

Marc Bonnet, Attilio Frangi,  
Christian Rey

**The finite element method in  
solid mechanics**



# Introduction

---

The ever-increasing development of inexpensive computing resources is an essential component of technological progress. A mere personal computer currently enables one to perform realistic simulations of highly complex physical systems. This, together with matching advances in algorithm and software design, have elevated scientific computing to an essential component of research and industry.

Computational solid mechanics fulfils a variety of needs, such as:

- An often cheaper substitute to experimentation, computing allows accurate predictions for the response of mechanical systems of known characteristics under known excitations, for the purpose of e.g. gaining physical insight or performing parameter studies.
- Computing is also an essential component of design and optimization, as it allows virtual testing of many design variants.
- The industrial manufacturing processes themselves can be simulated, analysed and optimised using computational mechanics methodologies.
- Certification of compliance with regulations or building codes.
- Evaluation of safety or life expectancy of mechanical components subjected to fracture, fatigue, damage, wear, corrosion...
- A posteriori analyses, e.g. for finding the causes of structural accidents
- Analysis of experimental data obtained on complex structures for (e.g. constitutive) identification purposes; solution of inverse problems

Computation is, in particular, intensively used in all branches of industry involving mechanics: automotive, railway, aerospace, civil engineering, energy, microelectronics... It is applied to the analysis and design not only of industry products, but also to the means of their production. Computational mechanics is thus a necessary component in the training and skills of engineers and researchers in engineering sciences.

**Simulation methodologies** The numerical simulation of many mechanical or physical systems may be based upon several possible approaches, including:

- Finite elements (based on weighted-residual formulations of field equations, the virtual work principle, or the stationarity of energy functionals);
- Boundary elements (based on boundary integral equation reformulations of linear field equations);
- Finite differences (where derivatives in field equations are replaced by differential quotients);

- Finite volumes, discontinuous Galerkin methods (based on element-wise integral form of the relevant conservation equations);
- Spectral element methods (based on approximation spaces using high-degree polynomials).

The nature and characteristics of the configuration being analysed then determines which approach is best. It is often the case that several of the above-listed methodologies are employed in combination, especially when multi-physics coupling or space-time formulations are involved.

Among the above-listed methodologies, the finite element method (FEM) is particularly important, as it is used in most cases for the analysis of deformable solids (and possibly other physical properties, e.g. fluid or electromagnetic, as well), to the point of being now the reference approach for modelling structures or materials. This book, aimed at graduate students, engineers and researchers, thus focuses on the FEM for solids and structures.

**Objectives of this book.** This book arises from a course taught since 2004 to last-year students of Ecole Polytechnique (Palaiseau, France). It is an augmented and revised version of a previous edition (Bonnet and Frangi, 2006, in French).

The main goal of this book is to introduce the reader, assumed to be familiar with standard continuum solid mechanics, to the main concepts and methods underlying the FEM as an analysis methodology for deformable solids. Covering the full spectrum of computational solid mechanics, a fast-growing and increasingly diverse field, is impossible within a reasonably-sized book<sup>1</sup>, and not desirable for introducing newcomers to this field. Instead, this book focuses on topics that are at the core of the FEM for solids, mostly aiming at providing the reader with a clear grasp of the essentials, sufficient background for reading and exploiting the research literature on computational solid mechanics, and a working knowledge of the main implementational issues of the FEM. Accordingly, analyses under quasistatic conditions (for either linear and nonlinear constitutive properties) or time-evolutive conditions (diffusion and dynamics, only for linear constitutive properties) are addressed. Even within this somewhat reduced framework, and also for teaching consistency, some choices have been made:

- Most of the book is set within the standard small-strain framework for three-dimensional (and occasionally plane-strain and plane-stress) analyses. Structural models (beams, plates, shells) and the associated FEM concepts, that involve specialized developments and added technicalities, are only hinted at.
- Familiarity with standard concepts of deformable solid mechanics and elasticity theory is assumed. Abbreviated coverage of linear elastic fracture mechanics and elastic-plastic constitutive models are given in an effort to make this book self-sufficient.

---

<sup>1</sup>As evidenced by three-volume reference publications such as the *Encyclopedia of Computational Mechanics* (Wiley, 2004) or the book set by O. C. Zienkiewicz, R. L. Taylor and J. Z. Zhu (Butterworth-Heinemann, 2005)

- The variety of constitutive models also could not possibly be exhaustively covered, and this book is mostly restricted to linear elasticity (the natural, and in practice very important, framework for presenting fundamental FEM concepts) and usual elastic-plastic models (also important in practice, and used as the standard example for explaining how nonlinear constitutive properties are handled in the FEM).
- Mathematical aspects of the FEM (functional analysis framework, convergence theorems, error analysis and evaluation) are restricted to the statement of some important results, given without proof.

The book assumes from the reader a working knowledge of continuum mechanics and, to a lesser extent, of fracture mechanics and plasticity. The latter two topics are briefly reviewed, respectively in Chapters 7 and 9, in an effort to make this book reasonably self-contained.

**Bibliography.** The available published material on the theoretical, mathematical, implementational and practical aspects of the finite element method is truly enormous. The relatively modest bibliography given in this book (pages 345-348) makes no attempt at a thorough bibliographical overview, and the reader will no doubt find that many important references are not cited. Instead, in keeping with the pedagogical nature of this book, the selected references provided include a sample of seminal works and others that are intended to serve as pointers for the reader intent on going deeper into some of the aspects touched upon. Regarding the latter purpose, the three-volume *Encyclopedia of Computational Mechanics* published by J. Wiley and Sons certainly is a most useful reference, as it gives a broad and detailed coverage of computational mechanics.

**Overview of book contents.** The following topics (numbered by chapter) are addressed. They are divided into two parts, respectively devoted to basic concepts and a selection of advanced topics. Accordingly, Part I addresses the FEM for linear static or time-dependent problems:

1. Background on linear elasticity, with emphasis on variational principles for energy functionals. The Galerkin method in general form.
2. The concept of isoparametric finite element; meshes and mesh conformity, the finite element approach to interpolation.
3. The finite element method for linear elastic solids: element matrices, assembly procedure, nodal forces, direct and iterative solvers, handling of essential boundary conditions, post-processing.
4. Thermoelasticity with transient heat transfer; time-marching methods for the heat diffusion equation (explicit and implicit methods, stability).
5. Dynamics of elastic solids; time-marching methods for second-order equations (the Newmark family of algorithms, stability, algorithmic energy conservation).

Then, Part II addresses nonlinear constitutive models, fracture mechanics, and mixed formulations:

6. Treatment of incompressible or quasi-incompressible materials; mixed formulations
7. Linear elastic fracture mechanics: review of fundamental concepts, local approach (using special elements) and global, energy-based approach (energy release rate,  $J$ -integral).
8. Nonlinear material or structural behavior: motivation and examples. Solution algorithms for frictionless contact and nonlinear elasticity. Newton-type algorithms for nonlinear equilibrium problems.
9. Elastic-plastic materials and structures: review of history-dependent constitutive model and equations, implicit local constitutive integration based on radial return, global implicit incremental algorithm based on the Newton-Raphson method and the consistent tangent stiffness.

As a distinguishing feature of this book, each chapter contains not only sections devoted to theory and concepts presented in general terms, but also other sections (interspersed with the former) devoted to detailed description of specific features (e.g. the construction of a specific finite element), annotated Matlab code and/or numerical examples produced with it, or worked-out analytical examples. The latter type of section is materialized by a smaller font and a vertical grey strip on the left.

**Initiation to programming and practical aspects of finite elements.** The various fundamental concepts expounded in this book are implemented in a set of Matlab routines, written by the authors. The Matlab scripting language, to which the reader is expected to be familiar, facilitates a hands-on approach to understanding the workings and implementation of the finite element method through concise and explicit coding. Of course, these teaching-oriented routines are very rudimentary in comparison with finite element codes used in industry or research, and are not meant to reflect the state of the art in this respect.

The MATLAB codes presented in this book are freely downloadable. At the time of writing of this book, they are available from the McGraw-Hill website [www.ateneonline.it/bonnet](http://www.ateneonline.it/bonnet) and from the personal pages of the authors.

The reader is encouraged to get in touch with the authors<sup>2</sup> should they encounter any difficulties in obtaining the codes.

**Colour version of the Figures.** Some of the figures employed in the exercises are reproduced in black & white in the printed version. The original colour version can be freely downloaded from the website [www.ateneonline.it/bonnet](http://www.ateneonline.it/bonnet).

---

<sup>2</sup>[mbonnet@ensta.fr](mailto:mbonnet@ensta.fr), [attilio.frangi@polimi.it](mailto:attilio.frangi@polimi.it), [christian.rey@lmt.ens-cachan.fr](mailto:christian.rey@lmt.ens-cachan.fr)

# Contents

---

<b>Introduction</b>	<b>v</b>
<b>I Basic concepts</b>	<b>1</b>
<b>1 Elastic equilibrium problems and their approximate solution</b>	<b>3</b>
1.1 Equilibrium equations for an elastic solid . . . . .	4
1.2 Virtual work principle. Weak formulation of equilibrium . . . . .	7
1.3 Minimization principles. Variational formulation of equilibrium . . . . .	9
1.4 Approximate minimization: the Galerkin method . . . . .	12
1.5 Application: pressurized spherical shell . . . . .	19
1.6 Scalar problems for conductive media . . . . .	28
1.7 Supplementary material . . . . .	30
1.8 Summary . . . . .	32
<b>2 Isoparametric elements</b>	<b>33</b>
2.1 Example: plane strain analyses and linear triangles . . . . .	33
2.2 The concept of isoparametric element . . . . .	39
2.3 Applications and exercises . . . . .	53
<b>3 The finite element method for linear problems</b>	<b>57</b>
3.1 The finite element method based on isoparametric elements . . . . .	57
3.2 Element integrals . . . . .	61
3.3 Assemblage . . . . .	72
3.4 Numerical solution of the linear system . . . . .	75
3.5 Convergence . . . . .	82
3.6 Code genlin . . . . .	88
3.7 Extensions and exercises . . . . .	97
<b>4 Transient heat conduction and linear thermoelasticity</b>	<b>115</b>
4.1 Transient heat conduction: finite element semi-discretization . . . . .	115
4.2 Transient heat conduction: discrete time integration . . . . .	119
4.3 Linear thermoelasticity . . . . .	133
4.4 Applications and exercises . . . . .	138

<b>5</b>	<b>Dynamical analysis of elastic solids</b>	<b>145</b>
5.1	General notions on the dynamics of elastic solids . . . . .	145
5.2	Weak formulation and finite element semi-discretization . . . . .	148
5.3	Modal analysis . . . . .	151
5.4	Direct integration algorithms: the Newmark family . . . . .	157
5.5	Conservation of mechanical energy . . . . .	173
5.6	The code dynamics for 2D and 3D applications . . . . .	176
5.7	Introduction to the spectral element method . . . . .	178
5.8	Additional examples and exercises . . . . .	179
<b>II</b>	<b>Advanced topics</b>	<b>187</b>
<b>6</b>	<b>Penalty approaches, mixed formulations</b>	<b>189</b>
6.1	The incompressibility constraint . . . . .	190
6.2	Mixed functionals . . . . .	201
6.3	Bending of slender structures . . . . .	209
6.4	Diffusion-transport equation . . . . .	215
<b>7</b>	<b>Application to linear elastic fracture mechanics</b>	<b>219</b>
7.1	Linear elastic fracture mechanics . . . . .	219
7.2	Purpose of computational fracture mechanics . . . . .	224
7.3	Numerical evaluation of stress intensity factors . . . . .	225
7.4	Numerical computation of energy release rates . . . . .	235
<b>8</b>	<b>Introduction to nonlinear solid mechanics. Nonlinear elasticity</b>	<b>245</b>
8.1	Overview and illustrations of nonlinear structural behaviors . . . . .	245
8.2	Newton-type iterative methods . . . . .	250
8.3	Nonlinear elasticity . . . . .	255
8.4	Applications and exercises . . . . .	266
<b>9</b>	<b>Numerical elastoplasticity</b>	<b>275</b>
9.1	Small-strain elastoplastic behavior: a review . . . . .	276
9.2	Numerical elasto-plasticity: problem formulation . . . . .	281
9.3	Local integration of elastoplastic constitutive relations . . . . .	284
9.4	Example: illustration of the radial return algorithm . . . . .	291
9.5	Global equilibrium: linearized weak formulation . . . . .	295
9.6	Global equilibrium: discretization . . . . .	304
9.7	Algorithmic issues . . . . .	307
9.8	The code plasticity for plane-strain elastoplastic analysis . . . . .	310
9.9	Example: illustration of algorithms on a notched specimen . . . . .	314



<b>Appendices</b>	<b>319</b>
<b>A Gaussian quadrature rules</b>	<b>321</b>
A.1 Gauss-Legendre points for the segment $[-1, 1]$ . . . . .	321
A.2 Gauss points and weights for the reference triangle . . . . .	322
A.3 Gauss points and weights for the reference tetrahedron . . . . .	322
A.4 Gauss-Lobatto points and weights for the segment $[-1, 1]$ . . . . .	323
<b>B Introduction to the use of the codes developed in this book</b>	<b>325</b>
B.1 Geometry and mesh files . . . . .	326
B.2 Analysis file for genlin . . . . .	331
B.3 Reading the input files . . . . .	333
B.4 Graphical post-processing . . . . .	337
B.5 Geometry files for fracture mechanics problems . . . . .	341
B.6 Code genlin_fast . . . . .	342
<b>References</b>	<b>345</b>
<b>Index</b>	<b>349</b>



## **Part I**

# **Basic concepts**



## Elastic equilibrium problems and their approximate solution

---

Mechanical structures may exhibit very diverse behaviors and responses, depending on the properties of their constitutive materials, the applied loadings, the way structural elements are linked or assembled... The concept of finite element provides a generic framework that allows the design and implementation of approximate methods for the computation of structural responses. This framework has the necessary flexibility for addressing the most complex geometrical configurations and the whole spectrum of material and structural behavior.

The simplest constitutive model is that of linear elasticity under the small-deformation assumption (SDA). It is described in detail in all textbooks on continuum solid mechanics, e.g. Salençon (2001). Many materials conform to this model under "moderate" loading conditions that are such that the stress is at any point within an *elasticity domain* whose precise definition depends on the material (Besson et al., 2009). The elastic linear constitutive model within the SDA is very important in applications because many structures must in practice be designed so that their response remains linearly elastic under all "normal" loading conditions.

It is thus sensible to begin the exposition of approximate solution methods for the mechanics of deformable solids by initially restricting it to the case of linear elasticity under the SDA, which is the only situation where *all* governing equations are linear. This introductory chapter aims, within this framework, at laying the necessary background for developing the finite element method. The starting point is the set of local governing equations for three-dimensional linear elasticity (Section 1.1), to which the reader is assumed to be familiar. The virtual work principle (Section 1.2) is then used for deriving variational and weak formulations. In particular, the well-known variational energy principles of linear elasticity (Salençon, 2001; Oden and Reddy, 1976) are rederived using the concept of error in constitutive relation (Section 1.3). The Galerkin method is then introduced as a generic procedure allowing the construction of approximate solutions to either the variational or the weak formulation (section 1.4), with the link between those two kinds of formulations clarified in the process, and a simple example with spherical symmetry is presented next (Sec. 1.5). The case of scalar conductivity-type problems, mathematically similar to (and simpler than) linear elasticity, is addressed in Section 1.6.

## 1.1 Equilibrium equations for an elastic solid

Consider a solid body whose undeformed configuration occupies the open<sup>1</sup> domain  $\Omega \subset \mathbb{R}^3$ . The solid material is assumed to have a linearly elastic behavior and to follow the small deformation assumption (SDA).

### 1.1.1 Field equations and boundary conditions

Assume that the solid body is subjected to a volume force density  $\mathbf{f}$  in  $\Omega$  and a surface force density  $\mathbf{T}^D$  on a part  $S_T$  of its boundary  $\partial\Omega$  while being constrained by prescribed displacements  $\mathbf{u}^D$  over the remaining part  $S_u$  of  $\partial\Omega$  (Fig. 1.1). Under the foregoing assumptions and conditions, its equilibrium state is characterized at any  $\mathbf{x} \in \Omega$  by the displacement vector  $\mathbf{u}(\mathbf{x})$ , the linearized strain tensor  $\boldsymbol{\varepsilon}(\mathbf{x})$  and the Cauchy stress tensor  $\boldsymbol{\sigma}(\mathbf{x})$ , which verify the field equations

$$\boldsymbol{\varepsilon}(\mathbf{x}) = \frac{1}{2}(\nabla \mathbf{u} + \nabla^T \mathbf{u})(\mathbf{x}) \quad (\mathbf{x} \in \Omega), \quad (1.1a)$$

$$\operatorname{div} \boldsymbol{\sigma}(\mathbf{x}) + \rho \mathbf{f}(\mathbf{x}) = 0 \quad (\mathbf{x} \in \Omega), \quad (1.1b)$$

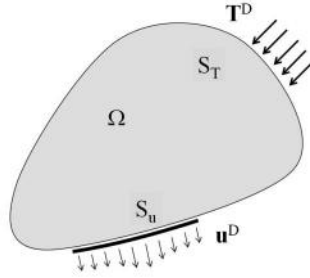
$$\boldsymbol{\sigma}(\mathbf{x}) = \mathcal{A} : \boldsymbol{\varepsilon}(\mathbf{x}) \quad (\mathbf{x} \in \Omega) \quad (1.1c)$$

and the boundary conditions

$$\mathbf{u}(\mathbf{x}) = \mathbf{u}^D(\mathbf{x}) \quad (\mathbf{x} \in S_u), \quad (1.1d)$$

$$\boldsymbol{\sigma}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = \mathbf{T}^D(\mathbf{x}) \quad (\mathbf{x} \in S_T). \quad (1.1e)$$

The surfaces  $S_u$  and  $S_T$  must necessarily define a partition of  $\partial\Omega$ , that is, must verify  $S_u \cup \overline{S_T} = \partial\Omega$ ,  $S_u \cap S_T = \emptyset$ . The boundary conditions (1.1d,e) are then said to be *well-posed*, in that they guarantee existence and uniqueness of the strain and stress solving problem (1.1a–e). To also ensure uniqueness of the displacement  $\mathbf{u}$ , it is in addition necessary that the kinematical data prevent all potential



**Figure 1.1:** Equilibrium of an elastic solid body: notations.

<sup>1</sup>We follow the standard mathematical convention according to which spatial domains  $\Omega$  are open sets. The *closure*  $\overline{\Omega}$  of  $\Omega$  is thus  $\overline{\Omega} = \Omega \cup \partial\Omega$ , with  $\Omega \cap \partial\Omega = \emptyset$ .

infinitesimal rigid-body motions (see Salençon, 2001), i.e. displacements of the form

$$\mathbf{v} = \mathbf{a} + \mathbf{b} \wedge \mathbf{x}, \quad \varepsilon[\mathbf{v}] = \mathbf{0} \quad (1.2)$$

(where  $\mathbf{a} \in \mathbb{R}^3$  and  $\mathbf{b} \in \mathbb{R}^3$  define a translation and an infinitesimal rotation, respectively) for which the linearized strain vanishes. Such rigid-body motions are avoided whenever the surface  $S_u$  has nonzero surface area.

Conditions (1.1d,e) are not the only possibilities: one may for example prescribe conditions that enforce symmetry (see Section 1.7.1) or periodicity (relationships linking displacements on two opposite sides). In all cases, the boundary conditions must be such that three linearly independent relationships between the displacement vector and the traction vector  $\mathbf{T} := \boldsymbol{\sigma} \cdot \mathbf{n}$  are enforced at any point of  $\partial\Omega$  (with prescribed displacement or tractions consider as special forms of such relationships).

Equations (1.1a–e) are sometimes collectively referred to as the *strong formulation* of the elastic equilibrium problem (in contrast with *weak formulations* introduced in Sections 1.2 and 1.3). As is the case for any continuum mechanics model, they can be split into three fundamental types of equations:

- (i) Kinematical compatibility equations (1.1a,d);
- (ii) Equilibrium (or balance) equations (1.1b,e);
- (iii) Constitutive equation (1.1c).

### 1.1.2 Sets of admissible fields

To emphasize and implement the above distinction, it is useful to introduce sets of fields that are consistent with given displacement or force data. The set  $\mathcal{C}(\mathbf{u}^D)$  of displacements that are kinematically admissible with respect to prescribed displacements  $\mathbf{u}^D$  is thus defined by

$$\mathcal{C}(\mathbf{u}^D) = \{ \mathbf{v} \mid \mathbf{v} \text{ sufficiently regular in } \Omega \text{ and } \mathbf{v} = \mathbf{u}^D \text{ on } S_u \}, \quad (1.3)$$

while the set  $\mathcal{S}(\mathbf{T}^D, \mathbf{f})$  of stress fields that are *statically admissible* with applied body and surface forces  $(\mathbf{T}^D, \mathbf{f})$  is defined by

$$\mathcal{S}(\mathbf{T}^D, \mathbf{f}) = \{ \boldsymbol{\tau} \mid \operatorname{div} \boldsymbol{\tau} + \rho \mathbf{f} = \mathbf{0} \text{ in } \Omega, \boldsymbol{\tau} \cdot \mathbf{n} = \mathbf{T}^D \text{ on } S_T \}. \quad (1.4)$$

It is convenient to also introduce the set  $\mathcal{C}(\mathbf{0})$  of displacements that are kinematically admissible with respect to prescribed zero displacements:

$$\mathcal{C}(\mathbf{0}) = \{ \mathbf{w} \mid \mathbf{w} \text{ sufficiently regular in } \Omega \text{ and } \mathbf{w} = \mathbf{0} \text{ on } S_u \} \quad (1.5)$$

and to simply denote by  $\mathcal{C}$  the set of admissible displacements which are not constrained on the boundary:

$$\mathcal{C} = \{ \mathbf{w} \mid \mathbf{w} \text{ sufficiently regular in } \Omega \}. \quad (1.6)$$

Definitions (1.3), (1.5) and (1.6) postulate some regularity for the displacements fields, aiming to avoid "pathological", and in any case unphysical, situations. The smoothness requirements on displacement include continuity (i.e. the material is assumed to deform without breaking<sup>2</sup>). In fact, the strong formulation (1.1a–e) requires that displacements be twice continuously piecewise differentiable<sup>3</sup>. For the finite element method, based (as we will see) on a weak formulation, less stringent smoothness requirements on the displacements are sufficient:

$$\mathbf{v} \text{ and } \nabla \mathbf{v} \text{ are square-integrable on } \Omega. \quad (1.7)$$

As we will see in Section 1.3, this condition<sup>4</sup> essentially amounts to considering only displacements whose elastic strain energy is finite.

The elastic equilibrium problem (1.1a–e) may then be more compactly reformulated as

$$\text{find } (\mathbf{u}, \boldsymbol{\sigma}) \in \mathcal{C}(\mathbf{u}^D) \times \mathcal{S}(\mathbf{T}^D, \mathbf{f}) \text{ such that } \boldsymbol{\sigma} = \mathcal{A} : \boldsymbol{\varepsilon}[\mathbf{u}] \text{ (in } \Omega), \quad (1.8)$$

where the notation  $\boldsymbol{\varepsilon}[\mathbf{v}]$  stands for the linearized strain  $\boldsymbol{\varepsilon}$  associated to a given displacement  $\mathbf{v}$ , evaluated according to (1.1a). In other words, one seeks the kinematically admissible displacement field  $\mathbf{u}$  and the statically admissible stress field  $\boldsymbol{\sigma}$  (both being unique) which are linked by the elastic constitutive relation (1.1c).

### 1.1.3 Properties of the elastic constitutive relation

The fourth-order tensor  $\mathcal{A}$  featured in the constitutive relation (1.1c), called *elastic stiffness tensor* or *elasticity tensor* has the symmetry properties

$$\mathcal{A}_{ijkl} = \mathcal{A}_{jikl} = \mathcal{A}_{klij} \quad (1.9)$$

and defines a positive definite quadratic form over the symmetric second-order tensors:

$$\boldsymbol{\varepsilon} : \mathcal{A} : \boldsymbol{\varepsilon} > 0, \quad \forall \boldsymbol{\varepsilon}, \|\boldsymbol{\varepsilon}\| \neq 0 \text{ and } \boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^T. \quad (1.10)$$

In view of the symmetry properties (1.9), a general anisotropic constitutive behavior involves at most 21 independent elastic parameters. Isotropic materials are characterized by just two independent elastic parameters, and the tensor  $\mathcal{A}$  may be written in several ways in terms of usual isotropic elasticity parameters, e.g.:

$$\mathcal{A} = 3\lambda \mathcal{J} + 2\mu \mathcal{I} = 2\mu \left( \frac{3\nu}{1-2\nu} \mathcal{J} + \mathcal{I} \right) = 3\kappa \mathcal{J} + 2\mu \mathcal{K}. \quad (1.11)$$

In these expressions,  $(\lambda, \mu)$  are the Lamé constants,  $\nu$  is Poisson's ratio,  $\kappa$  is the bulk modulus ( $3\kappa = 3\lambda + 2\mu$ ), while the fourth-order tensors  $\mathcal{I}$ ,  $\mathcal{J}$ ,  $\mathcal{K}$  are defined as follows:

<sup>2</sup>Care will therefore be needed for extending such definitions for cracked solids, see Chapter 7.

<sup>3</sup>The "piecewise" proviso allowing for strain discontinuities through selected surfaces, e.g. across interfaces between bonded dissimilar interfaces.

<sup>4</sup>The set  $\mathcal{C}$  is then known in functional analysis as the Sobolev space  $H^1(\Omega)$  (Allaire, 2007).



- The tensor  $\mathcal{I}$  is associated with the identity for symmetric second-order tensors:

$$\mathcal{I}_{ijkl} = \frac{1}{2}(\delta_{ik}\delta_{jl} + \delta_{jk}\delta_{il}); \quad (1.12)$$

- The tensor  $\mathcal{J}$  is associated with the projection onto the subspace of isotropic second-order tensors:

$$\mathcal{J} = \frac{1}{3}\mathbf{I} \otimes \mathbf{I} \quad \text{i.e., using components: } \mathcal{J}_{ijkl} = \frac{1}{3}\delta_{ij}\delta_{kl}; \quad (1.13)$$

- The tensor  $\mathcal{K}$  is associated with the projection onto the supplementary subspace of deviatoric second-order symmetric tensors (i.e. of tensors with vanishing trace):

$$\mathcal{K} = \mathcal{I} - \mathcal{J}. \quad (1.14)$$

Those tensors verify the identities

$$\mathcal{J}:\mathcal{J} = \mathcal{J}, \quad \mathcal{K}:\mathcal{K} = \mathcal{K}, \quad \mathcal{J}:\mathcal{K} = 0. \quad (1.15)$$

In this case the constitutive equation (1.1c) takes the simple forms:

$$\boldsymbol{\sigma} = \lambda \text{Tr} \boldsymbol{\varepsilon} \mathbf{I} + 2\mu \boldsymbol{\varepsilon} = \kappa \text{Tr} \boldsymbol{\varepsilon} \mathbf{I} + 2\mu \boldsymbol{e} \quad (1.16)$$

where  $\boldsymbol{e}$  is the strain deviator tensor defined by  $\boldsymbol{e} = \boldsymbol{\varepsilon} - (1/3)\text{Tr} \boldsymbol{\varepsilon} \mathbf{I}$ .

Finally, the elastic compliance tensor  $\mathcal{S}$ , corresponding to the constitutive relation (1.1c) in inverted form, is defined by

$$\boldsymbol{\sigma} = \mathcal{A}:\boldsymbol{\varepsilon} \iff \boldsymbol{\varepsilon} = \mathcal{S}:\boldsymbol{\sigma}, \quad \text{with } \mathcal{A}:\mathcal{S} = \mathcal{S}:\mathcal{A} = \mathcal{I}. \quad (1.17)$$

This fourth-order tensor obeys the same symmetry properties (1.9) as  $\mathcal{A}$ , and is also positive definite:

$$\boldsymbol{\tau}:\mathcal{S}:\boldsymbol{\tau} > 0, \quad \forall \boldsymbol{\tau} \parallel \boldsymbol{\tau} \neq 0 \text{ and } \boldsymbol{\tau} = \boldsymbol{\tau}^T. \quad (1.18)$$

## 1.2 Virtual work principle. Weak formulation of equilibrium

The local balance equation (1.1b) may be recast into an equivalent integral form by means of dualisation, i.e. inner multiplication by an arbitrary field  $\boldsymbol{w} \in \mathcal{C}$  followed by integration over  $\Omega$ . With the help of the divergence theorem, this yields the *weak form* of the local balance equation:

$$\int_{\Omega} \boldsymbol{\sigma}:\boldsymbol{\varepsilon}[\boldsymbol{w}] \, dV = \int_{\Omega} \rho \boldsymbol{f} \cdot \boldsymbol{w} \, dV + \int_{\partial\Omega} \boldsymbol{T} \cdot \boldsymbol{w} \, dS \quad \forall \boldsymbol{w} \in \mathcal{C}. \quad (1.19)$$

This weak formulation of equilibrium between internal forces (characterised by the stress tensor) and externally applied loads in fact agrees with the virtual work principle applied to equilibrium (with the left- and right-hand sides of equation (1.19) respectively corresponding to the virtual work of internal forces – with a sign change – and of the external loads). It is important to emphasize that the surface integral gives the virtual work of *all* forces exerted on the boundary of the solid body, irrespective of any partition ( $S_u, S_T$ ) of the boundary used in the boundary conditions.

