forward and in prone position) and additional set of measurements per patient are necessary. These measurements

can be collected from photos with metric references that allow to convert the distance in pixels to real measurements.Examples of frontal and lateral views of the breast are displayed in figure 51. For each different position, one gathers 5 measurements. Assessments of $min_y$,

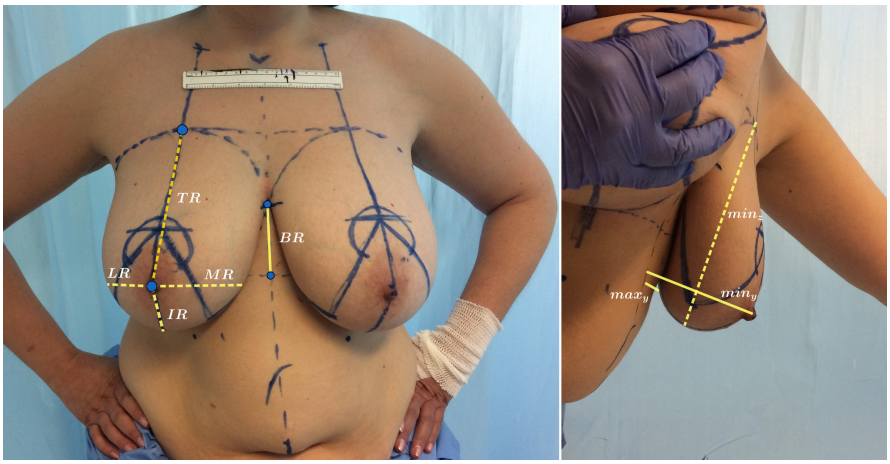



Figure 51: Visual Comparison of a patient's breast 18 months after the surgery with the digital model. Digital model on top of the patient's right side breast. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

$min_z$ and $max_y$ can be directly obtained from the pictures while the volume and surface area the breast have to be calculated. The volume is obtained with an anthropometric data and a formula proposed in Kayar et al. (2011):

$$volume = \frac{\pi}{3} \cdot min_y^2(IR + LR + MR - min_y) \tag{68}$$

where $IR$ represents the inferior breast radius (calculated from the inferior frontal perimeter), $LR$ represents the lateral breast radius which is approximated to have the same value as $IR$ and $MR$ which is the medial breast radius and that is obtained with the top frontal perimeter.

The breast skin surface area is similarly calculated using a simplification of the breast shape under gravity: The lower part of the breast up to the nipple is assumed to be a half sphere and the rest of the breast surface area is approximated to two thirds of the lateral side of a cylinder whose height is the same as the top frontal perimeter:

$$area = \frac{4}{3}\pi \cdot BR \cdot TR + 2\pi \cdot IR^2 \tag{69}$$

where $BR$ represents the radius of the base of the breast and $TR$ is given by the top frontal perimeter (line between the nipple and the middle top part of the breast). The measurements obtained from the pictures and from the formulas are approximations to the real values and therefore the final results will also be an approximation to the volume and area.

The parameter estimation method was tested in a real case situation with the patient shown in figure 45. Several pictures were taken before and after the breast reduction surgery. However, due to practical difficulties, only two positions were photographed: standing up and bending forward. For the prone position measurements, an interpolation was done which is based on the values of the many cases used to train the different NN. Indeed, a relation between the measurements for the three positions have been deduced and estimates of the unknown measurements for the prone position was calculated in order to deliver the measures vector ($\overline{M}$). Applying the hybrid model presented in chapter 7, one gets an estimate for the breast parameters which are reported in table 54.

| | $R$ | $H$ | $\lambda_{br}$ | $\lambda_{sk}$ | $\mu_{br}$ | $\mu_{sk}$ | ***Cost Function*** | ***Iterations*** |
|---|---|---|---|---|---|---|---|---|
| **Coarse** | 0.0731 | 0.0803 | 1068 | 2702 | 394 | 2330 | $4.25e^{-6}$ | 5 |
| **Medium** | 0.0640 | 0.0892 | 1081 | 3344 | 327 | 2726 | $8.59e^{-7}$ | 0 |
| **Thin** | 0.0650 | 0.0895 | 1155 | 4039 | 334 | 2930 | $3.94e^{-6}$ | 2 |

Table 54: Estimated parameters by the hybrid model considering three mesh sizes (coarse, medium and thin) and the differences between the estimated measurements and the reference measurements (Cost Function) as well as the number of iterations necessary to produce an accurate result.

To check the quality of the estimated parameters $\Lambda_{estimated}$ provided by the hybrid model, one computes the breast under gravity for the three positions with the numerical model and deduce the 15 associated measures $M(\Lambda_{estimated})$. Then the relative difference between the experimental measures $\overline{M}$ and $M(\Lambda_{estimated})$ provides the error between the model and the reality. Table 54 shows that the estimated parameters obtained with the hybrid model are very good since they are below the error threshold of $5e^{-6}$ which yields that the differences of measurements are smaller than one millimetre. The estimation using the medium size mesh was instantaneous while the one using the coarse and thin meshes took 5 and 2 iterations respectively. The digital breast is compared (visually) with the real case picture in figure 52 for two different positions (standing up of the left picture and bending forward approximately 45 degrees on the right picture). An excellent fitting between the digital breast and the real one before the surgery is observed. The digital model not only approximates the volume but also the shape. On the right picture, the patient is bending forward and small differences on the top part of the breast are visible. Such an deviation may result from the approximations of the measurements and the interpolation of the measurements of the breast assuming the patient in a prone position.

The breast reduction surgery was made using an frontal cut angle of 40 degrees as well as a value of $FL$ of 11cm, a value of $LL$ of 9.5cm and a value of $d$ of 3cm. Using these parameters with the breast model of chapter 3 one obtained a digital model representing

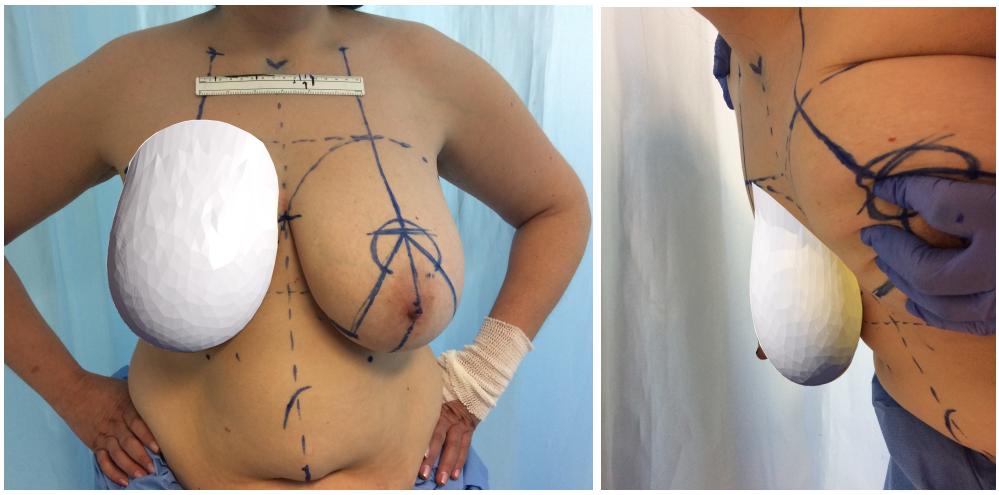


Figure 52: Visual Comparison of a patient's breast before the surgery with the digital model with the estimated parameters. Digital model on top of the patient's right side breast. Pictures of the patient standing up (left) and bending forward 45 degrees (right). Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

the breast after surgery. The visual comparison of the patient's breast and the digital model is displayed in figure 53.

The visual aspect is quite satisfactory, with a special mention for the nipple position which is very well located. The volume of the breast is also very similar and one only notices some few differences between the final shape of the digital breast with the real one (on the top part of the breast) that may derive from the bruising still visible on the breast which can make the breast swollen, *i.e.* the shape of the breast may not be the same after a recovery period. Nonetheless the approximation is really better with the parameters estimated using the method proposed in chapters 4 and 6 in relation with the breast shape obtained with the "try and check" optimisation. Numerical method that enable to establish an accurate breast model with correct parameters deriving from simple measurements obtained during consultations turns to be an important predictive tools for real life applications.

