UNIVERSITÀ degli STUDI di CATANIA

Corso di Laurea in
Scienze e Tecnologie Alimentari
Dipartimento D*i*3A
Sezione Meccanica e Meccanizzazione

CLAUDIA AGLIECO

First results on the assessment of impact damage of vegetables by means of the FEM approach

TESI PER IL CONSEGUIMENTO DEL TITOLO DI
DOTTORE DI RICERCA

Tutor: Chiar.mo Prof. Emanuele Cerruto

Coordinatore: Chiar.ma Prof.ssa Simona Consoli

Dottorato di ricerca in Ingegneria Agraria - XXVI ciclo
ANNO 2014

A mia nonna,
che non seppe insegnarmi nulla
tranne quella che sono
(cit. A.C.)

Acknowledgements

At this successful completion of this doctoral research work I naturally would like to take this moment to pause, reflect and acknowledge all those people who have in one way or another been very helpful throughout my period of stay and subsequent production of this thesis.

First of all I would like to extend my heartfelt gratitude to my professor, Prof. Emanuele Cerruto, for taking a chance with me; for the enormous support (technical and moral) and encouragement; for his kindness and for the inexplicable and overwhelming confidence he expended on me. The amount of knowledge and academic/scientific maturity I was able to gain under him is simply beyond any measure. For these I am, and will always be, truly grateful.

My sincere gratitude also goes to the Institut für Agrartechnik, Potsdam-Bornim (ATB) for the good opportunity to use their prestigious laboratory for my research and to work in a international and stimulating working environment. In particular I would like to thank the Prof. Dr. Ing. Klaus-Günter Gottschalk who, in addition to his kindness in accepting to be my co-professor, has also helped facilitate during my stay in Potsdam-Bornim. I will always appreciate the cheerful words of encouragement and advice he has always offered at every opportunity.

Last, but not least, I wish to thank my family for the constant support and love.

Abstract

Mechanical harvest and post-harvest handling induce numerous mechanical impacts on vegetables. These impacts may cause damage such as black-spot bruise, resulting in severe economic losses.

Impact forces and accelerations arising from collisions, are among the main indices taken into account when studying the damage of fruit and vegetables during post-harvest activities. A miniaturised Acceleration Measuring Unit (AMU) has been recently developed at the Institut für Agrartechnik, Potsdam-Bornim (ATB): when implanted into a real product like a potato tuber, it is able to measure the accelerations at the centre of the fruit deriving from a impact.

This PhD Thesis represents a first contribution on the study of mechanical impacts of vegetables (potato tubers), arising from mechanical harvest and post-harvest handling, by means of simulations based on the Finite Element Method (FEM) approach. Simulations were developed by using the Linux distribution `CAElinux2011`, that contains several technical-engineering software, among which stand out Salome-Meca and Code-Aster. Salome-Meca was used for modelling, meshing and post-processing activities, while Code-Aster was used for processing models.

The work has been developed in collaboration with the Institut für Agrartechnik, Potsdam-Bornim (ATB), Germany, where they were conducted laboratory tests (drop tests to measure impact forces and texture analyses to measure modulus of Young) with two spherical artificial fruits.

After the development of several preliminary simple models to gain familiarity with the computational software Salome-Meca and Code-Aster and to acquire an acceptable agreement between simulated and experimental

tests, it was carried out an extensive set of drop test simulations with a spherical artificial fruit aimed at evaluating the effects of drop height, size of the fruit, density and modulus of Young of the material, on the impact indices (maximum impact force and maximum acceleration at the centre of the fruit). Simulated material parameters were chosen to approach potato tubers properties.

All the factors examined (drop height, sphere diameter, modulus of Young and density of the material, mass of the fruit) affected the maximum impact force and the maximum acceleration at the centre of the sphere. Their increase always caused an increase in the maximum impact force, whereas the maximum acceleration at the centre of the sphere decreased vs sphere diameter, material density and mass, and increased vs drop height and modulus of Young. The decreasing trends are due to the cushioning effect produced by the sphere material itself.

Moreover, the maximum impact forces reported in the experimental results by Geyer *et al.* (2009), referring to drop tests of potato tubers onto steel plates, are in good agreement with the values of impact forces provided by the simulations. Instead, simulations provided acceleration values about twice as many those measured in the experimental results with the AMU device. This difference could be due to the implantation system of the AMU inside the tuber. In fact, comparing the measured impact force and the force computed by means the second law of Newton $(F = m \cdot a)$, Geyer *et al.* in the cited work report that the computed force was approximately half the measured one, meaning an under-estimation of the acceleration provided by the AMU.

Ultimately, the concordance between measured and simulated impact forces confirmed the validity of FEM approach, although the limitations owing to the simplicity of the model developed in this work.

Contents

List of Figures

List of Tables

Foreword

This PhD Thesis represents a first contribution on the study of mechanical impacts on vegetables (potato tubers) arising from mechanical harvest and post-harvest handling.

The impacts may cause damage such as black spot bruise, resulting in severe economic losses. Besides the direct effect of fruit bruising on vegetables quality, and, hence, the visual quality appreciation by the consumer, an important indirect effect can be identified as damaged tissue, a good avenue for entrance of decay organisms, which decrease the shelf-life of the fruit drastically.

A computer simulation technique, named Finite Element Method (FEM), was applied to get insight in the mechanical impact. FEM is essentially a numerical method to find approximate solution of problems in engineering. FEM is best understood from its practical application, known as Finite Element Analysis (FEA).

Finite element analysis is the modelling of products and systems in a virtual environment, for the purpose of finding and solving potential (or existing) structural or performance issues. FEA is used by engineers and scientists to mathematically model and numerically solve very complex structural, fluid, and multi-physics problems. FEA software can be utilised in a wide range of industries, but it is most commonly used in the aeronautical, biomechanical and automotive industries.

The assessment of impact damage of vegetables by means of FEM is the aim of this Thesis, that has been developed in collaboration with the Institut für Agrartechnik, Potsdam-Bornim (ATB), Germany. I spent a part of my doctorate in Potsdam under the supervision of the Prof. Dr. Ing. Klaus-Günter

Gottschalk, Senior scientist of Postharvest Technology Department.

The Thesis is structured in a first part relating to theoretical notions about the post-harvest and the FEM approach, whereas the latter end is focused on the experimental activity.

The first chapter focuses on the impact damage of potato tubers and on the development of potato-shaped instrumented devices to locate bruising risk zones and to prevent quality losses of commodity.

The second chapter presents some basic concept of FEM: what is FEM and the main steps required to develop a computational model using the FEM approach.

The third chapter is dedicated to the materials and methods of the research. In particular, the early paragraphs present the software used to develop the FEA and the preliminary studies developed to gain familiarity with it, while the remainder paragraphs show the experimental activity, dealing with the assessing, by means of drop test simulations of spherical pseudo-fruits, of the effects of material properties (density, size, modulus of elasticity, mass) and drop height on some impact indexes (impact force and acceleration).

Final, the last chapter discuss the results of the simulation tests.

Chapter 1

Post harvest of fruit and vegetables

1.1 General aspects

All fruit and vegetable products are subject to different stress levels both during harvest and during subsequent post-harvest processing. This stress sometimes becomes so great to cause damage to the produce, compromising its preservability (Lewis *et al.*, 2007). Even low levels of damage can bring about considerable economic loss, jeopardizing storage of the produce due to the risk of rotting that can then extend to entire batches.

It therefore becomes important, above all, to measure the intensity of the impacts to the produce during harvest and post-harvest activities and subsequently to correlate it with the probability of damage to the produce itself (Hyde *et al.*, 1992; Tennes *et al.*, 1991; Ito *et al.*, 1994; Blandini *et al.*, 2002; Blandini *et al.*, 2003).

It should also be considered that each product has its own level of resistance to external stresses, which depend on the level of ripeness and other intrinsic factors, so that it appears fundamental to evaluate the stress threshold under which no damage is produced, in order to obtain products of high quality and extended storage capacity.

1.2 Impact damage on potato tubers

Potato (*Solanum tuberosum*) is one of the mankind's most valuable food crops. It is grown in more countries than any other crop with the exception of maize, and ranks fourth in volume of production. Limitations to its productivity are, therefore, of widespread economic interest.

The potato tuber is largely parenchymatous, lacking specialised secondary thickened tissues. As a result, tubers are susceptible to various forms of damage during commercial production, including external and internal defects. Internal damage, resulting from the effects of impacts on tubers during harvesting operations alone, may cause losses exceeding 20% (Storey and Davies, 1992).

Whilst it is a simple matter to grade out tubers showing external damage, internal damage is not visible until after peeling. The symptoms of internal damage may or may not include visible tissue fractures, and ensuing colour development of damaged areas may involve the formation of yellow, red, brown, blue, grey and black pigmentation to varying degrees (Burton, 1989; Gray and Hughes, 1978; Storey and Davies, 1992).

Damage sometimes includes the development of floury-white regions in clear contrast to the cream background of undamaged tissue. Where there is no visible fracture and colour development proceeds to a black end-product, the syndrome is usually referred to as internal bruising, though the terms "black spot" and "blue spot" are also common in the literature (Hiller *et al.*, 1985).

Common feature of the internal damage is that it almost certainly results from impact. This may seem obvious, but the fact that external damage has not occurred, suggests that tuber tissue responds differentially because of intrinsic differences in the biological properties of different layers and/or because it has been exposed to different stresses.

For example, damage from impact at higher energy levels is usually visible as fracture through all external layers, simply because external tissue has been insufficiently strong to withstand the incident stress. Once this stress has been concentrated into a crack in the outside layer, then damage proceeds rapidly through deeper layers (McGarry *et al.*, 1995).

For impacts below this level of energy, deformation and distribution of stress within the tissues create the potential for damage. Physical and biochemical properties of the tissue determine the internal response to the impact.

1.3 Relationship between damage and methods of harvesting

Potatoes play a significant role in the human nutrition in many countries all over the world. An efficient crop production requires high mechanised processes in all stages. Therefore, potatoes are handled several times during harvesting, transport and storage. Each handling operation leads to mechanical damage depending on the height of fall and the number of drops involved.

This damage causes substantial economic losses to the fresh market and the potato processing industry. One of the most serious defect problem is the occurrence of "black spots". A lot of investigations took place in the recent years to examine the reason for formation of black spots and the susceptibility of the potato tubers to black spot development from mechanical impacts (Molema, 1999).

It was established that the black spot development depends, among others things, on physical, physiological and biochemical properties of the tuber. Other important influence parameters are genotype, development and environment (McGarry *et al.*, 1996).

Although all the interrelations are now better understood, the reduction of mechanical stress is the best way to avoid black spots. In consideration of this fact, a gentle handling of potatoes from the harvest to the storage has to be achieved.

Currently, two methods of harvesting are widely used:

1. The harvested potatoes are passed via a discharge belt onto a transport vehicle which is driving alongside the harvester.

2. The potatoes are collected in a hopper on board the harvester. When the hopper is full, the content is transferred to the transport vehicle, either on the field or at the edge of the field.

At the storage hall the tubers are tipped out, separated of soil and stones as far as possible and brought into storage via a conveyor belt system. During each handling stage the potatoes are exposed to mechanical stress which leads to damage (Table 1.1).

Table 1.1: *Proportion of potato damage per handling stage to total damages (Bouman, 1995).*

Handling stage	Damage (%)
Harvest	14.7
Transport and interim storage	0.5
Conveyor belt and stationary filler	1.5
Storage	20.8
Removal of potatoes	12.2
Loading lorry	14.2
Unloading lorry	3.0
Processing (sorting to packing)	33.1

The highest percentage of damage occurs during processing (33.1%), but farmers often cannot influence this damage. Important for the farmer is the question, how can be reduced the damage from harvest until the end of storage. In total, one-third of the damage occur from harvest until the end of the storage.

To avoid damage caused by handling, transport and storage, farmers are tending to fill the potatoes into storage boxes immediately after harvesting. Boxes are brought to the fields in transport vehicles and filled by the harvester.

1.4 Potato shaped instrumented devices

Potato tubers are often exposed to unnecessarily high mechanical loads at many steps in the production chain from harvesting to packaging (Bentini *et* *al.*, 2006). These mechanical impacts can cause various external and internal injuries in potatoes, particularly black spot bruise (Molema, 1999; Peters, 1996).

Although many efforts have been made to improve black spot bruising, the best approach is to eliminate or at least to reduce the risk of mechanical impacts during processing.

In a large number of studies the effects of actions influencing product quality during harvesting and subsequent processes are evaluated by using defined criteria. In order to reduce economic losses for growers and the potato industry due to mechanical damage, electronic spheres or potato-shaped instrumented devices are often used to measure handling impacts of tubers (Herold *et al.*, 2001; Van Canneyt *et al.*, 2003; Maly *et al.*, 2005; Praeger *et al.*, 2013; Opara & Pathare, 2014).

To characterise the most important sources of mechanical loads in the production chain, so-called artificial fruits such as IS 100 (Figure 1.1) (United State Department, Michigan Agricultural Experiment Station and Michigan State University, 1989, USA) (Zapp *et al.*, 1990), PTR 100 (Bioteknisk Istitut, 1990, Denmark), PMS 60 (Figure 1.2) (Institut für Agrartechnik, Potsdam-Bornim (ATB), 1992, Germany) (Herold *et al.*, 1996), PTR 200 (SM Enginnering Denmark, 1999) and the "Smart Spud" (Sensor Wireless, 2000, Canada) (Bollen, 2006) have been available since several years.

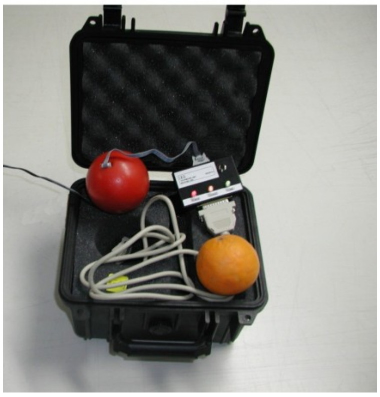


Figure 1.1: *Instrumented Sphere IS 100.*

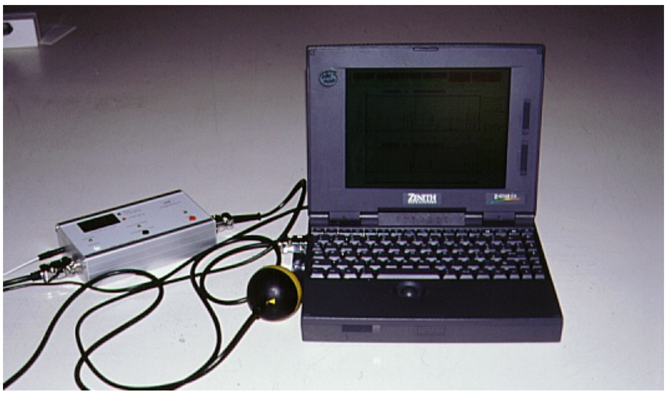


Figure 1.2: *Instrumented Sphere PMS 60 developed at ATB.*

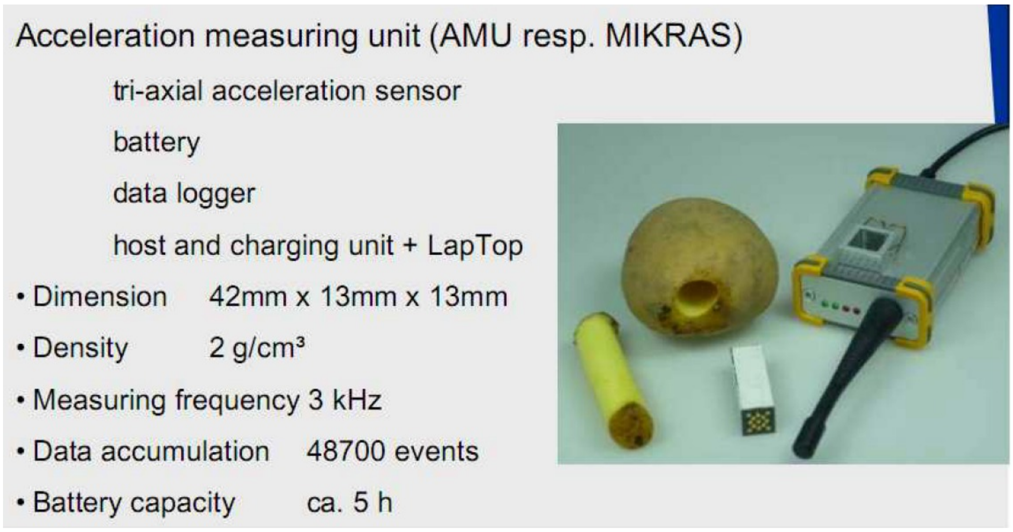
These instrumented devices are sufficiently equipped and frequently applied to locate those zones in the harvesting and processing chain that present a high level of risk of damage (Molema, 1999; Baheri, 1997). They consist of electronic impact measurement systems placed in a hard rubber or plastic body that simulates the real fruit. These instruments measure the actual mechanical impacts at different processing steps and can thus be helpful in identifying the potential risks of mechanical damage and in evaluating measures to reduce impact loads.

However, the damage predictive value of the earlier instruments IS 100, PTR 100 and PMS 60 has been frequently discussed in scientific literature: the information given by harvested and handled potatoes does not always correspond sufficiently with the information given by the instrumented devices. Among biological bruise susceptibility factors, natural variation in impact events, operating insufficiency of the devices for damage prediction purposes or difficult in data interpretation may cause this discrepancy (Leicher, 1992; Nerinckx and Verschoore, 1993).

The knowledge achieved so far by the instruments has not provided an important contribution to reducing the economic losses due to black spots in practice (Bollen *et al.*, 2001; Bollen, 2006). Indeed, the limited practical use of instruments is mainly due to the considerable differences between real and artificial fruit, largely restricting the transferability of measured impact data to mechanical load onto real products. The IS 100 and the other devices are not able to truly simulate the biological and physical properties (shape, elasticity, surface proprieties) of potato tubers.

Recently, a new approach has been proposed to overcome these disadvantages of artificial fruits. Based on a miniaturized impact detecting system, a self-contained Acceleration Measuring Unit (AMU) has been developed at the ATB Leibniz-Institut für Agrartechnik, Potsdam-Bornim, small enough to be fitted into a real product without significant changes of the product's properties (Figure 1.3). This AMU is suited to be integrated into a potato tuber and to record acceleration events acting on the tuber at many steps of the production chain.

The great advantage of the AMU is its ability to acquire impact data of a



Acceleration measuring unit (AMU resp. MIKRAS)

tri-axial acceleration sensor

battery

data logger

host and charging unit + LapTop

• Dimension	42mm x 13mm x 13mm
• Density	2 g/cm³
• Measuring frequency	3 kHz
• Data accumulation	48700 events
• Battery capacity	ca. 5 h

Figure 1.3: *Specification and view of miniaturised acceleration measuring unit (AMU) and its implantation in potato tuber (Praeger et al., 2013).*

potato tuber based on the tuber's actual physiological and physical properties. However, it is not known whether these properties affect the measuring characteristics of the AMU. Additionally, the technique of implanting the AMU (preparing the product, fitting and positioning the AMU) may influence the measuring data.

Chapter 2

Basic concepts on FEM

2.1 General aspects

The Finite Element Method (FEM), sometimes referred to as Finite Element Analysis (FEA), is a computational technique used to obtain approximate solutions of problems in engineering.

The basic idea in the finite element method is to find the solution of a complicated problem by replacing it by a simpler one. Since the actual problem is replaced by a simpler one, in finding the solution we will be able to find only an approximate solution rather than the exact solution.

The existing mathematical tools will not be sufficient to find the exact solution of most of the practical problems. Thus, in the absence of any other convenient method to find even the approximate solution of a given problem, we have to prefer the finite element method.

Moreover, in the finite element method, it will often be possible to improve or refine the approximate solution by spending more computational effort.

2.2 Brief history of the Finite Element Method

The term *finite element* was first used by Clough in 1960 in the context of plane stress analysis and has been in common usage since that time.

During the decades of the 1960s and 1970s, the finite element method was extended to applications in plate bending, shell bending, pressure vessels, and general three-dimensional problems in elastic structural analysis (Melosh, 1961; Melosh, 1963) as well as to fluid flow and heat transfer (Martin, 1968; Wilson and Nickell, 1966). Further extension of the method to large deflections and dynamic analysis also occurred during this time period (Turner *et al.*, 1960; Archer, 1965). An excellent history of the finite element method and detailed bibliography is given by Noor (1991).

The mathematical roots of the finite element method, instead, dates back at least a half century. Approximate methods for solving differential equations using trial solutions are even older in origin. Lord Rayleigh (1870) and Ritz (1909) used trial functions to approximate solutions of differential equations. Galerkin (1915) used the same concept for solutions.

The drawback in the earlier approaches, compared to the modern finite element method, is that the trial functions must apply over the entire domain of the problem of concern. While the Galerkin method provides a very strong basis for the finite element method not until the 1940s, when Courant (1943) introduced the concept of piecewise-continuous functions in a sub-domain, did the finite element method have its real start.

In the late 1940s, aircraft engineers were dealing with the invention of the jet engine and the needs for more sophisticated analysis of airframe structures to withstand larger loads associated with higher speeds. These engineers collectively known the flexibility of the method, in which the unknowns are the forces and the knowns are the displacements (Hutton, 2004).

The finite element method, in its most often-used form, corresponds to the displacement method, in which the unknowns are system displacements in response to applied force systems. This application of simple finite elements for the analysis of aircraft structure is considered as one of the key contributions in the development of the finite element method.

The digital computer provided a rapid means of performing the many calculations involved in the finite element analysis and made the method practically viable. Along with the development of high-speed digital computers, the application of the finite element method also progressed at a very impressive rate (Rao, 2004). The book by Przemieniecki (1968) presents the finite element method as applied to the solution of the stress analysis problems. Zienkiewcz and Cheung (1967) present the broad interpretation of the method and its applicability to any general filed problem.

2.3 Basics on the Finite Element Method

The Finite Element Method (FEM) has been developed into a key, indispensable technology in modelling and simulating advanced engineering systems in various fields like housing, transportation, communications, and so on. In building such advanced engineering systems, engineers and designers go through a sophisticated process of modelling, simulation, visualization, analysis, designing, prototyping, testing, and lastly, fabrication. Note that much work is involved before the fabrication of the final product or system. This is to ensure the workability of the finished product, as well as for cost effectiveness.

FEM was first used to solve problems of stress analysis, and then has been applied to many other problems like thermal analysis, fluid flow analysis, piezoelectric analysis, and many others. In all the applications, the analyst seeks to determine the distribution of some field variable: in stress analysis it is the displacement field or the stress field; in the thermal analysis it is the temperature field or the heat flux; in fluid flow it is the stream function or the velocity potential function; and so on.

FEM is a numerical method seeking an approximated solution of the distribution of field variables in the problem domain that is difficult to obtain analytically (Liu and Quek, 2003). It is a way of getting a numerical solution to a specific problem. A Finite Element Analysis does not produce a formula as a solution nor does it solve a class of problems.

An unsophisticated description of FEM is that it involves cutting a structure into several elements (pieces of the structure), describing the behaviour of each element in a simple way, then reconnecting elements at "nodes" as if nodes were drops of glue that hold elements together. This process results in a set of simultaneous algebraic equations. In stress analysis these equations are equilibrium equation of the nodes. There are may be several hundred or several thousand of such equations, which means that computer implementation is mandatory (Cook, 1995).

There are numerous physical engineering problems in a particular system. As mentioned earlier, although FEM was initially used for stress analysis, many other physical problems can be solved. Mathematical models of FEM have been formulated for many physical phenomena in engineering systems. Common physical problems solved using the standard FEM include:

- mechanics for solids and structures;
- heat transfer;
- fluid mechanics.

2.4 Computational modelling using FEM

The behaviour of a phenomenon in a system depends upon the geometry or domain of the system, the property of the material or medium, and the boundary, initial and loading conditions.

For an engineering system, the geometry or domain can be very complex. Further, the boundary and initial conditions can also be complicated. It is therefore, in general, very difficult to solve the governing differential equation via analytical means. In practice, most of the problems are solved using numerical methods. Among these, the methods of domain discretization championed by FEM are the most popular, due to its practicality and versatility.

The procedure of computational modelling using FEM broadly consists of four steps (Liu and Quek, 2003):

1. modelling of the geometry;
2. meshing;
3. specification of material properties;
4. specification of boundary, initial and loading conditions.

2.5 Modelling of the geometry

Real structures, components or domains are in general very complex, and have to be reduced to a manageable geometry. Curved parts of the geometry and its boundary can be modelled using curves and curved surfaces. However, it should be noted that the geometry is eventually represented by a collection of elements, and the curves and curved surfaces are approximated by piecewise straight lines or flat surfaces, if linear elements are used.

Depending on the software used, there are many ways to create a proper geometry in the computer for the Finite Element (FE) mesh. Points can be created simply by keying in the coordinates. Lines and curves can be created by connecting the points or nodes. Surfaces can be created by connecting, rotating or translating the existing lines or curves; and solids can be created by connecting, rotating or translating the existing surfaces. Points, lines and curves, surfaces and solids can be translated, rotated or reflected to form new ones.

Graphic interfaces are often used to help in the creation and manipulation of the geometrical objects. There are numerous Computer Aided Design (CAD) software packages used for engineering design which can produce files containing the geometry of the designed engineering system. These files can usually be read in by modelling software packages, which can significantly save time when creating the geometry of the models.