One can then substitute into the virtual work equality (1.19) the relationship

$$\sigma(x) = \mathcal{A}:\varepsilon[u](x) \quad (x \in \Omega), \quad (1.20)$$

which results from combining the local compatibility (1.1a) and the constitutive relation (1.1c), and the boundary condition (1.1e), to obtain

$$\int_{\Omega} \boldsymbol{\varepsilon}[u] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[w] \, dV = \int_{\Omega} \rho \boldsymbol{f} \cdot \boldsymbol{w} \, dV + \int_{S_T} \boldsymbol{T}^D \cdot \boldsymbol{w} \, dS + \int_{S_u} [\boldsymbol{\sigma} \cdot \boldsymbol{n}] \cdot \boldsymbol{w} \, dS \quad \forall \boldsymbol{w} \in \mathcal{C} \quad (1.21)$$

The above identity has the following notable features: (i) it makes no reference to the kinematic data (1.1d), and (ii) it involves the traction vector  $\mathbf{T} := \boldsymbol{\sigma} \cdot \mathbf{n}$  on  $S_u$ , which is *a priori* unknown and represents reactions caused by prescribing displacements on  $S_u$ . In view of these remarks, one may reformulate the elastic equilibrium problem (1.1a–e) in either of two ways from the virtual work equality (1.21), depending on whether one prefers to eliminate the reaction or to retain the kinematic data (1.1d).

**First variant: weak formulation obtained by reaction elimination.** To eliminate the reaction  $T$  over  $S_u$  from the weak formulation, one only has to restrict (1.21) to the virtual fields that are compatible with zero boundary data, i.e.  $\boldsymbol{w} \in \mathcal{C}(\mathbf{0})$ . The kinematic data (1.1d) is then accounted for through the definition of the set  $\mathcal{C}(\boldsymbol{u}^D)$  of admissible displacements, among which  $\boldsymbol{u}$  is sought. One thus arrives at the following *weak formulation* for the equilibrium problem (1.1a–e):

$$\begin{aligned} & \text{find } \mathbf{u} \in \mathcal{C}(\mathbf{u}^D) \text{ such that} \\ & \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (1.22) \end{aligned}$$

**Second variant: weak formulation obtained by kinematic data incorporation.** One may instead prefer to retain the reaction  $\mathbf{T}$  over  $S_u$  in the final weak formulation. The latter is then obtained by supplementing (1.21) with condition (1.1d) written in weak form, using a "virtual traction vector"  $\mathbf{T}'$  that is allowed to have arbitrary values on  $S_u$ . It reads:

$$\begin{aligned} & \text{find } (\boldsymbol{u}, \boldsymbol{T}) \in \mathcal{C} \times \mathcal{C}'[S_u] \text{ such that} \\ & \int_{\Omega} \varepsilon[\boldsymbol{u}] : \boldsymbol{\mathcal{A}} : \varepsilon[\boldsymbol{w}] \, dV - \int_{S_u} \boldsymbol{T} \cdot \boldsymbol{w} \, dS = \int_{\Omega} \rho \boldsymbol{f} \cdot \boldsymbol{w} \, dV + \int_{S_T} \boldsymbol{T}^D \cdot \boldsymbol{w} \, dS \\ & \qquad \qquad \qquad \forall \boldsymbol{w} \in \mathcal{C}, \tag{1.23} \\ & \int_{S_u} \boldsymbol{u} \cdot \boldsymbol{T}' \, dS = \int_{S_u} \boldsymbol{u}^D \cdot \boldsymbol{T}' \, dS \quad \forall \boldsymbol{T}' \in \mathcal{C}'[S_u], \end{aligned}$$

where the set  $\mathcal{C}'[S_u]$  contains all traction vector fields on  $S_u$  defined by duality with respect to the elements of  $\mathcal{C}$ , i.e. such that the integrals over  $S_u$  in (1.23) are defined for any displacement field belonging to  $\mathcal{C}$ .

**Remark.** The approach whereby a weak formulation is constructed by substituting local compatibility and constitutive equations into the balance equation in weak form (i.e. a virtual work identity), introduced here within the limited framework of linear elasticity, has in fact much more generality. It is in particular used repeatedly in this book, for instance in connection with dynamics (Chapter 5) or non-linear constitutive properties (Chapter 8).

### 1.3 Minimization principles. Variational formulation of equilibrium

In some cases, and in particular when considering (as done in this chapter) the equilibrium of elastic solids, the solution may alternatively be characterized by the fact that it achieves the minimization of an energy. In fact, early developments of the finite element method were based on that viewpoint. This section explains the energy-minimization approach.

#### 1.3.1 Error in constitutive relation

Recall that the governing equations of solid mechanics are divided into three fundamental groups (compatibility, balance, constitutive behavior), see Section 1.1.1. Let the *error in constitutive relation* functional  $\mathcal{E}(\mathbf{v}, \boldsymbol{\tau})$  be defined by

$$\mathcal{E}(\mathbf{v}, \boldsymbol{\tau}) = \frac{1}{2} \int_{\Omega} (\boldsymbol{\tau} - \mathcal{A}:\boldsymbol{\varepsilon}[\mathbf{v}]) : \mathcal{S} : (\boldsymbol{\tau} - \mathcal{A}:\boldsymbol{\varepsilon}[\mathbf{v}]) \, dV, \quad (1.24)$$

where  $\mathbf{v}$  is a displacement field,  $\boldsymbol{\tau}$  a stress field and  $\mathcal{S}$  is the elastic compliance tensor defined by (1.17). This functional evaluates, in the energy norm sense, the constitutive equation gap between a given displacement field  $\mathbf{v}$  and a given stress field  $\boldsymbol{\tau}$ . Moreover, since  $\mathcal{S}$  is positive definite, the functional  $\mathcal{E}$  has the following properties:

$$\begin{aligned} \mathcal{E}(\mathbf{v}, \boldsymbol{\tau}) &\geq 0 \quad \forall (\mathbf{v}, \boldsymbol{\tau}), \\ \mathcal{E}(\mathbf{v}, \boldsymbol{\tau}) &= 0 \Rightarrow \boldsymbol{\tau} - \mathcal{A}:\boldsymbol{\varepsilon}[\mathbf{v}] = 0 \text{ in } \Omega. \end{aligned} \quad (1.25)$$

The statement (1.8) of the elastic equilibrium problem can then be reformulated in terms of  $\mathcal{E}$ :

$$\text{find } (\mathbf{u}, \boldsymbol{\sigma}) \in \mathcal{C}(\mathbf{u}^D) \times \mathcal{S}(\mathbf{T}^D, \mathbf{f}) \quad \text{such that} \quad \mathcal{E}(\mathbf{u}, \boldsymbol{\sigma}) = 0. \quad (1.26)$$

This reformulation and properties (1.25) together imply that the solution  $(\mathbf{u}, \boldsymbol{\sigma})$  of the elastic equilibrium problem minimizes the constitutive relation error. This remark allows to seek  $(\mathbf{u}, \boldsymbol{\sigma})$  by solving the minimization problem

$$\begin{aligned} \text{find } (\mathbf{u}, \boldsymbol{\sigma}) \in \mathcal{C}(\mathbf{u}^D) \times \mathcal{S}(\mathbf{T}^D, \mathbf{f}) \quad \text{such that} \\ \mathcal{E}(\mathbf{u}, \boldsymbol{\sigma}) \leq \mathcal{E}(\mathbf{v}, \boldsymbol{\tau}) \quad \forall (\mathbf{v}, \boldsymbol{\tau}) \in \mathcal{C}(\mathbf{u}^D) \times \mathcal{S}(\mathbf{T}^D, \mathbf{f}). \end{aligned} \quad (1.27)$$

For  $(\mathbf{u}, \boldsymbol{\sigma})$  to solve a well-posed elastic equilibrium problem, one must have  $\mathcal{E}(\mathbf{u}, \boldsymbol{\sigma}) = 0^5$ .

---

<sup>5</sup>If the boundary conditions are well-posed (see Section 1.1), problem (1.1a–e) is uniquely

The concept of error in constitutive relation can be extended to non-linear constitutive models (Ladevèze et al., 1999; Ladevèze, 1999); this generalization is outside the scope of this book.

### 1.3.2 Potential and complementary energies

In the minimization problem (1.27), both the displacement and the stress are *a priori* sought. This problem can however be very easily uncoupled into two independent minimization problems whose respective unknown is the displacement and the stress. To see this, the error in constitutive relation functional (1.24) is written in expanded form as

$$\mathcal{E}(\mathbf{v}, \boldsymbol{\tau}) = \frac{1}{2} \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{v}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{v}] \, dV + \frac{1}{2} \int_{\Omega} \boldsymbol{\tau} : \boldsymbol{\mathcal{S}} : \boldsymbol{\tau} \, dV - \int_{\Omega} (\boldsymbol{\tau} : \boldsymbol{\varepsilon}[\mathbf{v}]) \, dV.$$

The first term depends only on the displacement, and the second only on the stress. Moreover, the version (1.19) of the virtual work identity holds for any  $\boldsymbol{\tau} \in \mathcal{S}(\mathbf{T}^D, \mathbf{f})$ , and  $\mathbf{w} = \mathbf{v}$  may be used therein as a virtual field, allowing to recast the third term as

$$\begin{aligned} \int_{\Omega} \boldsymbol{\tau} : \boldsymbol{\varepsilon}[\mathbf{v}] \, dV &= \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{v} \, dV + \int_{\partial\Omega} [\boldsymbol{\tau} \cdot \mathbf{n}] \cdot \mathbf{v} \, dS \\ &= \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{v} \, dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{v} \, dS + \int_{S_u} [\boldsymbol{\tau} \cdot \mathbf{n}] \cdot \mathbf{u}^D \, dS, \end{aligned}$$

where the second equality uses the admissibility conditions (1.1d,e). The error in constitutive relation  $\mathcal{E}(\mathbf{v}, \boldsymbol{\tau})$  is thus found to be decomposed into the sum of a *potential energy*  $\mathcal{P}(\mathbf{v})$  and a *complementary energy*  $\mathcal{P}^*(\boldsymbol{\tau})$ :

$$\mathcal{E}(\mathbf{v}, \boldsymbol{\tau}) = \mathcal{P}(\mathbf{v}) + \mathcal{P}^*(\boldsymbol{\tau}) \quad (1.28)$$

where  $\mathcal{P}(\mathbf{v})$  and  $\mathcal{P}^*(\boldsymbol{\tau})$  are defined by

$$\mathcal{P}(\mathbf{v}) = \mathcal{W}(\mathbf{v}) - \mathcal{F}(\mathbf{v}), \quad (1.29)$$

$$\mathcal{P}^*(\boldsymbol{\tau}) = \mathcal{W}^*(\boldsymbol{\tau}) - \mathcal{F}^*(\boldsymbol{\tau}), \quad (1.30)$$

---

solvable (possibly up to an arbitrary additive rigid-body displacement), so that any solution of the minimization problem (1.27) verifies  $\mathcal{E}(\mathbf{u}, \boldsymbol{\sigma}) = 0$ .

In certain situations, which are outside the scope of this book, *overdetermined* boundary data are available (e.g. from experiments), so that the displacement *and* the traction are simultaneously known on some part of the boundary. One still may in such cases define and minimize an error in constitutive relation functional. The minimization problem (1.27) may then yield a solution  $(\mathbf{u}, \boldsymbol{\sigma})$  such that  $\mathcal{E}(\mathbf{u}, \boldsymbol{\sigma}) > 0$ . This indicates that the available model of the solid body, characterised by  $\Omega$  and  $\boldsymbol{\mathcal{A}}$ , is inconsistent with the overdetermined boundary data. When the latter results from experiments, and hence is representative of the actual solid,  $\mathcal{E}(\mathbf{u}, \boldsymbol{\sigma}) > 0$  exposes the fact that the model  $(\Omega, \boldsymbol{\mathcal{A}})$  does not correctly reflect the physical reality. This may be caused by e.g. inadequate knowledge of elastic properties, hidden flaws such as cracks or cavities which are unaccounted in the geometric model  $\Omega$  (see the review paper by Bonnet and Constantinescu, 2005, and the references therein), incompatible boundary data (Baranger and Andrieux, 2006)...

with

$$\begin{aligned}\mathcal{W}(\mathbf{v}) &:= \frac{1}{2} \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{v}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{v}] \, dV, & \mathcal{F}(\mathbf{v}) &:= \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{v} \, dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{v} \, dS, \\ \mathcal{W}^*(\boldsymbol{\tau}) &:= \frac{1}{2} \int_{\Omega} \boldsymbol{\tau} : \boldsymbol{\mathcal{S}} : \boldsymbol{\tau} \, dV, & \mathcal{F}^*(\boldsymbol{\tau}) &:= \int_{S_u} [\boldsymbol{\tau} \cdot \mathbf{n}] \cdot \mathbf{u}^D \, dS.\end{aligned}$$

### 1.3.3 Energy minimization, variational formulations

Decomposition (1.28) clearly shows that the displacement and stress solving a given elastic equilibrium problem can be determined separately, by solving two independent problems. The displacement solution  $\mathbf{u}$  is found by minimizing the potential energy among all kinematically admissible displacements:

$$\mathbf{u} = \arg \min_{\mathbf{v} \in \mathcal{C}(\mathbf{u}^D)} \mathcal{P}(\mathbf{v}) \quad (1.31)$$

while the stress solution  $\boldsymbol{\sigma}$  is found by minimizing the complementary energy among all statically admissible stresses:

$$\boldsymbol{\sigma} = \arg \min_{\boldsymbol{\tau} \in \mathcal{S}(\mathbf{T}^D, \mathbf{f})} \mathcal{P}^*(\boldsymbol{\tau}) \quad (1.32)$$

(the notation  $x = \arg \min_y f(y)$  means that  $x$  is a value of the argument  $y$  such that  $f(y)$  is minimized). Recall that  $(\mathbf{u}, \boldsymbol{\sigma})$  solve a well-posed equilibrium problem only if they verify

$$\mathcal{E}(\mathbf{u}, \boldsymbol{\sigma}) = \mathcal{P}(\mathbf{u}) + \mathcal{P}^*(\boldsymbol{\sigma}) = 0.$$

**Stationarity equation for the potential energy.** The variation of the potential energy  $\mathcal{P}$  about an admissible field  $\mathbf{v}$  may be written in the form

$$\mathcal{P}(\mathbf{v} + \eta \mathbf{w}) - \mathcal{P}(\mathbf{v}) = \eta \langle \mathcal{P}'(\mathbf{v}), \mathbf{w} \rangle + o(\eta) \quad (\text{with } \mathbf{w} \in \mathcal{C}(\mathbf{0})), \quad (1.33)$$

where the variation  $\eta \mathbf{w}$  of  $\mathbf{v}$ , with amplitude  $\eta \geq 0$ , must vanish on  $S_u$  so that the perturbed field  $\mathbf{v} + \eta \mathbf{w}$  remains kinematically admissible. The expansion (1.33) defines (if it exists) the tangent linear operator  $\mathcal{P}'(\mathbf{v})$  of  $\mathcal{P}$  at  $\mathbf{v}$ , the notation  $\langle \mathcal{P}'(\mathbf{v}), \mathbf{w} \rangle$  then standing for the directional derivative of  $\mathcal{P}$  at  $\mathbf{v}$  along the direction  $\mathbf{w}$ . The displacement field  $\mathbf{u} \in \mathcal{C}(\mathbf{u}^D)$  that minimizes the potential energy  $\mathcal{P}$  is such that, for any admissible field of the form  $\mathbf{v} = \mathbf{u} + \eta \mathbf{w}$ , the variation of  $\mathcal{P}$  vanishes to first order in  $\eta$ :

$$\left. \frac{d}{d\eta} \mathcal{P}(\mathbf{v} + \eta \mathbf{w}) \right|_{\eta=0} = 0, \quad \text{i.e.} \quad \langle \mathcal{P}'(\mathbf{u}), \mathbf{w} \rangle = 0 \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (1.34)$$

This condition, written in explicit form using definition (1.29), leads to the equation

find  $\mathbf{u} \in \mathcal{C}(\mathbf{u}^D)$  such that

$$\int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}), \quad (1.35)$$

known as the *variational formulation* of the linear elasticity problem (1.1a–e) in that it stems from the variational principle (1.34). One moreover notes that (1.35) coincides with the weak formulation (1.22) stemming from the virtual work principle.

**Weak formulation vs. variational formulation.** The term *weak formulation* refers to the weighted-residual form of a field equation, resulting from its multiplication by a virtual field and integration over the geometrical domain over which the equation is considered. A mathematically more rigorous explanation, besides, is that weak formulations are local equations written in the sense of distributions, which remain meaningful under weaker smoothness requirements than the initial pointwise equations considered in the classical sense. Equation (1.19), which expresses the virtual work principle, is the weak formulation of the local balance equation (1.1b). A weak formulation is referred to as a *variational formulation* if it also expresses the conditions under which some functional (usually of an energetic nature) is stationary. The equilibrium of an elastic solid being (for instance) formulable as the minimization of the potential energy, the weak formulation (1.22) thus coincides with the variational formulation (1.35) associated with this energy minimization principle.

### 1.3.4 Well-posedness of the weak formulation

Considering for simplicity (and without loss of generality) the case where  $\mathbf{u}^D = \mathbf{0}$ , the weak formulation (1.22) belongs to a larger class of problems of the form

$$\text{find } \mathbf{u} \in \mathcal{V} \text{ such that: } \mathcal{A}(\mathbf{u}, \mathbf{w}) = \mathcal{F}(\mathbf{w}), \quad \forall \mathbf{w} \in \mathcal{V} \quad (1.36)$$

where

- (i)  $\mathcal{V}$  is a functional Hilbert space, defined in the case of the weak formulation (1.22) by  $\mathcal{V} = \{\mathbf{w} \in H^1(\Omega), \mathbf{w} = \mathbf{0} \text{ on } S_u\}$  in terms of the Sobolev space  $H^1(\Omega)$  of vector fields that are square-integrable over  $\Omega$  together with their first-order (distributional) derivatives;
- (ii) The bilinear form  $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$  is continuous, i.e. there exists a strictly positive constant  $C_1$  such that  $\mathcal{A}(\mathbf{u}, \mathbf{w}) \leq C_1 \|\mathbf{u}\|_{\mathcal{V}} \|\mathbf{w}\|_{\mathcal{V}}$ ;
- (iii) The bilinear form  $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$  is *coercive*, i.e. there exists a strictly positive constant  $C_2$  such that  $C_2 \|\mathbf{u}\|_{\mathcal{V}} \|\mathbf{w}\|_{\mathcal{V}} \leq \mathcal{A}(\mathbf{u}, \mathbf{w})$ ;
- (iv) The linear form  $\mathcal{F} : \mathcal{V} \rightarrow \mathbb{R}$  is continuous, i.e. there exists a strictly positive constant  $C_3$  such that  $\mathcal{F}(\mathbf{w}) \leq C_3 \|\mathbf{w}\|_{\mathcal{V}}$ .

The Lax-Milgram theorem (see e.g. Ciarlet, 1980) ensures that any weak formulation of the form (1.36) satisfying assumptions (i) to (iv) above has a unique solution  $\mathbf{u} \in \mathcal{V}$ , which moreover depends continuously on the data underlying  $\mathcal{F}$ .

## 1.4 Approximate minimization: the Galerkin method

The sets  $\mathcal{C}(\mathbf{u}^D)$  and  $\mathcal{S}(\mathbf{T}^D, \mathbf{f})$  of admissible fields are infinite-dimensional linear (affine) spaces. It is therefore in general impossible in practice to find the absolute minimum of the potential energy  $\mathcal{P}(\mathbf{v})$  or of the complementary energy  $\mathcal{P}^*(\boldsymbol{\tau})$

due to geometrical complexity of most real mechanical systems. On the other hand, to seek an *approximate* minimum  $\mathcal{P}(\mathbf{v})$  or  $\mathcal{P}^*(\boldsymbol{\tau})$  by restricting the search to finite-dimensional subspaces of admissible field is a perfectly workable approach, known as the Galerkin method.

#### 1.4.1 The Galerkin method for the potential energy

This approach consists in considering the potential energy  $\mathcal{P}(\mathbf{v})$  for displacement fields  $\mathbf{v}$  of the form

$$\mathbf{v}_N(\mathbf{x}) = \mathbf{u}^{(D)}(\mathbf{x}) + \sum_{K=1}^N \alpha_K \boldsymbol{\varphi}^K(\mathbf{x}) \quad \text{with } \mathbf{u}^{(D)} \in \mathcal{C}(\mathbf{u}^D), \boldsymbol{\varphi}^K \in \mathcal{C}(\mathbf{0}), \quad (1.37)$$

where  $\mathbf{u}^{(D)}$  is an arbitrary, sufficiently regular, extension to the whole body  $\Omega$  of the boundary data  $\mathbf{u}^D$  on  $S_u$ <sup>6</sup> and each  $\boldsymbol{\varphi}^K$  vanishes on  $S_u$ , ensuring that any  $\mathbf{v}$  of the form (1.37) belongs to  $\mathcal{C}(\mathbf{u}^D)$ , i.e. is consistent with the kinematic boundary data. The Galerkin method thus consists in choosing *a priori* a representation based on a finite set of functions  $(\mathbf{u}^{(D)}; \boldsymbol{\varphi}^1, \dots, \boldsymbol{\varphi}^N)$ , and using as unknowns the coefficients  $\alpha_K$  of expansion (1.37), often called *generalized displacements*.

The potential energy  $\mathcal{P}(\mathbf{v})$  for any displacement of the form (1.37) is then given by

$$\mathcal{P}(\mathbf{v}) = \frac{1}{2} \{\alpha\}^T [\mathbb{K}] \{\alpha\} - \{\alpha\}^T \{\mathbb{F}\} + \mathcal{P}(\mathbf{u}^{(D)}) = P(\{\alpha\}) \quad (1.38)$$

in terms of the vector  $\{\alpha\} \in \mathbb{R}^N$  of *generalized displacements* (or *degrees of freedom*), such that

$$\{\alpha\} = \{\alpha_1, \dots, \alpha_N\}^T, \quad (1.39)$$

the (square, symmetric) *stiffness matrix*  $[\mathbb{K}] \in \mathbb{R}^{N,N}$ , whose entries are given by

$$[\mathbb{K}]_{IJ} = \int_{\Omega} \boldsymbol{\varepsilon}[\boldsymbol{\varphi}^I] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\boldsymbol{\varphi}^J] dV \quad (1 \leq I, J \leq N), \quad (1.40)$$

and the vector  $\{\mathbb{F}\} \in \mathbb{R}^N$  of *generalized forces*, whose entries are given by

$$\{\mathbb{F}\}_I = - \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}^{(D)}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\boldsymbol{\varphi}^I] dV + \int_{\Omega} \rho \mathbf{f} \cdot \boldsymbol{\varphi}^I dV + \int_{S_T} \mathbf{T}^D \cdot \boldsymbol{\varphi}^I dS \quad (1 \leq I \leq N). \quad (1.41)$$

The Galerkin approach for the potential energy  $\mathcal{P}(\mathbf{v})$  then consists in finding an approximate solution  $\mathbf{u}_N$  by solving the problem

$$\mathbf{u}_N = \arg \min_{\mathbf{v} \in \mathcal{C}_N(\mathbf{u}^D)} \mathcal{P}(\mathbf{v}) \quad (1.42)$$

with

$$\mathcal{C}_N(\mathbf{0}) = \text{span}\{\boldsymbol{\varphi}^K, K = 1, \dots, N\} \subset \mathcal{C}(\mathbf{0}), \quad \mathcal{C}_N(\mathbf{u}^D) = \mathbf{u}^{(D)} + \mathcal{C}_N(\mathbf{0})$$

---

<sup>6</sup> $\mathbf{u}^{(D)}$  may well vanish in part of  $\Omega$ , but must nevertheless be understood as being defined over the whole domain.

Equation (1.38) introduces a notational convention often used in engineering publications on the finite element method, and which will be followed throughout this book. This convention consists in enclosing within brackets all vectors and matrices associated to the finite-dimensional approximation of the unknown fields (e.g. the generalized displacements  $\{\alpha\}$  or the stiffness matrix  $[\mathbb{K}]$ ), with curly brackets  $\{\cdot\}$  used for vectors and square brackets  $[\cdot]$  for matrices. It aims at making a clear distinction between (i) vectors and tensors defined in relation with the physical space (e.g. position vector  $\mathbf{x}$ , displacement  $\mathbf{v}$ , strain  $\boldsymbol{\varepsilon}$ ,...), and (ii) one- or two-dimensional arrays defined in relation with the abstract (linear or affine) finite-dimensional approximation space generated by a chosen set of basis functions on which the Galerkin method is based. Moreover, as suggested by (1.39),  $\{\alpha\}$  denotes a column N-vector and its transposed counterpart  $\{\alpha\}^T$  a row N-vector, so that (for example) the expression  $\{\alpha\}^T [\mathbb{K}] \{\alpha\}$  yields a scalar, consistently with usual conventions of matrix algebra<sup>7</sup>. A related notational convention, used here and thereafter, consists in employing upright subscripts for entries of arrays in approximation space (e.g.  $\{\mathbb{F}\}_I$ ,  $[\mathbb{K}]_{IJ}$ ,  $[K_e]_{pq}$ , ...) and italic subscripts for denoting components of vectors, tensors... in physical space (e.g.  $u_i$ ,  $\sigma_{ij}$ , ...).

The stiffness matrix  $[\mathbb{K}]$  is symmetric and positive, as a direct consequence of the symmetry and positive definiteness of the linear elastic constitutive relation (Section 1.1.3). If, in addition, no rigid-body motion (1.2) belongs to the approximation space defined by (1.37) (i.e. no choice of  $\{\alpha\}$  yields one), which must in particular be the case whenever  $S_u$  has nonzero measure, any linear combination of the  $\varphi^K$  has a nonzero strain energy. In that case, the stiffness matrix is *positive definite* (i.e.  $\{\alpha\} \neq \{0\} \Rightarrow \{\alpha\}^T [\mathbb{K}] \{\alpha\} > 0$ ), which implies its invertibility. Conversely, if representation (1.37) allows some rigid-body motion, the stiffness matrix is not invertible, since  $[\mathbb{K}] \{\alpha\} = \{0\}$  for any  $\{\alpha\}$  such that (1.37) yields a rigid-body motion. Such possibility must be avoided in practical computations.

Assuming  $[\mathbb{K}]$  to be positive definite, the potential energy  $P(\{\alpha\})$  defined by (1.38) is a strictly convex function of  $\{\alpha\}$  and thus has a unique minimizer. The latter is found by setting to zero the partial derivatives of  $P(\{\alpha\})$  (necessary optimality condition for unconstrained minimisation), which leads to the following linear system with N equations and N unknowns:

$$[\mathbb{K}] \{\alpha\} = \{\mathbb{F}\} \quad (1.43)$$

Upon inversion, the optimal generalized displacements are thus given by

$$\{\alpha^{\min}\} = [\mathbb{K}]^{-1} \{\mathbb{F}\}. \quad (1.44)$$

The approximate displacement solution  $\mathbf{u}_N$  for the elastic equilibrium problem is thus given by (1.37) with  $\{\alpha\} = \{\alpha^{\min}\}$ , i.e.

$$\mathbf{u}_N(\mathbf{x}) = \mathbf{u}^{(D)}(\mathbf{x}) + \sum_{K=1}^N \alpha_K^{\min} \varphi^K(\mathbf{x}). \quad (1.45)$$

<sup>7</sup>In particular, the MATLAB syntax follows the same convention, the transposition being denoted by the prime ( $\mathbf{X}'$  being the transpose of the vector or matrix array  $\mathbf{X}$ )



**Approximate stress solution.** Applying the elastic constitutive relation to  $\mathbf{u}_N$ , one obtains an approximate stress solution  $\boldsymbol{\sigma}[\mathbf{u}_N]$ , given by

$$\boldsymbol{\sigma}[\mathbf{u}_N] := \mathcal{A}:\boldsymbol{\varepsilon}[\mathbf{u}^{(D)}](x) + \sum_{K=1}^N \alpha_K^{\min} \mathcal{A}:\boldsymbol{\varepsilon}[\boldsymbol{\varphi}^K](x).$$

The stress field  $\boldsymbol{\sigma}[\mathbf{u}_N]$  is not in general statically admissible, and in particular does not minimize the complementary energy (unless the displacement  $\mathbf{u}_N$  given by (1.45) happens to be the exact solution of the elastic equilibrium problem, i.e. the absolute minimizer of  $\mathcal{P}(\mathbf{u})$ ).

