CONTACT DETECTION AND CONSTRAINTS ENFORCEMENT FOR THE SIMULATION OF PELLET/CLAD THERMO-MECHANICAL CONTACT IN NUCLEAR FUEL RODS

A Dissertation

by

DAMIEN THOMAS LEBRUN-GRANDIÉ

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Jean C. Ragusa
Co-Chair of Committee,	Wolfgang Bangerth
Committee Members,	Marvin L. Adams
	Ryan G. McClarren
Head of Department,	Yassin A. Hassan

May 2014

Major Subject: Nuclear Engineering

ABSTRACT

As fission process heats up the fuel rods, UO_2 pellets stacked on top of each other swell both radially and axially, while the surrounding Zircaloy cladding creeps down, so that the pellets eventually come into contact with the clad. This exacerbates chemical degradation of the protective cladding and high stress values may enable the formation and propagation of cracks, thus threatening the integrity of the clad. Along these lines, pellet-cladding interaction establishes itself as a major concern for fuel rod design and core operation in light water reactors. Accurately modeling fuel behavior is challenging because the mechanical contact problem strongly depends on temperature distribution and the pellet-clad coupled heat transfer problem is, in turn, affected by changes in geometry induced by body deformations and stresses generated at the contact interface.

Our work focuses on active set strategies to determine the actual contact area in high-fidelity coupled physics fuel performance codes. The approach consists of two steps: in the first one, we determine the boundary region on standard finite element meshes where the contact conditions shall be enforced to prevent objects from occupying the same space. For this purpose, we developed and implemented an efficient parallel search algorithm for detecting mesh inter-penetration and vertex/mesh overlap. The second step deals with solving the mechanical equilibrium taking into account the contact conditions computed in the first step. To do so, we developed a modified version of the multi-point constraint strategy. While the original algorithm was restricted to the Jacobi preconditioned conjugate gradient method, our approach works with any Krylov solver and does not put any restriction on the type of preconditioner used. The multibody thermo-mechanical contact problem is tackled using modern numerics, with continuous finite elements and a Newton-based monolithic strategy to handle nonlinearities (the one stemming from the contact condition itself as well as the one due to the temperature-dependence of the fuel thermal conductivity, for instance) and coupling between the various physics components (gap conductance sensitive to the clad-pellet distance, thermal expansion coefficient or Young's modulus affected by temperature changes, etc.). We will provide different numerical examples for contact problems using one and multiple bodies in order to demonstrate the performance of the method.

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Prof. Jean Ragusa for giving me the opportunity to come to Texas A&M University for my dissertation and work on a project that has been both interesting and enjoyable. It has been a great pleasure and unique learning experience to work under your guidance. Thank you for putting your confidence in me and in the work I have accomplished here.

I thank Prof. Wolfgang Bangerth for his encouragements and for the useful discussions we had. Thank you for accepting to be the co-supervisor of this dissertation.

I also would like to express my gratitude to Dr. Kevin Clarno for hosting me two consecutive summers at Oak Ridge National Laboratory. I am grateful to all members of the Advanced MultiPhysics group who have been always available for me. In particular, I would like to thank Dr. Rahul Sampath for his guidance.

Unfortunately, I am not able to list all of the individuals who have helped me in many different ways. I would like to sincerely thank all of them.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

CHAPTER I

INTRODUCTION

The cladding of a nuclear fuel rod constitutes the very first barrier against the release of radioactive fission products to the environment. As such, maintaining its integrity is a major concern for light water reactor safety and it is thus crucial to be able to simulate and predict accurately cladding damage process. Pellet-cladding interaction (PCI) is a notoriously important failure mode, especially as burnup increases [3]. Contact between UO_2 pellets and the inner surface of the Zirconium alloy cladding may lead to several mode of failure.

Fuel performance analyses deal with the complex interactions of various physical phenomena. Over the last forty years, the knowledge gained from post-irradiation examinations of fuel rods has greatly contributed to a better understanding of PCI. Today, with the ongoing advances in computational science and computer technology, PCI modeling and high-fidelity numerical simulations can also give a new insight into fuel/clad behavior during normal operation and under accident conditions.

Several key challenges need to be addressed before modeling and simulation can be used to predict with high levels of fidelity the behavior of fuel rods in a nuclear reactor. One of these challenges deals with the modeling of thermo-mechanical contact between multiple bodies (fuel pellets and clad) and this is the main thrust of this dissertation. Indeed, the anticipated grid resolution for high-fidelity simulations will be such that each of the $\sim$300 fuel pellets per rod can be finely meshed in 3D and each pellet may undergo a different power/irradiation history, resulting in different mesh displacements for each pellet. A contact search will need to be performed, in parallel, with the neighboring pellets and the surrounding clad. Once contact has been detected, constraints will need to be formulated and enforced into the equation system. Finally, because fuel behavior analysis is, by nature, a multiphysics and even multiscale problem, the methods we propose to handle thermo-mechanical contact should be tested with the current state-of-the-art multiphysics solution strategies.

This dissertation is organized as follows: Chapter II presents the equations governing light water reactor (LWR) fuel rod behavior, then Chapter III discusses the current state-of-the-art in nuclear fuel performance simulations and Chapter IV provides an overview of the methods used by modern multiphysics applications. Chapter V focuses on solution strategies for the treatment of contact problems, Chapter VI describes in detail an algorithm for efficient contact detection, Chapter VII discusses different techniques to enforce the constraints from contact. Chapter VIII gives a number of numerical results demonstrating the ability to model thermo-mechanical contact between fuel pellets and cladding.

CHAPTER II

PHYSICS MODELS

Here, we formulate the nuclear fuel performance problem. We describe briefly the problem geometry before giving, first, the thermal and, second, the mechanical governing laws that dictate fuel behavior. These are essentially a standard nonlinear heat conduction equation coupled with a displacement model. We have opted for a linear elastic displacement since the main focus of this Dissertation is on contact detection and contact enforcement. The fuel rod behavior governing equations are coupled together via boundary conditions for thermal transfer between fuel pellets and cladding and mechanical contact. We close this Chapter by discussing the challenges associated with solving such a system of equations.

II.1 Problem geometry

A typical fuel rod is a four-meter-long Zirconium alloy annular tube of about 1cm in diameter. A rod contains roughly 300 fuel pellets of cylindrical shape, piled to form the fuel stack. A free space called plenum is left in the upper part of the rod to allow for fuel stack elongation and to accommodate for the release of gaseous and volatile fission products. The fuel stack is maintained during handling operation by a spring placed in the plenum and the fuel rod is hermetically sealed at both ends after being filled with a neutral gas, typically helium.

Fuel pellets are shaped from Uranium dioxide by a pressing process and are then sintered at high temperature in a controlled atmosphere for several hours. They are generally designed with end dishes to counterbalance the hour-glassing of pellets and chamfers to facilitate their introduction into the clad tubes. Even though the most detailed computational mesh may account for these, in our research we will often work with simple cylindrical representations but a simulations that employs dished and chamfered pellets will also be provided. The cladding is made of Zirconium alloy which, in addition to being highly corrosion-resistant, has a low neutron absorption. It will be modeled as a regular cylindrical tube. The fuel rods are bundled into fuel assemblies, typically in a 17-by-17 array for PWRs and a 10-by-10 for BWRs, but fuel performance simulations are mostly performed at the single rod level.

As fission process heats up the fuel pellets, the relatively poor thermal conductivity of UO_2 results in strong radial temperature gradients. The differential thermal expansion along the fuel pellet radius causes then the pellet ends to bow outward which leads to the so-called hour-glass shape. The edges of the top and bottom pellet surfaces deform further the cladding with high local strains and stresses, producing bulges that give the commonly observed bamboo-like profile along the rod length. Figure II.1 gives a schematic representation of the pellet-cladding mechanical interaction (PCMI).

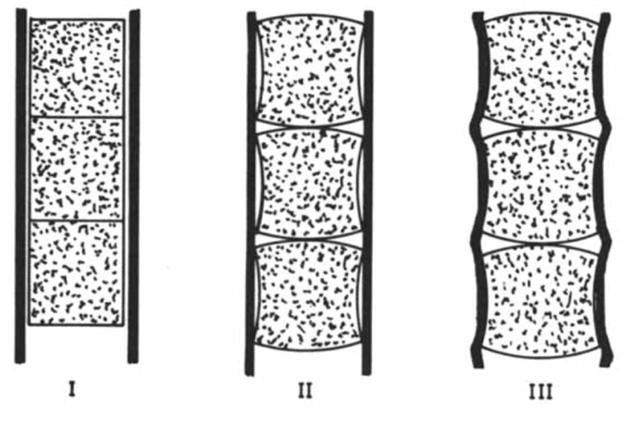


Figure II.1: Schematic representation of mechanical interaction between fuel pellets and cladding in a LWR fuel rod [1]. A number of UO_2 pellets stacked atop each other into the cladding tube: As fabricated (I), prior to PCMI (II), after PCMI (III).

II.2 Heat transfer

The temperature distribution throughout fuel pellets and cladding is calculated according to the conservation energy principle given by the standard heat conduction equation:

$$\rho C_p \frac{\partial T}{\partial t} + \nabla \cdot \boldsymbol{\phi_q} = \begin{cases} S_q, & \forall \boldsymbol{X} \in \Omega^{fuel} = \bigcup_i \Omega^{pellet,i} \\ 0, & \forall \boldsymbol{X} \in \Omega^{clad} \end{cases} . \tag{II.1}$$

$T(\boldsymbol{X}, t)$ is the temperature at point $\boldsymbol{X}$ in the interior of the domain $\Omega^{fuel} \cup \Omega^{clad}$ and at time t. The heat flux $\boldsymbol{\phi_q}$ is expressed using Fourier's law as:

$$\boldsymbol{\phi_q} = -k \nabla T, \tag{II.2}$$

where the material thermal conductivity $k(T)$ is a function of temperature, making Equation (II.1) nonlinear. In principle, heat capacity C_p and density ρ may also be temperature-dependent. Material properties are sensitive to burnup and irradiation history as well and one may include such models where, for instance, k deteriorates in the cladding as the water-side oxide layer grows.

The source term $S_q(\boldsymbol{X}, t)$ accounts for nuclear heating from fission reactions within the fuel pellets. Most fuel performance codes include power profiles that are sensitive to burnup and plutonium buildup at the pellet rim due to the ^{238}U self-shielding. It can also be derived from separate neutronics calculations. Heat generation induced by gamma absorption or by the exothermic oxidation process of Zirconium are often left aside, so we will take S_q equal to 0 in the cladding.

The surface heat fluxes between neighboring bodies are prescribed by means of the gap conductance h^{gap} as follows:

$$\boldsymbol{\phi_q} \cdot \boldsymbol{n} = h^{gap}(T - T^{neighbor}), \qquad \forall \boldsymbol{X} \in \partial\Omega^{fuel} \cup \partial\Omega^{inner\ clad}. \tag{II.3}$$

Equation (II.3) models an effective heat conduction across the gap proportional to the difference between gap interface temperatures T and $T^{neighbor}$. $\boldsymbol{n}$ is the outward-pointing unit normal at point $\boldsymbol{X}$ on the domain boundary $\partial\Omega$. The heat transfer coefficient h^{gap} depends notably on the gap width or contact pressure and thus is coupled to the displacement $\boldsymbol{u}(\boldsymbol{X}, t)$. Nevertheless, the apparent simplicity of the equation hides a more intricate reality: the conductance across the gap or interface between UO_2 and Zircaloy may be considered as the sum of three terms: (i) heat transfer across the gap by conduction through the gap, (ii) solid conductance across points or areas of contact between fuel and cladding, and (iii) radiative heat transfer. Convective heat transfer within the gap is generally neglected. We emphasize that the *neighbor* designation encapsulates thermal transfer from fuel pellet to the cladding as well as between two pellets, even though, to the best of our knowledge, the latter is often omitted in computations.

Finally, heat transfer from the cladding outer surface to surrounding water coolant is given by:

$$\boldsymbol{\phi_q} \cdot \boldsymbol{n} = h^{coolant}(T - T^{coolant}), \qquad \forall \boldsymbol{X} \in \partial\Omega^{outer\ clad}, \tag{II.4}$$

where $T^{coolant}$ is the bulk coolant/moderator temperature and $h^{coolant}$ a convective heat transfer coefficient depending upon coolant properties and flow regime. In some instances, $T^{coolant}$ is obtained by coupling the heat equation to a fluid flow model.

II.3 Mechanics

Cladding and fuel pellets deformations are computed with the mechanical equilibrium equations:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{b} = \boldsymbol{0}, \qquad \forall \boldsymbol{X} \in \Omega^{fuel} \cup \Omega^{clad}, \tag{II.5}$$

where $\boldsymbol{b}(\boldsymbol{X}, t)$ are body forces and ρ stands for the material density. The Cauchy stress tensor, $\boldsymbol{\sigma}$, is coupled to the displacement $\boldsymbol{u}(\boldsymbol{X}, t)$ via the infinitesimal strain tensor[1]:

$$\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T\right), \tag{II.6}$$

and Hooke's law:

$$\boldsymbol{\sigma} = \boldsymbol{C} : \boldsymbol{\varepsilon}^{\boldsymbol{el}} = \boldsymbol{C} : \left(\boldsymbol{\varepsilon} - \sum \boldsymbol{\varepsilon}^{\boldsymbol{in}}\right). \tag{II.7}$$

The fourth-order stiffness tensor $\boldsymbol{C}$ is dependent upon Poisson's ratio ν and Young's modulus $E(T)$. In Equation (II.7), we use the fact that the infinitesimal strain $\boldsymbol{\varepsilon}$ can be expressed as the sum of the elastic strain $\boldsymbol{\varepsilon}^{\boldsymbol{el}}$ plus inelastic components $\sum \boldsymbol{\varepsilon}^{\boldsymbol{in}}$ that consist of various contributions. Fuel performance codes generally include models for thermal expansion, material plastic deformation, expansion due to swelling and contraction due to densification, temperature-, stress-, and irradiation-induced creep, as well as relocation:

$$\sum \boldsymbol{\varepsilon}^{\boldsymbol{in}} = \boldsymbol{\varepsilon}^{\boldsymbol{th}} + \boldsymbol{\varepsilon}^{\boldsymbol{pl}} + \boldsymbol{\varepsilon}^{\boldsymbol{sw}} + \boldsymbol{\varepsilon}^{\boldsymbol{de}} + \boldsymbol{\varepsilon}^{\boldsymbol{cr}} + \boldsymbol{\varepsilon}^{\boldsymbol{re}}.$$

[1] In the limit of small deformations, displacement gradients are $o(1)$ and the strain tensor can be linearized, $\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T + \nabla \boldsymbol{u}(\nabla \boldsymbol{u})^T\right) \approx \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T\right)$. Hence, one important assumption of the model is that the maximum displacement will always be considerably smaller than the characteristic dimension of the objects which undergo the deformation.

In the context of coupled heat transport/thermo-mechanical contact simulations, one contribution representative of the problem is the thermal strain $\boldsymbol{\varepsilon}^{\boldsymbol{th}} = \alpha \boldsymbol{I} \left(T - T_{ref}\right)$, where $\alpha(T)$ is the thermal expansion coefficient, T_{ref} is the temperature at which thermal strain is zero, and $\boldsymbol{I}$ is the identity tensor. In this work, we will often consider that $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^{\boldsymbol{el}} + \boldsymbol{\varepsilon}^{\boldsymbol{th}}$.

Homogeneous Dirichlet boundary conditions are prescribed on $\Gamma^D \subset \partial\Omega$ in order to eliminate rigid body rotation and translation modes:

$$\boldsymbol{u} = \boldsymbol{0}, \qquad \forall \boldsymbol{X} \in \Gamma^D. \tag{II.8}$$

A surface load can be prescribed onto the cladding outer surface to account for the external coolant pressure. Internal gas pressure forces may as well be applied onto the inner surface of the cladding and boundary regions of the fuel pellets not subject to contact with the clad. Let Γ^C denote the possible contact boundary which, a priori, is equal to $\partial\Omega^{fuel} \cup \partial\Omega^{inner\ clad}$. We distinguish the "active" contact zone $\Gamma^{C,A}$ where contact occurs from the "inactive" part $\Gamma^{C,I} := \Gamma^C \setminus \Gamma^{C,A}$. The surface traction $\boldsymbol{t} = \boldsymbol{\sigma} \cdot \boldsymbol{n}$ is given on $\Gamma^N \subset \partial\Omega \setminus \Gamma^D$:

$$\boldsymbol{t} = \begin{cases} -P^{ext}\boldsymbol{n}, & \forall \boldsymbol{X} \in \Gamma^N \cap \partial\Omega^{outer\ clad} \\ -P^{int}\boldsymbol{n}, & \forall \boldsymbol{X} \in \Gamma^{C,I} \end{cases}.$$

Since the main focus of this research is contact modeling, we will spare the effort associated with the use of a model for the evolution of the internal pressure P^{int} as a function of irradiation history and ignore these Neumann boundary conditions.

Handling contact between neighboring bodies is somewhat more delicate. We give here an overview of the considerations that yield the common formulation of the contact conditions, which are another source of nonlinearity in the system of equations for the coupled problem.

Contact mechanics between bodies is essentially based on the non-penetration condition and the action-reaction principle. We introduce the gap gauge function $\boldsymbol{g} = \boldsymbol{x}^{neighbor} - \boldsymbol{x}$, which, when dotted against the normal $\boldsymbol{n}$ is a measure of the distance to the neighboring body for points of Γ^C. $\boldsymbol{x} \equiv \boldsymbol{X} + \boldsymbol{u}$ denotes the coordinates in the current configuration of the body, where $\boldsymbol{X}$ is related to the initial configuration and $\boldsymbol{u}$ is the displacement field. $\boldsymbol{x}^{neighbor}$ designates the minimum distance projection of $\boldsymbol{x}$ onto the surface of the neighbor.

Contact takes place when $\boldsymbol{g} \cdot \boldsymbol{n} = 0$. In that case, the condition $\boldsymbol{t} \cdot \boldsymbol{n} < 0$, stating that contact forces can only be compressive, must be verified. If there is a gap between the bodies, then $\boldsymbol{g} \cdot \boldsymbol{n} > 0$ and $\boldsymbol{t} = \boldsymbol{0}$ holds, which leads us to the following formulation for the contact conditions along the normal vector:

$$\left.\begin{array}{r} \boldsymbol{g} \cdot \boldsymbol{n} \geq 0 \\ \boldsymbol{t} \cdot \boldsymbol{n} \leq 0 \\ (\boldsymbol{g} \cdot \boldsymbol{n})(\boldsymbol{t} \cdot \boldsymbol{n}) = 0 \end{array}\right\}, \qquad \forall \boldsymbol{X} \in \Gamma^C := \partial\Omega^{fuel} \cup \partial\Omega^{inner\ clad}. \tag{II.9}$$

From here, we can picture two limiting cases. If we assume frictionless contact, only the compressive normal stresses are transmitted through the contact interface and bodies are free to slip in the tangential direction. The shear traction (tangential component of the surface traction) is null. Conversely, we can impose that the points that are in contact are not allowed to move in a tangential direction, which often referred to as “stick” or “no-slip” condition. In brief, we have:

$$\begin{array}{lllr} \text{slip condition:} & \boldsymbol{t} \cdot \boldsymbol{n} = -\boldsymbol{t}^{neighbor} \cdot \boldsymbol{n} & \text{and } \boldsymbol{t} - (\boldsymbol{t} \cdot \boldsymbol{n})\boldsymbol{n} = \boldsymbol{0} & \forall \boldsymbol{X} \in \Gamma^{C,A} \\ \text{stick condition:} & \boldsymbol{g} = \boldsymbol{0} & \text{and } \boldsymbol{t} = -\boldsymbol{t}^{neighbor} & \forall \boldsymbol{X} \in \Gamma^{C,A} \end{array} .$$

II.4 Discussion

Accurately modeling fuel behavior is challenging because the mechanical contact problem strongly depends on temperature distribution, and the coupled pellet-cladding heat transfer problem, in turn, is affected by changes in geometry induced by bodies deformations and stresses generated at contact interface. Consequently, we see that tackling the problem as formulated above will involve solving a system of nonlinear tightly-coupled equations.

CHAPTER III

CURRENT STATUS OF FUEL PERFORMANCE SIMULATIONS

In principle, the fuel rod performance problem is three-dimensional in nature. However, the fuel rod geometry suggests an axisymmetric approach. Arguing that, in normal operation, and even more in accidental conditions, the radial temperature gradient dominates over the axial gradient [3], most fuel performance codes reduce further the original problem to a one-dimensional calculation. Analyzing the fuel rod at several axial positions with a 1-D radial description, these codes are referred to as quasi-two-dimensional or one-and-a-half-D codes.

The steady-state single-rod code FRAPCON [21] and the corresponding transient code FRAPTRAN [22], used by the United States Nuclear Regulatory Commission, fall into this category. They predict fuel rod performance in PWRs and BWRs by modeling the materials response of both the fuel and cladding, under normal operating conditions or under fast transient and accident condition respectively. Their Japanese and European counterparts, respectively FEMAXI [40] and TRANSURANUS [30], work along the same lines. These codes have the capability to calculate quantities like cladding and fuel temperatures, cladding and fuel strains, cladding waterside corrosion, fission gas release from the fuel and rod internal pressure as a function of irradiation history. Their approach to the problem is essentially empirical. In good part because they have been able to reduce so dramatically the computational cost while maintaining robust data, they are useful for analyzing the entire fuel rod during a complicated, long power history.

Such codes have their merits but are unable to capture local effects such as the fuel pellets hour-glassing which induce increased strains and stresses on the cladding at pellet extremities, distorting the cylindrical shape of the cladding into a stalk of bamboo. Such detailed analysis requires a multidimensional treatment. Notable exceptions would be the real 2-D code FALCON [37] which offer the possibility to analyze axisymmetric (r, z) or plane (r, θ) problems and TOUTATIS [4] as an example of 3-D code. The fact remains that all the approaches listed so far are weakly coupled in the sense that the thermal and mechanical analyses are performed separately in an operator-split fashion.

Over the last half decade, there has been several efforts to develop next-generation of three-dimensional high-fidelity simulation tools. Among them we note the application ALCYONE from the French new fuel simulation platform PLEIADES [31], Idaho National Laboratory's BISON fuel performance code [34] or the recent AMP (Advanced Multi-Physics) code [13]. BISON is probably the most prominent effort at the present time; its mechanical contact treatment between fuel pellets and the inner surface of the cladding is described in [43]. Nonetheless, contact modeling for fuel rod behavior applications is at a somewhat early developmental stage regarding algorithms capable to handle the mechanical interaction between bodies in the specific context of multidimensional nonlinear multi-physics modeling.

Since thermal calculations rely on heat fluxes from the pellets to the cladding, it is of paramount importance to accurately treat the contact problem [41]. Consequently, this highlights the importance for future fuel performance codes to improve the state-of-the-art in therms of thermo-mechanical contact modeling. The main goal of this Dissertation is to develop algorithms to model accurately the mechanical interaction between pellets and clad. To conclude this literature review, we would also like to mention another approach developed to handle contact; it is based on a mortar-finite-element discretization of the contact interface [23] but has only been applied to two-dimensional problems so far.

Finally, we intentionally left aside other approaches that used commercial softwares such as COMSOL Multiphysics [36, 33, 32] or ABAQUS [42]. Up to now, some existing softwares may offer attractive out-of-the-box features, allowing users to implement custom models, up to certain extent, but they cannot address the needs of advanced LWR fuel rod behavior modeling in terms of problem size and parallel computations. In addition, since their source code is usually proprietary, they may not be suitable frameworks for developing new solution algorithms.

CHAPTER IV

MULTIPHYSICS STRATEGIES

This chapter gives an overview of current solution techniques for multiphysics applications. It begins with a description of coupling techniques, followed by a general discussion on solver methods and a discussion on spatial and temporal discretizations of the individual physics components. Further details on modern approaches to the multiphysics coupling of PDE-based models can be found in the recent review article [28].

IV.1 Strategies to tackle the coupled system

In this section, we review approaches to handle multiphysics coupling between different physical models. But first of all, let us introduce some generic notation that will be useful throughout the chapter. We rewrite the coupled temperature evolution and mechanical equilibrium problem as

$$\frac{\partial T}{\partial t} = f_1(T, u), \tag{IV.1}$$

and

$$F_2(T, u) = 0. \tag{IV.2}$$

T and u, as usual, are the solution for the temperature and displacement fields. For convenience, we use the same operator notation for both the continuous and the discrete formulations, which are simply distinguished by the context.

When the transient thermal transfer problem in Equation (IV.1) is semi-discretized in time, we put it into the residual form

$$F_1(T, u) = 0 \tag{IV.1bis}$$

and we solve sequentially the single-physics problems (IV.1bis) and (IV.2) to obtain values of the solution $\{T(t_n), u(t_n)\}$ at a series of discrete times $\{t_n\}_{n=0,1,2,\ldots,N_t}$. Note that these equations may generically be used to describe either a single step of our original transient problem or simply a steady-state version of it. In our notation, uppercase operator F_* denotes the residual for the individual component y_* in an equilibrium problem, and, in an evolution problem, lowercase f_* denotes its tendency. f_* is sometimes referred to as "steady-state" residual as opposed to the "transient" residual $\frac{\partial y_*}{\partial t} - f_*$. In our dissertation work, the multiphysics problem has only two components, $y_1 \equiv T$ and $y_2 \equiv u$, but we could include additional models[1] and then we would have $* \in \{1, 2, \ldots, N_c\}$, N_c standing for the total number of physics components.

IV.1.1 Conventional operator splitting approach

From a practical standpoint, it is very tempting to reuse existing independent mono-disciplinary codes, which are, ideally, in widespread use and have been rigorously tested. This common practice leads to the "loosely coupled" operator-splitting approach. In our case, it would consist in solving the thermal problem (IV.1bis) for the unknown distribution of temperatures, given the body deformation, and solving the mechanics problem (IV.2) for the unknown displacements field, given the temperature profile.

Coupling is taken into account by iterating over the pair of single-physics problem

[1] e.g., we could couple (IV.1) and (IV.2) to a model for the diffusion of Oxygen in UO_{2+x}, since both thermal conductivity and mechanical properties of the fuel pellet are sensitive to Oxygen hyperstoichiometry x, [32].

in the manner of Gauss-Seidel, as shown Algorithm 1. A Jacobi-like coupling scheme is also possible. Nevertheless, if not iterated and properly converged, this approach will not fully resolve the nonlinearities between the physics components.

Algorithm 1 Gauss-Seidel coupling technique

```
given $\{T^0, u^0\}$
for $k = 1, 2, \ldots$, until convergence
    solve for $T$ in $F_1(T, u^{k-1}) = 0$ and set $T^k = T$
    solve for $u$ in $F_2(T^k, u) = 0$ and set $u^k = u$
end for
```

The simplest approach to the transient problem is given in Algorithm 2 which produces solution values at times $t_0 < t_1 < t_2 < \ldots < t_{N_t}$. An inner loop may be placed inside each time step in which the coupling variables are updated in order to resolve the lagged nonlinearities in the operator split approach. There is no point in using higher-order discretizations in time otherwise [35].

Algorithm 2 Basic operator splitting

```
given $\{T(t_0), u(t_0)\}$
for $n = 1, 2, \ldots, N_t$
    compute one time step in $\partial_t T + f_1(T, u(t_{n-1})) = 0$ to obtain $T(t_n)$
    solve for $u$ in $F_2(T(t_n), u) = 0$ and set $u(t_n) = u$
end for
```

IV.1.2 Monolithic solution procedure

In order to avoid inconsistencies in the handling of the coupling terms that may occur with traditional operator splitting techniques, the problem can be formulated in terms of a single residual

$$F(y) := \begin{pmatrix} F_1(T,u) \\ F_2(T,u) \end{pmatrix} = 0, \tag{IV.3}$$

where $y = (T,u)$ refers generically to the multiphysics solution, which has only two components in our case but may have N_c components more generally.

Algorithm 3 Newton's method

```
given $y^0$
for $k = 1, 2, \ldots$, until convergence
    solve $J(y^{k-1})\delta y = -F(y^{k-1})$
    update $y^k = y^{k-1} + \delta y$
end for
```

Newton's method can be used to solve Equation (IV.3). Its basic form is given in Algorithm 3. By definition, the Jacobian matrix J is given by

$$J(y) := \begin{pmatrix} \frac{\partial F_1}{\partial T} & \frac{\partial F_1}{\partial u} \\ \frac{\partial F_2}{\partial T} & \frac{\partial F_2}{\partial u} \end{pmatrix}. \tag{IV.4}$$

Diagonal blocks represent individual (uncoupled) physics components, whereas off-diagonal blocks give the multiphysics coupling between them. Because of the inclusion of the latter, the method is regarded as "tightly coupled".

Assume that the system $F(y) = 0$ arises from the coupling of N_c individually well-posed discrete problems, and that it has N_d degrees of freedom total. J, which is by definition the matrix of all first-order partial derivatives of the residual F with respect to the vector of unknown $y = (y_1, y_2, \ldots, y_{N_c})^T$, is a $N_d \times N_d$ matrix which has the same sparsity structure as standard stiffness matrices.

Note that if the residuals and their derivatives are not sufficiently smooth or if one is not willing to write the extra amount of code to assemble the matrix, the Jacobian may be calculated numerically by finite difference while looping over the volume elements to evaluate the residual. A matrix-free approach is also feasible as we will see in the next section.

IV.2 Solvers

Multiphysics problems are almost inevitably nonlinear, but their solution requires solving a series of linear subproblems. In this section, we first briefly summarize currently available techniques for solving systems of linear equations, and, second, give a short review of methods for nonlinear problems.

IV.2.1 Methods for systems of linear equations

Systems of linear equations naturally arise in multiphysics applications and linear solvers are often computational bottlenecks. Consider the system of linear equations

$$Ay = b \quad A \in \mathbb{R}^{N_d \times N_d} \quad y, b \in \mathbb{R}^{N_d} \tag{IV.5}$$

representing either an entire multiphysics problem or some subproblem encountered in the solution algorithm of that problem. N_d stands for the number of degrees of freedom, which depends on the number of coupled physics and their spatial discretization. Methods for solving the equation fall into two categories: direct and iterative methods.

Direct methods generally scale too poorly as N_d increases to be considered practical for realistic multiphysics problem. When dealing with reasonably small sparse systems, LU factorization can be an option for inverting the matrix [15]. But current best practices for the efficient solving of Equation (IV.5) are based almost exclusively on multigrid methods and preconditioned Krylov subspace methods.

IV.2.1.1 Krylov subspace methods

Let $r_0 = b - Ay_0$ be the initial residual. Krylov subspace methods produce a converging sequence of approximations $\{y_j\}_{j \geq 1}$ to the solution in the the space $\mathcal{K}_j(r_0, A) = \text{span}\{r_0, Ar_0, \ldots, A^{j-1}r_0\}$ minimizing the residual over the space generated from the sequential application of the operator A on r_0.

In the case A is a symmetric positive definite matrix[2], the conjugate gradient method (CG) will most likely be the method of choice. The preconditioned form of CG is given in Algorithm 4. The method is based on a three-term recurrence relation and uses a small number of auxiliary vectors. Only the storage of the previous search direction p_j, residual r_j, and approximation y_j vectors is actually required to construct the next p_{j+1}, r_{j+1}, and y_{j+1}.

[2] i.e., $A = A^T$ and $v^T Av > 0$ for all nonzero vector v.

Algorithm 4 Preconditioned conjugated gradient method

```
$r_0 \leftarrow b - Ay_0$
$z_0 \leftarrow M^{-1}r_0$
$p_0 \leftarrow z_0$
for $j \leftarrow 0, 1, 2, \ldots$ do
    $w_j \leftarrow Ap_j$
    $\alpha_j \leftarrow \frac{r_j^T z_j}{p_j^T w_j}$
    $y_{j+1} \leftarrow y_j + \alpha_j p_j$
    $r_{j+1} \leftarrow r_j - \alpha_j w_j$
    if $r_{j+1}$ "sufficiently small" then break end if
    $z_{j+1} \leftarrow M^{-1}r_{j+1}$
    $\beta_j \leftarrow \frac{r_{j+1}^T z_{j+1}}{r_j^T z_j}$
    $p_{j+1} \leftarrow z_{j+1} + \beta_j p_j$
end for
```

When paired with a suitable preconditioner M to accelerate convergence, CG is a robust and efficient method for solving (IV.5). Unfortunately, in multiphysics problems, circumstances where A is symmetric and positive definite are rarely encountered. In this work, however, the individual block representing mechanics is actually symmetric and positive definite (see assembly of the stiffness matrix K in Chapter V) and we propose an approach to enforce the constraints of contact that preserve that property (see Chapter VII), so, when considering mechanics only, we will be able to work with PCG.

Otherwise, the generalized minimal residual method (GMRES) is often used to solve the linear system (IV.5). The method constructs an orthonormal basis for the

Krylov space and maintain $r_j \perp \mathcal{K}_j(r_0, A)$. The major drawback of GMRES is that the amount of work and the storage required per iteration grow linearly with number of iterations j. The motivates the use of the restarted[3] and truncated[4] versions of the algorithm. An other alternative for a general non symmetric A matrix is the biconjugate stabilized gradient method (BiCGStab). We refer to [38] for more details on these Krylov subspace methods.