However, in many cases, complex objects read directly from a CAD file may need to be modified and simplified before performing meshing or discretisation. It may be worth mentioning that there are CAD packages which incorporate modelling and simulation packages, and these are useful for the rapid prototyping of new products.

2.6 Meshing

Meshing is performed to discretise the geometry created into small pieces called elements or cells.

Mesh generation is a very important task of the pre-process. It can be a very time consuming task to the analyst, and usually an experienced analyst will produce a more credible mesh for a complex problem. The domain has to be meshed properly into elements of specific shapes such as triangles and quadrilaterals. Information, such as element connectivity, must be created during the meshing for use later in the formation of the FEM equations.

It is ideal to have an entirely automated mesh generator, but unfortunately this is currently not available in the market. A semi-automatic pre-processor is available for most commercial application software packages. There are also packages designed mainly for meshing. Such packages can generate files of a mesh, which can be read by other modelling and simulation packages.

Triangulation is the most flexible and well-established way in which to create meshes with triangular elements. It can be made almost fully automated for two-dimensional (2D) planes, and even three-dimensional (3D) spaces. Therefore, it is commonly available in most of the pre-processors. The additional advantage of using triangles is the flexibility of modelling complex geometry and its boundaries. The disadvantage is that the accuracy of the simulation results based on triangular elements is often lower than that obtained using quadrilateral elements. Quadrilateral element meshes, however, are more difficulty to generate in an automated manner.

2.7 Property of the material

Many engineering systems consist of more than one material. Property of materials can be defined either for a group of elements or each individual element, if needed. For different phenomena to be simulated, different sets of material properties are required. For example, Young's modulus and shear modulus are required for the stress analysis of solids and structures, whereas the thermal conductivity coefficient will be required for a thermal analysis.

Inputting of a material's properties into a pre-processor is usually straightforward; all the analyst needs to do is key in the data on material properties and specify either to which region of the geometry or which elements the data applies. However, obtaining these properties is not always easy. There are commercially available material databases to choose from, but experiments are usually required to accurately determine the property of materials to be used in the system.

2.8 Boundary, initial and loading conditions

Boundary, initial and loading conditions play a decisive role in solving the simulation. Inputting these conditions is usually done easily using commercial pre-processors, and it is often interfaced with graphics. Users can specify these conditions either to the geometrical identities (points, lines or curves, surfaces, and solids) or to the elements or grids.

Again, to accurately simulate these conditions for actual engineering systems, requires experience, knowledge and proper engineering judgements. The boundary, initial and loading conditions are different from problem to problem.

Chapter 3

Materials and Methods

3.1 General aspects

The goal of this PhD Thesis is to evaluate the impact damage of vegetables resulting from mechanical harvest and post-harvest handling by means of the Finite Element Method (FEM) approach.

The Thesis has been developed in collaboration with the Institut für Agrartechnik, Potsdam-Bornim (ATB), Germany, and the referent was the Prof. Dr. Ing. Klaus-Günter Gottschalk.

During the stay in Potsdam, they were conducted laboratory tests (drop tests and texture analysis) with artificial fruits and the experimental results were compared with data of Finite Element Analysis (FEA) to verify correctness and reliability of FEM approach.

The Finite Element Analysis is a computer simulation technique used in the engineering analysis. This technique employs the Finite Element Method with the aim to obtain approximate solutions of problems in engineering.

All the software used to develop the FEA has in common the partition of the analysis process in three steps:

1. the pre-processing: to develop the finite element model;

2. the processing: to resolve the element finite problem, taking into account material properties, load and boundary conditions;

3. the post-processing: to elaborate and represent the solution.

Nowadays are available a great variety of FEA software, both open source and commercial. In this Thesis the Linux distribution `CAElinux2011` has been used. It contains several technical-engineering software, among which stand out Salome-Meca and Code-Aster. The first is a model building, meshing and post-processing software, while the latter is a processing software.

3.2 The Salome-Meca software

Over the last decade, the improvements in computer hardware and software have brought significant changes in the capabilities of simulation software. New computer power made possible the emergence of simulations that are more realistic (complex 3D geometries being treated instead of 2D ones), more complex (multi-physics and multi-scales being taken into account) and more meaningful (with propagation of uncertainties).

Since 2001, in order to facilitate and improve this process, CEA (Commissariat à l'énergie atomique et aux énergies alternatives) and EDF (Électricité de France) have developed a software platform named Salome that provides tools for building more complex and integrated applications.

Salome-Meca is an open-source software that provides a generic platform for pre- and post-Processing for numerical simulations. It is based on an open and flexible architecture made of reusable components. The platform has been built using a collaborative development approach and is therefore available under the LGPL license (http://www.salome-platform.org).

Salome-Meca provides modules and services that can be combined to create integrated applications that make the scientific codes easier to use and well interfaced with their environment. It is used in nuclear research and industrial studies by CEA and EDF in the fields of nuclear reactor physics, structural mechanics, thermo-hydraulics, nuclear fuel physics, material science, geology and waste management simulation, electromagnetism and radioprotection.

Salome-Meca can be used as standalone application for generation of a CAD model, its preparation for numerical calculations and post-processing of the calculation results. Salome-Meca can also be used as a platform for integration of external third-party numerical codes to produce new applications for the full life-cycle management of CAD models.

Two different modes of interaction with Salome-Meca components are systematically provided:

1. a graphic interface coupled with 3D graphic interaction (Qt4, VTK);

2. a text interface based on the Python language.

Both modes provide the same set of functionalities and Salome-Meca offers easy short cuts from one mode to the other.

3.3 The Code-Aster software

Developed since 1989 by EDF and for EDF's needs in computational mechanics, Code-Aster has demonstrated the possibility to combine in a unique software two so-called antagonistic aims:

1. an efficient software for engineering studies (about 300 users in-house and thousands as free users) with quality assurance requirements;

2. a numerical platform for software development products of the EDF's research in various computational mechanics fields.

Being constantly developed, updated and upgraded with new models, Code-Aster represents by now 1 200 000 lines of source code, most of it in Fortran and Python. The documentation of Code-Aster represents more than 12 000 pages: user's manuals, theory manuals compiling EDF's know-how in mechanics, example problems, verification manuals. All of these documents are available on line at www.code-aster.org.

The company EDF has chosen to freely distribute the program for the following reasons:

- create a sizeable user's group to speed up the process of identifying and correction of errors;

- increase the level of competence thanks to an extensive collaboration in the academy community with university, research centres, laboratories and specialist companies;
- encourage co-operative development of the program, sharing the experience and the new functionalities implemented by individual users in the greatest number possible of fields of application.

Code-Aster offers a full range of multi-physical analysis and modelling methods that go well beyond the standard functions of a software for thermomechanical computation:

- static and dynamic mechanics, linear or non-linear;
- modal analysis, harmonic and random response, seismic analysis;
- acoustics, thermodynamics;
- fracture, damage and fatigue;
- multi-physics, drying and hydration, metallurgy analysis, soil structure, fluid structure interactions;
- geometric and material non linearities, contact and friction.

3.4 FEM analysis

FEM analysis via Salome-Meca and Code-Aster basically requires four steps, each involving a software module. Modelling of geometry and meshing steps are generated by means of Salome-Meca, whereas specification of material properties, boundary, initial and loading conditions are described by using Code-Aster. In detail:

1. model building (GEOMETRY MODULE): geometric construction of the object (by Salome-Meca);
2. model meshing (MESH MODULE): mesh generation, creation of groups of surfaces on which to apply loads and constrains and definition of groups of volumes to which assign materials (by Salome-Meca);

3. model solving (CODE-ASTER MODULE): assigning materials and boundary conditions, analysis and resolution FE (by Code-Aster);
4. view of the results (POST-PRO MODULE): visualisation of the results and post-processing data (by Salome-Meca).

All steps can be performed by means of suitable interfaces and templates. Moreover, the geometry model and its meshing can be parametrised by means of a Python script, that allows easy changes in several aspects (size, position, rotation and so on).

3.5 Preliminary tests

Several simple models were preliminarily developed to gain familiarity with the computational software Salome-Meca and Code-Aster. A first group of models were developed to investigate their behaviour under static conditions, while in a second group of studies dynamic conditions were considered.

The first study (STUDY 1) was developed to verify the agreement between theoretical values and calculated values in terms of deformation and stress. Verified the correctness of the response, in the second test (STUDY 2) it was studied as to build an object composed of two or more materials. Finally, it was developed a third study (STUDY 3) showing a sphere of given material, inside which was placed a box of different material. This was a first, very simple model of a potato containing an acceleration or force sensor. On a defined area of the sphere it was applied a force to obtain the values of strain and stress on the box.

Forces and deformation were evaluated only under static conditions and linear behaviour of the materials.

In the second groups of studies dynamic conditions of analysis were examined and simple models of linear dynamic analysis and of non-linear dynamic analysis were developed.

With the model STUDY 4 it was developed a study where the applied force varies in time.

The impact damage of fruits is typically investigated through drop experiments: a fruit is dropped from a defined height onto an instrumented base. The FEM Analysis is perhaps the most appropriate technique for computer modelling of problems of this nature.

To represent a computer simulation model, it was developed a study (STUDY 5) showing a sphere of given material containing a box of different material, colliding against a rigid flat plane. Deformation, acceleration and velocity were evaluated under dynamic conditions.

Finally, the last preliminary test (STUDY 6) was equal to the previous study, but with shorter sampling time. The aim of the study was to analyse with greater detail the values of deformation and acceleration during the first impact between sphere and plane.

3.6 The experimental activity at ATB

Laboratory tests at Potsdam were aimed at measuring the impact force produced by two artificial fruits when dropped on a base provided with sensors. The two artificial fruits were of spherical form and with different mass, diameter, density and modulus of elasticity.

The first, named "BALL 1" or BIG BALL (Figure 3.1), had diameter of 68.4 mm (average value in the three axial directions: $x = 68.4$ mm, $y = 68.3$ mm, $z = 68.6$ mm), mass of 57 g and then density of 338 kg/m^3.

The second, named "BALL 2" or SMALL BALL (Figure 3.1), had diameter of 57.4 mm (average value in the three axial directions: $x = 58.2$ mm, $y = 57.1$ mm, $z = 57.0$ mm), mass of 134 g and then density of 1350 kg/m^3.

A summary of the main characteristics of the two spheres is reported in Table 3.1.

Table 3.1: *Main features of the two artificial fruits.*

Artificial fruits	Diameter, mm	Mass, g	Density, kg/m^3
BALL 1	68.4	57	338
BALL 2	57.4	134	1350

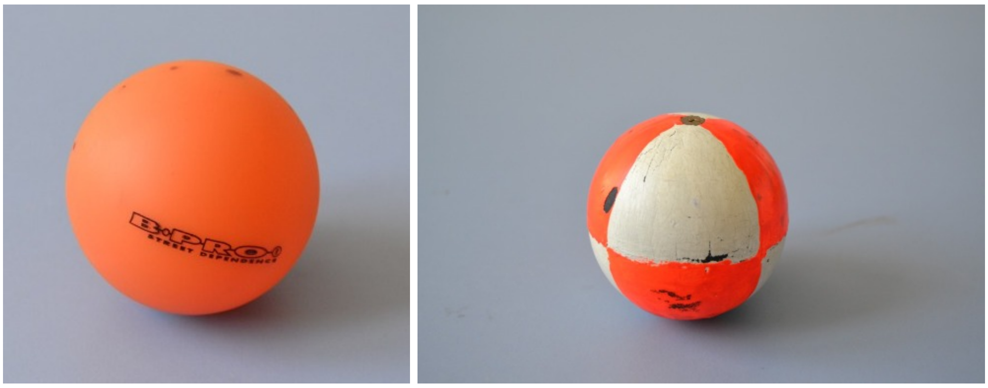


Figure 3.1: *The two artificial fruits used for the drop tests: BALL 1 (left) and BALL 2 (right).*

3.6.1 Measurement of the modulus of Young

The two spheres were preliminarily studied with a "texture analyser" (Figure 3.2), an instrument that records the curve stress-strain, from which it can be obtained the modulus of Young based on the Hertz theory.

To this end, the texture analyser was equipped with two spherical steel probes of different diameter (Figure 3.3) to press the artificial fruits. The two probe diameters were:

- probe 1: diameter = 12.7 mm;
- probe 2: diameter = 6.3 mm.

Ten points, enumerated in ascending order from 1 to 10, were located at random on each artificial fruit. Then the two spheres were fixed in a sand bed (Figure 3.4) and pressed with the probes in correspondence of the points previously fixed at a rate of 10 mm/min until a maximum force of 2 N was reached. During the test, the instrument recorded the maximum force and the displacement at the maximum force. The force limit of 2 N ensured that the relation between force and displacement stayed approximately linear during the movement of the probe. The measurements were performed one by one during approximately one hour.

According to Hertz's theory of elastic contact (Nuovo Colombo, 1985; Khodabakhshian and Emadi, 2011; Geyer *et al.*, 2009), when two elastic

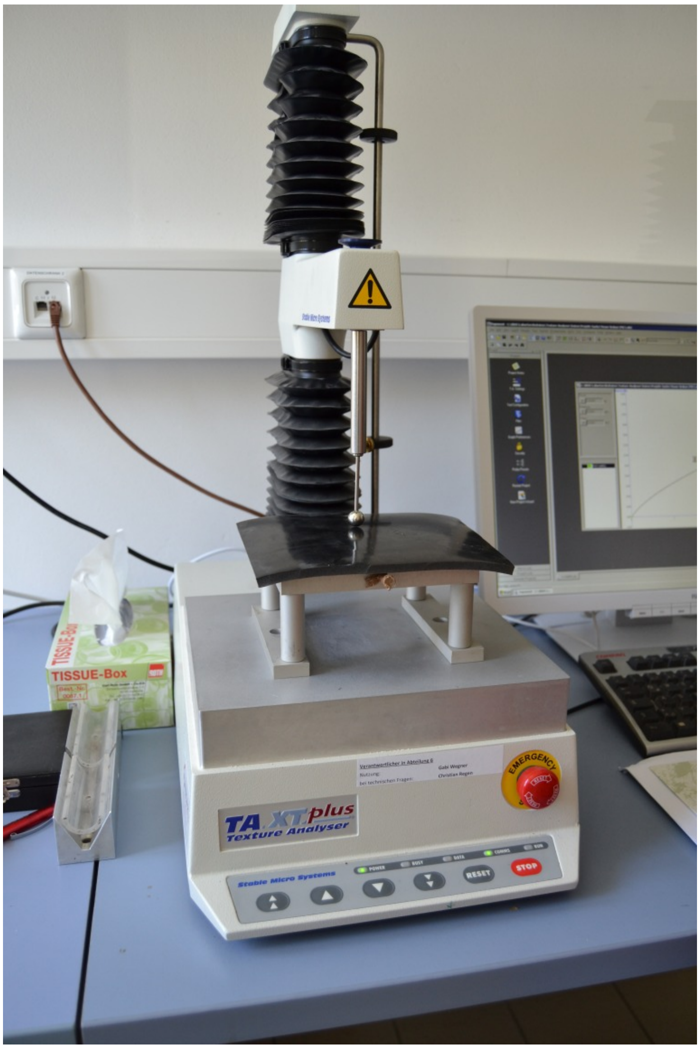


Figure 3.2: *The texture analyser for calculating the modulus of Young.*

spheres with diameters D_1 and D_2 are are brought into contact at a single point, if collinear forces F are applied so as to press the two spheres together, deformation takes place and a small contact area will replace the contact point of the unloaded state (Figure 3.5).

Hertz starts by assuming that the contacting solids are isotropic and linearly elastic, and also that the representative dimensions of the contact area are very small compared to the various radii of curvature of the undeformed bodies, in the vicinity of the contact interface.

Under these hypotheses, the deformation u, defined as the relative move-

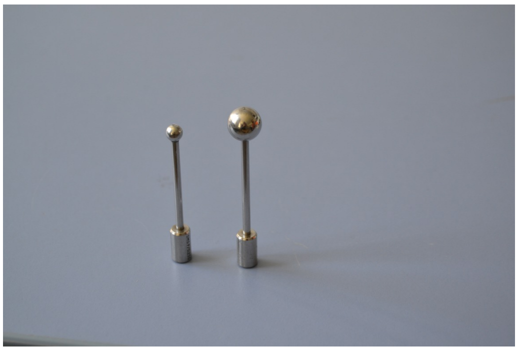


Figure 3.3: *Spherical probes for measuring the modulus of Young.*

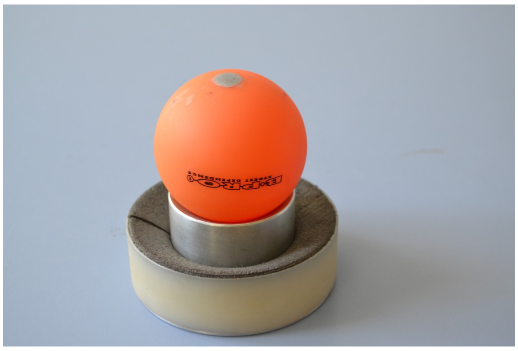


Figure 3.4: *Positioning of the sphere for measuring the modulus of Young.*

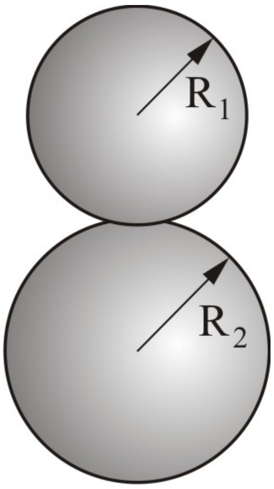


Figure 3.5: *Hertz contact of two spheres.*

ment of approach along the line of the applied force of two points, each of which in one of the two bodies, can be evaluated as:

$$u = 1.040\sqrt[3]{\frac{F^2 C_E^2}{C_C}}, \tag{3.1}$$

being:

- F: total force which presses the bodies against one another;
- $C_E = \dfrac{1-\nu_1^2}{E_1} + \dfrac{1-\nu_2^2}{E_2}$;

- $\frac{1}{C_C} = \frac{1}{D_1} + \frac{1}{D_2}$: a parameter that takes into account the curvature of the two spheres;
- ν_1, ν_2: moduli of Poisson of the two spheres;
- E_1, E_2: moduli of Young of the two spheres;
- D_1, D_2: diameters of the two spheres.

Equation 3.1 can be solved to obtain C_E:

$$C_E = \frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2} = \sqrt{\frac{u^3 C_C}{1.040^3 F^2}} \tag{3.2}$$

Assuming the sphere 1 as the probe and the sphere 2 as the artificial fruit, E_1 is much higher than E_2, so the term $\frac{1-\nu_1^2}{E_1}$ can be neglected and Equation 3.2 allows calculation of E_2:

$$E_2 = 1.061 \frac{F(1-\nu_2^2)}{u^{3/2} C_C^{1/2}}, \tag{3.3}$$

or (ASAE Standards, 1999):

$$E_2 = 1.061 \frac{(1-\nu_2^2)F}{u^{3/2}} \left(\frac{1}{D_1} + \frac{1}{D_2} \right)^{1/2}. \tag{3.4}$$

The value used for ν_2 was 0.49.

3.6.2 Drop tests with artificial fruits

After calculation of modulus of Young, the two artificial fruits were subjected to impact tests by using a drop simulator to control the impact dynamic (Figure 3.6, Geyer *et al.*, 2009).

It consisted of a frame with two vertical guide wires, taut clamped in parallel. A sliding carriage was free to move along the guide wires. An artificial fruit was placed on a circular hole in the middle of the carriage (thickness of 5 mm). The artificial fruits protruded slightly from the hole

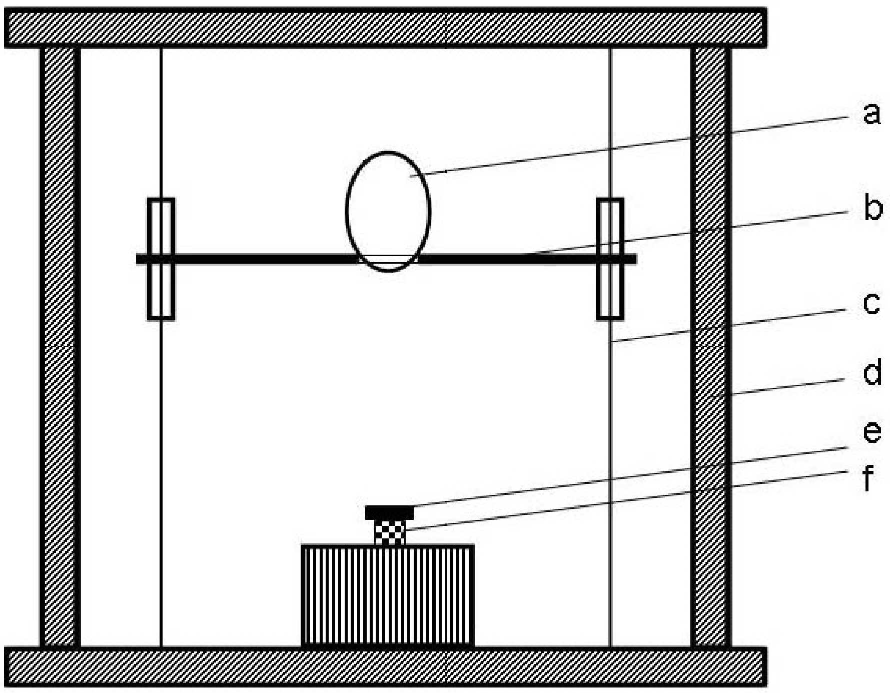


Figure 3.6: *Schematic view of the drop simulator (Geyer* et al.*, 2009): a = artificial fruit, b = sliding carriage, c = guide wire, d = frame, e = steel plate, and f = force sensor.*

depending on their radius of curvature. The diameter of the hole (40 mm) was wider than that of a circular plate (diameter 30 mm) that was rigidly coupled on a force sensor at the bottom of the frame.

Two impact materials were considered (Figure 3.7):

- Steel plate: thickness of 5 mm;
- PVC plate: thickness of 5 mm, applied over the steel plate.

The carriage was fixed at the preselected height and was released after placing the artificial fruit. Two drop heights were considered (10 cm and 25 cm) and 20 replicates for each test condition were carried out. During the impact on the plate, the force sensor recorded and stored on a PC the maximum force. The files are in tabular format and can be easily managed with a spreadsheet.

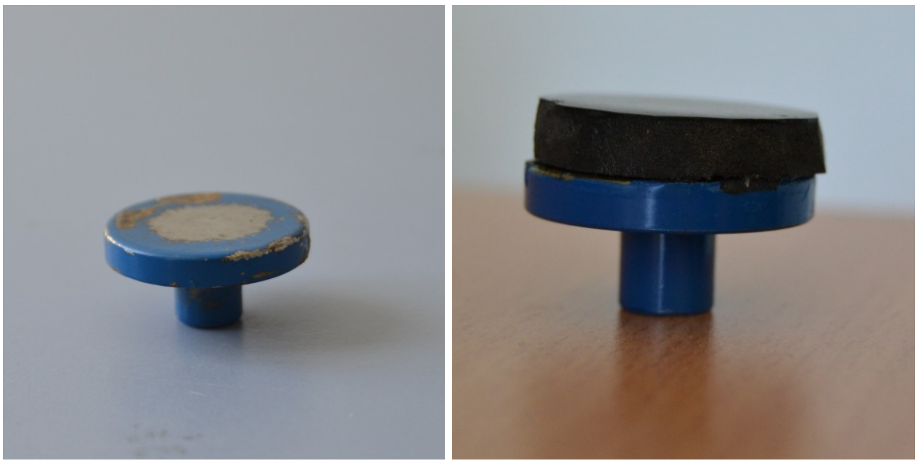


Figure 3.7: *The two impact materials: steel and PVC.*

3.7 Drop test simulations

The drop tests with the artificial fruits were simulated with Salome-Meca and Code-Aster, using as moduli of Young for the artificial fruits those measured with the experimental tests. The aims of these simulations were to develop a model that, if in agreement with the experimental results, could be used for further simulations.

The impact area was modelled as a disk with diameter of 5 cm and thickness of 0.5 cm. When necessary, it was covered with a disk of PVC of same diameter and thickness.

As these preliminary simulations provided good agreement between measured and simulated impact force, the model was used to develop an extensive set of simulations to analyse the effects on the impact force of material properties (density and modulus of Young), pseudo-fruit size (spheres of different diameters) and drop height.

In detail, the experimental design considered:

- sphere diameter D: 50, 55, 60, 65, 70, 75 and 80 mm;
- modulus of Young E: 0.5, 1.5, 2.5 and 3.5 MPa;
- material density ρ: 900, 1000 and 1100 $\mathrm{kg/m^3}$;

- drop height h: 10, 15, 20, 25, 35 and 50 cm.

With the given values of material density and sphere diameter, the mass of the sphere ranged from 60 up to 295 g. All values were chosen to approach potato tubers properties (Geyer *et al.*, 2009).

Only drop tests on steel were simulated, given their greater agreement with experimental results. Material behaviour was considered as elastic linear. The following quantities were extracted from each run test:

- the maximum impact force transmitted to the steel plate (to simulate the force sensor during drop tests);
- the maximum acceleration at the centre of the sphere (to simulate the AMU device, Figure 1.3);
- the maximum acceleration af a node of the sphere belonging to the contacting area with the steel plate.

Globally, 360 simulations were carried out, using always the same model. To reduce the simulation time, in the geometric model the sphere was placed at a distance $\Delta h = 3\,\mathrm{mm}$ above the impact plate and its initial velocity was calculated accordingly:

$$v_0 = \sqrt{2g(h - \Delta h)}, \tag{3.5}$$

being $g = 9.81\,\mathrm{m/s^2}$ the gravity acceleration and h the effective drop height.