**Interpretation in terms of balance of generalized forces.** It is natural to decompose the generalized force vector (1.41) as  $\{\mathbb{F}\} = \{\mathbb{F}^u\} + \{\mathbb{F}^{\text{ext}}\}$ , where  $\{\mathbb{F}^u\}$  corresponds to the first term of the right-hand side of (1.41), which synthesizes the effect of prescribed displacements, and  $\{\mathbb{F}^{\text{ext}}\}$  collects all contributions from given applied external loads. In addition, it is also natural to define the internal generalized forces  $\{\mathbb{F}^{\text{int}}\}$  by

$$\{\mathbb{F}^{\text{int}}\} = -[\mathbb{K}]\{\alpha\} + \{\mathbb{F}^u\}. \quad (1.46)$$

They are thus given, for any  $\mathbf{v}_N$  of the form (1.37), by

$$\{\mathbb{F}^{\text{int}}\}_I = - \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{v}_N] : \mathcal{A}:\boldsymbol{\varepsilon}[\boldsymbol{\varphi}^I] dV \quad (1 \leq I \leq N). \quad (1.47)$$

Using these definitions, the equilibrium equation (1.43) produced by the Galerkin method simply expresses that the internal and external generalized forces balance each other, i.e.

$$\{\mathbb{F}^{\text{int}}\} + \{\mathbb{F}^{\text{ext}}\} = \{0\}. \quad (1.48)$$

**Galerkin method for the variational formulation.** Representations (1.37) may also be used directly in the variational formulation (1.35). In other words, the Galerkin approach for the variational formulation consist in solving the problem

find  $\mathbf{u}_N \in \mathcal{C}_N(\mathbf{u}^D)$  such that

$$\int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}_N] : \mathcal{A}:\boldsymbol{\varepsilon}[\mathbf{w}] dV = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} dS \quad \forall \mathbf{w} \in \mathcal{C}_N(\mathbf{0}). \quad (1.49)$$

Accordingly, assuming  $\mathbf{u}$  to have the form (1.37) and since virtual fields  $\mathbf{w} \in \mathcal{C}_N(\mathbf{0})$  have the form

$$\mathbf{w}_N(\mathbf{x}) = \sum_{K=1}^N \beta_K \boldsymbol{\varphi}^K(\mathbf{x}), \quad (1.50)$$

the discretized variational formulation (1.49) becomes

$$\{\beta\}^T [\mathbb{K}] \{\alpha\} = \{\beta\}^T [\mathbb{F}] \quad \text{for any } \{\beta\} \in \mathbb{R}^N,$$

where  $[\mathbb{K}]$ ,  $\{\mathbb{F}\}$  are still defined by (1.40) and (1.41). This equation is equivalent to (1.43), and thus yields the approximate displacement solution defined by (1.44). The internal and external generalized forces  $\{\mathbb{F}^{\text{int}}\}_I$  and  $\{\mathbb{F}^{\text{ext}}\}_I$  are in fact the virtual work of the corresponding force densities for the virtual field  $\mathbf{w} = \boldsymbol{\varphi}^I$ .

### 1.4.2 Properties of the Galerkin approximation method

**The solution error is  $\mathcal{A}$ -orthogonal to  $\mathcal{C}_N(0)$ .** Let  $\mathbf{u} = \mathbf{u}_N + \Delta\mathbf{u}$ , where  $\mathbf{u}_N$  is the approximate solution (1.45) given by the Galerkin method ( $\Delta\mathbf{u}$  thus denotes the absolute solution error with respect to the exact solution  $\mathbf{u}$ ). For any virtual field  $\mathbf{w}_N$  of the form (1.50), the weak formulations for the exact and approximate problems produce the identities

$$\begin{aligned}\int_{\Omega} \varepsilon[\mathbf{u}] : \mathcal{A} : \varepsilon[\mathbf{w}_N] dV &= \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w}_N dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w}_N dS \\ \int_{\Omega} \varepsilon[\mathbf{u}_N] : \mathcal{A} : \varepsilon[\mathbf{w}_N] dV &= \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w}_N dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w}_N dS.\end{aligned}$$

Upon taking the difference of these identities, the solution error  $\Delta\mathbf{u}$  is found to be orthogonal (in the sense of the strain energy scalar product) to any virtual field in the approximation space, i.e.:

$$\int_{\Omega} \varepsilon[\Delta\mathbf{u}] : \mathcal{A} : \varepsilon[\mathbf{w}_N] dV = 0. \quad (1.51)$$

**Best approximation property.** Let us write and expand the strain energy of the difference  $\mathbf{u} - \mathbf{v}_N = \Delta\mathbf{u} + (\mathbf{u}_N - \mathbf{v}_N)$  between the exact solution and an arbitrary field of the form (1.37):

$$\begin{aligned}\int_{\Omega} \varepsilon[\mathbf{u} - \mathbf{v}_N] : \mathcal{A} : \varepsilon[\mathbf{u} - \mathbf{v}_N] dV &= \int_{\Omega} \varepsilon[\Delta\mathbf{u}] : \mathcal{A} : \varepsilon[\Delta\mathbf{u}] dV \\ &+ \int_{\Omega} \varepsilon[\mathbf{u}_N - \mathbf{v}_N] : \mathcal{A} : \varepsilon[\mathbf{u}_N - \mathbf{v}_N] dV + 2 \int_{\Omega} \varepsilon[\Delta\mathbf{u}] : \mathcal{A} : \varepsilon[\mathbf{u}_N - \mathbf{v}_N] dV.\end{aligned}$$

The field  $\mathbf{u}_N - \mathbf{v}_N$  being also representable by (1.50), the last integral above vanishes by virtue of the orthogonality property (1.51). One thus obtains the inequality

$$\int_{\Omega} \varepsilon[\Delta\mathbf{u}] : \mathcal{A} : \varepsilon[\Delta\mathbf{u}] dV \leq \int_{\Omega} \varepsilon[\mathbf{u} - \mathbf{v}_N] : \mathcal{A} : \varepsilon[\mathbf{u} - \mathbf{v}_N] dV, \quad (1.52)$$

which shows that  $\mathbf{u}_N$  is the best approximation of  $\mathbf{u}$ , in the energy norm sense, among all fields of the form (1.37).

**Underestimation of the strain energy.** Let the strain energy of the exact solution  $\mathbf{u} = \mathbf{u}_N + \Delta\mathbf{u}$  be expanded as

$$\begin{aligned}\int_{\Omega} \varepsilon[\mathbf{u}] : \mathcal{A} : \varepsilon[\mathbf{u}] dV &= \int_{\Omega} \varepsilon[\mathbf{u}_N] : \mathcal{A} : \varepsilon[\mathbf{u}_N] dV \\ &+ \int_{\Omega} \varepsilon[\Delta\mathbf{u}] : \mathcal{A} : \varepsilon[\Delta\mathbf{u}] dV + 2 \int_{\Omega} \varepsilon[\mathbf{u}_N] : \mathcal{A} : \varepsilon[\Delta\mathbf{u}] dV.\end{aligned}$$

Assume that zero displacements are prescribed (i.e.  $\mathbf{u}^D = \mathbf{0}$  on  $S_u$ ), so that the approximate solution  $\mathbf{u}_N$  also has the form (1.50). The last integral above then vanishes by virtue of the orthogonality property (1.51), leading to

$$\begin{aligned} \int_{\Omega} \varepsilon[\mathbf{u}_N] : \mathcal{A} : \varepsilon[\mathbf{u}_N] dV &= \int_{\Omega} \varepsilon[\mathbf{u}] : \mathcal{A} : \varepsilon[\mathbf{u}] dV - \int_{\Omega} \varepsilon[\Delta \mathbf{u}] : \mathcal{A} : \varepsilon[\Delta \mathbf{u}] dV \\ &< \int_{\Omega} \varepsilon[\mathbf{u}] : \mathcal{A} : \varepsilon[\mathbf{u}] dV. \end{aligned} \quad (1.53)$$

In other words, if  $\mathbf{u} \in \mathcal{C}(\mathbf{0})$ , the strain energy of the approximate solution  $\mathbf{u}_N$  is *smaller* than that of the exact solution  $\mathbf{u}$ . In this energy sense,  $\mathbf{u}_N$  thus approximates  $\mathbf{u}$  from below.

### 1.4.3 The Galerkin method for the complementary energy

This approximation approach consists in minimizing  $\mathcal{P}^*(\boldsymbol{\tau})$  among stress fields  $\boldsymbol{\tau}$  of the form

$$\boldsymbol{\tau}_N(\mathbf{x}) = \boldsymbol{\tau}^{(D)}(\mathbf{x}) + \sum_{K=1}^N \beta_K \boldsymbol{\tau}^K(\mathbf{x}) \quad \text{with } \boldsymbol{\tau}^{(D)} \in \mathcal{S}(\mathbf{T}^D), \boldsymbol{\tau}^K \in \mathcal{S}(\mathbf{0}), \quad (1.54)$$

constructed as the sum of a particular statically admissible stress tensor  $\boldsymbol{\tau}^{(D)}$  and a linear combination of self-equilibrated stresses  $\boldsymbol{\tau}^K$  that define a basis for a N-dimensional approximation space. The complementary energy  $\mathcal{P}^*(\boldsymbol{\tau})$  is then given, for any such stress tensor, by

$$\mathcal{P}^*(\boldsymbol{\tau}) = \frac{1}{2} \{\beta\}^T [\mathbb{S}] \{\beta\} - \{\beta\}^T \{\mathbb{U}^{(D)}\} + \mathcal{P}^*(\boldsymbol{\tau}^{(D)}) = P^*(\{\beta\}) \quad (1.55)$$

in terms of the *compliance matrix*  $[\mathbb{S}]$ , whose entries are given by

$$[\mathbb{S}]_{IJ} = \int_{\Omega} \boldsymbol{\tau}^I : \boldsymbol{\mathcal{S}} : \boldsymbol{\tau}^J dV \quad (1 \leq I, J \leq N), \quad (1.56)$$

and of the vector  $\{\mathbb{U}^{(D)}\}$  of *generalized displacements* associated to the kinematic and static data, whose entries are

$$\{\mathbb{U}^{(D)}\}_I = \int_{S_u} [\boldsymbol{\tau}^I \cdot \mathbf{n}] \cdot \mathbf{u}^D dS - \int_{\Omega} \boldsymbol{\tau}^I : \boldsymbol{\mathcal{S}} : \boldsymbol{\tau}^{(D)} dV.$$

The compliance matrix is  $N \times N$  square and positive definite, while  $\{\mathbb{U}^{(D)}\}$  and  $\{\beta\}$  are N-vectors.

The Galerkin method then consists in solving the approximate problem

$$\boldsymbol{\sigma}_N = \arg \min_{\boldsymbol{\tau} \in \mathcal{S}_N(\mathbf{T}^D, \mathbf{f})} \mathcal{P}^*(\boldsymbol{\tau}) \quad (1.57)$$

with

$$\begin{aligned} \mathcal{S}_N(\mathbf{0}, \mathbf{0}) &= \text{span}\{\boldsymbol{\tau}^K, K = 1, \dots, N\} \subset \mathcal{S}(\mathbf{0}, \mathbf{0}), \\ \mathcal{S}_N(\mathbf{T}^D, \mathbf{f}) &= \boldsymbol{\tau}^{(D)} + \mathcal{S}_N(\mathbf{0}, \mathbf{0}) \end{aligned}$$

The minimization of  $P^*(\{\beta\})$  with respect to  $\{\beta\}$  then yields

$$[\mathbb{S}]\{\beta\} = \{\mathbb{U}^{(D)}\} \Rightarrow \{\beta^{\min}\} = [\mathbb{S}]^{-1}\{\mathbb{U}^{(D)}\}. \quad (1.58)$$

The approximate stress solution  $\sigma_N$  is therefore given by (1.54) with  $\{\beta\} = \{\beta^{\min}\}$ . In general, applying the inverse form (1.17) of the constitutive relation to  $\sigma_N$  does not yield a compatible strain tensor, and therefore cannot be used for finding an approximate displacement solution.

#### 1.4.4 Discussion: from the Galerkin method to the finite element method

**Comparison of displacement- and stress-based approaches.** The Galerkin method applied to the potential energy (displacement-based approach) is in principle reasonably easy to implement since the construction of bases ( $\varphi^K$ ) of displacement fields fulfilling zero boundary data is not overly difficult. This results from the fact that, in addition to the kinematic constraints, the  $\varphi^K$  are only required to verify regularity conditions (continuity, finiteness of strain energy  $\mathcal{W}(\varphi^K)$ ).

The Galerkin method applied to the complementary energy (stress-based approach) is significantly harder to implement because bases ( $\tau^K$ ) must be constructed using self-equilibrated (and hence, in particular, divergence-free) stress fields. This makes constructing such approximation spaces more difficult (see Kempeneers, Beckers, and Debongnie, 2004, among others).

**The Galerkin method as foundation of the finite element method.** It is relatively easy to define bases ( $\varphi^K$ ) of displacement fields defined in the whole domain  $\Omega$ , for instance using polynomial or trigonometric functions. For example, systematic methods for constructing polynomial bases of arbitrary degree from families of orthogonal polynomials are available. This approach is at the root of *spectral methods* (Bernardi and Maday, 1997). The Galerkin method employing bases of (displacement or stress) fields defined in all of  $\Omega$  (and hence of a "global" character) has two notable characteristics:

- (a) The entries of the stiffness (or compliance) matrix are given by integrals over the whole domain  $\Omega$ ;
- (b) The stiffness (or compliance) matrix is *fully-populated*: all coefficients  $[\mathbb{K}]_{IJ}$  given by (1.40) (or  $[\mathbb{S}]_{IJ}$  given by (1.56)) *a priori* have nonzero values.

Issue (a) makes applications to complex geometrical configurations cumbersome, as the required numerical quadrature is difficult to implement and computationally expensive. Issue (b) also adversely affects the computational efficiency, since solving a linear system of equations such as (1.43) is faster when the featured matrix (which for this example is  $[\mathbb{K}]$ ) is *sparse* (a matrix is said to be sparse when a large proportion of its entries are zeros). Moreover, it is also difficult to ensure that such globally-defined bases for the approximation spaces satisfy, as they must, the required kinematical-admissibility boundary conditions when complex geometrical configurations are considered.

The finite element method, which will be described in detail in Chapters 2 and 3, is a particular version of the Galerkin method whose design aims at avoiding the above-described shortcomings. The main idea underlying the finite element method is the definition of approximation spaces in terms of basis functions  $\varphi^K$  whose geometrical support is "small" (and hence of a "local" character), i.e. which vanish outside a small region  $\Omega_I \subset \Omega$  of simple shape. Consequently:

- (a) Numerical quadratures are restricted to small domains, allowing for a simpler implementation and a faster numerical evaluation. For instance, the evaluation of  $[\mathbb{K}]_{IJ}$  defined by (1.40) now only requires a quadrature over  $\Omega_I \cap \Omega_J$ .
- (b) Moreover, all entries  $[\mathbb{K}]_{IJ}$  or  $[\mathbb{S}]_{IJ}$  such that  $\Omega_I \cap \Omega_J = \emptyset$  vanish, resulting in very sparse stiffness or compliance matrices.
- (c) The local character of the basis functions greatly facilitates the enforcement of kinematical constraints.

## 1.5 Application: pressurized spherical shell

This standard example, whose exact solution is simple and well-known, will serve as a convenient illustration of the concepts developed in this chapter. A spherical shell, bounded by internal and external concentric spherical surfaces of respective radii  $R_1$  and  $R_2$ , is subjected to a uniform pressure  $p$  on its internal face, while a uniform radial displacement  $d$  is prescribed on its external face. Spherical coordinates  $r, \theta, \varphi$  emanating from the spherical shell center are used. The constitutive material of the shell is linearly elastic, isotropic and isothermal, no pre-stress is applied, and the standard small-deformation framework is used.

The spherical symmetry of the loading and geometry implies that of the response. The displacement and strain are thus *a priori* sought in the form

$$\mathbf{u} = u(r)\mathbf{e}_r, \quad \boldsymbol{\varepsilon} = \frac{du}{dr}\mathbf{e}_r \otimes \mathbf{e}_r + \frac{u}{r}\mathbf{e}_\theta \otimes \mathbf{e}_\theta + \frac{u}{r}\mathbf{e}_\varphi \otimes \mathbf{e}_\varphi \quad (1.59)$$

where the radial displacement  $u(r)$  is the primary unknown. The stress tensor predicted by the elastic constitutive relation (1.1c) is then given by

$$\boldsymbol{\sigma} = \sigma_{rr}\mathbf{e}_r \otimes \mathbf{e}_r + \sigma_{\theta\theta}\mathbf{e}_\theta \otimes \mathbf{e}_\theta + \sigma_{\varphi\varphi}\mathbf{e}_\varphi \otimes \mathbf{e}_\varphi \quad (1.60)$$

with

$$\sigma_{rr} = \lambda \left( \frac{du}{dr} + 2\frac{u}{r} \right) + 2\mu \frac{du}{dr} \quad \sigma_{\theta\theta} = \sigma_{\varphi\varphi} = \lambda \left( \frac{du}{dr} + 2\frac{u}{r} \right) + 2\mu \frac{u}{r} \quad (1.61)$$

Finally, the boundary conditions are:

$$\sigma_{rr}(R_1) = -p \quad u(R_2) = d \quad (1.62)$$

### 1.5.1 Strong formulation

Combining equations (1.1a–c) written in spherical coordinates and taking into account the spherical symmetry, the strong formulation of the elastic equilibrium of the shell reduces to the following differential equation for  $u$

$$\frac{d\sigma_{rr}}{dr} + \frac{2\sigma_{rr} - \sigma_{\theta\theta} - \sigma_{\varphi\varphi}}{r} = (\lambda + 2\mu) \left( \frac{d^2u}{dr^2} + 2\frac{1}{r} \frac{du}{dr} - 2\frac{u}{r^2} \right) = 0 \quad (R_1 < r < R_2) \quad (1.63)$$

supplemented with the boundary conditions (1.62). The exact solution is then easily found to be given by

$$u(r) = ar + \frac{b}{r^2}, \quad \sigma_{rr} = A - 2\frac{B}{r^3}, \quad \sigma_{\theta\theta} = \sigma_{\varphi\varphi} = A + \frac{B}{r^3} \quad (1.64)$$

with

$$a = \frac{A}{E}(1 - 2\nu) \quad b = \frac{B}{E}(1 + \nu)$$

and where the constants  $A, B$  are determined by the boundary conditions (1.62):

$$A = \frac{-(1 + \nu)pR_1^3 + 2dER_2^2}{(1 + \nu)R_1^3 + 2(1 - 2\nu)R_2^3} \quad B = R_1^3R_2^2 \frac{dE + (1 - 2\nu)pR_2}{(1 + \nu)R_1^3 + 2(1 - 2\nu)R_2^3}$$

This exact solution will now be used for comparisons with approximate solutions.

### 1.5.2 Weak formulation

Equation (1.63) is multiplied by  $w(r) \in \mathcal{C}$  and the result integrated over  $\Omega$  (with the resulting factor  $4\pi$  omitted for simplicity):

$$\int_{R_1}^{R_2} \left( \frac{d\sigma_{rr}}{dr} + \frac{2\sigma_{rr} - \sigma_{\theta\theta} - \sigma_{\varphi\varphi}}{r} \right) wr^2 dr = 0 \quad \forall w \in \mathcal{C} \quad (1.65)$$

An integration by parts gives

$$\begin{aligned} \int_{R_1}^{R_2} \left( \frac{d\sigma_{rr}}{dr} + \frac{2}{r}\sigma_{rr} \right) wr^2 dr &= \int_{R_1}^{R_2} \left( \frac{d(\sigma_{rr}r^2)}{dr} \right) w dr \\ &= [\sigma_{rr}wr^2]_{R_1}^{R_2} - \int_{R_1}^{R_2} \left( \sigma_{rr} \frac{dw}{dr} \right) r^2 dr. \end{aligned}$$

Substituting this identity into (1.65) and using boundary condition (1.62) at  $r = R_1$ , one obtains the weak formulation

$$\int_{R_1}^{R_2} \left( \sigma_{rr} \frac{dw}{dr} + (\sigma_{\theta\theta} + \sigma_{\varphi\varphi}) \frac{w}{r} \right) r^2 dr = pR_1^2 w(R_1) + \sigma_{rr}(R_2)R_2^2 w(R_2) \quad \forall w \in \mathcal{C} \quad (1.66)$$

Equation (1.66) corresponds to the form (1.19) of the weak formulation, which includes the reaction (here  $\sigma_{rr}(R_2)$ ) arising from enforcing the displacement at  $r = R_2$ . To obtain the form (1.22) of the weak formulation that does not involve the reaction, the virtual fields are restricted to  $\mathcal{C}(0) = \{w \mid w(R_2) = 0\}$  so as to cancel the second term in the right-hand side of (1.66). One obtains

$$\int_{R_1}^{R_2} \left( \sigma_{rr} \frac{dw}{dr} + (\sigma_{\theta\theta} + \sigma_{\varphi\varphi}) \frac{w}{r} \right) r^2 dr = pR_1^2 w(R_1) \quad \forall w \in \mathcal{C}(0). \quad (1.67)$$

In a way, knowing the solution  $u$  at  $r = R_2$  makes it unnecessary to write an equation at  $r = R_2$ ; this is achieved by setting the virtual field to zero at that location.

### 1.5.3 Variational formulation

We know that the potential energy functional is minimized in the space  $\mathcal{C}(\mathbf{u}^D)$  of kinematically admissible displacements (here, continuous and piecewise-differentiable functions  $v(r)$  such that  $v(R_2) = d$ ) by the equilibrium solution (1.64):

$$u = \arg \min_{v \in \mathcal{C}(\mathbf{u}^D)} (\mathcal{W}[v] - \mathcal{F}[v]) \quad (1.68)$$

with

$$\begin{aligned} \mathcal{W}[v] &= \frac{1}{2} \int_{\Omega} (\sigma_{rr}[v] \varepsilon_{rr}[v] + \sigma_{\theta\theta}[v] \varepsilon_{\theta\theta}[v] + \sigma_{\varphi\varphi}[v] \varepsilon_{\varphi\varphi}[v]) \, dV \\ &= \frac{4\pi}{2} \int_{R_1}^{R_2} \left\{ (\lambda + 2\mu) \left( \frac{dv}{dr} \right)^2 + 4\lambda \frac{v}{r} \frac{dv}{dr} + 4(\lambda + \mu) \left( \frac{v}{r} \right)^2 \right\} r^2 \, dr \\ \mathcal{F}[v] &= 4\pi p R_1^2 v(R_1) \end{aligned}$$

The minimization of  $\mathcal{P}[v] = \mathcal{W}[v] - \mathcal{F}[v]$  in the infinite-dimensional space  $\mathcal{C}(\mathbf{u}^D)$  yields the exact solution (1.64). Since the functional  $\mathcal{P}[v]$  is convex, its minimum is equivalent to finding its stationary point. For any  $v \in \mathcal{C}(\mathbf{u}^D)$ , one can put  $v(r) = u(r) + \eta w(r)$ , with  $w \in \mathcal{C}(0)$ . The stationarity condition for  $\mathcal{P}$  is then equivalent to the equation

$$\lim_{\eta \rightarrow 0} \eta^{-1} (\mathcal{P}[u + \eta w] - \mathcal{P}[u]) = 0 \quad \forall w \in \mathcal{C}(0)$$

which leads to

$$\begin{aligned} \int_{R_1}^{R_2} \left\{ (\lambda + 2\mu) \frac{du}{dr} \frac{dw}{dr} + 2\lambda \left( \frac{u}{r} \frac{dw}{dr} + \frac{w}{r} \frac{du}{dr} \right) + 4(\lambda + \mu) \frac{u}{r} \frac{w}{r} \right\} r^2 \, dr \\ - pw(R_1) R_1^2 = 0 \quad \forall w \in \mathcal{C}(0) \end{aligned} \quad (1.69)$$

One notes, invoking the expressions of the strains and stresses in terms of  $u$ , that the stationarity condition (1.69) can also be set in the form

$$\begin{aligned} \int_{R_1}^{R_2} \left\{ \sigma_{rr}[u] \varepsilon_{rr}[w] + \sigma_{\theta\theta}[u] \varepsilon_{\theta\theta}[w] + \sigma_{\varphi\varphi}[u] \varepsilon_{\varphi\varphi}[w] \right\} r^2 \, dr - pw(R_1) R_1^2 = 0 \\ \forall w \in \mathcal{C}(0) \end{aligned} \quad (1.70)$$

which corresponds in fact to (1.35) with the spherically-symmetric form (1.60) taken into account.

### 1.5.4 Approximate solution: the finite difference method

The finite difference method may be applied for solving the problem in strong form, i.e. the differential equation

$$\frac{d^2 u}{dr^2} + 2 \frac{1}{r} \frac{du}{dr} - 2 \frac{u}{r^2} = 0, \quad R_1 < r < R_2 \quad (1.71)$$

with boundary conditions (1.62). To this end, the domain of interest, namely the segment  $R_1 \leq r \leq R_2$ , is sampled using  $N_N$  nodes whose radial coordinate is  $r^{(i)} =$

$R_1 + (i - 1)h$  ( $1 \leq i \leq N_N$  (i.e. a uniform node spacing  $h = (R_2 - R_1)/(N_N - 1)$  is assumed). This approximate solution approach aims at finding the nodal values  $u^{(i)} = u(r^{(i)})$  by enforcing (1.71) at each node, with derivatives replaced therein by finite-difference approximations. For instance, the centered finite difference approach leads to

$$\begin{aligned}\frac{du}{dr}(r^{(i)}) &= \frac{u^{(i+1)} - u^{(i-1)}}{2h} + O(h^2) \\ \frac{d^2u}{dr^2}(r^{(i)}) &= \frac{u^{(i+1)} - 2u^{(i)} + u^{(i-1)})}{h^2} + O(h^2)\end{aligned}\quad (1.72)$$

and (1.71) with the above approximations is enforced at all interior nodes ( $2 \leq i \leq N_N - 1$ ). Applying boundary conditions (1.62) at the endpoints gives the additional equations

$$(\lambda + 2\mu)\frac{u^{(2)} - u^{(1)}}{h} + 2\lambda\frac{u^{(1)}}{R_1} = -p, \quad u^{(N_N)} = d \quad (1.73)$$

The whole set of equations thus obtained allows to find the nodal values  $u^{(i)}$ . This approach, very simple to formulate and implement on this particular example, becomes complicated for vector-valued unknowns and geometrically complex domains, in particular due to the difficulty in enforcing the relevant boundary conditions and in creating a sufficiently regular grid of nodes. Such limitations make finite difference methods ill-suited to solid and structural mechanics, for which other methods are preferred, in particular the finite element methods.

### 1.5.5 Approximate solution: the Galerkin method using polynomials

A possible choice of basis functions for applying the Galerkin method is now presented. Let the particular extension  $u^{(D)}$  of the displacement boundary data be defined as

$$u^{(D)}(r) = d \frac{r - R_1}{R_2 - R_1}$$

and define the approximation space through functions  $v_N(r) \in \mathcal{C}(u^{(D)})$  of the form

$$v_N(r) = u^{(D)}(r) + \sum_{I=1}^N \alpha_I (r - R_2)^I$$

where  $N \geq 0$  defines the dimension (i.e. richness) of the space. For  $N = 0$  one has  $v_N(r) = u^{(D)}(r)$ .

To define this kind of approximation space, made of functions defined globally (i.e. in the whole domain  $\Omega$ ) is easy for the present example but becomes very difficult for complex geometries because of the necessity to satisfy kinematical boundary conditions. Like the finite difference method, the Galerkin method with global basis functions is thus ill-suited to the geometrical complexity usually encountered in the analysis of mechanical structures.

The implementation of the Galerkin method with the above-defined approximation on the present example is easy, and details are left to the reader. Table 1.1 shows the convergence of the total potential energy with increasing richness (i.e. polynomial degree)  $N$ . The first line gives the relative error between the approximate and exact values of the potential energy, while the second and third lines show absolute errors on



the displacement and its derivative, evaluated in terms of the norm defined by

$$\|f\|_0 := \left( \int_{R_1}^{R_2} f^2 r^2 dr \right)^{1/2}$$

Polynomial degree $N$	0	1	2	3
Relative error on $\mathcal{P}$	1,29087	0,442361	0,0596442	0,00501164
$\ u - u_N\ _0$	0,157699	0,0341987	0,00790153	0,00201395
$\ (du/dr) - (du_N/dr)\ _0$	0,423933	0,244242	0,0810537	0,0226432

**Table 1.1:** Convergence towards the exact solution (with  $E = 1$ ,  $\nu = 0,3$ ,  $R_1 = 1$ ,  $R_2 = 2$ ,  $p = 1$ ,  $d = -0,1$ )

### 1.5.6 Approximate solution: Galerkin method with piecewise-linear fields

As an introduction to more-complex problems considered in the remainder of this book, we now examine and discuss a MATLAB procedure for solving numerically the variational formulation for spherical shell problem using a finite element-type approximation space. The MATLAB file `sphere_B2_1S.solid.m` contains the code that is described next.