IV.2.1.2 Preconditioning

It is well-known that the convergence of Krylov methods depends on spectral properties of the linear system matrix A [38]. Often the matrix A arising from the discretization of PDEs is ill-conditioned so we replace the original system in Equation (IV.5) by the right preconditioned system

$$M^{-1}Ay = M^{-1}b \tag{IV.6}$$

or the left preconditioned system

$$AM^{-1}(My) = b \tag{IV.7}$$

via solving $AM^{-1}w = b$ for w and $My = w$ for y. Both (IV.6) and (IV.7) yield the same solution y as the original system provided that the preconditioner matrix M is not singular. The goal is to transform the original system into one which has better convergence properties, usually by reducing the condition number of the system

[3] The method is then referred to as GMRES(m). The difficulty consists in choosing an appropriate number m of iterations after which the procedure will be restarted. Unfortunately, they are no general guidelines to make that choice. If too small, GMRES(m) may be slow to converge, or even fail to converge. Conversely, a value of m larger than necessary involves excessive work and uses more storage.

[4] This variant, known as DQGMRES(k), is based on an incomplete orthogonalization procedure. At iteration $j > k$, the vector $A^{j-1}r_0$ is orthogonalized against the k previous Krylov vectors instead of all of them.

matrix $M^{-1}A$ or AM^{-1} since the rate of convergence increases for most iterative linear solver as $\kappa(A)$ decreases. M may represent an operation or a sequence of operations that somehow approximates the effect of A^{-1} on a vector, but really, any kind of transformation making it easier to solve in terms of iterations and CPU time is acceptable.

There is a trade-off in the choice of M between the cost of applying M^{-1} at each step of the iterative linear solver and the reduction in the number of required iterations to reach convergence. The cheapest preconditioner would certainly be $M = I$ since $M^{-1} = I$, but, clearly, this results in the original linear system $Ay = b$ and the preconditioner does nothing. At the other extreme, the choice $M = A$ gives $M^{-1}A = AM^{-1} = I$, which has the optimal condition number of 1, requiring a single iteration for convergence. But in this case $M^{-1} = A^{-1}$ and applying the preconditioner is just as difficult as solving the original system. Therefore, we choose M as somewhere between these two extremes in an attempt to achieve a minimal number of iterations while keeping the operator M^{-1} as simple as possible.

IV.2.2 Methods for nonlinear systems

Nonlinear systems of the form

$$A(y)y = b \quad y, b \in \mathbb{R}^{N_d}, \tag{IV.8}$$

where operator A is a function of y, typically arise from the discretization of partial derivative equations in multiphysics problems. The fixed-point iteration, method also known as Picard iteration, is a simple and robust strategy to solve (IV.8). The method proceeds by a series of successive linearizations of the nonlinear system of equations, as described in Algorithm 5.

The main disadvantage of the fixed-point iteration is that it converges slowly

Algorithm 5 Fixed-point iteration

```
given $y^0$
for $k = 1, 2, \ldots,$ until convergence
    solve for $y$ in $A(y^{k-1})y = b$ and set $y^k = y$
end for
```

(typically linear) even when starting with a good initial guess y^0. Newton's method, given earlier in Algorithm 3, is often preferred because it offers faster convergence (up to quadratic).

The class of inexact Newton's methods generalizes Algorithm 3 by allowing computation of the update δy with an iterative method, only requiring that

$$\|J(y^{k-1})\delta y + F(y^{k-1})\| \leq \eta_k \|F(y^{k-1})\| \tag{IV.9}$$

where the tolerance sequence $\{\eta_k \in (0,1)\}$ is used to control the level of accuracy throughout the iteration [17, 18]. As explained before, the direct computation of the δy may be prohibitively expensive for multiphysics problems.

The Jacobian-free Newton-Krylov method (JFNK) is an important variant because it eliminates the need of identify and implement the Jacobian. The action of J on Krylov vectors is approximated by

$$J(y^{k-1})v \approx \frac{F(y^{k-1} + \epsilon v) - F(y^{k-1})}{\epsilon} \tag{IV.10}$$

where ϵ is a small perturbation. The matrix J is never formed, only calls to the function F are required. Of course, efficiency of JFNK depends critically on preconditioning the inner Krylov iterations [29].

Note that, fixed-point and inexact Newton methods can be used to solve multiphysics problems in a fully coupled manner, but they can also be used to implement multiphysics coupling strategies such as Algorithms 1 and 2. In any case, a matrix-free Krylov subspace approach can be advantageous because it makes possible the reuse of existing mono-disciplinary codes to compute portions of the vector function F.

IV.3 Space and time discretizations

In our dissertation work, we use finite elements for the space discretization which is generally the method of choice in analysis of heat transfer and structural mechanics. But it does not have to be, other physics components may as well be using finite difference or finite volume discretizations. All physics components do not even necessarily have to be defined on the same mesh $\mathcal{T}^h$ nor the same physical domain Ω. About that, [16] discuss matrix assembly strategies when using multiple meshes, in the extreme case using one different mesh for each physical component present in the code. For simplicity though, in our work we use a single mesh that spans the entire domain for both physics, heat transfer and thermo-mechanical contact.

In any case, what is always needed is a mean to transfer data between models and discretization schemes. As far as we are concerned, since the two physics operators are hosted on the same mesh, data transfer is straightforward in the continuum but complications come at the interface with contact and heat exchange between bodies. Mortar finite-elements, first introduced by [6], can be used for treating interface conditions and are becoming popular in the context of contact problems[45]. We will discuss in further details the mapping between boundaries of different objects in Chapter V.

IV.3.1 Finite element discretization

Let us consider the thermal aspect of our problem as an example here, the finite element formulation for the mechanics is derived in Chapter V. We consider the following time-dependent heat conduction equation equipped with initial and boundary conditions

$$\begin{cases} \rho C_p \frac{\partial T}{\partial t} - \nabla \cdot k \nabla T = S_q(\boldsymbol{X}, t) & \text{in } \Omega \\ T(\boldsymbol{X}, t_0) = T_0(\boldsymbol{X}) & \text{in } \Omega \\ T(\boldsymbol{X}, t) = T_{bd}(\boldsymbol{X}, t) & \text{on } \Gamma^D \subset \partial\Omega \\ -k \nabla T \cdot \boldsymbol{n} = \phi_q(\boldsymbol{X}, t) & \text{on } \Gamma^N = \partial\Omega \setminus \Gamma^D \end{cases} \tag{IV.11}$$

for all $t \geq t_0$. In brief, $T = T(\boldsymbol{X}, t)$ is the material temperature, ρ and C_p denote the density and heat capacity. Thermal conductivity k is typically a function of T, which introduces nonlinearity in the model. S_q is a heat source. The initial temperature distribution T_0 is prescribed at time $t = t_0$ over the whole domain Ω. Dirichlet boundary condition specifies the temperature T_{bd} on Γ^D at any time $t > t_0$, whereas Neumann condition gives the heat flux ϕ_q across the boundary Γ^N and couples the thermal problem to mechanics. $\boldsymbol{n}$ stands for the outward-pointing normal along $\partial\Omega$.

For the sake of simplicity in introducing the finite element discretization, we choose to approximate the time derivative using backward Euler

$$\frac{\partial T}{\partial t} \approx \frac{T_n - T_{n-1}}{\Delta t_n} \tag{IV.12}$$

with variable-size time step $\Delta t_n = t_n - t_{n-1}$. We derive the weak form of that system of equations, by multiplying from the left by a test function φ from the space $V = \{v \in H^1(\Omega) \mid v|_{\Gamma^D} = 0\}$ and integrating over the domain Ω. We obtain[5]

$$\langle \varphi, \rho C_p \frac{T_n - T_{n-1}}{\Delta t_n} \rangle_\Omega + \langle \nabla\varphi, k(T_n) \nabla T_n \rangle_\Omega = \langle \varphi, S_q \rangle_\Omega + \langle \varphi, \phi_q \rangle_{\Gamma^N} \tag{IV.13}$$

given T_{n-1} and for all $\varphi \in V$, with the standard notation $\langle v, u \rangle = \int vu$.

[5] We integrate by parts via Green's theorem $-\langle \varphi, \nabla \cdot k \nabla T_n \rangle_\Omega = \langle \nabla\varphi, k \nabla T_n \rangle_\Omega - \langle \varphi, k \nabla T_n \cdot \boldsymbol{n} \rangle_{\partial\Omega}$, use the fact that φ is zero on Γ^D, and substitute $-k \nabla T \cdot \boldsymbol{n}$ by ϕ_q on Γ^N.

We seek a solution over some finite dimensional subspace[6] $U^h = \text{span}\{\varphi_i\}$ by substituting $T(\boldsymbol{X},t) = \sum_j \varphi_j(\boldsymbol{X})T_j(t)$ into Equation (IV.13) and requiring that the equation be satisfied for integration against the set $\{\varphi_i\}$ of basis functions. The T_j are unknown expansion coefficients, or degrees of freedom, we need to determine. In matrix form, it yields

$$\frac{1}{\Delta t_n} M(T_n - T_{n-1}) + KT_n = f \tag{IV.14}$$

where $M_{ij} = \langle \varphi_i, \varphi_j \rangle_\Omega$ is a regular mass matrix, $K_{ij} = \langle \nabla\varphi_i, \frac{k}{\rho C_p} \nabla\varphi_j \rangle_\Omega$ is a "non-linear" stiffness matrix[7], and the right-hand side $f_i = \langle \varphi_i, \frac{1}{\rho C_p} S_q \rangle_\Omega + \langle \varphi_i, \frac{1}{\rho C_p} \phi_q \rangle_{\Gamma^N}$ accounts for the heat source and heat fluxes across the boundary.

When assembling the system, integrals over the whole domain are computed as sums of integrals over all elements $\mathcal{K}$ of the triangulation $\mathcal{T}^h$, e.g. $M_{ij} = \sum_{\mathcal{K}\in\mathcal{T}^h} \langle \varphi_i, \varphi_j \rangle_\mathcal{K}$, and the contribution of each element is approximated by means of a numerical quadrature $M_{ij}^\mathcal{K} \approx \sum_q w_q \varphi_i(\boldsymbol{X}_q)\varphi_j(\boldsymbol{X}_q)$. w_q and $\boldsymbol{X}_q$ are, respectively, the weights and integration points of the quadrature rule. Note that integrals cannot always be computed exactly (in particular those involving complex material properties) even though we can easily achieve it for the mass matrix entries which we took as an example here.

IV.3.2 Time integration techniques

The coupled heat transport and thermo-mechanical contact problem in this work is a bit particular in the sense only the thermal component of the solution is actually allowed to evolve in time, governed by its own physics, while mechanical contact

[6] Typically, $U^h \subset V$ is the discrete space of piecewise polynomial functions of degree p built on top of the triangulation $\mathcal{T}^h$ of Ω, the discretization parameter h being a measure of the mesh size. We construct U^h by piecing together basis shape functions $\varphi_i^\mathcal{K}$ defined on element $\mathcal{K} \in \mathcal{T}^h$, which have the property $\varphi_i^\mathcal{K}(\boldsymbol{X}_j) = \delta_{ij}$, where $\boldsymbol{X}_j$ denotes the coordinates of node j.

[7] Indeed the thermal conductivity is temperature-dependent, $k = k(T_n)$, but the equation is linearized in the fixed-point iteration or Newton's method. The stiffness matrix is assembled given T_n^{k-1} and we solve for T_n^k (see previous section on nonlinear solvers).

being always considered at equilibrium. Nevertheless, multiphysics problems typically couple models with very disparate timescales and the tendency $f(t, y)$ ends up having components with widely different dynamics. This calls for the use of implicit time integration schemes with favorable stability properties.

In order to obtain numerical approximations to the solution of the initial value problem

$$\frac{\partial y}{\partial t} = f(t, y), \qquad y(t_0) = y_0 \tag{IV.15}$$

at a sequence of discrete times $t_1 < t_2 < \ldots < t_{N_t}$, we employ Runge-Kutta methods. Given the value of the solution y_n at time t_n, we compute $y_{n+1} \approx y(t_{n+1})$ according to

$$y_{n+1} = y_n + \Delta t_n \sum_{i=1}^{s} b_i f(t_n + c_i \Delta t_n, Y_i), \tag{IV.16}$$

where

$$Y_i = y_n + \Delta t_n \sum_{j=1}^{s} a_{ij} f(t_n + c_j \Delta t_n, Y_j). \tag{IV.17}$$

A method with s stages can be represented represented by a “Butcher tableau”

$$\begin{array}{c|cccc}
c_1 & a_{11} & a_{12} & \ldots & a_{1s} \\
c_2 & a_{21} & a_{22} & \ldots & a_{2s} \\
\vdots & \vdots & \vdots & & \vdots \\
c_s & a_{s1} & a_{s2} & \ldots & a_{ss} \\
\hline
 & b_1 & b_2 & \ldots & b_s
\end{array}$$

indicating all the values of the coefficients a_{ij}, b_i, and c_i. The explicit methods are those where the matrix a is strictly lower triangular, whereas implicit methods include non-zero coefficients on and above the diagonal.

Explicit methods are synonyms for reduced computational cost (implementation based on s evaluation of f per time step) but these are generally unsuitable because their region of absolute stability is relatively small. When multiphysics problem include stiff components, implicit methods become more appropriate. The simplest example of an implicit method is Backward Euler which is first-order, unconditionally stable and non-oscillatory; but often, methods with higher-order p will be preferred so that larger time step can be used while still achieving the same level of accuracy.

If the matrix a is full, then a single nonlinear system of size $s \times N_d$ needs to be solved simultaneously for all Y_i, which is a very expensive process that typically dominates the overall computational costs. An important class of higher-order fully implicit methods allow a significant reduction of these costs. For these so called SDIRK methods, a is lower triangular with identical nonzero diagonal entries ($a_{ij} = 0$ for $i < j$ and $a_{ii} = \gamma$) and s nonlinear systems of size N_d are solved sequentially for the Y_i instead (refer to Equation (IV.17)). More details on Runge-Kutta methods can be found in [11].

CHAPTER V

ALGORITHMS FOR CONTACT

This chapter describes solution algorithms to handle the nonlinearity in the mechanical problem formulation due to the contact conditions. Contact detection and enforcement of the contact constraints are discussed in more detail in the next two chapters, respectively.

The contact conditions given in Equation (II.9) introduce nonlinearity in the continuum mechanics problem; this is solved using an iterative solution algorithm. The typical approach to solve the contact problem is to introduce the concept of active and inactive sets to distinguish between the set of vertices and associated degrees of freedom describing the boundary of a body that is in contact, on the one hand, and the ones that are not in contact on the other hand. The determination of the actual contact interface, aka the active set, is performed using the following iterative procedure that consists in two steps:

(i) In the first step, the active set is updated. This involves (a) searching for additional faces where contact conditions shall be enforced to prevent objects from occupying the same space (i.e., non-penetration condition) and (b) ensuring that the forces transmitted across the contact interface are compressive (i.e., verifying that there still is contact).

(ii) The second step deals with solving the mechanical equilibrium, taking into account the contact conditions computed in the first step.

V.1 Discretization of the continuum

For simplicity, we consider the mechanics problem only, independently of its coupling to other physics components. First, we derive briefly the finite element formulation for the continuum $w$ithout contact (refer to Chapter IV for more detail on the linear elastic model).

We take the standard equilibrium equation governing the deformation of two linear elastic bodies, labelled 1 and 2,

$$\nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{b} = \boldsymbol{0} \quad \text{on } \Omega = \Omega_1 \cup \Omega_2,$$

where $\boldsymbol{\sigma} = \boldsymbol{C} : \boldsymbol{\epsilon}(\boldsymbol{u})$ is the stress tensor and $\rho \boldsymbol{b}$ represent body forces (see Chapter II). We prescribe Dirichlet (displacement) and Neumann (traction) boundary conditions:

$$\boldsymbol{u} = \boldsymbol{u}_{bd} \quad \text{on } \Gamma^D \quad \text{and} \quad \boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{p} \quad \text{on } \Gamma^N.$$

We derive the finite element matrix formulation in the usual fashion by multiplying (component-wise) the equation above by test functions $\boldsymbol{\varphi}_i$ and integrating over the whole domain. We obtain

$$\begin{pmatrix} K_1 & 0 \\ 0 & K_2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}, \tag{V.1}$$

where the unknown vector u contains the node displacements, the right-hand side f represents body forces as well as surface traction

$$f_*[i] = \langle \boldsymbol{\varphi}_i, \rho \boldsymbol{b} \rangle_{\Omega_*} + \langle \boldsymbol{\varphi}_i, \boldsymbol{p} \rangle_{\Omega_* \cap \Gamma^N} \tag{V.2}$$

and the stiffness matrix K is defined by

$$K_*[i,j] = \langle \boldsymbol{\varepsilon}(\boldsymbol{\varphi}_i), \boldsymbol{\sigma}(\boldsymbol{\varphi}_j)\rangle_{\Omega_*}, \tag{V.3}$$

with $* \in \{1,2\}$.

We observe that K is block diagonal. Coupling between the two bodies will be added later once we formulate the contact constraints. We also note that when integrating by parts, we would obtain a term of the form $\langle \nabla\boldsymbol{\varphi}_i, \boldsymbol{\sigma}(\boldsymbol{\varphi}_j)\rangle_{\Omega_*}$ but we replaced it with the term involving the symmetric gradient $\boldsymbol{\varepsilon}(\boldsymbol{\varphi}_i) := \frac{1}{2}(\nabla\boldsymbol{\varphi}_i + (\nabla\boldsymbol{\varphi}_i)^T)$ instead of $\nabla\boldsymbol{\varphi}_i$. Due to the symmetry of the fourth-rank tensor $\boldsymbol{C}$ that relates stress to strain (recall that $\boldsymbol{\sigma} = \boldsymbol{C} : \boldsymbol{\varepsilon}$), the two terms are equivalent but the symmetric version is more convenient to work with. In particular, it allows the use of the preconditioned conjugate gradient method (see Chapters IV and VII).

V.2 Formulation of the contact constraints

V.2.1 Contact conditions in the normal direction

Assume that the two bodies come in contact with one another, then the classical conditions for contact [46] state that

$$\left.\begin{aligned} \boldsymbol{g}\cdot\boldsymbol{n} &\geq 0 \\ \boldsymbol{t}\cdot\boldsymbol{n} &\leq 0 \\ (\boldsymbol{g}\cdot\boldsymbol{n})(\boldsymbol{t}\cdot\boldsymbol{n}) &= 0 \end{aligned}\right\} \text{ on } \Gamma^C \subseteq \partial\Omega \setminus \Gamma^D, \tag{V.4}$$

where $\boldsymbol{g}\cdot\boldsymbol{n}$ is a measure of the distance between the two objects at points on Γ^C, the domain boundary where contact is possible. $-\boldsymbol{t}\cdot\boldsymbol{n}$ represents the contact pressure.

We further split Γ^C as follows: let $\Gamma^{C,A}$ be the actual contact zone and $\Gamma^{C,I} := \Gamma^C \setminus \Gamma^{C,A}$ its complement. We can rewrite the above equation as two different conditions to be verified on the disjoint surface areas

$$\left.\begin{array}{lll}\text{Contact is active:} & \boldsymbol{g}\cdot\boldsymbol{n}=0 \text{ and } \boldsymbol{t}\cdot\boldsymbol{n}<0 & \text{on } \Gamma^{C,A} \\ & & \\ \text{Contact is inactive:} & \boldsymbol{g}\cdot\boldsymbol{n}>0 \text{ and } \boldsymbol{t}=\boldsymbol{0} & \text{on } \Gamma^{C,I}\end{array}\right\}. \tag{V.5}$$

Equation (V.5) states that either the gap is closed and inward normal forces are present where the two objects meet (non-adhesive contact) or that the objects are not touching one another and no forces are transmitted.

The above equations are the conditions for no-penetration and no-adhesion. In addition, the stress vectors acting on either side of the contact interface $\Gamma^{C,A}$ must obey the action-reaction principle: $\boldsymbol{t}|_{\partial\Omega_1} = -\boldsymbol{t}|_{\partial\Omega_2}$ at the points of contact. In the case of frictionless contact, the tangential component of the traction (shear stresses) is zero.

V.2.2 Friction model in the tangential direction

The mathematical condition in Equation (V.5) dictates the mechanical behavior in the normal direction. Taken “as is”, i.e., without adding any further constraints in the tangential direction, this would correspond to contact without friction between the two bodies. It is often referred to as a “slip” condition. Another option would consist in forbidding the relative motion of the two objects in the tangent plane at points of contact. This is the so-called “stick” condition.

In summary, these are two limiting cases: for all points on the contact interface $\Gamma^{C,A}$, either

(i) slip case: $\boldsymbol{g}\cdot\boldsymbol{n}=0$ and $\boldsymbol{t}-(\boldsymbol{t}\cdot\boldsymbol{n})\boldsymbol{n}=\boldsymbol{0}$, or

(ii) stick case: $\boldsymbol{g}\cdot\boldsymbol{n}=0$ and $\boldsymbol{g}-(\boldsymbol{g}\cdot\boldsymbol{n})\boldsymbol{n}=\boldsymbol{0}$ which can be combined together to obtain the more simple and compact condition $\boldsymbol{g} = \mathbf{0}$.

In both cases, the forces transmitted across the contact interface must be compressive and the action-reaction principle applies, i.e. $\boldsymbol{t} \cdot \boldsymbol{n} < 0$ and $\boldsymbol{t}|_{\partial\Omega_1} = -\boldsymbol{t}|_{\partial\Omega_2}$ on $\Gamma^{C,A}$.

Although being a rough approximation for the frictional behavior of the two contacting bodies, the Coulomb friction model is widely used in the numerical simulation of surface mechanical interaction. It prescribes when to trigger the change from stick to slip condition as tangent contact stresses increase. There will be no slip at the point of contact as long as the following condition holds

$$\|\boldsymbol{t}_t\| \leq \mu \|\boldsymbol{t}_n\|.$$

Here, the surface traction has been broken up into normal and tangential components, $\boldsymbol{t}_n = (\boldsymbol{t} \cdot \boldsymbol{n})\boldsymbol{n}$ and $\boldsymbol{t}_{\boldsymbol{t}} = \boldsymbol{t} - \boldsymbol{t}_n$. μ is a friction coefficient, an empirical property that depends on materials properties. The quantity $\mu\|\boldsymbol{t}_n\|$ represents the Coulomb friction bound. When the tangential component of the traction $\|\boldsymbol{t}_n\|$ exceeds this critical value, sliding occurs.

In our dissertation work, we did not implement the Coulomb friction model and all numerical examples presented in Chapter VIII are either with infinite friction coefficient (the first results with direct elimination of the contact constraints, in Sections VIII.2.1 and VIII.2.2, fall into that category but the approach is later extended to frictionless contact in Section VIII.2.3), or frictionless (those obtained using Lagrange multipliers in Section VIII.1 as well as the some of the results with direct elimination).

V.2.3 Discretization of the contact interface

The constraints equations depend greatly upon which representation we choose for the actual contact interface $\Gamma^{C,A}$. Arbitrarily, we designate Ω_1 and Ω_2 as master and slave, respectively. Let $\Gamma_m^{C,A} := \partial\Omega_1 \cap \Gamma^{C,A}$ and $\Gamma_s^{C,A} := \partial\Omega_2 \cap \Gamma^{C,A}$ be the contact zones on both sides of the contact interface. A continuous description of the problem would yield $\Gamma^{C,A} \equiv \Gamma_m^{C,A} \equiv \Gamma_s^{C,A}$, but, in general, $\Gamma_m^{C,A} \neq \Gamma_s^{C,A}$ holds for discretized surfaces, because slave boundary faces are unlikely to be perfectly aligned with the master ones and the two meshes will not match at the contact interface. For a unique definition of the contact surface, one of the surfaces of the bodies in contact has to be chosen as common surface. Here, we select the contact surface on the slave side, $\Gamma^{C,A} := \Gamma_s^{C,A}$.

Then, we have to select how to apply the contact constraints: they can either be fulfilled in a weak sense on faces or be enforced strongly at nodes. The first option is somewhat more consistent with a finite element approach but the second may be easier to implement. In our work, we will often refer to these two approaches as ”face-to-face” for the weak integral formulation and as ”node-to-face” for the strong point-wise enforcement of the constraints, respectively. Note that after discretization of the continuous bodies, it is nearly impossible to model perfect contact between the two bodies, so whether the constraints are formulated in the strong or weak sense, local penetrations will always be permitted.

We provide numerical examples for both approaches in Chapter VIII (results with Lagrange multipliers are obtained with a face-to-face approach and results using a direct elimination of the constraints are computed by means of a node-to-face approach). Note that the choice face-to-face versus node-to-face actually has an impact on the representation of the contact interface. The former considers $\Gamma^{C,A}$ as a collection of faces on the slave side, where the non-penetration condition has been violated and constraints are formulated in response, whereas the later describes the contact interface with a cloud of slave vertices. The implications of such a choice are discussed further in Section V.2.5.

V.2.4 Concept of active and inactive sets

At this point, we introduce some notations needed in order to formulate a discretized version of the contact constraints and, more generally, to describe the solution algorithm.

Let $\mathcal{S}$ contain all the vertices on $\Gamma^C = \Gamma^{C,A} \cup \Gamma^{C,I}$. Our goal is to find the correct subset $\mathcal{A}$ of $\mathcal{S}$ for which contact occurs. We call this the active set, and its complement $\mathcal{I} := \mathcal{S} \setminus \mathcal{A}$ is the inactive set. By $\mathcal{M}$ we denote all vertices on the master side of the contact interface and by $\mathcal{N}$ all the other ones.

Let us make here the connection with the face-to-face approach we described previously. The active set then consists of all the vertices that support the slave faces on the contact interface. It is even more straightforward for the node-to-face approach, the element of $\mathcal{A}$ are exactly the same slave nodes used to describe $\Gamma^{C,A}$.

We rewrite the problem discrete formulation for the two-body case in Equation (V.1) under the form

$$\begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 & 0 \\ K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} & 0 & 0 \\ 0 & 0 & K_2^{\mathcal{A}\mathcal{A}} & K_2^{\mathcal{A}\mathcal{N}} \\ 0 & 0 & K_2^{\mathcal{N}\mathcal{A}} & K_2^{\mathcal{N}\mathcal{N}} \end{pmatrix} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} \\ f_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} \end{pmatrix}. \tag{V.6}$$

(recall 1=master, 2=slave). Here, for simplicity all the elements of $\mathcal{I}$ have been transferred into $\mathcal{N}$, which now denotes more generally all vertices that are neither in

$\mathcal{A}$ nor in $\mathcal{M}$. Note that we have kept separated the degrees of freedom distributed over the two different bodies, namely those contained in $u_1^{\mathcal{N}}$ and in $u_2^{\mathcal{N}}$. The reason is that we would like to preserve/emphasize the uncoupled nature of the system of equations without the contact constraints.

The iterative strategy to find the active set $\mathcal{A}$ is described in Section V.3.1. But first, we derive the constraints equations that couple the slave and master unknowns.

V.2.5 Constraints formulation

The contact constraints equations enforce the non-penetration condition at vertices/on faces of the slave where that condition would not be satisfied otherwise. In this section, we derive first the node-to-face and then the face-to-face formulations. The latter seems a more natural choice with finite elements even if we will see that the former also presents some advantages.

In the following, we consider the stick condition, i.e., $\boldsymbol{g} = \boldsymbol{0}$ on $\Gamma^{C,A}$. The constraints formulation for the slip condition is similar, except that we would work from a scalar projection of the condition in the direction of the normal, i.e., $\boldsymbol{g} \cdot \boldsymbol{n} = 0$ on $\Gamma^{C,A}$. In both cases, we obtain a relation coupling the slave degrees of freedom $u_2^{\mathcal{A}}$ to the master ones $u_1^{\mathcal{M}}$.

V.2.5.1 Node-to-face formulation

Contact detection determines which slave nodes penetrate into the master body and adds them to the active set $\mathcal{A}$. For each slave node $p \in \mathcal{A}$, we store the identity of the master face onto which the node is projected, together with the local coordinates of the contact point on that master face, ξ_p and η_p. Then, we use this information and write the no-slip contact condition:

$$x[p] = \sum_{q \in \mathcal{F}_p} C[p,q]x[q] \qquad \forall p \in \mathcal{A}, \tag{V.7}$$

where the coefficients $C[p,q]$ are the values of the shape functions of the master at the point of contact, i.e., $C[p,q] = \varphi_q(\xi_p, \eta_p)$ (q is an index on the master nodes). The subset $\mathcal{F}_p \subset \mathcal{M}$ contains the nodes of the master supporting the face on which the slave node p displacement is constrained. x is the vector which contains all the nodes coordinates in the current configuration of the two bodies.

Let us recall that $x = X + u$, with X referring to the nodes location in the initial configuration and u giving their displacement. This allows us to rewrite the equation (V.7) as follows

$$u[p] = \sum_{q \in \mathcal{F}_p} C[p,q]u[q] + d[p], \tag{V.8}$$

where we introduced the displacement d required to bring the slave nodes from their original position to the point of contact, i.e.,

$$d[p] = \sum_{q \in \mathcal{F}_p} C[p,q]X[q] - X[p]. \tag{V.9}$$

We collect all constraint relationships in a matrix form, shown below,

$$u_2^{\mathcal{A}} = Cu_1^{\mathcal{M}} + d_2^{\mathcal{A}}. \tag{V.10}$$

$u_2^{\mathcal{A}}$ does not represent independent degrees of freedom any more and can be eliminated using the above equation. In 3D, using eight-node hexahedral elements (`hex8`),

$|\mathcal{F}_p| = 4$ for all $p \in \mathcal{A}$, so C is a matrix with four entries per row.

$$\begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & C & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ d_2^{\mathcal{A}} \\ 0 \end{pmatrix} \tag{V.11}$$

V.2.5.2 Face-to-face formulation

The previous strong point-wise condition $\boldsymbol{g} = \boldsymbol{0}$ at slave nodes in the node-to-face approach can be replaced by a weak integral condition $\int_{\Gamma^{C,A}} \boldsymbol{\varphi}\boldsymbol{g} = \boldsymbol{0}$ enforced on slave faces, where component-wise multiplication (along each spatial direction) by a test function $\boldsymbol{\varphi} \in \boldsymbol{V}$ is implied. A natural choice for the discretized space $\boldsymbol{V}^h$ consists in keeping the nodal basis functions $\{\boldsymbol{\varphi}_p\}_{p\in\mathcal{A}}$ employed in a the FEM formulation of the mechanics problem, Equation (V.1). The non-penetration condition is enforced in a weak sense:

$$\langle \boldsymbol{\varphi}_p, \boldsymbol{g}(\boldsymbol{\varphi}_q)\rangle_{\Gamma^{C,A}} = \boldsymbol{0} \quad \forall (p,q) \in \mathcal{A} \times (\mathcal{A} \cup \mathcal{M}) \tag{V.12}$$

yielding the equation

$$A u_1^{\mathcal{M}} - M u_2^{\mathcal{A}} = g_2^{\mathcal{A}}, \tag{V.13}$$

where M is the mass matrix

$$M[p,q] = \langle \boldsymbol{\varphi}_p, \boldsymbol{\varphi}_q \rangle_{\Gamma^{C,A}} \qquad \forall (p,q) \in \mathcal{A}^2, \tag{V.14}$$

and A is the coupling matrix that connects slave and master degrees of freedom

$$A[p,q] = \langle \boldsymbol{\varphi}_p, \boldsymbol{\varphi}_q \rangle_{\Gamma^{C,A}} \quad \forall (p,q) \in \mathcal{A} \times \mathcal{M}. \tag{V.15}$$

The vector $g = (0, 0, g_2^A, 0)^T$ is a measure of the gap between slave and master faces in the initial configuration. Refer to Equation (V.11) for the meaning of the four components in such a vector. It is given by

$$g[p] = \langle \boldsymbol{\varphi}_p, X[p] - \sum_{q \in \mathcal{E}_p} A[p,q] X[q] \rangle_{\Gamma^{C,A}} \qquad \forall p \in \mathcal{A}. \tag{V.16}$$

Here, the subset $\mathcal{E}_p := \{q \in \mathcal{M} \mid A[p,q] \neq 0\}$ contains the nodes of the master faces overlapping with the basis shape function φ_p of slave. It involves evaluating the gap function at all quadrature integration points of a face where contact is active. $\mathcal{E}_p$ is determined when quadrature integration points on the slave faces that meet at node p are projected onto master faces. It contains all the master nodes supporting the faces onto which $\boldsymbol{g}$ mapped at least one quadrature point.

Eliminating the slave degrees of freedom

Some of the constraints enforcement techniques that are presented in Chapter VII require inverting the relation in Equation (V.13) to obtain a direct expression of the slave degrees of freedom $u_2^{\mathcal{A}}$ as a linear combination of the master nodal displacements $u_1^{\mathcal{M}}$. When assuming face-to-face contact, this implies inverting the mass matrix M, resulting in

$$u_2^{\mathcal{A}} = M^{-1}(A u_1^{\mathcal{M}} - g_2^{\mathcal{A}}). \tag{V.17}$$

M is sparse but M^{-1} is dense a priori. The solution strategy involves applying $(M^{-1}A)$ and $(M^{-1}A)^T$ multiple times to copy the master values to the slave or add the slave to the master in vectors (cf. Section VII.2). Therefore, as a general rule, it is advangeous if the computational cost associated with evaluate their action on $u_1^{\mathcal{A}}$ and $u_2^{\mathcal{M}}$, respectively, is reduced.