## 8.2 WEB PLATFORM

Breast reduction surgery is a complex plastic intervention that requires surgeon's experience in order to achieve functionally and aesthetically good results. Computer simulation program based on biomechanical models allows one to better understand the influence of preoperative key parameters on the post surgical breast geometry of the patient. Simulation of scenarios with different preoperative markings could prevent errors during the intervention planning or to add modifications when the objective was not achieved. Further real-time simulations will ease decision-making and will allow error correction during the

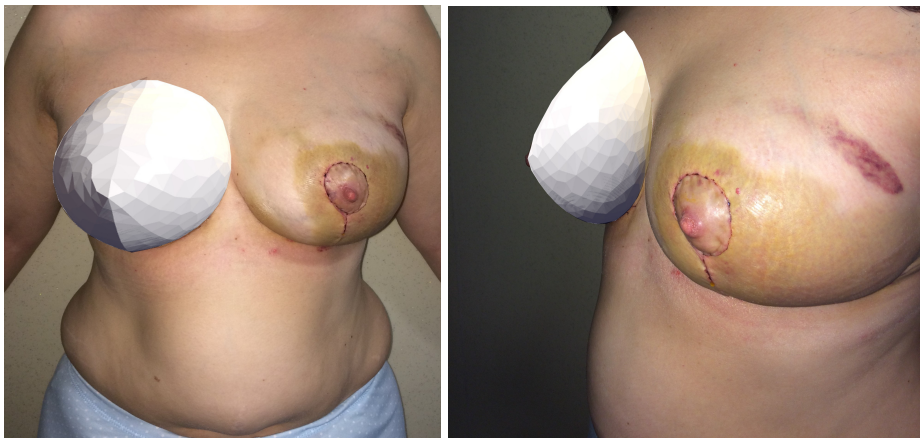


Figure 53: Visual Comparison of a patient's breast with the digital model with the estimated parameters. Digital model on top of the patient's right side breast. Pictures of the patient standing up with a frontal (left) and lateral view (right) taken a short period after the surgery. Photographies taken by Dr. Augusta Cardoso and reproduced with kind permission from Hospital de Gaia.

interventions. However the interaction between the researchers and the doctors is limited to time and space restrictions (they usually work in different locations). To provide the fully accessible expertise tool, a web platform was developed.

### 8.2.1 *Objectives*

The main objective of this platform is to provide 24h/24h available tool via any browser to use the breast model and experiment scenarios. The tools have to be user-friendly, well-adapted to the health context in order to be employed in an autonomous way. The easy accessibility of such platform allows doctors to experiment different surgery approaches at any time and anywhere. Moreover, they can validate results and provide feedback more quickly (saving the necessity to meet in person with the researchers). Alterations were made to the code (like the optimisation proposed in chapter 5) in line with the doctors feedback to correct visual bugs and small defaults. Other possible usages of this type of platform include:

- Helping surgeons to prepare future breast surgeries;
- A tool for medical students to gain experience and develop sensitivity regarding breast surgical techniques and a new insight to the breast behaviour;
- A virtual environment that allow doctors and surgeons that live in different countries to share information about breast surgeries they've performed.

### 8.2.2 *Platform Structure*

The web platform provides a user-friendly tool to test the breast model and simulate breast reduction surgeries. This platform is located at `http://surgery.natura2.di.uminho.pt/vaps.html` and composed of several components:

- Interface;
- Breast Simulator Code;
- Server Code.

The interface consists in web pages that were created using Hypertext Markup Language (HTML) and Cascading Style Sheets (CSS) to design the page structure, Three.js library and the WebGL API to render the breast meshes (pre-surgery breast, incised breast, sutured breast and post-surgery breast) and other JavaScript code to improve the pages performance (*e.g.* responsiveness). Figure 54 depicts the web platform interface. The simulation

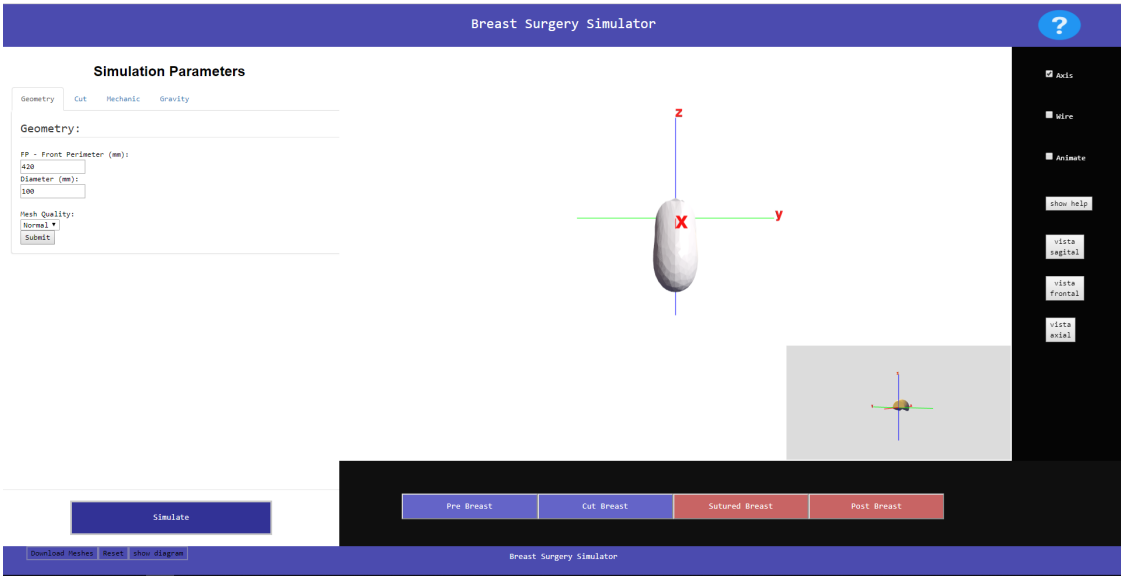



Figure 54: Web platform interface.

parameters are located in the left panel, the digital breast is drawn in a central position. In the right panel there are buttons to provide help for manipulating the mesh (like drawing axis or placing the mesh in a frontal position) and 4 buttons that allow to change the breast view between the pre-surgical breast as well as the breast with incisions, the sutured breast and the breast after the surgery.

The breast simulator code receives the input from the web page and runs the code that will generate the breast mesh which in turn is displayed in the web page. The code is

divided into different GMSH and C++ files responsible for performing the different operations. The GMSH files contains the breast geometric parameters that run with the GMSH software to generate the mesh. One file generates the breast geometry before the surgery occurs (scheme based on figure 26 of subsection 4.2.1) while an other file generates the breast geometry with the incisions (scheme based on figure 46 of section 8.1). Those files are then used by the C++ compiled code to simulate the breast behaviour (in the pre-surgical breast simulating the breast under gravity and for the incised breast simulate the suturing process as well as the effect of gravity on the sutured breast).

The server code implemented in Hypertext Preprocessor (PHP) handles the flow from the simulation code and operates the background methods by enabling the communication between the interface and the simulator code. The PHP scripts also ensure that multiple users can use the platform at the same time by managing temporary copies of the key files that are associated to a session id. This way, many users can perform their tests without the possible interference of others.

*Simulation Parameters*

To perform a numerical surgery for suturing the breast, the operator has to set different values onto the various simulation parameters:

1. Breast geometry;
2. Knitting and cutting parameters;
3. Mechanical Parameters;
4. Gravity Field.

After the practitioner validation of the data, the breast geometry lies on two parameters: diameter of the breast's base and the frontal perimeter of the breast (figure 55). All the coefficients derive from the $R$ and $H$ parameters used in the breast model of chapter 4. In the web platform, those values are given in millimetres and in the geometry section of the simulation parameters the user can also choose the mesh granularity (coarse, medium or thin). It is important to mention that this platform does not still possess the implicit mechanism to estimate the breast parameters from a set of 15 measurements. This will only be implemented in a future version.

After setting the geometrical parameters then the user can set the incision parameters which can be seen in figure 56. Except for the cut angle which is in degrees, the other incision parameters such as the frontal distance ($FL$), the lateral incision length ($LL$) and the height of the back incision ($d$) are set in millimetres. $FL$ is a knitting condition which is parameterised to set the new nipple position. The angle of the incisions ($\alpha$) depends on the amount of breast mass the surgeon plans to remove. The lateral cut is controlled

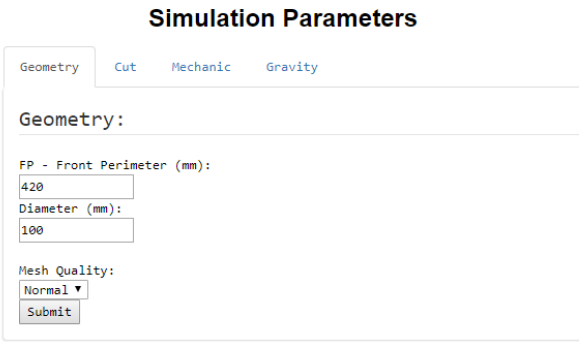



Figure 55: Web platform breast geometry interface inputs.

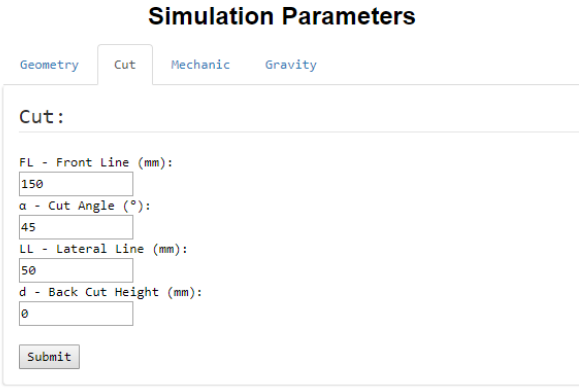



Figure 56: Web platform breast incision interface inputs.

by the junction line between the thorax and the breast and the end of the frontal cuts parameterised with $LL$. Notice that the surgeon tries to balance the amount of tissue to remove so that a natural final shape can be achieved.