A partition of the segment  $R_1 \leq r \leq R_2$  into  $N_E$  "pieces" (elements) of equal length is introduced. The number  $N_E \rightarrow NE$  of elements is chosen by the user, as are the radii  $R_1 \rightarrow R1$  and  $R_2 \rightarrow R2$ , the elastic coefficients  $E$  et  $\nu \rightarrow nu$ , and given values for the internal pressure  $p$  at  $r = R_1$  and the prescribed displacement  $d$  at  $r = R_2$ . The symbol  $\rightarrow$  is used here and thereafter to denote the MATLAB "translation" of the corresponding quantity.

```

R1=1; R2=2;           % inner and outer radii
NE=6;                 % number of elements
E=1.;                 % Young modulus
nu=.3;                % Poisson coefficient
p=1;                  % value of inner pressure
d=-.1;                % value of imposed disp at r=R2

```

The  $N_N = N_E + 1$  element endpoints are called "nodes". From the input data, the nodal coordinates (stored in the `coor` array) and the Lamé constants are defined:

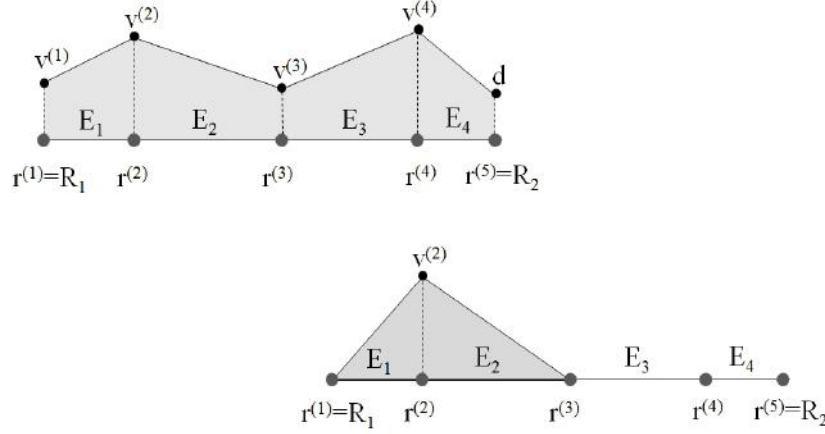
```

h=(R2-R1)/NE;         % size of one element
NN=NE+1;              % number of nodes
coor=[R1:h:R2]';      % nodal coords: uniform discret.
mu=E/(2*(1+nu));      % definition of Lamé constants
lambda=2*nu*mu/(1-2*nu);

```

One now defines the finite-dimensional space  $\mathcal{C}_h(d)$  of functions  $v(r)$  that are (i) continuous, (ii) compatible with the kinematical data  $d$  and (iii) piecewise-linear (i.e. have a linear variation on each element), with  $h := (R_2 - R_1)/N_E$  denoting the element length. All functions belonging to  $\mathcal{C}_h(d)$  are then fully defined by their values  $v^{(i)} := v(r^{(i)})$  at the nodes, called *nodal values* (Fig. 1.2).

The vector  $U$  is created for storing the displacement nodal values  $v^{(i)}$ . The elements and nodes are assumed to be numbered sequentially along the increasing radial coordinate  $r$ . The first  $neq = NN - 1$  nodal values are the main unknowns, the kinematic



**Figure 1.2:** Sphere problem. Piecewise linear displacement field  $v \in \mathcal{C}_h(d)$

admissibility reducing to the requirement  $v^{(NN)} = v(R_2) = d$  (the last entry of  $U$  is thus set to that value).

```
neq=NN-1;           % number of unknown nodal values
U=zeros(NN,1);      % init. displacements
U(NN)=d;             % puts imposed displ. on last node
```

The variational formulation stipulates that the approximate displacement solution is the function  $u_h \in \mathcal{C}_h(d)$  such that

$$u_h = \arg \min_{v_h \in \mathcal{C}_h(d)} (\mathcal{W}[v_h] - \mathcal{F}[v_h])$$

with  $\mathcal{W}$  and  $\mathcal{F}$  as defined in (1.68). Since the potential energy is additive with respect to the elements (by virtue of being an integral) and the basis functions are defined element-wise, it is natural to adopt an element-by-element treatment.

**Analysis of one element.** Consider, accordingly, a generic element  $E_e$  (with number  $e$ ) defined by the  $e$ -th and  $(e+1)$ -th nodes. Let

$$\{V_e\} = \{v^{(e)} \quad v^{(e+1)}\}^T$$

denote the list of nodal displacements associated with the element. An arclength coordinate  $s := r - r^{(e)}$ ,  $0 \leq s \leq h$  ( $h$ : element length) is also introduced. The approximate displacement being, by assumption, linear over the element, its value at any point  $s$  depends only on  $v^{(e)}, v^{(e+1)}$ :

$$v_h(s) = v^{(e)} \left(1 - \frac{s}{h}\right) + v^{(e+1)} \frac{s}{h} \quad (1.74)$$

$$\frac{dv_h}{dr} = \frac{dv_h}{ds} = \frac{1}{h} (-v^{(e)} + v^{(e+1)}) \quad (1.75)$$

which can be recast in the form

$$v_h(s) = \{N\}^T \{V_e\}, \quad \frac{dv_h}{dr} = \frac{dv_h}{ds} = \{B\}^T \{V_e\}$$

having set

$$\{N\} = \left\{1 - \frac{s}{h}, \quad \frac{s}{h}\right\}^T, \quad \{B\} = \frac{1}{h} \{-1 \quad 1\}^T$$

The radial coordinate  $r$  is itself a function of  $s$ :  $r = r^{(e)} + s$ . We are now in a position to evaluate the contribution of the  $e$ -th element to the strain energy  $\mathcal{W}$ :

$$\begin{aligned} \frac{1}{2} \int_0^h \left\{ (\lambda + 2\mu) \left( \frac{dv_h}{dr} \right)^2 + 4\lambda \frac{v_h}{r} \frac{dv_h}{dr} + 4(\lambda + \mu) \left( \frac{v_h}{r} \right)^2 \right\} r^2 ds \\ = \frac{1}{2} \{V_e\}^T [K_e] \{V_e\} \end{aligned}$$

where the matrix  $[K_e]$ , known as the element stiffness matrix, is given by

$$\begin{aligned} [K_e] &= \int_0^h \left\{ (\lambda + 2\mu) \{B\} \{B\}^T r^2 + 2\lambda (\{N\} \{B\}^T + \{B\} \{N\}^T) r \right. \\ &\quad \left. + 4(\lambda + \mu) (\{N\} \{N\}^T) \right\} ds \\ &= \frac{\lambda + 2\mu}{3h} (r^{(e+1)2} + r^{(e)} r^{(e+1)} + r^{(e)2}) \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \\ &\quad + \frac{\lambda}{3} \begin{bmatrix} -2(h + 3r^{(e)}) & -h \\ -h & 2(2h + 3r^{(e)}) \end{bmatrix} + \frac{2(\lambda + \mu)h}{3} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \quad (1.76) \end{aligned}$$

These developments are easy to translate into code, for example using the MATLAB language. The function `B2_1S_solid_Ke(X,lambda,mu)` takes as input the element nodal coordinates (vector  $X$ ) and the elasticity coefficients (scalars  $\lambda$  and  $\mu$ ) and evaluates (1.76):

```
function Ke=B2_1S_solid_Ke(X,lambda,mu);

r1=X(1); r2=X(2); % radial coordinates of nodes
h=r2-r1; % length of element
Ke=(lambda+2*mu)*... % element stiffness matrix
    (r2^2+r1*r2+r1^2)/(3*h)*[1 -1;-1 1]+...
    2*lambda/6*...
    [-2*(h+3*r1) -h; -h 2*(2*h+3*r1)]+...
    4*(lambda+mu)*h/6*[2 1;1 2];
```

**Analysis of the whole structure.** The strain energy being additive, one has

$$\mathcal{W}[v_h] = \frac{1}{2} \sum_{e=1}^{N_E} \{V_e\}^T [K_e] \{V_e\} \quad (1.77)$$

Introducing the vector  $\{\mathbb{V}\}$  of nodal displacements of any  $v \in \mathcal{C}_h(d)$  at the first  $N_N - 1$  nodes,  $\mathcal{W}[v_h]$  given by (1.77) and  $\mathcal{F}[v_h]$  may be decomposed as

$$\mathcal{W}[v_h] = \frac{1}{2} \{\mathbb{V}\}^T [\mathbb{K}] \{\mathbb{V}\} + \{\mathbb{V}\}^T \{\mathbb{F}^d\} + C_1 \quad (1.78)$$

$$\mathcal{F}[v_h] = \{\mathbb{V}\}^T \{\mathbb{F}^p\} \quad (1.79)$$

where  $[\mathbb{K}]$ ,  $\{\mathbb{F}^d\}$ ,  $\{\mathbb{F}^p\}$ ,  $C_1$  do not depend on  $\{\mathbb{V}\}$  (the constant  $C_1$  depends on the kinematical data  $d$ , the material parameters and the discretization). Up to an additive constant, the potential energy then takes the form

$$\mathcal{W}[v_h] - \mathcal{F}[v_h] = \frac{1}{2} \{\mathbb{V}\}^T [\mathbb{K}] \{\mathbb{V}\} - \{\mathbb{V}\}^T \{\mathbb{F}\} \quad \text{with} \quad \{\mathbb{F}\} = \{\mathbb{F}^p\} - \{\mathbb{F}^d\}$$

Computing the matrix  $[\mathbb{K}]$  and the vector  $\{\mathbb{F}\}$  is known as an *assemblage* operation, effected by means of a procedure applied sequentially to each element. A distinction is made between the last element ( $e = N_E$ ), which supports the prescribed displacement, and the other elements. For element  $e = N_E$ , one has (taking into account the symmetry of  $[K_{N_E}]$ ):

$$\{V_{N_E}\}^T [K_{N_E}] \{V_{N_E}\} = [K_{N_E}]_{11} (v^{(N_N-1)})^2 + 2 [K_{N_E}]_{12} v^{(N_N-1)} d + [K_{N_E}]_{22} d^2$$

The strain energy of that element thus involves a quadratic term (which contributes to  $[\mathbb{K}]$ ), a linear term (which contributes to  $\{\mathbb{F}^d\}$ ) and a constant term ( $C_1 = [K_{N_E}]_{22} d^2$ , which is ignored).

The other elements yield quadratic terms only, and hence contribute solely to the matrix  $[\mathbb{K}]$ . The list

$$\{I_e\} = \{e, e+1\}$$

is used to express the contribution to  $[\mathbb{K}]$  of element  $e$ :

$$[\mathbb{K}]_{IJ} = [\mathbb{K}]_{IJ} + [K_e]_{pq} \quad \forall p, q \in \{1, 2\} \quad \text{and} \quad I = \{I_e\}_p, \quad J = \{I_e\}_q$$

```

K=zeros(neq,neq);           % allocates stiffness matrix
F=zeros(neq,1);             % allocates rhs side
for e=1:NE                   % assemblage of stiffness matrix
    X=[coor(e) coor(e+1)];    % creates segment
    Ke=B2_1S_solid_Ke(X,lambda,mu); % element stiffness matrix
    Ie=[e e+1];              % sets nodal degrees of freedom
    if e<NE                   % if not last element
        K(Ie,Ie)=K(Ie,Ie)+Ke; % the whole Ke goes into K
    else
        K(neq,neq)=K(neq,neq)+Ke(1,1); % else only the first coefficient
        F(neq)=-Ke(1,2)*U(NN); % and Ke also contributes to rhs
    end
end

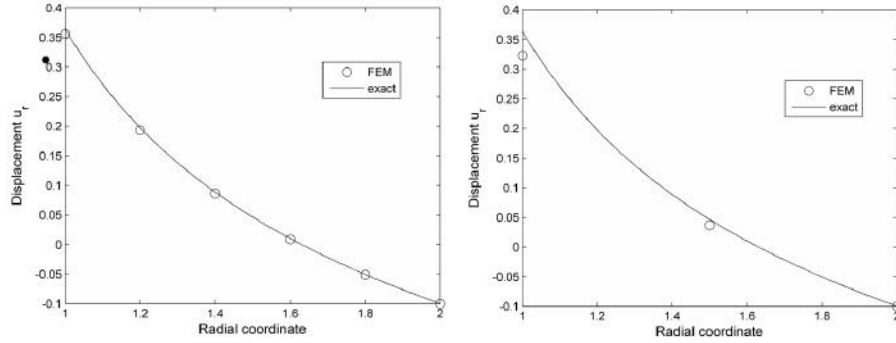
```

Setting up the vector  $\{\mathbb{F}^p\}$  is on the other hand a trivial operation here, since  $\mathcal{F}[v_h] = R_{1p}^2 \{\mathbb{V}\}_1$ :

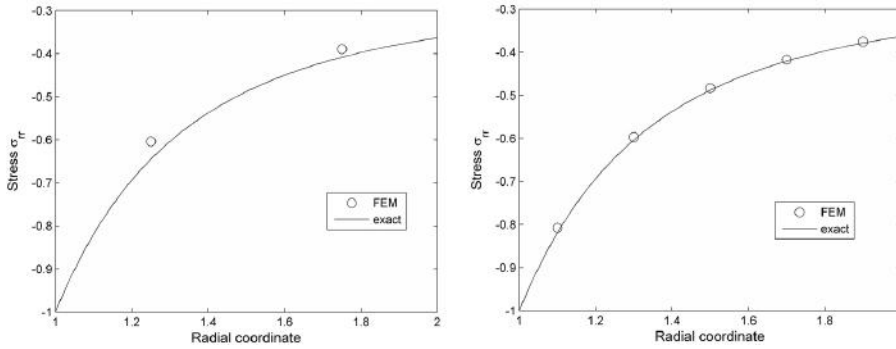
```

F(1)=F(1)+coor(1)^2*p;      % contribution from inner pressure

```



**Figure 1.3:** Nodal displacements using 2 elements (left) or 5 elements (right)



**Figure 1.4:** Radial stress  $\sigma_{rr}$  at element midpoints, using 2 elements (left) or 5 elements (right)

The *best* field in the chosen space  $C_h(d)$  is therefore defined by the vector  $\{\mathbf{U}\}$  solving the minimization problem

$$\{\mathbf{U}\} = \arg \min_{\{\mathbf{V}\} \in \mathbb{R}^{N_N-1}} \left( \frac{1}{2} \{\mathbf{V}\}^T [\mathbf{K}] \{\mathbf{V}\} - \{\mathbf{V}\}^T \{\mathbf{F}\} \right) \quad (1.80)$$

which is equivalent to solving the linear system

$$[\mathbf{K}] \{\mathbf{U}\} = \{\mathbf{F}\} \quad (1.81)$$

The displacement vector can then be filled with the first  $N_N - 1$  nodal values:

```
U(1:NN-1)=K\F;           % solution of linear system
```

The solution stage being finished, the code now enters a post-processing phase, where the radial stress is evaluated at each element center and compared with the available exact solution. Reading and explaining this part of the code is left to the reader.

### 1.5.7 Approximate solution of the weak formulation

The MATLAB function implemented for approximately solving the variational formulation is also suitable for solving the weak formulation. Following equation (1.67), the approximate solution  $u_h$  is sought in  $\mathcal{C}_h(d)$  (chosen as the same space of piecewise-linear continuous functions of Section 1.5.6) so as to verify

$$\int_{R_1}^{R_2} \left\{ (\lambda + 2\mu) \frac{du_h}{dr} \frac{dw_h}{dr} + 2\lambda \left( \frac{u_h}{r} \frac{dw_h}{dr} + \frac{w_h}{r} \frac{du_h}{dr} \right) + 4(\lambda + \mu) \frac{u_h}{r} \frac{w_h}{r} \right\} r^2 dr - pw_h(R_1)R_1^2 = 0 \quad \forall w \in \mathcal{C}_h(0) \quad (1.82)$$

The space  $\mathcal{C}_h(0)$  is the space of piecewise-linear continuous functions that vanish at the endpoint  $r = R_2$ .

Denoting by  $\{\mathbb{W}\}$  the  $(N_N - 1)$ -vector of nodal values associated with  $w \in \mathcal{C}_h(0)$ , a development following closely that of Section 1.5.6 leads to the discretized version of (1.82):

$$\{\mathbb{W}\}^T [\mathbb{K}] \{\mathbb{U}\} - \{\mathbb{W}\}^T \{\mathbb{F}\} = 0 \quad \forall \{\mathbb{W}\} \quad (1.83)$$

where  $[\mathbb{K}]$  and  $\{\mathbb{F}\}$  have the same definition as in equation (1.81). Equation (1.83) is equivalent to the linear system

$$[\mathbb{K}] \{\mathbb{U}\} = \{\mathbb{F}\}, \quad (1.84)$$

since setting  $\{\mathbb{W}\}_1 = 1$ ,  $\{\mathbb{W}\}_{I \neq 1} = 0$  in (1.83) yields the  $I$ -th row of (1.84).

### 1.5.8 Exercises

**Exercise 1.1** (potential energy). *Modify the code in file sphere\_B2\_1S\_solid.m to allow that procedure to compute the total potential energy of the approximate solution and study the convergence of that quantity towards its exact value.*

**Exercise 1.2** (hollow cylinder subject to internal and external pressure). *Write the strong, variational and weak formulations for a hollow cylinder (with internal and external radii  $R_1$ ,  $R_2$ ) subjected to an internal pressure  $p_1$  and an external pressure  $p_2$ , assuming plane-strain deformations. Write a MATLAB procedure for solving the weak formulation using piecewise-linear displacements.*

## 1.6 Scalar problems for conductive media

In addition to linear elasticity, a number of other commonly-used physical models describing equilibrium or steady-state situations involve linear boundary value problems whose structure is very similar to, and somewhat simpler than, that of the elastic equilibrium problem (1.1a-e). Such models usually involve scalar field variables, and media whose constitutive behavior is defined in terms of a conductivity. Hence weak formulations, and later finite element solution methodologies, can be formulated for such models by a straightforward transposition of the arguments used for linear elasticity.

**Field equations.** Consider a conducting material occupying the domain  $\Omega$ , characterized by its isotropic (but possibly space-dependent, i.e. heterogeneous) conductivity coefficient  $k$ <sup>8</sup>. Such models involve a primary scalar field variable  $\varphi$ , often termed *potential* and whose meaning depends on the physical context (see examples thereafter) and a flux vector  $\mathbf{q}$ . Considering only time-independent situations, the relevant governing field equations for a medium occupying the domain  $\Omega \subset \mathbb{R}^D$  ( $D = 2, 3$ ) then are the equilibrium equation

$$\operatorname{div} \mathbf{q}(\mathbf{x}) = g(\mathbf{x}) \quad (\mathbf{x} \in \Omega), \quad (1.85)$$

where  $g$  is an internal source density assumed to be known, and (for linear behavior) the constitutive equation

$$\mathbf{q}(\mathbf{x}) = -k \nabla \varphi(\mathbf{x}) \quad (\mathbf{x} \in \Omega) \quad (1.86)$$

Combining (1.85) and (1.86) of course yields

$$\operatorname{div} (k \nabla \varphi)(\mathbf{x}) + g(\mathbf{x}) = 0 \quad (\mathbf{x} \in \Omega) \quad (1.87)$$

The above equations are relevant to many steady-state physical models, such as:

- Heat conduction:  $\varphi$  is the (time-independent) temperature,  $\mathbf{q}$  the heat flux according to Fourier's law,  $k$  the thermal conductivity of the material and  $g$  the internal heat source;
- Electrostatics:  $\varphi$  is the electrostatic potential,  $\mathbf{q}$  the current density,  $k$  the electrostatic conductivity and  $g$  is related to the electric charge density in  $\Omega$ ;
- Potential flow:  $\varphi$  is the velocity potential,  $\mathbf{q}$  the velocity field,  $g$  may model sources (inflow) or sinks (outflow), and normally  $k = 1$  as there is no relevant physical conductivity;
- Groundwater flow (Darcy model):  $\varphi$  is the static head,  $\mathbf{q}$  the fluid velocity,  $\mathbf{k}$  is the hydraulic conductivity tensor,  $g$  may model sources (inflow) or sinks (outflow);
- Torsion of a cylindrical rod:  $\varphi$  is the warping,  $g = 2\mu\beta$  (with  $\beta$  the torsional angle per unit rod length), and  $k = 1$  again (only the 2D case being of relevance,  $\Omega$  being the cross-section of the rod).

**Boundary conditions.** Equation (1.87) must be supplemented with a boundary condition at each point of the boundary  $\partial\Omega$ . Only two types of boundary conditions are considered here, where either the normal flux or the potential is prescribed at any given boundary point. These boundary conditions have the form

$$-k \nabla \varphi \cdot \mathbf{n} = q^D \quad \text{on } S_q \quad \varphi = \varphi^D \quad \text{on } S_\varphi. \quad (1.88)$$

with  $S_\varphi \cap S_q = \emptyset$ ,  $S_\varphi \cup S_q = \partial\Omega$  and where  $\varphi^D$  and  $q^D$  are prescribed values of potential and flux, respectively.

---

<sup>8</sup>Some media have anisotropic conduction properties, in which case the scalar conductivity must be replaced with a (second-order, positive definite) conductivity tensor  $\mathbf{k}$  and the constitutive relation (1.86) becomes  $\mathbf{q} = -\mathbf{k} \cdot \nabla \varphi$ . The derivation of the relevant weak formulation is straightforward and is left as an exercise

**Weak formulation.** In full analogy with the linear elastic case, the numerical solution of the potential problems (1.87)-(1.88) is based on a weak formulation. Let  $\mathcal{C}(\varphi^D)$  define the space of potential fields that are admissible with prescribed values  $\varphi^D$  over  $S_\varphi$ :

$$\mathcal{C}(\varphi^D) = \{w \mid w \text{ sufficiently regular in } \Omega \text{ and } w = \varphi^D \text{ on } S_\varphi\} \quad (1.89)$$

To establish the weak formulation verified by  $\varphi$ , equation (1.87) is multiplied by a scalar virtual field  $w \in \mathcal{C}(0)$  and integrated over  $\Omega$ , to obtain

$$-\int_{\Omega} \operatorname{div}(k\nabla\varphi)w \, dV = \int_{\Omega} gw \, dV.$$

Then, applying the divergence theorem to the first integral, one obtains

$$\begin{aligned} \int_{\Omega} \operatorname{div}(k\nabla\varphi)w \, dV &= \int_{\partial\Omega} (k\nabla\varphi \cdot \mathbf{n})w \, dS - \int_{\Omega} k\nabla\varphi \cdot \nabla w \, dV \\ &= - \int_{S_q} q^D w \, dS - \int_{\Omega} k\nabla\varphi \cdot \nabla w \, dV, \end{aligned}$$

with the last equality stemming from the boundary condition (1.88b) and the assumption  $w \in \mathcal{C}(0)$ . Combining the two previous identities leads to the weak formulation

find  $\varphi \in \mathcal{C}(\varphi^D)$  such that

$$\int_{\Omega} k\nabla\varphi \cdot \nabla w \, dV = \int_{\Omega} gw \, dV - \int_{S_q} q^D w \, dS \quad \forall w \in \mathcal{C}(0). \quad (1.90)$$

## 1.7 Supplementary material

### 1.7.1 Boundary conditions

To keep the exposition simple, only two possibilities were considered for the boundary conditions at a given boundary point, by prescribing either the traction vector or the displacement vector. Other types of conditions may occur, and variational or weak formulations must be modified accordingly.

As an example, consider the case where the boundary  $\partial\Omega$  involves three disjoint surfaces  $S_u, S_T, S_{\text{sym}}$ , with boundary conditions defined by

$$\boldsymbol{\sigma}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = \mathbf{T}^D(\mathbf{x}) \quad (\mathbf{x} \in S_T), \quad (1.91)$$

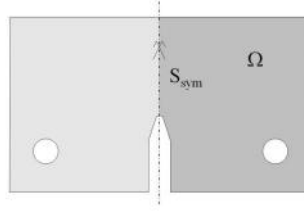
$$\mathbf{u}(\mathbf{x}) = \mathbf{u}^D(\mathbf{x}) \quad (\mathbf{x} \in S_u), \quad (1.92)$$

$$\mathbf{u}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = 0 \quad (\mathbf{x} \in S_{\text{sym}}), \quad (1.93)$$

$$\boldsymbol{\sigma}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) - \sigma_n(\mathbf{x})\mathbf{n}(\mathbf{x}) = \mathbf{0} \quad (\mathbf{x} \in S_{\text{sym}}), \quad (1.94)$$

where  $\sigma_n := \mathbf{n} \cdot \boldsymbol{\sigma} \cdot \mathbf{n}$  is the (scalar) normal stress. As before, surfaces  $S_u, S_T$  support prescribed displacements and tractions, respectively, while conditions (1.93)





**Figure 1.5:** Example of symmetry with respect to a plane. The computational domain  $\Omega$  supporting the formulation to be solved is here the right half of the actual solid body.

and (1.94) corresponds to allowing a frictionless tangential motion along the surface  $S_{\text{sym}}$ . This new type of condition is useful in practice because it reflects the expected behavior of displacements and tractions on a geometrical symmetry plane when the loading conditions are also symmetric. They allow to formulate the equilibrium problem on a smaller domain through exploitation of symmetry with respect to planes (see Figure 1.5 for an example).

The definition of spaces of admissible fields must then be adapted to this new boundary condition structure. They are now defined by

$$\mathcal{C}(\mathbf{u}^D) = \{ \mathbf{v} \mid \mathbf{v} \text{ continuous in } \Omega, \mathbf{v} = \mathbf{u}^D \text{ on } S_u \text{ and } \mathbf{u} \cdot \mathbf{n} = 0 \text{ on } S_{\text{sym}} \},$$

$$\mathcal{C}(\mathbf{0}) = \{ \mathbf{v} \mid \mathbf{v} \text{ continuous in } \Omega, \mathbf{v} = \mathbf{0} \text{ on } S_u \text{ and } \mathbf{u} \cdot \mathbf{n} = 0 \text{ on } S_{\text{sym}} \}.$$

and

$$\mathcal{S}(\mathbf{T}^D, \mathbf{f}) = \left\{ \boldsymbol{\tau} \mid \operatorname{div} \boldsymbol{\tau} + \rho \mathbf{f} = 0 \text{ in } \Omega, \begin{cases} \boldsymbol{\tau} \cdot \mathbf{n} = \mathbf{T}^D \text{ on } S_T \\ \boldsymbol{\tau} \cdot \mathbf{n} - \tau_n \mathbf{n} = \mathbf{0} \text{ on } S_{\text{sym}} \end{cases} \right\},$$

Following the argumentation of Section 1.3, the potential and complementary energy are still defined by expressions (1.29) and (1.30). The symmetry surface  $S_{\text{sym}}$  is involved in the definition of sets of admissible fields, and also implicitly in the fact that  $S_u$  and  $S_T$  no longer define a partition of the boundary  $\partial\Omega$  of the computational domain.

### 1.7.2 Error estimation for the approximate solution

Assume that methods for minimizing both the potential energy and the complementary energy are simultaneously available for solving a given elastic equilibrium problem of the form (1.1a–e). The accuracy of the pair  $(\mathbf{u}_N, \boldsymbol{\sigma}_N)$  made of the optimal kinematically admissible displacement and statically admissible stress yielded by the respective minimizations (restricted to  $N$ -dimensional approximation spaces) may then be measured by evaluating the error in constitutive equation

$$\mathcal{E}(\mathbf{u}_N, \boldsymbol{\sigma}_N) = P(\{\alpha^{\min}\}) + P^*(\{\beta^{\min}\}),$$

which is zero if both  $\mathbf{u}_N$  and  $\boldsymbol{\sigma}_N$  are exact solutions. A meaningful interpretation of the value of  $\mathcal{E}(\mathbf{u}_N, \boldsymbol{\sigma}_N)$  requires setting it a non-dimensional form. For example, one may compute the ratio

$$\frac{P(\{\alpha^{\min}\}) + P^*(\{\beta^{\min}\})}{|P(\{\alpha^{\min}\})| + |P^*(\{\beta^{\min}\})|},$$

which compares the deviation to zero of  $\mathcal{E}(\mathbf{u}_N, \boldsymbol{\sigma}_N)$  to the absolute magnitude of potential (or complementary energy). Accurate solutions then correspond to small values of this ratio.

## 1.8 Summary

In this chapter, we showed on elastic equilibrium problems how to establish and exploit weak formulations that are equivalent to the original boundary-value problem stated in strong form (i.e. using pointwise field equations and boundary conditions). These weak formulations either result from the virtual work principle, in which the compatibility and constitutive relations are applied, or are the stationarity equations of energy minimization principles (when they are available, as is the case for linear elasticity).

A general approach for seeking approximate solutions then rests upon the Galerkin method, which consists in representing unknown and virtual displacements using the same basis, which is chosen *a priori*. Chapters 2 and 3 will then show that the finite element method is, in its essence, nothing than a particular form of the Galerkin method whose basis functions are designed to have a "local" geometrical support.

## Isoparametric elements

---

In the field of solid mechanics the finite element method has gained a dominant status among numerical methods largely thanks to its flexibility in treating arbitrary configurations and loadings. This is achieved by representing the analysis domain as a collection of elements of simple shape (the *finite elements*) on each of which the unknown fields are approximated by means of polynomial basis functions. In most of this book we will focus on the so called *displacement-based* finite element method, in which the displacement is taken as primary unknown. In this framework, the notion of isoparametric finite elements is of paramount importance for the representation of geometry and for the interpolation of fields in two- and three-dimensional media.

In Section 2.1, as an introductory example, we first analyse a three-node triangular element for plane-strain analyses. Next the notion of isoparametric interpolation is developed and extended in Section 2.2 to a general setting. Proceeding as in Chapter 1, the main focus of the presentation will be on the typical vector-valued problem of linear solid mechanics (Section 1.1). However, exploiting similarities, the same concepts will be rapidly extended to the scalar boundary value problems introduced in Section 1.6 through examples in Chapter 3.