For each model, the tetrahedron containing the centre of the sphere was selected and its four nodes (N_1, N_2, N_3 and N_4) were localised. The acceleration of each node was recorded and the average values was considered as representative of the acceleration of the centre of the sphere (the acceleration that should be measured by the AMU sensor).

Accelerations and impact forces were analysed at varying the model parameters: diameter, density, mass and modulus of Young of the sphere and drop height.

Data analyses and graphical representations were carried out by using the open source software *R* (R Core Team, 2013).

Chapter 4

Results and Discussion

4.1 General aspects

Results discuss firstly the preliminary models aimed at gaining familiarity with the software, then the experimental tests carried out at Potsdam laboratories to measure modulus of Young and impact force of two artificial fruits, and finally the extensive set of simulations aimed at evaluating the effects of size, density, modulus of Young of the sphere and drop height on the impact parameters (maximum impact force and maximum acceleration at the centre of the sphere).

Python code to build the models and Code-Aster commands to analyse them are also provided for a reproduction of the results.

4.2 Preliminary tests

4.2.1 Study 1

Theoretical aspects

With reference to a simple and homogeneous isotropic parallelepiped block (Figure 4.1), the Hooke's law for linear-elastic materials, in the simplest form, is given by:

$$\sigma = E \cdot \epsilon, \tag{4.1}$$

with:

- σ: stress;
- E: modulus of Young;
- ϵ: strain.

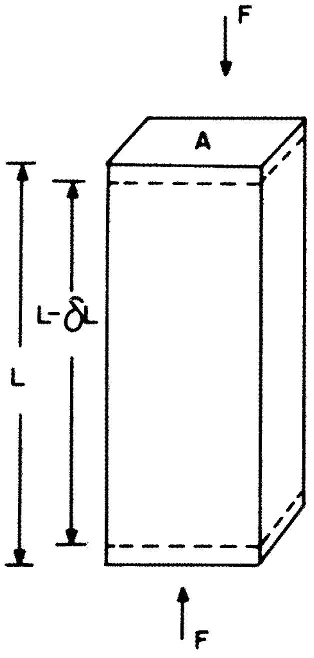


Figure 4.1: *Normal stress.*

The strain ϵ is given by:

$$\epsilon = \frac{\Delta L}{L}, \tag{4.2}$$

with:

- ΔL: deformation;
- L: initial length.

The stress σ is given by:

$$\sigma = \frac{F}{A}, \tag{4.3}$$

with:

- F: applied force;
- A: area.

From Equation (4.1) we obtain:

$$\epsilon = \frac{\sigma}{E}. \tag{4.4}$$

Equating Equations (4.2) and (4.4), we obtain:

$$\epsilon = \frac{\Delta L}{L} = \frac{\sigma}{E} \tag{4.5}$$

and then:

$$\Delta L = \frac{\sigma}{E} L. \tag{4.6}$$

Given stress, modulus of Young, and initial length, Equation (4.6) allows the calculation of the deformed shape.

Study 1 refers to an object parallelepiped-shaped with size (x, y and z values) $10\,\mathrm{cm} \times 10\,\mathrm{cm} \times 100\,\mathrm{cm}$ and modulus of Young $E = 100\,\mathrm{N/cm^2}$. This object is placed on a plane that imposes the boundary conditions (no movements of the base); on the opposite face it is applied a force equal to 1000 N.

Being the area of application of the force $A = 100\,\mathrm{cm^2}$, from Equation (4.3) we obtain the theoretical value of stress:

$$\sigma = \frac{1000\,\mathrm{N}}{100\,\mathrm{cm^2}} = 10\,\mathrm{N/cm^2}. \tag{4.7}$$

From Equation (4.6) we obtain the theoretical value of deformation:

$$\Delta L = \frac{10\,\mathrm{N/cm^2}}{100\,\mathrm{N/cm^2}} 100\,\mathrm{cm} = 10\,\mathrm{cm}. \tag{4.8}$$

Geometrical model building

The next step is the building in Salome-Meca of the model with the aim to verify the correctness of the solution provided by the software. To this purpose it was verified the agreement between the theoretical values of ΔL and σ and those calculated after the FEM analysis.

Primarily, by means of the `Geometry` module of Salome-Meca, it was built a box with size 10 cm $\times$ 10 cm $\times$ 100 cm. Salome-Meca provides some primitive entities to build models. A box can be defined by specifying two vertices (its opposite corners), or by specifying its dimensions along the coordinate axes and with edges parallel to them.

After, the box was "exploded" by using the `Explode` function: in this way the box is divided in its face components. Moreover, it is possible to detect the face, which was named `Pressure`, on which to apply the stress σ, and the face, which was named `Base`, regarded as constrain. Single faces can be selected with the mouse pointer.

Meshing

The goal is now to divide the box volume into a mesh, a bunch of small volumes on which it will be applied the material properties and it will be calculated the stresses resulting from the load.

The `Mesh` module of Salome-Meca provides several algorithms in order to produce a mesh. The menu `Mesh`, `Create Mesh`, opens a dialogue box called `Create Mesh`. Selecting the `Box` value as geometrical object, you will be prompted to fill Hypothesis and Algorithm. Here there is a multiple selection for both fields. We used the algorithm `NETGEN 3D-2D-1D` (triangulation). A new element named `Mesh_1` will be created.

Subsequently, it is necessary the designation of the faces on which it will be applied the loads and of the volumes on which it will be applied the material properties. When the mesh and all geometrical entities (faces and volumes) have been created, the mesh will be exported to a `MED` file format that can be read by Code-Aster.

Model processing via Code-Aster

Model processing is carried out by using a script, that is a Python code that can be edited according to the needs. There is a small utility, named

EFICAS, that enables to input the material properties, the loads, the boundary conditions and so on. EFICAS can be launched via the ASTK wizard, a graphical user interface for Code-Aster.

To assign material properties, we used the DEFI_MATERIAU directive. In a linear study, a material's name and three numerical values are required: modulus of Young E, Poisson coefficient NU and volumetric mass RHO. In this study the material's name entered is Mat1.

To assign load conditions, the AFFE_CHAR_MECA directive is used. To apply a boundary condition to the face Base of our box, the DDL_IMPO instruction assigns at the GROUP_MA Base null displacement along all the three coordinates. To apply the homogeneous pressure to the face Pressure of our box, the PRES_REP instruction assigns at the GROUP_MA Pressure the desired value of loads (10 N/cm^2).

Finally, the MECA_STATIC instruction provides the solution and the fields for nodes and elements are saved in a MED file. Here is the command file.

```
# Linear Statics with 3D linear solid elements
# template by J.Cugnoni, CAELinux.com, 2005

DEBUT();

# Read MED MESH File
# First command LIRE_MAILLAGE is used to read the MED mesh file
# generated by Salome.
# To Do:
# Enter the name of your Salome Mesh in b_format_med->NOM_MED

MeshLin=LIRE_MAILLAGE(UNITE=20,
                      FORMAT='MED',
                      NOM_MED='Mesh_1',
                      INFO_MED=1,);

# Assigns a physical model to geometric entities.
# Here we assume that all the geometric entities (TOUT=OUI)
# are used for mechanical simulation (PHENOMENE=MECANIQUE)
# with 3D solid elements.

# To Do (Optional):
# you can assign other physics or element types (like shells
# for example) to some of the elements by replacing TOUT=OUI
# in AFFE with GROUP_MA = TheElementGroupYouWantToModel

FEMLin=AFFE_MODELE(MAILLAGE=MeshLin,
                   AFFE=_F(TOUT='OUI',
                           PHENOMENE='MECANIQUE',
                           MODELISATION='3D',),);

# Material properties
# To Do:
# Enter your material properties in this section if necessary
# Copy/Paste the DEFI_MATERIAU command to add a second material

Mat1=DEFI_MATERIAU(ELAS=_F(E=10,
                           NU=0.3,
                           RHO=1000,),);

# Assign Material properties to Elements
# To Do:
# If you need more than one material, you need to enter pairs
# of element group <-> materials by duplicating the AFFE option.

Mat=AFFE_MATERIAU(MAILLAGE=MeshLin,
                  MODELE=FEMLin,
                  AFFE=_F(TOUT='OUI',
                          MATER=Mat1,),);

# Boundary conditions
# This section defines the boundary conditions of the FEA, use
# DDL_IMPO on selected groups to impose displacements and
# FORCE_*/PRESSION to apply forces/pressures to selected groups

# To Do:
# for each boundary conditions, you need to choose the
# appropriate option, for example DDL_IMPO for imposed
# displacements, and assign this option to a selected region of
# the mesh by using the GROUP_NO option for a node group or the
# GROUP_MA for face/volume groups.

BCnd=AFFE_CHAR_MECA(MODELE=FEMLin,
                    DDL_IMPO=_F(GROUP_MA='Base',
                                DX=0.0,
                                DY=0.0,
                                DZ=0.0,),
                    PRES_REP=_F(GROUP_MA='Pressure',
                                PRES=1,),);

# Finite Element Solution

Solution=MECA_STATIQUE(MODELE=FEMLin,
                       CHAM_MATER=Mat,
                       EXCIT=_F(CHARGE=BCnd,),
                       SOLVEUR=_F(METHODE='MUMPS',
                                  RENUM='AUTO',),);

# Compute VonMises Stress / Strain

Solution=CALC_ELEM(reuse=Solution,
                   RESULTAT=Solution,
                   OPTION=('SIEF_ELNO', 'SIEQ_ELNO',
                           'SIGM_ELNO',),);
```

```
Solution=CALC_NO(reuse=Solution,
                 RESULTAT=Solution,
                 OPTION=('REAC_NODA', 'SIEQ_NOEU',
                         'SIEF_NOEU', 'SIGM_NOEU',),);

# Write Results to MED file

IMPR_RESU(MODELE=FEMLin,
          FORMAT='MED',
          UNITE=80,
          RESU=_F(RESULTAT=Solution,
                  TOUT_CHAM='OUI',
                  TOUT_CMP='OUI',),);

FIN(FORMAT_HDF='OUI',);
```

EFICAS only works on the command file; once it has been saved, one must come back to ASTK to launch the Code-Aster calculation. Code-Aster runs and displays its output in a non interactive shell window.

Post-processing module

Come back to Salome-Meca, this time using the post-processor module, where you can graphically display the results saved by Code-Aster in a .RES.MED. file. By importing this file in Salome-Meca, we can display the Solution DEPL and Solution SIGM_ELNO fields, that are the displacements and the stresses respectively.

To visualize the displacements, we can plot the Deformed Shape (Figure 4.2). The displacement values of the nodes belonging to the face Pressure are 9.94 cm, very close to the theoretical value of 10 cm (Figure 4.3). Therefore the theoretical values of deformation agree with those computed by the software.

Finally, to visualize the stress, the SIGM_ELNO field allow plotting the stress distribution along the bar. A coloured box and a graduate colour map is

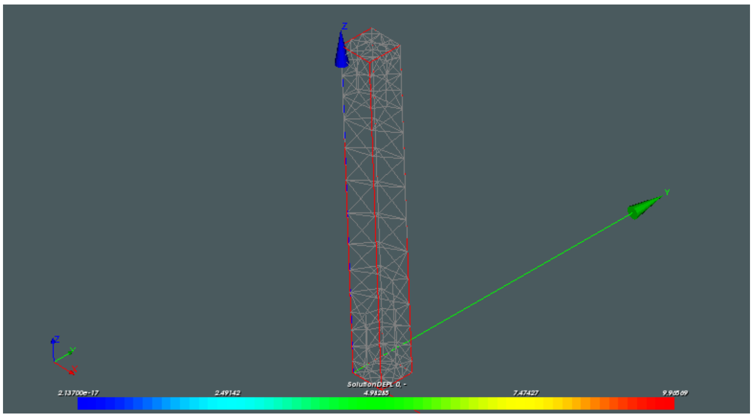


Figure 4.2: *Deformed shape.*

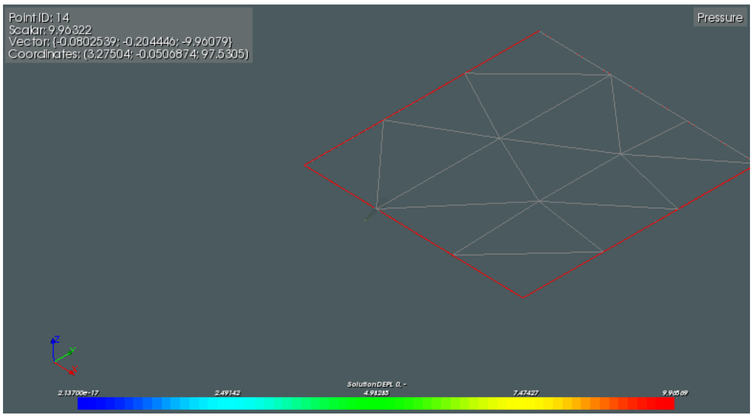

Point ID: 14
Scalar: 9.96322
Vector: (-0.0802539; -0.204446; -9.96079)
Coordinates: (3.27504; -0.0506874; 97.5305)

Figure 4.3: *Displacements of the Pressure face nodes.*

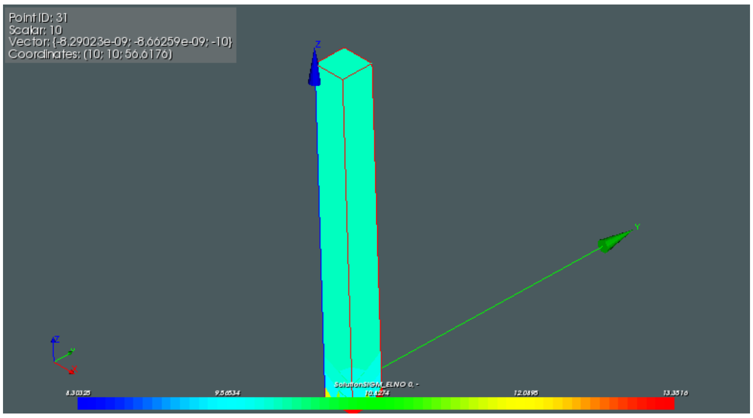

Point ID: 31
Scalar: 10
Vector: (-8.29023e-09; -8.66259e-09; -10)
Coordinates: (10; 10; 56.6176)

Figure 4.4: *Stress distribution along the box.*

displayed (Figure 4.4). The colour of the box is the same along the bar and the numerical value assigned to that colour is $10\,\mathrm{N/cm^2}$. This confirms that the value of stress is the same in the entire box, as it should be.

4.2.2 Study 2

The aim of the Study 2 was to analyse a simple object composed of two materials. In the same time, all the steps pertaining to te building of the geometric model and its meshing were coded in a Python script, which allows for a parametrisation of the object geometry and then for fixing new values according to the needs.

It refers to an object composed of two overlapping parallelepipeds. That on the top, called `Box_1`, has size $10\,\mathrm{cm} \times 10\,\mathrm{cm} \times 100\,\mathrm{cm}$ and modulus of Young $E = 100\,\mathrm{N/cm^2}$, while that on the bottom, called `Box_2`, has size $10\,\mathrm{cm} \times 10\,\mathrm{cm} \times 200\,\mathrm{cm}$ and modulus of Young $E = 50\,\mathrm{N/cm^2}$. This object is placed on a plane, which imposes the constraint conditions on the base, while on the face opposite the base it is applied a force equal to $1000\,\mathrm{N}$.

For each parallelepiped it was verified the agreement between theoretical values and simulated values of displacements ΔL and stress σ.

Considering only the `Box_2`, the theoretical value of stress σ_2 is equal to:

$$\sigma_2 = \frac{1000\,\mathrm{N}}{100\,\mathrm{cm^2}} = 10\,\mathrm{N/cm^2}. \tag{4.9}$$

The corresponding value ΔL_2 of the deformation is:

$$\Delta L_2 = \frac{10\,\mathrm{N/cm^2}}{50\,\mathrm{N/cm^2}} \cdot 200\,\mathrm{cm}. \tag{4.10}$$

Considering only the `Box_1`, the theoretical value of stress σ_1 is equal to:

$$\sigma_1 = \frac{1000\,\mathrm{N}}{100\,\mathrm{cm^2}} = 10\,\mathrm{N/cm^2}, \tag{4.11}$$

while the theoretical value of deformation ΔL_1 is equal to:

$$\Delta L_1 = \frac{10\,\mathrm{N/cm^2}}{100\,\mathrm{N/cm^2}} \cdot 100\,\mathrm{cm}. \tag{4.12}$$

Really, the face of the `Box_1` on which it is applied the pressure has a theoretical value of total deformation given by the sum of ΔL_2 and ΔL_1:

$$\Delta L = \Delta L_1 + \Delta L_2 = 50\,\mathrm{cm}. \tag{4.13}$$

The model building of the object in Salome-Meca can be executed not only via graphical interface, but also with textual interface based on Python program language. Below the Python code of the Study 2.

```
### Loading libraries
import sys
import salome
salome.salome_init()
theStudy = salome.myStudy

### GEOM component
import GEOM
import geompy
import math
import SALOMEDS

geompy.init_geom(theStudy)

# size Box_1
DX1 = 10.0
DY1 = 10.0
DZ1 = 200.0

# size Box_2
DX2 = 10.0
```

```
DY2 = 10.0
DZ2 = 100.0

Box_1 = geompy.MakeBoxDXDYDZ(DX1, DY1, DZ1)
geompy.addToStudy(Box_1, 'Box_1')

Box_2 = geompy.MakeBoxDXDYDZ(DX2, DY2, DZ2)
geompy.TranslateDXDYDZ(Box_2, 0, 0, DZ1)
geompy.addToStudy(Box_2, 'Box_2')

Fuse_1 = geompy.MakeFuse(Box_1, Box_2)
geompy.addToStudy(Fuse_1, 'Fuse_1')

Partition_1 = geompy.MakePartition([Fuse_1], [Box_1, Box_2],
              [], [], geompy.ShapeType["SOLID"], 0, [], 0)
[Box_1_1, Box_2_1] = geompy.ExtractShapes(Partition_1,
                     geompy.ShapeType["SOLID"], True)
[Face_1, Face_2, Face_3, Face_4, Base, Face_6, Pressure,
         Face_8, Face_9, Face_10, Face_11] =
         geompy.ExtractShapes(Partition_1,
         geompy.ShapeType["FACE"], True)

geompy.addToStudy(Partition_1, 'Partition_1')
geompy.addToStudyInFather(Partition_1, Pressure, 'Pressure')
geompy.addToStudyInFather(Partition_1, Base, 'Base')
geompy.addToStudyInFather(Partition_1, Box_1_1, 'Box_1')
geompy.addToStudyInFather(Partition_1, Box_2_1, 'Box_2')

### SMESH component
import smesh, SMESH, SALOMEDS
import NETGENPlugin

Mesh_1 = smesh.Mesh(Partition_1)
```

```
NETGEN_2D3D = Mesh_1.Tetrahedron(algo=smesh.FULL_NETGEN)
NETGEN_3D_Parameters = NETGEN_2D3D.Parameters()
NETGEN_3D_Parameters.SetMaxSize(2)
NETGEN_3D_Parameters.SetSecondOrder(0)
NETGEN_3D_Parameters.SetOptimize(1)
NETGEN_3D_Parameters.SetFineness(4)
isDone = Mesh_1.Compute()

Box_1_2 = Mesh_1.GroupOnGeom(Box_1_1, 'Solid_1', SMESH.VOLUME)
Box_2_2 = Mesh_1.GroupOnGeom(Box_2_1, 'Solid_2', SMESH.VOLUME)
Base_1 = Mesh_1.GroupOnGeom(Base, 'Base', SMESH.FACE)
Pressure_1 = Mesh_1.GroupOnGeom(Pressure, 'Pressure', SMESH.FACE)

Box_1_2.SetName('Box_1')
Box_2_2.SetName('Box_2')
[Box_1_2, Box_2_2, Base_1, Pressure_1] = Mesh_1.GetGroups()

# Set object names
smesh.SetName(Mesh_1.GetMesh(), 'Mesh_1')
smesh.SetName(NETGEN_2D3D.GetAlgorithm(), 'NETGEN_2D3D')
smesh.SetName(NETGEN_3D_Parameters, 'NETGEN 3D Parameters')
smesh.SetName(Box_1_2, 'Box_1')
smesh.SetName(Box_2_2, 'Box_2')
smesh.SetName(Base_1, 'Base')
smesh.SetName(Pressure_1, 'Pressure')
```

The script is interpreted by Salome-Meca, so performing the geometrical building of the object and its meshing.

The Code-Aster command file is the following:

```
DEBUT();


MeshLin=LIRE_MAILLAGE(UNITE=20,
                      FORMAT='MED',
                      NOM_MED='Mesh_1',
                      INFO_MED=1,);

FEMLin=AFFE_MODELE(MAILLAGE=MeshLin,
                   AFFE=_F(TOUT='OUI',
                           PHENOMENE='MECANIQUE',
                           MODELISATION='3D',),);

Mat1=DEFI_MATERIAU(ELAS=_F(E=100,
                           NU=0.3,
                           RHO=1000,),);

Mat2=DEFI_MATERIAU(ELAS=_F(E=50,
                           NU=0.3,
                           RHO=1000,),);

Mat=AFFE_MATERIAU(MAILLAGE=MeshLin,
                  MODELE=FEMLin,
                  AFFE=(_F(GROUP_MA='Box_1',
                           MATER=Mat1,),
                        _F(GROUP_MA='Box_2',
                           MATER=Mat2,),),);

BCnd=AFFE_CHAR_MECA(VERI_NORM='OUI',
                    MODELE=FEMLin,
                    DDL_IMPO=_F(GROUP_MA='Base',
                                DX=0.0,
                                DY=0.0,
                                DZ=0.0,),
                    PRES_REP=_F(GROUP_MA='Pressure',
                                PRES=10,),);
```

```
Solution=MECA_STATIQUE(MODELE=FEMLin,
                       CHAM_MATER=Mat,
                       EXCIT=_F(CHARGE=BCnd,),
                       SOLVEUR=_F(METHODE='MUMPS',
                                  RENUM='AUTO',),);

Solution=CALC_ELEM(reuse=Solution,
                   RESULTAT=Solution,
                   OPTION=('SIEF_ELNO', 'SIEQ_ELNO',
                           'SIGM_ELNO',),);

Solution=CALC_NO(reuse=Solution,
                 RESULTAT=Solution,
                 OPTION=('REAC_NODA', 'SIEQ_NOEU',
                         'SIEF_NOEU', 'SIGM_NOEU',),);

IMPR_RESU(MODELE=FEMLin,
          FORMAT='MED',
          UNITE=80,
          RESU=_F(RESULTAT=Solution,
                  TOUT_CHAM='OUI',
                  TOUT_CMP='OUI',),);

FIN(FORMAT_HDF='OUI',);
```

As in Study 1, in the post-processor Module the interesting lines are `Solution DEPL` and `Solution SIGM_ELNO` fields, that is displacements and stresses respectively.

The deformed shape is shown in Figure 4.5; the values of displacements of the nodes belonging to the face on the top of the `Box_2` are 40.27 cm, near to the theoretical values of deformation of 40 cm (0.6% error, Figure 4.6).

In the `Box_1` the values of the total displacement of the nodes belonging to the face `Pressure` are 49.93 cm, again very close to the theoretical value of 50 cm (Figure 4.7).

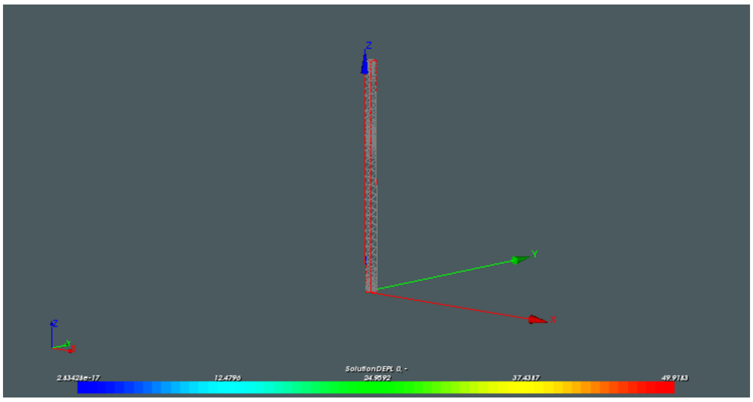
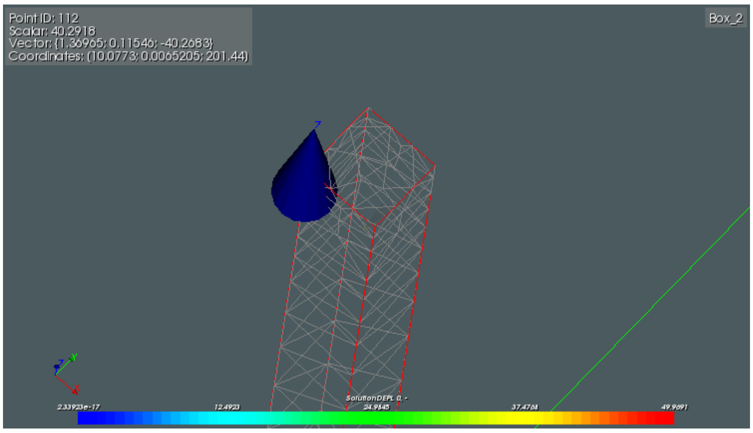
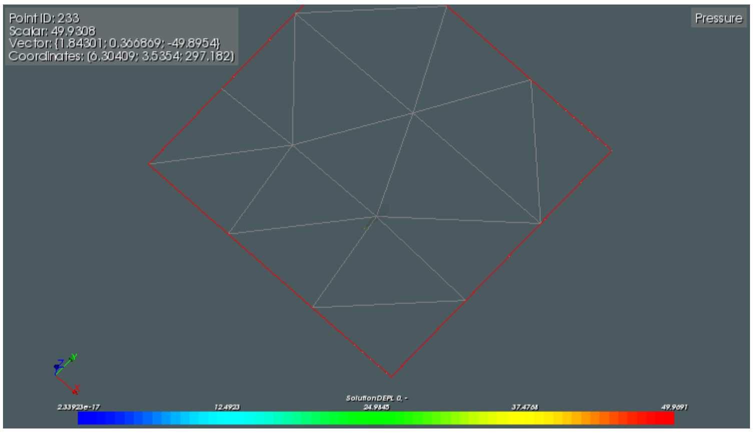


Figure 4.5: *Deformed shape of the two boxes.*



```
Point ID: 112
Scalar: 40.2918
Vector: {1.36965; 0.11546; -40.2683}
Coordinates: (10.0773; 0.0065205; 201.44)
```

Figure 4.6: *Deformed shape of the Box_2.*



```
Point ID: 233
Scalar: 49.9308
Vector: {1.84301; 0.366869; -49.8954}
Coordinates: (6.30409; 3.5354; 297.182)
```

Figure 4.7: *Displacements of the Pressure face nodes.*

Finally, the stress distribution along the boxes (Figure 4.8) shows a constant value of 10 N/cm^2, according to the theory.