After fixing the geometry, the user go on to the mechanic pan and setup the parameters (figure 57). The physical coefficients of the breast fat and glandular tissue $\lambda$ and $\mu$ (which are the same as the parameters $\lambda_{br}$ and $\mu_{br}$ of the model in chapter 3), the ones of the breast skin $\lambda_p$ and $\mu_p$ (which are the same as the parameters $\lambda_{sk}$ and $\mu_{sk}$ of the model in chapter 3), and additional parameters such as the skin thickness, the value of the Chassaignac coefficient and the value of the breast density ($\rho$). The surgeon can also use an complementary offline software such as parameter identification methods to feed the data or they can tune the parameters with successive experiences in order to reproduce a breast shape that better suits the real breast.

The last section corresponds to the gravity field definition. As shown in figure 58, gravity is a vector that enables to represent the patient position, *i.e.* standing up, lying down and bending forward.

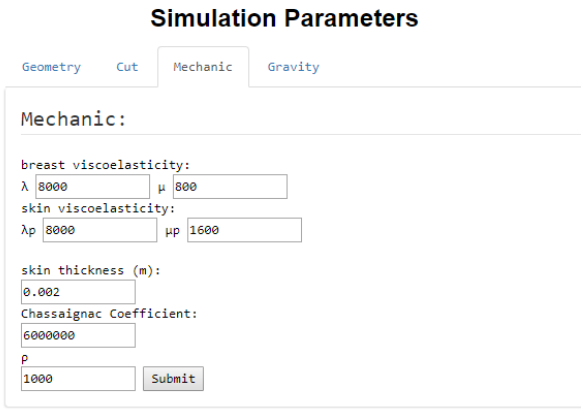



Figure 57: Web platform breast mechanic interface inputs.

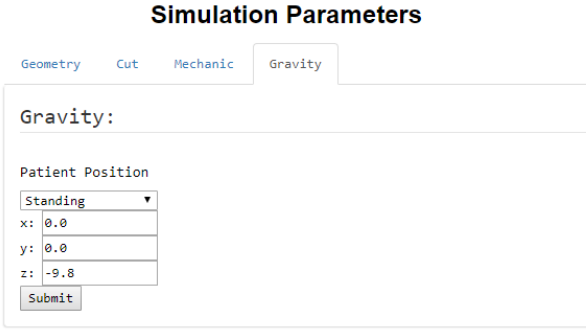



Figure 58: Web platform breast gravity interface inputs.

*Breast Mesh*

The WebGL API renders the breast's mesh on the local computer. There are however 4 different breast meshes which represent the different stages of the breast surgical procedure: pre-surgical breast; incised/cut breast; sutured breast and post-surgical breast. When first accessing the web page, it allows the user to see the pre-surgical breast and the incised breast according to the default parameters (figure 59). The pre-surgical breast (left image on figure 59), as a companion small view of the breast with incisions on the lower right corner. This view mode can be accessed by pressing the 'Pre Breast' button. To obtain an alternative view of the breast with cuts, the 'Cut Breast' button is activated and displays the mesh with more details (right image on figure 59).

Once activating the simulation button, the breast simulation code runs, performs the suturing leading and displays the final results of the surgery simulation (figure 60). Comparisons between figures 59 and 60 exemplified the colour system used with the 'Sutured Breast' and 'Post Breast' and optional click enables to change from red to blue. The user can see the sutured breast in a free stress state (left image of figure 60) and under gravity

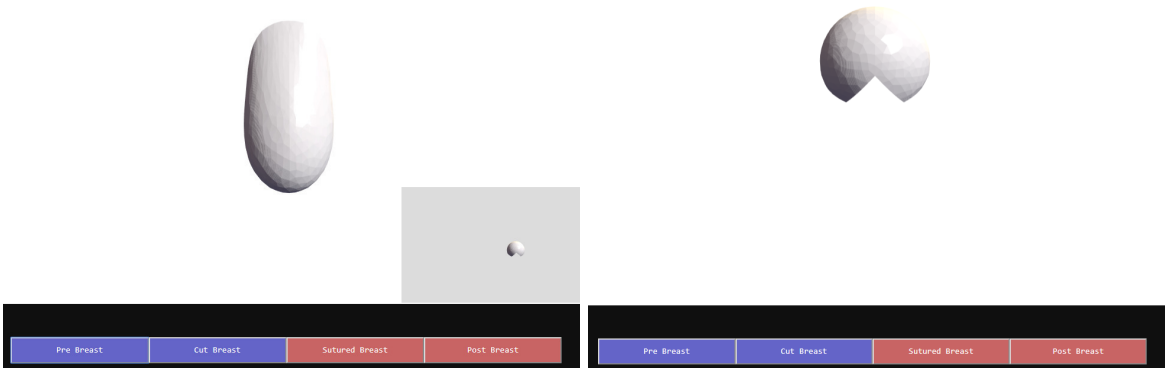



Figure 59: Web platform rendering of the pre-surgical (left) and incision (right) breast meshes.




Figure 60: Web platform rendering of the sutured breast (left) and the post-surgery breast (right).

(right image of figure 60). When the simulation ends, the portion of removed breast mass is report in a green rectangle the value (in millilitres).

*Helpers*

The platform also includes helpers in the form of diagrams and explanatory text. Figure 61 gives an example of a text helper which is triggered when the mouse hovers above the label of an input field, in this particular case above the cut angle $\alpha$. The platform also has an instruction helper which can be triggered by pressing the button 'show help' on the right panel. Figure 62 shows the instructions that can help beginner users that might not know how to perform simulations. A complementary helper is given by a diagram as it is shown in figure 63. This diagram shows a frontal and lateral view of the breast with incisions helping to explain what some of the parameters (geometrical and incision/cut) mean. It is displayed when pressing the 'show diagram' button on the lower left corner below the 'Simulate' button.

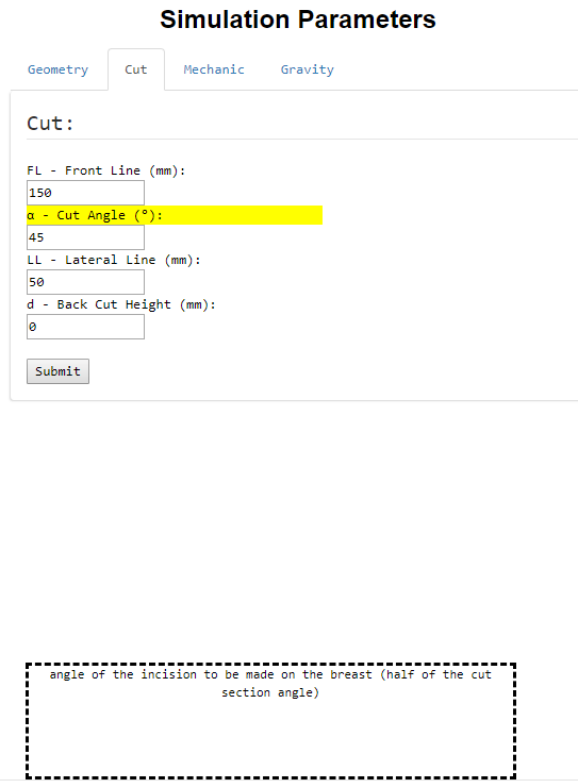



Figure 61: Web platform text helper.

### 8.2.3 *Workflow*

When a user accesses the web platform, he/she receives a session id handled by a PHP script. This script creates a folder where one stores the files to be read by the C++ codes when running the simulations. These files include the storing of the breast simulation parameters used to execute the simulations.

To perform a breast reduction surgery simulation, the user has to setup the simulation parameters and then press the 'Simulate' button to run the C++ code that performs the suturing and simulates the effect of gravity on the sutured breast. As figure 64 shows, when the simulation step is activated, it triggers a green rectangle that informs the user that the server is running the simulation. On the background, the compiled program that performs the suturing and gravity simulation reads the files with the submitted parameters and completes the simulation process generating the breast mesh files. When the simulation completes, the PHP script returns information to the local browser and updates the page (namely the 'Sutured Breast' and 'Post Breast' buttons). Then the user see the result of the surgical simulation and if eventually repeat or test other scenarios by modifying the simulation parameters and re-run the code.

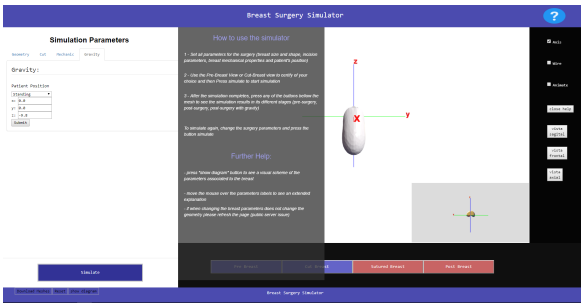



Figure 62: Web platform user instructions.

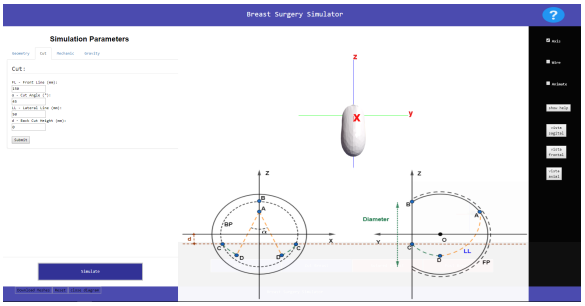



Figure 63: Web platform cut diagram.