Some of the concepts will be presented resorting to a notation which reflects the data structure adopted in the codes developed for this book and commented in later chapters. This is done in full generality without limiting the validity of the presentation.

### 2.1 Example: plane strain analyses and linear triangles

In this introductory section we will look for approximate solutions of a specific class of three-dimensional problems defined on a cylindrical domain which is unbounded along the axial direction  $x_3$ . Moreover, the boundary conditions are assumed to be compatible with a displacement field of the form:

$$\mathbf{u}(x_1, x_2, x_3) = u_1(x_1, x_2)\mathbf{e}_1 + u_2(x_1, x_2)\mathbf{e}_2. \quad (2.1)$$

In particular, forces and displacements are assumed to be prescribed only along the directions  $\mathbf{e}_1$  and  $\mathbf{e}_2$ , and also to be invariant with respect to  $x_3$ . In the literature of solid mechanics these conditions are summarized by the term *plane-strain*

*conditions.* In this context, it is convenient to denote by  $\Omega$  the two-dimensional cross section lying on the plane  $x_3 = 0$ , and by  $S_u$  and  $S_T$  the partition of the curve  $\partial\Omega$  bounding  $\Omega$  into lines where displacements and tractions are enforced, respectively.

### 2.1.1 Mesh

It is natural and straightforward to introduce, for any planar region  $\Omega$ , a partition  $\Omega_h$ , called *mesh*, consisting of a collection of  $N_E$  triangular regions with straight sides. The vertices of the triangles are called *nodes* (Figure 2.1) and the triangles themselves are called *elements*. The nodes, of coordinates  $x^{(n)}$ , are numbered from 1 to  $N_N$  according to a *global numbering*.

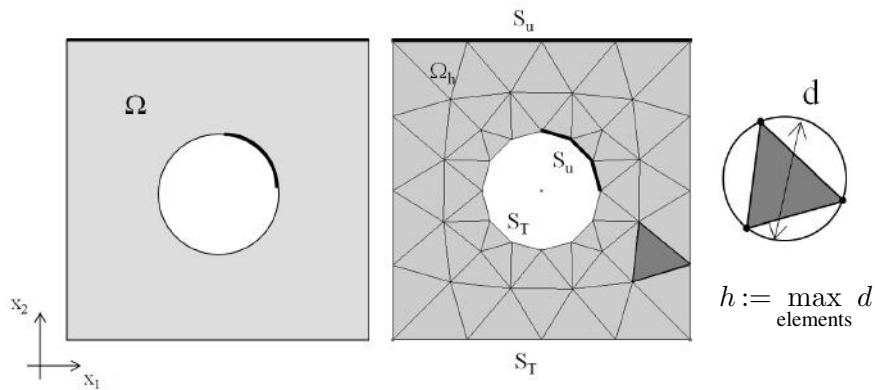
The approximation consists in the fact that the boundary  $\partial\Omega$  of  $\Omega$ , which is generally curvilinear, is represented by a sequence of straight segments. It is anyway clear that:

- (i) the original  $\Omega$  can be approximated to any desired accuracy by employing a mesh of sufficiently small triangles;
- (ii) for any  $\Omega$  of polygonal shape, one can achieve  $\Omega_h = \Omega$ .

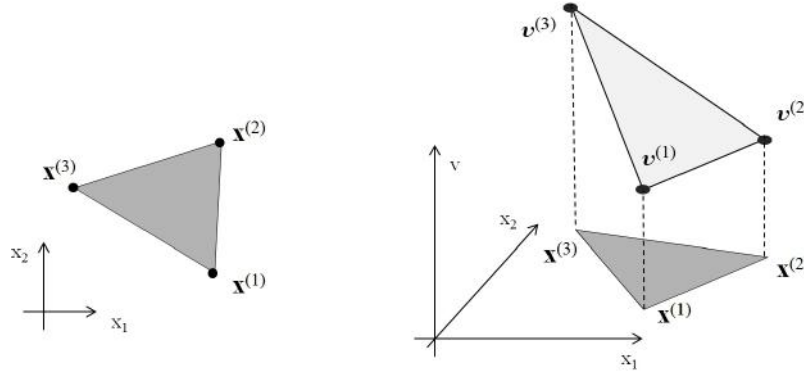
The index  $h$  in  $\Omega_h$  symbolically represents the characteristic size of the elements. It is defined in terms of the diameters  $d$  of the circles circumscribed to the triangles in the mesh (Figure 2.1). This notation, adopted here, is frequently used in the literature on finite elements and is also employed to indicate the approximate nature of other quantities (for instance the displacements in the equation 2.2 below).

### 2.1.2 Local interpolation

Having fixed the choice of the mesh, we seek an approximate solution in the space of admissible fields which are *linear* over each triangle and *continuous*.



**Figure 2.1:** Mesh made of three-noded triangular elements and definition of parameter  $h$



**Figure 2.2:** Three-node triangular element (left), linear interpolation of nodal values for  $v$  (right).

A systematic procedure for representing this space rests on the concept of *nodal displacements*. Let us focus on the  $e$ -th triangle  $E_e$  of the mesh, defined by its three nodes of global number  $e_1, e_2$  and  $e_3$ . The indices 1, 2, 3 define a *local ordering* of the nodes on the element. When analysing a single element it is customary, for the sake of simplicity, to identify the nodes using the local numbering. Hence, the three nodal coordinates are denoted  $\mathbf{x}^{(1)}$ ,  $\mathbf{x}^{(2)}$ ,  $\mathbf{x}^{(3)}$  and values of the displacement field on the nodes, which will be henceforth called *nodal values*, are  $v^{(1)}$ ,  $v^{(2)}$ ,  $v^{(3)}$ , respectively. The nodal values (Figure 2.2) are interpolated linearly over the element according to:

$$v_h(\mathbf{x}) = N_1(\mathbf{x})v^{(1)} + N_2(\mathbf{x})v^{(2)} + N_3(\mathbf{x})v^{(3)} \quad (\mathbf{x} \in E_e), \quad (2.2)$$

where the functions  $N_i$ , also called *local shape functions*, are linear in  $(x_1, x_2)$ :

$$N_i(\mathbf{x}) = c_0^{(i)} + c_1^{(i)}x_1 + c_2^{(i)}x_2 \quad (i = 1, 2, 3) \quad (2.3)$$

and must satisfy the conditions:

$$N_i(\mathbf{x}^{(j)}) = \delta_{ij} \quad (i, j = 1, 2, 3) \quad (2.4)$$

Imposing (2.4), one can immediately obtain the explicit expression of  $c_j^{(i)}$  in (2.3). For example, one has for  $N_1$ :

$$c_0^{(1)} = (x_1^{(2)}x_2^{(3)} - x_2^{(2)}x_1^{(3)})/2S, \quad c_1^{(1)} = x_2^{(23)}/2S, \quad c_2^{(1)} = x_1^{(32)}/2S \quad (2.5)$$

where  $x_i^{(k\ell)} := x_i^{(k)} - x_i^{(\ell)}$  and  $S$  is the area of triangle  $E_e$ , given by

$$2S = \|\mathbf{x}^{(21)} \wedge \mathbf{x}^{(31)}\| = x_1^{(21)}x_2^{(31)} - x_2^{(21)}x_1^{(31)}.$$

The corresponding coefficients entering the shape functions  $N_2(\mathbf{x})$  and  $N_3(\mathbf{x})$  are then obtained from (2.5) by circular permutation of (1, 2, 3).

### 2.1.3 Global interpolation

In order to build a global representation of an admissible displacement field, boundary conditions are respected by setting  $v_j^{(n)} = u_j^D(\mathbf{x}^n)$  on  $S_u$  and  $v_j^{(n)}$  free otherwise. If the displacement field is now built over every triangle according to (2.2), (2.3), (2.5), then it is continuous over the whole approximate domain  $\Omega_h$  and can be recast in the form:

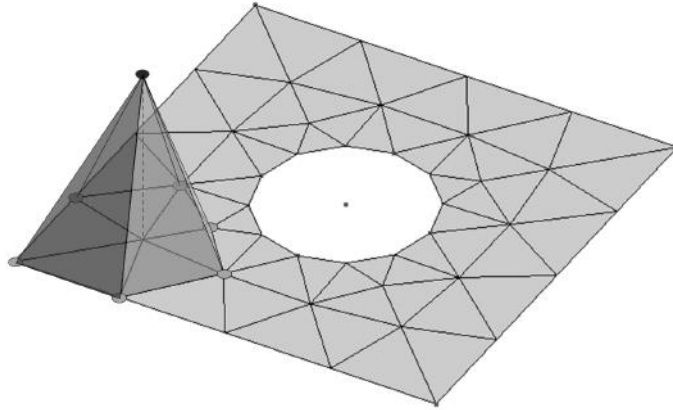
$$\mathbf{v}_h(\mathbf{x}) = \sum_{n=1}^{N_N} \tilde{N}_n(\mathbf{x}) \mathbf{v}^{(n)} = \mathbf{u}_h^{(D)}(\mathbf{x}) + \mathbf{v}_h^{(0)}(\mathbf{x}) \quad (2.6)$$

with

$$\begin{aligned} \mathbf{u}_h^{(D)}(\mathbf{x}) &:= \sum_{(n,j) | v_j^{(n)} \text{ prescr.}} \tilde{N}_n(\mathbf{x}) u_j^D(\mathbf{x}^{(n)}) \mathbf{e}_j, \\ \mathbf{v}_h^{(0)}(\mathbf{x}) &:= \sum_{(n,j) | v_j^{(n)} \text{ free}} \tilde{N}_n(\mathbf{x}) v_j^{(n)} \mathbf{e}_j \end{aligned}$$

where:

- $n = 1, \dots, N_N$  refers to the *global numbering* of the nodes in the mesh since, by convention, global interpolations refer to global numbering;
- The *global shape function*  $\tilde{N}_n(\mathbf{x})$  associated with a node  $\mathbf{x}^{(n)}$  has a support  $\Omega^{(n)}$  which is local as it is limited to the patch of triangles sharing  $\mathbf{x}^{(n)}$  as a vertex; it is linear over each triangle, is equal to 1 in  $\mathbf{x} = \mathbf{x}^{(n)}$  and vanishes at all the other nodes of the patch (Figure 2.3); actually it is the collection of all the local shape functions associated with node  $\mathbf{x}^{(n)}$  on the elements of  $\Omega^{(n)}$ .

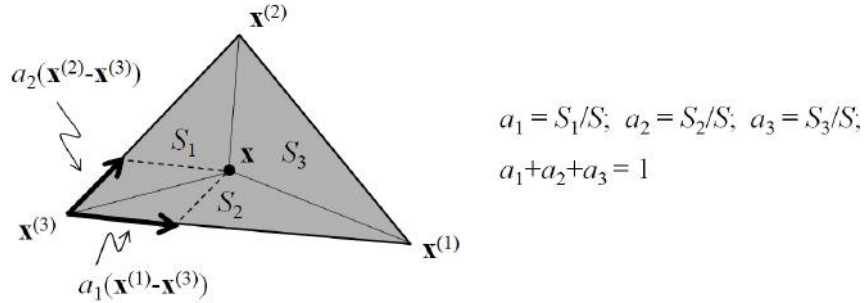


**Figure 2.3:** Global shape function defined starting from linear triangles.

The representation (2.6) is of the form (1.37), where the particular field  $\mathbf{u}_h^{(D)} \in \mathcal{C}_h(\mathbf{u}^D)$  corresponds to the first sum, the coefficients  $\alpha_K$  are the unknown displacements components  $u_j^{(n)}$  and the basis functions  $\varphi^K(\mathbf{x}) \in \mathcal{C}_h(\mathbf{0})$  are given by  $\varphi^K(\mathbf{x}) = \tilde{N}_n(\mathbf{x})\mathbf{e}_j$  such that  $v_j^{(n)}$  is free, assuming a suitable renumbering  $(n, j) \rightarrow K$ . It is thus possible to build an approximate problem according to the Galerkin method described in Section 1.4. It is worth remarking that the support is now the *approximate* domain  $\Omega_h$  and the shape functions are defined over  $\Omega_h$  and not over the exact domain  $\Omega$ . In particular the stiffness matrix and the vector of generalized forces can be obtained through integrals of the type (1.40) and (1.41) computed on the approximate domain  $\Omega_h$  and its boundary, while the function  $\mathbf{u}_h^{(D)}(\mathbf{x})$  is kinematically admissible with the displacement data *in the sense of the finite element approximation* (since  $\mathbf{u}_h^{(D)}(\mathbf{x}) \in \mathcal{C}_h(\mathbf{u}^D)$  and  $\mathbf{v}_h^{(0)}(\mathbf{x}) \in \mathcal{C}_h(\mathbf{0})$  by construction).

#### 2.1.4 Towards the concept of isoparametric elements

As an introduction to the general theory of isoparametric elements, we reformulate here the linear triangle analysed in the previous section under a different perspective. Let us consider Figure 2.4 and let  $\mathbf{x}$  be a generic point within the element. Connecting  $\mathbf{x}$  with the nodes  $\mathbf{x}^{(i)}$ , three sub-triangles of area  $S_i$  are introduced. The three ratios  $a_i = S_i/S$  are usually called *area coordinates* and sum to unity.



**Figure 2.4:** Parametric coordinates and area coordinates for a linear triangle.

With the definitions given in Figure 2.4, one has

$$\mathbf{x} = \mathbf{x}^{(3)} + a_1(\mathbf{x}^{(1)} - \mathbf{x}^{(3)}) + a_2(\mathbf{x}^{(2)} - \mathbf{x}^{(3)}) = a_1\mathbf{x}^{(1)} + a_2\mathbf{x}^{(2)} + a_3\mathbf{x}^{(3)}.$$

This equation can be recast in matrix form as:

$$\begin{Bmatrix} 1 \\ x_1 \\ x_2 \end{Bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ x_1^{(1)} & x_1^{(2)} & x_1^{(3)} \\ x_2^{(1)} & x_2^{(2)} & x_2^{(3)} \end{bmatrix} \begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} \quad (2.7)$$

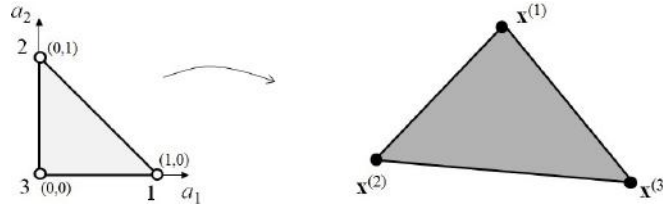
while the inverse relationship is:

$$\begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \frac{1}{2S} \begin{bmatrix} x_1^{(2)} x_2^{(3)} - x_1^{(3)} x_2^{(2)} & x_2^{(23)} & x_1^{(32)} \\ x_1^{(3)} x_2^{(1)} - x_1^{(1)} x_2^{(3)} & x_2^{(31)} & x_1^{(13)} \\ x_1^{(1)} x_2^{(2)} - x_1^{(2)} x_2^{(1)} & x_2^{(12)} & x_1^{(21)} \end{bmatrix} \begin{Bmatrix} 1 \\ x_1 \\ x_2 \end{Bmatrix} \quad (2.8)$$

with  $x_k^{(ij)} := x_k^{(i)} - x_k^{(j)}$ . The coefficients of the matrix in (2.8) coincide with (2.5) and its circular permutations, which implies that  $N_i = a_i$ :

$$\mathbf{x} = a_1 \mathbf{x}^{(1)} + a_2 \mathbf{x}^{(2)} + a_3 \mathbf{x}^{(3)} \quad (2.9)$$

$$\mathbf{v}_h = a_1 \mathbf{v}^{(1)} + a_2 \mathbf{v}^{(2)} + a_3 \mathbf{v}^{(3)} \quad (2.10)$$



**Figure 2.5:** Triangular finite element. Mapping from the parametric space

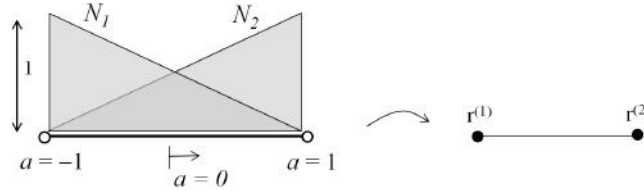
Two remarks are worth stressing at this stage:

1. Equation (2.9) can be interpreted (see Figure 2.5) as a mapping of the unit triangle from the parametric space  $(a_1, a_2)$  onto the physical element; the parameters of the mapping are the nodal coordinates. Since the mapping is linear, three nodes are required.
2. Physical coordinates (geometry) and displacements are interpolated in terms of the same shape functions which, themselves, are expressed in terms of parametric coordinates.

As will be stressed in the sequel, these two points correspond to the founding ideas of the theory of isoparametric elements.

### Exercise: 2-node line element

Another simple example comes from the application developed in Section 1.5.6 for the analysis of the hollow sphere under pressure. Let us indeed consider a straight segment of length  $h$  and nodal coordinates  $r^{(1)}$  and  $r^{(2)}$ .



**Figure 2.6:** Shape functions for a two-node line element



Using the abscissa  $s$ ,  $0 \leq s \leq h$  and defining the two linear shape functions:

$$N_1(s) = 1 - \frac{s}{h}, \quad N_2(s) = \frac{s}{h}$$

the representation of geometry can be expressed in the form:

$$r(s) = r^{(1)} + s = N_1(s)r^{(1)} + N_2(s)r^{(2)}$$

In the literature on finite elements it is however more customary to employ, instead of  $s$ , a parametric coordinate  $a$ ,  $-1 \leq a \leq 1$ , such that  $a = -1 + 2s/h$  and the shape functions become (Figure 2.6):

$$N_1(a) = \frac{1}{2}(1 - a), \quad N_2(a) = \frac{1}{2}(1 + a) \quad (2.11)$$

Finally the representation of geometry and linear interpolation of displacements can be expressed as (see also 1.74):

$$r = N_1(a)r^{(1)} + N_2(a)r^{(2)} \quad (2.12a)$$

$$v_h = N_1(a)v^{(1)} + N_2(a)v^{(2)} \quad (2.12b)$$

The two representations are hence of the same form as (2.9)-(2.10), respectively.

## 2.2 The concept of isoparametric element

The approach described in the previous section is limited to plane problems, unavoidably approximates curved boundaries with polygonal shapes and leads to a poor representation of strains (and stresses) since they are piecewise constant. These are strong drawbacks, especially in view of applications to complicated three-dimensional structures (see the example of Figure 2.7) where the accurate prediction of stress concentration is one of the main goals.

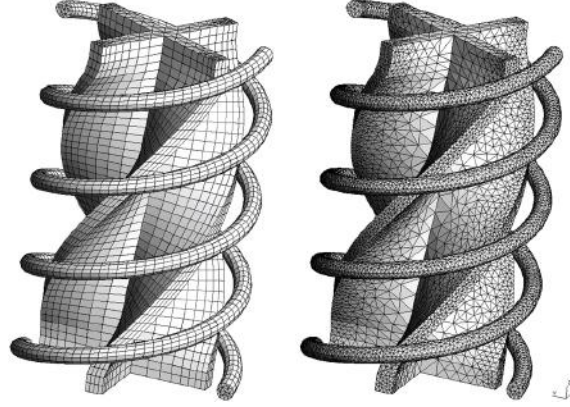
The removal of these limitations can be achieved by suitably generalising the point of view of Section 2.1.4. This will lead to the introduction of the notion of isoparametric elements which will easily allow to:

- (a) define three dimensional finite elements;
- (b) better approximate curved boundaries and guarantee conformal meshes;
- (c) interpolate displacements with higher order shape functions and improve the representation of strains and stresses

### 2.2.1 A family of isoparametric elements

Any isoparametric element is conceived as a mapping of a suitable (reference) element  $\Delta$  of normalized size and shape, defined in the parametric space, onto the actual element  $E$  in physical space. The mapping has a form similar to (2.9):

$$\mathbf{x} = \sum_{k=1}^{n_e} N_k(\mathbf{a}) \mathbf{x}^{(k)} \quad \mathbf{x} \in E, \mathbf{a} \in \Delta \quad (2.13)$$



**Figure 2.7:** Mesh of complicated three dimensional solids (pictures taken from <http://geuz.org/gmsh/>, web-site of the free mesh-generation code GMSH).

where  $n_e$  is the number of nodes (of coordinates  $\mathbf{x}^{(k)}$ ) and  $N_k(\mathbf{a})$  are  $n_e$  shape functions defined in the parametric space;  $\mathbf{a}$  denotes the vector of *parametric coordinates* of a point in the reference element  $\Delta$ .

Some examples of such elements are given in Tables 2.1, 2.2 and 2.3 for 1D, 2D and 3D applications, respectively. The names adopted for the elements are not universal, but are employed throughout this book to simplify both the ensuing discussions and the implementation in the codes developed in the book.

All these elements are based on the following (non-exhaustive) list of possible reference unit elements  $\Delta \subset \mathbb{R}^D$

- the unit segment ( $D = 1$ ):  
 $\Delta = \{a \mid -1 \leq a \leq 1\};$
- the unit square ( $D = 2$ ):  
 $\Delta = \{(a_1, a_2) \mid -1 \leq a_1, a_2 \leq 1\};$
- the unit triangle ( $D = 2$ ):  
 $\Delta = \{(a_1, a_2) \mid (a_1, a_2) \geq (0, 0), 1 - a_1 - a_2 \geq 0\};$
- the unit cube ( $D = 3$ ):  
 $\Delta = \{(a_1, a_2, a_3) \mid -1 \leq a_1, a_2, a_3 \leq 1\};$
- the unit tetrahedron ( $D = 3$ ):  
 $\Delta = \{(a_1, a_2, a_3) \mid (a_1, a_2, a_3) \geq (0, 0, 0), 1 - a_1 - a_2 - a_3 \geq 0\}.$

Each type of isoparametric finite element is also characterized by the number of nodes and by the associated shape functions. Anticipating a property (P1) which will be discussed in the sequel, the shape functions are designed in such a way that there exist a *fixed* set of  $n_e$  points in the reference element, called *master nodes*, which are mapped onto the physical nodes of coordinates  $\mathbf{x}^{(k)}$ . In general, the term *master element* denotes the combination of a specific choice of unit element and a specific choice of master nodes.

Master and physical elements	Shape functions
<p>B2</p> <p>master element</p> <p>physical element</p> <p>1 <math>a = -1</math> <math>a = 0</math> 2 <math>a = 1</math></p>	$N_1 = \frac{1}{2}(1 - a)$ $N_2 = \frac{1}{2}(1 + a)$
<p>B3</p> <p>1 <math>a = -1</math> 3 <math>a = 0</math> 2 <math>a = 1</math></p>	$N_1 = \frac{1}{2}a(1 - a)$ $N_2 = \frac{1}{2}a(1 + a)$ $N_3 = 1 - a^2$

**Table 2.1:** Examples of standard one-dimensional finite elements.

The left picture in Tables 2.1, 2.2 and 2.3 always defines the master element in the parametric space, with the master nodes shown by a circle, while the right picture depicts a “representative” physical element with physical nodes denoted by black dots.

The line elements in Table 2.1 are referred to as B2 (two nodes, linear interpolation) and B3 (three nodes, quadratic interpolation) elements. The order of the nodes on the B3 master element should be remarked, node 3 being placed in the middle at  $a = 0$ .

Table 2.2 collects some 2D elements. The simplest possible choices are the linear triangle (T3), already analysed in Section 2.1, and the bilinear quadrangle (Q4) which generate physical elements with straight sides. In these cases the master and physical nodes coincide with the vertices of the elements. Higher order elements, like the quadratic triangle T6 (Figure 2.8) and the 8-node quadrangle Q8 also allow to represent curved boundaries. In order to enrich the interpolation of the T3 and Q4, respectively, new master nodes are introduced at the midpoint of each side, according to a conventional ordering. It is worth stressing that the interpolation chosen is a complete linear polynomial for the T3 element and a complete quadratic polynomial for the T6 element. Other elements, on the contrary, feature polynomial interpolation bases that are incomplete: the Q4 element uses a complete linear interpolation plus the bilinear term  $a_1 a_2$ , and the Q8 uses a complete quadratic interpolation plus some of the cubic terms.

Similarly, two examples for three-dimensional elements are provided in Table 2.3: the linear tetrahedron P4 and the 8-node cube H8. The formulation of their higher order counterparts (ten-node tetrahedron P10 and twenty-node cube) is left as an exercise.

All these elements are endowed with specific properties which will turn out to be of the greatest importance for the formulation of the method.

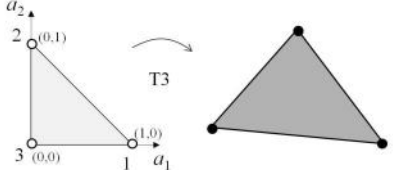
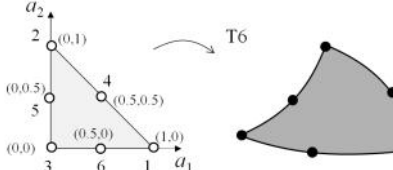
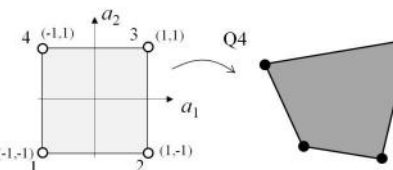
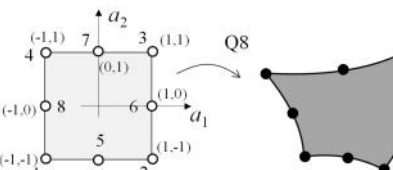
Master and physical elements	Shape functions
	$N_1 = a_1$ $N_2 = a_2$ $N_3 = 1 - a_1 - a_2$
	$N_1 = a_1(2a_1 - 1)$ $N_2 = a_2(2a_2 - 1)$ $N_3 = (1 - a_1 - a_2)(1 - 2a_1 - 2a_2)$ $N_4 = 4a_1a_2$ $N_5 = 4a_2(1 - a_1 - a_2)$ $N_6 = 4a_1(1 - a_1 - a_2)$
	$N_1 = \frac{1}{4}(1 - a_1)(1 - a_2)$ $N_3 = \frac{1}{4}(1 + a_1)(1 + a_2)$ $N_2 = \frac{1}{4}(1 + a_1)(1 - a_2)$ $N_4 = \frac{1}{4}(1 - a_1)(1 + a_2)$
	$N_1 = \frac{1}{4}(1 - a_1)(1 - a_2)(-1 - a_1 - a_2)$ $N_2 = \frac{1}{4}(1 + a_1)(1 - a_2)(-1 + a_1 - a_2)$ $N_3 = \frac{1}{4}(1 + a_1)(1 + a_2)(-1 + a_1 + a_2)$ $N_4 = \frac{1}{4}(1 - a_1)(1 + a_2)(-1 - a_1 + a_2)$ $N_5 = \frac{1}{2}(1 - a_1^2)(1 - a_2)$ $N_6 = \frac{1}{2}(1 - a_2^2)(1 + a_1)$ $N_7 = \frac{1}{2}(1 - a_1^2)(1 + a_2)$ $N_8 = \frac{1}{2}(1 - a_2^2)(1 - a_1)$

Table 2.2: Examples of standard two-dimensional finite elements.

**Exact representation of nodes (property P1).** For every type of element, the shape functions satisfy the condition that

$$N_k(\mathbf{a}^{(\ell)}) = \delta_{k\ell} \quad (1 \leq k, \ell \leq n_e). \quad (2.14)$$

where  $\mathbf{a}^{(\ell)}$  denotes the parametric coordinates of the  $\ell$ -th master node. Thanks to this property, the representation (2.13) is guaranteed to be exact at the nodes, i.e.

$$\mathbf{x}^{(\ell)} = \sum_{k=1}^{n_e} N_k(\mathbf{a}^{(\ell)}) \mathbf{x}^{(k)} \quad (1 \leq \ell \leq n_e), \quad (2.15)$$

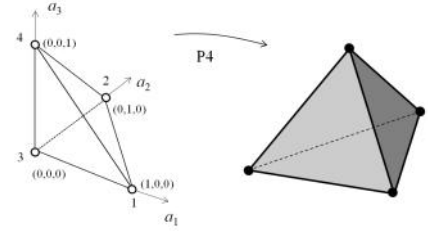
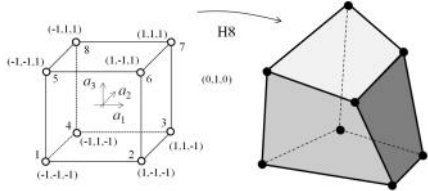
Master and physical elements	Shape functions
	$N_1(a_1, a_2, a_3) = a_1$ $N_2(a_1, a_2, a_3) = a_2$ $N_3(a_1, a_2, a_3) = a_3$ $N_4(a_1, a_2, a_3) = 1 - a_1 - a_2 - a_3$
	$N_1(a_1, a_2, a_3) = \frac{1}{8}(a_1 - 1)(a_2 - 1)(a_3 - 1)$ $N_2(a_1, a_2, a_3) = \frac{1}{8}(a_1 + 1)(a_2 - 1)(a_3 - 1)$ $N_3(a_1, a_2, a_3) = \frac{1}{8}(a_1 + 1)(a_2 + 1)(a_3 - 1)$ $N_4(a_1, a_2, a_3) = \frac{1}{8}(a_1 - 1)(a_2 + 1)(a_3 - 1)$ $N_5(a_1, a_2, a_3) = \frac{1}{8}(a_1 - 1)(a_2 - 1)(a_3 + 1)$ $N_6(a_1, a_2, a_3) = \frac{1}{8}(a_1 - 1)(a_2 + 1)(a_3 + 1)$ $N_7(a_1, a_2, a_3) = \frac{1}{8}(a_1 + 1)(a_2 + 1)(a_3 + 1)$ $N_8(a_1, a_2, a_3) = \frac{1}{8}(a_1 + 1)(a_2 - 1)(a_3 + 1)$

Table 2.3: Examples of standard three-dimensional finite elements.