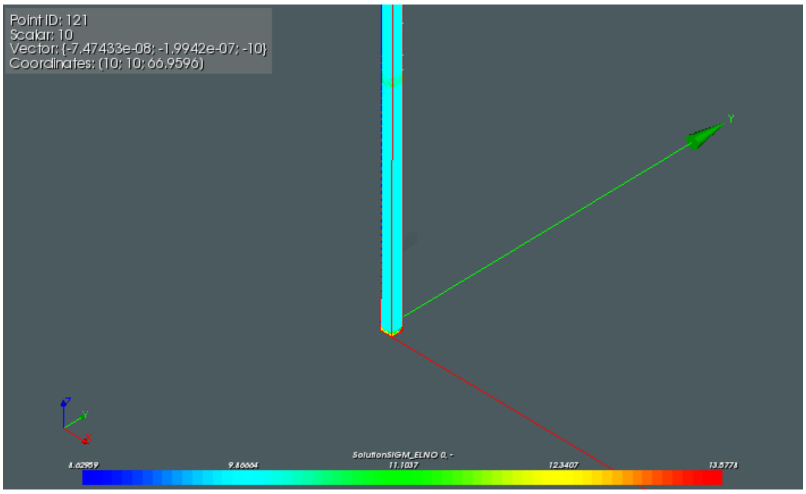

Point ID: 121
Scalar: 10
Vector: {-7.47433e-08; -1.9942e-07; -10}
Coordinates: (10; 10; 66.9596)

Figure 4.8: *Stress distribution along the boxes.*

4.2.3 Study 3

The Study 3 shows an object sphere-shaped with radius $R = 5$ cm (named `Fruit`) at the centre of which is placed an object parallelepiped-shaped with size 2 cm × 3 cm × 1 cm (named `Sensor`). The two objects are fused together and are of different materials. On a defined area of the sphere, a segment named `Pressure`, it was applied a pressure of 1000 N/cm^2, while the opposite area, a segment named `Base`, was regarded as the constraint. To locate both areas, it was constructed an arc lying on the sphere, subjected to a revolution of 360° around one of the Cartesian axis.

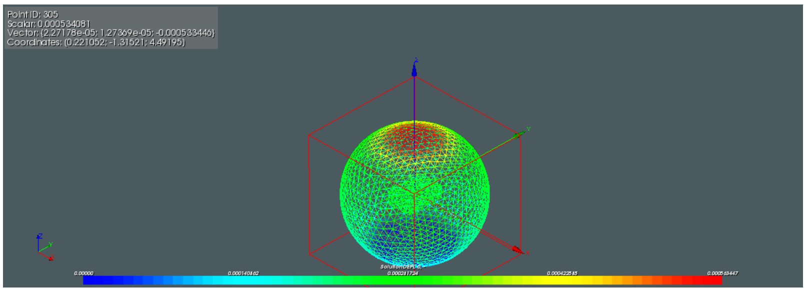
The goal of this test is to know the values of deformation and stress on the parallelepiped faces. Below it is showed the Python code.

```
### Loading libraries
import math
import salome
import GEOM
import geompy

salome.salome_init()
theStudy = salome.myStudy

### GEOM component
R = 5.  # sphere radius
dx = 2. # box size
dy = 3. # box size
dz = 1. # box size
alfad = 30. # angle used to define the arc, degree
alfar = alfad * math.pi/180. # radian

Sphere_1 = geompy.MakeSphereR(R)
Box_1 = geompy.MakeBoxDXDYDZ(dx, dy, dz)
geompy.TranslateDXDYDZ(Box_1, -dx/2., -dy/2., -dz/2.)
Origin = geompy.MakeVertex(0, 0, 0)

Fuse_1 = geompy.MakeFuse(Sphere_1, Box_1)

P1 = geompy.MakeVertex(0, R*math.sin(alfar), R*math.cos(alfar))
P2 = geompy.MakeVertex(0, 0, R)
Arc_1 = geompy.MakeArcCenter(Origin, P1, P2, False)
Vector_z = geompy.MakeVectorDXDYDZ(0, 0, 1)
Revolution_1 = geompy.MakeRevolution(Arc_1, Vector_z,
               360*math.pi/180.0)
Mirror_1 = geompy.MakeMirrorByPoint(Revolution_1, Origin)

Partition_1 = geompy.MakePartition([Fuse_1], [Box_1], [], [],
              geompy.ShapeType["SOLID"], 0, [], 0)

[Face_7, Face_8, Face_9, Face_10, Face_11, Face_12, Face_13] =
         geompy.ExtractShapes(Partition_1,
         geompy.ShapeType["FACE"], True)
[Solid_1, Solid_2] = geompy.ExtractShapes(Partition_1,
                     geompy.ShapeType["SOLID"], True)

geompy.addToStudy(Sphere_1, 'Sphere_1')
geompy.addToStudy(Box_1, 'Box_1')
geompy.addToStudy(Origin, 'Origin')
geompy.addToStudy(Fuse_1, 'Fuse_1')
geompy.addToStudy(P1, 'P1')
geompy.addToStudy(P2, 'P2')
geompy.addToStudy(Arc_1, 'Arc_1')
geompy.addToStudy(Vector_z, 'Vector_z')
geompy.addToStudy(Revolution_1, 'Revolution_1')
geompy.addToStudy(Mirror_1, 'Mirror_1')
geompy.addToStudy(Partition_1, 'Partition_1')

geompy.addToStudyInFather(Partition_1, Face_7, 'Face_7')
geompy.addToStudyInFather(Partition_1, Face_8, 'Face_8')
geompy.addToStudyInFather(Partition_1, Face_9, 'Face_9')
geompy.addToStudyInFather(Partition_1, Face_10, 'Face_10')
geompy.addToStudyInFather(Partition_1, Face_11, 'Face_11')
geompy.addToStudyInFather(Partition_1, Face_12, 'Face_12')
geompy.addToStudyInFather(Partition_1, Face_13, 'Face_13')
geompy.addToStudyInFather(Partition_1, Solid_1, 'Solid_1')
geompy.addToStudyInFather(Partition_1, Solid_2, 'Solid_2')

### SMESH component
import smesh, SMESH, SALOMEDS
import NETGENPlugin

Mesh_1 = smesh.Mesh(Partition_1)

NETGEN_2D3D = Mesh_1.Tetrahedron(algo=smesh.FULL_NETGEN)
```

```
NETGEN_3D_Parameters_1 = NETGEN_2D3D.Parameters()
NETGEN_3D_Parameters_1.SetMaxSize(0.5)
NETGEN_3D_Parameters_1.SetSecondOrder(0)
NETGEN_3D_Parameters_1.SetOptimize(1)
NETGEN_3D_Parameters_1.SetFineness(4)

isDone = Mesh_1.Compute()

F7 = Mesh_1.GroupOnGeom(Face_7, 'F7', SMESH.FACE)
F8 = Mesh_1.GroupOnGeom(Face_8, 'F8', SMESH.FACE)
F9 = Mesh_1.GroupOnGeom(Face_9, 'F9', SMESH.FACE)
F11 = Mesh_1.GroupOnGeom(Face_11, 'F11', SMESH.FACE)
F12 = Mesh_1.GroupOnGeom(Face_12, 'F12', SMESH.FACE)
F13 = Mesh_1.GroupOnGeom(Face_13, 'F13', SMESH.FACE)

# Filters to select Base and Pressure elements
# on the sphere surface
filter = smesh.GetFilter(smesh.FACE, smesh.FT_LyingOnGeom,
         Revolution_1, Tolerance=0.1)
ids = Mesh_1.GetIdsFromFilter(filter)
Pressure = Mesh_1.MakeGroupByIds('Pressure', smesh.FACE, ids)

filter = smesh.GetFilter(smesh.FACE, smesh.FT_LyingOnGeom,
         Mirror_1, Tolerance=0.1)
ids = Mesh_1.GetIdsFromFilter(filter)
Base = Mesh_1.MakeGroupByIds('Base', smesh.FACE, ids)

Sensor = Mesh_1.GroupOnGeom(Solid_1, 'Sensor', SMESH.VOLUME)
Fruit = Mesh_1.GroupOnGeom(Solid_2, 'Fruit', SMESH.VOLUME)

[F7, F8, F9, F11, F12, F13, Pressure, Base, Sensor, Fruit] =
     Mesh_1.GetGroups()
```

```
# set object names
smesh.SetName(Mesh_1.GetMesh(), 'Mesh_1')
smesh.SetName(NETGEN_2D3D.GetAlgorithm(), 'NETGEN_2D3D')
smesh.SetName(NETGEN_3D_Parameters_1, 'NETGEN 3D Parameters')
smesh.SetName(F7, 'F7')
smesh.SetName(F8, 'F8')
smesh.SetName(F9, 'F9')
smesh.SetName(F11, 'F11')
smesh.SetName(F12, 'F12')
smesh.SetName(F13, 'F13')
smesh.SetName(Pressure, 'Pressure')
smesh.SetName(Base, 'Base')
smesh.SetName(Sensor, 'Sensor')
smesh.SetName(Fruit, 'Fruit')
```

Figure 4.9 shows the mesh corresponding to the deformed shape, while Figure 4.10 shows the stress distribution on the `Sensor` block.



```
Point ID: 305
Scalar: 0.000534081
Vector: {2.27178e-05; 1.27369e-05; -0.000533446}
Coordinates: (0.221052; -1.31521; 4.49195)
```

Figure 4.9: *Deformed shape of the sphere.*

Moreover, the software allows to know, both in tabular and graphical format, the stress values along a defined line. For example, the `Face 11` of the `Sensor` (the upper side, in front of the segment where it was applied the load), was divided in 10 lines, and the values of stress along each line were plotted. In Figure 4.11 it is reported the stress distribution along the central line: the maximum value (1182 N/cm^2) is comparable with the applied load (1000 N/cm^2).

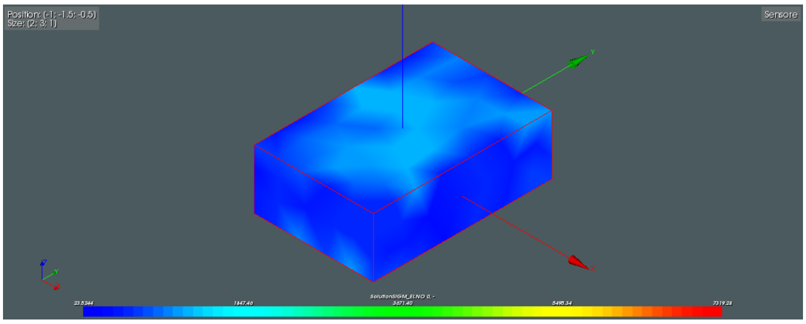


Figure 4.10: *Stress distribution on the sensor inside the sphere.*

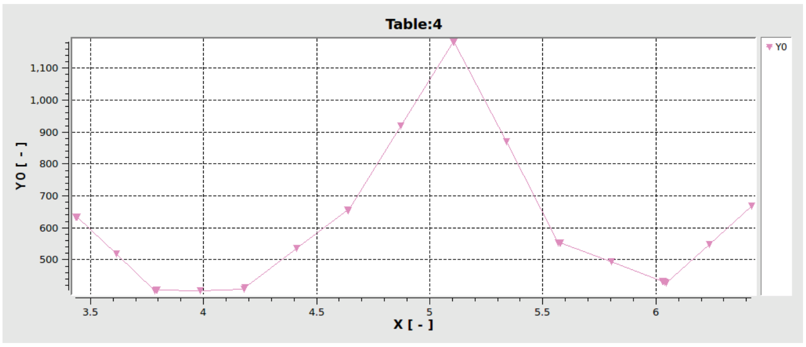


Figure 4.11: *Stress distribution along a line on the upper face of the sensor inside the sphere.*

4.2.4 Study 4

The purpose of Study 4 was to develop a model where the applied force varies in time. The temporal loading must be given in the form of a linear combination of constant forces assembled in time.

The model refers to a parallelepiped-shaped object with size (x, y and z values) of 0.1 m × 0.1 m × 0.4 m and modulus of Young $E = 10\,000$ Pa. This object is placed on a plane that imposes the boundary conditions (no movements of the base); on the opposite face it is applied an increasing pressure from 0 up to 1000 Pa. The corresponding theoretical value of deformation is:

$$\Delta L = \frac{\sigma}{E} L = 0.04\,\text{m}. \tag{4.14}$$

Below the Python code to build the model.

```
### Loading libraries
import sys
import salome
import GEOM
import geompy

salome.salome_init()
theStudy = salome.myStudy
geompy.init_geom(theStudy)

### GEOM component
dx = 0.1 # box size
dy = 0.1 # box size
dz = 0.4 # box size

Box_1 = geompy.MakeBoxDXDYDZ(dx, dy, dz)

[Face_1, Face_2, Base, Pressure, Face_5, Face_6] =
    geompy.ExtractShapes(Box_1, geompy.ShapeType["FACE"], True)

geompy.addToStudy(Box_1, 'Box_1')
geompy.addToStudyInFather(Box_1, Face_1, 'Face_1')
geompy.addToStudyInFather(Box_1, Face_2, 'Face_2')
geompy.addToStudyInFather(Box_1, Base, 'Base')
geompy.addToStudyInFather(Box_1, Pressure, 'Pressure')
geompy.addToStudyInFather(Box_1, Face_5, 'Face_5')
geompy.addToStudyInFather(Box_1, Face_6, 'Face_6')

### SMESH component
import smesh, SMESH, SALOMEDS
import NETGENPlugin

Mesh_1 = smesh.Mesh(Box_1)
```

```
NETGEN_2D3D = Mesh_1.Tetrahedron(algo=smesh.FULL_NETGEN)
NETGEN_3D_Parameters = NETGEN_2D3D.Parameters()
NETGEN_3D_Parameters.SetMaxSize(0.0424264)
NETGEN_3D_Parameters.SetSecondOrder(0)
NETGEN_3D_Parameters.SetOptimize(1)
NETGEN_3D_Parameters.SetFineness(4)

isDone = Mesh_1.Compute()

Base_1 = Mesh_1.GroupOnGeom(Base, 'Base', SMESH.FACE)
Pressure_1 = Mesh_1.GroupOnGeom(Pressure, 'Pressure', SMESH.FACE)
smesh.SetName(Mesh_1, 'Mesh_1')

# set object names
smesh.SetName(Mesh_1.GetMesh(), 'Mesh_1')
smesh.SetName(NETGEN_2D3D.GetAlgorithm(), 'NETGEN_2D3D')
smesh.SetName(NETGEN_3D_Parameters, 'NETGEN 3D Parameters')
smesh.SetName(Base_1, 'Base')
smesh.SetName(Pressure_1, 'Pressure')
```

When the mesh and all the geometrical entities (faces and volumes) have been created, the mesh was exported so that Code-Aster could read the data. The load was applied in steps defined by a time function and a multiplication factor. The time step is defined by the DEFI_LIST_REEL command, whereas the scaling function castle is defined by the DEFI_FONCTION command. In total 200 time steps were generated. The solution process is managed by the DYNA_LINE_TRAN command.

The file command is showed below.

```
DEBOUT();

# Read MED MESH File
MeshLin=LIRE_MAILLAGE(UNITE=20,
                      FORMAT='MED',
                      NOM_MED='Mesh_1',
                      INFO_MED=1,);

FEMLin=AFFE_MODELE(MAILLAGE=MeshLin,
                   AFFE=_F(TOUT='OUI',
                          PHENOMENE='MECANIQUE',
                          MODELISATION='3D',),);

steel=DEFI_MATERIAU(ELAS=_F(E=10000,
                           NU=0.28,
                           RHO=7850,
                           AMOR_ALPHA=0.1,),);

Mat=AFFE_MATERIAU(MAILLAGE=MeshLin,
                  MODELE=FEMLin,
                  AFFE=_F(TOUT='OUI',
                         MATER=steel,),);

# Boundary conditions
BCnd=AFFE_CHAR_MECA(MODELE=FEMLin,
                    DDL_IMPO=(_F(GROUP_MA='Base',
                                 DX=0.0,
                                 DY=0.0,
                                 DZ=0.0,),
                              _F(GROUP_MA='Pressure',
                                 DX=0.0,
                                 DY=0.0,),),
                    PRES_REP=_F(GROUP_MA='Pressure',
                                PRES=1000,),);

# Multiplication factor on the load
tsteps = 200;
t0 = 0.0;
```

```
tc = 0.002;
t1 = (1.00001 * tc);
te = 5;

time=DEFI_LIST_REEL(DEBUT=t0,
                    INTERVALLE=_F(JUSQU_A=te,
                                  NOMBRE=tsteps,),
                    INFO=2,
                    TITRE='time',);

castle=DEFI_FONCTION(NOM_PARA='INST',
                     VALE=(t0, 0.00,
                           tc, 0.00,
                           t1, 1.00,
                           te, 1.00,),
                     INFO=2,
                     TITRE='castle',);

MACRO_MATR_ASSE(MODELE=FEMLin,
                CHAM_MATER=Mat,
                CHARGE=BCnd,
                NUME_DDL=CO('numdof'),
                MATR_ASSE=(_F(MATRICE=CO('Mstiff'),
                              OPTION='RIGI_MECA',),
                           _F(MATRICE=CO('Mmasse'),
                              OPTION='MASS_MECA',),
                           _F(MATRICE=CO('Mdampg'),
                              OPTION='AMOR_MECA',),),);

solu=DYNA_LINE_TRAN(MODELE=FEMLin,
                    CHAM_MATER=Mat,
                    MATR_MASS=Mmasse,
                    MATR_RIGI=Mstiff,
                    MATR_AMOR=Mdampg,
                    SCHEMA_TEMPS=_F(BETA=0.25,
                                    GAMMA=0.5,),
                    EXCIT=_F(CHARGE=BCnd,
                             FONC_MULT=castle,),
                    INCREMENT=_F(LIST_INST=time,),);

# Write Results to MED file
solu=CALC_ELEM(reuse=solu,
               RESULTAT=solu,
               OPTION=('SIEF_ELNO', 'SIEQ_ELNO',),);

IMPR_RESU(FORMAT='MED',
          UNITE=80,
          RESU=_F(MAILLAGE=MeshLin,
                  RESULTAT=solu,
                  NOM_CHAM=('DEPL', 'ACCE', 'VITE',),),);
FIN();
```

In the post-processor module the interesting lines are `Solution DEPL`, `Solution VITE` and `Solution ACCE`. These three groups are the displacements, the velocities and the accelerations, respectively.

It is possible to select interactively a node belonging to the object and display its curve trend of displacement, velocity and acceleration with respect to the time. It was selected the `Node 5`, belonging to the face on the top of the box. Figure 4.12 reports the modulus of the displacement.

After the initial oscillations due to the linear increasing in the load applied on the top of the box, the displacement approaches 0.04 m. Therefore the theoretical value of deformation agrees with that achieved by the simulation.

The vertical component of the velocity trend of the same node is reported in Figure 4.13. After the initial oscillations, it approaches zero, as it should be. Finally, Figure 4.14 shows the acceleration trend.