## 8.3 CONCLUSIONS

Surgical simulations represent a crucial predictive tool to prepare and design a surgery act by reducing the risk and increasing the predictability of the surgery. Comparisons are drawn between predictions and real situations to assess the efficiency and the accuracy of the method to provide the final shape of the operated breast. Parameter identifications such as the mechanical parameters of the breast turn to be a crucial goal since their values dramatically change with the person and have to be provided for each patient in order to obtain a more suitable simulation.

The estimation of the breast parameters using tuning by iterative experiences indicates that it is possible to approximate the breast model parameters in such a way that the digital breast better fit with the real breast. However such a process can be very time consuming and, for users that are not familiar with the breast model, it is easy to achieve satisfactory results. The hybrid estimation method presented in chapter 7 is quite efficient and demonstrates the potential that this method will have in the future. The number of tests made with real cases is still in progress to completely validate a reliable method but the results are promising.

Simple access and friendly user tools are necessary to enable the surgeon to adopt and efficiently use numerical tools. A Web support simulator for breast knitting has been developed and surgeon experiences were reported. The digital breasts that were used as

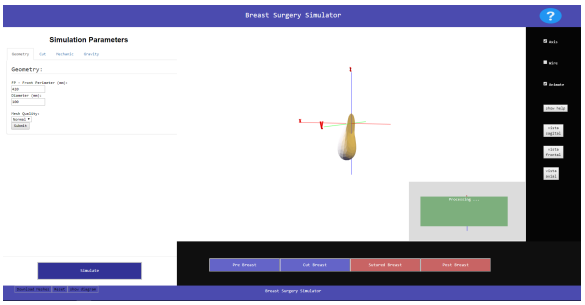



Figure 64: Web platform simulating the surgical procedure.

comparison with the real life cases in section 8.1 were created using this platform. The web interface has multiple usages that can benefit not only the doctors but also medical students and ultimately the patients.

Future work includes the development of more features and the improvement of the web platform. Additional tests with real life cases are necessary to ensure the effectiveness of the parameter estimation method. Moreover the fact that the process of estimating the parameters requires 15 measurements can also be improved. Indeed, a 8 or 10 measurements for parameter identification would be more easier in practice. It is the main reasons why that parameter identification method was not yet implemented in the web platform but future version will include the feature.

# 9

# CONCLUSION

The breast mechanical behaviour is a complex system to reproduce numerically. The various studies point out that modelling every breast tissue and mechanics is still intractable, hence, every model in literature is based on a set of simplifications. These models showed that they can reproduce the behaviour of breasts with some accuracy. However most of them are very complex and their application is sometimes very restrictive.

The breast model developed in the scope of this project allows to accurately predict the behaviour of the breast, given the surgical parameters (in this particular case breast reduction surgery parameters). The simplicity of the model and the assumptions made, namely the approximate spherical shape of the breast in a state of free stress and the new knitting conditions provide a good approach to the development of a simple breast model that reproduces the behaviour of the breast tissues.

The overall performance of the simulation method is good although in the first versions of the code it could took a few minutes to process data for meshes with just a 1000 nodes. The results of the model performance show the ability of the model to guide surgeons for planing surgeries. It also provides an helpful medical assistant for students to gain some experience and develop a certain sensitivity when studying surgical cases.

A new parameter identification method for estimating the geometrical and mechanical parameters of a breast model in function of measurements performed by doctors during consultations was presented and analysed. Such parameters are involved in the numerical model to accurately evaluate the breast deformation after suturing. The accuracy of the model in representing not only the breast shape and size but the behaviour as well enables the surgeon to evaluate complex situations and adapt procedures. It reduces the probability of making errors and gives a visual expectation of the final shape post surgery. The method allows doctors to handle virtual breast without needing to wait for long (mainly in terms of waiting periods) and expensive exams such as MRI and CT scans.

The optimisation of the code architecture was very successful. With the times of the new simulation it is possible to perform the estimation of parameters more quickly (10 minutes instead of 7 hours for a coarse mesh). The change of the data structures proved to have a great impact in the code. The exploit of the vectorisation units of the modern CPU showed

improvements as well as the parallelisation of the code, with the latter having an higher impact in terms of overall performance. The implementation of a code in the GPU showed an alternative method whose results show a potential usage with more refined meshes in the future.

However, despite the great speedup obtained, the breast simulation code architecture was not good enough to provide a real-time response, specially for the process of estimating the breast parameters. The NN approach offers an alternative way to efficiently assess the mechanical and geometrical parameter of the breast model by returning a very good approximation in real time. The comparisons that were drawn between the two methods clearly point out in favour of a neural network method. Such a conclusion does not mean that classical bio-mechanical models should be discarded.

Despite being very fast, the NN models can sporadically provide poor estimations, while on the other hand, despite being reliable, the iterative method can take hours to estimate the breast parameters. Combining these two methods, one obtains a new hybrid model which has the strengths of both. The NN is used to provide a first instantaneous estimation of the parameters which are then used by the iterative method to refine and ensure the quality of the estimation. The tests show that the NN trained models can obtain a fast estimation of the parameters which are accurate in 99% of the cases, and on the other 1% it reduces the number of iterations required by the iterative method to find an accurate solution. This new approach of estimating the breast parameters show the potential of combining ML methods and models, like NN, together with other numerical methods, as the iterative inverse problem solver, to create more efficient estimators/predictors.

Comparisons are drawn between predictions and real situations to assess the efficiency and the accuracy of the method to provide the final shape of the operated breast. Parameter identifications such as the mechanical parameters of the breast turned to be a crucial goal since their values dramatically change with the person and have to be provided for each patient in order to obtain a more suitable simulation. The estimation of the breast parameters using tuning by iterative experiences indicates that it is possible to approximate the breast model parameters in such a way that the digital breast better fits the real breast. However, such a process can be very time consuming and, for users that are not familiar with the breast model, it is not easy to achieve satisfactory results. The parameter estimation method presented in chapter 7 is quite efficient and demonstrates the potential that this method will have in the future. The number of tests made with real cases is still in progress to completely validate a reliable method but the results are promising.

Simple access and friendly user tools are necessary to enable the surgeon to adopt and efficiently use numerical tools. A Web support simulator for breast knitting has been developed and surgeon experiences were reported. The digital breasts that were used as comparison with the real life cases in section 8.1 were created using this platform. The web

interface has multiple usages that can benefit not only the doctors but also medical students and ultimately the patients.

## 9.1 FUTURE WORK

Despite the good results, there is still work to make the breast model more robust, namely additional tests to consolidate the work already done in this project. There were some considerations made regarding the breast model which concerns its evolution in a posterior phase. This includes the study of other breast structures like Cooper's ligaments or exploring other primitives like splines or ellipsoidal shapes.

Concerning the parameter estimation method, the results obtained are very positive and show that the method has potential for usage in real life cases. There are still some aspects of the method that can be improved, namely the measurements used to estimate the breast parameters which some are not ideal, specially relating to $\mu_{sk}$. In terms of the lack of influence that some measurements have in some parameters, the issue can be solved by considering and testing other measurements like the frontal perimeter of the breast or even the lateral contour of the breast (convexity of the shape) as well as removing the measures that have small to no effect. Moreover the fact that the process of estimating the parameters requires 15 measurements can also be improved. Indeed, a 8 or 10 measurements for parameter identification would be more easier in practice. Besides these issues, there are other aspects that were considered but that are out of the scope of this project which include adding additional parameters to the breast model and finding the least amount of measures in order to obtain a good estimation.

As future work, it would also be interesting to consider other measurement method to assess the values of the bio-mechanical parameters such that the Inverse Problem becomes less sensitive to the volatility of some parameters such as $\mu_{sk}$ (improve the solvability and conditioning). One other aspect is the impact assessment of these models for real-life cases as well as the elaboration of a data set with real life cases.

Additional work includes the development of more features and the improvement of the web platform. It is the main reasons why that parameter identification method was not yet implemented in the web platform but future version will include this feature.

# BIBLIOGRAPHY


Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

J. Allard, S. Cotin, F. Faure, P.-J. Bensoussan, F. Poyer, C. Duriez, H. Delingette, and L. Grisoni. Sofa-an open source framework for medical simulation. In Studies in Health Technology and Informatics, volume 125, pages 13–18, 2007.

AMD. Software optimization guide for amd family 15h processors, 2015.

V. Arbabi, B. Pouran, H. Weinans, and A.A. Zadpoor. Combined inverse-forward artificial neural networks for fast and accurate estimation of the diffusion coefficients of cartilage based on multi-physics models. Journal of Biomechanics, 49(13):2799–2805, 2016. doi: 10.1016/j.jbiomech.2016.06.019.

Taiwo Oladipupo Ayodele. Types of machine learning algorithms. In Yagang Zhang, editor, New Advances in Machine Learning, chapter 3, pages 1–31. Intechopen, 2010. doi: 10.5772/56672.

FS Azar, DX Metaxas, and MD Schnall. A deformable finite element model of the breast for predicting mechanical deformations under external perturbations. Academic Radiology, Volume 8, Issu 10, pages 965–975, 2001. doi: 10.1016/S1076-6332(03)80640-2.

FS Azar, DX Metaxas, and Schnall MD. Methods for modeling and predicting mechanical deformations of the breast under external perturbations. Medical Image Analysis, pages 1–27, 2002. doi: 10.1016/S1361-8415(01)00053-6.