Connection with node-to-face

We can draw a parallel between the two formulations by remarking that the use of a reduced quadrature rule when assembling the terms in Equation (V.13) yields back the node-to-face formalism. If we employ a quadrature rule whose points are located at the nodes, then the mass matrix M can be lumped, making the linear solve of Equation (V.17) trivial. Furthermore, this approximation yields exactly the node-to-face constraints presented earlier, in Equation (V.10). This proof is immediate if we replace the integration over slave faces as a weighted sum over the special quadrature points and observe that $\boldsymbol{\varphi}_p(\boldsymbol{x}_q) = \delta_{pq}$.

Primal-dual spaces

Wohlmuth et al. suggest to write the weak formulation of the non-penetration condition on the dual space $\boldsymbol{W}$ instead of the primal $\boldsymbol{V}$ used in Equation (V.12), which yields

$$\langle \boldsymbol{\psi}_p, \boldsymbol{g}(\boldsymbol{\varphi}_q) \rangle_{\Gamma^{C,A}} = \mathbf{0} \quad \forall (p,q) \in \mathcal{A} \times (\mathcal{A} \cup \mathcal{M}) \tag{V.18}$$

Following [44], we introduce the dual basis shape functions $\{\boldsymbol{\psi}_p\}_{p\in\mathcal{A}}$ and take advantage of the biorthogonality relation

$$\langle \boldsymbol{\psi}_p, \boldsymbol{\varphi}_q \rangle_{\Gamma^{C,A}} = \delta_{pq} \langle \mathbf{1}, \boldsymbol{\varphi}_q \rangle_{\Gamma^{C,A}} \qquad \forall (p,q) \in \mathcal{A}^2, \tag{V.19}$$

to derive a constraints that has the form

$$A^* u_1^{\mathcal{M}} - D u_2^{\mathcal{A}} = g_2^{*\mathcal{A}} \tag{V.20}$$

where the matrix D is diagonal. Its entries are given by

$$D[p,p] = \langle \mathbf{1}, \boldsymbol{\varphi}_p \rangle_{\Gamma^{C,A}} \quad \forall p \in \mathcal{A}. \tag{V.21}$$

In Equation (V.20), the matrix A^* couples the nodal shape functions on the master side and the dual basis functions on the slave side

$$A^*[p,q] = \langle \boldsymbol{\psi}_p, \boldsymbol{\varphi}_q \rangle_{\Gamma^{C,A}} \quad \forall (p,q) \in \mathcal{A} \times \mathcal{M}, \tag{V.22}$$

and the vector g^* for the initial gap is computed using

$$g^*[p] = \langle \boldsymbol{\psi}_p, X[p] - \sum_{q \in \mathcal{E}_p^*} A^*[p,q] X[q] \rangle_{\Gamma^{C,A}} \quad \forall p \in \mathcal{A}. \tag{V.23}$$

Projecting the constraints onto the dual basis is advantageous since eliminating the slave degrees of freedom become straightforward. It does not require the tedious inversion of a mass matrix any more, it only involves inverting the diagonal entries of D, according to

$$u_2^{\mathcal{A}} = D^{-1}(A^* u_1^{\mathcal{M}} - g_2^{*\mathcal{A}}). \tag{V.24}$$

Summary

In this dissertation, we will often refer to Equation (V.13) as the general form for the contact constraints, but most algorithms will be easily adjusted to work with either type of contact constraints. We will see later in Chapter VII that only the direct elimination approach will require M to be invertible.

V.3 Active set strategy

The concept of active and inactive sets has been introduced in the previous section. This section deals with the determination of actual active set.

V.3.1 Iterative approach

Mathematically, the iterative strategy to identify the actual contact interface can be describe as follows: determine

$$\Gamma_{k+1}^{C,A} = \left\{ \boldsymbol{X} \in \Gamma_k^{C,I} \mid \boldsymbol{g} \cdot \boldsymbol{n} \leq 0 \right\} \cup \left\{ \boldsymbol{X} \in \Gamma_k^{C,A} \mid \boldsymbol{t} \cdot \boldsymbol{n} < 0 \right\} \tag{V.25}$$

by iterating over k until convergence. At each iteration k, we compute the displacement field $\boldsymbol{u}$ and we search among the points for which contact was inactive the ones that violate the non-penetration condition if we move the mesh in accordance with the solution. Similarly, among the points that were already in contact, we identify those at which the transmitted force is no longer compressive (refer to Equation (V.5)).

The iterative scheme to predict the correct active and inactive sets $\mathcal{A}_{k+1}$ and $\mathcal{I}_{k+1}$ is the following:

(0) Initialize $\mathcal{A}_0$ and $\mathcal{I}_0$ such that $\mathcal{A}_0 \cup \mathcal{I}_0 = \mathcal{S}$ and $\mathcal{A}_0 \cap \mathcal{I}_0 = \emptyset$.

Typically, compute:

$$\mathcal{A}_0 = \{ v \in \mathcal{S} \mid \boldsymbol{g} \cdot \boldsymbol{n} \leq 0 \}$$

$$\mathcal{I}_0 = \{ v \in \mathcal{S} \mid \boldsymbol{g} \cdot \boldsymbol{n} > 0 \}$$

Set $k = 0$.

(1) Solve the problem with the active set $\mathcal{A}_k$ and the associated constraints (V.6) and (V.12).

(2) Update active and inactive sets.

$$\mathcal{A}_{k+1} = \{v \in \mathcal{A}_k \mid \boldsymbol{t} \cdot \boldsymbol{n} > 0\} \cup \{v \in \mathcal{I}_k \mid \boldsymbol{g} \cdot \boldsymbol{n} \leq 0\}$$

$$\mathcal{I}_{k+1} = \{v \in \mathcal{A}_k \mid \boldsymbol{t} \cdot \boldsymbol{n} \leq 0\} \cup \{v \in \mathcal{I}_k \mid \boldsymbol{g} \cdot \boldsymbol{n} > 0\}$$

(3) If $\mathcal{A}_{k+1} = \mathcal{A}_k$ and $\mathcal{I}_{k+1} = \mathcal{I}_k$, then stop.

Otherwise set $k = k + 1$ and go to step (1).

The algorithm uses a discretized version of the update criterion in Equation (V.25) for step (2), which is described in more details in the next subsection. The methods to enforce the contact constraints at each iteration and compute the displacement field $\boldsymbol{u}^k$ in step(1) are discussed in Chapter VII.

V.3.2 Activating/deactivating a slave vertex or face

As a general rule, contact is activated where the non-penetration condition is violated and deactivated if the contact forces are no longer compressive. In this subsection we describe in more details how the active and inactive sets are updated throughout the computation.

V.3.2.1 Case of node-to-face contact

For all inactive slave vertices, we want to compute the quantity $g_n = \boldsymbol{g} \cdot \boldsymbol{n}$ which measures the distance between the two bodies. The gap function $\boldsymbol{g}$ maps points of the possible contact boundary Γ_s^C of the slave surface to points of Γ_m^C on the master. In particular, $\boldsymbol{g}$ projects any slave vertex/node $x[p]$ onto faces of the master and returns the closest point $\sum_{q \in \mathcal{F}_p} C[p, q]x[q]$, sitting on the master face that we mark as $\mathcal{F}_p$. Note that the normal vector $\boldsymbol{n}$ is not necessarily well-defined at the mesh vertices. A possible alternative is to take the normal vector computed at its projection on $\mathcal{F}_p$

and make it point in the opposite direction. Nevertheless, what really matters in g_n is more the correctness of the sign than the accuracy of the value [46]. In the case of penetration, $g_n(x[p]) \leq 0$ holds and we store the $C[p,q]$ coefficients so that we can formulate a new constraint for slave node p (refer to Equation (V.8)).

For all active slave vertices, we are interested in the contact pressure, which is given by $p_n = -\boldsymbol{t} \cdot \boldsymbol{n}$. The surface traction $\boldsymbol{t}$, by definition, is equal to $\boldsymbol{C} : \boldsymbol{\varepsilon}(\boldsymbol{u})$ dotted against $\boldsymbol{n}$ (i.e., $\boldsymbol{t} = \boldsymbol{\sigma} \cdot \boldsymbol{n}$). Unfortunately, since the strain is calculated as a function of the displacement partial derivatives, which are discontinuous across elements, $\boldsymbol{t}$ is undefined at the nodes of the mesh. But here also, only the sign of contact pressure is really important, not its actual value. Hence, one option would be to just look at the point force which we can obtain by adding the contributions from all slave elements that have p as a vertex. Instead, in our implementation, we take advantage of the action-reaction principle and perform the computation at the contact point on the master side. It is convenient because we access $\mathcal{F}_p$ and $C[p,q]$ to obtain the normal vector anyway. If the contact pressure $p_n(x[p])$ is found to be negative, then we remove p from the active set and discard the associated contact constraint.

In summary, once we have solved for the displacement $\boldsymbol{u}^k$ field at iteration k, the strategy to determine the next active and inactive sets is given in Algorithm 6.

V.3.2.2 Case of face-to-face contact

We proceed slightly differently: $\boldsymbol{g}$, $\boldsymbol{t}$, and $\boldsymbol{n}$ are evaluated at quadrature integration points instead of being computed at slave nodes. Integrals over slave face elements $\partial\Omega_{\mathcal{F}}$ are performed using a numerical quadrature according to the formula

$$\int_{\partial\Omega_{\mathcal{F}}} f(\boldsymbol{x}) ds \approx \sum_q f(x_q) w_q. \tag{V.26}$$

Algorithm 6 Strategy to update the active set in the node-to-face approach

```
initialize $\mathcal{A}_{k+1} = \mathcal{I}_{k+1} = \emptyset$

for all node $p$ in inactive set $\mathcal{I}_k$
    compute $g_n(x[p])$
    if $g_n(x[p]) > 0$ then
        add $p$ to $\mathcal{A}_{k+1}$ (the node remains inactive)
    else
        add $p$ to $\mathcal{I}_{k+1}$ (activate the node)
    end if
end for

for all node $p$ in active set $\mathcal{A}_k$
    compute $t_n(x[p])$
    if $t_n(x[p]) \leq 0$ then
        add $p$ to $\mathcal{A}_{k+1}$ (the node remains active)
    else
        add $p$ to $\mathcal{I}_{k+1}$ (deactivate the node)
    end if
end for
```

w_q are the weights of the quadrature rule and $x_q = \sum_{p' \in \mathcal{F}} x[p']\varphi_{p'}(\xi_q, \eta_q)$ gives the current location of the quadrature point q on the possible slave contact boundary. (ξ_q, η_q) are local coordinates on $\partial\Omega_{\mathcal{F}}$ and p' points to the slave nodes supporting the face element. We purposefully use p' and not p here to highlight that p' are not the slave nodes in contact, but the slave nodes supporting the slave faces ($\mathcal{F}$) in contact.

Fortunately[1] here is that, as long as we are not using a special quadrature rule that has the integration points located at nodes $p' \in \mathcal{F}$, the normal vector and the surface traction can be directly computed on the slave side. Note that even if only four Gauss points are usually sufficient to integrate exactly when using piecewise linear finite elements (which is standard for mechanics), it is recommended to use a

[1] This has its importance for parallel simulations, since it reduces communication cost. We do not have to transfer data back and forth between the processor that owns the slave face and the one(s) knowing about the master face(s) on which the quadrature points are projected onto.

Gaussian quadrature rule of higher order for the contact terms [19]. This is mainly due to the fact that integration points on the same slave face get projected onto different master faces.

The strategy to update the active set is almost the same as for node-to-face except that we loop over face elements instead of nodes and that we compute average values for the gap and the normal traction instead of nodal values.

V.3.3 Efficiency and robustness

Here are a few essential questions regarding efficiency and robustness of our algorithm:

- Does it converge to the solution?
- How fast is the convergence? Does it depend upon problem size?
- How efficient is the algorithm? (number of operations within an iteration of the active set strategy, total number of iterations, memory used)

The first question is essential. Unfortunately, multibody contact problems do not admit analytical solutions in general. In Sections VIII.2.1 and VIII.2.2, we revisited two numerical examples that we found in the literature. In the first example with the stacked cubes, we perform a convergence study using a reference solution on a finer mesh to evaluate discretization errors. The second example, where a cylinder is pressed onto a quasi-rigid brick, allows us to compare the computed stresses and displacements against some analytical values.

We note that most problems are represented by a three-dimensional discretization, leading to very large number of unknowns. The number of degrees of freedom N_d grows inversely to the cube of the mesh size h. One of the biggest challenge in computational contact mechanics is the detection of mesh inter-penetration and vertex/mesh overlap [46]. The contact search has to be performed numerous times throughout the computation (one search per each step of the active set iteration and a full active set iteration at each time or loading step) which calls for a robust and efficient implementation of the search. The number of nodes and faces on the possible contact boundary Γ^C is proportional to h^{-2} and, unless accelerated, projection onto faces of the neighbor or search over all its volume elements for each of these nodes and faces are $O(h^{-2})$ and $O(h^{-3})$ respectively. We present a fast a reliable contact detection algorithm in Chapter VI which ensure that the update of the active set does not dominate the time for computation. As a result, the bottleneck in our computation is always the Krylov solver which is quite standard in multiphysics problems.

Different techniques for enforcing contact constraints are discussed in Chapter VII. These have advantages and disadvantages concerning efficiency, accuracy, or robustness. For instance, when using the penalty method, small penalty values induce imprecision, but the iterative solver will perform poorly if ϵ is too high. With Lagrange multipliers, the size of the linear systems is increased and additional effort is needed for preconditioning. Also, the use of infinite friction coefficients, notably with the direct elimination approach, raise a supplementary issue in convergence analysis. In some cases, it requires the load to be applied gradually over several steps.

V.3.4 Closing remarks

V.3.4.1 Strain-stress transformation rules

We have to move/update the mesh to after each active set iteration k to detect whether any slave node or face that was previously inactive penetrates into the master and must be activated in response. Yet, we actually perform all our computations in the initial configuration Ω^0, which makes the computation much easier. If we were not doing so, not only would we have to keep track of the history of the stress tensor $\boldsymbol{\sigma}^k$, but, more importantly, at the end of every step we would have to transfer it to the new deformed domain Ω^k before we can compute the next incremental displacement $\Delta\boldsymbol{u}^{k+1}$.

Here, in short, is what would happen otherwise. At each iteration k, we would be looking for $\Delta\boldsymbol{u}^k \in \{\boldsymbol{w} \in (H^1(\Omega^{k-1}))^3 \mid \boldsymbol{w} = \boldsymbol{0} \text{ on } \Gamma^D\}$ such that

$$\langle C : \boldsymbol{\varepsilon}(\Delta\boldsymbol{u}^k), \boldsymbol{\varepsilon}(\boldsymbol{\varphi})\rangle_{\Omega^{k-1}} = f(\boldsymbol{\varphi}) - \langle \boldsymbol{\sigma}^{k-1}, \boldsymbol{\varepsilon}(\boldsymbol{\varphi})\rangle_{\Omega^{k-1}} \tag{V.27}$$

for all test functions $\boldsymbol{\varphi}$, and where the linear functional f on the right-hand-side accounts for the body forces and the traction prescribed on Γ^N. Then the stress variable $\boldsymbol{\sigma}^k$ would be updated according to

$$\boldsymbol{\sigma}^k = \boldsymbol{\sigma}^{k-1} + \boldsymbol{C} : \boldsymbol{\varepsilon}(\Delta\boldsymbol{u}^k) \text{ on } \Omega^{k-1}, \tag{V.28}$$

and it would be used in the next iteration $k+1$ in the term $\langle \boldsymbol{\sigma}^k, \boldsymbol{\varepsilon}(\boldsymbol{\varphi})\rangle_{\Omega^k}$ as a correction to the right-hand side forcing term. And here comes the complication: $\boldsymbol{\sigma}^k$ was computed on Ω^{k-1} and it is defined with respect to the coordinate system in that particular configuration of the two bodies. It needs to be transferred to the distorted domain Ω^k which involves [14] the infinitesimal rotation tensor $\boldsymbol{\omega}(\Delta\boldsymbol{u}^k) := \frac{1}{2}(\nabla\boldsymbol{u} - (\nabla\boldsymbol{u})^T)$ describing the rotation due to the displacement increment $\Delta\boldsymbol{u}^k$ at each point:

$$\boldsymbol{\sigma}^k = \boldsymbol{\omega}(\Delta\boldsymbol{u}^k)^T \left[\boldsymbol{\sigma}^{k-1} + \boldsymbol{C} : \boldsymbol{\varepsilon}(\Delta\boldsymbol{u}^k)\right] \boldsymbol{\omega}(\Delta\boldsymbol{u}^k) \text{ on } \Omega^k. \tag{V.29}$$

V.3.4.2 Load stepping

In our dissertation work, some of the numerical examples in Chapter VIII using the stick condition for contact (e.g., linear elastic cylinder pressed against a hard brick or cladding deformation due to the thermal expansion of the pellets) require us to apply the load in a stepwise fashion. We scale the forcing term (e.g., prescribed pressure on the boundary or non-uniform temperature profile across the material) by means of some load parameter κ^n that we increase in a series of small increments until it reaches the desired final value. Assume we prescribed a surface load $\boldsymbol{p}$ on Γ^N. The nth step consists in determining the displacement $\boldsymbol{u^n}$ that satisfies

$$\left.\begin{aligned} \langle C : \boldsymbol{\varepsilon}(\boldsymbol{u}^n), \boldsymbol{\varepsilon}(\boldsymbol{\varphi})\rangle_{\Omega^0} &= \langle \kappa^n \boldsymbol{p}, \boldsymbol{\varphi}\rangle_{\Gamma^N} \\ \\ \langle \boldsymbol{g}(\boldsymbol{u}^n), \boldsymbol{\varphi}\rangle_{\Gamma_n^{C,A}} &= \boldsymbol{0} \end{aligned}\right\} \quad \forall \boldsymbol{\varphi}. \tag{V.30}$$

Whether solving for successive time or loading steps, we always make sure to compute the nodal displacement with respect to the initial configuration of the bodies Ω^0.

CHAPTER VI

CONTACT DETECTION

This chapter presents an efficient and robust search algorithm that can be applied to detect mesh interpenetration and vertex/mesh overlap in the context a contact mechanics problem. The algorithm determines whether some point is located within a given mesh or not. If the answer to that question is yes, the identity of the volume element which contains that point together with its local coordinates in the frame of reference of the element are returned.

The search for contact is a tedious task but it is an actual cornerstone in the solution strategy presented in Chapter V Detecting contact is trivial in one-dimensional calculations but becomes quickly more complicated as the number of dimensions increases, especially if efficiency and scalability considerations are brought into play.

In the context of fuel performance simulations, the geometry and topology of the contact interface are rather simple. The fuel pellets of cylindrical shape are stacked atop each other in the cladding tube. The interaction is hence limited to immediate neighbors as defined by the initial configuration. In other words, besides the cladding inner surface in the vicinity of a given pellet, one pellet in the stack may only come into contact with the lower surface of the pellet above and the upper surface of the one below.

In the light of this observation, a naive implementation of contact search to determine whether one point, say, on top of a pellet penetrates into the pellet above, would consist in checking whether that point lies inside any of the cells on the neighboring pellet mesh. Unfortunately, not only the computational cost for the request associated with a single point grows as meshes are refined and the number of cells increases, but also the number of requests itself augments rapidly, since it is tied to the amount of vertices or integration points on all boundary elements on the discretized surface of the pellet.

We cannot do much about the $O(1/h^2) = O(|\mathcal{S}|)$ growth of the “number of requests” as the grid is refined, but we can greatly improve the overall scalability by controlling the cost of a single request. This can be achieved if we manage to restrict the fine search over cells that are in the immediate surrounding by means of a preliminary coarse filtering.

VI.1 Coarse parallel search

The aim of the coarse search is to inexpensively identify the subset of elements that contains a given point. The idea behind is we want to restrict the fine local search for a point to a small finite number of volume elements. We would like this number to be relatively small, and, more particularly, we do not want it to increase drastically as the meshes is refined.

VI.1.1 Partitioning the search domain

Traversing the entire mesh and searching for a point in each of the volume elements would be a computationally very expensive task. The key to accelerate the search is space partitioning. We divide the physical space occupied by the mesh into several non-overlapping subregions. Any point in the space can then be identified to lie in only one and exactly one of the subregions.

The domain is partitioned using an octree representation. The bounding box of the mesh (i.e., the smallest axis-aligned hyper-rectangle that contains it) is taken as the initial search space and recursively divided into eight octants. Figure VI.1 shows an example with two levels of octree refinement. For a given maximum level of refinement m, any octant is uniquely determined by the integer coordinate $(x, y, z) \in$

$\{0, \ldots, 2^m - 1\}$ of its back left lower corner node with respect to the octree coordinate system and its depth of refinement (level l, $0 \leq l \leq m$). Note that points are scaled in each dimension to be positive integers such that all octants are cubes whose side length is a power of two, and corner coordinates that are multiples of that side length.

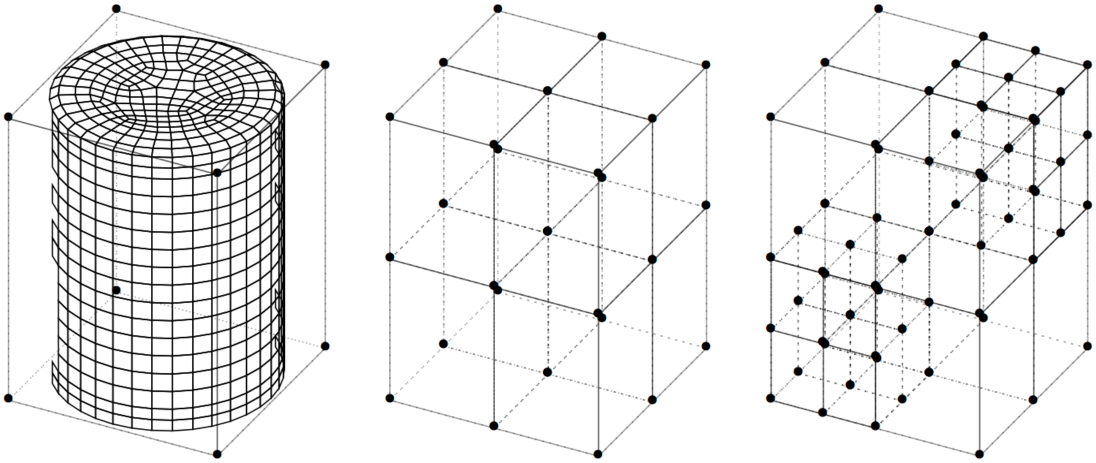


Figure VI.1: Recursive subdivision of the mesh bounding box into octants.

All octants can be organized into a tree data structure, as represented in Figure VI.2. Only octants with no descendants are stored. These are the leaves of the octree, covering the search domain with neither holes nor overlaps. The octant leaves are sorted using a Z-curve which yields their Morton index. The Morton index is simply calculated by interleaving the binary representations of the octant corner point coordinate values, as illustrated in Table VI.1. Morton ordering (also known as Z-ordering) is a simple way in which to optimize the usage of cache when accessing localized areas of memory. We refer to [10] and [39] for more details on the properties of octrees and Morton encoding. Connecting indices in ascending numerical order

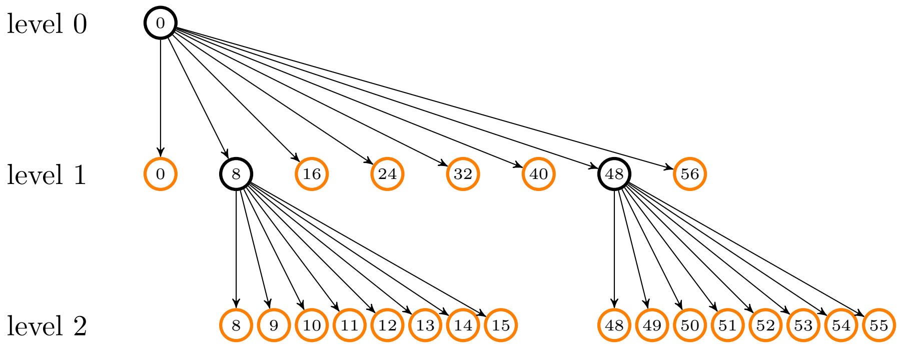


Figure VI.2: Corresponding octree to Figure VI.1. The leaves of the octree are colored in orange. The number shown are Morton indices.

produces the Z-shaped space-filling curve that is plotted on Figure VI.3.

Table VI.1: Computing Morton indices for the children of the level-one octant with corner point at (2, 0, 0) (lower left octant facing the reader in Figure VI.1), assuming maximum depth is two. Corner point coordinates are given both in decimal and binary bases.

Children location	Corner point (x, y, z)	Morton index
lower-left-back	$(2, 0, 0) \equiv (10, 00, 00)$	$001000 \equiv 8$
lower-left-front	$(3, 0, 0) \equiv (11, 00, 00)$	$001001 \equiv 9$
lower-right-back	$(2, 1, 0) \equiv (10, 01, 00)$	$001010 \equiv 10$
lower-right-front	$(3, 1, 0) \equiv (11, 01, 00)$	$001011 \equiv 11$
upper-left-back	$(2, 0, 1) \equiv (10, 00, 01)$	$001100 \equiv 12$
upper-left-front	$(3, 0, 1) \equiv (11, 00, 01)$	$001101 \equiv 13$
upper-right-back	$(2, 1, 1) \equiv (10, 01, 01)$	$001110 \equiv 14$
upper-right-front	$(3, 1, 1) \equiv (11, 01, 01)$	$001111 \equiv 15$

VI.1.2 Mapping between octants and mesh volume elements

We have seen in the previous section how to subdivide the search domain. Now, we would like to identify what volume elements on the mesh are in the vicinity of each of the subregions[1] so that we will be able to restrict the fine search to these

[1] i.e., determine which elements intersect with which subregion.

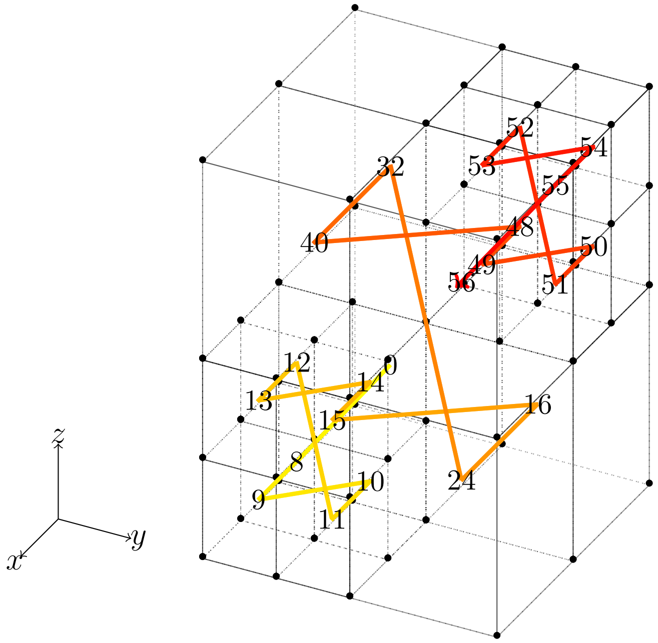


Figure VI.3: Space-filling Z-curve showing the total ordering of all the octree leaves.

elements. In other words, the goal is to associate leaf octants with mesh volume elements.

We proceed as follows: For all mesh volume elements, we establish the list of octree leaves that overlap with the element bounding box. That list may point to one or several octants, depending on the location of the element and on how fine the space partition is. Then, we invert the map in such a manner that each of the leaf octants is matched with volume elements that had the octant listed.

The main thing as for search efficiency and scalability is to control the number of volume elements by leaf octant. Ideally, we would ensure that it does not grow excessively when meshes are made finer by adaptively subdividing further the octants. Nevertheless, because of the relative uniformity in element size and shape in the meshes we typically work with, we have only employed regular grids so far and used some heuristic[2] to determine the refinement level of the octree. Figure VI.4 illustrates the coarse filtering for a fuel pellet element of various levels of mesh refinement.

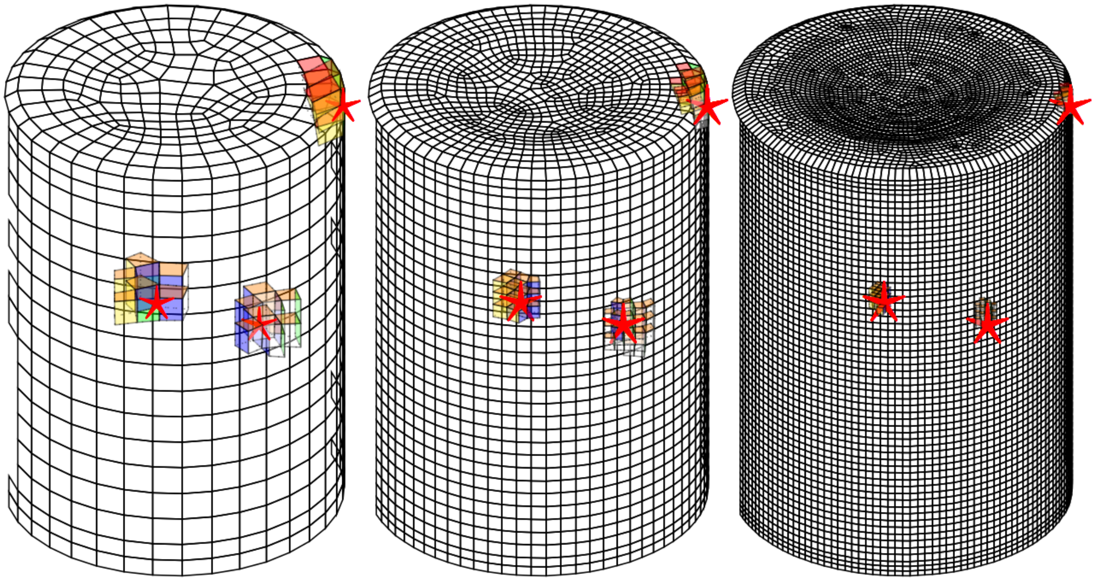


Figure VI.4: Searching for three random points over three pellet meshes, with respectively, from left to right, 3705, 26146, and 183210 elements. The coarse search indicates which volume elements could possibly contain the points.

VI.1.3 Coarse filtering

Subdividing space and mapping subregions to mesh volume elements can be seen as an initial setup stage of the coarse search. It only needs to be done once as long as the mesh does not change. The coarse search per se only consists in finding out what subregion of space (octant leaf) the point is lying in and then reading the map to see what elements are likely to contain it.

[2] The octree depth of refinement is chosen to be proportional to $\log_2 \lceil N_e^{1/3} \rceil$, which ensures that the ratio total number of volume elements N_e to number of octants is constant.

Consider a point we would like to search for over some mesh that has been partitioned into a collection of leaf octants before and suppose that all leaves are maintained in sorted order. Identifying in which one of the subregions the point is lying in is relatively easy. We first determine which octant of the maximum level ($l = m$) contains the point, which only implies scaling the point coordinates in each direction and truncating them to integers to obtain the octant corner point location in the octree coordinate system. Next, we compute its Morton index and simply perform a binary search over the sorted list of leaf octants which returns either directly at the octant in the list with the same index if it exists or at its ancestor.

Figure VI.5 sums up the coarse search procedure. Computing in parallel with distributed meshes make things somewhat more complicated but the strategy remains the same. The ordered list of octants is distributed across processes that own the mesh. We have to find out first what process has the mapping information and then perform the fine search on processes that own the volume elements. involves two stages[3] of communication between processes for the coarse search before the actual point-element test and communicating back the results.

VI.2 Local fine search

Once a subset of candidate elements has been determined by the coarse search, the process is complete by a local fine search. The local fine search must address the following problem: We would like to determine whether a given volume element contains some point. Then, if the answer is “yes”, we need to know the location of that point with respect to the coordinate system of that element. Since detecting mesh overlap involves answering these questions over and over again, we understand

[3] First, forward the request to the processor which owns the octant leaf (containing the point which is searched for) in order to be able to read the map, and, second, dispatch to the processors which own the elements to perform the fine search.