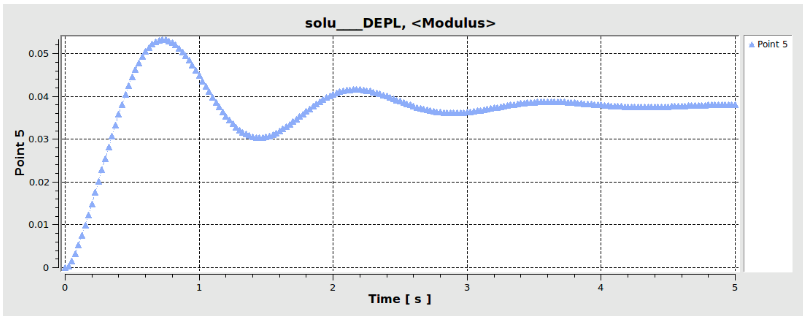


Figure 4.12: *Displacement of the "Node 5" over the time.*

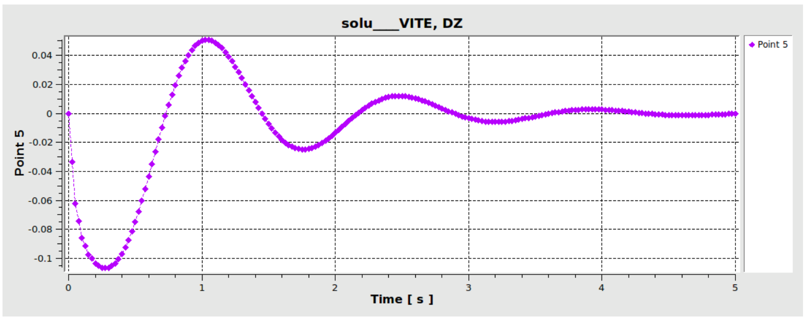


Figure 4.13: *Velocity of the "Node 5" over time.*

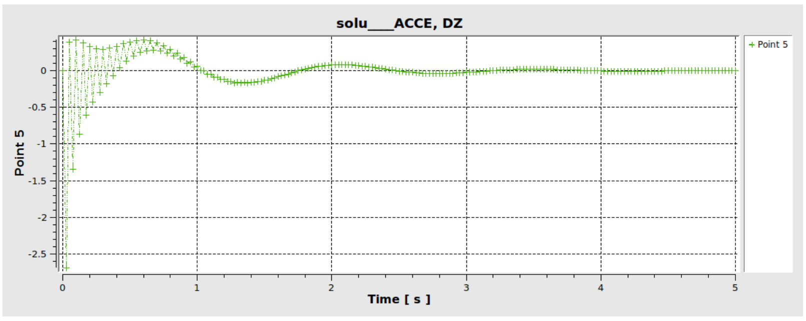


Figure 4.14: *Acceleration of the "Node 5" over time.*

4.2.5 Study 5

The Study 5 models a sphere (`fruit`) with radius of 0.05 m, at the centre of which is placed an object parallelepiped-shaped (`sensor`) with size of 0.02 m $\times$ 0.02 m $\times$ 0.02 m. The sphere is placed at a distance of 0.05 m above a 1 m $\times$ 1 m $\times$ 0.1 m plane. The sphere falls in the z direction on the plane considering only the force of gravity. The boundary conditions are: zero translational displacement for the rigid flat plan, zero displacements in the x and y-direction for the sphere. Below the Python code of the Study 5.

```
### Loading of libraries
import sys
import salome
salome.salome_init()
theStudy = salome.myStudy

### GEOM component
import GEOM
import geompy
import math
import SALOMEDS

DXp = 1.0 # plane size
DYp = 1.0
DZp = 0.1

DXs = 0.02 # sensor size
DYs = 0.02
DZs = 0.02

R = 0.05 # sphere radius

DZ = DZp + 0.10 # translation of the sphere above the plane

geompy.init_geom(theStudy)

plane = geompy.MakeBoxDXDYDZ(DXp, DYp, DZp)
geompy.TranslateDXDYDZ(plane, -DXp/2, -DYp/2, 0)

geompy.addToStudy(plane, 'plane')
```

```
[Face_1, Face_2, z, floor, y, x] = geompy.ExtractShapes(plane,
        geompy.ShapeType["FACE"], True)

geompy.addToStudyInFather(plane, z, 'z')
geompy.addToStudyInFather(plane, floor, 'floor')
geompy.addToStudyInFather(plane, y, 'y')
geompy.addToStudyInFather(plane, x, 'x')

sphere = geompy.MakeSphereR(R)

sensor = geompy.MakeBoxDXDYDZ(DXs, DYs, DZs)
geompy.TranslateDXDYDZ(sensor, -DXs/2, -DYs/2, -DZs/2)

geompy.addToStudy(sphere, 'sphere')
geompy.addToStudy(sensor, 'sensor')

Fuse_1 = geompy.MakeFuse(sphere, sensor)
Cut_1 = geompy.MakeCut(sphere, sensor)

geompy.addToStudy(Fuse_1, 'Fuse_1')
geompy.addToStudy(Cut_1, 'Cut_1')

Partition_1 = geompy.MakePartition([Fuse_1],
   [sensor, Cut_1], [], [], geompy.ShapeType["SOLID"], 0, [], 0)

geompy.TranslateDXDYDZ(Partition_1, 0, 0, DZ)

[sensor, fruit] = geompy.ExtractShapes(Partition_1,
    geompy.ShapeType["SOLID"], True)

[Face_1, Face_2, shell] = geompy.ExtractShapes(Partition_1,
    geompy.ShapeType["SHELL"], True)
```

```
[Face_1, Face_2, Face_3, Face_4, Face_5, Face_6, Face_7] =
   geompy.ExtractShapes(Partition_1,
   geompy.ShapeType["FACE"], True)

geompy.addToStudy(Partition_1, 'Partition_1' )
geompy.addToStudyInFather(Partition_1, Face_1, 'Face_1')
geompy.addToStudyInFather(Partition_1, Face_2, 'Face_2')
geompy.addToStudyInFather(Partition_1, Face_3, 'Face_3')
geompy.addToStudyInFather(Partition_1, Face_5, 'Face_5')
geompy.addToStudyInFather(Partition_1, Face_6, 'Face_6')
geompy.addToStudyInFather(Partition_1, Face_7, 'Face_7')
geompy.addToStudyInFather(Partition_1, shell, 'shell')
geompy.addToStudyInFather(Partition_1, sensor, 'sensor')
geompy.addToStudyInFather(Partition_1, fruit, 'fruit')

### SMESH component
import smesh, SMESH, SALOMEDS

smesh.SetCurrentStudy(theStudy)
import NETGENPlugin

Mesh_plane = smesh.Mesh(plane)

NETGEN_2D3D = Mesh_plane.Tetrahedron(algo=smesh.FULL_NETGEN)
NETGEN_3D_Parameters = NETGEN_2D3D.Parameters()
NETGEN_3D_Parameters.SetMaxSize(0.1)
NETGEN_3D_Parameters.SetSecondOrder(0)
NETGEN_3D_Parameters.SetOptimize(1)
NETGEN_3D_Parameters.SetFineness(2)
isDone = Mesh_plane.Compute()

z_1 = Mesh_plane.GroupOnGeom(z, 'z', SMESH.FACE)
floor_1 = Mesh_plane.GroupOnGeom(floor, 'floor', SMESH.FACE)
```

```
y_1 = Mesh_plane.GroupOnGeom(y, 'y', SMESH.FACE)
x_1 = Mesh_plane.GroupOnGeom(x, 'x', SMESH.FACE)
plane_1 = Mesh_plane.GroupOnGeom(plane, 'plane', SMESH.VOLUME)

[z_1, floor_1, y_1, x_1, plane_1] = Mesh_plane.GetGroups()

Mesh_Partition_1 = smesh.Mesh(Partition_1)

NETGEN_2D3D_1 =
    Mesh_Partition_1.Tetrahedron(algo=smesh.FULL_NETGEN)
NETGEN_3D_Parameters_1 = NETGEN_2D3D_1.Parameters()
NETGEN_3D_Parameters_1.SetMaxSize(0.01)
NETGEN_3D_Parameters_1.SetSecondOrder(0)
NETGEN_3D_Parameters_1.SetOptimize(1)
NETGEN_3D_Parameters_1.SetFineness(2)
isDone = Mesh_Partition_1.Compute()

fruit_1 = Mesh_Partition_1.GroupOnGeom(fruit, 'fruit',
    SMESH.VOLUME)

sensor_1 = Mesh_Partition_1.GroupOnGeom(sensor, 'sensor',
    SMESH.VOLUME)
shell_1 = Mesh_Partition_1.GroupOnGeom(shell, 'shell',
    SMESH.FACE)
Face_1_1 = Mesh_Partition_1.GroupOnGeom(Face_1, 'Face_1',
    SMESH.FACE)
Face_2_1 = Mesh_Partition_1.GroupOnGeom(Face_2, 'Face_2',
    SMESH.FACE)
Face_3_1 = Mesh_Partition_1.GroupOnGeom(Face_3, 'Face_3',
    SMESH.FACE)
Face_5_1 = Mesh_Partition_1.GroupOnGeom(Face_5, 'Face_5',
    SMESH.FACE)
Face_6_1 = Mesh_Partition_1.GroupOnGeom(Face_6, 'Face_6',
    SMESH.FACE)
Face_7_1 = Mesh_Partition_1.GroupOnGeom(Face_7, 'Face_7',
    SMESH.FACE)

Mesh_1 = smesh.Concatenate([Mesh_plane.GetMesh(),
    Mesh_Partition_1.GetMesh()], 1, 0, 1e-05)

[z_2, floor_2, y_2, x_2, plane_2, fruit_2, sensor_2, shell_2,
    Face_1_2, Face_2_2, Face_3_2, Face_5_2, Face_6_2,
    Face_7_2 ] = Mesh_1.GetGroups()
[z_1, floor_1, y_1, x_1, plane_1] = Mesh_plane.GetGroups()
[fruit_1, sensor_1, shell_1, Face_1_1, Face_2_1, Face_3_1,
    Face_5_1, Face_6_1, Face_7_1] = Mesh_Partition_1.GetGroups()
[z_2, floor_2, y_2, x_2, plane_2, fruit_2, sensor_2, shell_2,
    Face_1_2, Face_2_2, Face_3_2, Face_5_2, Face_6_2,
    Face_7_2] = Mesh_1.GetGroups()

## Set object names
smesh.SetName(Mesh_plane.GetMesh(), 'Mesh_plane')
smesh.SetName(NETGEN_2D3D.GetAlgorithm(), 'NETGEN_2D3D')
smesh.SetName(NETGEN_3D_Parameters, 'NETGEN 3D Parameters')

smesh.SetName(z_1, 'z')
smesh.SetName(floor_1, 'floor')
smesh.SetName(y_1, 'y')
smesh.SetName(x_1, 'x')
smesh.SetName(plane_1, 'plane')

smesh.SetName(Mesh_Partition_1.GetMesh(), 'Mesh_Partition_1')
smesh.SetName(NETGEN_3D_Parameters_1, 'NETGEN 3D Parameters')

smesh.SetName(fruit_1, 'fruit')
smesh.SetName(sensor_1, 'sensor')
smesh.SetName(shell_1, 'shell')
smesh.SetName(Face_1_1, 'Face_1')
smesh.SetName(Face_2_1, 'Face_2')
smesh.SetName(Face_3_1, 'Face_3')
smesh.SetName(Face_5_1, 'Face_5')
smesh.SetName(Face_6_1, 'Face_6')
smesh.SetName(Face_7_1, 'Face_7')

smesh.SetName(Mesh_1.GetMesh(), 'Mesh_1')


smesh.SetName(z_2, 'z')
smesh.SetName(floor_2, 'floor')
smesh.SetName(y_2, 'y')
smesh.SetName(x_2, 'x')
smesh.SetName(plane_2, 'plane')
smesh.SetName(fruit_2, 'fruit')
smesh.SetName(sensor_2, 'sensor')
smesh.SetName(shell_2, 'shell')
smesh.SetName(Face_1_2, 'Face_1')
smesh.SetName(Face_2_2, 'Face_2')
smesh.SetName(Face_3_2, 'Face_3')
smesh.SetName(Face_5_2, 'Face_5')
smesh.SetName(Face_6_2, 'Face_6')
smesh.SetName(Face_7_2, 'Face_7')
```

In Code-Aster the contact between the two objects is defined by the `DEFI_CONTACT` command. This command makes it possible to describe the areas subjected to conditions of unilateral contact with or without friction. The solution process is managed by the `DYNA_NON_LINE` command. Below it is showed the file comm.

```
DEBUT();

# Read MED MESH File
```

```
MeshLin=LIRE_MAILLAGE(UNITE=20,
                      FORMAT='MED',
                      NOM_MED='Mesh_1',
                      INFO_MED=1,);

# Assigns a physical model to geometric entities.
FEMLin=AFFE_MODELE(MAILLAGE=MeshLin,
                   AFFE=_F(TOUT='OUI',
                           PHENOMENE='MECANIQUE',
                           MODELISATION='3D',),);

# Material properties
Steel=DEFI_MATERIAU(ELAS=_F(E=2.1e11,
                            NU=0.27,
                            RHO=7800.0,),);

Fruit=DEFI_MATERIAU(ELAS=_F(E=1000000,
                            NU=0.35,
                            RHO=980,),);

# Assign Material properties to Elements
Mat=AFFE_MATERIAU(MAILLAGE=MeshLin,
                  MODELE=FEMLin,
                  AFFE=(_F(GROUP_MA='plane',
                           MATER=Steel,),
                        _F(GROUP_MA='fruit',
                           MATER=Fruit,),
                        _F(GROUP_MA='sensor',
                           MATER=Steel,),),);

# Boundary conditions
BCnd=AFFE_CHAR_MECA(MODELE=FEMLin,
                    DDL_IMPO=(_F(GROUP_MA='x',
                                    DX=0.0,),
                                 _F(GROUP_MA='y',
                                    DY=0,),
                                 _F(GROUP_MA='z',
                                    DZ=0,),),);

gravity=AFFE_CHAR_MECA(MODELE=FEMLin,
                       PESANTEUR=_F(GRAVITE=9.81,
                                    DIRECTION=(0, 0, -1,),),);

pre_velo=CREA_CHAMP(TYPE_CHAM='NOEU_DEPL_R',
                    OPERATION='AFFE',
                    MAILLAGE=MeshLin,
                    AFFE=_F(GROUP_MA=('fruit', 'sensor',),
                            NOM_CMP=('DX', 'DY', 'DZ',),
                            VALE=(0, 0, 0,),),);

contact=DEFI_CONTACT(MODELE=FEMLin,
                     FORMULATION='DISCRETE',
                     REAC_GEOM='AUTOMATIQUE',
                     ZONE=_F(APPARIEMENT='MAIT_ESCL',
                             GROUP_MA_MAIT='floor',
                             GROUP_MA_ESCL='shell',
                             ALGO_CONT='LAGRANGIEN',),);

timelst=DEFI_LIST_REEL(DEBUT=0,
                       INTERVALLE=_F(JUSQU_A=1,
                                     NOMBRE=1000,),);

# Finite Element Solution
solu=DYNA_NON_LINE(MODELE=FEMLin,
                   CHAM_MATER=Mat,
                   EXCIT=(_F(CHARGE=BCnd,),
                                   _F(CHARGE=gravity,),),
                            CONTACT=contact,
                            COMP_ELAS=_F(RELATION='ELAS',
                                         DEFORMATION='GROT_GDEP',
                                         ITER_INTE_MAXI=100,
                                         TOUT='OUI',),
                            ETAT_INIT=_F(VITE=pre_velo,
                                         PRECISION=1e-04,),
                            INCREMENT=_F(LIST_INST=timelst,),
                            SCHEMA_TEMPS=_F(SCHEMA='NEWMARK',
                                            FORMULATION='DEPLACEMENT',),
                            NEWTON=_F(REAC_INCR=1,
                                      PREDICTION='ELASTIQUE',
                                      MATRICE='TANGENTE',
                                      REAC_ITER=1,
                                      REAC_ITER_ELAS=1,),
                            CONVERGENCE=_F(ITER_GLOB_MAXI=500,),
                            SOLVEUR=_F(SYME='OUI',),
                            INFO=1,);

# Compute VonMises Stress / Strain
solu=CALC_ELEM(reuse=solu,
               RESULTAT=solu,
               OPTION=('SIEF_ELNO', 'SIEQ_ELNO',),);

solu=CALC_NO(reuse=solu,
             RESULTAT=solu,
             OPTION='SIEQ_NOEU',);

# Write Results to MED file
IMPR_RESU(MODELE=FEMLin,
          FORMAT='MED',
          UNITE=80,
          RESU=_F(MAILLAGE=MeshLin,
                  RESULTAT=solu,
                  NOM_CHAM=('DEPL', 'VITE', 'ACCE',),),);

FIN(FORMAT_HDF='OUI',);
```

In post-processing, a node belonging to the bottom of the sphere (`Node 295`), where it happens the contact with the plane, was selected. Its displacement along the *z*-axis is reported in Figure 4.15.

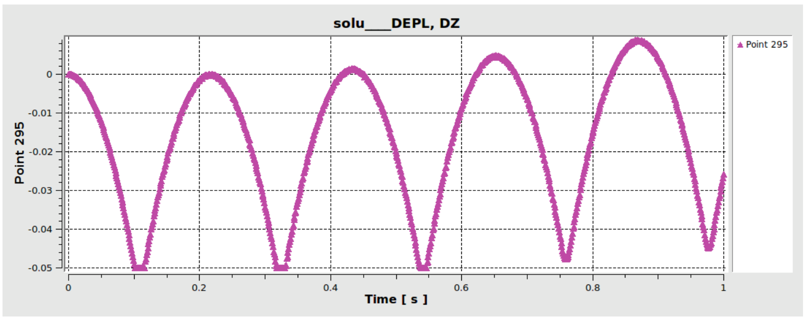


Figure 4.15: *Displacement along the z-axis of the "Node 295" over the time.*

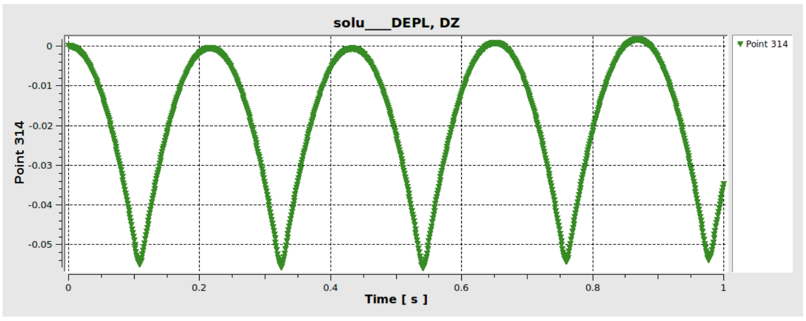


Figure 4.16: *Displacement along the z-axis of the "Node 314" over time.*

The curve trend shows rebound peaks of around 0.05 m, that is the distance between the sphere and the plane. At the end of the curve the values of the peaks tends to move away from the theoretical value probably because of instability of the model over a long period of simulation.

A second node (314) was selected on the bottom face of the sensor. Its displacement is shown in Figure 4.16.

It can be noticed that the maximum displacement is greater than 0.05 m because of the compression of the sphere.

4.2.6 Study 6

The goal of the Study 6 was to know what happens during the first impact between sphere and plane. The simulation was carried out as in Study 5, but with different sampling time. In the operand `INTERVALLE` of the `DEFI_LIST_REEL` command, in fact, the end of the interval (`JUSQU_A`=0.2) is set to 0.2, whereas the number of steps is set to 1000.

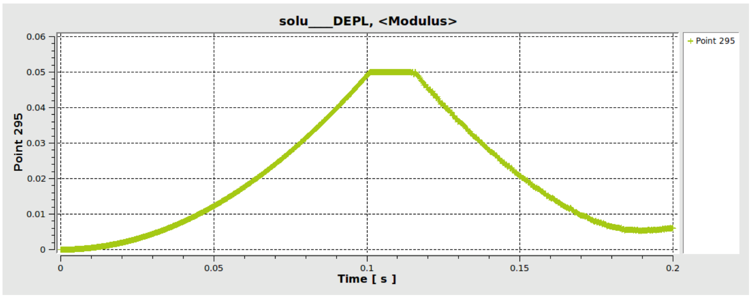


Figure 4.17: *Modulus of the displacement of the "Node 295" over time.*

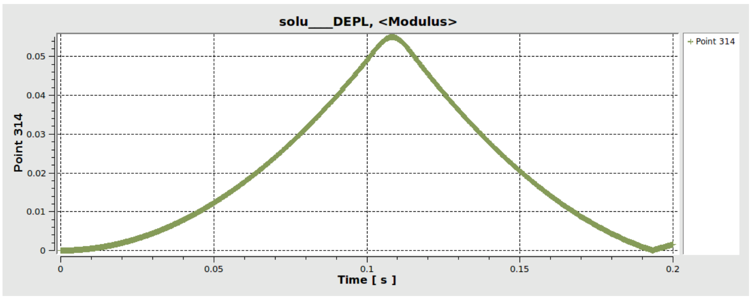


Figure 4.18: *Modulus of the displacement of the "Node 314" over time.*

Figures 4.17 and 4.18 report the displacements of the same nodes (295 and 314). Figure 4.17 allows calculation of contact time between sphere and plane (the trait with constant displacement), while Figure 4.18 allows calculation the maximum compression of the sphere.

4.3 Measurement of the modulus of Young

Figure 4.19 reports the box-plots of the modulus of Young measurements at varying artificial fruit and probe diameter.

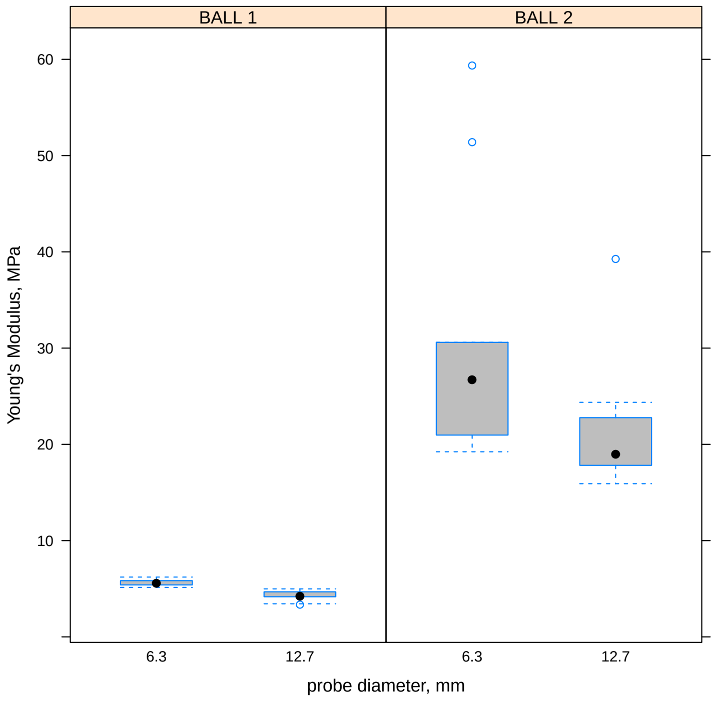


Figure 4.19: *Box-plots of the modulus of Young values at varying artificial fruits and probe diameters.*

The Figure shows significant differences between the moduli of Young for the same artificial fruit at varying the probe diameter. In fact, the analysis of variance (ANOVA), with artificial fruit and probe as source of variation and modulus of Young as dependent variable, showed significant differences due both to artificial fruits and probes (Table 4.1).

Table 4.1: *Results of the analysis of variance on modulus of Young values.*

Source of variation	Df	F value	Pr(>F)	
Artificial fruit	1	74.585	2.68e-10	***
Probe	1	4.702	0.0368	*
Interaction	1	2.553	0.1188	ns
Residuals	36			

***: p-level=0.001; *: p-level=0.05; ns: not significant; Df: degree of freedom.

This could be related to some inaccuracies in measuring the small deformations during the compression tests, that should be different with the two probes. But the ANOVA on deformation values (Table 4.2) showed that this parameter was not significantly affected by the probe type.

Table 4.2: *Results of the analysis of variance on deformation values.*

Source of variation	Df	F value	Pr(>F)	
Artificial fruit	1	883.507	<2e-16	***
Probe	1	0.094	0.761	ns
Interaction	1	0.062	0.805	ns
Residuals	36			

***: p-level=0.001; ns: not significant; Df: degree of freedom.

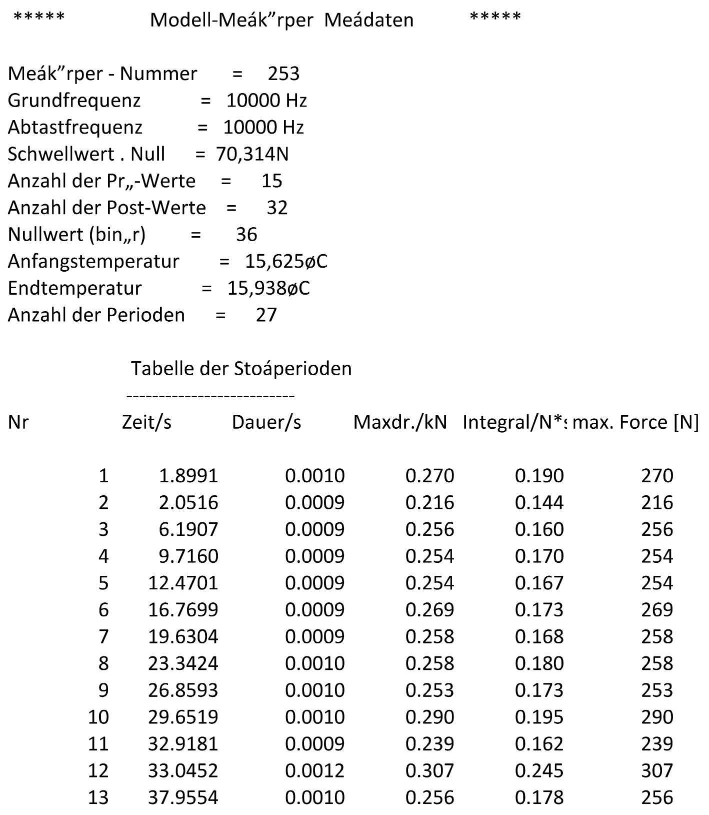
Taking into account these observations, the mean values of deformations and moduli of Young were those reported in Table 4.3.

Table 4.3: *Mean values of deformations and moduli of Young (E).*

	Deformation, mm		E, MPa	
Artificial fruit	Probe 1	Probe 2	Probe 1	Probe 2
BALL 1	0.240	0.243	4.3	5.6
BALL 2	0.086	0.087	21.3	30.5

4.4 Drop tests

Figure 4.20 reports an example of the file provided by the force impact recorder. It contains a list with all the impacts and, for each impact, the time of occurrence, its duration, the maximum force and the time integral of the force profile.



```
*****          Modell-Meák"rper  Meádaten          *****

Meák"rper - Nummer      =    253
Grundfrequenz           =  10000 Hz
Abtastfrequenz          =  10000 Hz
Schwellwert . Null      = 70,314N
Anzahl der Pr„-Werte    =    15
Anzahl der Post-Werte   =    32
Nullwert (bin„r)        =    36
Anfangstemperatur       =  15,625øC
Endtemperatur           =  15,938øC
Anzahl der Perioden     =    27

                 Tabelle der Stoáperioden
                 -------------------------
```

Nr	Zeit/s	Dauer/s	Maxdr./kN	Integral/N*s	max. Force [N]
1	1.8991	0.0010	0.270	0.190	270
2	2.0516	0.0009	0.216	0.144	216
3	6.1907	0.0009	0.256	0.160	256
4	9.7160	0.0009	0.254	0.170	254
5	12.4701	0.0009	0.254	0.167	254
6	16.7699	0.0009	0.269	0.173	269
7	19.6304	0.0009	0.258	0.168	258
8	23.3424	0.0010	0.258	0.180	258
9	26.8593	0.0010	0.253	0.173	253
10	29.6519	0.0010	0.290	0.195	290
11	32.9181	0.0009	0.239	0.162	239
12	33.0452	0.0012	0.307	0.245	307
13	37.9554	0.0010	0.256	0.178	256

Figure 4.20: *Example of file with force data.*

All data were grouped for artificial fruit, impact material and drop height. Rebounds were excluded from the analyses. The box-plots of the impact force values are reported in Figure 4.21, whereas average values are reported in Table 4.4.

On average, with BALL 1 and with drop height from 10 cm, the impact

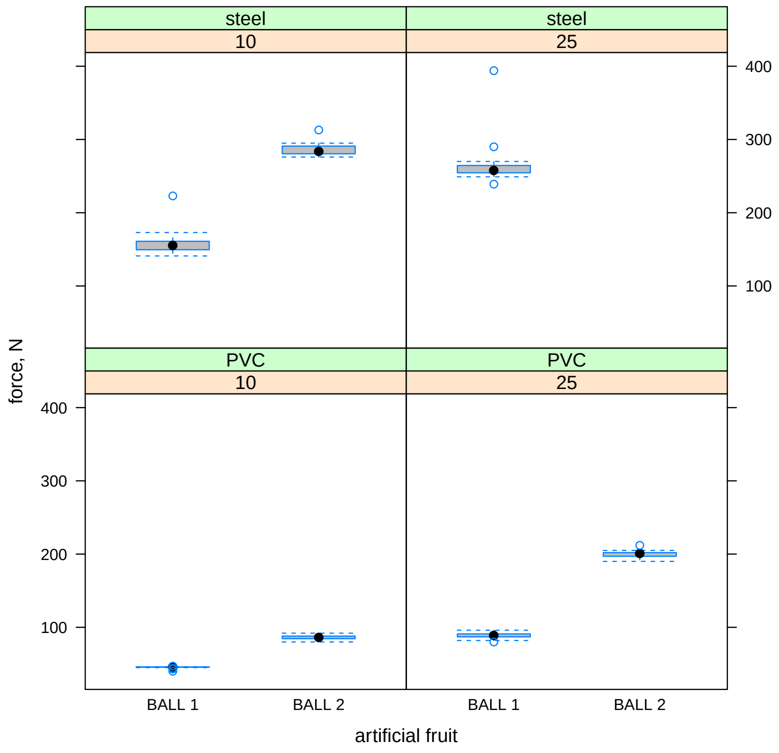


Figure 4.21: *Box-plots of the force values measured during the drop tests with the two artificial fruits.*

force increased 3.5 times (from 45 to 158 N) changing the impact material (from PVC to steel). When the drop height was 25 cm, the increase was on average 3.0 times (from 89 to 266 N).