Ahmad Al Badawi, Bharadwaj Veeravalli, Khin Mi Mi Aung, and Brahim Hamadicharef. Accelerating subset sum and lattice based public-key cryptosystems with multi-core cpus and gpus. Journal of Parallel and Distributed Computing, Volume 119, pages 179–190, 2018. doi: 10.1016/j.jpdc.2018.04.014.

S. Biswas and S. Acharyya. Gene expression profiling by estimating parameters of gene regulatory network using simulated annealing: A comparative study. Souvenir of the 2014 IEEE International Advance Computing Conference, IACC 2014, pages 56–61, 2014. doi: 10.1109/IAdCC.2014.6779294.

David Blythe. Rise of the graphics processor. Proceedings of the IEEE, 2008.

OpenMP Architecture Review Board. Openmp application program interface, version 4.5, 2015.

Giuseppe Bonaccorso. Machine Learning Algorithms. Packt Publishing Limited, 2017. ISBN 978-1-78588-962-2.

Joseph F. Bonnans, Jean C. Gilbert, Claude Lemarechal, and Claudia A. Sagastizábal. Newtonian methods. In Numerical Optimization, pages 51–66, 2006. doi: 10.1007/978-3-540-35447-5.

A. Bounaïm, S. Holm, and W. Chenand Å. Ødegråd. Detectability of breast lesions with cari ultrasonography using a bioacoustic computational approach. Computers and Mathematics with Applications, Volume 54, Issue 1, pages 69–106, 2007. doi: 10.1016/j.camwa.2006.03.037.

Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2, pages 690–696, 2000. doi: 10.1109/CVPR.2000.854941.

Randal E. Bryant and David R. O'Hallaron. Computer Systems: A Programmer's Perspective, 3dd Edition. Pearson, 2016. ISBN 9780134092669.

Augusta Cardoso, Gustavo Coelho, Horácio Zenha, Vera Sá, Georgi Smirnov, and Horácio Costa. Computer simulation of breast reduction surgery. Aesthetic Plastic Surgery, pages 68–76, 2013a. doi: 10.1007/s00266-012-9996-9.

Augusta Cardoso, João Martins, Gustavo Coelho, Rui Barbosa, and Horácio Costa. Modified vertical reduction mammaplasty: ten years of experience. European Journal of Plastic Surgery, Volume 37, Number 4, pages 215–226, 2013b. doi: 10.1007/s00238-013-0927-z.

Maria João Cardoso, Jaime Cardoso, Natália Amaral, Isabel Azevedo, Lise Barreau, Mario Bernardo, David Christie, Susy Costa, Florian Fitzal, José L Fougo, Jørgen Johansen, Douglas Macmillan, Maria Piera Mano, Lea Regolo, José Rosa, Luís Teixeira, and Conny Vrieling. Turning subjective into objective: The bcct.core software for evaluation of cosmetic results in breast cancer conservative treatment. The Breast, Volume 16, Issue 5, pages 456–461, 2007. doi: 10.1016/j.breast.2007.05.002.

Marlene Carriço. Doentes vão esperar 90 dias no máximo por uma endoscopia, uma tac ou uma ressonância. Observador, 2017.

P.P. Chang, S.A. Mahlke, and W.W. Hwu. Using profile information to assist classic code optimizations. Software: Practice and Experience, Volume 21, Issue 12, pages 1301–1321, 1991. doi: 10.1002/spe.4380211204.

Huan Chen, Xuefeng Zhao, Xiao Lu, and Ghassan Kassab. Non-linear micromechanics of soft tissues. International Journal of Non-Linear Mechanics, pages 79–85, 2013. doi: 10.1016/j.ijnonlinmec.2013.03.002.

W.Y. Chen, P.P. Chung, and W.-M.W. Hwu. The effect of code expanding optimizations on instruction cache design. IEEE Transactions on Computers, Volume 42, Issue 9, pages 1045–1057, 1993. doi: 10.1109/12.241594.

François Chollet et al. Keras. https://keras.io, 2015.

O. Comas, Z.A. Taylor, J. Allard, S. Ourselin, S. Cotin, and J. Passenger. Efficient nonlinear fem for soft tissue modelling and its gpu implementation within the open source framework sofa. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5104 LNCS:28–39, 2008.

Ivan F. Costa and Remis Balaniuk. Lem - an approach for real time physically based soft tissue simulation. Proceedings - IEEE International Conference on Robotics and Automation, Volume 3, pages 2337–2343, 2001. doi: 10.1109/ROBOT.2001.932971.

A. Peréz del Palomar, B. Calvo, J. Herrero, J. López, and M. Doblaré. A finite element model to accurately predict real deformations of the breast. Medical Engineering and Physics, Volume 30, Issue 9, pages 1089–1097, 2008. doi: 10.1016/j.medengphy.2008.01.005.

H. Delingette. Toward realistic soft-tissue modeling in medical simulation. Proceedings of the IEEE, Volume 86, Issue 3, pages 512–523, 1998. doi: 10.1109/5.662876.

Hervé Delingette and Nicholas Ayache. Soft tissue modeling for surgery simulation. Handbook of Numerical Analysis Volume 12, pages 453–550, 2004. doi: 10.1016/S1570-8659(03)12005-4.

Ayon Dey. Machine learning algorithms: A review. International Journal of Computer Science and Information Technologies, Volume 7, Issue 3, pages 1174–1179, 2016. ISSN 0975-9646.

D. Dharma, C. Jonathan, A.I. Kistidjantoro, and A. Manaf. Material point method based fluid simulation on gpu using compute shader. In Proceedings - 2017 International Conference on Advanced Informatics: Concepts, Theory and Applications, ICAICTA 2017, 2017. doi: 10.1109/ICAICTA.2017.8090962.

Jean Donea, Antonio Huerta, J.-Ph Ponthot, and A. Rodríguez-Ferran. Arbitrary lagrangian-eulerian methods. In Encyclopedia of Computational Mechanics. John Wiley & Sons, Ltd, 2004. doi: 10.1002/0470091355.ecm009.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12:2121–2159, 2011.

G. Dufaye, A. Cherouat, J.M. Bachman, and H. Borouchaki. Advanced finite element modelling for the prediction of 3d breast deformation. European Journal of Computational Mechanics, volume 22, Issue 2-4, pages 170–182, 2013. doi: 10.1080/17797179.2013.820902.

W. A. Edelstein, M. Mahesh, and J. A. Carrino. Mri: Time is dose—and money and versatility. Journal of the American College of Radiology, Volume 7, Issue 8, pages 650–652, 2010. doi: 10.1016/j.jacr.2010.05.002.

Maximilian Eder, Stefan Raith, Jalil Jalali, Alexander Volf, Markus Settles, Hans-Günther Machens, and Laszlo Kovacs. Comparison of different material models to simulate 3-d breast deformations using finite element analysis. Annals of Biomedical Engineering, Volume 42, Issue 4, pages 843–857, 2014. doi: 10.1007/s10439-013-0962-8.

Maximillian Eder, Fee v. Waldenfels, Alexandra Swobodnik, Markus Klöppel, Ann-Kathrin Pape, Tibor Schuster, Stefan Raith, Elena Kitzler, Nikolaos A. Papadopulos, Hans-Günther Machens, and Laszlo Kovacs. Objective breast symmetry evaluation using 3-d surface imaging. The Breast, Volume 21, Issue 2, pages 152–158, 2012. doi: 10.1016/j.breast.2011.07.016.

Björn Eiben, Vasileios Vavourakis, John H. Hipwell, Sven Kabus, Thomas Buelow, Cristian Lorenz, Thomy Mertzanidou, Sara Reis, Norman R. Williams, Mohammed Keshtgar, and David J. Hawkes. Symmetric biomechanically guided prone-to-supine breast image registration. Annals of Biomedical Engineering, Volume 44, Issue 1, pages 154–173, 2016. doi: 10.1007/s10439-015-1496z.

Encyclopaedia Britannica: mammary gland. Mammary gland, 2018. URL https://www.britannica.com/science/mammary-gland.

N. Faria, R. Silva, and J.L. Sobral. Impact of data structure layout on performance. Proceedings of the 2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, pages 116–120, 2013. doi: 10.1109/PDP.2013.24.

Jacques Ferlay, Isabelle Soerjomataram, Rajesh Dikshit, Sultan Eser, Colin Mathers, Marise Rebelo, Donal Maxwell Parkin, David Forman, and Freddie Bray. Cancer incidence and mortality worldwide: Sources, methods and major patterns in globocan 2012.

International Journal of Cancer Volume 136, Issue 5, pages 359–386, 2014. doi: 10.1002/ijc.29210.

Thomas Filleron. Comparing sensitivity and specificity of medical imaging tests when verification bias is present: The concept of relative diagnostic accuracy. European Journal of Radiology, Volume 98, pages 32–35, 2018. doi: 10.1016/j.ejrad.2017.10.022.

Kemal Findikcioglu, Fulya Findikcioglu, Selahattin Ozmen, and Tuba Guclu. The impact of breast size on the vertebral column: A radiologic study. Aesthetic Plastic Surgery Volume 31, Issue 1, pages 23–27, 2007. doi: 10.1007/s00266-006-0178-5.