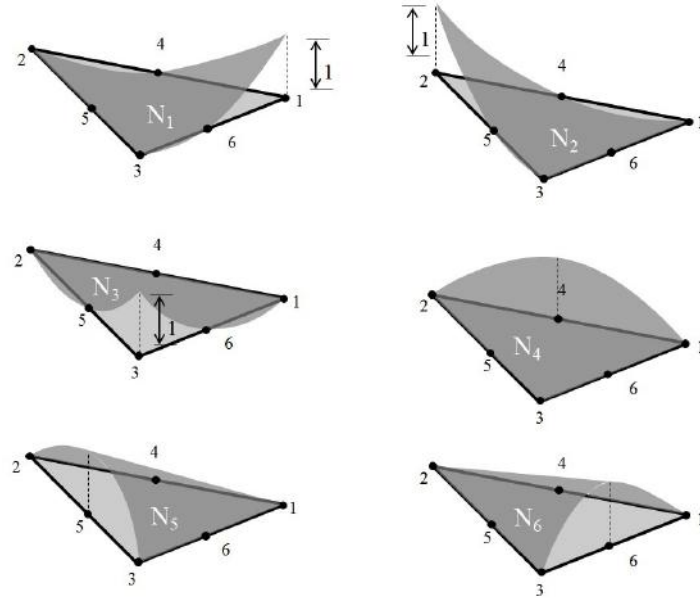


Figure 2.8: Shape function for the element T6

**Representation of edges and faces (property P2).** A given shape function is associated with a node and vanishes on all the geometrical features (edges and faces) of the element which do not contain that node. Hence the geometrical representation of edges and faces only depends on the coordinates of the nodes lying on that specific feature.

As a consequence of this remark and of the expression of the shape functions given in Tables 2.1, 2.2 and 2.3 (and others defined similarly), the elements analysed herein have an intrinsic hierarchical structure, in the sense that the restriction of one element (and of its shape functions) to a feature of its boundary coincides with a lower-dimensional isoparametric element. As an example, let us consider a T6 triangle and a B3 bar. Each edge of the T6 master element is a segment with three nodes and is equivalent to the master element of a B3. In Section 2.2.4, it is shown that the restriction of the shape functions of the T6 to any of the edges coincide, after a suitable change of coordinates and node renumbering, with the shape functions of the B3 element. Another example is provided by the P4 tetrahedron, whose faces and edges coincide with T3 triangles and B2 segments.

**Unit sum (property P3).** For all the elements described so far, one has:

$$\sum_{k=1}^{n_e} N_k(\mathbf{a}) = 1, \quad \forall \mathbf{a} \in \Delta \quad (2.16)$$

i.e. the shape functions always sum to unity (they are said to achieve a *partition of unity*).

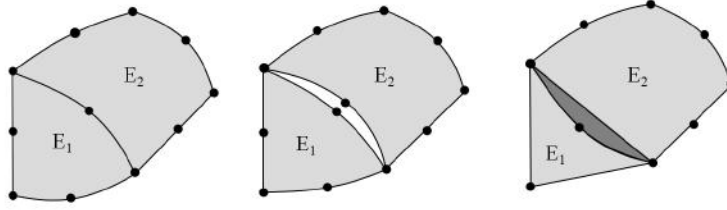
### 2.2.2 Mesh creation and mesh conformity

Having introduced a family of isoparametric elements and their main properties, we are now ready to discuss their employment for the representation of the domain geometry in a general context where  $\Omega \subset \mathbb{R}^D$ .

We start by assuming a partition of  $\Omega$  into  $N_E$  regions  $\mathcal{E}_e$  ( $1 \leq e \leq N_E$ ) of admissible shape in the sense that, for each region, there exists a unit element  $\Delta_e$ , of the family analysed before, which can be continuously and exactly mapped onto  $\mathcal{E}_e$ . This is an abstract operation, in the sense that we do not provide the expression of the mappings at this level. The only mild requirement is associated with the notion of *conformity* which will be further discussed in the sequel and that can be here expressed as follows. Any  $\mathcal{E}_e$  is defined by a list of points, edges and (in 3D) faces which we will here call geometrical features. Conformity simply requires that two neighbouring regions can only share (if they do) complete geometrical features (e.g. they cannot share only a portion of a face or of an edge) and that perfect continuity must be guaranteed between the regions.

While at the continuous level is straightforward to achieve a conformal abstract partition, this task is arduous at the discrete level. Non-physical holes or intersections like the situations depicted in Figure 2.9 on the right must be avoided.

If isoparametric elements are adopted, this can be easily achieved by respecting the following strategy. Each region  $\mathcal{E}_e$  is approximated with an isoparametric



**Figure 2.9:** Conformal (left) and non-conformal meshes (middle and right) of two elements.

finite element  $E_e$  obtained from the unit element  $\Delta_e$  after a suitable (not arbitrary!) selection of the type of element and of physical nodes on the boundary of  $\mathcal{E}_e$ . If two regions share a geometrical feature (edge or face), the two elements selected for their discretization must *reduce* to the same type of isoparametric element on that feature (see property P2 of Section 2.2). Moreover the coordinates of the physical nodes on the feature must be the same on both elements. The properties of isoparametric elements guarantee that the geometrical representation of the common feature will be exactly the same.

The situation of Figure 2.9 on the right is not conformal since the T3 triangle reduces to a 2-node B2 segment on the common edge, while the Q8 element reduces to a B3 line; on the contrary, the middle picture is not conformal because on the two elements the middle master node on the common edge is mapped to different physical nodes.

### 2.2.3 Mesh data structure

Employing a notation that will be always adopted in the sequel of the book and in the MATLAB codes, the nodal coordinates are stored in the list of structures `nodes`. In particular the nodal coordinates are saved in the vector array `coor` of dimension  $D$  and `nodes(i).coor(j)` yields  $x_j^{(i)}$ .

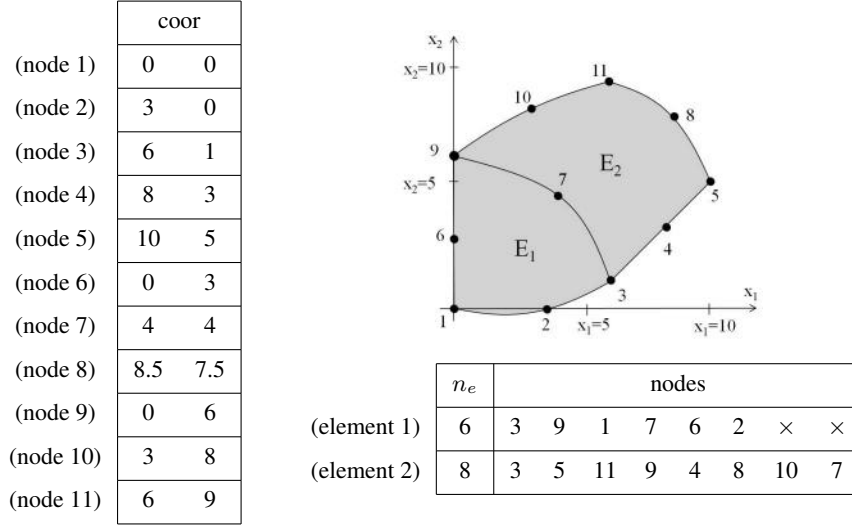
Data concerning elements are stored in the list of structures `elements`. The number of nodes per element  $n_e \rightarrow ne$  is contained in the field `elements(i).ne`; the list of nodes of the  $i$ -th element (connectivity of the element) is contained in the vector field `elements(i).nodes(1:n_e)` where the MATLAB-style notation  $(1:n_e)$  denotes the set of integers from 1 to  $n_e$ .

The difference between *local* and *global numbering* should be stressed at this stage. Every master element is associated with a fixed local ordering of the master nodes given in Tables 2.1-2.3 and ranging from 1 to  $n_e$ . On the contrary, the integer  $n$  such that  $n = \text{elements}(i).\text{nodes}(k)$  will be called the global number of the  $k$ -th master node of the  $i$ -th element.

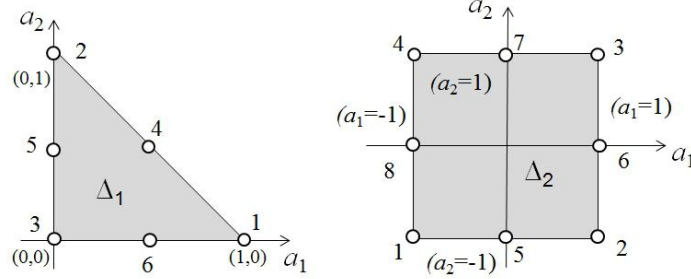
### 2.2.4 Example: conformity between a T6 and a Q8

Let us consider the mesh of the Figure 2.10, made of one T6 and one Q8. The shape functions are defined in Table 2.2 as well as the unit triangle  $\Delta_1$  and of the unit square

$\Delta_2$ , respectively (see also Figure 2.11). Let us examine in detail the representation of the common edge.



**Figure 2.10:** Mesh made of one T6 and one Q8; coordinates of the nodes and connectivity



**Figure 2.11:** Triangle T6 and quadrangle Q8: master elements  $\Delta_1$  and  $\Delta_2$  and conventions for local numbering.

In order to do so we need to analyse the restriction of shape functions to this line, knowing a priori that only  $N_1, N_2$  and  $N_4$  do not vanish identically there for the T6 and  $N_1, N_4$  and  $N_8$  for the Q8 (we recall that the numbering of shape functions follows the local ordering of nodes on the master element). The common edge corresponds to the line  $1 - a_1 - a_2 = 0$  on  $\Delta_1$  (triangle) and  $a_1 + 1 = 0$  on  $\Delta_2$  (quadrangle). Let us now introduce a common parametric coordinate defined on the edge and such that  $a \in [-1, 1]$ . We set  $(a_1, a_2) = ((1 - a)/2, (1 + a)/2)$  ( $\Delta_1$ , segment  $\{1, 4, 2\}$ ) and  $(a_1, a_2) = (-1, a)$  ( $\Delta_2$ , segment  $\{1, 8, 4\}$ ), as indicated in the picture. The restriction of the shape functions is:

$$\begin{aligned} \hat{N}_1(a) &= a(a-1)/2, & \hat{N}_4(a) &= 1-a^2, & \hat{N}_2(a) &= a(a+1)/2, & (\Delta_1); \\ \hat{N}_1(a) &= a(a-1)/2, & \hat{N}_8(a) &= 1-a^2, & \hat{N}_4(a) &= a(a+1)/2 & (\Delta_2). \end{aligned}$$



As anticipated by Property P2 of Section 2.2, the restrictions to the common edge of the shape functions coincide with the shape functions of the B3 element and, since the nodes on the edge are given the same coordinates in both elements, conformity is guaranteed.

### 2.2.5 Regularity of the parametric representation

In order to generate an admissible parametric representation of the geometry, other conditions must be met. Indeed the mapping (2.13) from the unit element  $\Delta$  to  $E$  must be onto and one-to-one, i.e. a bijection. To enforce this condition we introduce the *jacobian matrix*  $\mathbf{J}$  and the *jacobian*  $J$  of the mapping (2.13):

$$\mathbf{J}(\mathbf{a}) = \left[ \frac{\partial x_i}{\partial a_j} \right]_{1 \leq i, j \leq D}, \quad J(\mathbf{a}) = \text{Det} \mathbf{J}, \quad (2.17)$$

where the coefficients of  $\mathbf{J}$  are given by

$$J_{ij} = \frac{\partial x_i}{\partial a_j} = \sum_{k=1}^{n_e} \frac{\partial N_k}{\partial a_j}(\mathbf{a}) x_i^{(k)} \quad (1 \leq i, j \leq D)$$

and are continuous polynomial functions of  $\mathbf{a}$  (and so is  $J(\mathbf{a})$  as well). The mapping (2.13) is a bijection if  $J(\mathbf{a}) \neq 0 \forall \mathbf{a} \in \Delta$ . If we focus on a specific type of element for which the shape functions are given and fixed, the condition on the jacobian determinant translates into a condition limiting the choice of the nodal coordinates  $\mathbf{x}^{(k)}$ . Indeed, in many cases one can find suitable combinations of nodes for which  $J = 0$  for some  $\mathbf{a}$ , with the exception of the simplest elements (e.g. B2, T3, P4) which have a constant jacobian.

If the jacobian determinant vanishes somewhere in  $\Delta$ , the mapping (2.13) is “pathological” and the element  $E$  presents self-penetrations. Practically speaking, a vanishing  $J$  is a sign of deficiency of a mesh, often indicating exaggerated distortion due to insufficient refinement or other similar pathologies.

Finally it is worth recalling that the jacobian enters in the definition of the differential volume element as:

$$dV(\mathbf{x}) = J(\mathbf{a}) dV(\mathbf{a}). \quad (2.18)$$

#### Example: straight B3 element

As a simple example of interest for applications in fracture mechanics, let us consider a straight B3 element (see Table 2.1) aligned along the  $x$  axis and with nodal coordinates  $x^{(1)} = 0$ ,  $x^{(3)} = \alpha$ ,  $x^{(2)} = 1$ , with  $0 < \alpha < 1$ :

$$x = N_1(a) \times 0 + N_2(a) \times 1 + N_3(a) \times \alpha \quad (a \in \Delta = [-1, 1]) \quad (2.19)$$

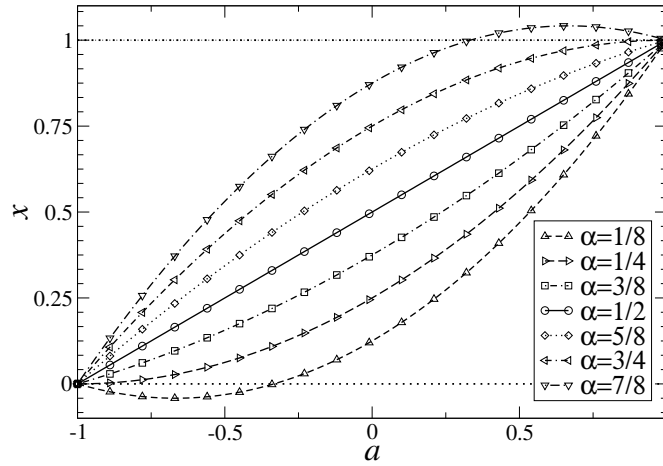
This means that, while  $x^{(1)}$  and  $x^{(2)}$  are kept fixed at the ends of the interval  $E = [-1, 1]$ , the “middle” node is free to move along the segment according to the value of  $\alpha$ . Indeed it is worth stressing that while the third master node is always placed in

$a = 0$ , i.e. the middle of  $\Delta$ , this does not imply an equivalent constraint on the physical node.

The jacobian of the representation (2.19) is here

$$J(a) = \frac{dx}{da} = (1 - 2\alpha)a + \frac{1}{2}. \quad (2.20)$$

It is straightforward to check that  $J(a)$  remains positive everywhere in  $\Delta = [-1, 1]$  only if  $1/4 \leq \alpha \leq 3/4$ . For all the other values of  $\alpha$ ,  $J(a)$  changes sign on  $\Delta$ , and the image of  $\Delta$  through the mapping (2.19) is hence larger than  $E = [-1, 1]$ . For instance,  $\alpha = 1/8$  gives  $x \in [-25/24, 1]$ , see Figure 2.12, and the portion  $x \in [-25/24, -1]$  outside  $E$  is swept twice.



**Figure 2.12:** Interpolation of three aligned nodes: physical coordinate  $x$  in terms of the parameter  $a$  (the jacobian  $J(a)$  is positive for the cases  $\alpha = 3/8, 1/2, 5/8$ ).

### 2.2.6 Local representation of displacements

After completing the representation of the problem geometry, we continue the parallel with Section 2.1.4 and decide to express the interpolation  $v_h$  of the displacement field  $v$  in terms of nodal displacements  $v^{(k)}$ .

$$v_h(x) = \sum_{k=1}^{n_e} N_k(a) v^{(k)} \quad \text{for every } x \in E_e \text{ defined by (2.13).} \quad (2.21)$$

where  $N_k(a)$  are the very same shape functions employed in (2.13). The term “isoparametric” hence refers to the fact that the same shape functions (also called interpolation functions) are employed in order to represent the geometry and the unknown field.

In view of the application of this concept in a code, it is convenient to place the set of nodal values associated with element  $E_e$  in a one-dimensional array

$\{V_e\}$  of length  $D \times n_e$  according to the convention

$$\{V_e\} = \{\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(n_e)}\}^T = \{v_1^{(1)}, v_2^{(1)}, \dots, v_D^{(n_e)}\}^T. \quad (2.22)$$

Introducing a second array  $\{v_h\}$  filled with the  $D$  components of the vector field  $\mathbf{v}_h$  on  $E$ , the interpolation formula (2.21) can be rewritten in matrix notation

$$\{v_h(\mathbf{x})\} = [N(\mathbf{a})]\{V_e\} \quad \text{for every } \mathbf{x} \in E_e \text{ defined by (2.13),} \quad (2.23)$$

where the coefficients of the  $D \times (D \times n_e)$  matrix  $[N(\mathbf{a})]$  are filled with the shape functions. If  $D = 3$

$$[N(\mathbf{a})] = \begin{bmatrix} N_1(\mathbf{a}) & 0 & 0 & \dots & N_{n_e}(\mathbf{a}) & 0 & 0 \\ 0 & N_1(\mathbf{a}) & 0 & \dots & 0 & N_{n_e}(\mathbf{a}) & 0 \\ 0 & 0 & N_1(\mathbf{a}) & \dots & 0 & 0 & N_{n_e}(\mathbf{a}) \end{bmatrix}$$

**Condition of unit sum of shape functions (partition of unity).** The representation formula (2.21) for the displacement field introduces a new constraint on the admissible forms of the shape functions  $N_k(\mathbf{a})$ . This originates from a necessary condition for convergence which can be intuitively formulated as follows. Let us consider a sequence of meshes created by successive refinements such that the maximum diameter  $h$  of the elements decreases. It is hence expected that, locally on one specific element, the displacement field differs from its Taylor first-order expansion by a small quantity. Strains and stresses in the exact solution become almost constant over one element. It is thus imperative that isoparametric elements be able to represent exactly every linear displacement field corresponding either to constant non-vanishing strains or to rigid body motions. We now consider the most general affine displacement field  $\mathbf{v}(\mathbf{x})$ :

$$\mathbf{v}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b},$$

where  $\mathbf{A}$  is a constant  $D \times D$  matrix and  $\mathbf{b}$  is a constant  $D$ -vector. Let us now evaluate the exact field at the nodes  $\mathbf{x}^{(k)}$  of an element  $E$

$$\mathbf{v}^{(k)}(\mathbf{x}) = \mathbf{A}\mathbf{x}^{(k)} + \mathbf{b}. \quad (2.24)$$

and then create an interpolated field  $\mathbf{v}_h(\mathbf{x})$  according to (2.21)

$$\begin{aligned} \mathbf{v}_h(\mathbf{x}) &= \sum_{k=1}^{n_e} N_k(\mathbf{a}) \{\mathbf{A}\mathbf{x}^{(k)} + \mathbf{b}\} = \mathbf{A} \left\{ \sum_{k=1}^{n_e} N_k(\mathbf{a}) \mathbf{x}^{(k)} \right\} + \left\{ \sum_{k=1}^{n_e} N_k(\mathbf{a}) \right\} \mathbf{b} \\ &= \mathbf{A}\mathbf{x} + \left\{ \sum_{k=1}^{n_e} N_k(\mathbf{a}) \right\} \mathbf{b}. \end{aligned}$$

As a consequence, in order to guarantee that  $\mathbf{v}_h(\mathbf{x})$  exactly coincides with the original  $\mathbf{v}$ , i.e.

$$\mathbf{v}_h(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b},$$

one has to enforce the condition:

$$\sum_{k=1}^{n_e} N_k(\mathbf{a}) = 1 \quad \text{for every } \mathbf{a} \in \Delta. \quad (2.25)$$

According to property P3 of Section 2.2 all the shape functions discussed herein satisfy this condition and are hence admissible.

Any isoparametric element is thus capable of representing exactly every displacement field with constant gradient. In particular for an infinitely refined mesh ( $h \rightarrow 0$ ), the strains  $\varepsilon[\mathbf{u}_h]$  associated with the interpolation  $\mathbf{u}_h$  of the exact displacement field  $\mathbf{u}$ , approximate correctly the true strains  $\varepsilon[\mathbf{u}]$ .

These remarks are often operatively implemented into the so called *patch test* (Zienkiewicz and Taylor, 2000a) which is employed to test the correct implementation in a code of a new element. An example will be discussed in Section 5.8.3.

### 2.2.7 Global representation of displacements

Starting from the representations (2.13) and (2.21) of the position vector and of the displacements defined locally element by element, one can easily express a “global” interpolation over the whole  $\Omega_h$  in the form

$$\mathbf{v}_h(\mathbf{x}) = \sum_{n=1}^{N_N} \tilde{N}_n(\mathbf{x}) \mathbf{v}^{(n)} \quad (\mathbf{x} \in \Omega_h). \quad (2.26)$$

If we denote by  $\Omega^{(n)}$  the union of the elements containing node  $\mathbf{x}^{(n)}$ , the global shape functions  $\tilde{N}_n(\mathbf{x})$  in (2.26) are such that:

- (a) The *support* of  $\tilde{N}_n(\mathbf{x})$  is  $\Omega^{(n)}$ ; in other words,  $\tilde{N}_n(\mathbf{x}) = 0$  for  $\mathbf{x} \notin \Omega^{(n)}$ .
- (b) If  $E_e \subset \Omega^{(n)}$ , then  $\tilde{N}_n(\mathbf{x}) = N_k(\mathbf{a})$  with  $n = \text{elements}(e) \cdot \text{nodes}(k)$  and  $\mathbf{x}, \mathbf{a}$  connected via (2.13).

As a consequence of (2.25), also the global shape functions satisfy the property

$$\sum_{n=1}^{N_N} \tilde{N}_n(\mathbf{x}) = 1 \quad (\mathbf{x} \in \Omega_h).$$

The representation (2.26) is expressed in terms of a sum over all the nodes and does not make any distinction between unknown or prescribed nodal values.

### 2.2.8 Application: prescribed displacements and numbering of unknowns

In order to make a distinction between those components  $v_j^{(n)}$  of nodal displacement which are prescribed by boundary conditions and those which are unknown (*free*), we add a new field dof (degree of freedom) to each item of the list nodes. The length of dof depends on the specific application (e.g. D in the case of linear elasticity). A number is attributed to each component of dof according to the convention:

$$\begin{aligned} \text{nodes}(n).\text{dof}(j) &> 0 & u_j^{(n)} \text{ free,} \\ \text{nodes}(n).\text{dof}(j) &= -1 & u_j^{(n)} \text{ prescribed.} \end{aligned} \quad (2.27)$$

The role of dof is also to introduce a numbering for the free nodal values. We will say that  $i = \text{nodes}(n).\text{dof}(j) > 0$  is the global number of the unknown nodal value, in the sense that  $v_j^{(n)}$  will occupy the  $i$ -th position in the list  $\{\mathbb{V}\}$  of all the unknowns:

$$\{\mathbb{V}\} = \{v_j^{(n)} \mid \text{nodes}(n).\text{dof}(j) > 0, (1 \leq n \leq N_N, 1 \leq j \leq D)\}. \quad (2.28)$$

Also,  $N$  will denote the overall number of unknown nodal values, i.e. the length of  $\{\mathbb{V}\}$ . Since the numbering is assumed to be sequential,  $N = \max_{n,j} \text{dof}(n,j)$ .

As an example, let us reconsider the mesh of Figure 2.10, in plane-strain conditions, and assume that the displacement is prescribed in both directions on the edge with nodes 5, 8, 11. The field dof having length  $D = 2$ , one possible numbering is (for convenience of pagination the two components of dof are presented in separate lines)

$$\text{nodes}(:).\text{dof}(1) = \{1, 3, 5, 7, 0, 9, 11, 0, 13, 15, 0\} \quad (2.29)$$

$$\text{nodes}(:).\text{dof}(2) = \{2, 4, 6, 8, 0, 10, 12, 0, 14, 16, 0\} \quad (2.30)$$

and the total number of unknowns is  $N = 16$ . The array  $\{\mathbb{V}\}$  associated with an interpolated displacement field  $\mathbf{u}_h$  becomes

$$\{\mathbb{V}\} = \{v_1^{(1)}, v_2^{(1)}, \dots, v_1^{(4)}, v_2^{(4)}, v_1^{(6)}, v_2^{(6)}, v_1^{(7)}, v_2^{(7)}, v_1^{(9)}, v_2^{(9)}, v_1^{(10)}, v_2^{(10)}\}^T$$

The degrees of freedom associated with nodes 5, 8 and 11 do not appear in the global vector of unknowns  $\{\mathbb{V}\}$ , since they correspond to prescribed displacements.

As a consequence, the representation (2.26) can be put in the form

$$\mathbf{v}_h(\mathbf{x}) = \mathbf{u}_h^{(D)}(\mathbf{x}) + \mathbf{v}_h^{(0)}(\mathbf{x}) \quad (\mathbf{x} \in \Omega_h) \quad (2.31)$$

with

$$\begin{aligned} \mathbf{u}_h^{(D)}(\mathbf{x}) &= \sum_{(n,j) \mid \text{nodes}(n).\text{dof}(j) \leq 0} \tilde{N}_n(\mathbf{x}) u_j^D(\mathbf{x}^{(n)}) \mathbf{e}_j \quad \in \mathcal{C}_h(\mathbf{u}^D), \\ \mathbf{v}_h^{(0)} &= \sum_{(n,j) \mid \text{nodes}(n).\text{dof}(j) > 0} \tilde{N}_n(\mathbf{x}) v_j^{(n)} \mathbf{e}_j \quad \in \mathcal{C}_h(\mathbf{0}), \end{aligned} \quad (2.32)$$

where  $\mathcal{C}_h(\mathbf{u}^D)$  and  $\mathcal{C}_h(\mathbf{0})$  denote the spaces of kinematically admissible fields (with the data  $\mathbf{u}^D$  and with  $\mathbf{0}$  respectively) *in the sense of the finite element discretization*.

The representation (2.31)–(2.32) is a particular form of the Galerkin representation (1.37), with

$$\varphi^K(\mathbf{x}) = \tilde{N}_n(\mathbf{x}) \mathbf{e}_j \text{ and } \alpha_K = u_j^{(n)} \quad \text{with } K = \text{nodes}(n).\text{dof}(j). \quad (2.33)$$

### 2.2.9 Gradient and strain tensor

In view of the application of isoparametric elements to problems of solid mechanics, we discuss herein how to compute the gradient or the strain tensor starting from the displacement field (2.21) on  $E$ . By definition of a gradient, formula (2.21) yields:

$$d\mathbf{v}_h = \nabla \mathbf{v}_h \cdot d\mathbf{x}. \quad (2.34)$$

Expressing the two differentials  $d\mathbf{x}$  and  $d\mathbf{v}_h$  in terms of the parametric coordinates  $\mathbf{a}$  starting from the interpolations (2.13) and (2.21), one gets:

$$d\mathbf{x} = \mathbf{J} \cdot d\mathbf{a}, \quad d\mathbf{v}_h = \mathbf{J}[\mathbf{v}] \cdot d\mathbf{a},$$

where  $\mathbf{J}[\mathbf{v}]$  is the jacobian matrix associated with  $\mathbf{v}_h$ , defined by

$$J[\mathbf{v}]_{ij} = \frac{\partial v_{h,i}}{\partial a_j} = \sum_{k=1}^{n_e} \frac{\partial N_k}{\partial a_j}(\mathbf{a}) v_i^{(k)}. \quad (2.35)$$

Replacing  $d\mathbf{x}$  and  $d\mathbf{v}_h$  in (2.34) and imposing the identity for any choice of  $d\mathbf{a}$ , we get the expression of the gradient of  $\mathbf{v}_h$  in terms of the parametric coordinates:

$$\nabla \mathbf{v}_h(\mathbf{x}) = \mathbf{J}[\mathbf{v}](\mathbf{a}) \cdot \mathbf{J}^{-1}(\mathbf{a}), \quad \mathbf{x} \text{ and } \mathbf{a} \text{ connected by (2.13)}. \quad (2.36)$$

Due to the linearity of (2.35) with respect to the nodal components  $v_i^{(k)}$ , for any kind of isoparametric element, a gradient matrix  $[G_V]$  can be defined such that, employing the notation of (2.23), the expression (2.36) can be recast into

$$\{\nabla \mathbf{v}_h(\mathbf{x})\} = [G_V(\mathbf{a})]\{V_e\}, \quad \text{for every } \mathbf{x} \in E_e. \quad (2.37)$$

where  $\{\nabla \mathbf{v}_h\}$  denotes an array filled with the components of the gradient. In the following sections some examples will be discussed in detail. Moreover, the linearized strain  $\varepsilon[\mathbf{v}_h]$  can then easily be expressed in a form similar to (2.37), see Section 3.1.4.