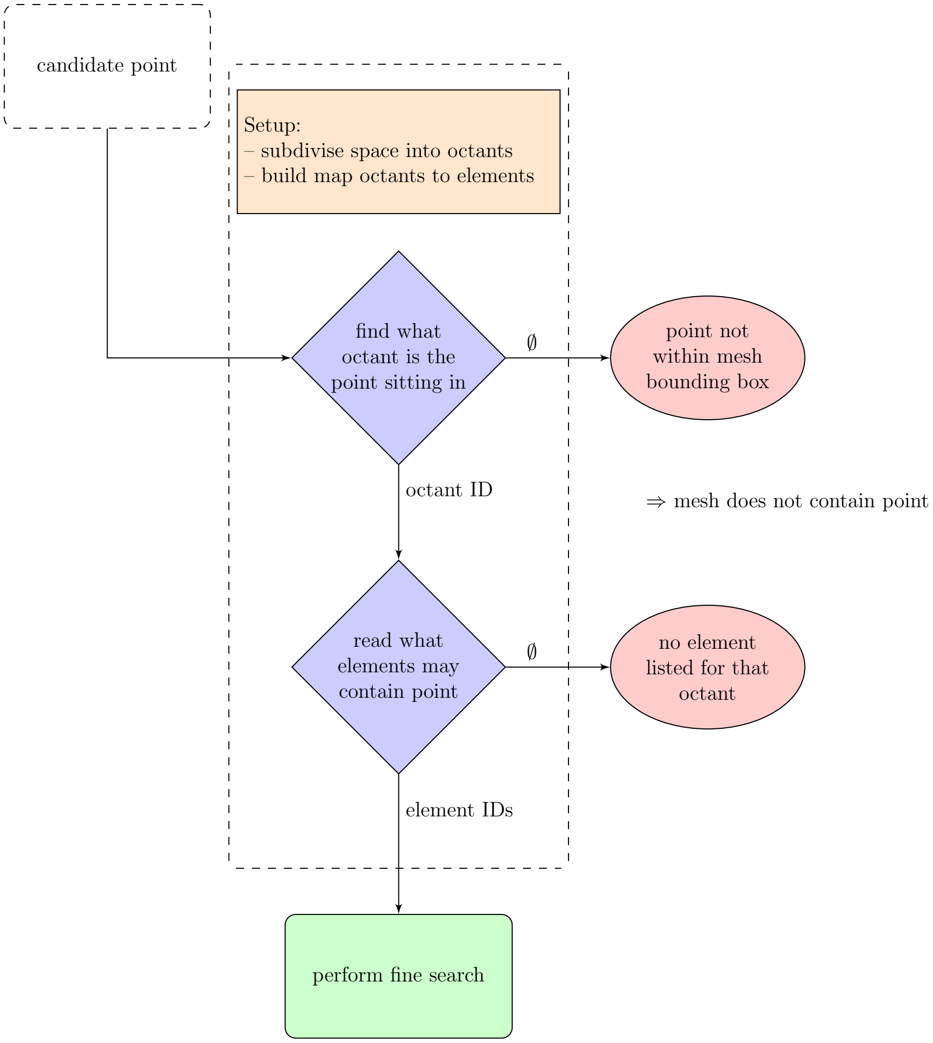

candidate point
Setup:
– subdivise space into octants
– build map octants to elements
find what octant is the point sitting in
∅
point not within mesh bounding box
octant ID
⇒ mesh does not contain point
read what elements may contain point
∅
no element listed for that octant
element IDs
perform fine search

Figure VI.5: Flowchart illustrating the mesh level strategy to assess which volume elements may contain a point.

that we truly need the search to be robust and efficient.

VI.2.1 Mapping points to the reference volume element

The conventional brute-force way to assess whether a point is contained within a volume element is to directly compute the point coordinates in the frame of reference in which the element is a cube of edge length 2 centered at the origin and check that all of its coordinates lie between -1 and 1. This involves handling a system of 3 coupled nonlinear equations.

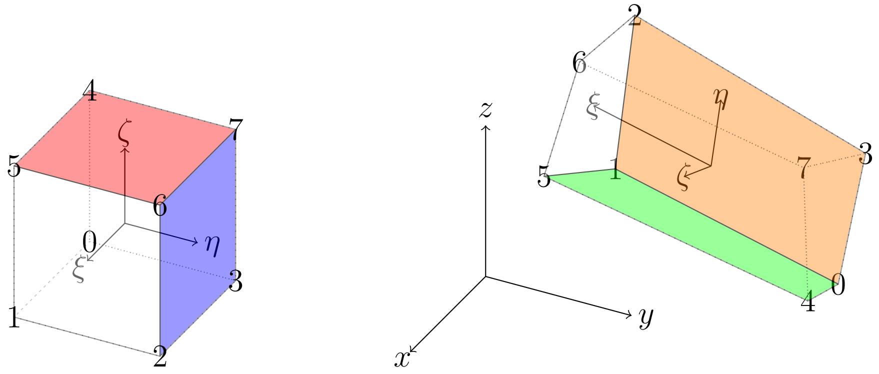


Figure VI.6: The regular hexahedron mapped onto some distorted volume element.

Figure VI.6 illustrates the three-dimensional mapping between the eight-node hexahedral reference element and some “real” mesh volume element. The mapping from local coordinates (ξ, η, ζ) to global coordinates (x, y, z) is given by the function $F : \mathbb{R}^3 \to \mathbb{R}^3$ such that

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \\ z \end{pmatrix} = F(\xi, \eta, \zeta) = \begin{pmatrix} \sum_i x_i \varphi_i(\xi, \eta, \zeta) \\ \sum_i y_i \varphi_i(\xi, \eta, \zeta) \\ \sum_i z_i \varphi_i(\xi, \eta, \zeta) \end{pmatrix},$$

where (x_i, y_i, z_i) are the global coordinates of node i, and $\varphi_i : \mathbb{R}^3 \to \mathbb{R}$ the associated linear shape function, as indicated in Table VI.2. The Jacobian matrix of the

Table VI.2: Numbering of the nodes in the regular hexahedron and associated shape functions.

i	(ξ_i, η_i, ζ_i)	$\varphi_i(\xi, \eta, \zeta)$
0	$(-1, -1, -1)$	$0.125(1-\xi)(1-\eta)(1-\zeta)$
1	$(+1, -1, -1)$	$0.125(1+\xi)(1-\eta)(1-\zeta)$
2	$(+1, +1, -1)$	$0.125(1+\xi)(1+\eta)(1-\zeta)$
3	$(-1, +1, -1)$	$0.125(1-\xi)(1+\eta)(1-\zeta)$
4	$(-1, -1, +1)$	$0.125(1-\xi)(1-\eta)(1+\zeta)$
5	$(+1, -1, +1)$	$0.125(1+\xi)(1-\eta)(1+\zeta)$
6	$(+1, +1, +1)$	$0.125(1+\xi)(1+\eta)(1+\zeta)$
7	$(-1, +1, +1)$	$0.125(1-\xi)(1+\eta)(1+\zeta)$

transformation is

$$\boldsymbol{J}_F(\xi, \eta, \zeta) = \begin{pmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial x}{\partial \eta} & \dfrac{\partial x}{\partial \zeta} \\ \dfrac{\partial y}{\partial \xi} & \dfrac{\partial y}{\partial \eta} & \dfrac{\partial y}{\partial \zeta} \\ \dfrac{\partial z}{\partial \xi} & \dfrac{\partial z}{\partial \eta} & \dfrac{\partial z}{\partial \zeta} \end{pmatrix},$$

which can be written in terms of the shape functions as

$$\boldsymbol{J}_F(\xi, \eta, \zeta) = \begin{pmatrix} \sum_i \dfrac{\partial \varphi_i}{\partial \xi} x_i & \sum_i \dfrac{\partial \varphi_i}{\partial \eta} x_i & \sum_i \dfrac{\partial \varphi_i}{\partial \zeta} x_i \\ \sum_i \dfrac{\partial \varphi_i}{\partial \xi} y_i & \sum_i \dfrac{\partial \varphi_i}{\partial \eta} y_i & \sum_i \dfrac{\partial \varphi_i}{\partial \zeta} y_i \\ \sum_i \dfrac{\partial \varphi_i}{\partial \xi} z_i & \sum_i \dfrac{\partial \varphi_i}{\partial \eta} z_i & \sum_i \dfrac{\partial \varphi_i}{\partial \zeta} z_i \end{pmatrix}.$$

We are interested in the inverse transformation that maps global coordinates (x, y, z)

back to (ξ, η, ζ). We define the residual vector $\boldsymbol{R}$ as

$$\boldsymbol{R}(\xi, \eta, \zeta) = \begin{pmatrix} \sum\limits_i x_i \varphi_i - x \\ \sum\limits_i y_i \varphi_i - y \\ \sum\limits_i z_i \varphi_i - z \end{pmatrix}.$$

For simplicity of notation, let us also refer to the reference and global coordinates as Ξ and $\boldsymbol{X}$, respectively. Given $\boldsymbol{X}$, Ξ is solution of

$$\boldsymbol{R}(\Xi) = \mathbf{0}, \tag{VI.1}$$

which we solve by means of the standard Newton-Raphson method

$$\Xi^{n+1} = \Xi^n - \boldsymbol{J}_F^{-1}(\Xi^n)\boldsymbol{R}(\Xi^n). \tag{VI.2}$$

We pick an initial guess Ξ^0 and iterate over n until convergence. Finally, we check whether $\Xi \in [-1, +1]^3$ and conclude whether the point lies inside the volume element or not.

Unfortunately, convergence is not guaranteed and a strategy that would consist in systematically attempting to map points to the reference volume will lead to disaster[4]. In the following subsection, we describe two common techniques to reduce the chances of having Newton failing.

[4] It would force us to impose some maximum number of Newton steps and consider that non-convergence is synonym for point not contained in the volume element, which means a lot of iterations for nothing...

VI.2.2 Initial guess and bounding box

It is well known that a poor choice of initial point Ξ^0 may lead to the non-convergence of the algorithm. Thus it is essential that we start the Newton iteration with an initial guess that is reasonably close to the true solution. To obtain a good estimate, we determine the vector from the element centroid, $\boldsymbol{O} = F(0,0,0)$, to the point $\boldsymbol{X}$, and we multiply it by the inverse of the Jacobian matrix evaluated at $\boldsymbol{O}$, i.e.

$$\Xi^0 = \boldsymbol{J}_F^{-1}(0,0,0)(\boldsymbol{X} - \boldsymbol{O}).$$

Also we spare ourselves the trouble of trying to map points for which we can readily tell it is obvious they are not within the volume element. It is pointless to run the newton iteration for a point if its global coordinates along each of the three axes are not bounded by the extrema of all the nodes coordinates in the same direction. Mathematically, it means that we want to make sure that the point we consider verifies

$$\boldsymbol{X} \in [\min_i x_i, \max_i x_i] \times [\min_i y_i, \max_i y_i] \times [\min_i z_i, \max_i z_i].$$

Geometrically, it can be seen as us testing whether $\boldsymbol{X}$ is contained within the bounding box associated to the volume element, as shown in Figure VI.7, the bounding box being the smallest axis-aligned rectangular parallelepiped in which the nodes supporting the element are enclosed.

Combining both a good initial guess and the bounding box check is a good decision since it is relatively inexpensive and improves efficiency. More importantly it considerably decreases the rate of occurrence of failure when performing the inverse nonlinear mapping (Newton iteration in Equation VI.2). However, this still does

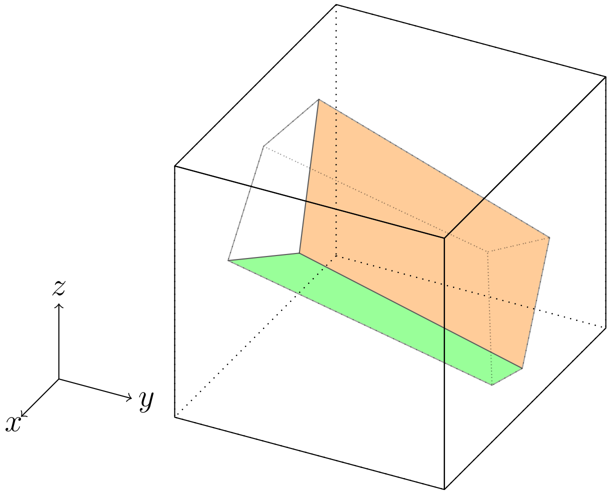


Figure VI.7: Axis-aligned bounding box.

not meet our expectations in term of robustness. As it turns out, cases where the volume element is not even that distorted but where the point sits far in one corner of the bounding box can severely challenge Newton's method. Adding a line search to the algorithm does not resolve the convergence issues. A common trick used by video games developers to address that is to rotate the bounding box and have it oriented such that it fits more closely to the element. The thought behind this is that the challenging points are away from the element and we should not have tried to map them in the first place. We propose a similar idea that we present in the next subsection.

VI.2.3 Bounding polyhedron

After applying the displacement field to the mesh, the four nodes supporting a face of the hexahedron do not necessarily lie in the same geometric plane any more and form a tetrahedron. The face of the distorted volume element lie somewhere inside that tetrahedron. Among the four triangular faces the tetrahedron consists of, we select the pair "above" the face of the volume element which will help us construct a tight bounding envelope for the element that we call bounding polyhedron. If the nodes are coplanar, then it doesn't matter what pair of triangles we keep. Figure VI.8 illustrates how the real element face is trapped between two pairs of triangular faces.

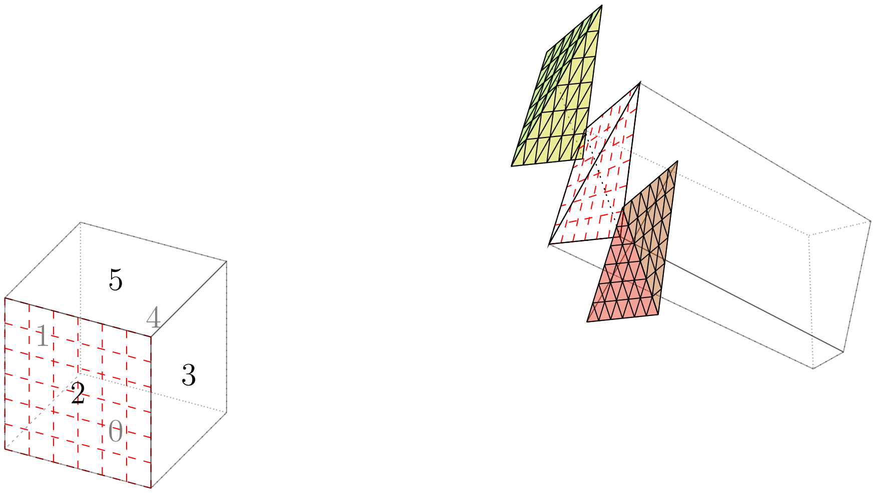


Figure VI.8: Mapping of one of the faces of the regular hexahedron. The real distorted face is contained in a tetrahedron which is flat if the four nodes supporting the face are coplanar. The triangular faces of the tetrahedron have been paired and moved aside for visualization purposes.

Thus, the bounding polyhedron, shown in Figure VI.9, consists of twelve oriented triangular faces (two triangles per face times six faces in the hexahedron). For each of the triangular faces, we take the vector pointing from the triangle centroid to the point and compute its dot product with the normal to the triangle. The sign tells us right away whether the point should be discarded. If all of the tests are passed, then the point lies within in the bounding polyhedron and we perform the mapping onto the reference element.

The computational cost associated to the bounding polyhedron test is minimal but it does a spectacular job at filtering the points by discarding the ones that were

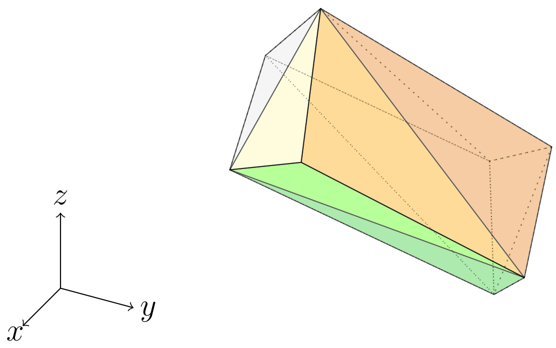


Figure VI.9: Bounding polyhedron.

causing trouble with the Newton iteration. In any case, it totally eliminates the non-convergence issues.

VI.2.4 Algorithm for the fine search

The main steps in the fine search process are summarized in Figure VI.10. Mapping global to local involve a nonlinear solve with Newton's iteration. The bounding box and bounding polyhedron tests ensure that this is done as a last resort, and, hence reduce the overall computational cost of the search.

Originally, we developed that algorithm a tool to interpolate solution fields on meshes. This is especially useful to post-process the finite element solution since, a priori, we cannot approximate the field at a point unless we know where it is located on the mesh. The first application of the search after it was implemented was data transfer between models.

Projecting back to the surface a point which is found within a boundary element is straightforward, and except for frictionless contact, it can be argued whether larger displacement increments, which would make surface points penetrate deeper into the foreign body, should be allowed or not. If allowed, then we eventually end up with the same initial problem of finding the minimum distance projection of a point onto a collection of faces, which, unless accelerated, is $O(1/h^2)$. We stress the fact that, although this algorithm pairs up particularly well with a node-to-face approach, it is not suitable when using face-to-face since it does not evaluate positive values of the gap width g_n, which would almost inevitably occur for a number of quadrature points. We note that, for the same reason, the approach presented in this chapter probably would not be the best choice if a model of heat transfer between bodies sensitive to g_n is included, because the width of the open gap would then need to be eventually computed.

Along the same lines of the acceleration that we propose here, the key in order to significantly reduce the cost of a single evaluation of g_n as h decreases consists in controlling the number of projection attempts onto master faces for a given slave node or face. The search is limited to immediate neighbors in the initial configuration (or the list of plausible candidates for interaction can be updated in the course the simulation if the need arises).

VI.3 Numerical results for the parallel search

We test the parallel scalability of the proposed search algorithm. We carry out strong and weak scaling studies up to 32 processors. We use a cylindrical mesh representing a fuel pellet with radius 4.095 mm and height 10 mm. The mesh contains 1,245,184 elements, and we search for 3,200,000 points, randomly distributed in the mesh. Having in mind a contact detection problem for pellet-clad interaction, we also measure the time needed, not only to interpolate these points in the mesh but also project then back onto the outer boundary.

Figure VI.11 shows how the runtime decreases as the number of processors is increased for a fixed problem size and Figure VI.12 gives the strong scaling efficiency for our search algorithm (i.e., sum parallel coarse and local fine search runtimes) as well as the interpolation and projection applications. Search and interpolation scale well.

For the weak scaling study, we use the same geometry but with different levels of resolution. The number of elements is doubled each time the number of processors is doubled, starting at 38,912 elements on one processors, up to 1,245,184 elements on 32 processors. Figure VI.13 indicates how the runtime varies with the number of processors for a problem size per processor that is kept to be roughly the same and Figure VI.14 plots the computed weak scaling efficiency. We observed that projection and interpolation operations scale extremely well. The initial search degrades significantly. However, in situation where the meshes between the physics components are fixed, this penalty occurs only once, at the initialization phase of the simulation.

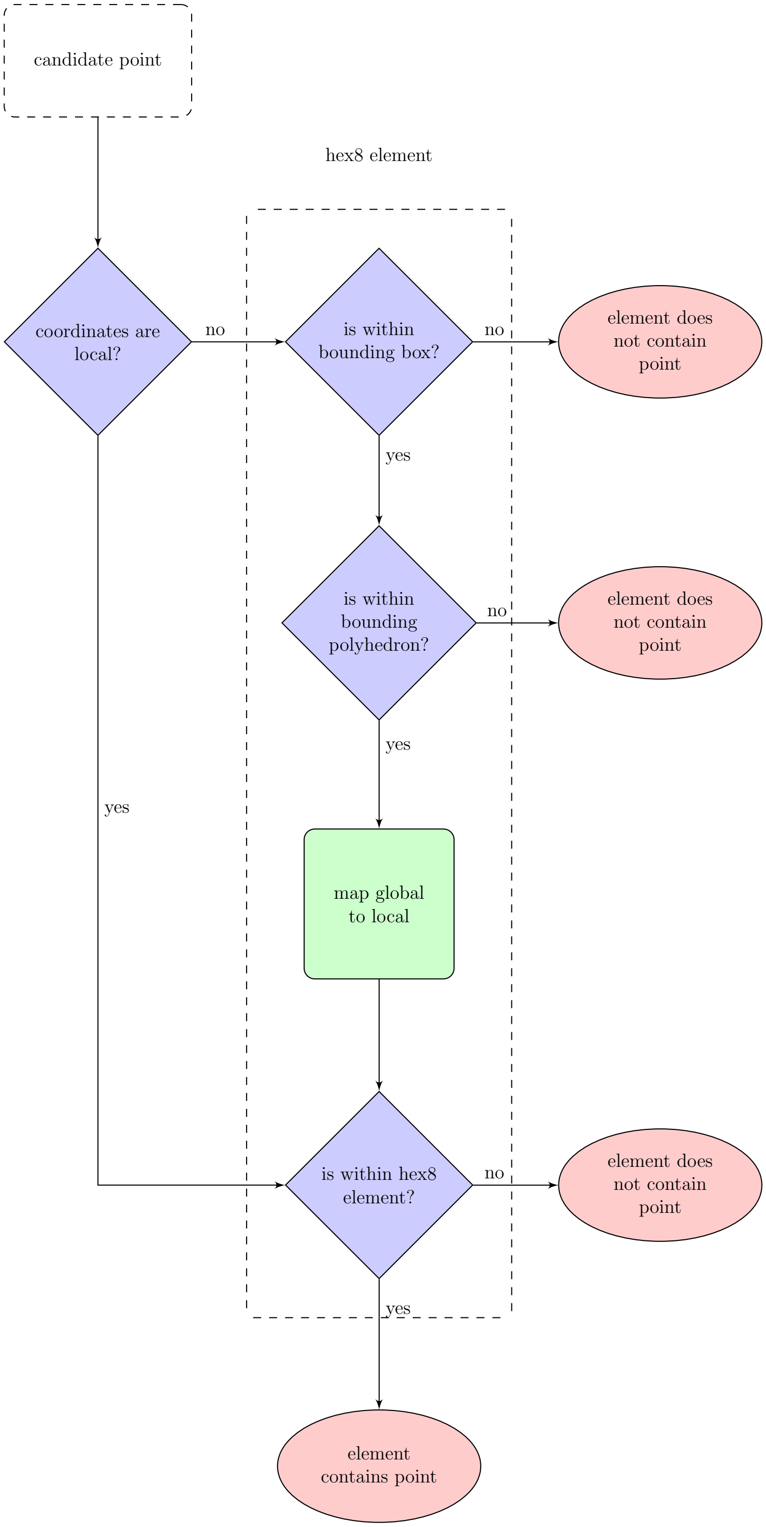


Figure VI.10: Flowchart representing the element level algorithm to determine whether a given hex8 element contains a point.

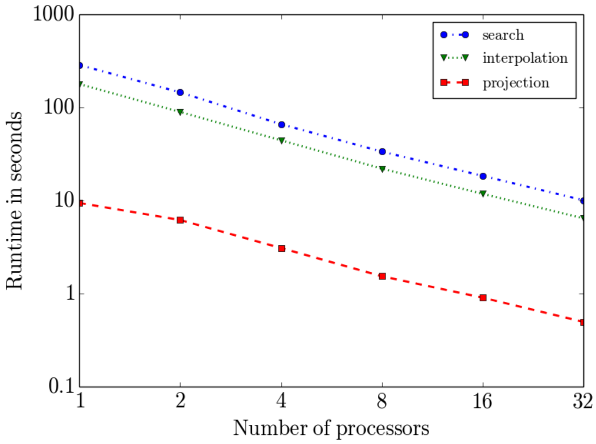


Figure VI.11: Strong scaling study: Run times

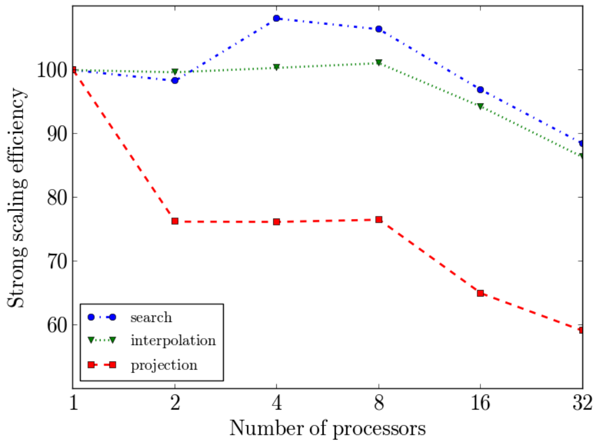


Figure VI.12: Strong scaling study: Efficiency

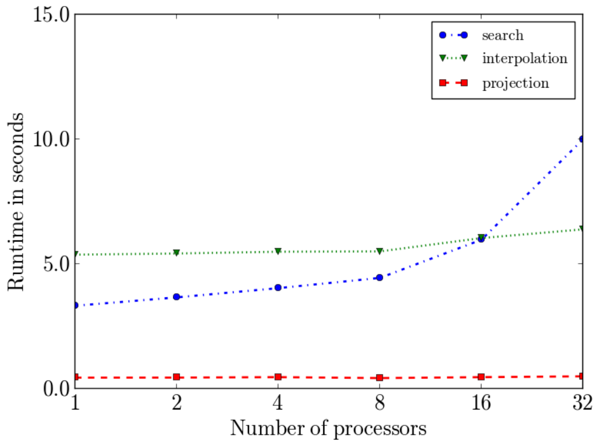


Figure VI.13: Weak scaling study: Run times

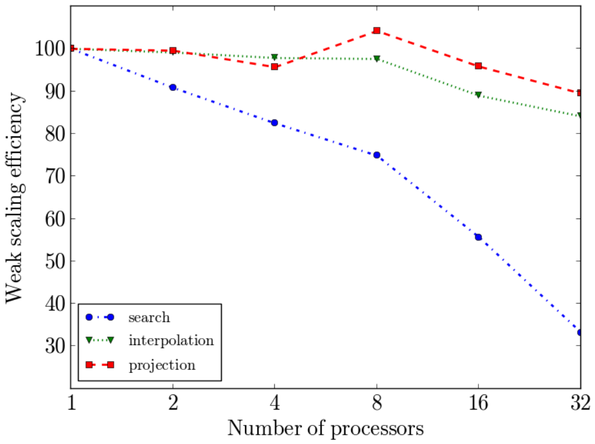


Figure VI.14: Weak scaling study: Efficiency

CHAPTER VII

CONSTRAINTS ENFORCEMENT

In this Chapter, we discuss different techniques to incorporate the contact constraints into the variational formulation of the mechanics problem. We give first an overview of the two methods most commonly used by codes that are able to handle contact, namely the Lagrange multipliers and the penalty methods [46]. Then, we present another option to enforce contact that consists in a direct elimination of the constraints.

Consider two bodies, labelled 1 and 2, coming in contact. Arbitrarily, body 1 is designated as master and body 2 as slave. For simplicity, we leave aside other physics components and focus on the mechanics of contact. We have seen in Chapter V that the vector of nodal displacements u is solution of the linear system:

$$\begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 & 0 \\ K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} & 0 & 0 \\ 0 & 0 & K_2^{\mathcal{A}\mathcal{A}} & K_2^{\mathcal{A}\mathcal{N}} \\ 0 & 0 & K_2^{\mathcal{N}\mathcal{A}} & K_2^{\mathcal{N}\mathcal{N}} \end{pmatrix} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} \\ f_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} \end{pmatrix} \tag{V.6}$$

where K_i are stiffness matrices and f_i right-hand-side load vectors for body i, $i \in \{1, 2\}$. The system is block-diagonal and coupling between bodies is given by the contact constraints which have been written in the form

$$Au_1^{\mathcal{M}} - Mu_2^{\mathcal{A}} = g_2^{\mathcal{A}}. \tag{V.13}$$

$\mathcal{A}$ is the set of "active" slave vertices located on the contact interface. Their displacement u_2^A is constrained by the master faces supported by vertices in $\mathcal{M}$. $\mathcal{N}$ contains all the other vertices. A is a coupling matrix , M a mass matrix, and g_2^A a vector giving a measure of the gap between the two bodies.

VII.1 Optimization techniques commonly used

VII.1.1 Penalty method

A classical optimization technique is the penalty method, which prescribes that we add the following surface traction term on contact interface $\Gamma^{C,A}$:

$$\boldsymbol{t} = \epsilon_n(\boldsymbol{g} \cdot \boldsymbol{n})\boldsymbol{n} + \epsilon_t(\boldsymbol{g} - (\boldsymbol{g} \cdot \boldsymbol{n})\boldsymbol{n}), \qquad \forall \boldsymbol{X} \in \Gamma^{C,A}, \qquad \epsilon_n, \epsilon_t > 0.$$

Here, $\boldsymbol{t}$ has been broken up into its normal and tangential components.

The penalty parameter in normal direction ϵ_n can be seen as the stiffness of an imaginary spring that gets compressed as a point penetrates into a foreign body and pushes back against it towards the surface. The point penetration depth depends upon ϵ_n and the constraint equation is only fulfilled in the limit as $\epsilon_n \to \infty$. Conversely, $\epsilon_n = 0$ represents the unconstrained problem.

Its tangential counterpart, ϵ_t, is holding back the slip motion of the point in the contact tangent plane. We distinguish to limiting cases: (i) $\epsilon_t \to \infty$ which means that the point sticks to the other body and is unable to move in a tangential direction. (ii) $\epsilon_t = 0$ that corresponds to the total absence of friction forces, when the point is free to slide in the tangential plane of the contact area.

For simplicity, we assume either an isotropic penalty $\epsilon_n = \epsilon_t = \epsilon$ (stick) or $\epsilon_t = 0$ and $\epsilon_n = \epsilon$ (slip).

$$
\begin{pmatrix}
K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 & 0 \\
K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} + \epsilon A^T A & -\epsilon A^T M & 0 \\
0 & -\epsilon A & K_2^{\mathcal{A}\mathcal{A}} + \epsilon M & K_2^{\mathcal{A}\mathcal{N}} \\
0 & 0 & K_2^{\mathcal{N}\mathcal{A}} & K_2^{\mathcal{A}\mathcal{A}}
\end{pmatrix}
\begin{pmatrix}
u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}}
\end{pmatrix}
=
\begin{pmatrix}
f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} + \epsilon A^T g_2^{\mathcal{A}} \\ f_2^{\mathcal{A}} - \epsilon g_2^{\mathcal{A}} \\ f_2^{\mathcal{N}}
\end{pmatrix}
\qquad \text{(VII.1)}
$$

We can derive a symmetric version of (VII.1) if write the constraints given in Equation (V.13) under the form $Au_1^{\mathcal{M}} - Mu_2^{\mathcal{A}} = d_2^{\mathcal{A}}$, with $C = M^{-1}A$ and $d_2^{\mathcal{A}} = M^{-1}g_2^{\mathcal{A}}$,

$$
\begin{pmatrix}
K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 & 0 \\
K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} + \epsilon C^T C & -\epsilon C^T & 0 \\
0 & -\epsilon C & K_2^{\mathcal{A}\mathcal{A}} + \epsilon & K_2^{\mathcal{A}\mathcal{N}} \\
0 & 0 & K_2^{\mathcal{N}\mathcal{A}} & K_2^{\mathcal{A}\mathcal{A}}
\end{pmatrix}
\begin{pmatrix}
u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}}
\end{pmatrix}
=
\begin{pmatrix}
f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} - \epsilon C^T d_2^{\mathcal{A}} \\ f_2^{\mathcal{A}} + \epsilon d_2^{\mathcal{A}} \\ f_2^{\mathcal{N}}
\end{pmatrix}
\qquad \text{(VII.2)}
$$

An obvious advantage of the penalty method over the Lagrange multipliers strategy is that it does not introduce any additional equations. Unfortunately, it is also well-known that the method suffers from ill-conditioning that worsens as the penalty values increase [7], while, as stated before, constraints are satisfied exactly only in the limit of infinite penalty values.

VII.1.2 Lagrange multipliers

Another approach to include the contact constraints into the variational problem consists in introducing an additional variable, namely the Lagrange multiplier $\boldsymbol{\lambda}$, and interpreting it as the reaction forces exerted by the cladding onto the pellets when contact becomes active. Looking back at the mechanics in the problem formulation we gave previously, cf. subsection 2.3, and assuming, say, no-slip conditions, we prescribe the following on the contact interface:

$$\left.\begin{aligned} &\boldsymbol{t}|_{\partial\Omega_2} = \boldsymbol{\lambda} \\ &\boldsymbol{t}|_{\partial\Omega_1} = -\boldsymbol{\lambda} \\ &\boldsymbol{g} = \boldsymbol{0} \end{aligned}\right\}, \qquad \forall \boldsymbol{X} \in \Gamma^{C,A}.$$

The first two equations enforce the action-reaction principle and we treat them as additional boundary conditions. The third one describes the non-penetration condition and actually involves supplementary equations in the variational form of the problem.

$$\begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 & 0 & 0 \\ K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} & 0 & 0 & A^T \\ 0 & 0 & K_2^{\mathcal{A}\mathcal{A}} & K_2^{\mathcal{A}\mathcal{N}} & -M^T \\ 0 & 0 & K_2^{\mathcal{N}\mathcal{A}} & K_2^{\mathcal{N}\mathcal{N}} & 0 \\ 0 & A & -M & 0 & 0 \end{pmatrix} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \\ \lambda_2^{\mathcal{A}} \end{pmatrix} = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} \\ f_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} \\ g_2^{\mathcal{A}} \end{pmatrix} \tag{VII.3}$$

This turns our problem into a saddle point problem and has several implications that make the solution process somewhat more laborious. First, not only has the size of the system we want to solve increased, but it also changes at every active set iteration as $\mathcal{A}$ is updated. Second, the iterative solution of such saddle problems often performs poorly without proper preconditioning [5].

VII.2 A direct approach to enforce constraints

We have reviewed two optimization methods that can be used for contact problems and have discussed briefly their respective advantages and drawbacks. Nevertheless, the constraints equation may also be enforced directly when contact occurs. In this section, we first present the direct elimination procedure, then demonstrate that it produces the same solution as the Lagrange multiplier method, and finally give details on its implementation.