K Bo Foreman, Leland E. Dibble, John Droge, Randy Carson, and W Bradford Rockwell. The impact of breast reduction surgery on low-back compressive forces and function in individuals with macromastia. Plastic and Reconstructive Surgery, Volume 124, Issue 5, pages 1393–1399, 2009. doi: 10.1097/PRS.0b013e3181b988aa.

A. Gefen and B. Dilmoney. Mechanics of the normal woman's breast. Technology and Health Care, pages 259–271, 2007. ISSN 09287329.

J. L. Gennisson, T. Deffieux, M. Fink, and M. Tanter. Ultrasound elastography: Principles and techniques. Diagnostic and Interventional Imaging, Volume 94, Issue 5, pages 487–495, 2013. doi: 10.1016/j.diii.2013.01.022.

Sarah Gibson, Joe Samosky, Andrew Mor, Christina Fyock, Eric Grimson, Takeo Kanade, Ron Kikinis, Hugh Lauer, Neil McKenzie, Shin Nakajima, Hide Ohkami, Randy Osborne, and Akira Sawada. Simulating arthroscopic knee surgery using volumetric object representations, real-time volume rendering and haptic feedback. Lecture Notes in Computer Science (including subseries Lecture Notes in Artifical Intelligence and Lecture Notes in Bioinformatics), Volume 1205, pages 369–378, 1997. doi: 10.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org.

Susan L. Graham, Peter B. Kessler, and Marshall K. Mckusick. Gprof: A call graph execution profiler. SIGPLAN Not., Volume 17, Issue 6, pages 120–126, 1982. doi: 10.1145/872726.806987.

Salvatore Di Gregorio, Giuseppe Filippone, William Spataro, and Giuseppe A.Trunfio. Accelerating wildfire susceptibility mapping through gpgpu. Journal of Parallel and Distributed Computing, Volume 73, Issue 8, pages 1183–1194, 2013. doi: 10.1016/j.jpdc.2013.03.014.

K. Guo, F. Xu, T. Yu, X. Liu, Q. Dai, and Y. Liu. Real-time geometry, albedo, and motion reconstruction using a single rgb-d camera. ACM Transactions on graphics, Volume 36, Issue 3, 2017. doi: 10.1145/3083722.

Merih Guray and Aysegul A. Sahin. Benign breast diseases: Classification, diagnosis, and management. The Oncologist Volume 11, number 5, pages 435–449, 2006. doi: 10.1634/theoncologist.11-5-435.

Azam Hamidinekoo, Erika Denton, Andrik Rampun, Kate Honnor, and Reyer Zwiggelaar. Deep learning in mammography and breast histology, an overview and future trends. Medical Image Analysis, Volume 47, pages 45–67, 2018. doi: 10.1016/j.media.2018.03.006.

Lianghao Han, John H Hipwell, Christine Tanner, Zeike Taylor, Thomy Mertzanidou, Jorge Cardoso, Sebastien Ourselin, and David J Hawkes. Development of patient-specific biomechanical models for predicting large breast deformation. Physics in Medicine & Biology, pages 455–472, 2012a. doi: 10.1088/0031-9155/57/2/455.

Lianghao Han, ohn Hipwell, Zeike Taylor, Christine Tanner, Sebastien Ourselin, and David J. Hawkes. Fast deformation simulation of breasts using gpu-based dynamic explicit finite element method. In Digital Mammography, pages 728–735. Springer Berlin Heidelberg, 2012b. doi: 10.1007/978-3-642-13666-5_98.

John L. Hennessy and David A. Patterson. Computer Architecture A Quantitative Approach, 6th Edition. Morgan Kaufmann Publishers, 2017. ISBN 9780128119051.

Helga Henseler, Balvinder S. Khambay, Adrian Bowman, Joanna Smith, J. Paul Siebert, Susanne Oehler, Xiangyang Ju, Ashraf Ayoub, and Arup K. Ray. Investigation into accuracy and reproducibility of a 3d breast imaging system using multiple stereo cameras. Journal of Plastic, Reconstructive & Aesthetic Surgery, Volume 64, Issue 5, pages 577–582, 2011. doi: 10.1016/j.bjps.2010.08.044.

Helga Henseler, Alina Kuznetsova, Peter Vogt, and Bodo Rosenhahn. Validation of the kinect device as a new portable imaging system for three-dimensional breast assessment. Journal of Plastic, Reconstructive & Aesthetic Surgery, pages 483–488, 2014. doi: 10.1016/j.bjps.2013.12.025.

Helga Henseler, Sarah Kim Bonkat, Peter Maria Vogt, and Bodo Rosenhahn. The kinect recording system for objective three- and four-dimensional breast assessment with image overlays. Journal of Plastic, Reconstructive & Aesthetic Surgery, Volume 69, Issue 2, pages 27–34, 2016. doi: 10.1016/j.bjps.2015.10.021.

Gerhard A. Holzapfel. Biomechanics of soft tissue. In Handbook of Materials Behaviour Models. Academic Press, 2001. doi: 10.1016/B978-012443341-0/50107-1.

T. Hopp, M. Dietzel, P.A. Baltzer, P. Kreisel, W.A. Kaiser, H. Gemmeke, and N.V. Ruiter. Automatic multimodal 2d/3d breast image registration using biomechanical fem models and intensity-based optimization. Medical Image Analysis, pages 209–218, 2013. doi: 10.1016/j.media.2012.10.003.

Libo Huang, Nong Xiao, Zhiying Wang, Yongwen Wang, and Mingche Lai. Efficient multimedia coprocessor with enhanced simd engines for exploiting ilp and dlp. Parallel Computing, Volume 39, Issue 10, pages 586–602, 2013. doi: 10.1016/j.parco.2013.06.004.

M.B. Huber, S.L. Lancianese, M.B. Nagarajan, I.Z. Ikpot, A.L. Lerner, and A. Wismüller. Prediction of biomechanical properties of trabecular bone in mr images with geometric features and support vector regression. IEEE Transactions on Biomedical Engineering, 58 (6):1820–1826, 2011. doi: 10.1109/TBME.2011.2119484.

Intel. Intel 64 and ia-32 architectures optimization reference manual, 2016.

Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. UIST'11 - Proceedings of the 24th Annual ACM Symposiuum on User Interface Software and Technology, pages 559–568, 2011. doi: 10.1145/2047196.2047270.

Robert A. Jesinger. Breast anatomy for the interventionalist. Techniques in Vascular and Interventional Radiology, pages 3–9, 2014. doi: 10.1053/j.tvir.2013.12.002.

Michael J.Gibson, Edward C.Keedwell, and Dragan A. Savić. An investigation of the efficient implementation of cellular automata on multi-core cpu and gpu hardware. Journal of Parallel and Distributed Computing, Volume 77, pages 11–25, 2015. doi: 10.1016/j.jpdc.2014.10.011.

Stian F. Johnsesn, Zeike A. Taylor, Matthew J. Clarkson, John Hipwell, Marc Modat, Bjoern Eiben, Lianghao Han, Yipeng Hu, Thomy Mertzanidou, David J. Hawkes, and Sebastien Ourselin. Niftysim: A gpu-based nonlinear finite element package for simulation of soft tissue biomechanics. International Journal of Computer Assisted Radiology and Surgery, Volume 10, Issue 7, pages 1077–1095, 2015. doi: 10.1007/s11548-014-1118-5.

R. Kayar, S. Civelek, M. Cobanoglu, O. Gungor, H. Catal, and M. Emiroglu. Five methods of breast volume measurement: A comparative study of measurements of specimen volume in 30 mastectomy cases. Breast Cancer: Basic and Clinical Research, 5(1):43–52, 2011. doi: 10.4137/BCBCR.S6128.

A: Kaye, R. F. T. Stepto, W. J. Work, J. V. Alemán, and A. Ya. Malkin. Definition of terms relating to the non-ultimate mechanical properties of polymers. Pure and Applied Chemistry, Volume 70, Issue 3, pages 701–754, 1998. doi: 10.1351/pac199870030701.

Minhaj Ahmad Khan. Improving performance through deep value profiling and specialization with code transformation. Computer Languages, Systems & Structures, pages 193–203, 2011. doi: 10.1016/j.cl.2011.08.001.

Khan Academy: Hooke's Law. Hooke's law, 2018. URL www.khanacademy.org/science/physics/work-and-energy/hookes-law/a/what-is-hookes-law.

M Kimura and H. Saito. 3d reconstruction based on epipolar geometry. IEICE Transactions on Information and Systems, Volume E84-D, Issue 12, pages 1690–1697, 2000. ISSN 09168532.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. CoRR, abs/1412.6980, 2014. URL http://arxiv.org/abs/1412.6980.

T.A. Krouskop, T.M. Wheeler, F. Kallel, B.S. Garra, and T. Hall. Elastic moduli of breast and prostate tissues under compression. Ultrasonic Imaging, Volume 20, Issue 4, pages 260–274, 1998. doi: 10.1177/016173469802000403.

K. Langer. On the anatomy and physiology of the skin. British Journal of Plastic Surgery, Volume 31, Issue 1, pages 3–8, 1978. doi: 10.1016/0007-1226(78)90003-6.

Oren Lapid, E Joline de Groof, Leonard UMC Corion, Mark JC Smeulders, and Chantal MAM van der Horst. The effect of breast hypertrophy on patient posture. Archives of Plastic Surgery, pages 559–563, 2013. doi: 10.5999/aps.2013.40.5.559.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, pages 436–444, 2015. doi: 10.1038/nature14539.