When testing BALL 2, the increase was 3.3 times with drop height of 10 cm. So, on average, the PVC reduced the impact force to one third respect to the steel.

When comparing the two artificial fruits, the second (BALL 2) produced impact forces greater from 1.8 to 2.2 times those produced by BALL 1: this results is mainly due to its greater mass, density and modulus of Young.

Table 4.4: *Mean values of impact force measured during the drop tests.*

Artificial fruit	Drop height, cm	Impact material	Force, N
BALL 1	10	PVC	45.4
BALL 1	10	steel	158.3
BALL 1	25	PVC	89.1
BALL 1	25	steel	266.2
BALL 2	10	PVC	86.0
BALL 2	10	steel	286.4
BALL 2	25	PVC	199.9
BALL 2	25	steel	–

4.5 Drop test simulation

4.5.1 Preliminary simulations

Drop tests were simulated with Salome-Meca and Code-Aster software to asses whether measured and simulated forces were in agreement when using artificial fruit parameters (diameter, density, modulus of Young) comparable with those measured experimentally.

Figure 4.22 reports the geometric model used for the simulations. The simulated impact force was that on the base of the steel disk.

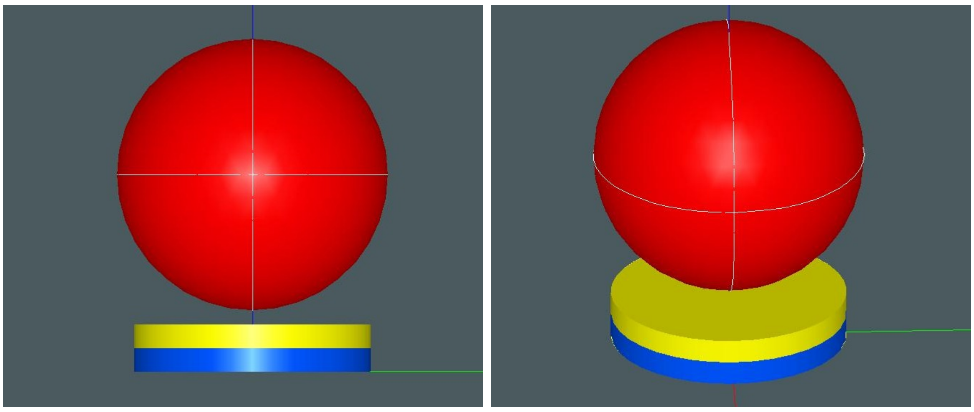


Figure 4.22: *Geometric model used for the drop tests.*

The first simulations were carried out considering impacts against steel and changing the modulus of elasticity (E) of the sphere materials. The density of the spheres was kept constant, being this parameter more easily measurable. The coefficient of Poisson was kept constant for both spheres (0.49)

When an acceptable agreement between simulated and measured values was reached, the modulus E of the spheres was kept constant and the impacts against PVC were simulated, changing its E and density.

The final parameter values used for simulations are those reported in Table 4.5.

Table 4.5: *Parameters used for the drop tests.*

	Modulus of Young, MPa		Density,	Coefficient of
Material	Measured	Simulated	kg/m^3	Poisson
BALL 1	4.25–5.64	14.8	338	0.49
BALL 2	21.36–30.51	22.4	1350	0.49
PVC		0.425	500	0.49
Steel		210000	7800	0.27

Figure 4.23 reports the comparison between simulated and measured force.

Even if further experimental test could better validate the model, the overall results were quite good: the values of E for both artificial fruits found with the simulations were of the same order of magnitude of those experimentally measured in Potsdam, based on the Hertz theory. In particular, the simulated value of the modulus of Young for BALL 2 (22.4 MPa) was in the range between the two measured values (21.36–30.51 MPa). Instead, as far as BALL 1 is concerned, the simulated value of the modulus of Young (14.8 MPa) was 2.6–3.5 times as much the measured values, but nevertheless it was considered acceptable, also considering the variability in the experimental measurements when using different probes.

As far as the impact force, only for one test (impact material = PVC, drop height = 25 cm, artificial fruit = BALL 2) the simulated force was under

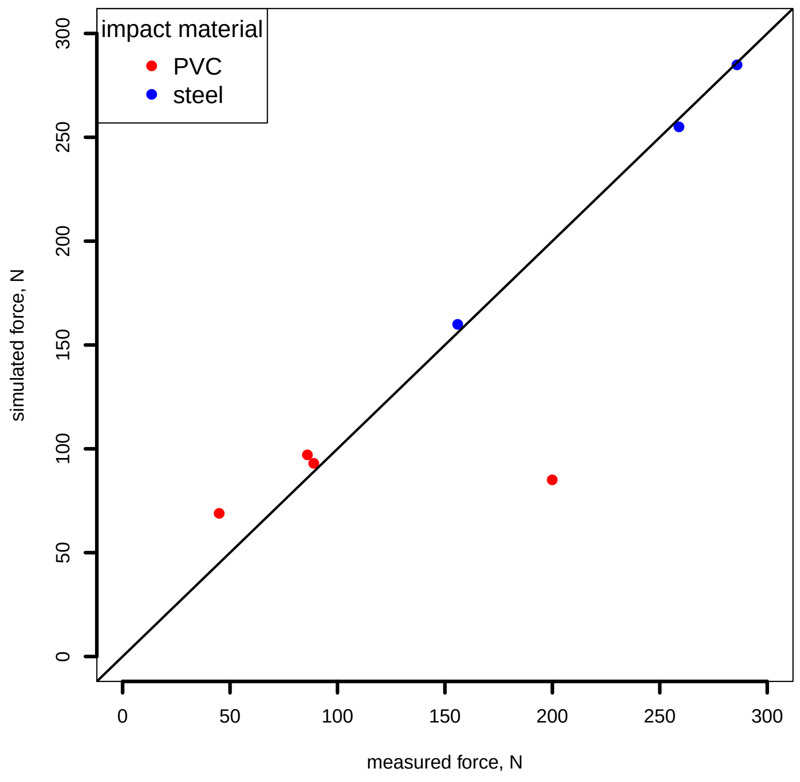


Figure 4.23: *Comparison between simulated and measured force in drop tests.*

estimated with respect to the measured one. This could be due to the lack of information in modelling the PVC behaviour.

4.5.2 Model building

Gained good agreement between simulated and measured parameters, the model was extensively used for other analyses on the impact forces and accelerations. Below the Python code to build the model. The several models differ for the size of the sphere only.

```
import sys
import salome

salome.salome_init()
theStudy = salome.myStudy
```

```
### GEOM component
import GEOM
import geompy
import math
import SALOMEDS

geompy.init_geom(theStudy)

RC = 0.025 # steel plate cilinder radius, m
DZ = 0.005 # steel plate cilinder height, m

Cilindro = geompy.MakeCylinderRH(RC, DZ)
[Base, Face_2, Piano] = geompy.ExtractShapes(Cilindro,
       geompy.ShapeType["FACE"], True)

# sphere
RS = 55.0/2000.0 # Sphere radius, m
Sfera = geompy.MakeSphereR(RS)
DH = 0.003 # drop height when simulation starts, m

geompy.TranslateDXDYDZ(Sfera, 0, 0, DZ+RS+DH)
[Guscio] = geompy.ExtractShapes(Sfera,
           geompy.ShapeType["FACE"], True)

Vertex_1 = geompy.MakeVertex(0, 0, DZ+RS+DH)

geompy.addToStudy(Cilindro, 'Cilindro')
geompy.addToStudy(Sfera, 'Sfera')
geompy.addToStudy(Vertex_1, 'Vertex_1')

geompy.addToStudyInFather(Sfera, Guscio, 'Guscio')
geompy.addToStudyInFather(Cilindro, Base, 'Base')
geompy.addToStudyInFather(Cilindro, Face_2, 'Face_2')
geompy.addToStudyInFather(Cilindro, Piano, 'Piano')

### SMESH component
import smesh, SMESH, SALOMEDS

smesh.SetCurrentStudy(theStudy)
import NETGENPlugin

MeshSfera = smesh.Mesh(Sfera)
NETGEN_2D3D = MeshSfera.Tetrahedron(algo=smesh.FULL_NETGEN)
NETGEN_3D_Parameters = NETGEN_2D3D.Parameters()
NETGEN_3D_Parameters.SetMaxSize(0.01)
NETGEN_3D_Parameters.SetSecondOrder(0)
NETGEN_3D_Parameters.SetOptimize(1)
NETGEN_3D_Parameters.SetFineness(4)
NETGEN_3D_Parameters.SetLocalSizeOnShape(Vertex_1, 0.005)
isDone = MeshSfera.Compute()

Guscio_1 = MeshSfera.GroupOnGeom(Guscio, 'Guscio', SMESH.FACE)
Sfera_1 = MeshSfera.GroupOnGeom(Sfera, 'Sfera', SMESH.VOLUME)

MeshCilindro = smesh.Mesh(Cilindro)
NETGEN_2D3D_1 = MeshCilindro.Tetrahedron(algo=smesh.FULL_NETGEN)
NETGEN_3D_Parameters_1 = NETGEN_2D3D_1.Parameters()
NETGEN_3D_Parameters_1.SetMaxSize(0.01)
NETGEN_3D_Parameters_1.SetSecondOrder(0)
NETGEN_3D_Parameters_1.SetOptimize(1)
NETGEN_3D_Parameters_1.SetFineness(4)
isDone = MeshCilindro.Compute()

BaseN = MeshCilindro.GroupOnGeom(Base, 'BaseN', SMESH.NODE)
```

```
PianoN = MeshCilindro.GroupOnGeom(Piano, 'PianoN', SMESH.NODE)
Base_1 = MeshCilindro.GroupOnGeom(Base, 'Base', SMESH.FACE)
Piano_1 = MeshCilindro.GroupOnGeom(Piano, 'Piano', SMESH.FACE)
Cilindro_1 = MeshCilindro.GroupOnGeom(Cilindro, 'Cilindro',
             SMESH.VOLUME)

Mesh_1 = smesh.Concatenate([MeshSfera.GetMesh(),
         MeshCilindro.GetMesh()], 1, 0, 1e-05)
[Guscio_2, Sfera_2, BaseN_1, PianoN_1, Base_2, Piano_2,
         Cilindro_2] = Mesh_1.GetGroups()

## set object names
smesh.SetName(MeshSfera.GetMesh(), 'MeshSfera')
smesh.SetName(NETGEN_2D3D.GetAlgorithm(), 'NETGEN_2D3D')
smesh.SetName(NETGEN_3D_Parameters, 'NETGEN 3D Parameters')
smesh.SetName(Guscio_1, 'Guscio')
smesh.SetName(Sfera_1, 'Sfera')

smesh.SetName(MeshCilindro.GetMesh(), 'MeshCilindro')
smesh.SetName(NETGEN_3D_Parameters_1, 'NETGEN 3D Parameters')
smesh.SetName(BaseN, 'BaseN')
smesh.SetName(PianoN, 'PianoN')
smesh.SetName(Base_1, 'Base')
smesh.SetName(Piano_1, 'Piano')
smesh.SetName(Cilindro_1, 'Cilindro')

smesh.SetName(Mesh_1.GetMesh(), 'Mesh_1')
smesh.SetName(Guscio_2, 'Guscio')
smesh.SetName(Sfera_2, 'Sfera')
smesh.SetName(BaseN_1, 'BaseN')
smesh.SetName(PianoN_1, 'PianoN')
smesh.SetName(Base_2, 'Base')
smesh.SetName(Piano_2, 'Piano')
smesh.SetName(Cilindro_2, 'Cilindro')
```

After meshing, it were identified:

- the four nodes (N_1, N_2, N_, N_4) belonging to the tetrahedron localised in the centre of the sphere;
- the node (N_5) localised in the point of the impact between sphere and plate.

Figure 4.24 reports the geometric model after meshing, when the sphere diameter was 80 mm.

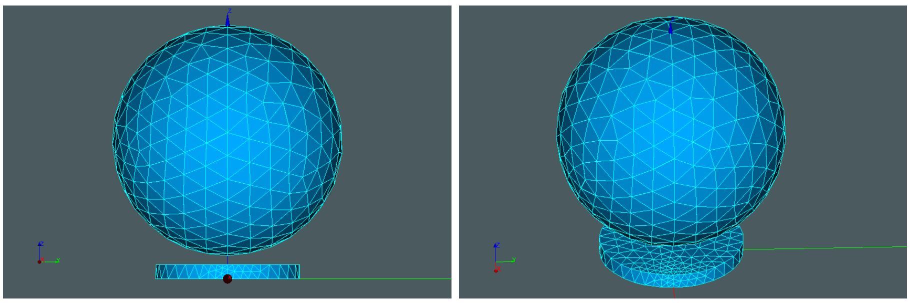


Figure 4.24: *Geometric model used for drop tests after meshing.*

The cylindrical plate was characterised by 252 nodes, 74 edges, 478 faces and 672 volumes. The values for the sphere, when the diameter ranged from 50 mm to 80 mm, were 334–765 nodes, 14–15 edges, 570–600 faces and 1357–3474 volumes.

In Code-Aster command file the following directives were used:

- `CREA_CHAMP` to assign the initial velocity (`pre_velo`) to the sphere;
- `CALC_NO` to calculate the nodal forces (`REAC_NODA`)
- `POST_RELEVE_T` to extract the acceleration of the selected nodes and the impact force transmitted to the steel plate. Values are provided by the software in tabular form (`IMPR_TABLE`).

Below the command file.

```
DEBUT();
# Read MED MESH File
MeshLin=LIRE_MAILLAGE(UNITE=20,
                      FORMAT='MED',
                      NOM_MED='Mesh_1',
                      INFO_MED=1,);

# Assigns a physical model to geometric entities.
FEMLin=AFFE_MODELE(MAILLAGE=MeshLin,
                   AFFE=_F(TOUT='OUI',
                          PHENOMENE='MECANIQUE',
                          MODELISATION='3D',),);

# Material properties
Steel=DEFI_MATERIAU(ELAS=_F(E=2.1e11,
                           NU=0.27,
                           RHO=7800.0,),);

# Material parameters
E_B = 5e5;
NU_B = 0.49;
RHO_B = 900;
UN_SUR_K_B = ((3*(1 - (2*NU_B)))/E_B);

Ball=DEFI_MATERIAU(ELAS=_F(E=E_B,
                          NU=NU_B,
                          RHO=RHO_B,),
                   LEMAITRE=_F(N=1,
                              UN_SUR_K=UN_SUR_K_B,),);

# Assign Material properties to Elements
Mat=AFFE_MATERIAU(MAILLAGE=MeshLin,
                  MODELE=FEMLin,
                  AFFE=(_F(GROUP_MA='Cilindro',
                           MATER=Steel,),
                        _F(GROUP_MA='Sfera',
                            MATER=Ball,),),);

# Boundary conditions
gravity=AFFE_CHAR_MECA(MODELE=FEMLin,
                       PESANTEUR=_F(GROUP_MA='Sfera',
                                    GRAVITE=9.81,
                                    DIRECTION=(0, 0, -1,),),);

BCnd=AFFE_CHAR_MECA(MODELE=FEMLin,
                    DDL_IMPO=_F(GROUP_MA='Base',
                                DX=0.0,
                                DY=0,
                                DZ=0,),);

# Initial velocity
v10 = -1.3795;
v15 = -1.6983;
v20 = -1.9659;
v25 = -2.2014;
v35 = -2.6092;
v50 = -3.1227;

pre_velo=CREA_CHAMP(TYPE_CHAM='NOEU_DEPL_R',
                    OPERATION='AFFE',
                    MAILLAGE=MeshLin,
                    AFFE=_F(GROUP_MA='Sfera',
                            NOM_CMP=('DX', 'DY', 'DZ',),
                            VALE=(0, 0, v10,),),);

contact=DEFI_CONTACT(MODELE=FEMLin,
                     FORMULATION='DISCRETE',
                     REAC_GEOM='AUTOMATIQUE',
                     ZONE=_F(APPARIEMENT='MAIT_ESCL',
                             GROUP_MA_MAIT='Piano',
                             GROUP_MA_ESCL='Guscio',
                             ALGO_CONT='LAGRANGIEN',),);

# Solution
timelst=DEFI_LIST_REEL(DEBUT=0,
                       INTERVALLE=_F(JUSQU_A=0.006,
                                     PAS=0.0001,),);

# Compute VonMises Stress / Strain
solu=DYNA_NON_LINE(MODELE=FEMLin,
                   CHAM_MATER=Mat,
                   EXCIT=(_F(CHARGE=gravity,),
                          _F(CHARGE=BCnd,),),
                   CONTACT=contact,
                   COMP_INCR=_F(RELATION='LEMAITRE',
                                DEFORMATION='GDEF_HYPO_ELAS',
                                GROUP_MA='Sfera',),
                   COMP_ELAS=_F(RELATION='ELAS',
                                DEFORMATION='PETIT',
                                RESI_INTE_RELA=1e-06,
                                GROUP_MA='Cilindro',),
                   ETAT_INIT=_F(VITE=pre_velo,),
                   INCREMENT=_F(LIST_INST=timelst,),
                   SCHEMA_TEMPS=_F(SCHEMA='HHT',
                                   FORMULATION='DEPLACEMENT',),
                   NEWTON=_F(REAC_INCR=1,
                             PREDICTION='ELASTIQUE',
                             MATRICE='TANGENTE',
                             REAC_ITER=1,
                              REAC_ITER_ELAS=1,),
                    CONVERGENCE=_F(ITER_GLOB_MAXI=100,),
                    ARCHIVAGE=_F(PAS_ARCH=1,
                                 PRECISION=1e-06,),);

solu=CALC_ELEM(reuse=solu,
               MODELE=FEMLin,
               RESULTAT=solu,
               OPTION=('SIEF_ELNO', 'EPSI_ELNO',
                       'EPSI_ELGA', 'SIEQ_ELNO',),);

solu=CALC_NO(reuse=solu,
             RESULTAT=solu,
             OPTION=('REAC_NODA', 'SIEQ_NOEU',
                     'SIEF_NOEU', 'FORC_NODA',),);

# Write Results to MED file
Forza=POST_RELEVE_T(ACTION=(_F(OPERATION='EXTRACTION',
                               INTITULE='ForzeNod',
                               RESULTAT=solu,
                               NOM_CHAM='REAC_NODA',
                               GROUP_NO='PianoN',
                               RESULTANTE='DZ',),
                            _F(OPERATION=('EXTRACTION',),
                               INTITULE='Accelerazione',
                               RESULTAT=solu,
                               NOM_CHAM='ACCE',
                               GROUP_NO='N_1',
                               RESULTANTE=('DX', 'DY', 'DZ',),),
                            _F(OPERATION=('EXTRACTION',),
                               INTITULE='Accelerazione',
                               RESULTAT=solu,
                               NOM_CHAM='ACCE',
                                GROUP_NO='N_2',
                                RESULTANTE=('DX', 'DY', 'DZ',),),
                             _F(OPERATION=('EXTRACTION',),
                                INTITULE='Accelerazione',
                                RESULTAT=solu,
                                NOM_CHAM='ACCE',
                                GROUP_NO='N_3',
                                RESULTANTE=('DX', 'DY', 'DZ',),),
                             _F(OPERATION=('EXTRACTION',),
                                INTITULE='Accelerazione',
                                RESULTAT=solu,
                                NOM_CHAM='ACCE',
                                GROUP_NO='N_4',
                                RESULTANTE=('DX', 'DY', 'DZ',),),
                             _F(OPERATION=('EXTRACTION',),
                                INTITULE='Accelerazione',
                                RESULTAT=solu,
                                NOM_CHAM='ACCE',
                                GROUP_NO='N_5',
                                RESULTANTE=('DX', 'DY',
                                            'DZ',),),),);

IMPR_TABLE(TABLE=Forza,);

IMPR_RESU(MODELE=FEMLin,
          FORMAT='MED',
          UNITE=80,
          RESU=_F(RESULTAT=solu,
                  TOUT_CHAM='OUI',
                  TOUT_CMP='OUI',),);

FIN(FORMAT_HDF='OUI',);
```

4.5.3 Effect of drop height

Figures 4.25, 4.26 and 4.27 report, for three sphere diameters, the maximum impact force trend vs the drop height at varying density and modulus of Young of the material. Trends were well explained by second order equations, with determination coefficients ranging from 0.996 to 1.000.

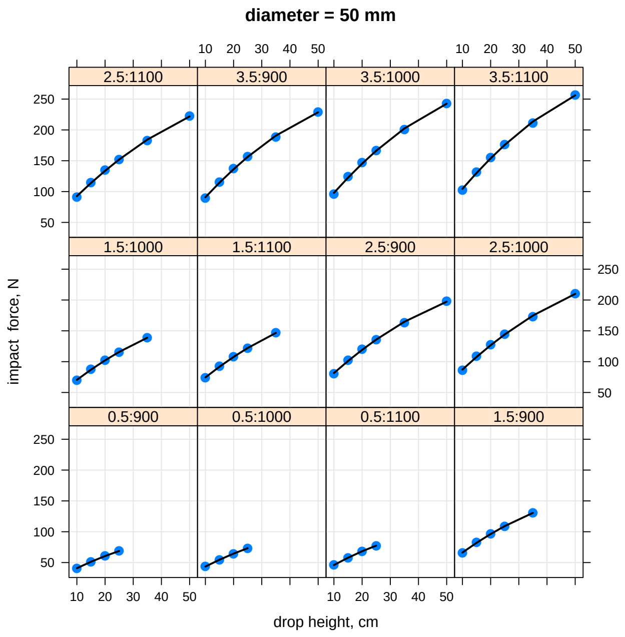


Figure 4.25: *Maximum impact force vs drop height at varying modulus of Young* (MPa) *and density* (kg/m^3) *when the sphere diameter was* 50 mm.

When the modulus of Young was 3.5 MPa, the sphere diameter 50 mm and the sphere density 1100 kg/m^3, the impact force increased from 103 N to 257 N when the drop height increased from 10 cm to 50 cm. With the sphere diameter of 80 mm, the impact force increased from 266 N to 442 N when the drop height increased fro 10 cm to 25 cm.

Geyer *et al.* (2009) reports that, when dropping potato tubers with mass in the range 100–210 g from 10 cm and 25 cm on steel plate, the maximum impact force was on average about 140 and 250 N respectively. These simulations, in similar conditions (mass = 102–199 g, modulus of Young = 2.5–3.5 MPa, density = 900–1100 kg/m^3), provided maximum impact forces of 154 N and

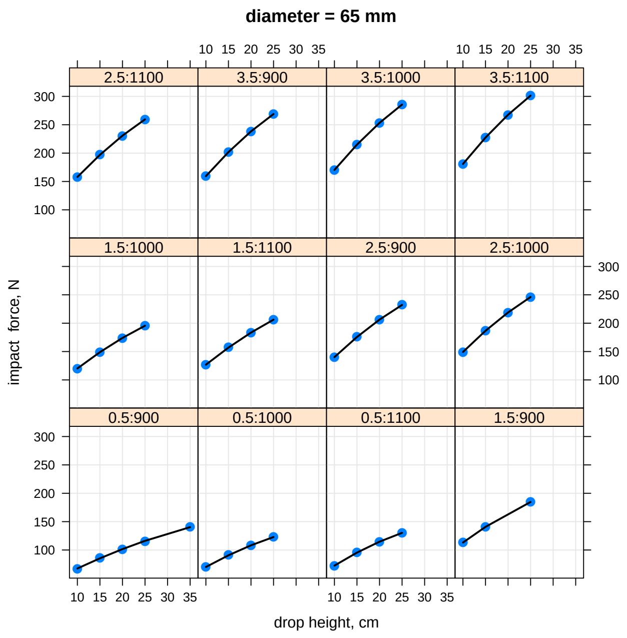


Figure 4.26: *Maximum impact force vs drop height at varying modulus of Young* (MPa) *and density* (kg/m^3) *when the sphere diameter was* 65 mm.

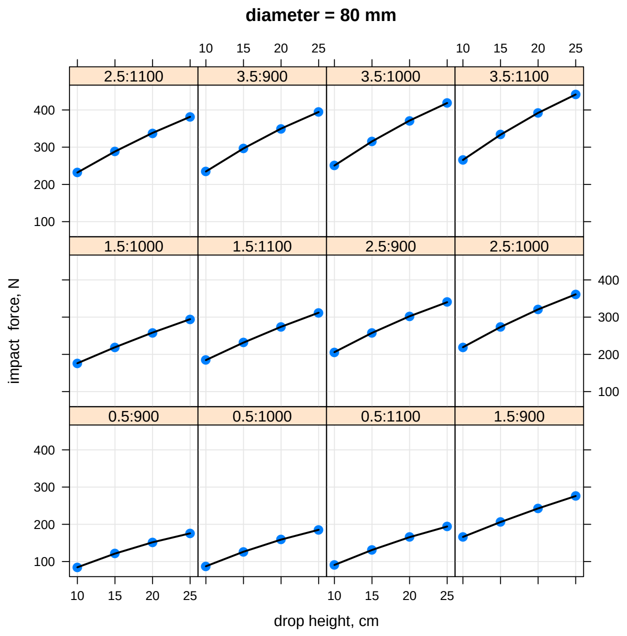


Figure 4.27: *Maximum impact force vs drop height at varying modulus of Young* (MPa) *and density* (kg/m^3) *when the sphere diameter was* 80 mm.