VII.2.1 Direct elimination

Assume no-slip conditions (i.e., infinite friction coefficient) on the contact interface $\Gamma^{C,A}$. We may express the displacements $\boldsymbol{u}|_{\Omega_2}$ on the slave side of $\Gamma^{C,A}$ as a function of the displacements on the master side $\boldsymbol{u}|_{\Omega_1}$, i.e.,

$$\forall \boldsymbol{X} \in \Gamma^{C,A}, \qquad \boldsymbol{g} = \boldsymbol{0} \qquad \Rightarrow \qquad \boldsymbol{u}|_{\partial\Omega_2} = \boldsymbol{u}|_{\partial\Omega_1} + \boldsymbol{X}|_{\partial\Omega_1} - \boldsymbol{X}|_{\partial\Omega_2}. \tag{VII.4}$$

As a result, we should be able to eliminate all degrees of freedoms associated with the slave nodal displacements, hence reducing the number of unknowns in the problem. In addition, we would like to perform the elimination in such a manner that the symmetry and positive definiteness of the mechanics discretization is not altered [46]. This gives a significant advantage over the two methods we presented previously since it allows us to keep on using classical, well-established preconditioners.

We express the displacements of the slave nodes $u_2^{\mathcal{A}}$ as linear combination of the master nodes displacements $u_1^{\mathcal{M}}$

$$u_2^{\mathcal{A}} = C u_1^{\mathcal{M}} + d_2^{\mathcal{A}}. \tag{VII.5}$$

If we consider node-to-face contact, matrix C has four non-zero entries by line corresponding to the values of the shape functions at the point of contact on the master quadrilateral face, and vector $d_2^{\mathcal{A}}$ is the displacement/shift bringing the slave node from its original position to the point of contact. If, instead, we consider face-to-face contact, multiplying some vector by the matrix C and adding the vector d to another vector involve inverting the mass matrix M. Indeed, Equation (V.6) yields

$C = M^{-1}A$ and $d_2^{\mathcal{A}} = -M^{-1}g_2^{\mathcal{A}}$ (refer to Chapter V for more detail on the different types of constraints).

Anyway, $u_2^{\mathcal{A}}$ entries are not degrees of freedom of our problem any more and the vector of unknown nodal displacements can be reduced to $(u_1^{\mathcal{N}}, u_1^{\mathcal{M}}, u_2^{\mathcal{N}})^T$. We introduce the matrix Q such that

$$\begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix} = \underbrace{\begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & C & 0 \\ 0 & 0 & I \end{pmatrix}}_{Q} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ d_2^{\mathcal{A}} \\ 0 \end{pmatrix} = \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ Cu_1^{\mathcal{M}} + d_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix}, \qquad \text{(VII.6)}$$

and we plug it into Equation (V.6), which yields

$$\underbrace{\begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 & 0 \\ K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} & 0 & 0 \\ 0 & 0 & K_2^{\mathcal{A}\mathcal{A}} & K_2^{\mathcal{A}\mathcal{N}} \\ 0 & 0 & K_2^{\mathcal{N}\mathcal{A}} & K_2^{\mathcal{N}\mathcal{N}} \end{pmatrix}}_{K} Q \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} \\ f_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} \end{pmatrix} - K \begin{pmatrix} 0 \\ 0 \\ d_2^{\mathcal{A}} \\ 0 \end{pmatrix}. \qquad \text{(VII.7)}$$

As is often the case with mechanics, we like to keep matrices symmetric, so we multiply both sides of the above equation from the left by Q^T and obtain

$$Q^T K Q \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix} = Q^T \left(\begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} \\ f_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} \end{pmatrix} - K \begin{pmatrix} 0 \\ 0 \\ d_2^{\mathcal{A}} \\ 0 \end{pmatrix} \right). \qquad \text{(VII.8)}$$

We develop the above equation and derive the reduced form of the system

$$\begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 \\ K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} + C^T K_2^{\mathcal{A}\mathcal{A}} C & C^T K_2^{\mathcal{A}\mathcal{N}} \\ 0 & K_2^{\mathcal{N}\mathcal{A}} C & K_2^{NN} \end{pmatrix} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} + C^T f_2^{\mathcal{A}} - C^T K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} - K_2^{\mathcal{N}\mathcal{A}} d_2^{\mathcal{A}} \end{pmatrix}, \tag{VII.9}$$

or $K^* u^* = f^*$.

VII.2.2 Quick proof that both Lagrange multipliers method and direct elimination lead to the same solution

Clearly, the first row of Equations (VII.3) and (VII.9) are identical. Let us invert the relation in fifth row of Equation (VII.3). It yields a direct expression of the slave nodal displacements $u_2^{\mathcal{A}}$ as a linear combination of the master degrees of freedom $u_1^{\mathcal{M}}$

$$u_2^{\mathcal{A}} = M^{-1}\left(A u_1^{\mathcal{M}} + g_2^{\mathcal{A}}\right) \equiv C u_1^{\mathcal{M}} + d_2^{\mathcal{A}}.$$

When substituted into the fourth row of of Equation (VII.3), it produces

$$K_2^{\mathcal{A}\mathcal{N}} C u_1^{\mathcal{M}} + K_2^{\mathcal{N}\mathcal{N}} u_2^{\mathcal{N}} = f_2^{\mathcal{N}} - K_2^{\mathcal{N}\mathcal{A}} d_2^{\mathcal{A}},$$

which is exactly the third row of (VII.9). When used in the third row of (VII.3), $u_2^{\mathcal{A}} = C u_1^{\mathcal{M}} + d_2^{\mathcal{A}}$ gives

$$K_2^{\mathcal{A}\mathcal{A}}\left(C u_1^{\mathcal{M}} + d_2^{\mathcal{A}}\right) + K_2^{\mathcal{A}\mathcal{N}} u_2^{\mathcal{N}} - M^T \lambda_2^{\mathcal{A}} = f_2^{\mathcal{A}},$$

from which the Lagrange multipliers values $\lambda_2^{\mathcal{A}}$ can be obtained

$$\lambda_2^{\mathcal{A}} = M^{-T}\left(K_2^{\mathcal{A}\mathcal{A}} C u_1^{\mathcal{M}} + K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}} + K_2^{\mathcal{A}\mathcal{N}} u_2^{\mathcal{N}} - f_2^{\mathcal{A}}\right).$$

We replace into the second row of (VII.3)

$$K_1^{\mathcal{M}\mathcal{N}} u_1^{\mathcal{M}} + K_1^{\mathcal{M}\mathcal{M}} u_1^{\mathcal{M}} + A^T M^{-T} \left(K_2^{\mathcal{A}\mathcal{A}} C u_1^{\mathcal{M}} + K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}} + K_2^{\mathcal{A}\mathcal{N}} u_2^{\mathcal{N}} - f_2^{\mathcal{A}} \right) = f_1^{\mathcal{M}}.$$

We recognize that $A^T M^{-T} = (M^{-1}A)^T = C^T$ and write

$$K_1^{\mathcal{M}\mathcal{N}} u_1^{\mathcal{M}} + K_1^{\mathcal{M}\mathcal{M}} u_1^{\mathcal{M}} + C^T K_2^{\mathcal{A}\mathcal{A}} C u_1^{\mathcal{M}} + C^T K_2^{\mathcal{A}\mathcal{N}} u_2^{\mathcal{N}} = f_1^{\mathcal{M}} + C^T f_2^{\mathcal{A}} - C^T K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}},$$

which we identify as the second row of (VII.9). The proof is compete.

VII.2.3 Implementation of the direct elimination procedure

Here, we describe the implementation of an elegant and efficient procedure for the direct elimination of constantly evolving constraints into a system of equations, which was suggested by Rahul Sampath from the AMP group. It has not been published yet.

Algorithm 7 Modified PCG algorithm

```
$f \leftarrow f - Kd$
$u_0^s \leftarrow C u_0^m$
$r_0 \leftarrow f - K u_0$
$r_0^m \leftarrow r_0^m + C^T r_0^s$
$r_0^s \leftarrow 0$
$z_0 \leftarrow M^{-1} r_0$
$z_0^s \leftarrow C z_0^m$
$p_0 \leftarrow z_0$
for $j \leftarrow 0, 1, 2, \ldots$ do
    $w_j \leftarrow K p_j$
    $w_j^m \leftarrow w_j^m + C^T w_j^s$
    $w_j^s \leftarrow 0$
    $\alpha_j \leftarrow \frac{r_j^T z_j}{p_j^T w_j}$
    $u_{j+1} \leftarrow u_j + \alpha_j p_j$
    $r_{j+1} \leftarrow r_j - \alpha_j w_j$
    if $r_{j+1}$ “sufficiently small” then break end if
    $z_{j+1} \leftarrow M^{-1} r_{j+1}$
    $z_{j+1}^s \leftarrow C z_{j+1}^m$
    $\beta_j \leftarrow \frac{r_{j+1}^T z_{j+1}}{r_j^T z_j}$
    $p_{j+1} \leftarrow z_{j+1} + \beta_j p_j$
end for
```

Algorithm 7 gives the pseudocode to solve the contact problem $K^* u^* = f^*$. It uses a modified version of preconditioned conjugate gradient method, which differs from the standard implementation of PCG (refer to Algorithm 4) essentially in two places: (a) evaluation of the matrix-vector product, and (b) application of the preconditioner.

Any standard implementation of PCG will provide hooks for the user to implement their matrix-vector product and preconditioner. Let us describe how these need to be defined so that the procedure can be used with any implementation of PCG. Then, let us explain what needs also to be done before and after the Krylov iteration. Note that we can apply the same procedure with other iterative methods such as GMRES or BiCGStab.

VII.2.3.1 Reduced operator K^*

The iterative solution of the linear system (VII.9) only requires the matrix-vector product

$$\begin{pmatrix} r_1^{\mathcal{N}} \\ r_1^{\mathcal{M}} \\ r_2^{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 \\ K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} + C^T K_2^{\mathcal{A}\mathcal{A}} C & C^T K_2^{\mathcal{A}\mathcal{N}} \\ 0 & K_2^{\mathcal{N}\mathcal{A}} C & K_2^{NN} \end{pmatrix} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix}$$

but the reduced matrix K^* does not need to be constructed. Below is the procedure to evaluate $r^* = K^* u^*$ at each iteration. For simplicity[1], we have dropped the body indices.

1) copy master to slave $u^{\mathcal{A}} = Cu^{\mathcal{M}}$

   i.e., $u = \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ Cu_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix}$

[1] Making a distinction between the two bodies is useful to emphasize the uncoupled nature of Equation (V.6) if no further constraints from contact are added (i.e., apart from (V.13)), but $u_1^{\mathcal{N}}$ and $u_2^{\mathcal{N}}$ are regarded as being essentially the same thing for all the constraints enforcement techniques we present in this chapter. They represent degrees of freedom/nodes that are neither in $\mathcal{M}$ nor in $\mathcal{A}$, so we may as well write $u = (u^{\mathcal{N}}, u^{\mathcal{M}}, u^{\mathcal{A}})^T$ and $u^* = (u^{\mathcal{N}}, u^{\mathcal{M}})^T$. Extra body indices complicate unnecessarily the notation in this section.

2) apply mechanics operator $r = Ku$

$$\text{i.e., } r = \begin{pmatrix} K_1^{\mathcal{NN}} u_1^{\mathcal{N}} + K_1^{\mathcal{NM}} u_1^{\mathcal{M}} \\ K_1^{\mathcal{MN}} u_1^{\mathcal{N}} + K_1^{\mathcal{MM}} u_1^{\mathcal{M}} \\ K_2^{\mathcal{AA}} C u_1^{\mathcal{M}} + K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ K_2^{\mathcal{NA}} C u_1^{\mathcal{M}} + K_2^{\mathcal{NN}} u_2^{\mathcal{N}} \end{pmatrix}$$

3) add slave to master $r^{\mathcal{M}} = r^{\mathcal{M}} + C^T r^{\mathcal{A}}$

$$\text{i.e., } r = \begin{pmatrix} K_1^{\mathcal{NN}} u_1^{\mathcal{N}} + K_1^{\mathcal{NM}} u_1^{\mathcal{M}} \\ K_1^{\mathcal{MN}} u_1^{\mathcal{N}} + K_1^{\mathcal{MM}} u_1^{\mathcal{M}} + C^T K_2^{\mathcal{AA}} C u_1^{\mathcal{M}} + C^T K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ K_2^{\mathcal{AA}} C u_1^{\mathcal{M}} + K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ K_2^{\mathcal{NA}} C u_1^{\mathcal{M}} + K_2^{\mathcal{NN}} u_2^{\mathcal{N}} \end{pmatrix}$$

4) set slave to zero $r^{\mathcal{A}} = 0$

$$\text{i.e., } r = \begin{pmatrix} K_1^{\mathcal{NN}} u_1^{\mathcal{N}} + K_1^{\mathcal{NM}} u_1^{\mathcal{M}} \\ K_1^{\mathcal{MN}} u_1^{\mathcal{N}} + (K_1^{\mathcal{MM}} + C^T K_2^{\mathcal{AA}} C) u_1^{\mathcal{M}} + C^T K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ 0 \\ K_2^{\mathcal{NA}} C u_1^{\mathcal{M}} + K_2^{\mathcal{NN}} u_2^{\mathcal{N}} \end{pmatrix}$$

It is convenient to work with the full vectors u and r instead of their reduced version. For that purpose, we zero out the slave component of r in the last step, so that the norms and inner-product computed in the iterative solution are not affected by the unwanted $r_2^{\mathcal{A}}$ entries.

In practice, we define the action of the matrix on vectors to be the successive application of 2), 3), and 4) whereas 1) goes to the preconditioner. Another subroutine needs to be implemented as well for the enforcement of constraints, namely

5) add shift to slave $u^{\mathcal{A}} = u^{\mathcal{A}} + d^{\mathcal{A}}$

$$\text{i.e., } u = \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} + d_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix}$$

2) is provided by the regular mechanics operator on the two bodies. 1), 3), 4), and 5) are implemented in the contact operator.

VII.2.3.2 Reduced right-hand-side f^*

In order to form the reduced version of the right-hand side f^*, we add an extra pre-processing step where we do

(i) $f = f - Kd$

or $v = 0$, $v^{\mathcal{A}} = v^{\mathcal{A}} + d^{\mathcal{A}}$, $w = Kv$, and $f = f - w$ where v and w are two auxiliary vectors.

$$\text{i.e., } f = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} \\ f_2^{\mathcal{A}} - K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} - K_2^{\mathcal{N}\mathcal{A}} d_2^{\mathcal{A}} \end{pmatrix}$$

(ii) $f^{\mathcal{M}} = f^{\mathcal{M}} + C^T f^{\mathcal{S}}$

$$\text{i.e., } f = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} + C^T f_2^{\mathcal{A}} - C^T K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}} \\ f_2^{\mathcal{A}} - K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} - K_2^{\mathcal{N}\mathcal{A}} d_2^{\mathcal{A}} \end{pmatrix}$$

(iii) $f^{\mathcal{S}} = 0$

$$\text{i.e., } f = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} + C^T f_2^{\mathcal{A}} - C^T K_2^{\mathcal{A}\mathcal{A}} d_2^{\mathcal{A}} \\ 0 \\ f_2^{\mathcal{N}} - K_2^{\mathcal{N}\mathcal{A}} d_2^{\mathcal{A}} \end{pmatrix}$$

In practice, in order to have the initial residual r_0 properly computed in the Krylov subspace method, we also need to do.

(iv) $u_0^{\mathcal{S}} = C u_0^{\mathcal{M}}$

$$\text{i.e., } u_0 = \begin{pmatrix} {u_0}_1^{\mathcal{N}} \\ {u_0}_1^{\mathcal{M}} \\ C{u_0}_1^{\mathcal{M}} \\ {u_0}_2^{\mathcal{N}} \end{pmatrix}$$

And here we understand why body indices are not always handy to work with...

$$r_0 = f - \text{MatVec}(u_0),\ r_0^{\mathcal{M}} = r_0^{\mathcal{M}} + C^T r_0^{\mathcal{A}},\ \text{and } r_0^{\mathcal{A}} = 0 \text{ i.e., } r_0 = \begin{pmatrix} {r_0}_1^{\mathcal{N}} \\ {r_0}_1^{\mathcal{M}} \\ C{u_0}_1^{\mathcal{M}} \\ {r_0}_2^{\mathcal{N}} \end{pmatrix}$$

VII.2.3.3 Solution u

Similarly, we add a final post-processing step after iterative solution to recover u from u^*

(v) $u^{\mathcal{A}} = C u^{\mathcal{M}}$

$$\text{i.e., } u = \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ C u_1^{\mathcal{M}} \\ u_2^{\mathcal{N}} \end{pmatrix}$$

(vi) $u^{\mathcal{A}} = u^{\mathcal{A}} + d^{\mathcal{A}}$

$$\text{i.e., } u = \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ Cu_1^{\mathcal{M}} + d_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix}$$

CHAPTER VIII

NUMERICAL RESULTS

In this Chapter, we perform numerical tests for contact problems using the Lagrangian-multipliers approach and the method based on a direct elimination of constraints. When using Lagrange multipliers, a fully coupled thermo-mechanical problem is solved. Demonstration for the direct elimination method is carried out by considering only the mechanics problem but its extension to a fully coupled problem is similar to that of the Lagrange multipliers technique.

VIII.1 Fully coupled results using Lagrange multipliers for contact

Here, we provide two numerical examples using Lagrange multipliers to enforce constraints at the contact interface. The first one consists in a single fuel pellet coming into contact with its protective cladding, treating the latter as a rigid body, while the second one introduces contact between multiple deformable bodies. We tackle the problem using modern numerics, in the manner of [34], with a Newton-based monolithic strategy to handle both nonlinearities (coming from the temperature-dependence of the fuel thermal conductivity for instance) and coupling between the various physics components (gap conductance sensitive to the clad-pellet distance, thermal expansion coefficient or Youngs modulus affected by temperature changes, etc.).

VIII.1.1 Single pellet with rigid cladding

We propose a first test problem to demonstrate how Lagrange multipliers can be used to enforce contact. Our model of the LWR nuclear fuel rod consists of a single cylindrical UO_2 pellet placed within the Zircaloy cladding tube, as depicted in Figure VIII.1. We assume a constant uniform heat source and postulate that the cladding behaves as a rigid undeformable body. The latter assumption turns the search for contact into a simple test: to determine whether the non-penetration condition is violated at some point on the outer surface of the pellet, we only have to compare its radial position on the displaced mesh against the clad radius, R^{clad}. Furthermore, we ignore the frictional forces between fuel and cladding.

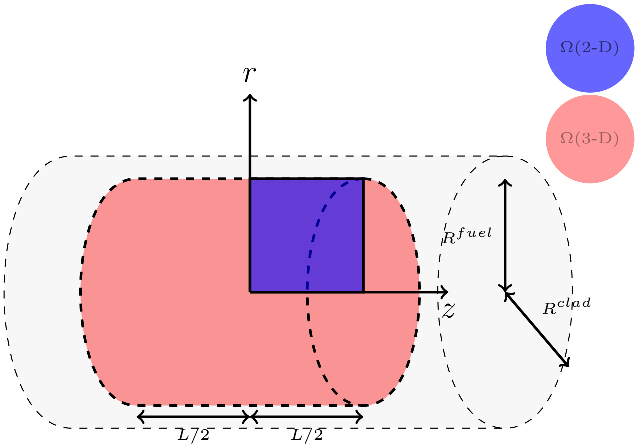


Figure VIII.1: Problem geometry for the single pellet test case.

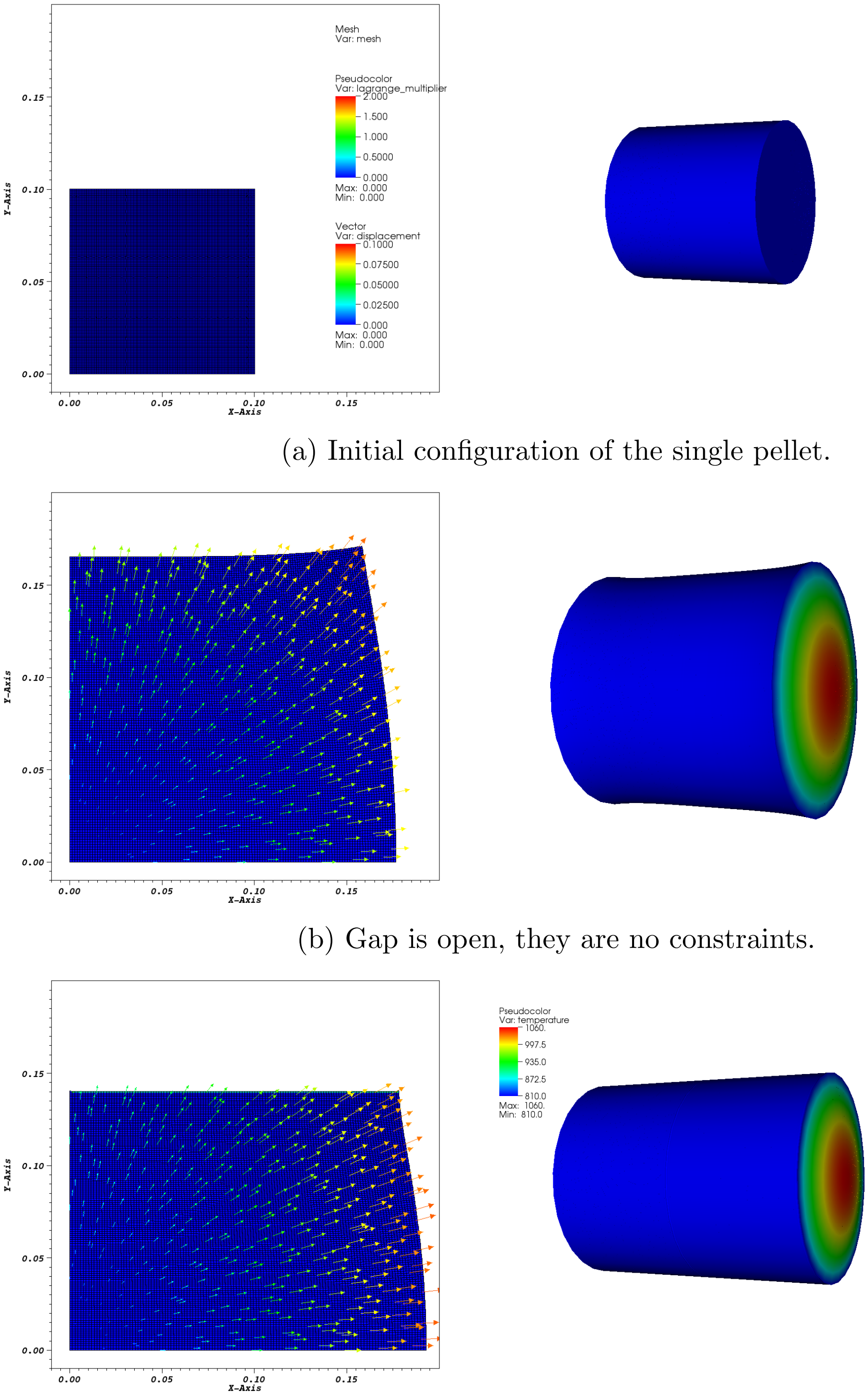
We consider both cases, no clad (gap is always open and constraints are inactive) and closed gap as shown on Figure VIII.2, respectively in Figure VIII.2b and Figure VIII.2c. Figure VIII.2a shows the initial mesh, before applying the solution displacement field. Although the problem does not use physical properties, the goal of this test case is to demonstrate the ability to constrain the expansion of the pellet. The deformations are exaggerated through the material mechanical properties. Lamé parameters[1] $\lambda = (E\nu)/((1+\nu)(1-2\nu))$ and $\mu = E/(2(1+\nu))$ are set equal

[1] A simple expression of the stress is given by $\sigma_{ij} = \lambda\delta_{ij}\epsilon_{kk} + 2\mu\epsilon_{ij}$.

to 1 and the coefficient of expansion α is used as a parameter to control the swelling cylindrical body.

Figure VIII.2b represents the case where no constraints are present. The solution Lagrange multiplier is zero over all the domain since there is no contact. The solution temperature presents a parabolic profile from the hotter centerline to the colder outer surface of the pellet and the resulting differential thermal expansion in the r-direction yields the famous hour-glass shape that we recognize on the displaced mesh. Boundary conditions are $u_z|_{r=0} = 0$ and $u_r|_{z=0} = 0$.

On Figure VIII.2c, however, contact will be active when $r + u_r \geq R^{clad}$. The swelling in the radial direction is now constrained and we notice the reaction forces exerted by the surrounding clad onto the outer surface of the fuel element. With a frictionless contact hypothesis, only the normal component of the surface traction is transmitted through the contact interface, the pellet is free to slip in the tangential direction and consequently expands further along the z-axis.



(a) Initial configuration of the single pellet.

(b) Gap is open, they are no constraints.

(c) Gap is closed, swelling is constrained in the radial direction.

Figure VIII.2: Numerical results with a single pellet. The color on the 2D slices (left) depicts the Lagrange multiplier and arrows show the displacement field. The 3D bodies give the material temperature.

VIII.1.2 Multiple pellets test case

We consider now a multiple-body problem where a number of Uranium dioxide pellets are stacked atop of each other. The surrounding protective cladding is still handled as a rigid body. We give the problem geometry in Figure VIII.3 and show some sample results in Figure VIII.4 for five pellets alternatively without and with obstacles in both the r and z directions. On the upper part of the latter figure, the coloration depicts the displacement field magnitude. We consider here the case were no cladding is present, i.e., the swelling is not restricted in the positive z direction as well as along the pellet radius. In the middle graph, we give the temperature distribution within the five pellets. As can be seen from the figure, we added an obstacle that restricts swelling along the z-axis. The bottom subfigure shows the axial Lagrange multipliers, accounting for reaction forces at contact interface between pellets. Here the cladding activates contact in the r direction as well.

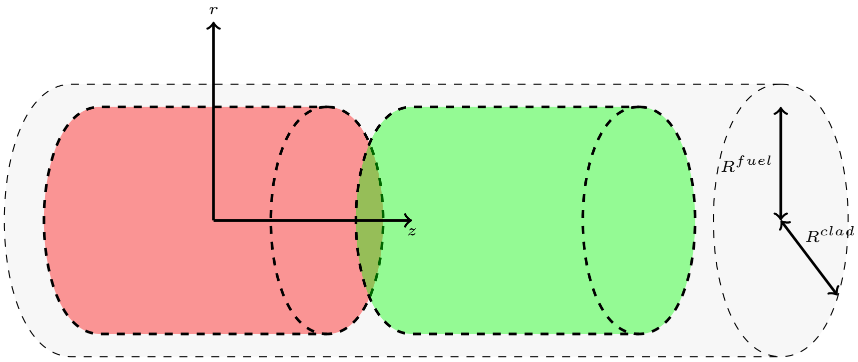


Figure VIII.3: Multibody contact problem geometry.

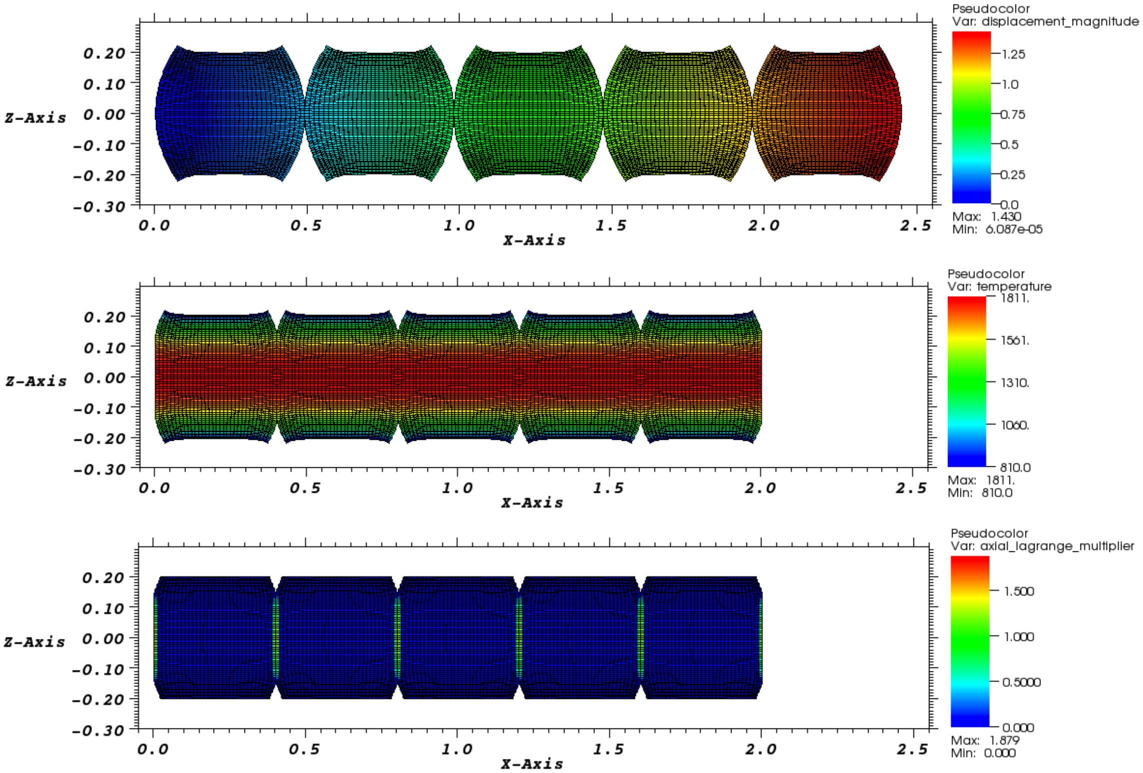


Figure VIII.4: Sample results with multiple pellets. The three figures correspond to three different scenarios, respectively, from top to bottom, with no restrictions to the expansion of the pellet stack, then adding an obstacle to the growth along the z axis, and finally setting constraints both along the z and r directions. The coloration depicts the displacement magnitude (top), temperature (middle), and axial compression forces (bottom). Note that the magnitude of the displacement field was magnified for visual purposes.

VIII.2 Contact with direct elimination of constraints

In Section VIII.1, we have been able to obtain a fully coupled solution of the coupled heat transport and thermo-mechanical contact for multiple pellets, but the cladding was always treated as a rigid body. Solving the problem with a cladding that deforms as well requires the ability to search for contact. Previously, the combined use of a power profile which does not vary along the direction axis of the pellet stack (z-direction) and matching meshes for the fuel pellets allowed us to avoid this difficulty. Indeed, surface vertices of two pellets stacked one on top the other would remain lined up throughout the entire computation. Hence, evaluating the size of the axial pellet-pellet gap could simply be done by comparing z-coordinates of nodes on either side, which has virtually no cost if we construct and store a map between them after the pellet meshes are initialized. Measuring the radial pellet-clad gap was also straightforward since it boil down to checking whether r exceeded the cladding inner radius (which was fixed). Unfortunately, we cannot assume that the projections of the pellet surface nodes onto the inner surface of the cladding will keep their position with respect to the clad faces after fission process starts heating up the fuel.

The contact detection algorithm described in Chapter VI was coded and used in a node-to-face contact operator which implements the direct elimination of constraints method (refer to Chapter VII). In this section, we first revisit two standard numerical examples of mechanical contact in order to understand better how the solution algorithms presented in Chapter V perform with infinite friction coefficient and demonstrate that our node-to-face contact operator is able to handle contact properly. Then, we show that the method can be extended to frictionless contact and perform numerical tests using multigrid as a preconditioner. Finally, we give a last example where we show that it can be applied to solve the pellet-cladding mechanical interaction problem.

VIII.2.1 Stacked cubes

In our first example, we consider the problem depicted in Figure VIII.5, inspired from [25, 26, 27]. The two bodies in their reference configuration are axis-aligned cubes with edge length 10 mm: the lower body, arbitrarily chosen to be the master, is given by $\Omega_m := (0, 0.01) \times (0, 0.01) \times (0, 0.01)$, and the slave body, stacked on top of it, by $\Omega_s := (0, 0.01) \times (0, 0.01) \times (0.01, 0.02)$. For the material parameters, we set Young's modulus E and Poisson's ratio ν to be 15 GPa and 0.2 on Ω_m; we use $E = 20$ GPa and $\nu = 0.4$ on Ω_s. The lower boundary of Ω_m ($z = 0$) and upper of Ω_s ($z = 0.02$) are clamped, so we impose $\boldsymbol{u} = \mathbf{0}$ on these. Following [27] [2], we apply load $\boldsymbol{p} = (-0.1, 0, -1)^T$ on Ω^m's right side ($x = 0.01$) and $(0.1, 0, -1)^T$ N/mm^2 on the left side of Ω_s ($x = 0$).

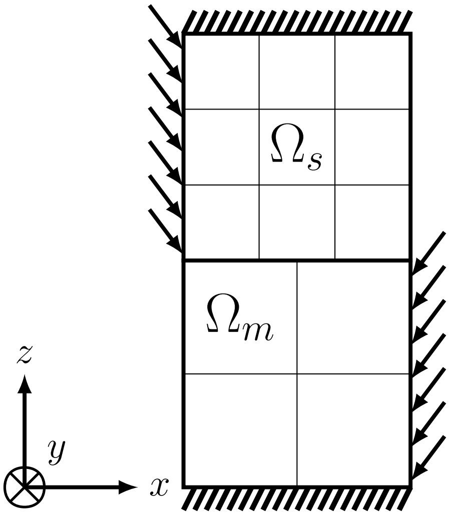


Figure VIII.5: Stacked cubes: Problem definition.