Zhongyu Li, Xiaofan Zhang, Henning Müller, and Shaoting Zhang. Large-scale retrieval for medical image analytics: A comprehensive review. Medical Image Analysis, Volume 43, pages 66–84, 2018. doi: 10.1016/j.media.2017.09.007.

A. Lisjak, O.K. Mahabadi, L. He, B.S.A. Tatone, P. Kaifosh, S.A. Haque, and G. Grasselli. Acceleration of a 2d/3d finite-discrete element code for geomechanical simulations using general purpose gpu computing. Computers and Geotechnics, 100:84–96, 2018. doi: 10.1016/j.compgeo.2018.04.011.

Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, and Clara I. Sánchez. A survey on deep learning in medical image analysis. Medical Image Analysis, Volume 42, pages 60–88, 2017. doi: 10.1016/j.media.2017.07.005.

André Liu and António Fernandes. Open source debuggers and integration with a 3d engine. In Encontro Português de Computação Gráfica 2014, 2014.

Yan-Lin Liu, Pei-Yuan Liu, Mei-Lan Huang, Jui-Ting Hsu, Ruo-Ping Han, and Jay Wu. Simulation of breast compression in mammography using finite element analysis: A preliminary study. Radiation Physics and Chemistry, Volume 140, pages 295–299, 2017. doi: 10.1016/j.radphyschem.2017.01.017.

Yuen-Jong Liu. Aesthetics of the female breast: Correlation of pluralistic evaluations with volume and surface area. Master's thesis, Yale University, 2009.

D. Lopes, S. Clain, R.M.S. Pereira, G.J. Machado, G. Smirnov, and I. Vasilevskiy. Numerical simulation of breast reduction with a new knitting condition. International Journal for Numerical Methods in Biomedical Engineering, Volume 33, Issue 2, 2017a. doi: 10.1002/cnm.2796.

Diogo Lopes, Stéphane Clain, Augusta Cardoso, Igor Vasilevskiy, and Georgi Smirnov. Breast simulation applications in real-life cases. In $7^o$ Congresso Nacional de Biomecânica, Guimarães, Portugal, Janeiro 2017b.

D.G. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, Volume 60, Issue 2, pages 91–110, 2004. doi: 10.1023/B:VISI:0000029664.99615.94.

Hamid Mahmoodian and Leila Ebrahimian. Using support vector regression in gene selection and fuzzy rule generation for relapse time prediction of breast cancer. Biocybernetics and Biomedical Engineering, 36(3):466 – 472, 2016. ISSN 0208-5216. doi: https://doi.org/10.1016/j.bbe.2016.03.003.

Ana Maia. Doentes esperam 16 meses para fazer uma ressonância no sns. Diário de Notícias Online, 2016.

Yogesh K. Mariappan, Kevin J. Glaser, and Richard L. Ehman. Magnetic resonance elastography: A review. Clinical Anatomy, Volume 23, Issue 5, pages 497–511, 2010. doi: 10.1002/ca.21006.

Michelle Marino. Constitutive modeling of soft tissues. In Reference Module in Biomedical Sciences. Elsevier, 2018. doi: 10.1016/B978-0-12-801238-3.99926-4.

Filipe Marques, Stéphane Clain, Gaspar J. Machado, Bruno Martins, Olga S. Carneiro, and João M. Nóbrega. A novel heat transfer coefficient identification methodology for the profile extrusion calibration stage. Applied Thermal Engineering, Volume 103, pages 102–111, 2016. doi: 10.1016/j.applthermaleng.2016.04.013.

F. Martínez-Martínez, M.J. Rupérez-Moreno, M. Martínez-Sober, J.A. Solves-Llorens, D. Lorente, A.J. Serrano-López, S. Martínez-Sanchis, C. Monserrat, and J.D. Martín-

Guerrero. A finite element-based machine learning approach for modeling the mechanical behaviour of the breast tissues under compression in real-time. Computers in Biology and Medicine, Volume 90, pages 116–124, 2017. doi: 10.1016/j.compbiomed.2017.09.019.

Walter Maurel, Daniel Thalmann, Yin Wu, and Nadia Magnenat Thalmann. Biomechanical Models for Soft Tissue Simulation. Springer, Berlin, Heidelberg, 1998. doi: 10.1007/978-3-662-03589-4. Esprit Basic Research Series.

Robert S. McCuskey, Stephen W. Carmichael, and Darrell G. Kirch. The importance of anatomy in health professions education and the shortage of qualified educators. Academic Medicine, Volume 80, Issue 4, pages 349–351, 2005.

A. Melo and Red Hat. The new linux 'perf' tools, 2010.

Catherine Ménard, Yves Livert, Delphine Canivet, Laetitia Van Achte, Christine Farvacques, Aurore Liénard, Isabelle Merckaert, Christine Reynaert, Jean-Louis Slachmuylder, Jean-François Durieux, Jean Klastersky, and Darius Razavi. Development of the multidimensional analysis of patient outcome predictions (md.pop) during medical encounters. Patient Education and Counseling, Volume 101, Issue 1, pages 52–58, 2018. doi: 10.1016/j.pec.2017.07.004.

S. Mitra and C. Balaji. A neural network based estimation of tumour parameters from a breast thermogram. International Journal of Heat and Mass Transfer, 53(21-22):4714–4727, 2010. doi: 10.1016/j.ijheatmasstransfer.2010.06.020.

M.Kuhlmann, E.C. Fear, A. Ramirez-Serrano, and S. Federico. Mechanical model of the breast for the prediction of deformation during imaging. Medical Engineering & Physics, Volume 35, Issue 4, pages 470–478, 2013. doi: 10.1016/j.medengphy.2012.06.012.

M.R. Mohammadi, E. Fatemizadeh, and M.H. Mahoor. A joint dictionary learning and regression model for intensity estimation of facial aus. Journal of Visual Communication and Image Representation, 47:1 – 9, 2017. ISSN 1047-3203. doi: https://doi.org/10.1016/j.jvcir.2017.05.002.

G.J. Narlikar and G.E. Blelloch. Pthreads for dynamic and irregular parallelism. Proceedings of the 1998 ACM/IEEE Conference on Supercomputing, 1998. doi: 10.1109/SC.1998.10005.

J. Nickolls and W.J. Dally. The gpu computing era. IEEE Micro, 30(2):56–69, 2010. doi: 10.1109/MM.2010.41.

LC Nuzzi, FE Cerrato, ML Webb, HR Faulkner, EM Walsh, AD DiVasta, AK Greene, and BI Labow. Psychological impact of breast asymmetry on adolescents: A prospective

cohort study. Plastic and Reconstructive Surgery, Volume 134, Issue 6, pages 1116–1123, 2014. doi: 10.1097/PRS.0000000000000736.

NVIDIA. Nvidia tesla v100 gpu architecture, 2017.

Helder P. Oliveira, Jaime S. Cardoso, Andre Magalhães, and Maria J. Cardoso. Methods for the aesthetic evaluation of breast cancer conservation treatment: A technological review. Current Medical Imaging Reviews, Volume 9, Issue 1, pages 32–46, 2013. doi: 10.2174/1573405611309010006.

OpenGL: Compute Shader. Compute shader, 2018. URL https://www.khronos.org/opengl/wiki/Compute_Shader.

J.D. Owens, M. Houston, D. Luebke, S. Green, J.E. Stone, and J.C. Phillips. Gpu computing. Proceedings of the IEEE, 96(5):879–899, 2008. doi: 10.1109/JPROC.2008.917757.

G.C. Papanicolaou and S.P. Zaoutsos. Viscoelastic constitutive modeling of creep and stress relaxation in polymers and polymer matrix composites. In Creep and Fatigue in Polymer Matrix Composites. Woodhead Publishing, 2001. ISBN 978-1-84569-656-6.

Guillaume Picinbono, Jean-Christophe Lombardo, Hervé Delingette, and Nicholas Ayache. Improving realism of a surgery simulator : Linear anisotropic elasticity, complex interactions and force extrapolation. Journal of Visualization and Computer Animation banner, Volume 13, Issue 3, pages 147–167, 2002. doi: 10.1002/vis.257.

Stefanie T.L. Pöhlmann, Elaine Harkness, Christopher J. Taylor, Ashu Gandhi, and Susan M. Astley. Preoperative implant selection for unilateral breast reconstruction using 3d imaging with the microsoft kinect sensor. Journal of Plastic, Reconstructive & Aesthetic Surgery, Volume 70, Issue 8, pages 1059–1067, 2017. doi: 10.1016/j.bjps.2017.04.005.

Steven P. Poplack, Keith D. Paulsen, Alexander Hartov, Paul M. Meaney, Brian W. Pogue, Tor D. Tosteson, Margaret R. Grove, Sandra K. Soho, and Wendy A. Wells. Electromagnetic breast imaging: Average tissue property values in women with negative clinical findings. Radiology, Volume 231, Issue 2, pages 571–580, 2004. doi: 10.1148/radiol.2312030606.

Akizur Rahman and Ravie Chandren Muniyandi. Review of gpu implementation to process of rna sequence on cancer. Informatics in Medicine Unlocked, Volume 10, pages 17–26, 2018. doi: 10.1016/j.imu.2017.10.008.