271 N, in good agreement with the experimental results.

Similar trends were showed by the maximum acceleration at the centre of the sphere (Figure 4.28, 4.29 and 4.30), with determination coefficients ranging from 0.976 to 1.000.

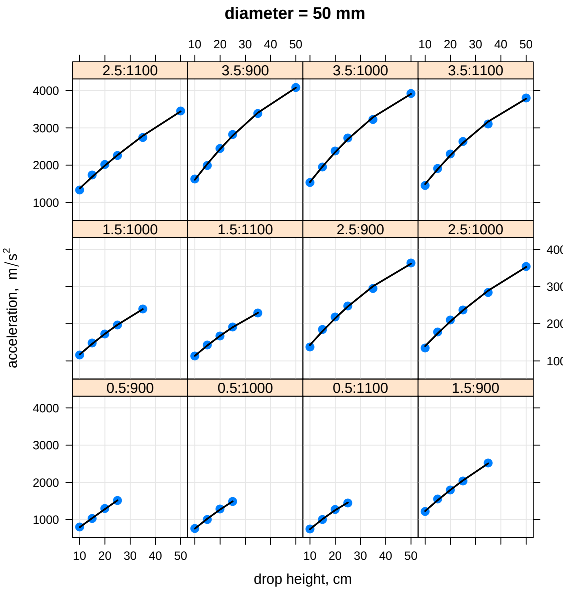


Figure 4.28: *Maximum acceleration at the centre of the sphere vs drop height at varying modulus of Young* (MPa) *and density* (kg/m^3) *when the sphere diameter was* 50 mm.

When the modulus of Young was 3.5 MPa, the sphere diameter 50 mm and the sphere density 1100 kg/m^3, the maximum acceleration at the centre of the sphere increased from 1455 m/s^2 to 3806 m/s^2 when the drop height increased from 10 cm to 50 cm. With the sphere diameter of 80 mm, the impact force increased from 928 m/s^2 to 1537 m/s^2 when the drop height increased fro 10 cm to 25 cm.

The cited Authors (Geyer *et al.*, 2009) report maximum acceleration values at the centre of the potatoes, measured with the AMU device in the aforementioned conditions, of around 50 g (490 m/s^2) for drop heights of 10 cm and of around 80 g (785 m/s^2) for drop heights of 25 cm. Simulations, instead, provided average values of acceleration about 2.4 times greater (1200 m/s^2

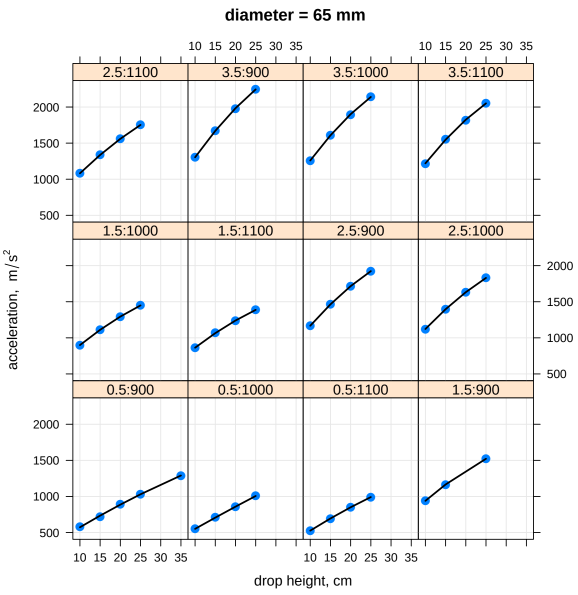


Figure 4.29: *Maximum acceleration at the centre of the sphere vs drop height at varying modulus of Young* (MPa) *and density* (kg/m^3) *when the sphere diameter was* 65 mm.

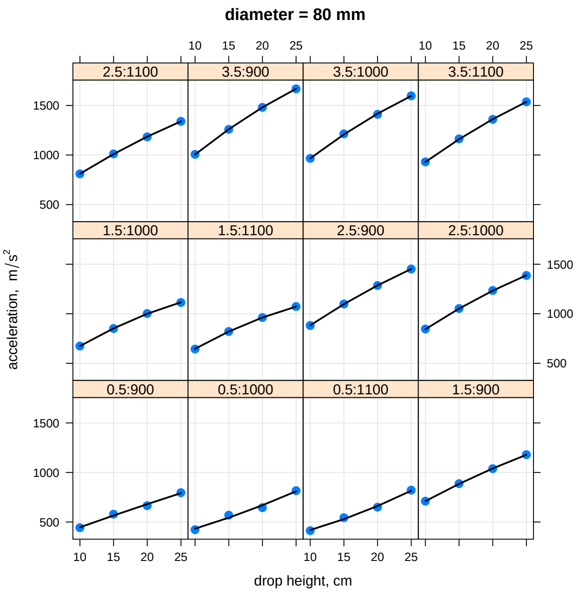


Figure 4.30: *Maximum acceleration at the centre of the sphere vs drop height at varying modulus of Young* (MPa) *and density* (kg/m^3) *when the sphere diameter was* 80 mm.

with drop height of 10 cm and 2000 m/s^2 with drop height of 25 cm). This difference could be due to the implantation system of the AMU device inside the tuber.

The acceleration in the impact zone was on average from 10 to 18 times higher that at the centre of the sphere, when the diameter increased from 50 to 80 mm.

4.5.4 Effect of modulus of Young

Figure 4.31 reports the average value of the impact force vs the modulus of Young. The trend was approximately linear, with coefficient of determination $R^2 = 0.97$, significant at p-level = 0.05. When the modulus of Young increased from 0.5 up to 3.5 MPa, the impact force increased from 101 up to 232 N.

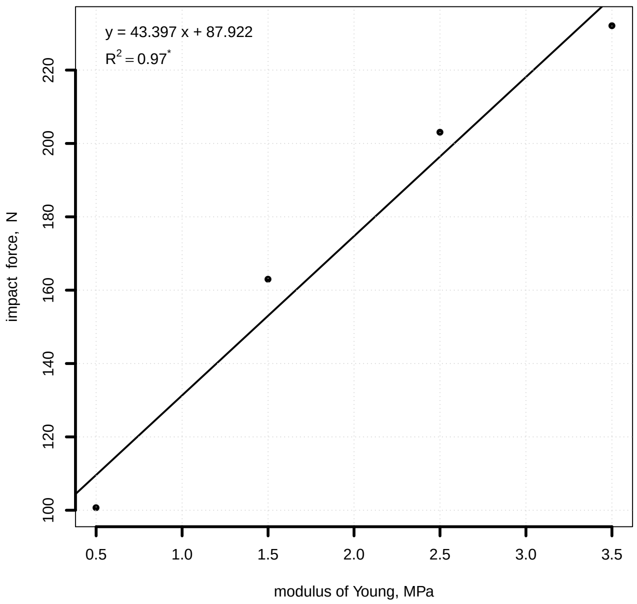


Figure 4.31: *Average impact force vs modulus of Young.*

The trends for some sphere diameters (50, 65 and 80 mm) are showed in figures 4.32, 4.33 and 4.34. They were all linear with coefficients of determination ranging from 0.916 up to 1.000. The increase in impact force per unit increase of modulus of Young $\left(\frac{\Delta F}{\Delta E}\right)$ ranged from 16 to 81 N/MPa.

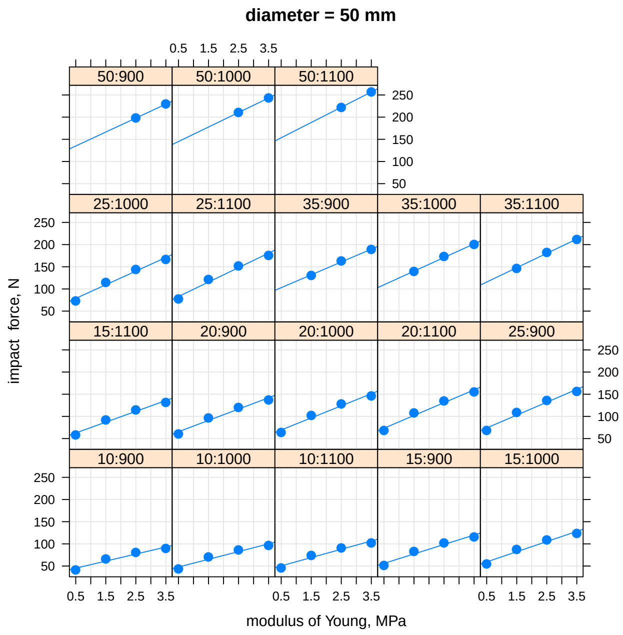


Figure 4.32: *Maximum impact force vs modulus of Young at varying drop height* (cm) *and density* (kg/m^3) *when the sphere diameter was* 50 mm.

The average acceleration at the centre of the sphere increased linearly with the modulus of Young (coefficient of determination equal to 0.98, significant at *p*-level = 0.01). When the modulus of Young increased from 0.5 to 3.5 MPa, the acceleration increased from 926 to 1940 m/s^2. The increase in acceleration (m/s^2) per unit increase in modulus of Young (MPa) was on average 341 (m/s^2)/MPa.

Figures 4.35, 4.36 and 4.37 report the single trends for some sphere diameters (50, 65 and 80 mm) at varying material density and drop height.

Finally, the acceleration of the node in the impact area was not affected by the Young modulus: it was on average 23 550 m/s^2, from 12 to 26 times that at the centre of the sphere.

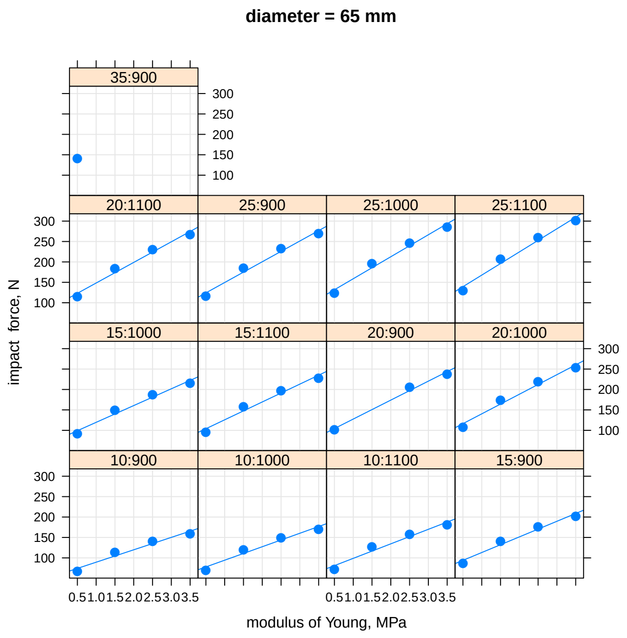


Figure 4.33: *Maximum impact force vs modulus of Young at varying drop height (cm) and density* (kg/m^3) *when the sphere diameter was* 65 mm.

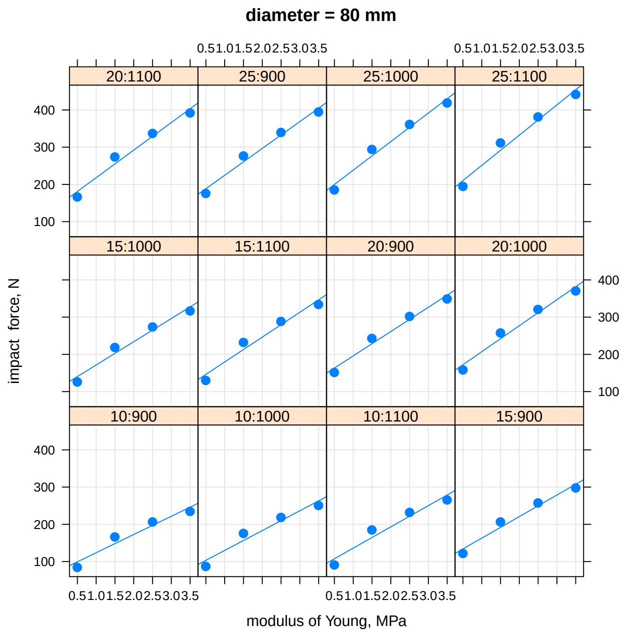


Figure 4.34: *Maximum impact force vs modulus of Young at varying drop height (cm) and density* (kg/m^3) *when the sphere diameter was* 80 mm.

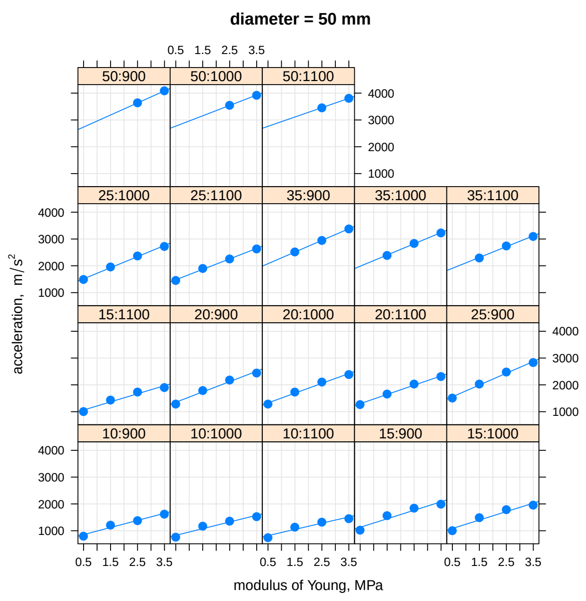


Figure 4.35: *Maximum acceleration at the centre of the sphere vs modulus of Young at varying drop height* (cm) *and density* (kg/m^3) *when the sphere diameter was* 50 mm.

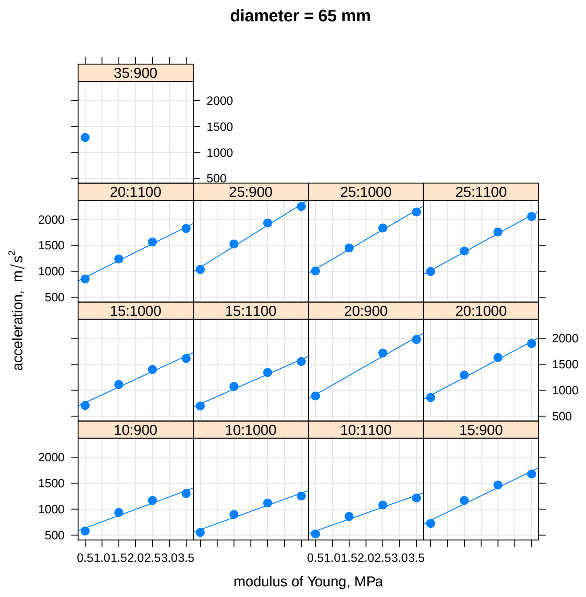


Figure 4.36: *Maximum acceleration at the centre of the sphere vs modulus of Young at varying drop height* (cm) *and density* (kg/m^3) *when the sphere diameter was* 65 mm.

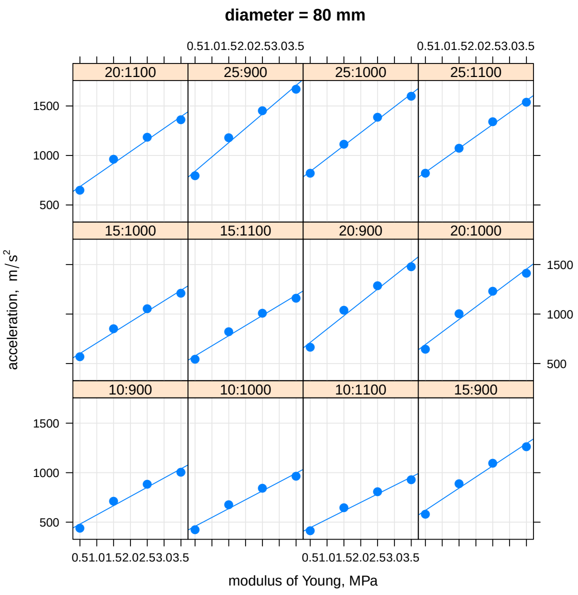


Figure 4.37: *Maximum acceleration at the centre of the sphere vs modulus of Young at varying drop height* (cm) *and density* (kg/m^3) *when the sphere diameter was* 80 mm.

4.5.5 Effect of diameter

The effect of the sphere diameter on the impact force is showed in Figures 4.38 and 4.39 at varying drop height and material density, for two values of modulus of Young (0.5 and 3.5 MPa).

The trends were linearly increasing, with coefficients of determination ranging from 0.946 to 1.000. Slopes increased at increasing drop height, modulus of Young and density from 1.453 N/mm (drop height = 10 cm, density = 1000 kg/m^3, modulus of Young = 0.5 MPa) to 11.050 N/mm (drop height = 50 cm, density = 1100 kg/m^3, modulus of Young = 3.5 MPa). As an example, when the drop height was 25 cm and the density 1100 kg/m^3, the impact force increased from 77 N (diameter of 50 mm) to 194 N (diameter of 80 mm) when the modulus of Young was 0.5 MPa and from 176 N to 442 N when it was 3.5 MPa.

The maximum acceleration values at the centre of the sphere are shown in Figures 4.40 and 4.41 for the same two values of modulus of Young (0.5 and 3.5 MPa).

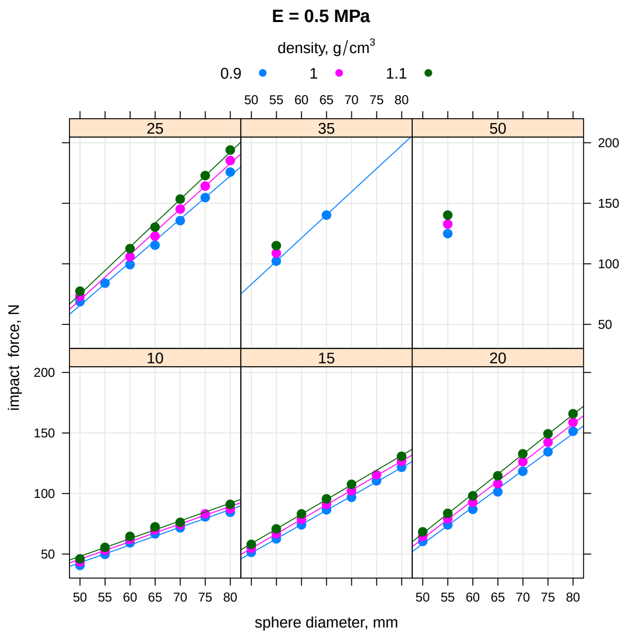


Figure 4.38: *Maximum impact force vs sphere diameter at varying drop height* (cm) *and density when the modulus of Young was* 0.5 MPa.

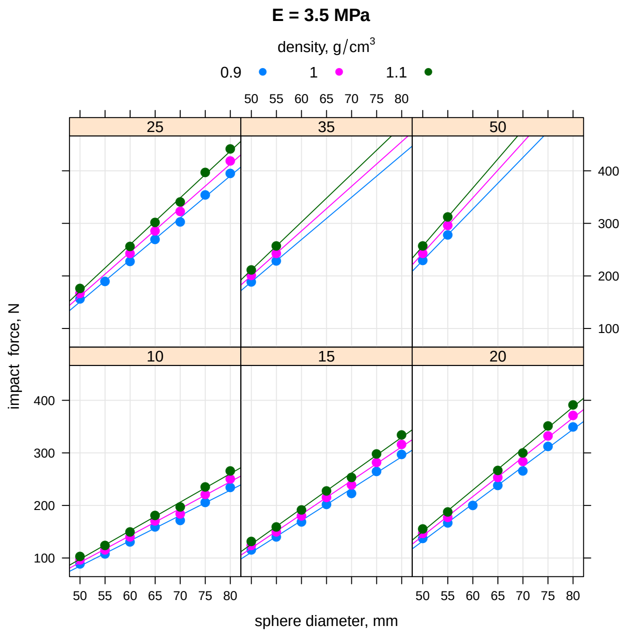


Figure 4.39: *Maximum impact force vs sphere diameter at varying drop height* (cm) *and density when the modulus of Young was* 3.5 MPa.

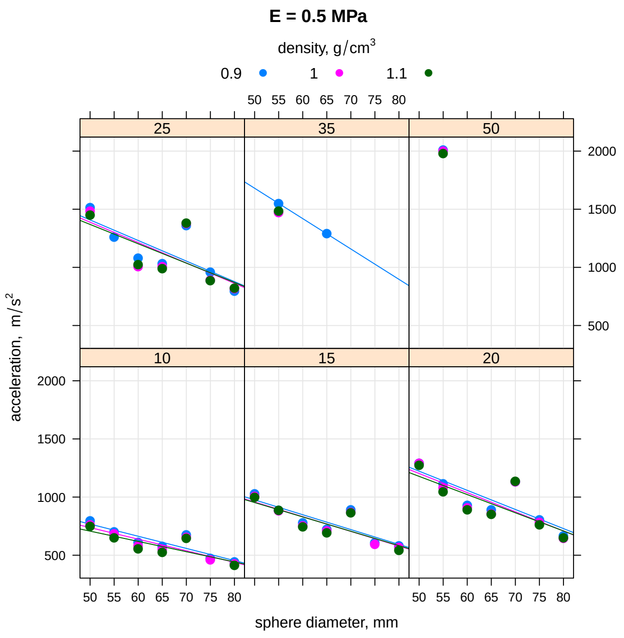


Figure 4.40: *Maximum acceleration at the centre of the sphere vs sphere diameter at varying drop height* (cm) *and density when the modulus of Young was* 0.5 MPa.

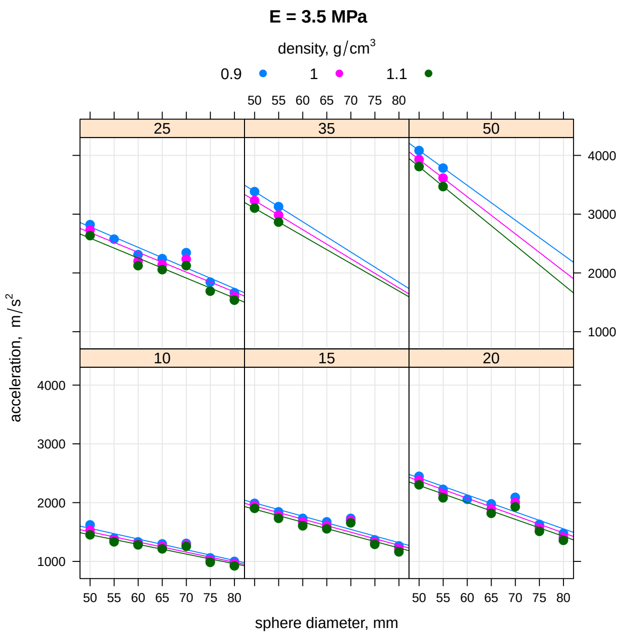


Figure 4.41: *Maximum acceleration at the centre of the sphere vs sphere diameter at varying drop height* (cm) *and density when the modulus of Young was* 3.5 MPa.

They showed a linear decreasing trend, with negligible differences due to the material density. The decreasing trend is due to the cushioning effect produced by the sphere material itself: when the size of the sphere increases, there is a greater distance between impact zone and centre of the sphere, so the acceleration measured at the centre of the sphere is reduced.

4.5.6 Effect of density

The effect of the material density on the impact force is showed in Figure 4.42 when the modulus of Young was 3.5 MPa. Similar graphs were obtained for the other values of the modulus of Young. The trends were linear: increasing material density, maximum impact force increased proportionally. The effect was more pronounced at higher sphere diameters and higher drop heights.

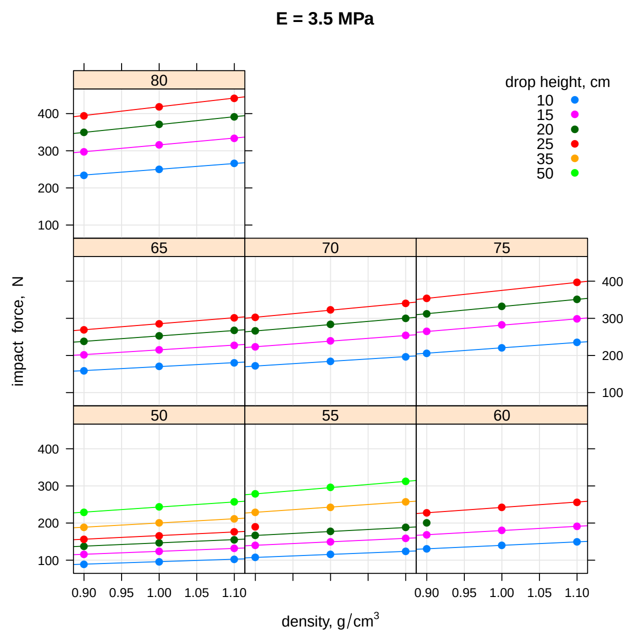


Figure 4.42: *Maximum impact force vs material density at varying drop height and sphere diameter* (mm) *when the modulus of Young was* 3.5 MPa*.*

The trend of the acceleration at the centre of the sphere was linear too (Figure 4.43, when the modulus of Young was 3.5 MPa), but in decreasing direction.