### Application: quadratic triangle T6

Let us analyse the T6 triangular element of Table 2.2 assuming that the nodal coordinates are stored in the  $6 \times 2$  matrix  $[X]$ , such that  $x_i^{(k)} = [X]_{ki}$ . The components  $[J]_{ij}$  of the jacobian matrix  $[J]$  are

$$[J]_{ij} = \frac{\partial x_i}{\partial a_j} \quad \text{with} \quad \frac{\partial x_i}{\partial a_j} = \sum_{k=1}^6 \frac{\partial N_k}{\partial a_j} x_i^{(k)} = \sum_{k=1}^6 [X]_{ki} [D]_{kj}$$

where  $[D]_{kj} = \partial N_k / \partial a_j$  are the derivatives of the shape functions with respect to the parameters. Recalling that  $a_3 = 1 - a_1 - a_2$ , one has

$$[D] = \begin{bmatrix} 4a_1 - 1 & 0 & -4a_3 + 1 & 4a_2 & -4a_2 & 4(a_3 - a_1) \\ 0 & 4a_2 - 1 & -4a_3 + 1 & 4a_1 & 4(a_3 - a_2) & -4a_1 \end{bmatrix}^T$$

Finally, the jacobian matrix  $[J]$  reads  $[J] = [X]^T [D]$  and its determinant is easily computable. In MATLAB form these formulas become

```
D=[4*a1-1 0 -4*a3+1 4*a2 -4*a2 4*(a3-a1);
  0 4*a2-1 -4*a3+1 4*a1 4*(a3-a2) -4*a1]';
J=X'*D;
detJ=J(1,1)*J(2,2)-J(1,2)*J(2,1);
```

A similar procedure holds also for the gradient of  $v_h$

$$\frac{\partial v_{h,i}}{\partial x_j} = \sum_{k=1}^6 \frac{\partial N_k}{\partial x_j} v_i^{(k)} = \frac{\partial N_k}{\partial a_m} \frac{\partial a_m}{\partial x_j} v_i^{(k)} = [D]_{km} [J]_{mj}^{-1} v_i^{(k)}$$

which is expressed in terms of the inverse of the jacobian matrix  $[J]^{-1}$ . If we introduce the matrix  $[G]$  the coefficients  $[G]_{kj}$  of which are the derivatives of the shape functions  $N_k$  with respect to  $x_j$ , that is:

$$[G] = [D] [J]^{-1}$$

one has:

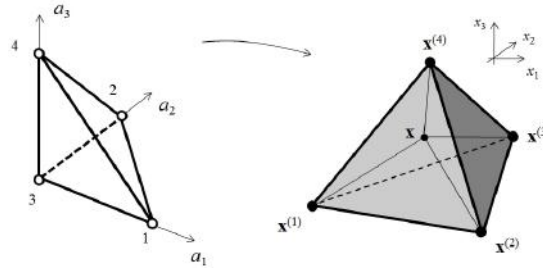
$$\frac{\partial v_{h,i}}{\partial x_j} = \sum_{k=1}^6 [G]_{kj} v_i^{(k)}$$

The numerical computation of  $[G]$  can be easily performed in the MATLAB environment as

```
invJ=1/detJ*[J(2,2) -J(1,2); -J(2,1) J(1,1)];
G=D*invJ;
```

## 2.3 Applications and exercises

### 2.3.1 Linear and quadratic tetrahedra P4 and P10



**Figure 2.13:** P4 tetrahedron: master element (left) and physical element (right)

Let us consider the tetrahedral element  $E$  of volume  $V$  and nodes  $\mathbf{x}^{(i)}$  depicted in Figure 2.13, right. Any point  $\mathbf{x}$  within the element induces a partition of the element into four sub-elements of tetrahedral shape of volume  $V_i$  (by convention we assume that the  $i$ -th sub-tetrahedron is the one which does not have  $\mathbf{x}^{(i)}$  as a vertex). Similarly to what done in Section 2.1.4 with area coordinates, we now introduce four *volume* parameters:

$$a_1 = V_1/V, \quad a_2 = V_2/V, \quad a_3 = V_3/V, \quad a_4 = V_4/V$$

subjected to the constraint  $a_1 + a_2 + a_3 + a_4 = 1$ . Moreover, since  $a_1, a_2, a_3$  are subjected to the bounds

$$0 \leq a_1 \leq 1, \quad 0 \leq a_2 \leq 1 - a_1, \quad 0 \leq a_3 \leq 1 - a_1 - a_2,$$

the master element in the parameter space is the unit tetrahedron on the left of Figure 2.13. The four parameters  $a_1, a_2, a_3, a_4$  can be employed as shape functions for an isoparametric tetrahedral element:  $N_1 = a_1, N_2 = a_2, N_3 = a_3, N_4 = a_4$ . Indeed they satisfy the three properties of Section 2.2: (i)  $N_i(\mathbf{a}^{(j)}) = \delta_{ij}$ ; (ii)  $N_i$  vanishes on edges and faces not containing the  $j$ -th node; (iii) the shape functions sum up to unity (i.e. satisfy the partition of unity property) and the element is thus linear complete. Hence the geometry representation is:

$$\mathbf{x} = a_1 \mathbf{x}^{(1)} + a_2 \mathbf{x}^{(2)} + a_3 \mathbf{x}^{(3)} + a_4 \mathbf{x}^{(4)} \quad (2.38)$$

and any field will be interpolated using the same functions. Let us now compute the gradient matrix  $[G]$ , which is constant in  $E$ . To this aim we first need to evaluate the gradient of the volume coordinates

$$\frac{\partial N_k}{\partial x_i} = \frac{\partial a_k}{\partial x_i}$$

The terms  $\partial a_k / \partial x_i$  are the coefficients of the inverse of the Jacobian matrix which is constant and could be computed as done for the T3 triangle. However, we will here follow a different procedure. Let  $S_i$  denote the face (of area  $S_i$ ) opposite to node  $\mathbf{x}^{(i)}$  and  $\mathbf{n}_i$  be the unit vector orthogonal to  $S_i$  and pointing towards  $\mathbf{x}^{(i)}$ . The gradient  $\nabla a_i$  is necessarily a constant vector, in view of (2.38). Since  $a_i$  is constant on any plane orthogonal to  $\mathbf{n}_i$ ,  $\nabla a_i$  is parallel  $\mathbf{n}_i$ ; moreover  $a_i = 0$  on  $S_i$  and  $a_i = 1$  on  $\mathbf{x}^{(i)}$ . Hence  $\nabla a_i = \mathbf{n}_i / h_i$  where  $h_i$  is the distance of  $\mathbf{x}^{(i)}$  from  $S_i$ . But  $3V = S_i h_i$ , and the wedge product between two sides of  $S_i$  (taken in the proper ordering) gives  $2S_i \mathbf{n}_i$ . Finally:

$$\begin{aligned} \nabla a_1 &= \frac{1}{6V} (\mathbf{x}^{(32)} \wedge \mathbf{x}^{(42)}) & \nabla a_2 &= \frac{1}{6V} (\mathbf{x}^{(13)} \wedge \mathbf{x}^{(43)}) \\ \nabla a_3 &= \frac{1}{6V} (\mathbf{x}^{(21)} \wedge \mathbf{x}^{(41)}) & \nabla a_4 &= \frac{1}{6V} (\mathbf{x}^{(31)} \wedge \mathbf{x}^{(21)}) \end{aligned}$$

As an exercise, we present the MATLAB instructions for the computation of the gradient matrix  $[G]$  (for the definition see Section 2.2.9). The nodal coordinates are accessible via the  $4 \times 3$  matrix  $X$  such that  $x_i^{(n)} = X(n, i)$ .

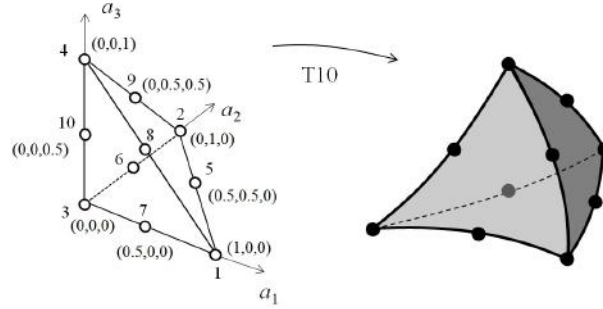
```
n1=cross(X(4,:)-X(2,:),X(3,:)-X(2,:)); % vector 2S_1 n_1
n2=cross(X(4,:)-X(3,:),X(1,:)-X(3,:)); % vector 2S_2 n_2
n3=cross(X(4,:)-X(1,:),X(2,:)-X(1,:)); % vector 2S_3 n_3
n4=cross(X(2,:)-X(1,:),X(3,:)-X(1,:)); % vector 2S_4 n_4
vol6=(X(4,:)-X(1,:))*n4; % 6*V
G=[n1 n2 n3 n4]'/vol6; % gradient matrix
```

Assume now that we want to enrich our element by creating a tetrahedral element for which the interpolation in the parametric space is a complete quadratic polynomial. Since a complete quadratic polynomial has ten terms, the element should have ten nodes, as illustrated in Figure 2.14.

The shape functions are:

$$\begin{aligned} N_1 &= a_1(2a_1 - 1), & N_2 &= a_2(2a_2 - 1), \\ N_3 &= a_3(2a_3 - 1), & N_4 &= a_4(2a_4 - 1), \\ N_5 &= 4a_1a_2, & N_6 &= 4a_2a_3, & N_7 &= 4a_3a_1, \\ N_8 &= 4a_1a_4, & N_9 &= 4a_2a_4, & N_{10} &= 4a_3a_4 \end{aligned}$$





**Figure 2.14:** P10 tetrahedron: master element (left) and physical element (right)

The first four shape functions are such that  $N_i = 0$  for  $a_i = 0$  and  $a_i = 1/2$ . The other shape functions are associated with nodes at the middle of a given edge and vanish on the triangular faces that do not share that edge. It is left as an exercise to verify that the gradient matrix can be computed as follows, in terms of the parametric coordinates

```

D=[4*a1-1,0,0;
   0,4*a2-1,0;
   0,0,4*a3-1;
   -4*a4+1,-4*a4+1,-4*a4+1;
   4*a2,4*a1,0;
   0,4*a3,4*a2;
   4*a3,0,4*a1;
   4*a4-4*a1,-4*a1,-4*a1;
   -4*a2,4*a4-4*a2,-4*a2;
   -4*a3,-4*a3,4*a4-4*a3];
J=X'*D;
detJ=det(J);
invJ=inv(J);
G=D*invJ;

```

% der of shape functions  
 % w.r.t. a\_1,a\_2,a\_3  
  
 % jacobian matrix  
 % jacobian  
  
 % gradient of shape functions

### 2.3.2 Transition element

We present how an isoparametric transition element can be designed for connecting standard Q4 and Q8 quadrilateral elements (see Figure 2.15, where  $E_2$  is the sought new element collecting quadrilaterals  $E_1$  and  $E_3$ ). The transition element  $E_2$  must have three nodes on the side which will be mapped on the interface with the Q8 element and two nodes on the side which will be mapped on the interface with the Q4 element (see Figure 2.15 for their coordinates in the parametric space). No additional nodes are necessary and hence the master element has globally 5 nodes and will be denoted by Q5.

For nodes 2 and 3 the shape functions of the Q4 satisfy the desired property  $N_i(\mathbf{a}^{(j)}) = \delta_{ij}$  and we take:

$$N_2^{Q5}(a_1, a_2) = N_2^{Q4}(a_1, a_2) = \frac{1}{4}(1 + a_1)(1 - a_2)$$

$$N_3^{Q5}(a_1, a_2) = N_3^{Q4}(a_1, a_2) = \frac{1}{4}(1 + a_1)(1 + a_2)$$

On the contrary, for node 5 this property is satisfied by the Q8 shape function:

$$N_5^{Q5}(a_1, a_2) = N_8^{Q8}(a_1, a_2) = \frac{1}{2}(1 - a_1)(1 - a_2^2)$$

For nodes 1 and 4 the situation is slightly more complicated, since neither Q4 nor Q8 shape functions are satisfactory (Q4 shape functions do not vanish on node 5 and Q8 shape functions do not respect the unit sum property). However:

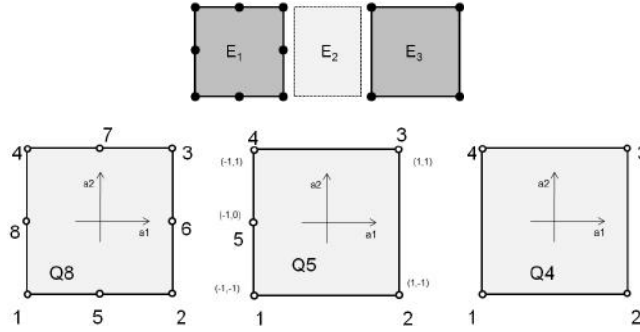
$$N_1^{Q5}(a_1, a_2) = N_1^{Q4}(a_1, a_2) - \frac{1}{2}N_8^{Q8}(a_1, a_2)$$

$$N_4^{Q5}(a_1, a_2) = N_4^{Q4}(a_1, a_2) - \frac{1}{2}N_8^{Q8}(a_1, a_2)$$

do guarantee  $N_i(\mathbf{a}^{(j)}) = \delta_{ij}$ . Moreover:

$$\sum_{i=1}^5 N_i^{Q5}(a_1, a_2) = \sum_{i=4}^4 N_i^{Q4}(a_1, a_2) = 1$$

and the element is an admissible isoparametric element.



**Figure 2.15:** Transition between a Q8 element and a Q4 element

### 2.3.3 Pressurized spherical shell and B3 elements

A simple FE code has been developed in Section 1.5.6 for the analysis of a pressurized spherical shell employing B2 elements.

**Exercise 2.1.** Develop an extension of the code to quadratic B3 elements and compare with the results of `sphere_B2_1S_solid.m`. Analyse the different convergence rates towards the exact solution.

## The finite element method for linear problems

---

In this chapter, we present the displacement-based finite element method for constructing approximate solutions to equilibrium problems of linear elasticity. We start from the variational formulation stemming from the minimisation of total potential energy functional, and more specifically from its Galerkin version (Chapter 1), and exploit the notion of isoparametric elements (Chapter 2). The formulation of the approximate problem is treated in Section 3.1, while the numerical evaluation of the integrals yielding the element matrices, and in particular the numerical quadrature techniques adopted, are the topic of Section 3.2. The assemblage of the global matrix of the linear system is presented next in Section 3.3, and the solution of the resulting linear system by means of direct and iterative techniques is discussed in Section 3.4. The introduction of these basic notions is complemented by some results concerning the convergence of the approximate solution to the exact solution (Section 3.5). The essential notions introduced in this chapter for the displacement-based formulation of linear elasticity are implemented in the MATLAB code `genlin.m`, described in Section 3.6. They are finally complemented by various applications and examples (Section 3.7), some of them demonstrating extensions to scalar problems like thermal equilibrium, and will be generalized to more involved contexts in the sequel of the book.

### 3.1 The finite element method based on isoparametric elements

#### 3.1.1 Procedure based on a weak formulation

The weak formulation stemming from the principle of virtual power is the starting point for building the approximate solution methodology of a broad class of problems in solid mechanics. In the case of linear elasticity of interest in this chapter, the construction of the approximate problem starts from the weak formulation (1.22) and essentially consists of the following steps:

- (i) build a discretized domain  $\Omega_h = \cup_e E_e$  using a conformal mesh (Section 2.2.2) and introduce in (1.22) the approximation of the geometry (2.13) for every element  $E_e$ :

$$\mathbf{x} = \sum_{k=1}^{n_e} N_k(\mathbf{a}) \mathbf{x}^{(k)} \quad (\mathbf{x} \in E_e, 1 \leq e \leq N_E);$$

- (ii) seek the unknown displacement field  $\mathbf{u}$  in the form (2.31)–(2.32), that is (adopting the notations of Section 2.2.7)

$$\begin{aligned} \mathbf{u}_h(\mathbf{x}) &= \mathbf{u}_h^{(D)}(\mathbf{x}) + \mathbf{u}_h^{(0)}(\mathbf{x}) \in \mathcal{C}_h(\mathbf{u}^D) \quad (\mathbf{x} \in \Omega_h), \\ \text{with } \mathbf{u}_h^{(D)}(\mathbf{x}) &= \sum_{(n,j) \mid \text{nodes}(n).\text{dof}(j) \leq 0} \tilde{N}_n(\mathbf{x}) u_j^D(\mathbf{x}^{(n)}) \mathbf{e}_j \in \mathcal{C}_h(\mathbf{u}^D); \\ \text{and } \mathbf{u}^{(0)} &= \sum_{(n,j) \mid \text{nodes}(n).\text{dof}(j) > 0} \tilde{N}_n(\mathbf{x}) u_j^{(n)} \mathbf{e}_j \in \mathcal{C}_h(\mathbf{0}); \end{aligned} \quad (3.1)$$

- (iii) use *kinematically admissible* virtual fields  $\mathbf{w}$  of the form (2.31)–(2.32), i.e.

$$\mathbf{w}(\mathbf{x}) = \sum_{(n,j) \mid \text{nodes}(n).\text{dof}(j) > 0} \tilde{N}_n(\mathbf{x}) w_j^{(n)} \mathbf{e}_j \in \mathcal{C}_h(\mathbf{0}) \quad (\mathbf{x} \in \Omega_h). \quad (3.2)$$

The discretized form of the weak formulation (1.22) can then be written as

$$\begin{aligned} \text{Find } \mathbf{u}_h^{(0)} \in \mathcal{C}_h(\mathbf{0}) \text{ such that: } \forall \mathbf{w} \in \mathcal{C}_h(\mathbf{0}), \\ \int_{\Omega_h} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(0)}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = - \int_{\Omega_h} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(D)}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV \\ + \int_{\Omega_h} \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_{T,h}} \mathbf{T}^D \cdot \mathbf{w} \, dS. \end{aligned} \quad (3.3)$$

Let  $N$  be the number of degrees of freedom which remain unknown after the enforcement of displacement boundary conditions. Collecting the nodal values  $\mathbf{u}_j^{(n)}$  and  $\mathbf{w}_j^{(n)}$  such that  $\text{nodes}(n).\text{dof}(j) > 0$  in the one-dimensional arrays  $\{\mathbb{U}\}$  and  $\{\mathbb{W}\}$  of length  $N$ , according to the convention put forward in Section 2.2.7, the discretized weak form (3.3) becomes

$$\begin{aligned} \text{find } \{\mathbb{U}\} \in \mathbb{R}^N \text{ such that: } \forall \{\mathbb{W}\} \in \mathbb{R}^N, \\ \{\mathbb{W}\}^T [\mathbb{K}] \{\mathbb{U}\} = \{\mathbb{W}\}^T (\{\mathbb{F}^u\} + \{\mathbb{F}^{\text{ext}}\}) = \{\mathbb{W}\}^T \{\mathbb{F}\}, \end{aligned} \quad (3.4)$$

where  $[\mathbb{K}]$  is the stiffness matrix of the structure and  $\{\mathbb{F}\}$  the vector of generalized forces (nodal forces) expressed as the sum of contributions  $\{\mathbb{F}^u\}$  due to prescribed boundary displacements and  $\{\mathbb{F}^{\text{ext}}\}$  due to external volume and surface forces. The weak form (3.4) is equivalent to the linear system of unknown  $\{\mathbb{U}\} \in \mathbb{R}^N$ :

$$[\mathbb{K}] \{\mathbb{U}\} = \{\mathbb{F}\}. \quad (3.5)$$

### 3.1.2 Procedure starting from the potential energy functional

In the case of linear elasticity we recall that  $\mathbf{u}$  can be sought by minimising the potential energy  $\mathcal{P}(\mathbf{v})$ ,  $\mathbf{v} \in \mathcal{C}(\mathbf{u}^D)$  and that the weak formulation (1.22) expresses the stationarity condition for the minimization of  $\mathcal{P}(\mathbf{v})$ . Limiting ourselves to the fields  $\mathbf{v}$  of  $\mathcal{C}_h(\mathbf{u}^D)$ , i.e. of the form

$$\mathbf{v}_h(\mathbf{x}) = \mathbf{u}_h^{(D)}(\mathbf{x}) + \mathbf{v}_h^{(0)}(\mathbf{x})$$

with  $\mathbf{v}_h^{(0)} \in \mathcal{C}_h(\mathbf{0})$  defined according to (3.2) and associated with the one-dimensional array  $\{\mathbb{V}\}$  of length  $N$  collecting the nodal values, the total potential energy reads

$$\begin{aligned} \mathcal{P}(\mathbf{v}_h) &= \frac{1}{2} \int_{\Omega_h} \boldsymbol{\varepsilon}[\mathbf{v}_h] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{v}_h] dV - \int_{\Omega_h} \mathbf{f} \cdot \mathbf{v}_h dV - \int_{S_{T,h}} \mathbf{T}^D \cdot \mathbf{v}_h dS \\ &= \frac{1}{2} \int_{\Omega_h} \boldsymbol{\varepsilon}[\mathbf{v}_h^{(0)}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{v}_h^{(0)}] dV + \int_{E_e} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(D)}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{v}_h^{(0)}] dV \\ &\quad - \int_{\Omega_h} \mathbf{f} \cdot \mathbf{v}_h^{(0)} dV - \int_{S_{T,h}} \mathbf{T}^D \cdot \mathbf{v}_h^{(0)} dS + \mathcal{P}(\mathbf{u}_h^{(D)}), \end{aligned}$$

or, using a matrix notation

$$\mathcal{P}(\mathbf{v}_h) = \frac{1}{2} \{\mathbb{V}\}^T [\mathbb{K}] \{\mathbb{V}\} - \{\mathbb{V}\}^T \{\mathbb{F}\} + \mathcal{P}(\mathbf{u}_h^{(D)}). \quad (3.6)$$

The minimisation of  $\mathcal{P}(\mathbf{v})$  for  $\mathbf{v} \in \mathcal{C}_h(\mathbf{0})$  hence reduces to the discrete minimisation problem

$$\{\mathbb{U}\} = \arg \min_{\{\mathbb{V}\} \in \mathbb{R}^N} \left( \frac{1}{2} \{\mathbb{V}\}^T [\mathbb{K}] \{\mathbb{V}\} - \{\mathbb{V}\}^T \{\mathbb{F}\} + \mathcal{P}(\mathbf{u}_h^{(D)}) \right),$$

for which the stationarity condition is nothing but the linear system (3.5).

### 3.1.3 Operative steps. Assemblage procedure

In practical terms, the computation of the stiffness matrix  $[\mathbb{K}]$  and the nodal force vector  $\{\mathbb{F}\}$  does not proceed by explicit substitution of the definitions (3.1) and (3.2) in (3.3), since this would inconveniently necessitate explicit employment of the global shape functions. Instead, an *assemblage* procedure is preferred, resting on the element-by-element computation of all required integrals. This approach exploits the fact that the integrals appearing in the equations (3.3) and (3.4) defining  $[\mathbb{K}]$  and  $\{\mathbb{F}\}$  are performed over  $\Omega_h$  and its boundary, and are hence *additive with respect to the finite elements*.

Indeed, using the partition  $\Omega_h = \cup_{e=1}^{N_E} E_e$  in (3.3) and (3.4) yields:

$$\{\mathbb{W}\}^T [\mathbb{K}] \{\mathbb{U}\} = \sum_{e=1}^{N_E} \int_{E_e} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(0)}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] dV, \quad (3.7a)$$

$$\{\mathbb{W}\}^T \{\mathbb{F}^u\} = \sum_{e=1}^{N_E} \left\{ - \int_{E_e} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(D)}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] dV \right\} \quad (3.7b)$$

$$\{\mathbb{W}\}^T \{\mathbb{F}^{\text{ext}}\} = \sum_{e=1}^{N_E} \left\{ \int_{E_e} \mathbf{f} \cdot \mathbf{w} dV + \int_{\Gamma_T^e} \mathbf{T}^D \cdot \mathbf{w} dS \right\} \quad (3.7c)$$

where  $\mathbf{w} \in \mathcal{C}_h(\mathbf{0})$  is a virtual field associated with the mesh and  $\Gamma_T^e = \partial E_e \cap S_T$  denotes the portion of the boundary of element  $E_e$  situated on  $S_T$  (which is actually empty for a large proportion of the elements).

The expressions (3.7a)-(3.7c) suggest that the practical computation of the matrix  $[\mathbb{K}]$  and of the right hand side  $\{\mathbb{F}\}$  can be performed by

- (i) evaluating the integrals over each element (element integrals) and each portion of the boundary,
- (ii) using these contributions to fill the coefficients of the global matrix and RHS vector.

These two operations are the basic steps for the construction of a finite element model and will be described in detail in Sections 3.2 (element matrices) and 3.3 (assemblage of the global model).

It is actually more expedient to compute the element contributions to (3.7a) and (3.7b) together

$$\begin{aligned} \{\mathbb{W}\}^T [\mathbb{K}] \{\mathbb{U}\} - \{\mathbb{W}\}^T \{\mathbb{F}^u\} &= \sum_{e=1}^{N_E} \int_{E_e} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(0)} + \mathbf{u}_h^{(D)}] : \mathcal{A} : \boldsymbol{\varepsilon}[\mathbf{w}] dV \\ &= \sum_{e=1}^{N_E} \int_{E_e} \boldsymbol{\varepsilon}[\mathbf{u}_h] : \mathcal{A} : \boldsymbol{\varepsilon}[\mathbf{w}] dV = -\{\mathbb{W}\}^T \{\mathbb{F}^{\text{int}}\} \quad (3.8) \end{aligned}$$

and to extract a posteriori the contributions to  $[\mathbb{K}]$  and  $\{\mathbb{F}\}$ . Indeed, the contribution of the elements to the stiffness matrix in (3.7a) and to the generalized forces associated with the prescribed displacements in (3.7b) both come from the same integral yielding the virtual power of internal forces for the element, according to the interpretation (1.48) of equation (3.5) in terms of equilibrium of internal and external generalized forces.

### 3.1.4 “Engineering notation”

The transposition of these concepts into a computer code is made easier by resorting to lists (column table with one index) and matrices (table with two indices). In particular the components of the stress and strain tensor will be arranged into two arrays with six entries each:

$$\begin{aligned} \{\sigma\} &= \{\sigma_{11} \ \sigma_{22} \ \sigma_{33} \ \sigma_{12} \ \sigma_{13} \ \sigma_{23}\}^T, \\ \{\varepsilon\} &= \{\varepsilon_{11} \ \varepsilon_{22} \ \varepsilon_{33} \ 2\varepsilon_{12} \ 2\varepsilon_{13} \ 2\varepsilon_{23}\}^T. \end{aligned} \quad (3.9)$$

Similarly, the linear elastic constitutive law (1.1c) can be recast in the matrix form

$$\{\sigma\} = [A] \{\varepsilon\}, \quad (3.10)$$

where  $[A]$  is a  $6 \times 6$  symmetric matrix. In the case of 3D isotropic linear elasticity,  $[A]$  is given by:

$$[A] = \begin{bmatrix} \lambda + 2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda + 2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda + 2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix} \quad (3.11)$$

As it will be demonstrated later on examples, this procedure can also be applied to (2.37) in order to express the relationship between strain and nodal displacements on an isoparametric element as

$$\{\varepsilon[\mathbf{u}_h](\mathbf{x})\} = [B(\mathbf{a})]\{U_e\}, \quad (3.12)$$

where the array  $\{U_e\}$  collects the element nodal displacements according to the convention (2.22) and  $[B(\mathbf{a})]$  is a  $6 \times (n_e \times D)$  matrix whose precise formulation involves the gradient matrix  $[G_V(\mathbf{a})]$  defined by (2.37) and therefore depends on the type of element used. The virtual work of internal stresses in (3.8) can be expressed as:

$$\begin{aligned} \varepsilon[\mathbf{u}_h] : \mathcal{A} : \varepsilon[\mathbf{w}] &= \varepsilon[\mathbf{w}] : \mathcal{A} : \varepsilon[\mathbf{u}_h] = \{\varepsilon[\mathbf{w}]\}^T [A] \{\varepsilon[\mathbf{u}_h]\} \\ &= \{W_e\}^T [B(\mathbf{a})]^T [A] [B(\mathbf{a})] \{U_e\}. \end{aligned} \quad (3.13)$$

The introduction of the factor 2 in (3.9) for  $\{\varepsilon\}$  actually allows to express (3.10) and (3.13) in terms of the same matrix  $[A]$ . All the engineering representations discussed herein clearly have to be adapted to the context. For instance, in the case of plane strain conditions, the strain and stress arrays will have three entries and  $[A]$  will be a  $3 \times 3$  matrix. These notational conventions, known as engineering, or Voigt, notation for tensors, are specifically adopted in all the MATLAB codes developed for this book.

### 3.2 Element integrals

In this section we will detail the techniques allowing to evaluate the contributions of each element (henceforth referred to as element contributions) to the expressions (3.7c) and (3.8).