Vijay Rajagopal, Angela Lee, Jae-Hoon Chung, Ruth Warren, Ralph P. Highnam, Martyn P. Nash, and Poul M. F. Nielsen. Creating individual-specific biomechanical models of the breast for medical image analysis. Academic Radiology, pages 1425–1436, 2008. doi: 10.1016/j.acra.2008.07.017.

Vijayaraghavan Rajagopal, Jae-Hoon Chung, David Bullivant, Poul M. F. Nielsen, and Martyn P. Nash. Determining the finite elasticity reference state from a loaded configuration. International Journal For Numerical Methods In Engineering, pages 1434–1451, 2007. doi: 10.1002/nme.2045.

M. Salzmann and R. Urtasun. Beyond feature points: Structured prediction for monocular non-rigid 3d reconstruction. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Volume 7575, Part 4, pages 245–259, 2012. doi: 10.1007/978-3-642-33765-9_18.

E. Saniei, S. Setayeshi, M.E. Akbari, and M. Navid. Parameter estimation of breast tumour using dynamic neural network from thermal pattern. Journal of Advanced Research, 7 (6):1045–1055, 2016. doi: 10.1016/j.jare.2016.05.005.

GR Scott, CL Carson, and GL Borah. Maximizing outcomes in breast reduction surgery: a review of 518 consecutive patients. Plastic and Reconstructive Surgery, Volume 116, Issue 6, pages 1633–1639, 2005. doi: 10.1097/01.prs.0000187145.44732.1b.

B. Seligman, S. Tuljapurkar, and D. Rehkopf. Machine learning approaches to the social determinants of health in the health and retirement study. SSM - Population Health, 4: 95–99, 2018. doi: 10.1016/j.ssmph.2017.11.008.

Shai Shalev-Shwartz and Shai Ben-David. Understanding Machine Learning. Cambridge University Press, 2014.

Rosa M.S. Sigrist, Joy Liau, Ahmed El Kaffas, Maria Cristina Chammas, and Juergen K. Willmann. Ultrasound elastography: Reviw of techniques and clinical applications. Theranostics, pages 1303–1329, 2017. doi: 10.7150/thno.18650.

MacArthur L. Stewart, Lorenzo M. Smith, and Neal Hall. A numerical investigation of breast compression: Lesion modeling. International Journal of Contemporary Advanced Mathematics , Volume 2, Issue 1, pages 9–21, 2011. doi: 10.1109/TBME.2011.2162063.

Vukasin Strbac, Jos Vander Sloten, and Nele Famaey. Analyzing the potential of gpgpus for real-time explicit finite element analysis of soft tissue deformation using cuda. Finite Elements in Analysis and Design, Volume 105, pages 78–89, 2015. doi: 10.1016/j.finel.2015.07.005.

Wenquing Sun, Tzu-Liang (Bill) Tseng, Jianying Zhang, and Wei Qian. Enhancing deep convolutional neural network scheme for breast cancer diagnosis with unlabeled data. Computerized Medical Imaging and Graphics, Volume 57, pages 4–9, 2017. doi: 10.1016/j.compmedimag.2016.07.004.

Eric Swanson. Evidence-Based Cosmetic Breast Surgery. Springer, 2017. ISBN 978-3-319-53957-7.

H. Talbot, N. Haouchine, I. Peterlik, J. Dequidt, C. Duriez, H. Delingette, and S. Cotin. Surgery planning and guidance using the sofa framework. In Eurographics, Zurich, Switzerland, May 2015.

Z.A. Taylor, M. Cheng, and S. Ourselin. High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. Medical Imaging, IEEE Transactions on, 27(5):650–663, 2008. ISSN 0278-0062. doi: 10.1109/TMI.2007.913112.

Z.A. Taylor, M. Cheng, and S. Ourselin. Real-time surgical simulation using reduced order finite element analysis. In Tianzi Jiang, Nassir Navab, JosienP.W. Pluim, and MaxA. Viergever, editors, Medical Image Computing and Computer-Assisted Intervention – MICCAI 2010, volume 6362 of Lecture Notes in Computer Science, pages 388–395. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15744-8. doi: 10.1007/978-3-642-15745-5_48.

Zeike A. Taylor, Mario Cheng, and Sébastien Ourselin. Real-time nonlinear finite element analysis for surgical simulation using graphics processing units. In Nicholas Ayache, Sébastien Ourselin, and Anthony Maeder, editors, Medical Image Computing and Computer-Assisted Intervention – MICCAI 2007, volume 4791 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-75756-6. doi: 10.1007/978-3-540-75757-3_85.

C. Thirumalai and R. Manzoor. Cost optimization using normal linear regression method for breast cancer type i skin. Proceedings of the International Conference on Electronics, Communication and Aerospace Technology, ICECA 2017, 2017-January:264–268, 2017. doi: 10.1109/ICECA.2017.8212813.

Winnie Thomas and Rohin D. Daruwala. Application aware scalable architecture for gpgpu. Journal of Systems Architecture, Volume 89, pages 73–83, 2018. doi: 10.1016/j.sysarc.2018.07.003.

T. Tieleman and G. Hinton. Lecture 6e — rmsprop: Divide the gradient by a running average of its recent magnitude., 2012. COURSERA: Neural Networks for Machine Learning.

M. Tonutti, G. Gras, and G.-Z. Yang. A machine learning approach for real-time modelling of tissue deformation in image-guided neurosurgery. Artificial Intelligence in Medicine, 80:39–47, 2017. doi: 10.1016/j.artmed.2017.07.004.

A.M. van der Zijden, B.E. Groen, E. Tanck, B. Nienhuis, N. Verdonschot, and V. Weerdesteyn. Estimating severity of sideways fall using a generic multi linear regression model based

on kinematic input variables. Journal of Biomechanics, 54:19 – 25, 2017. ISSN 0021-9290. doi: https://doi.org/10.1016/j.jbiomech.2017.01.033.

V. Vavourakis, B. Eiben, J. H. Hipwell, N. R. Williams, M. Keshtgar, and D. J. Hawkes. Multiscale mechano-biological finite element modelling of oncoplastic breast surgery - numerical study towards surgical planning and cosmetic outcome prediction. PLoS ONE, Volume 11, Issue 7, 2016. doi: 10.1371/journal.pone.0159766.

A. Vidaki, D. Ballard, A. Aliferi, T.H. Miller, L.P. Barron, and D. Syndercombe Court. Dna methylation-based forensic age prediction using artificial neural networks and next generation sequencing. Forensic Science International: Genetics, 28:225–236, 2017. doi: 10.1016/j.fsigen.2017.02.009.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 1, pages 1511–1518, 2001.

Parris S. Wellman, Robert D. Howe, Edward Dalton, and Kenneth A. Kern. Breast tissue stiffness in compression is correlated to histological diagnosis. Technical report, Harvard Bio Robotics Laboratory, Division of Engineering and Applied Sciences, Harvard University, 1999.

Peter N. T. Wells and Hai-Dong Liang. Medical ultrasound: imaging of soft tissue strain and elasticity. Journal of the Royal Society Interface, Volume 8, Issue 64, pages 1521–1549, 2011. doi: 10.1098/rsif.2011.0054.

J.S. Wheat, S. Choppin, and A. Goyal. Development and assessment of a microsoft kinect based system for imaging the breast in three dimensions. Medical Engineering & Physics, Volume 36, Issue 6, pages 732–738, 2014. doi: 10.1016/j.medengphy.2013.12.018.

H.M. Yin, L.Z. Sun, G. Wang, T. Yamada, J. Wang, and M.W. Vannier. Imageparser: a tool for finite element generation from three-dimensional medical images. BioMedical Engineering Online, Volume 3, 2004. doi: 10.1186/1475-925X-3-31.

H. Zhong and T. Peters. A real time hyperelastic tissue model. Computer Methods in Biomechanics and Biomedical Engineering, Volume 10, Issue 3, pages 185–193, 2007. doi: 10.1080/10255840701292732.

Yi Zhou, Fazhi He, Neng Hou, and Yimin Qiu. Parallel ant colony optimization on multi-core simd cpus. Future Generation Computer Systems, Volume 72, Part 2, pages 873–783, 2018. doi: 10.1016/j.future.2017.09.073.

Yun Zhou, Lori Boyd, and Celeste Lawson. Errors in medical imaging and radiography practice: A systematic review. Journal of Medical Imaging and Radiation Sciences, pages 435–441, 2015. doi: 10.1016/j.jmir.2015.09.002.

Fan Zhu, David Rodriguez Gonzalez, Trevor Carpenter, Malcolm Atkinson, and Joanna Wardlaw. Parallel perfusion imaging processing using gpgpu. Computer Methods and Programs in Biomedicine, Volume 108, Issue 3, pages 1012–1021, 2012. doi: 10.

M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, and M. Stamminger. Real-time non-rigid reconstruction using an rgb-d camera. ACM Transactions on Graphics, Volume 33, Issue 4, 2014. doi: 10.1145/2601097.2601165.

Jack Zoumaras and Julie Lawrence. Inverted-t versus vertical scar breast reduction: One surgeon's 5-year experience with consecutive patients. Aesthetic Surgery Journal, Volume 28, Issue 5, pages 521–526, 2008. doi: 10.1016/j.asj.2008.07.003.