The example is inspired from [27] but it differs from the original problem in several ways that we would like to emphasize here. First, we consider a three-dimensional version of the problem. We can reasonably assume that if we take a 2D slice far enough from the front and back boundaries, say at $y = 0.005$, the solution should be

[2] For some reasons, although, to this point, Huëber et al. were following [25] to the letter in [26] and [27], they apply a load 100 times weaker. An attempt of explanation for that could be that [25] provide the load in daN/mm^2 = 10^7 Pa which would have been erroneously converted to 10^5 Pa. Since in the original article, bodies deformation are amplified on the plot of the solution but the scaling factor is not specified, it is hard to tell. Nevertheless, we have chosen to go with the latter because more data has been published and is available for comparisons.

close to the two-dimensional scenario. One obvious drawback with 3D computations is that it will be more delicate to perform a sequence of mesh refinements because the number of unknowns increases considerably faster. A more fundamental difference comes from frictional forces. Our node-to-face approach to contact assumes no slip between objects coming into contact (i.e., infinite friction coefficient), whereas the original problem opted for zero friction. A direct consequence is that we cannot expect to obtain numbers that match exactly with the ones in [27]. Nevertheless, the example still remains a good way for us to test our contact operator and the active set strategy we implemented.

For the finite element computation, we use trilinear finite element functions on hexahedra. For our triangulation on level 0, every body defines one hexahedron, so the possible contact part Γ^c of Ω^s consists of four vertices. Contact problems generally do not admit analytical solutions. In order to obtain error estimates, we must compute a reference solution, $\boldsymbol{u}_{ref}$, corresponding to a mesh which is as fine as possible.

Table VIII.1: Stacked cubes: Active set strategy for the stacked cubes problem presented in Figure VIII.5.

l	$card(\mathcal{S})$	K_l	$\lvert\mathcal{A}_k\rvert$							
1	9	3	9	6	3					
2	25	3	25	15	10					
3	81	4	81	43	36	27				
4	289	6	289	136	104	89	87	85		
5	1089	6	1089	514	398	365	332	330		
6	4225	8	4225	1951	1505	1319	1251	1242	1241	1240

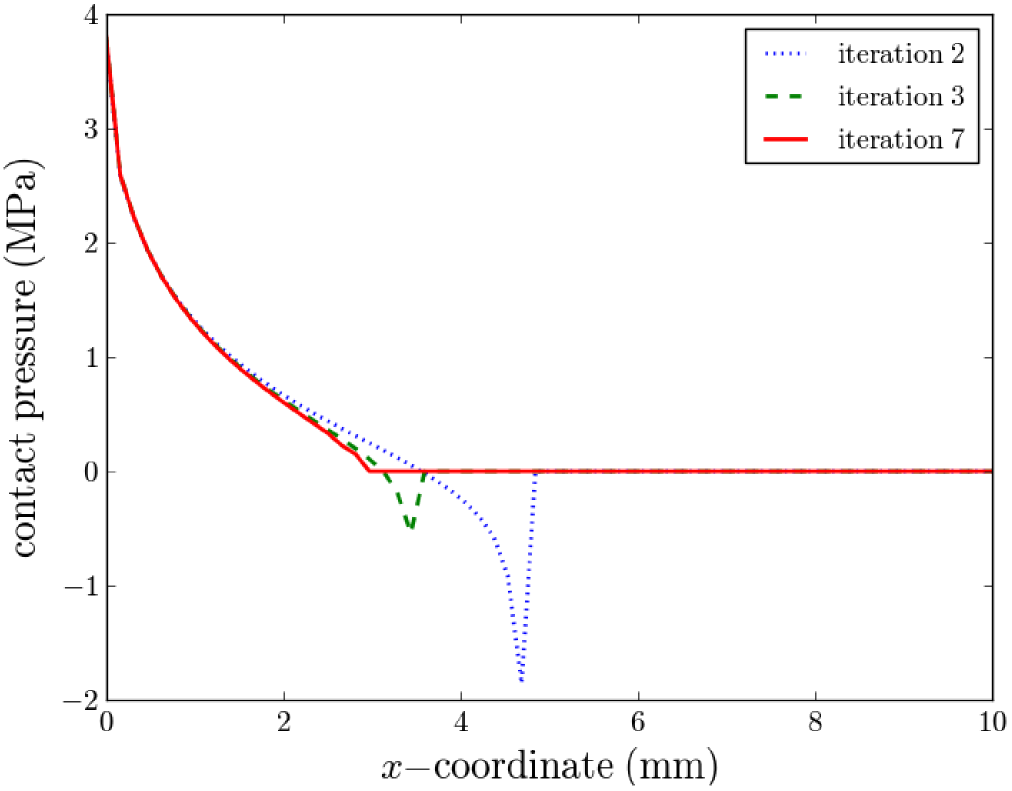


Figure VIII.6: Stacked cubes: Contact pressure along the center line at different iterations of the active set strategy on mesh refinement level #6.

On Figure VIII.6, we plotted the contact stresses in normal direction at iterations 2, 3, and 7, of the active set strategy on mesh refinement level #6. The data corresponds to the intersection of the contact area with the plane orthogonal to the y-axis that cuts the distorted domains into halves that are mirror images of each other, i.e. $\{(x, y, z) \in \Gamma^C \mid y = 0.005\}$. We refer to it as the center line of Γ^C, and we denote it $\mathcal{L}^C$. As one can see, $-\boldsymbol{t} \cdot \boldsymbol{n}$ gives some negative values on the active-set steps #2 and #3. Consequently, the associated nodes are removed from the active set in the next steps since we only allow compressive normal forces to be transmitted through the contact interface. The iterative procedure continue until the active set is unchanged. On step 7, $-\boldsymbol{t} \cdot \boldsymbol{n}$ is nonzero only in the part of Γ^C where the two bodies are actually in contact, and the values are all positive. Table VIII.1 reports the necessary iteration steps K_l of the active set strategy to solve the problem. As can be inferred from the data in Table VIII.1, it seems to depends linearly on the mesh refinement level l, when taking $\mathcal{A}_1 = \mathcal{S}$ and $\mathcal{I}_1 = \emptyset$. This is consistent with what Huëber et al. reported in [27]. $|\mathcal{A}_k|$ represents the number of active slave vertices at iteration k ($1 \leq k \leq K_l$) and $card(\mathcal{S})$ is the number of vertices in $\mathcal{S}$.

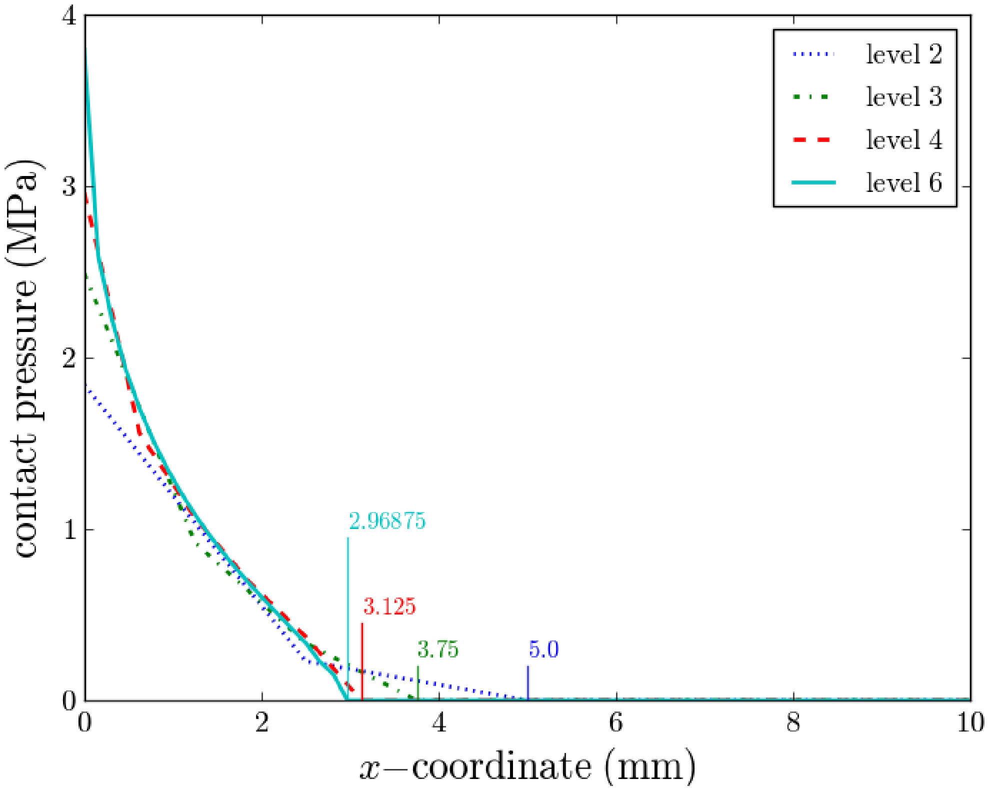


Figure VIII.7: Stacked cubes: Contact stresses in normal direction at several mesh refinement levels. The corresponding relative L_2-errors when taking solution on level 6 as reference solution are given in Table VIII.2. Loss-of-contact points are highlighted.

Table VIII.2: Stacked cubes: Relative L_2-error of the contact pressure with respect to a reference solution taken on refinement level 6 which has 524288 elements. Errors are computed along the center line of the contact area. The numerical convergence order are given.

level	# cells	$\frac{\|(\boldsymbol{t}\cdot\boldsymbol{n})_h-(\boldsymbol{t}\cdot\boldsymbol{n})_{ref}\|_{L_2(\mathcal{L}^C)}}{\|(\boldsymbol{t}\cdot\boldsymbol{n})_{ref}\|_{L_2(\mathcal{L}^C)}}$	
1	16	6.08026×10^{-1}	–
2	128	3.01121×10^{-1}	1.01
3	1024	1.63808×10^{-1}	0.89
4	8192	8.66598×10^{-2}	0.92
5	65536	(4.18504×10^{-2})	(1.05)

Figure VIII.7 shows the contact pressure at different refinement levels. We observe convergence for the points where slave and master lose contact. In Table VIII.2, we give the discretization errors of the contact pressure. Contact problems generally do not admit analytical solutions. In order to obtain error estimates, we used the finite element solution on level 6 as reference solution. On level 6, we have 524,288 elements equally divided between Ω_s and Ω_m. The mesh size h_{ref} for the reference solution satisfies $h_{ref} \leq 1/4h$ up to level 4. One may notice, we computed the discretization errors and convergence on level 5 and put values in parenthesis because a reference mesh size of $h_{ref} = 1/2h$ does not guarantee reliable numbers. As mentioned before, it is difficult to perform many mesh refinement series when computing in three dimensions. We measured a convergence rate of 0.92 at level 4, but he would be useful to perform a couple more uniform refinements to be able to assert that the convergence rates tend to 1.

The distorted cubes with the displacements magnitude $\|\boldsymbol{u}\|$ at first and last iterations of the active set strategy are presented on Figure VIII.8. Following [26], we magnified $1,000\times$ the displacements for visualization purposes, such that one can easily see the separation of the bodies on the right part of the contact zone as we observed numerically, i.e., $\boldsymbol{g} \cdot \boldsymbol{n} \geq 0$. Figure VIII.9 shows the effective von Mises stress[3] σ_{eff} and the forces $\boldsymbol{t}$ transmitted to the master at slave nodes on Γ^C. The data corresponds to a 2D slice orthogonal to y-axis and that intersects it at $z = 0.005$, such that $\boldsymbol{\mathcal{L}}^C$ actually follows the contact area on that slice. On Figure VIII.10, we plotted $\|\boldsymbol{u}\|$, σ_{eff}, and $-\boldsymbol{t} \cdot \boldsymbol{n}$ in 3D. Please note that we set a maximum on the color scale that cuts off the stress singularities in upper left corner of Ω_s, lower right of Ω_m, or at the left endpoint of the contact zone.

[3] The von Mises stress σ_{eff} is a measure of the stress level which is important in the analysis of plastic deformations. It is computed from the Cauchy stress tensor $\boldsymbol{\sigma} = (\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{yz}, \sigma_{zx}, \sigma_{xy})^T$ as

$$2\sigma_{\mathit{eff}}^2 = (\sigma_{xx} - \sigma_{yy})^2 + (\sigma_{yy} - \sigma_{zz})^2 + (\sigma_{zz} - \sigma_{xx})^2 + 6(\sigma_{yz}^2 + \sigma_{zx}^2 + \sigma_{xy}^2).$$

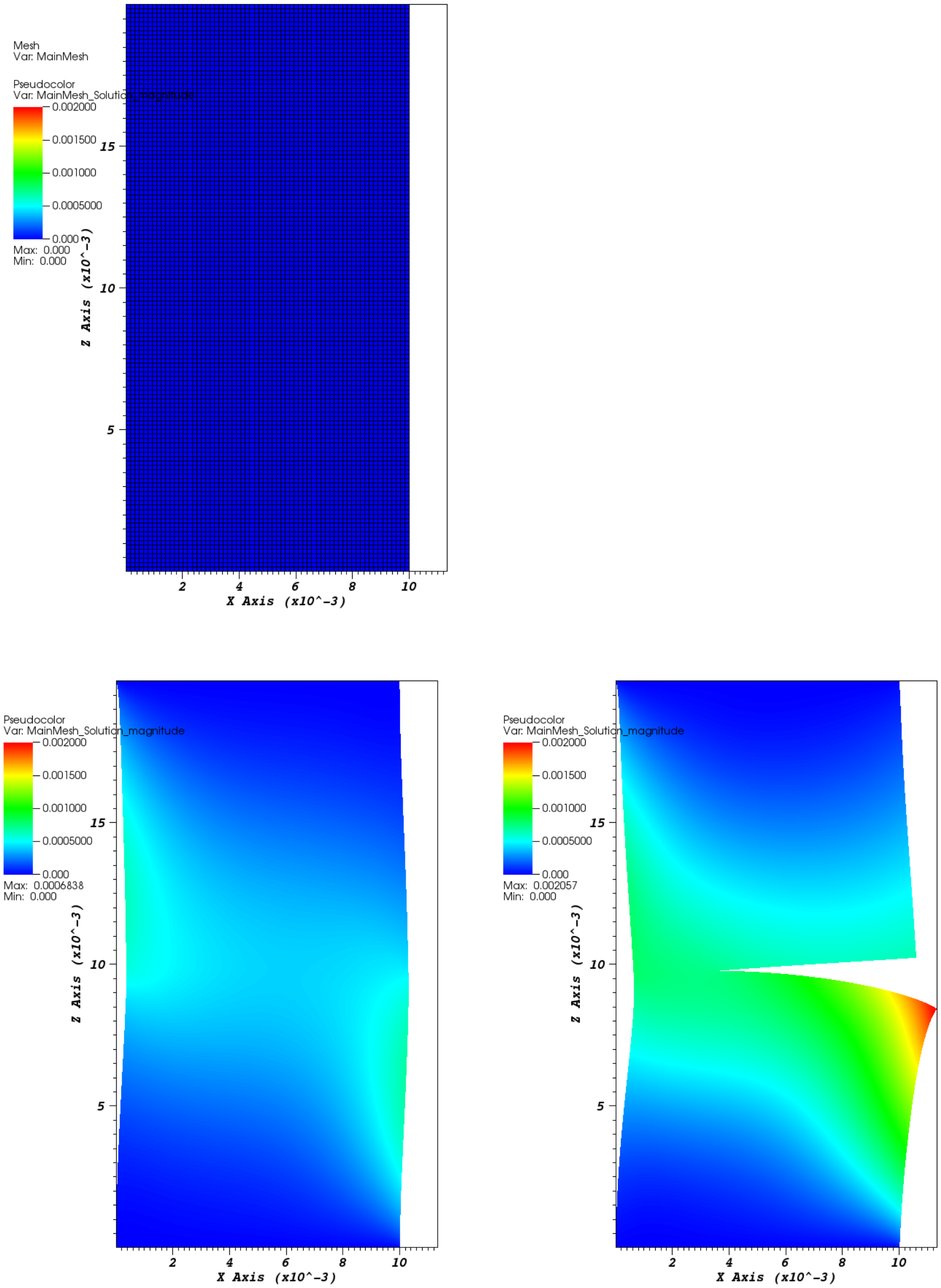


Figure VIII.8: Stacked cubes: Initial configuration on a central 2D slice (upper left) and distorted bodies with the magnitude of the displacements $\|\boldsymbol{u}\|$, at the first iteration of the active set strategy (lower left) and on the final solution (lower right). Distortion as been scaled by a factor 1,000 as in [26].

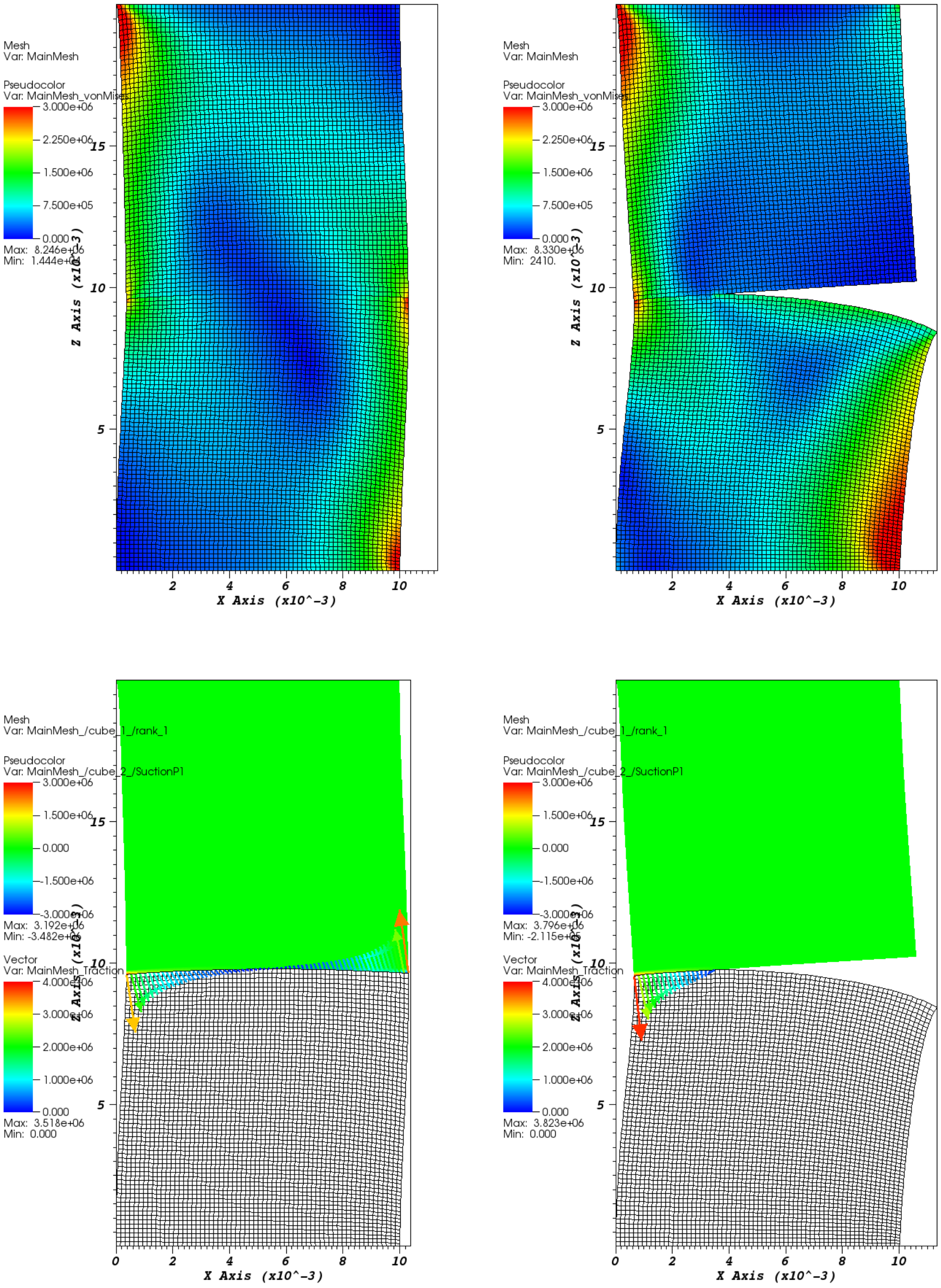


Figure VIII.9: Stacked cubes: Distorted bodies with the effective von Mises stress σ_{eff} (above) and with the surface traction $\boldsymbol{t}$ and contact pressure $-\boldsymbol{t}\cdot\boldsymbol{n}$ (below). Both the first step of the active set strategy (left) and the solution (right) are represented on mesh refinement level #6.

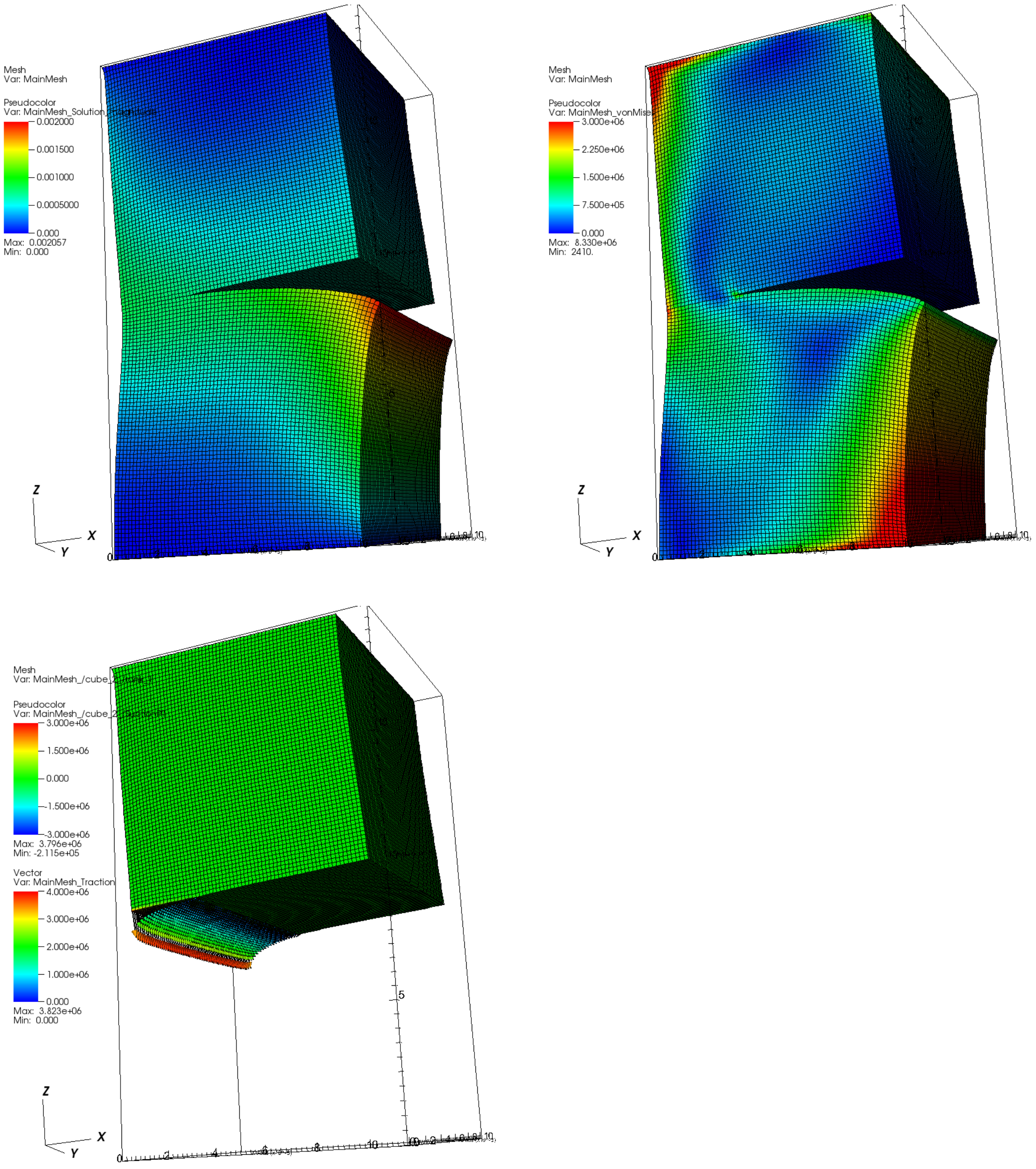


Figure VIII.10: Stacked cubes: Distorted bodies in 3D with the displacements magnitude $\|\boldsymbol{u}\|$ scaled by a factor 1,000 (upper left), the effective von Mises stress σ_{eff} (upper right), and the surface traction $\boldsymbol{t}$ on the master side of the contact interface (lower right).

VIII.2.2 Cylinder pressed onto a brick

Our second example is a Herztian contact problem of a linear elastic cylinder pressed against a hard brick. The contact stresses can be computed analytically. The problem geometry is defined in Figure VIII.11. The cylinder is sitting on top of the brick and it plays the role of the master, Ω_m. It has a height of 1 and its base is a circle of radius $r = 0.5$ in the xy-plane. Its axis is aligned with the z direction and the two endpoints are (0.5, 1, 0) and (0.5, 1, 1). The brick is given by $\Omega_s := (0, 1) \times (0, 0.5) \times (0, 1)$.

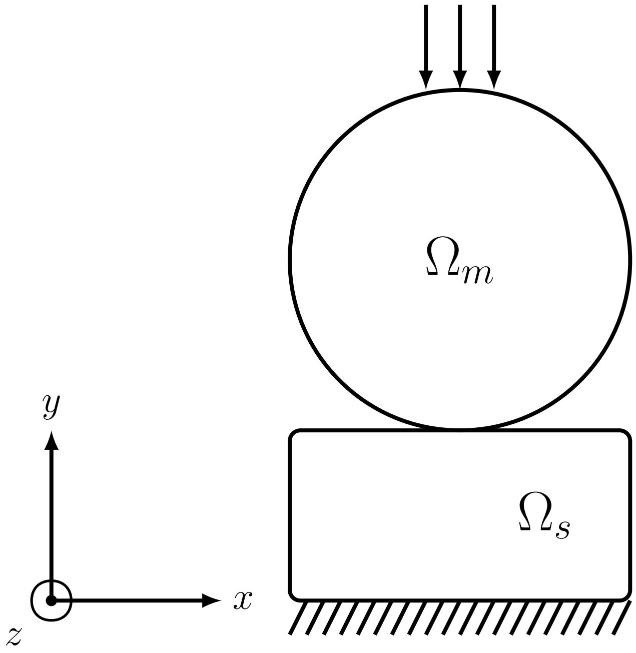


Figure VIII.11: Elastic cylinder pressed against a rigid planar surface: Problem definition.

If an elastic circle with radius r and material parameters E and ν is pressed by a single point load f on the top against a rigid plane, the analytical contact pressure for frictionless contact is given by

$$\begin{aligned}
&p_n(x) = \tfrac{2f}{\pi b^2}\sqrt{(b^2 - x^2)} && \text{for } |x| \leq b \\
&p_n(x) = 0 && \text{otherwise} \\
&b := 2\sqrt{\tfrac{fr(1-\nu^2)}{E\pi}}
\end{aligned} \tag{VIII.1}$$

where b represents the half-width of contact surface and x the distance to the center of the contact surface, where the pressure reaches its maximum value. Unfortunately, there is no analytical solution if friction is considered and the contact operator implementing the direct elimination approach implies infinite friction coefficient, but we can reasonably assume that, as long as the ratio b/R stays small, the differences between the two models will not be significant.

For our problem, instead of a rigid plane, we use a linear elastic brick with a relatively large Young's modulus. We apply homogeneous Dirichlet boundary conditions on its lower surface, i.e. $\boldsymbol{u}|_{y=0} = \mathbf{0}$, and we use $E = 1\text{GPa}$ and $\nu = 0.45$. The cylinder, assumed to be the master side (Ω_m), is pressed by a point load $\boldsymbol{f}|_{y=1.5} = (0, -100, 0)$. In practice though, to avoid a strong singularity in the upper part of the cylinder, the point load is replaced by a surface load as it was done in [12, 26]. We set the cylinder's material parameters to be $E = 7000\text{Pa}$ and $\nu = 0.3$.

Note that, for this problem, we cannot start with $\mathcal{A}_1 = \emptyset$ since Ω_m has no constraints in the y-direction so we would not have uniqueness of the solution due to rigid body motion. Instead, we set $\mathcal{A}_1 = \{p_m\}$, where p_m denotes the line of nodes along the z-axis which are touching the brick in the initial configuration.

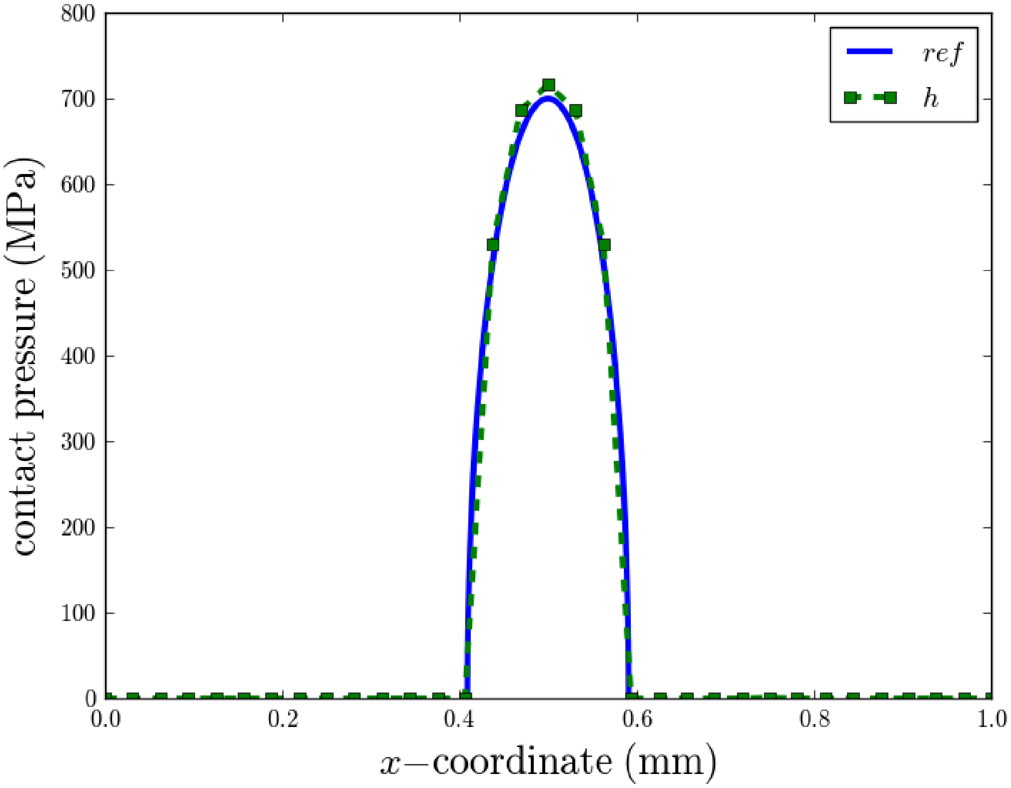


Figure VIII.12: Elastic cylinder: Contact pressure $p_n(x)$ along the center line of the brick perpendicular to the cylinder axis. Computed values using 49,152 elements and 10 loading steps are plotted as a green dashed line and the analytical solution is given by the blue solid line.

Using Equation (VIII.1), we calculate the analytical values for the maximal normal contact stresses of the Hertzian contact problem, $p_n(0) = 699.791$, and for the half width of the contact zone, $b = 0.0909728$. We compare them with 715.980 and 0.09375, obtained using 49,152 elements; this represents a relative error of about 2.3%. On Figure VIII.12 we plotted the computed normal contract stresses $p_n = -\boldsymbol{t} \cdot \boldsymbol{n}$ along the center line of the brick parallel to the x-axis next to the analytical solution. One can see that these are in good agreement.

In Figure VIII.13, we show the mesh in the initial configuration (upper right). We represented the displacement magnitude $\|\boldsymbol{u}\|$ (lower left) and the von Mises stresses σ_{eff} (lower right) on the distorted bodies. Limits on the color scales were changed for visualization purposes because of the singularities near the top of the cylinder, at places where the load f is applied. The contact area $\Gamma^{C,A}$ is a strip $x \in [-b, b]$ and cylinder's displacement approaches zero at the point $x = 0$ of maximum pressure because Young's modulus of the brick is two order of magnitude higher for the brick. Figure VIII.14 represents the stress component σ_{yy} and the contact forces in 3D.