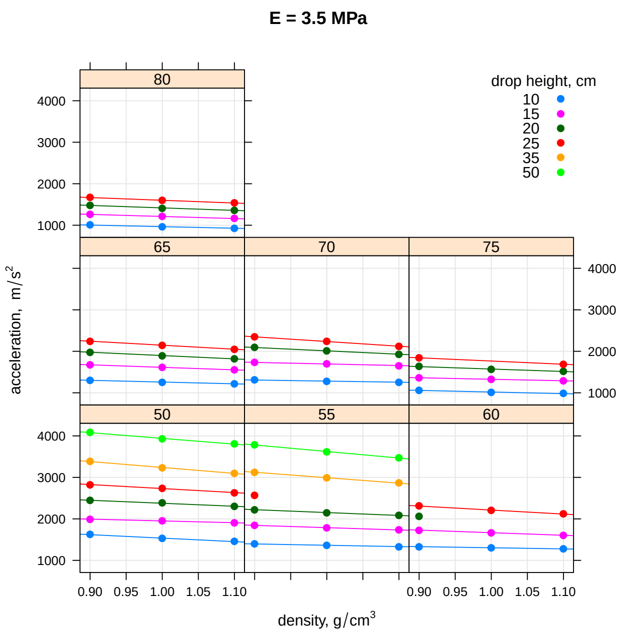


Figure 4.43: *Maximum acceleration at the centre of the sphere vs material density at varying drop height and sphere diameter* (mm) *when the modulus of Young was* 3.5 MPa.

Again, this results is due to the cushioning effect of the material, greater when the density was greater, keeping constant all the other parameters.

4.5.7 Effect of mass

Finally, size (diameter) and density were combined in one single parameter, the mass of the sphere. With the given values of diameter and density, mass values ranged from 59 to 295 g.

Figures 4.44 and 4.45 report the impact force values at varying the mass of the sphere, assuming as parameters the modulus of Young or the drop height.

Geyer *et al.* (2009) report that, when dropping potato tubers with mass of 100–120 g from 25 cm onto steel plates, the impact force ranged from 190 to 220 N (depending on the potato variety). Simulations showed that the impact force in the same conditions (mass of 102–113 g and modulus of

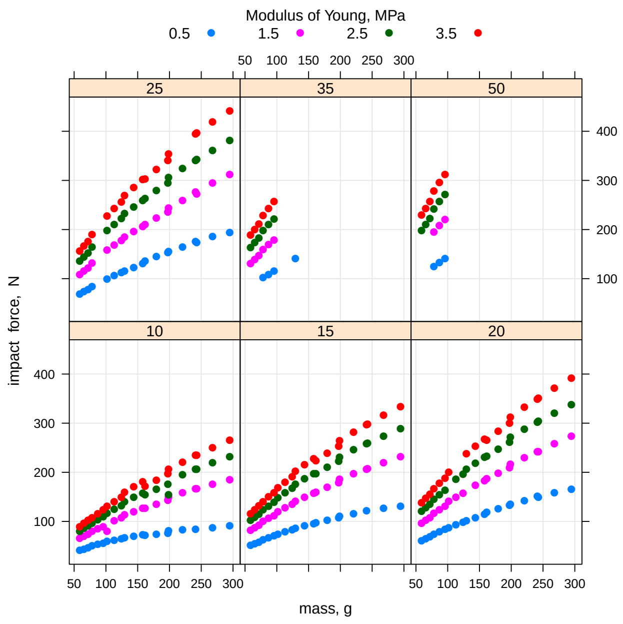


Figure 4.44: *Maximum impact force vs mass at varying drop height* (cm)*, assuming as parameter the modulus of Young.*

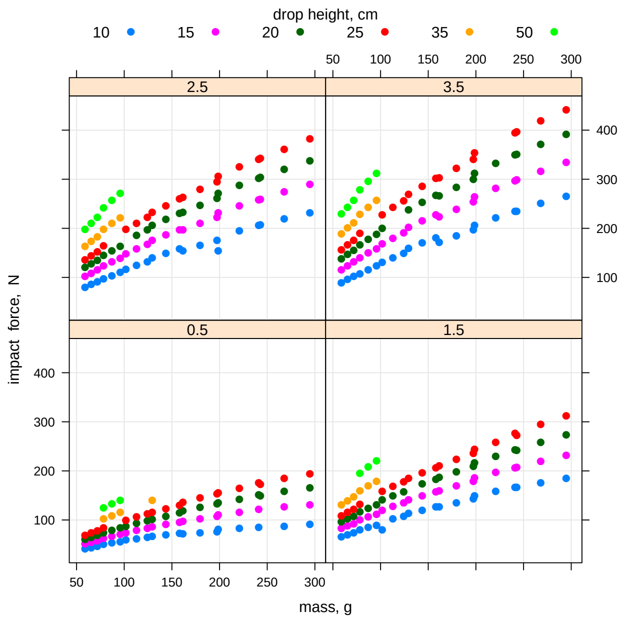


Figure 4.45: *Maximum impact force vs mass at varying modulus of Young* (MPa)*, assuming as parameter the drop height.*

Young of 2.5–3.5 MPa) ranged from 198 to 242 N, in good agreement with the experimental results. Again, when the mass tuber was 190–210 g, the measured impact force was 310–325 N. The simulations showed, for mass of 199–221 g, impact forces ranging from 306 to 325 N, again in good agreement with the experimental results.

Finally, Figures 4.46 and 4.47 report the acceleration values at the centre of the sphere at varying the mass of the sphere and assuming as parameters the modulus of Young or the drop height.

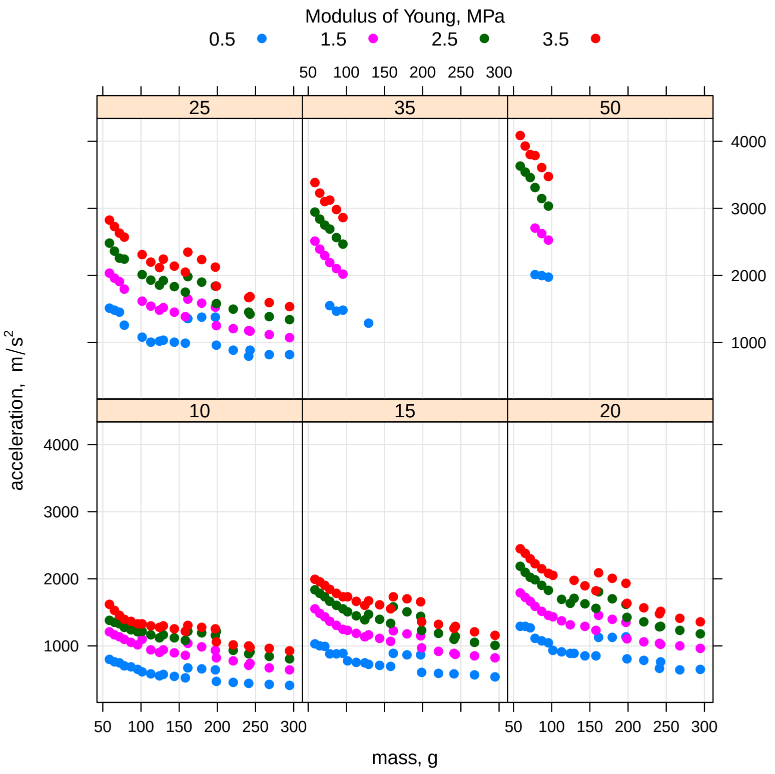


Figure 4.46: *Maximum acceleration at the centre of the sphere vs mass at varying drop height* (cm)*, assuming as parameter the modulus of Young.*

Both graphs show a decreasing trend in acceleration values, due to the cushioning effect of the material.

Geyer *et al.* (2009) report that, in the aforementioned conditions, the acceleration measured by the AMU device ranged from 94 to 95 *g* (922–

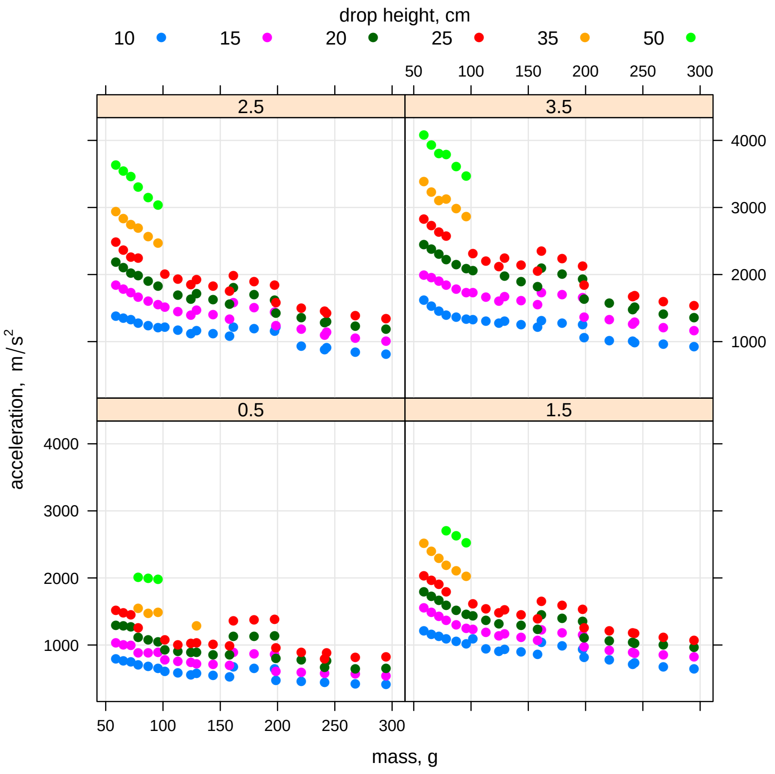


Figure 4.47: *Maximum acceleration at the centre of the sphere vs mass at varying modulus of Young* (MPa)*, assuming as parameter the drop height.*

932 m/s^2) for masses of 100–120 g and from 78 to 84 *g* (765–824 m/s^2) for masses of 190–210 g. Simulations, in similar conditions, provided acceleration values ranging from 1934 to 2314 m/s^2 for masses of 102–113 g and moduli of Young of 2.5–3.5 MPa and ranging from 1497 to 1843 m/s^2 for masses of 199–221 g.

Simulated acceleration values were therefore about twice as high as those measured. On the other hand, Geyer *et al.* (2009) report that, with the measured values of acceleration, computed impact forces:

$$F_{computed} = mass \times acceleration \tag{4.15}$$

were approximately half the measured ones, meaning an under-estimation of the acceleration with the AMU device, perhaps due to the implantation system of the AMU inside the tuber.

On the other hand, based on simulation tests, forces computed according Equation 4.15 and those provided by simulations were in good agreement (Figure 4.48).

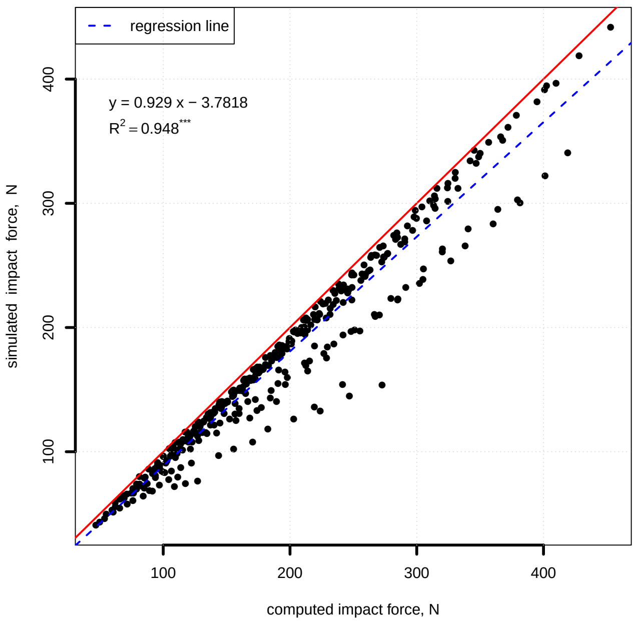


Figure 4.48: *Correlation between simulated and computed force.*

The regression equation provided a determination coefficient of 0.95, statistically significant at p-level of 0.001, and a slope equal to 0.923 N/N, very near to the theoretical value of 1.

Conclusions

Impact forces and accelerations arising from collisions, are among the main indices taken into account when studying the damage of fruit and vegetables during post-harvest activities. The probability of damage is usually assessed by dropping fruits in known conditions and evaluating the percentage of damaged fruits.

Several “artificial sensorised fruits”, able to measure forces or accelerations arising from impacts, have been designed and commercialised. A miniaturised Acceleration Measuring Unit (AMU) has been recently developed at the Institut für Agrartechnik, Potsdam-Bornim (ATB). When implanted into a real product like a potato tuber, it is able to measure the accelerations at the centre of the fruit deriving from a impact.

In this PhD Thesis it was simulated, by means of the Finite Element Analysis approach, the behaviour of spherical artificial fruits when dropped onto steel plates. Several simulations of drop tests of the artificial fruit at varying density and modulus of elasticity of the material, diameter of the fruit and drop height, were developed to know information about the impact parameters (maximum impact force and maximum acceleration at the centre of the sphere). Material properties were chosen to approach those of potato tubers.

Even if further studies are necessary to improve the model, the main results of the simulations allows the following conclusions:

- The maximum impact forces vs the drop height, when fixing density and modulus of Young of the material and diameter of the sphere, showed an increasing trend well explained by second order equations. When the diameter of the sphere was 55 mm, the average impact force increased from about 90 N to 220 N when the drop height increased from 10 cm to 50 cm. The experimental results by Geyer *et al.* (2009), referring to drop tests of potato tubers with mass in the range 100–210 g from 10 cm and 25 cm on steel plate, reported maximum impact force values of about 140 and 250 N respectively. The simulations, in similar conditions, provided maximum impact force values of 154 N and 271 N, in good agreement with the experimental results.

- Similar trends were showed by the maximum acceleration at the centre of the sphere. When the diameter of the sphere was 55 mm, the acceleration increased from about 1085 m/s^2 to 2850 m/s^2 when the drop height increased from 10 cm to 50 cm. Geyer *et al.* (2009), when dropping potato tubers in the aforementioned conditions, reported maximum acceleration values at the centre of the potatoes, measured with the AMU device, of around 490 m/s^2 for drop heights of 10 cm and of around 785 m/s^2 for drop heights of 25 cm. Simulations, instead, provided average values of acceleration about 2.4 times greater. This difference could be due to the implantation system of the AMU device inside the tuber.

- The average value of impact force vs modulus of Young showed a liner trend with coefficient of determination $R^2 = 0.97$. Averaging all the results, impact force increased from about 100 to 230 N when the modulus of Young increased from 0.5 to 3.5 MPa. Similarly, the average acceleration at the centre of the sphere increased linearly with the modulus of Young: when the modulus of Young increased from 0.5 to 3.5 MPa, the acceleration increased from 926 to 1940 m/s^2.

- The sphere diameter had a linear effect on the impact force: fixing drop height, modulus of Young and material density, an increase in the sphere diameter caused a linear increase in the maximum impact force. The maximum acceleration at the centre of the sphere vs sphere diameter, instead, reported a linear decreasing trend, with negligible differences due to the material density. The decreasing trend is due to the cushioning effect produced by the sphere material itself: when the size of the sphere increases, there is a greater distance between impact zone and centre of the sphere, so the acceleration measured at the centre of the sphere is reduced.

- The trends of the maximum impact force vs material density were linear: increasing material density, maximum impact force increased proportionally. The effect was more pronounced at higher sphere diameters and higher drop heights. The trends of the acceleration at the centre of the sphere vs material density were linear too, but in decreasing direction. Again, this results is due to the cushioning effect of the material, greater when the density was greater, keeping constant all the other parameters.

- Combining size and density of the sphere in one single parameter, the mass of the sphere, the simulations showed an increasing trend of the maximum impact force vs the mass. Considering masses ranging from 200 to 220 g, simulations provided impact forces ranging from 306 to 325 N when the drop height was 25 cm. Experimental tests in the same conditions, reported by Geyer *et al.* (2009), showed impact force values of 310–325 N, in good agreement with the simulations. The acceleration values at the centre of the sphere at varying the mass of the sphere and assuming as parameters the modulus of Young or the drop height, reported a decreasing trend, due to the cushioning effect of the material. The simulations provided acceleration values about twice as many those measured in the experimental results.

- Considering the force F_c computed accordingly to the second law of Newton ($F_c = m \cdot a$) and comparing F_c with the impact force F_i, the comparison gave good agreement in simulations, whereas in experimental tests F_c was approximately half the measured force F_i, meaning an under-estimation of the acceleration with the AMU device, perhaps due to the implantations system of the AMU inside the tuber.

All considered, the concordance between experimental and simulated tests, confirmed the validation of the FEM approach, although the limitations owing to the simplicity of the model developed in this work. Further development of the work could take into account:

- implementation of other simulation by FEM approach with more complex models that represent better the specific material of vegetables;
- development of other simulation on other types of vegetables to validate the method of analysis.

References

1. Archer J.S. (1965), *Consistent mass matrix formulations for structural analysis using finite element techniques*, Journal of the American Institute of Aeronautics and Astronautics 3, no. 10, 1965.

2. ASAE Standards (1999), *S368.3: Compression test of food materials of convex shape*, St. Joseph, Mich.: ASAE.

3. Baheri M. (1997), *Development of a method for prediction of potato mechanical damage in the chain of mechanized potato production*, PhD Thesis, University of Leuven, Belgium, 301.

4. Bentini M., Caprara C., Martelli R. (2006), *Harvesting damage to potato tubers by analysis of impacts recorded with an instrumented sphere*, Biosystems Eng. 94(1):75–85.

5. Blandini G., Cerruto E., Manetto G., (2002), *Indagine sul danneggiamento meccanico delle arance nelle linee di lavorazione*, Rivista di Ingegneria Agraria, Anno XXXIII, n. 2, June 2002, 1–12.

6. Blandini G., Cerruto E., Manetto G., Romano E. (2003), *Mechanical damage to oranges caused by repeated impact against steel*, proceedings of XXX CIOSTA–CIGR V "Management and Technology Applications to Empower Agriculture and Agro-Food Systems", Grugliasco (TO), September 22–24, 2003, vol. 1, 400–407.

7. Bollen A.F., Cox R.N., Delarue B.T, Painter B.J. (2001), *A descriptor for damage susceptibility of a population of produce*, J. Agric. Eng. Res. 78(4):391–395.

8. Bollen A.F. (2006), *Technological innovations in sensors for assessment of postharvest mechanical handling systems*, Intl. J. Postharvest Tech. and Innovation 1(1):16–31.

9. Bouman A. (1995): Beschädigungen der Kartoffeln vom Roden bis zum Abpacken, Kartoffelbau 46 (6): 242–244.

10. Burton W.G. (1989), *Post-harvest physiology*, The Potato, 3rd Edition, Longman Scientific and Technical, London, 423–431.

11. Clough R.W. (1960), *The finite element method in plane stress analysis*, Proceedings American Society of Civil Engineers, Second Conference on Electronic Computation, Pittsburgh, 1960.

12. Colombo (1985), *Nuovo Colombo - Manuale dell'Ingegnere*, 81-esima edizione, HOEPLI.

13. Cook R.D. (1995), *Finite element modeling for stress analysis*, John Wiley & Sons, Inc. 1995.

14. Courant R. (1943), *Variational methods for the solution of problems of equilibrium and vibrations*, Bulletin of the American Mathematical Society, 49, 1943.

15. Galerkin B.G. (1915), *Series solution of some problems of elastic equilibrium of rods and plates* (in Russian), Vestn. Inzh. Tekh. 19, 1915.

16. Geyer M. O., Praeger U., König C., Graf A., Truppel I., Schlüter O., Herold B. (2009), *Measuring behaviour of an acceleration measuring unit implanted in potatoes*, Transactions of the ASABE, Vol. 52(4):1267–1274.

17. Gray D., Hughes J.C. (1978), *Tuber quality*, P.M. Harris (Editor), The Potato Crop., The Scientific Basis for Improvement, Chapman and Hall, London, 504–544.

18. Herold H., Truppel I., Siering G., Geyer M. (1996), *A pressure measuring sphere for monitoring handling of fruit and vegetables*, Computer and Electronics in Agriculture, 15 (1996):73–88.

19. Herold B., Geyer M., Studman C.J., (2001), *Fruit contact pressure distributions — equipment*, Computers and Electronics in Agriculture, 32 (2001) 167–179.

20. Hiller L.K., Keller D.C., Thornton R.E. (1985), *Physiological disorders of potato tubers*, P.H. Li (Editor), Potato Physiology, Academic Press, Orlando, FL, 389–455.

21. Hutton D.V. (2004), *Fundamentals of finite element analysis – 1st ed.*, The MacGraw-Hill Companies, 2004.

22. Hyde G.M., Brown G.K., Timm E.J., Zhang W. (1992), *Instrumented sphere evaluation of potato packing line impacts*, Transactions of the ASAE, vol. 35(1):65–69.

23. Ito M., Sakai K., Hata S., Takai M. (1994), *Damage to the surface of potatoes from collision*, Transactions of the ASAE, 37(5):1431–1433.

24. Khodabakhshian R., Emadi B. (2011), *Determination of the modulus of elasticity in agricultural seeds on the basis of elasticity theory*, Middle-East Journal of Scientific Research, 7(3):367–373.

25. Leicher J. (1992), *Mechanical damage to potatoes: a shared responsibility*, Report N.D.A.L.T.P., Belgium, 7.

26. Lewis R., Yoxall A., Marshall M.B., Canty L.A. (2007), *Characterising pressure and bruising in apple fruit*, Wear, Volume 264 (1-2), 37–46.

27. Liu G.R., Quek S.S. (2003), *The finite element method, a practical course*, Butterworth-Heinemann, 2003.

28. Maly P., Hoffmann T., Fürll Ch. (2005), Gentle harvest of potatoes in storage boxes, Agricultural Engineering International: the CIGR Ejournal, Manuscript FP 05 002, vol. VII, October, 2005.

29. Martin H.C. (1968), *Finite element analysis of fluid flows*, Proceedings of the Second Conference on Matrix Methods in Structural Mechanics, Wright-Patterson Air Force Base, Kilborn, Ohio, October 1968.

30. McGarry A., Hole C.C., Drew R.L.K., Parsons N. (1995), *Internal damage in potato tubers: a critical review*, Horticulture Research International, Wellesbourne, 24.

31. McGarry A., Hole C.C., Drew R.L.K., Parsons N. (1996), *Internal damage in potato tubers: a critical review*, Postharvest Biology and Technology 8, 239–258.

32. Melosh R.J. (1961), *A stiffness method for the analysis of thin plates in bending*, Journal of Aerospace Sciences 28, no. 1 (1961).

33. Melosh R.J. (1963), *Structural analysis of solids*, Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, August 1963.

34. Molema G.J. (1999), *Mechanical force and subcutaneous tissue discolouration in potato*, PhD Thesis, Wageningen, The Netherlands, Wageningen University.

35. Nerinckx G., Verschoore R. (1993), *Use of the electronic potato as an indicator for the occurrence of mechanical damage to potatoes*, PhD Thesis, University of Ghent, Belgium, 133.

36. Noor A.K. (1991), *Bibliography of books and monographs on finite element technology*, Applied Mechanics Reviews, 44, no. 6, 1991.

37. Opara U.L., Pathare P.B. (2014), *Bruise damage measurement and analysis of fresh horticultural produce—A review*, Postharvest Biology and Technology, 91 (2014) 9–24.

38. Peters R. (1996), *Damage of potato tubers: A review*, Potato Research 39(4):479–484.

39. Przemieniecki J.S. (1968), *Theory of matrix structural analysis*, McGraw-Hill, New York, 1968.

40. Praeger U., Surdilovic J., Truppel I., Herold B., Geyer M., (2013), *Comparison of electronic fruits for impact detection on a laboratory scale*, Sensors 2013, 13, 7140–7155, doi:10.3390/s130607140.

41. R Core Team (2013), *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, URL http://www.R-project.org/.

42. Rao S.S. (2004), *The finite element method in engineering – 4th ed.*, Elsevier Science & Technology Books, 2004.

43. Rayleigh (Lord), (1870), *On the theory of resonance*, Transactions of the Royal Society (London) A161, 1870.

44. Ritz W. (item), *Uber eine neue methode zur losung gewissen variationsprobleme der mathematischen physik*, J. Reine Angew. Math., 135, 1909.

45. Storey R.M.J., Davies H.V. (1992), *Tuber quality*, P. Harris (Editor), The Potato Crop. Chapman and Hall, London, 507–569.

46. Turner M.J., Dill E.H., Martin H.C., Melosh R.J. (1960), *Large deflections of structures subjected to heating and external loads*, Journal of Aeronautical Sciences 27, 1960.

47. Tennes B.R., Zapp H.R., Armstrong P.R., Marshall D.E. (1991), *Combined impact sensing and data acquisition units for studies of bruising during handling of fruits and vegetables*, JAER, 49:189–196.

48. Van Canneyt T., Tijskens E., Ramon H., Verschoore R., Sonck B. (2003), *Characterisation of a potato-shaped instrumented device*, Biosystems Engineering 275–285.

49. Wilson E.L., Nickell R.E. (1966), *Application of the finite element method to heat conduction analysis*, Nuclear Engineering Design, 4, 1966.

50. Zapp H.R., Ehlert S.H., Brown G.K., Armstrong P.R., Sober S.S. (1990), *Advanced Instrumented Sphere (IS) for Impact Measurements*, Transactions of the ASAE, 33 (3):955–960.

51. Zienkiewcz O.C., Cheung Y.K. (1967), *The finite element method in structural and continuum mechanics*, McGraw-Hill, London, 1967.

Internet sites

1. http://www.code-aster.org
2. http://www.caelinux.org