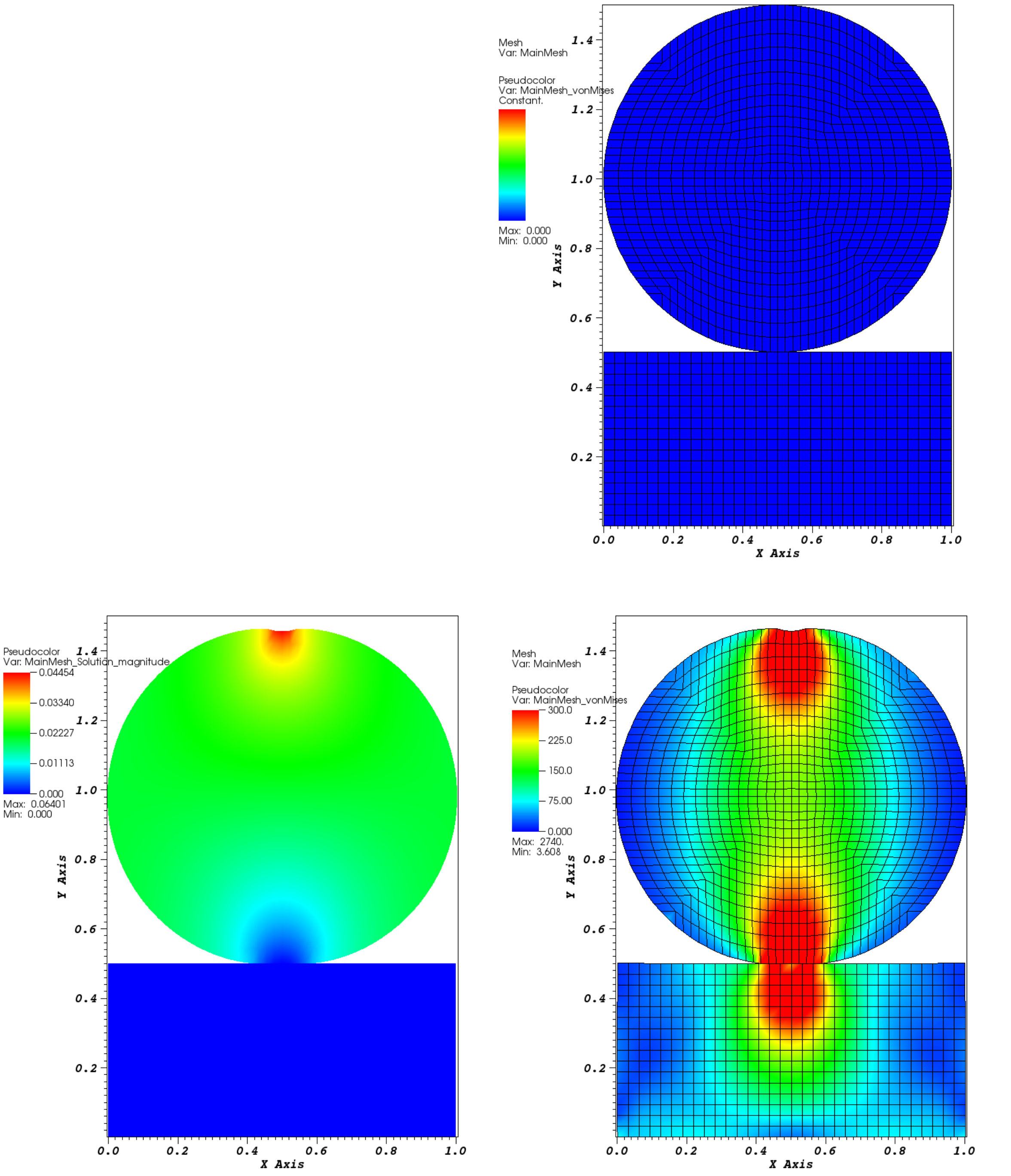


Figure VIII.13: Elastic cylinder: Initial configuration of the bodies on a 2D slice (upper right) and distorted bodies with the displacement magnitude $\|\boldsymbol{u}\|$ (lower left) and the effective von Mises stress σ_{eff} (lower right).

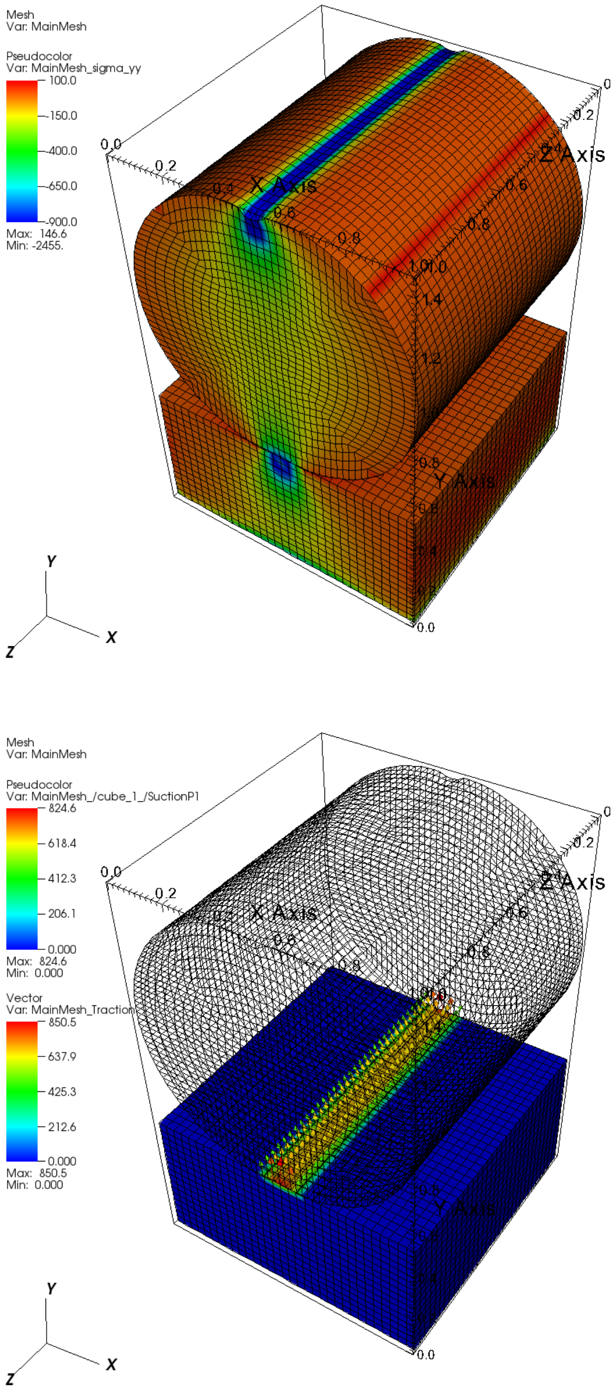


Figure VIII.14: Elastic cylinder: 3D deformation of a cylinder pressed onto a quasi-rigid brick with the stress component σ_{yy} (upper) and the normal contact pressure p_n on top of the brick and the resulting contact forces $\boldsymbol{t}$ onto the cylinder surface (lower).

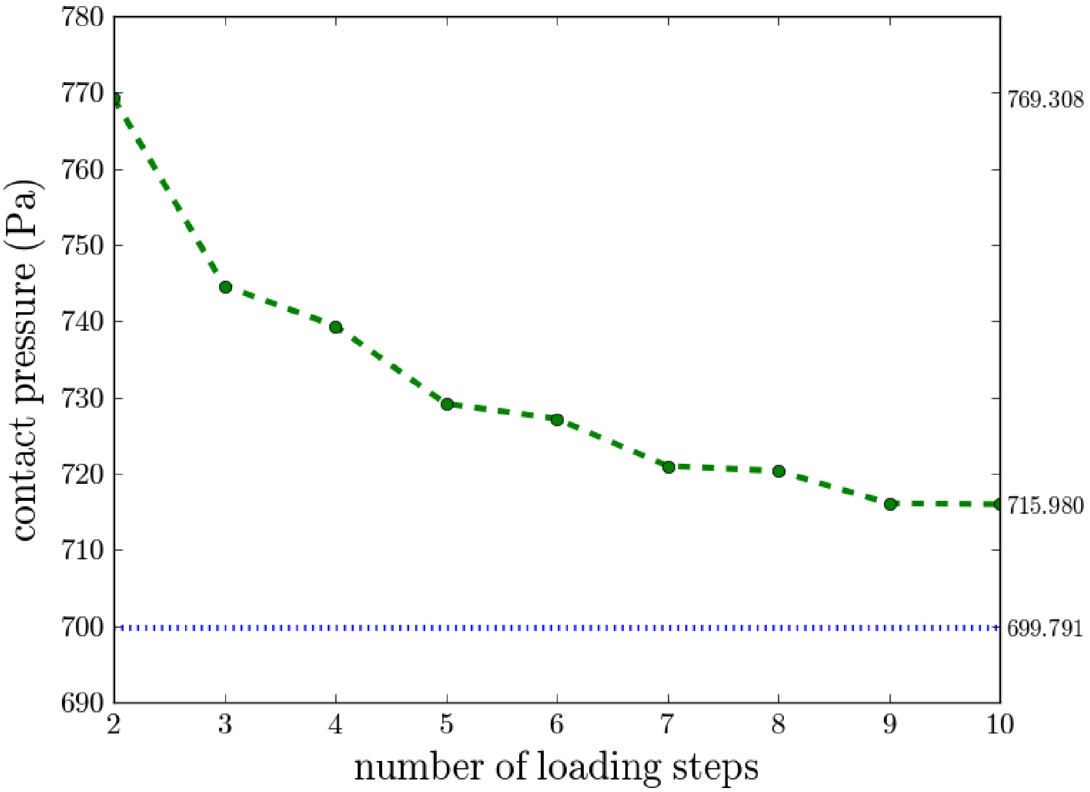


Figure VIII.15: Elastic cylinder: Convergence of the contact pressure with the loading steps. The dashed line indicates computed values of $p_n(0)$ for an increasing number of loading steps.

The use of infinite friction coefficient raises an issue we did not have with the previous numerical example with the stacked cubes. If we apply the load f too abruptly, the active set strategy paired with a direct elimination approach will converge to a solution, but this solution may be non-physical. At the first iteration, the surface nodes of Ω_s will penetrate quite deep into the foreign body Ω_m before the update of the active set corrects this and projects them back to the surface. The problem is that this projection is based on minimum distances and that these distances are measured on a temporary solution obtained with an inaccurate active set, which leads to the formulation of biased constraints. Because the stick contact condition does not allow slip in the tangent plane, these constraints, which are slightly off, will probably remain without being corrected, but they will also propagate the error in the next iterations. Rapid changes in the active set need to be examined carefully. As a general rule, loss of contact is numerically harmless but activation of too many slave nodes at once can be problematic.

An outer loop bringing the load to its full value incrementally, with a full active set iteration at each step, allows us to deal with that issue. Loading steps are standard in frictional contact and in the analysis of nonlinear material behavior more generally (e.g., plasticity). We plotted the value of maximum contact pressure $p_n(0)$ as a function of the number of loading steps on Figure VIII.15. We observe convergence to a value slightly higher than the one predicted by the theory. The difference can be explained as the combined effect of the discretization error and the fact that Hertzian contact theory assume zero friction. When solving with 20 loading steps we obtain $p_n(0) = 710.771$, which confirm that the algorithm converge to a solution as the number of steps increases. The difference with the analytical frictionless value is now under 1.6%. The use of loading steps increases the overall computation time but this needs to be put into perspective since the changes in the contact zone are not as abrupt and the active set iteration will most likely converge in fewer steps.

VIII.2.3 Extension to frictionless contact

The direct elimination of the constraints approach as presented in Section VII.2 was restricted to the contact with "stick" condition (i.e., infinite friction coefficient) since we assumed that the slave degrees of freedom were fully constrained by master faces which allowed us to eliminate the constrained degrees of freedom. In this section, we show that with a few modifications to our procedure, we can enforce the "slip" condition (i.e., frictionless contact). This is a noteworthy extension to our dissertation work because it enables the rapid implementation of other friction models (e.g., Coulomb model).

In the absence of friction, the constraints in Equation (VII.5) are projected onto the direction of the normals at the contact points[4]

$$nn^T u_2^{\mathcal{A}} = nn^T C u_1^{\mathcal{M}} + nn^T d_2^{\mathcal{A}}. \tag{VIII.3}$$

Only the normal component to the contact surface is constrained and the vector of unknown nodal displacement becomes

$$\begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix} = \begin{pmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & C' & O & 0 \\ 0 & 0 & 0 & I \end{pmatrix} \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ d_2^{\mathcal{A}} \\ 0 \end{pmatrix}, \tag{VIII.4}$$

where the operator $P := nn^T$ and $O := I - P$ represent respectively the projection onto and the orthogonalization against the normal direction. For convenience, we define $C' := PC$ and observe that $P^T = P$, $O^T = O$, and thus $C'^T = C^T P$.

The reduced form of the system for the contact problem is derived just as it was done for the infinite friction coefficient in Section VII.2 and we obtain the following

[4] For any slave node $p \in \mathcal{A}$, the slip contact condition reads

$$nn^T[p]u[p] = nn^T[p] \sum_q C[p,q]u[q] + nn^T[p]d_2[p] \tag{VIII.2}$$

where $nn^T[p] := (n[p]n^T[p])\cdot$, $n[p]$ being the normal to the master face at the point of contact on which node p is constrained. The sum over q can be restricted to the subset of nodes in $\mathcal{M}$ supporting the master face. For more details, please refer to Section V.2.5.

operator

$$K^* = \begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} & K_1^{\mathcal{N}\mathcal{M}} & 0 & 0 \\ K_1^{\mathcal{M}\mathcal{N}} & K_1^{\mathcal{M}\mathcal{M}} + C'^T K_2^{\mathcal{A}\mathcal{A}} C' & C'^T K_2^{\mathcal{A}\mathcal{A}} O & C'^T K_2^{\mathcal{A}\mathcal{N}} \\ 0 & O K_2^{\mathcal{A}\mathcal{A}} C' & O K_2^{\mathcal{A}\mathcal{A}} O & O K_2^{\mathcal{A}\mathcal{N}} \\ 0 & K_2^{\mathcal{N}\mathcal{A}} C' & K_2^{\mathcal{N}\mathcal{A}} O & K_2^{\mathcal{N}\mathcal{N}} \end{pmatrix} \qquad \text{(VIII.5)}$$

and right-hand-side

$$f^* = \begin{pmatrix} f_1^{\mathcal{N}} \\ f_1^{\mathcal{M}} + C'^T f_2^{\mathcal{A}} - C'^T K_2^{\mathcal{A}\mathcal{A}} P d_2^{\mathcal{A}} \\ O f_2^{\mathcal{A}} - O K_2^{\mathcal{A}\mathcal{A}} P d_2^{\mathcal{A}} \\ f_2^{\mathcal{N}} - K_2^{\mathcal{N}\mathcal{A}} P d_2^{\mathcal{A}} \end{pmatrix}. \qquad \text{(VIII.6)}$$

Here are the few modifications on the subroutines to solve for frictionless contact:

1) copy master to slave $u^{\mathcal{A}} = C' u^{\mathcal{M}} + O u^{\mathcal{A}}$

   i.e., $u = \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ C' u_1^{\mathcal{M}} + O u_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix}$

2) apply mechanics operator $r = Ku$

   i.e., $r = \begin{pmatrix} K_1^{\mathcal{N}\mathcal{N}} u_1^{\mathcal{N}} + K_1^{\mathcal{N}\mathcal{M}} u_1^{\mathcal{M}} \\ K_1^{\mathcal{M}\mathcal{N}} u_1^{\mathcal{N}} + K_1^{\mathcal{M}\mathcal{M}} u_1^{\mathcal{M}} \\ K_2^{\mathcal{A}\mathcal{A}} C' u_1^{\mathcal{M}} + K_2^{\mathcal{A}\mathcal{A}} O u_2^{\mathcal{A}} + K_2^{\mathcal{A}\mathcal{N}} u_2^{\mathcal{N}} \\ K_2^{\mathcal{N}\mathcal{A}} C' u_1^{\mathcal{M}} + K_2^{\mathcal{N}\mathcal{A}} O u_2^{\mathcal{A}} + K_2^{\mathcal{N}\mathcal{N}} u_2^{\mathcal{N}} \end{pmatrix}$

3) add slave to master $r^{\mathcal{M}} = r^{\mathcal{M}} + C'^T r^{\mathcal{A}}$

$$\text{i.e., } r = \begin{pmatrix} K_1^{\mathcal{NN}} u_1^{\mathcal{N}} + K_1^{\mathcal{NM}} u_1^{\mathcal{M}} \\ K_1^{\mathcal{MN}} u_1^{\mathcal{N}} + (K_1^{\mathcal{MM}} + C'^T K_2^{\mathcal{AA}} C') u_1^{\mathcal{M}} + C'^T K_2^{\mathcal{AA}} O u_2^{\mathcal{A}} + C'^T K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ K_2^{\mathcal{AA}} C' u_1^{\mathcal{M}} + K_2^{\mathcal{AA}} O u_2^{\mathcal{A}} + K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ K_2^{\mathcal{NA}} C' u_1^{\mathcal{M}} + K_2^{\mathcal{NA}} O u_2^{\mathcal{A}} + K_2^{\mathcal{NN}} u_2^{\mathcal{N}} \end{pmatrix}$$

4) set slave to zero $r^{\mathcal{A}} = Or^{\mathcal{A}}$

$$\text{i.e., } r = \begin{pmatrix} K_1^{\mathcal{NN}} u_1^{\mathcal{N}} + K_1^{\mathcal{NM}} u_1^{\mathcal{M}} \\ K_1^{\mathcal{MN}} u_1^{\mathcal{N}} + (K_1^{\mathcal{MM}} + C'^T K_2^{\mathcal{AA}} C') u_1^{\mathcal{M}} + C'^T K_2^{\mathcal{NA}} O u_2^{\mathcal{A}} + C'^T K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ O K_2^{\mathcal{AA}} C' u_1^{\mathcal{M}} + O K_2^{\mathcal{AA}} O u_2^{\mathcal{A}} + O K_2^{\mathcal{AN}} u_2^{\mathcal{N}} \\ K_2^{\mathcal{NA}} C' u_1^{\mathcal{M}} + K_2^{\mathcal{NA}} O u_2^{\mathcal{A}} + K_2^{\mathcal{NN}} u_2^{\mathcal{N}} \end{pmatrix}$$

5) add shift to slave $u^{\mathcal{A}} = u^{\mathcal{A}} + Pd^{\mathcal{A}}$

$$\text{i.e., } u = \begin{pmatrix} u_1^{\mathcal{N}} \\ u_1^{\mathcal{M}} \\ u_2^{\mathcal{A}} + P d_2^{\mathcal{A}} \\ u_2^{\mathcal{N}} \end{pmatrix}$$

As one can see, if we impose $P = I$ (and thus have $O = 0$) we recover the equations derived for the infinite friction coefficient. In addition, we could be combining the two types of conditions, thus enabling Coulomb-like friction models. The choice stick or slip condition can be done on an individual basis for each slave node. For instance, we may prescribe slip just for a number of slave vertices where the tangential component of the contact forces has exceeded some critical value.

The changes were implemented in our node-to-face contact operator. In the following, we revisit the problem with the stacked cubes from Section VIII.2.1 to demonstrate the new modeling capabilities. Figure VIII.16 shows side-by-side the calculated von Mises stresses σ_{eff} on distorted bodies for both stick and slip contact conditions. It is easy to notice that the lower left corner of the upper brick slipped to the right. The stresses pattern are quite similar. On Figure VIII.17, we report

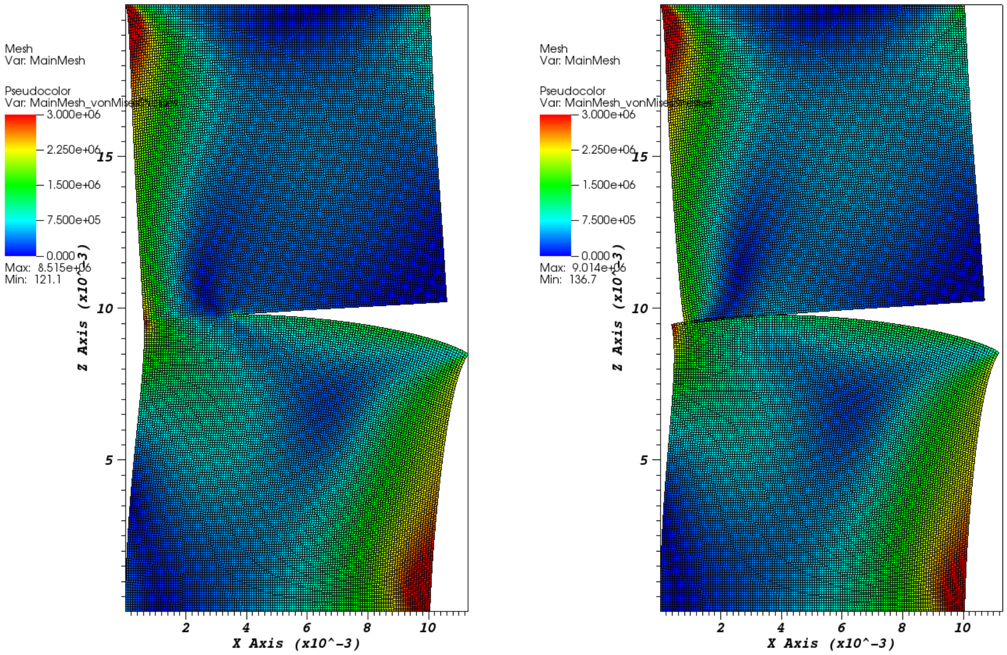


Figure VIII.16: Stacked cubes: Distorted bodies with the effective von Mises stress σ_{eff} in both cases, infinite friction coefficient (left) and zero friction (right).

the normal contact stresses $p_n = -\boldsymbol{t} \cdot \boldsymbol{n}$ on the centerline of the upper cube. As one can see, the contact pressure computed with zero friction is very close from what we obtained for an infinite coefficient of friction. The only noticeable difference is that the loss-of-contact point is slightly shifted in the direction of the applied forces.

VIII.2.4 Preconditioning

Mathematical modelling of three-dimensional problems quickly lead to systems with large numbers N_d of unknowns that need to be solved many times in multiphysics application. Unless we spend some effort on preconditioning to accelerate the convergence rate of the Krylov solver, high-fidelity simulation may become impractical.

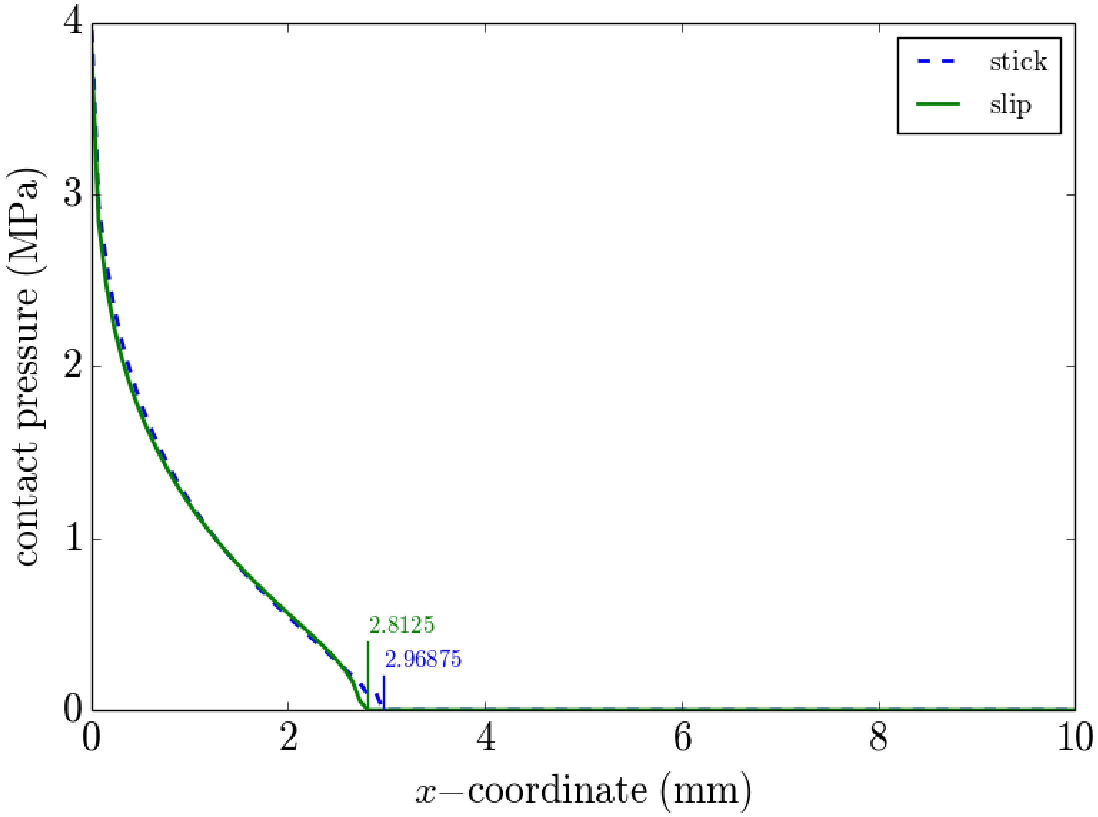


Figure VIII.17: Stacked cubes: Comparison of the contact pressure on the centerline of the upper cube lower face in the absence of friction (solid green line "stick" condition) versus when the coefficient of friction is infinite (dashed blue line "slip" condition). Loss-of-contact points are highlighted.

While highly efficient solvers on their own, multigrid methods also serve as excellent precondioners [8]. These play upon the fact that different components of the error are more effectively represented and eliminated on grids of different resolutions and use a grid hierarchy to damp all frequencies. The class of algebraic multigrid method (AMG) offers the potential for a black-box solver without any reference to an underlying mesh structure. Near-optimal efficiency has been demonstrated for large-scale 3D elasticity and plasticity problem [2].

In our implementation of the direct elimination approach to enforce the contact constraints (cf. Chapter VII), the blocks K_1 and K_2 are constructed at the beginning of the calculation and then remain untouched throughout the active set iteration. Only the constraint $Pu_2^{\mathcal{A}} = C'u_1^{\mathcal{M}} + Pd_2^{\mathcal{A}}$ is updated. A matrix-free approach is adopted, i.e., the reduced operator K^* is not actually formed but its action on Krylov vectors is computed as the successive application of operations involving the action of $K = \text{diag}(K_1, K_2)$ as well as C', C'^T, and P which are updated at each step of the iteration to find the correct active set $\mathcal{A}$.

K^* is not directly available, but matrices K^* and K are very similar and even more so as $|\mathcal{A}| \ll N_d$. Multigrid can be employed on K in an first attempt of preconditioning the contact problem. However, this comes with a caveat: each individual subproblem $K_i u_i = f_i$ is then required to be well-posed when taken apart from the original problem. Fortunately, it is the case in the problem with the two stacked cubes of Section VIII.2.1 but it is certainly not always true. Looking back at the problem with an elastic cylinder pressed onto a quasi rigid brick in Section VIII.2.2, the subproblem involving the brick is well posed if contact is disregarded but the one with the cylinder is not. Some kinds of boundary conditions need to be enforced to eliminate body motion in the direction the normal to the brick. For that reason, the latter is taken as an example in this section, to illustrate how this can be overcome.

Symmetry of the system has been preserved which allows to work with a preconditioned conjugate gradient (PCG). This is an advantage because CG uses litle memory for large-scale problems relatively to other methods (e.g., GMRES). In place of preconditioner, we use the multilevel preconditioning package ML from Trilinos [20, 24]. Homogeneous Dirichlet boundary conditions are added on the displacement in the y-direction on the segment of nodes that are initially in contact with the brick (i.e., $y = 0.5$) so that block K_1 is not singular any more. The conditions cannot be directly applied to the actual matrix K and our first choice was to provide ML with a modified action of K_1 onto the vector of nodal displacement in the cylinder that accounts for them. The number of CG iterations required to reach convergence was efficiently reduced but CPU time increased suspiciously taking away all benefits from preconditioning. Constructing a copy K_1 and applying the Dirichlet conditions on it divided the CPU time by a factor 10 on small problems (∼1,000 elements).

Table VIII.3: Elastic cylinder: Number of required CG iterations and CPU time in seconds to solve the problem with no preconditioner or using ML. l is the level of mesh refinement and $k = 0, 1, \ldots, K_l$ denotes the different steps of the active set iteration. $l = 0, 1, 2, 3$ represent 768, 6,144, 49,152, and 393,216 elements, respectively.

		Frictionless contact				No contact			
		NONE		ML		NONE		ML	
l	k	iter	time	iter	time	iter	time	iter	time
0	1	2,600	17.9	130	5.2	131	0.8	17	0.9
1	1	5,307	164.6	178	55.3	252	7.4	22	8.3
	2	4,395	157.8	313	95.3				
2	1	10,446	2,220.2	243	563.1	495	102.0	27	80.2
	2	8,376	1,989.6	961	2,212.0				
	3	8,655	2,011.3	656	1,549.3				
3	1	20,688	33,385.2	309	6,159.9	994	1568.4	31	694.8
	2	15,512	26,863.4	1,509	28,847.8				
	3	15,599	27,194.3	1,556	30,694.0				
	4	16,028	27,864.3	1,470	28,870.9				

Table VIII.3 reports the number of required CG iterations and CPU time to solve $K^* u = f^*$ for the unknown nodal displacement u at several levels of refinement l. Level 3 corresponds to 393,216 elements which represents about a million degrees of freedom. At each level, values are given for the different steps $k = 1, \ldots, K_l$ of the active step strategy and are compared to the solution of the problem with no contact (i.e., $K^* = K$). The data correspond to frictionless contact but similar results were obtained with an infinite coefficient of friction. We observe that ML significantly reduces the number of CG iterations but that it fails to prevent the iteration counts from growing as the mesh size h is decreased. We also note that the acceleration is more pronounced at the first step ($k = 1$) of the active set iteration which is precisely the case with the least differences between entries of K and K^* since $|\mathcal{A}|$ is minimal. For that reason, it would be interesting to be able to employ AMG on the actual operator K^*. The matrix-free approach yet appears to be unpractical and the necessary changes to our implementation in order to actually construct K^* are not trivial. We leave this as possible future work. As for the measured CPU times, we notice that, even though the reduction in terms of the iteration counts is encouraging, the use ML as preconditioner gives mixed results since it does not improve performances. It would be of great interest to test other implementations of AMG since it seems to us that the symmetric Gauss-Seidel smoother of ML is performing poorly.

VIII.2.5 Pellet-cladding mechanical interaction

We conclude the Results Chapter by a demonstration of contact modeling capabilities using a 3D model of UO_2 pellets + deformable Zr-4 clad. The assumed geometry shown in Figure VIII.18 includes two individual UO_2 pellets and the Zr-4 cladding with an initial gap width of 85 μm between them. More details on the fuel and cladding dimensions are given in Table VIII.4. Typical PWR operating conditions were used.

In each pellet, a uniform fission heat source was assumed. The temperature-dependent thermal conductivity for UO_2 is prescribed by

$$k[\mathrm{UO_2}](T) = 1.05 + \frac{2150}{200+T}\mathrm{W/m\ C.} \tag{VIII.7}$$

Other material properties such as Young's modulus, Poisson's ratio, thermal expan-

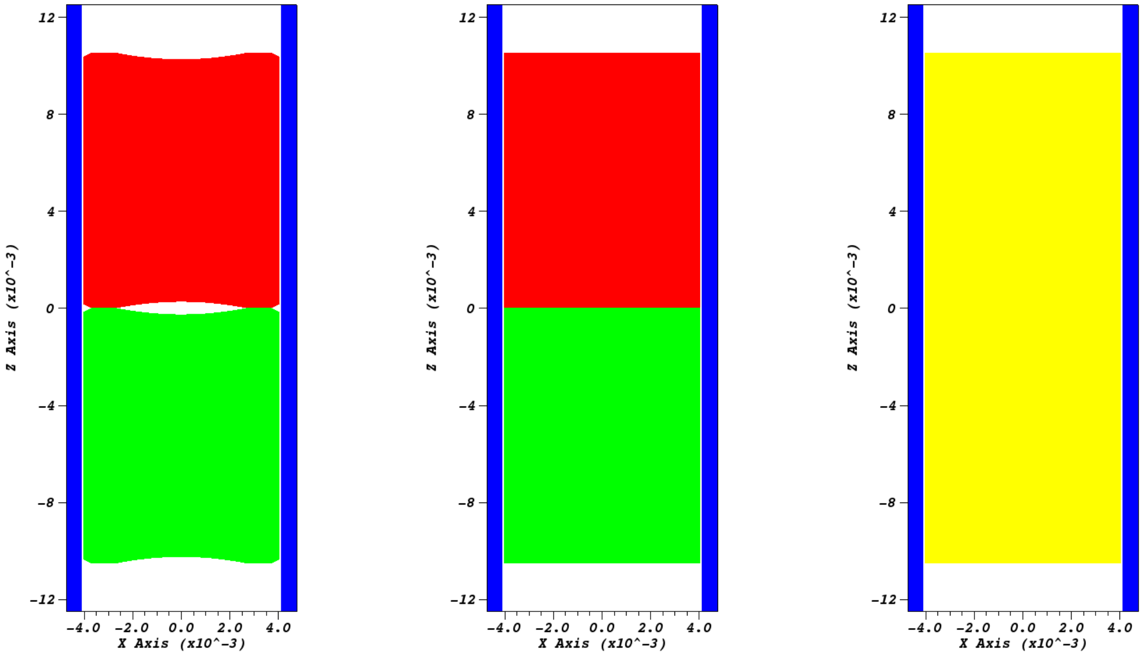


Figure VIII.18: PCMI: Problem geometry with two pellet elements (red and green) stacked one on top of the other and placed inside the cladding (blue). Our model (left) uses the double dish chamfered pellet geometry . For comparison, we show two simplified geometries with cylindrical pellets (middle) and a single “smeared” pellet (right).

sion coefficient, both for the fuel and for the Zr-4 cladding, are assumed constant. Table VIII.5 gather all the details about the values that were used in our calculations.

The linear heat generation rate is uniform over the rod length, equal to 20 kW/m. The bulk coolant temperature is set at 260 C with a convective heat transfer coefficient between the cladding of 7.5 kW/m^2.C. The gap conductance model describing the heat exchange between fuel and cladding was simplified, the heat transfer coefficient is just the gas thermal conductivity $k[\mathrm{He}]$ divided by the initial gap width g_n. The corresponding calculated radial temperature profile is plotted on Figure VIII.19 and shown on a 2D slice orthogonal to the rod axis on Figure VIII.20. The maximum

Table VIII.4: Geometric data and material for the PCMI calculation.

Fuel pellet	
material	Uranium oxide (UO_2)
outer radius	4.025 mm
height	10.5 mm
chamfer inner radius	3.725 mm
chamfer depth	0.15 mm
dish depth	0.26 mm
spherical radius of dish	2.65 mm
Clad	
material	Zircaloy (Zr-4)
outer radius	4.75 mm
inner radius	4.11 mm
gap width	0.085 mm
height	25.0 mm

temperature is along the centerline of the rod at about 929 C and T decreases with increasing radius. Values in the fuel pellet are much larger in magnitude than the temperatures observed in the cladding as the computed pellet surface temperature 473 C is compared to 378 C and 349 C at the cladding inner and outer radii, respectively. Correspondingly, thermal expansion is much larger in the fuel than in the cladding. However, fuel expansion only results in an outward displacement of about 30 μm near the pellet ends which is not sufficient to close the gap, even more so since, the cladding inner surface undergoes a radial thermal expansion of 7 μm in the same direction. This is not a surprise, gap closure and mechanical contact typically occurs later in fuel life, as burnup increases. Our calculations do not include models for the cladding creep down process nor for the irradiation-induced swelling of the fuel that occur during reactor operation. As a consequence, we cannot expect observing the initial gap to close under fresh fuel conditions. In order to experience gap closure with fresh fuel, one could exaggerate fuel swelling by artificially increasing the ther-

Table VIII.5: Summary of the material properties used in PCMI calculations for the fuel, the cladding, and the gas filling the gap between the two.

Property	Value
Poisson's ratio	
$\nu[\mathrm{UO}_2]$	0.345
ν[Zr-4]	0.3
Young's modulus	
$E[\mathrm{UO}_2]$	219 GPa
E[Zr-4]	75 GPa
thermal expansion coefficient	
$\alpha[\mathrm{UO}_2]$	10×10^{-6} C^{-1}
α[Zr-4]	5×10^{-6} C^{-1}
thermal conductivity	
$k[\mathrm{UO}_2]$	*** W/mC
k[Zr-4]	0.67 W/mC
k[He]	13.0 W/mC

*** depends on material temperature T (cf. Eq. (VIII.7))

mal expansion coefficient $\alpha[\mathrm{UO}_2]$. However, here we have opted not to artificially modify the material parameters but instead we "shrink" the cladding tube by 5% just before initiating the active set strategy.

Figure VIII.21 shows the magnitude of the displacement solution $\|\boldsymbol{u}\|$. We show the solution on the distorted bodies for two different cases: a/ keeping the clad original dimensions (left), and b/ shrinking it by 5% (right). In the first case, the gap remains fully open and there is no mechanical interaction between the two pellets and the clad. In the second one, contact occurs. Figure VIII.22 gives the normal stresses on the inner surface of the cladding. The bamboo ridge formation is obvious for the discrete-pellet simulations when the displacements are magnified but we have chosen not to. We measured a maximum contact pressure of 1.221 GPa at the triple-point.

Additional calculations were made using a simple cylindrical pellet shape for

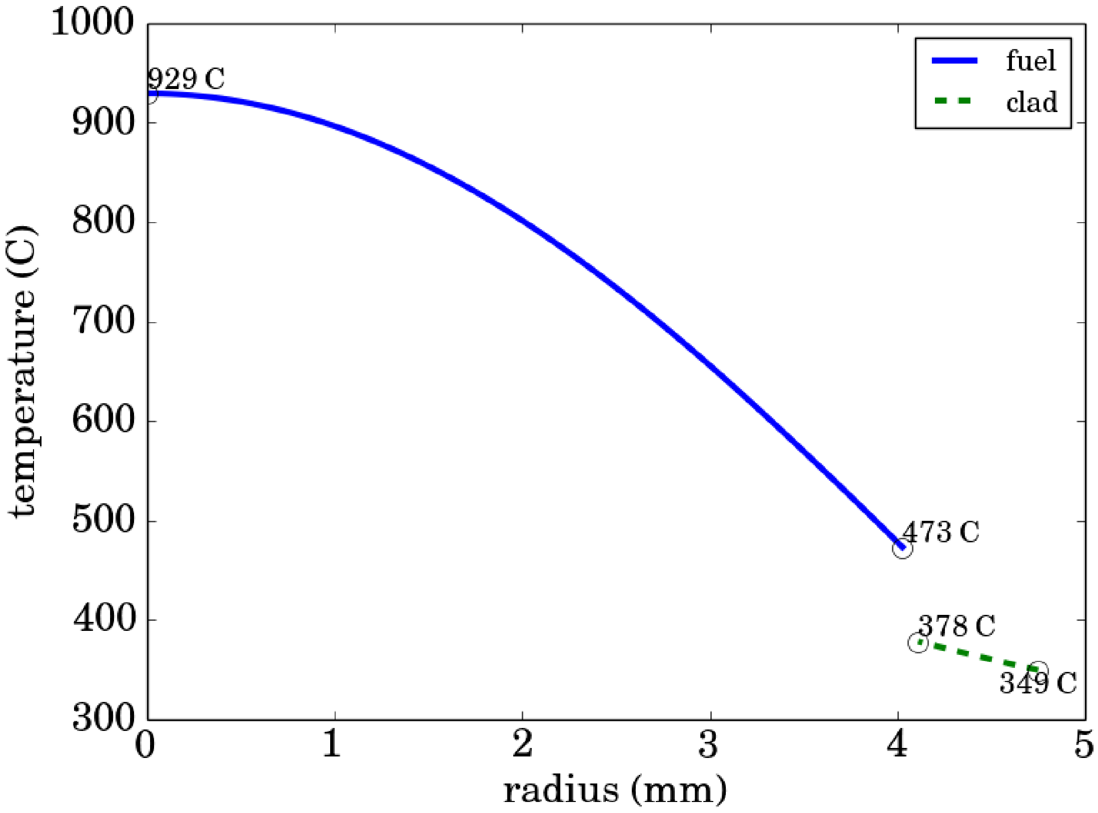


Figure VIII.19: PCMI: Radial temperature profile in the fuel and the clad.

comparison with the real geometry. It highlighted that pellet dishing is useful to control the axial expansion of the stack. Our calculations show that dishing pellets at both ends reduces the elongation of the stack in the z-direction by about 10%. The real geometry is beneficial because it slightly counterbalances the hourglassing of the fuel pellets which results in lower cladding stresses. We also performed numerical tests where we replaced the discrete-pellet geometry by the common smeared-pellet approximation where the fuel column is modeled as a single cylinder (cf. Figure VIII.18). As expected, the simplified geometry does not allow to model the stress increase at the triple point (i.e., where cladding meets pellet-pellet interface).

Cladding stresses are under-predicted by a factor of 2 in that region.

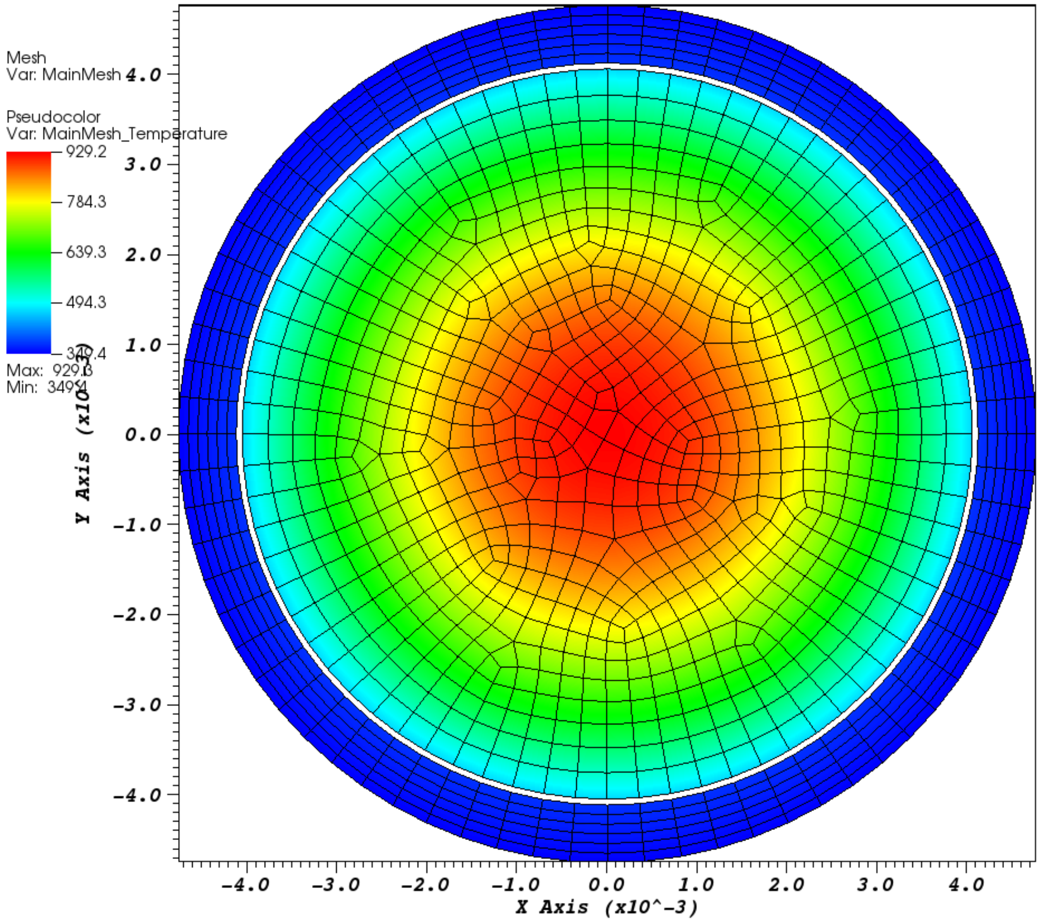


Figure VIII.20: PCMI: Initial configuration with the prescribed radial temperature profile T for the fuel and the clad.

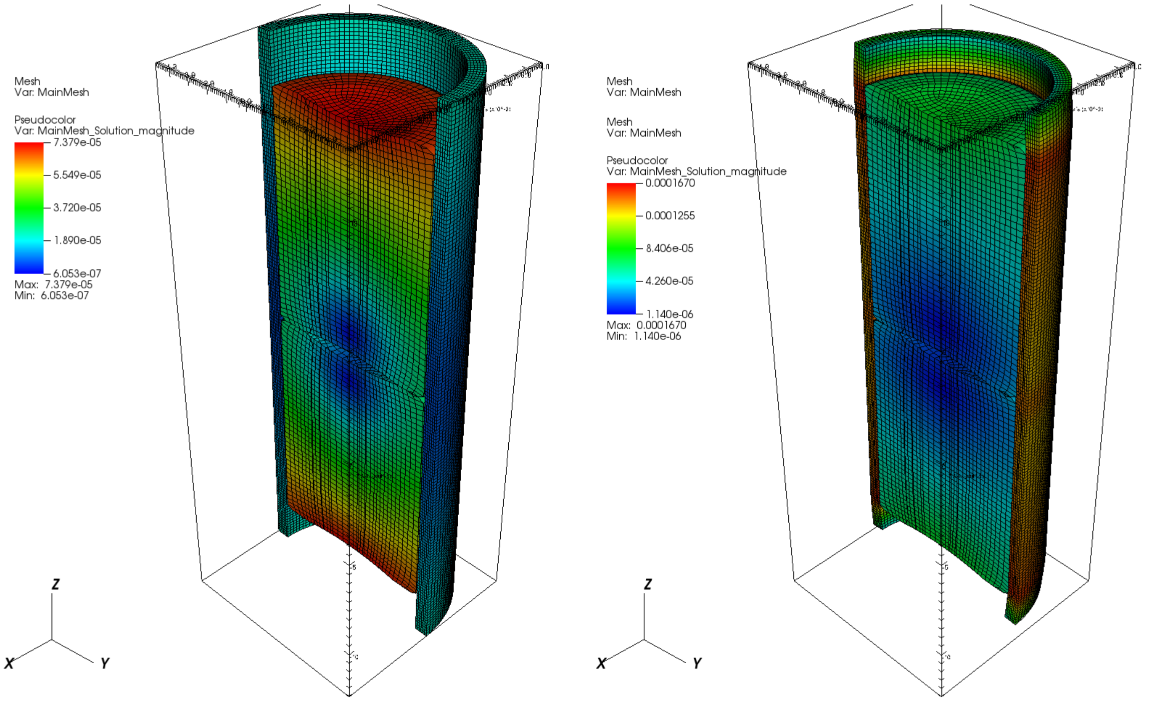


Figure VIII.21: PCMI: Magnitude of the displacement $\|\boldsymbol{u}\|$ on the distorted bodies for the real geometry. Both cases gap open (left) and gap closed (right) are represented.

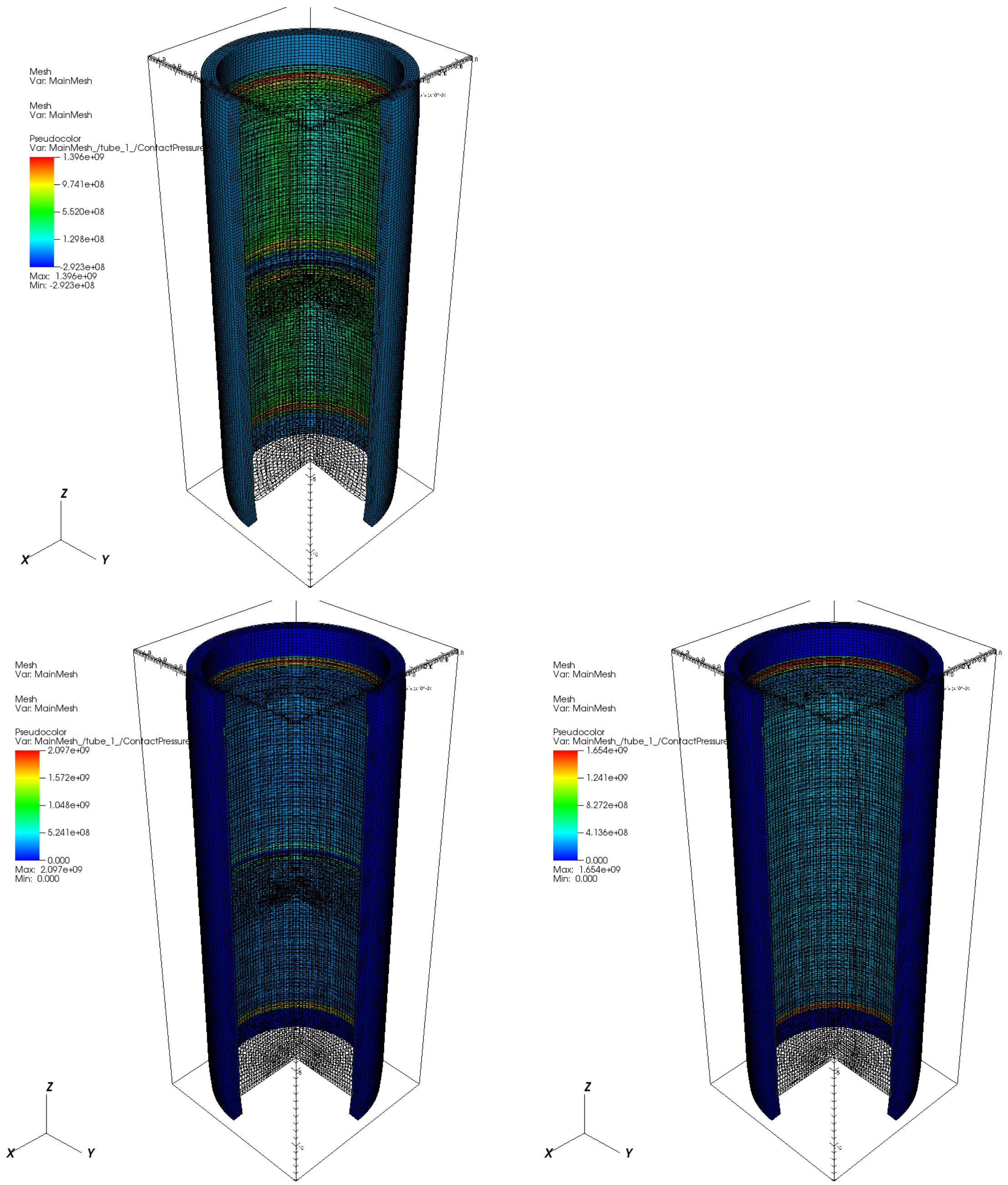


Figure VIII.22: PCMI: Distorted domain with the contact pressure $p_n = -\boldsymbol{t} \cdot \boldsymbol{n}$ for the real geometry (upper left). Numerical results when using cylindrical pellet (lower left) and with the smeared-pellet approximation (lower right) are given for comparison.

CHAPTER IX

CONCLUSION AND FUTURE WORK

IX.1 Conclusion

In this dissertation, we first introduced the challenges that need to be addressed by next-generation nuclear fuel performance codes. Stresses generated on the cladding and fuel centerline temperatures cannot be predicted with high levels of fidelity unless the pellet-clad mechanical interaction is accurately treated. Adequate resolution for high-fidelity simulations can only be achieved through three-dimensional fully coupled computations of the thermo-mechanical contact problem with multiple bodies, which necessitates the use of state-of-the-art multiphysics solution strategies.

We recalled that previous 1-1/2D approaches were unable to capture localized stress peaks occurring in the clad at pellet-pellet interface and explained that 3D development effort with more advanced numerical methods had only started in the late 2000s. We saw that research is now focusing on algorithms capable to handle contact and resolve heat fluxes across the gap with more accuracy.

We gave several possible formulations of the contact constraints and explained that they introduce another type of nonlinearity which is fundamentally different than the material nonlinearities usually encountered in multiphysics applications (e.g., temperature-dependence of the fuel thermal conductivity) and therefore are not tackled with a standard Newton-based iteration. We described an active set strategy as an iterative procedure which can be employed to determine the actual contact area.

We explained that the numerical solution of contact problems needed efficient parallel search algorithms to quickly identify the boundary regions where the non-penetration condition is violated and update the contact constraints. We proposed and implemented an efficient and robust search algorithm that was used for detecting mesh interpenetration and vertex/mesh overlap.

We discussed different techniques to incorporate the contact constraints into the variational formulation and suggested an alternative to the two methods most commonly used. The procedure consists of a direct elimination of the constrained slave degrees of freedom which is advantageous because it reduces the number of unknowns without dramatically altering the convergence properties of the system. We proposed a matrix-free implementation based on small operations on the slave and master part of the vector of nodal displacement to modify the action of the block diagonal matrix arising from the discretization of the problem without contact.

We provided a numerical example of a fully coupled thermo-mechanical contact problem between multiple fuel pellets and the cladding treated as a rigid body using Lagrange multipliers. We performed tests showing that our implementation of the direct elimination method was able to handle 3D contact properly and successfully applied it finally to the simulation of the pellet-clad mechanical interaction, with the cladding treated as a deformable body this time.

To end this conclusion, we recall the main goals of this dissertation work:

1. to propose and implement an efficient search algorithm for detecting mesh inter-penetration and vertex/mesh overlap,

2. to elaborate a procedure for the direct elimination of constraints,

3. to choose and implement a suitable method to enforce contact,

4. to define a strategy to identify and update the contact area Γ^C throughout the

computation,

5. to obtain a fully-coupled solution of the coupled heat transport and thermomechanical contact.

IX.2 Future work

Next, we propose several research directions worthy of further investigations:

Preconditioning: We attempted to speed up calculations by using algebraic multigrid techniques for preconditioning the contact problem and obtained mixed results when providing the diagonal blocks (i.e., system without the contact constraints) to the multigrid preconditioner. The number of iterations required to converge was significantly reduced but augmented as mesh size decreased. Our results suggested that constructing the actual full matrix (i.e., reduced system which incorporates the constraints from contact) may improve the preconditioner efficiency, since we observed that the iteration counts was far less sensitive to the grid resolution when the differences between the actual reduced system and its contact-free block diagonal version were minimal.

Physics model: We demonstrated the capability to model 3D contact in the context of fuel behavior analysis. For simplicity, only the thermal expansion term has been considered in the mechanics calculation. Other non elastic contributions to the strain in Equation (II.7) can be added to our model to account for plasticity, swelling, densification, thermal and irradiation creep, etc. More information on fuel and cladding material model can be found in [43]. Our algorithms for contact only require a model to compute the stress state within the pellets and the clad.

We ignored the sensitivity of our models to phenomena such as chemical diffusion of oxygen in the fuel. Coupling to a model for the transport of oxygen in UO_2 would be interesting since a number of material properties are known to be affected by changes of the oxygen stoichiometry x. Notably, it has been demonstrated [32] that accounting for the oxygen content thermal conductivity $k^{fuel}(T, x)$ in UO_{2+x} fuel elements leads to centerline temperatures that are substantially different from predictions performed using a composition-independent model $k^{fuel}(T)$.

Solution strategy: In this dissertation, we presented a contact solution strategy which consisted in a nested Newton iteration with an outer active set loop. It is possible to organize them slightly differently so that both the nonlinearity of the material behavior and the nonlinearity from contact are considered within the same iteration loop [9]. Merging the two loops was shown to decrease the computational time. We would like to stress the fact that the search for contact would then be performed at each iteration. Other approaches often use a predefined relation mapping the possible slave and master contact surfaces, which reduces considerably the cost associated to the update of the active set. In other words, geometrical search is performed only once, before the iteration starts, with respect to the initial body configuration.

Primal-dual approach: We briefly discussed the possibility of using the dual space to formulate face-to-face contact and explained that it was advantageous when inverting the relation between slave and master degrees of freedom because the biorthogonality of the finite element basis functions yields a diagonal mass matrix in Equation (V.20). It would be possible to replace the node-to-face constraint in our implementation of the direct elimination strategy.

Contact detection and gap width evaluation: Our search algorithm performs well but it does not measure positive values of the gap width. Yet, g_n is necessary to compute heat fluxes across the gap when it is open. The cost associated to the projection of a single point onto a collection of boundary faces is $O(1/h^2)$ unless accelerated (the number of projections that need to be performed also grows inversely to the square of the mesh size). A possible implementation of the acceleration would be to restrict the projection for a given slave vertex to a limited number of master faces that are the closest to the point in the initial configuration. If necessary the list of plausible candidates for interaction is updated in the course of the simulation.

REFERENCES

[1] S. Aas. Mechanical interaction between fuel and cladding. *Nuclear Engineering and Design*, 21(2):237–253, 1972.

[2] M. Adams and R. L. Taylor. Parallel multigrid solvers for 3d-unstructured large deformation elasticity and plasticity finite element problems. *Finite Elements in Analysis and Design*, 36(3–4):197–214, October 2000.

[3] D. D. Baron and L. Hallstadius. Fuel performance of light water reactors (uranium oxide and mox). In R. J. M. Konings, editor, *Comprehensive Nuclear Materials*, volume 2: Material Properties/Oxide Fuels for Light Water Reactors and Fast Neutron Reactors, chapter 19, pages 481–514. Elsevier, Oxford, United Kingdom, 2012.

[4] F. Bentejac and N. Hourdequin. Toutatis: an application of the cast3m finite element code for pci three-dimensional modeling. In *Proccedings of Pellet-Clad Interaction in Water Reactor Fuels*, pages 495–506, Aix-en-Provence, France, March 2004. OECD, Agence pour l'Energie Nucléaire.

[5] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 2005.

[6] Y. Bernardi C., Maday and A. Patera. Domain decomposition by the mortar element method. In H. Kaper, M. Garbey, and G. Pieper, editors, *Asymptotic and Numerical Methods for Partial Differential Equations with Critical Parameters*, volume 384 of *NATO ASI Series*, pages 269–286. Kluwer Academic Publishers, Dordrecht, the Netherlands, 1993.

[7] D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods*, volume 4 of *Optimization and neural computation series.* Athena Scientific, Belmont, Massachusetts, 1996.

[8] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial, Second Edition*, chapter 8. Algebraic Multigrid (AMG), pages 137–161. Society for Industrial and Applied Mathematics, 2000.

[9] S. Brunssen, F. Schmid, M. Schäfer, and B. Wohlmuth. A fast and robust iterative solver for nonlinear contact problems using a primal-dual active set strategy and algebraic multigrid. *International Journal for Numerical Methods in Engineering*, 69(3):524–543, 2007.

[10] C. Burstedde, L. Wilcox, and O. Ghattas. p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM Journal on Scientific Computing*, 33(3):1103–1133, 2011.

[11] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*, chapter Runge–Kutta Methods, pages 137–316. John Wiley & Sons, Ltd, Chichester, United Kingdom, second edition, 2008.

[12] C. Carstensen, O. Scherf, and P. Wriggers. Adaptive finite elements for elastic bodies in contact. *SIAM Journal on Scientific Computing*, 20(5):1605–1626, 1999.

[13] K. T. Clarno, B. Philip, W. K. Cochran, R. S. Sampath, S. Allu, P. Barai, S. Simunovic, M. A. Berrill, L. J. Ott, S. Pannala, G. A. Dilts, B. Mihaila, G. Yesilyurt, J. H. Lee, and J. E. Banfield. The amp (advanced multiphysics) nuclear fuel performance code. *Nuclear Engineering and Design*, 252(0):108–120, November 2012.

[14] Dassault Systemes, Providence, Rhode Island. *ABAQUS Theory Manual (version 6.11)*, 2011.

[15] T. A. Davis. Algorithm 832: Umfpack v4.3, an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30(2):196–199, June 2004.

[16] L. Dubcova, P. Solin, G. Hansen, and H. Park. Comparison of multimesh hp-fem to interpolation and projection methods for spatial coupling of thermal and neutron diffusion calculations. *Journal of Computational Physics*, 230(4):1182–1197, February 2011.

[17] S. Eisenstat and H. Walker. Globally convergent inexact newton methods. *SIAM Journal on Optimization*, 4(2):393–422, 1994.

[18] S. Eisenstat and H. Walker. Choosing the forcing terms in an inexact newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32, 1996.

[19] K. A. Fischer and P. Wriggers. Frictionless 2d contact formulations for finite deformations based on the mortar method. *Computational Mechanics*, 36(3):226–244, August 2005.

[20] M. Gee, C. Siefert, J. Hu, R. Tuminaro, and M. Sala. ML 5.0 smoothed aggregation user's guide. Technical Report SAND2006-2649, Sandia National Laboratories, Albuquerque, New Mexico, 2006.

[21] K. Geelhood, W. Luscher, and C. Beyer. Frapcon-3.4: A computer code for the calculation of steady-state, thermal-mechanical behavior of oxide fuel rods for high burnup. Technical Report NUREG/CR-7022, Volume 1, PNNL-19418, Pacific Northwest National Laboratory, March 2011.

[22] K. Geelhood, W. Luscher, and C. Beyer. Fraptran 1.4: A computer code for the transient analysis of oxide fuel rods. Technical Report NUREG/CR-7023, Volume 1, PNNL-19400, Pacific Northwest National Laboratory, March 2011.

[23] G. Hansen. A jacobian-free newton krylov method for mortar-discretized thermomechanical contact problems. *Journal of Computational Physics*, 230(17):6546–6562, July 2011.

[24] M. Heroux, R. Bartlett, V. H. R. Hoekstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, and A. Williams. An overview of trilinos. Technical Report SAND2003-2927, Sandia National Laboratories, Albuquerque, New Mexico, 2003.

[25] P. Hild and P. Laborde. Quadratic finite element methods for unilateral contact problems. *Applied Numerical Mathematics*, 41(3):401–421, June 2002.

[26] S. Hüeber and B. Wohlmuth. An optimal a priori error estimate for nonlinear multibody contact problems. *SIAM Journal on Numerical Analysis*, 43(1):156–173, 2005.

[27] S. Hüeber and B. I. Wohlmuth. A primal–dual active set strategy for nonlinear multibody contact problems. *Computer Methods in Applied Mechanics and Engineering*, 194(27–29):3147–3166, July 2005.

[28] D. E. Keyes, L. C. McInnes, C. Woodward, W. Gropp, E. Myra, M. Pernice, J. Bell, J. Brown, A. Clo, J. Connors, E. Constantinescu, D. Estep, K. Evans, C. Farhat, A. Hakim, G. Hammond, G. Hansen, J. Hill, T. Isaac, X. Jiao, K. Jordan, D. Kaushik, E. Kaxiras, A. Koniges, K. Lee, A. Lott, Q. Lu, J. Magerlein, R. Maxwell, M. McCourt, M. Mehl, R. Pawlowski, A. P. Randles, D. Reynolds, B. Rivière, U. Rüde, T. Scheibe, J. Shadid, B. Sheehan, M. Shephard, A. Siegel,

B. Smith, X. Tang, C. Wilson, and B. Wohlmuth. Multiphysics simulations: Challenges and opportunities. *International Journal of High Performance Computing Applications*, 27(1):4–83, February 2013.

[29] D. A. Knoll and D. E. Keyes. Jacobian-free newton–krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397, January 2004.

[30] K. Lassmann. Transuranus: a fuel rod analysis code ready for use. *Journal of Nuclear Materials*, 188:295–302, June 1992.

[31] B. Michel, J. Sercombe, C. Nonon, and O. Fandeur. Modeling of pellet cladding interaction. In R. J. M. Konings, editor, *Comprehensive Nuclear Materials*, volume 3: Advanced Fuels/Fuel Cladding/Nuclear Fuel Performance Modeling and Simulation, chapter 22, pages 677–712. Elsevier, Oxford, United Kingdom, 2012.

[32] B. Mihaila, M. Stan, J. Crapps, and D. Yun. Impact of thermal conductivity models on the coupling of heat transport, oxygen diffusion, and deformation in (u, pu) nuclear fuel elements. *Journal of Nuclear Materials*, 433(1–3):132–142, February 2013.

[33] B. Mihaila, M. Stan, J. Ramirez, A. Zubelewicz, and P. Cristea. Simulations of coupled heat transport, oxygen diffusion, and thermal expansion in uo2 nuclear fuel elements. *Journal of Nuclear Materials*, 394(2–3):182–189, November 2009.

[34] C. Newman, G. Hansen, and D. Gaston. Three dimensional coupled simulation of thermomechanics, heat, and oxygen diffusion in nuclear fuel rods. *Journal of Nuclear Materials*, 392(1):6–15, July 2009.

[35] J. C. Ragusa and V. S. Mahadevan. Consistent and accurate schemes for coupled neutronics thermal-hydraulics reactor analysis. *Nuclear Engineering and Design*, 239(3):566–579, March 2009.

[36] J. C. Ramirez, M. Stan, and P. Cristea. Simulations of heat and oxygen diffusion in uo2 nuclear fuel rods. *Journal of Nuclear Materials*, 359(3):174–184, December 2006.

[37] Y. Rashid, R. Dunham, and R. Montgomery. Fuel analysis and licensing code: Falcon mod01. Technical Report EPRI 1011307, Electric Power Research Institute, December 2004.

[38] Y. Saad. *Iterative Methods for Sparse Linear Systems.* SIAM, Philadelphia, Pennsylvania, second edition, 2003.

[39] H. Sundar, R. Sampath, and G. Biros. Bottom-up construction and 2:1 balance refinement of linear octrees in parallel. *SIAM Journal on Scientific Computing*, 30(5):2675–2708, 2008.

[40] M. Suzuki, H. Saito, and Y. Udagawa. Light water reactor fuel analysis code femaxi-7; models and structure. Technical Report JAEA-Data/Code 2010-035, Japan Atomic Energy Agency, March 2011.

[41] P. Van Uffelen and M. Suzuki. Oxide fuel performance modeling and simulations. In R. J. M. Konings, editor, *Comprehensive Nuclear Materials*, volume 3: Advanced Fuels/Fuel Cladding/Nuclear Fuel Performance Modeling and Simulation, chapter 19, pages 535–577. Elsevier, Oxford, United Kingdom, 2012.

[42] R. L. Williamson. Enhancing the abaqus thermomechanics code to simulate multipellet steady and transient lwr fuel rod behavior. *Journal of Nuclear Materials*, 415(1):74–83, 8 2011.

[43] R. L. Williamson, J. D. Hales, S. R. Novascone, M. R. Tonks, D. R. Gaston, C. J. Permann, D. Andrs, and R. C. Martineau. Multidimensional multiphysics simulation of nuclear fuel behavior. *Journal of Nuclear Materials*, 423(1–3):149–163, 4 2012.

[44] B. Wohlmuth. A mortar finite element method using dual spaces for the lagrange multiplier. *SIAM Journal on Numerical Analysis*, 38(3):989–1012, 2000.

[45] B. Wohlmuth. Variationally consistent discretization schemes and numerical algorithms for contact problems. *Acta Numerica*, 20:569–734, 2011.

[46] P. Wriggers. *Computational Contact Mechanics*. Springer, Heidelberg, Germany, second edition, 2006.