#### 3.2.1 Element stiffness matrix

The contribution of a generic element  $E_e$  to the virtual work (using the virtual field  $\mathbf{w}$ ) of the internal stresses associated with the field  $\mathbf{u}_h$  is, according to (3.8),

$$\int_{E_e} \varepsilon[\mathbf{u}_h] : \mathcal{A} : \varepsilon[\mathbf{w}] dV. \quad (3.14)$$

The isoparametric formulation adopted provides a suitable context for the evaluation of (3.14), since it naturally introduces a correspondence between  $E_e$  and the reference element  $\Delta_e$ , through the change of variables  $\mathbf{x} \in E_e \rightarrow \mathbf{a} \in \Delta_e$  defined by (2.13). Employing the engineering notation of Section 3.1.4, the virtual work of internal stresses is the integral of (3.13) while the differential volume element transforms according to (2.18). The element integral (3.14) can be hence rewritten introducing the *element stiffness matrix*  $[K_e]$ :

$$\int_{E_e} \varepsilon[\mathbf{u}_h] : \mathcal{A} : \varepsilon[\mathbf{w}] dV = \{W_e\}^T [K_e] \{U_e\} \quad (3.15)$$

$$\text{with} \quad [K_e] = \int_{\Delta_e} [B(\mathbf{a})]^T [A] [B(\mathbf{a})] J(\mathbf{a}) dV(\mathbf{a}) \quad (3.16)$$

It should be stressed that, in (3.16), the Jacobian  $J$  and the strain matrix  $[B]$ , depend on the specific element considered.

The element stiffness matrix stemming from this computation will contribute, through assemblage, to both the global stiffness matrix and the global nodal force vector associated with the prescribed boundary displacements.

### 3.2.2 Element stiffness matrix for a T3 element

Let us now consider the triangular element  $E_e$  described in Section 2.1.4, which corresponds to the isoparametric linear T3 of Table 2.2. As discussed in Section 2.1.4, the nodes should be labeled using their global numbers  $\{e_1, e_2, \dots, e_{n_e}\}$  expressing the connectivity of the element as defined in table nodes. However, for the sake of simplicity, we will employ the local numbering  $\{1, 2, 3, \dots, n_e\}$  in all the sections devoted to element contributions. For any T3 element, the lists of nodal values in (3.15) have six entries each, e.g.

$$\{U_e\} = \left\{ u_1^{(1)} \ u_2^{(1)} \ u_1^{(2)} \ u_2^{(2)} \ u_1^{(3)} \ u_2^{(3)} \right\}^T$$

for the case of  $\{U_e\}$ . The displacement field on the element is expressed in terms of the shape functions  $N_i(\mathbf{a}) = a_i$  and of the vector of nodal values in the matrix form (2.22), i.e.

$$\{u_h(\mathbf{x})\} = \begin{bmatrix} a_1 & 0 & a_2 & 0 & a_3 & 0 \\ 0 & a_1 & 0 & a_2 & 0 & a_3 \end{bmatrix} \{U_e\}, \quad (3.17)$$

where the vector  $\{u_h\}$  collects the two components of  $\mathbf{u}_h$ . Introducing the list of strains  $\{\varepsilon\} = \{\varepsilon_{11} \ \varepsilon_{22} \ 2\varepsilon_{12}\}^T$  the  $[B]$  matrix in (3.12) is found to be given by:

$$[B] = \begin{bmatrix} \frac{\partial a_1}{\partial x_1} & 0 & \frac{\partial a_2}{\partial x_1} & 0 & \frac{\partial a_3}{\partial x_1} & 0 \\ 0 & \frac{\partial a_1}{\partial x_2} & 0 & \frac{\partial a_2}{\partial x_2} & 0 & \frac{\partial a_3}{\partial x_2} \\ \frac{\partial a_1}{\partial x_2} & \frac{\partial a_1}{\partial x_1} & \frac{\partial a_2}{\partial x_2} & \frac{\partial a_2}{\partial x_1} & \frac{\partial a_3}{\partial x_2} & \frac{\partial a_3}{\partial x_1} \end{bmatrix}$$

or, using (2.8):

$$[B] = \frac{1}{2A} \begin{bmatrix} x_2^{(23)} & 0 & x_2^{(31)} & 0 & x_2^{(12)} & 0 \\ 0 & x_1^{(32)} & 0 & x_1^{(13)} & 0 & x_1^{(21)} \\ x_1^{(32)} & x_2^{(23)} & x_1^{(13)} & x_2^{(31)} & x_1^{(21)} & x_2^{(12)} \end{bmatrix} \quad (3.18)$$

If, for instance, plane-strain conditions are assumed, the isotropic linear elastic constitutive law becomes:

$$\{\sigma\} = \{\sigma_{11} \ \sigma_{22} \ \sigma_{12}\}^T = [A] \{\varepsilon\}$$

with  $[A] = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & (1-2\nu)/2 \end{bmatrix}$



Since strains and stresses are constant over the element the integral (3.15) can be easily performed analytically yielding

$$[K_e] = S [B]^T [A] [B]$$

where  $S$  is the element area.

These developments are implemented in MATLAB form by means of the function T3\_2A\_solid\_Ke computing  $[K_e] \rightarrow K_e$  given the  $3 \times 2$  input matrix  $X$  such that  $x_j^{(i)} = X(i, j)$ , containing the nodal coordinates, and the two-element list  $mate$  holding the Young modulus  $E$  and the Poisson coefficient  $\nu$  for the element:

```
function Ke=T3_2A_solid_Ke(X,mate)

E=mate(1);
nu=mate(2);
A=E/((1+nu)*(1-2*nu))*[1-nu,nu,0;           % plane strain stiff. matrix
                        nu,1-nu,0;
                        0,0,(1-2*nu)/2];
x11=X(1,1); x21=X(2,1); x31=X(3,1);         % nodal coordinates
x12=X(1,2); x22=X(2,2); x32=X(3,2);
S=.5*((x21-x11)*(x32-x12)-...               % element area
      (x31-x11)*(x22-x12));
B=[x22-x32,0,x32-x12,0,x12-x22,0;          % B matrix
   0,x31-x21,0,x11-x31,0,x21-x11;
   x31-x21,x22-x32,x11-x31, ...
   x32-x12,x21-x11,x12-x22]/(2*S);
Ke=S*B'*A*B;                               % element stiffness
```

### 3.2.3 Nodal forces associated with external loadings

The generalized nodal forces associated with the external loadings exerted on the element  $E_e$  originate from the element contributions

$$\int_{E_e} \mathbf{f} \cdot \mathbf{w} \, dV + \int_{\Gamma_T^e} \mathbf{T}^D \cdot \mathbf{w} \, dS = \{W_e\}^T \{F_e^{\text{vol}} + F_e^{\text{surf}}\} = \{W_e\}^T \{F_e^{\text{ext}}\}$$

coming from (3.7c). It is worth stressing again that these “nodal forces” are not in general true forces, but generalized forces that are work-conjugate (i.e. associated through duality) to the corresponding nodal displacements. For instance, the product  $\{W_e\}^T \{F_e^{\text{vol}}\}$  yields the virtual work of the imposed body forces in the virtual field interpolating the nodal values  $\{W_e\}$ .

**Body forces.** The restriction of the virtual field  $\mathbf{w}$  to the element  $E_e$  having the form (2.23), the integral over  $E_e$  can be expressed on the unit reference element  $\Delta_e$ :

$$\int_{E_e} \mathbf{f} \cdot \mathbf{w} \, dV = \{W_e\}^T \left\{ \int_{\Delta_e} [N(\mathbf{a})]^T \{f(\mathbf{x}(\mathbf{a}))\} J(\mathbf{a}) \, dV(\mathbf{a}) \right\} = \{W_e\}^T \{F_e^{\text{vol}}\}$$

where the vector  $\{f\}$  holds the components of  $\mathbf{f}$ .

**Surface forces.** The surface  $\Gamma_T^e$  is a boundary feature of an element, i.e. a face of a 3D element or an edge of a 2D element. However, as a consequence of the remarks of Section 2.2, the restriction of any isoparametric element to its boundary is another isoparametric element and can be treated as such with no need a priori to make reference to the parent element. This is the approach adopted in this book, especially as regards the MATLAB implementation, as detailed in the next section. As a consequence, the geometry of  $\Gamma_T^e$  is parametrized like that of any standard element

$$\mathbf{x} = \sum_{k=1}^{N_e} N_k(\mathbf{a}) \mathbf{x}^{(k)} \quad (\mathbf{a} \in \Delta_e),$$

Focusing now on the more involved case where  $\Gamma_T^e$  is a surface element, we first introduce the two covariant vectors

$$\mathbf{g}_1(\mathbf{a}) = \sum_{k=1}^{N_e} \frac{\partial N_k}{\partial a_1}(\mathbf{a}) \mathbf{x}^{(k)} \quad \mathbf{g}_2(\mathbf{a}) = \sum_{k=1}^{N_e} \frac{\partial N_k}{\partial a_2}(\mathbf{a}) \mathbf{x}^{(k)},$$

which define the *natural basis* for the tangent plane to the surface at  $\mathbf{x}(\mathbf{a})$ . The vector product of  $\mathbf{g}_1$  and  $\mathbf{g}_2$  is oriented along the unit normal  $\mathbf{n}$  to  $\Gamma_T^e$ ; moreover:

$$dS(\mathbf{a}) = \|\mathbf{g}_1(\mathbf{a}) \wedge \mathbf{g}_2(\mathbf{a})\| da_1 da_2 = J(\mathbf{a}) da_1 da_2, \quad (3.19)$$

The surface integral finally becomes

$$\int_{\Gamma_T^e} \mathbf{T}^D \cdot \mathbf{w} dS = \{W_e\}^T \left\{ \int_{\Delta_e} [N(\mathbf{a})]^T \{T^D(\mathbf{x}(\mathbf{a}))\} J(\mathbf{a}) da_1 da_2 \right\} = \{W_e\} \{F_e^{\text{surf}}\}$$

with the Jacobian  $J$  defined by (3.19). It should be remarked that, unlike for the case of a volume integral, the Jacobian  $J(\mathbf{a})$  is not a polynomial function of  $\mathbf{a}$  because of the square root associated with the Euclidean norm  $\|\cdot\|$  in (3.19).

In the specific case where the prescribed tractions have the form of a pressure ( $\mathbf{T}^D = -p\mathbf{n}$ ), the element nodal forces can be computed as:

$$\{F_e^{\text{surf}}\} = \left\{ \int_{\Delta_e} -p(\mathbf{x}(\mathbf{a})) [N(\mathbf{a})]^T \mathbf{g}_1(\mathbf{a}) \wedge \mathbf{g}_2(\mathbf{a}) da_1 da_2 \right\},$$

The analysis of the simpler case where  $\Gamma_T^e$  is a line element is treated through examples in the next section.

### 3.2.4 Nodal forces for a T3 element

**Distributed volume forces.** We assume that  $\mathbf{f}$  (for instance the self weight) is constant over the element. Following 3.17 we write

$$\int_{E_e} \mathbf{f} \cdot \mathbf{w}_h dV = \{W_e\}^T \left( \int_{E_e} [N]^T dV \right) \{f\} = \{W_e\}^T \frac{S}{3} \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}^T \{f\}$$

where the terms multiplying  $\{W_e\}$  are identified as  $\{F_e\}$ .

**Nodal forces due to surface tractions.** We now consider one edge of a T3 element subjected to given constant tractions. As anticipated, the restriction of the T3 element to an edge is a standard B2 line element. In order to simplify the procedure, we define a fictitious list  $\{T^D\}$  with two entries, one of which simply being set to zero in cases where the loading acts along a single specified direction.

The vector of virtual nodal values contains now four entries ordered according to (2.22), i.e.:

$$\{W_e\} = \left\{ w_1^{(1)} \ w_2^{(1)} \ w_1^{(2)} \ w_2^{(2)} \right\}^T$$

and the  $[N]$  matrix in (2.23) is

$$[N(a)] = \begin{bmatrix} N_1(a) & 0 & N_2(a) & 0 \\ 0 & N_1(a) & 0 & N_2(a) \end{bmatrix}$$

where the two shape functions  $N_1, N_2$  are defined in Table 2.1. If  $H$  denotes the length of the element, we have

$$\begin{aligned} \int_{\Gamma_e} \mathbf{T}^D \cdot \mathbf{w} \, dS &= \left\{ \int_0^H \mathbf{w} \, ds \right\} \cdot \mathbf{T}^D = \{W_e\}^T \frac{H}{2} \left\{ \int_{-1}^1 [N(a)]^T \, da \right\} \{T^D\} \\ &= \{W_e\}^T \frac{H}{2} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}^T \{T^D\} \end{aligned}$$

This expression is now recast in MATLAB form, assuming that the nodal coordinates  $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$  of the B2 element are stored in the matrix  $X$  such that (as usual)  $x_j^{(i)} = X(i, j)$ . The loading is defined by the intensity  $val$  and the traction direction  $idir$ . Conventionally, if  $idir=0$ , the load acts along the direction normal to the boundary.

```
function Fe=B2_2A_solid_Fe_surf(X,val,idir)

x11=X(1,1); x12=X(1,2);           % first node
x21=X(2,1); x22=X(2,2);           % second node
H=sqrt((x21-x11)^2+(x22-x12)^2);   % element length
NL=H/2*[1 0 1 0;                  %
        0 1 0 1];
TD=zeros(2,1);                    % traction vector
if idir>0,                         % if Cartesian direction
    TD(idir)=val;
else                               % if along normal
    n=[(x22-x12); -(x21-x11)]/H;  % unit normal vector
    TD=val*n;
end
Fe=NL'*TD;                        % nodal forces due to tractions
```

**Exercise 3.1** (Element stiffness and force vector for the P4 element). *Consider a linear tetrahedral element P4 (Sec. 2.3.1). Develop the MATLAB functions for computing the element stiffness matrix and the nodal forces due to surface tractions*

### 3.2.5 Numerical integration of element quantities

The element stiffness matrix and the vector of equivalent nodal forces introduced in Sections 3.2.1-3.2.3 can be computed in closed form only in some simple cases. Indeed, the integral appearing in (3.16) cannot be evaluated analytically in general because  $[B(\mathbf{a})]$  involves the inverse Jacobian matrix, see (3.12) and (2.36). As a consequence one has to resort to approximate techniques of numerical integration which are usually based on the concept of *Gauss points*. Any integration formula (also called *quadrature rule*) of this type employs  $G$  Gauss points  $\mathbf{a}_g$  and weights  $w_g$  to approximate the generic integral defined on the reference element, according to

$$\int_{\Delta} f(\mathbf{a}) dV(\mathbf{a}) \approx \sum_{g=1}^G w_g f(\mathbf{a}_g). \quad (3.20)$$

Several formulas of this type are available, depending on the different choices of  $\Delta$ , as discussed next. It is hence quite clear that one further advantage of working with isoparametric elements is that they are naturally mapped onto reference unit elements, for which quadrature rules are available. For instance, in the case of the element stiffness matrix, formula (3.20) leads to an approximation of (3.16) as

$$[K_e] \approx \sum_{g=1}^G w_g [B(\mathbf{a}_g)]^T [A] [B(\mathbf{a}_g)] J(\mathbf{a}_g). \quad (3.21)$$

**Line integrals.** The integral over the reference segment is approximated using the well known *Gauss-Legendre* quadrature rules with  $G$  points and weights:

$$\int_{-1}^1 f(a) da \approx \sum_{g=1}^G w_g f(a_g), \quad (3.22)$$

The quadrature formulas (3.22) are very accurate even for low  $G$  if they are applied to regular  $f$ , which is typically the case of element contributions. Specifically, it can be stated that:

- The Gaussian quadrature formulas (3.22) with  $G$  points is exact if  $f(a)$  is any polynomial of order  $\leq 2G - 1$ .

The abscissas  $a_g$  and weights  $w_g$  in (3.22) for any  $G$  shape certain properties:

- (i) All the abscissas are interior to the segment:  $-1 < a_g < 1$ ;
- (ii) The points and weights are distributed symmetrically over the segment: if  $a_g$  is a point of weight  $w_g$ , then  $-a_g$  is also a point with the same weight  $w_g$ ;
- (iii) The weights are strictly positive and their sum is equal to 2, clearly a necessary requirement in order to integrate correctly the constant  $f = 1$ .

For instance, the quadrature formula for  $G = 2$  is defined by

$$\int_{-1}^1 f(a) da \approx f(-1/\sqrt{3}) + f(1/\sqrt{3}), \quad (3.23)$$

and it is easy to verify directly that this formula integrates exactly every polynomial of degree  $\leq 3$ . Other possible choices are listed in Appendix A.

**Integrals over squares and cubes.** It is then straightforward to define numerical integration rules on a D-cube. Since the unit cube of dimension D is actually the cartesian product of segments, it is possible to build “product formulas” in any dimension starting from Gauss-Legendre one-dimensional quadrature rules. For instance, an integral over the unit square  $C^2 = \{-1 \leq a_1, a_2 \leq 1\}$  can be evaluated as

$$\int_{C^2} f(a_1, a_2) da_1 da_2 \approx \sum_{g_1=1}^G \sum_{g_2=1}^G w_{g_1} w_{g_2} f(a_{g_1}, a_{g_2}) \quad (3.24)$$

or an integral over the unit cube  $C^3 = \{-1 \leq a_1, a_2, a_3 \leq 1\}$ ,

$$\int_{C^3} f(a_1, a_2, a_3) da_1 da_2 da_3 \approx \sum_{g_1=1}^G \sum_{g_2=1}^G \sum_{g_3=1}^G w_{g_1} w_{g_2} w_{g_3} f(a_{g_1}, a_{g_2}, a_{g_3}) \quad (3.25)$$

where the numbers  $a_{g_i}$  and weights  $w_{g_i}$  are the same as in (3.22).

**Integrals over triangles and tetrahedra.** Since triangles for  $D = 2$ , tetrahedra for  $D = 3$  and, more generally, simplexes in any space dimension cannot be expressed as a cartesian product of segments, the numerical evaluation of integrals over these domains requires ad-hoc formulas. For instance

$$\int_{T^2} f(\mathbf{a}) da_1 da_2 \approx \sum_{g=1}^G w_g f(\mathbf{a}_g) \quad (3.26)$$

where  $T^2 = \{a_1 \geq 0, 0 \leq a_2 \leq 1 - a_1\}$  is the reference triangle and the points  $\mathbf{a}_g$  and weights  $w_g$  differs from those employed for 1D integrals. These quadrature formulas are often called *Gauss-Hammer* rules. As an example of the numerous options available (Lyness and Jespersen, 1975), we cite the rule with three points

$$\int_{T^2} f(\mathbf{a}) da_1 da_2 \approx \frac{1}{6} \left[ f\left(\frac{1}{6}, \frac{1}{6}\right) + f\left(\frac{1}{6}, \frac{2}{3}\right) + f\left(\frac{2}{3}, \frac{1}{6}\right) \right] \quad (3.27)$$

which integrates exactly over  $T^2$  any polynomial in  $(a_1, a_2)$  of total degree 2.

**Choice of the quadrature rule: complete (full) integration.** It is rather natural to employ such numerical quadrature techniques for the evaluation of element integrals. In the interest of computational speed, one should employ the lowest order quadrature rule guaranteeing sufficient accuracy. Consider for instance the element stiffness matrix  $[K_e]$ . In general, the functions being integrated do not reduce to polynomials due to the presence of the inverse Jacobian  $J^{-1}$ , and hence there is no simple criterion for choosing the size of the quadrature rule. However, following a practical guideline often adopted in the literature on finite elements, the quadrature is simply required to be *complete*. A quadrature rule is said to be complete if it integrates exactly the element quantity of interest when  $J$  is a constant, with the function being integrated then reducing to a polynomial.

**Choice of the quadrature rule: reduced integration and spurious modes.**

Expression (3.21) can be recast, using block-matrix notation, in the form  $[K_e] \approx [B_T]^T [A_T] [B_T]$ , with

$$[A_T] = \begin{bmatrix} w_1 J(\mathbf{a}_1) [A] & & 0 \\ & \ddots & \\ 0 & & w_G J(\mathbf{a}_G) [A] \end{bmatrix},$$

$$[B_T] = [ [B(\mathbf{a}_1)]^T \dots [B(\mathbf{a}_G)]^T ]^T.$$

Let  $n_c$  (resp.  $n_r$ ) denote the number of columns (resp. rows) of the matrix  $[B(\mathbf{a})]$ . Since  $[A_T]$  is invertible,  $[K_e]$  and  $[B_T]$  have same rank, so that

$$\text{rank}([K_e]) = \text{rank}([B_T]) \leq \min(n_c, G n_r)$$

On the other hand, the definition (3.15) of  $[K_e]$  implies that the kernel  $\text{Ker}([K_e])$  of  $[K_e]$  is the subspace of all infinitesimal rigid body displacement (1.2), and therefore has dimension  $n_{rig} = 3$  for 2D analyses or  $n_{rig} = 6$  for 3D analyses. This implies the requirement  $\text{rank}([K_e]) = n_c - n_{rig}$ , which can hold only if

$$n_c - n_{rig} \leq G n_r. \quad (3.28)$$

Otherwise,  $\text{Ker}([K_e])$  necessarily contains modes (often known as spurious modes) other than the rigid body modes. Condition (3.28) is necessary, but not sufficient (i.e. spurious modes may still exist when it is met). For a Q4 element,  $n_c = 8$ ,  $n_r = 3$  and  $n_{rig} = 3$ , and condition (3.28) becomes  $5 \leq 3G$ , i.e. at least 2 Gauss points are required (using 1 Gauss point gives rise to two spurious modes, see Chapter 6). Similarly, for a T6 element,  $n_c = 12$ ,  $n_r = 3$  and  $n_{rig} = 3$ , leading to  $9 \leq 3G$ , i.e. a 3-point Gauss rule is needed.

**3.2.6 Stiffness matrix for a T6 element**

As an application of the numerical integration techniques, we now discuss the implementation of the computation of element stiffness matrices for T6 elements, elaborating on the developments of Section 2.2.9. We start by providing the expression of matrix  $[B]$  in terms of the gradient matrix  $[G]$

$$B = \begin{bmatrix} G(1,1) & 0 & G(2,1) & 0 & G(3,1) & 0 & G(4,1) & 0 & G(5,1) & 0 & G(6,1) & 0; \\ 0 & G(1,2) & 0 & G(2,2) & 0 & G(3,2) & 0 & G(4,2) & 0 & G(5,2) & 0 & G(6,2); \\ G(1,2) & G(1,1) & G(2,2) & G(2,1) & G(3,2) & G(3,1) & \dots & \dots & \dots & \dots & \dots & \dots; \\ G(4,2) & G(4,1) & G(5,2) & G(5,1) & G(6,2) & G(6,1) \end{bmatrix};$$

which actually depends on the parametric coordinates. In order to evaluate integral (3.16) we remark that the product  $[B(\mathbf{a})]^T [A] [B(\mathbf{a})]$  is a second-degree polynomial in  $a_1, a_2$  if the Jacobian is constant. Hence a three point Gauss-Hammer quadrature rule is *complete* in the sense defined above and is here adopted for the quadrature.

Like for the T3 element addressed in Section 3.2.2, the input variables are the matrix  $X$  with the nodal coordinates and the list *mate* with the material properties.

First Gauss abscissae and weights are stored in `a_gauss` and `w_gauss`, respectively (with Gauss point coordinates given in terms of the 3 area coordinates  $(a_1, a_2, 1 - a_2 - a_2)$ ). Then the stiffness matrix `Ke` is initialized to zero and a loop is launched over the three Gauss points. At each Gauss point of abscissa `a=a_gauss(g,:)` the `B` matrix is evaluated together with the jacobian `detJ` and finally the contribution to the stiffness matrix is added `Ke=Ke+B'*A*B*detJ*w_gauss(g)` according to formula (3.26).

```
function Ke=T6_2A_solid_Ke(X,mate)

E=mate(1);
nu=mate(2);
A=E/((1+nu)*(1-2*nu))*[1-nu,nu,0;
                        nu,1-nu,0;
                        0,0,(1-2*nu)/2];

a_gauss=1/6*[4 1 1; 1 4 1; 1 1 4]; % Gauss abscissae
w_gauss=[1/6 1/6 1/6]; % Gauss weights
Ke=zeros(12,12);
for g=1:3, % loop over Gauss points
    a=a_gauss(g,:); % coordinates of gauss point
    D=[4*a(1)-1 0 -4*a(3)+1 4*a(2)... % derivative of shape functions...
        -4*a(2) 4*(a(3)-a(1)); % w.r.t. a_1,a_2
        0 4*a(2)-1 -4*a(3)+1 4*a(1) ...
        4*(a(3)-a(2)) -4*a(1)]';
    J=X'*D; % jacobian matrix
    detJ=J(1,1)*J(2,2)-J(1,2)*J(2,1); % jacobian
    invJ=1/detJ*[ J(2,2) -J(1,2); ... % inverse jacobian matrix
                 -J(2,1) J(1,1)];
    G=D*invJ; % gradient of shape functions
    B=[G(1,1) 0 G(2,1) 0 G(3,1) 0 ...
        G(4,1) 0 G(5,1) 0 G(6,1) 0;
        0 G(1,2) 0 G(2,2) 0 G(3,2)...
        0 G(4,2) 0 G(5,2) 0 G(6,2);
        G(1,2) G(1,1) G(2,2) G(2,1)...
        G(3,2) G(3,1) G(4,2) G(4,1)...
        G(5,2) G(5,1) G(6,2) G(6,1)];
    Ke=Ke+B'*A*B*detJ*w_gauss(g); % contribution to stiff matrix
end
```

### 3.2.7 Stiffness matrix for a Q4 element

As an example of application of product formulas, we now compute the stiffness matrix for a Q4 quadrangle. The shape functions are listed in Table 2.2. As above, we fix the order of quadrature for a complete integration by requiring that the stiffness matrix is computed exactly whenever the Jacobian is constant (which requires the physical element to be of rectangular shape). The derivatives of shape functions are linear and hence the product  $[B(\mathbf{a})]^T[A][B(\mathbf{a})]$  contains terms which are quadratic in  $a_1, a_2$ . As a consequence, the Cartesian product of the 1D two-point Gauss quadrature rule ensures completeness of the quadrature. The only difference with respect to the previous procedure for T6 elements is that the evaluation of the stiffness matrix, according to formula 3.24, requires a double loop (with counters  $g_1, g_2$ ), one over each reference segment of the tensor product employed to represent the unit reference square.

```

function Ke=Q4_2A_solid_Ke(X,mate)

E=mate(1);
nu=mate(2);
A=E/((1+nu)*(1-2*nu))*[1-nu,nu,0;
                        nu,1-nu,0;
                        0,0,(1-2*nu)/2];

a_gauss=1/sqrt(3)*[-1 1];           % Gauss abscissae
w_gauss=[1 1];                     % Gauss weights
Ke=zeros(8,8);
for g1=1:2,                          % first loop
    a1=a_gauss(g1);
    for g2=1:2,                      % second loop
        a2=a_gauss(g2);
        D=1/4*[-(1-a2) (1-a2) (1+a2) -(1+a2);
                -(1-a1) -(1+a1) (1+a1) (1-a1)]'; % w.r.t. a_1,a_2
        J=X'*D;                     % jacobian matrix
        detJ=J(1,1)*J(2,2)-J(1,2)*J(2,1); % jacobian
        invJ=1/detJ*[ J(2,2) -J(1,2); ... % inverse jacobian matrix
                      -J(2,1) J(1,1)];
        G=D*invJ;                   % gradient of shape functions
        B=[G(1,1) 0 G(2,1) 0 G(3,1) 0 ...
            G(4,1) 0 ;
            0 G(1,2) 0 G(2,2) 0 G(3,2)...
            0 G(4,2) ;
            G(1,2) G(1,1) G(2,2) G(2,1)...
            G(3,2) G(3,1) G(4,2) G(4,1)];
        Ke=Ke+B'*A*B*detJ*w_gauss(g1)*w_gauss(g2); % contrib. to stiffness matrix
    end
end

```

### 3.2.8 Nodal forces due to surface tractions for a B3 element

We consider here the situation where tractions are enforced on an edge of either a T6 or a Q8 element. In both cases the edge is actually a B3 isoparametric element, defined by the three nodes of coordinates  $\mathbf{x}^{(1)}$ ,  $\mathbf{x}^{(2)}$ ,  $\mathbf{x}^{(3)}$  and the shape functions listed in Table 2.1 and graphically represented in Figure 3.1. Input data are of the same type as those discussed in Section 3.2.4: the nodal coordinates of the B3 element collected in the  $3 \times 2$  matrix  $\mathbf{X}$ , the direction  $\text{idir}$  and the intensity  $\text{val}$  of the applied traction, assumed constant along the element. Then:

$$\int_{\Gamma_T^e} \mathbf{T}^D(\mathbf{x}) \cdot \mathbf{w}(\mathbf{x}) \, ds = \int_{-1}^1 \mathbf{T}^D(a) \cdot \mathbf{w}(a) J(a) \, da \simeq \sum_{g=1}^2 \mathbf{T}^D(a_g) \cdot \mathbf{w}(a_g) J(a_g) w_g$$

where  $J(a) = \|\mathbf{dx}/\mathbf{da}\|$  is the jacobian and a two-point Gauss quadrature rule has been adopted, with

```

a_gauss=[-1/sqrt(3) 1/sqrt(3)];
w_gauss=[1 1];

```

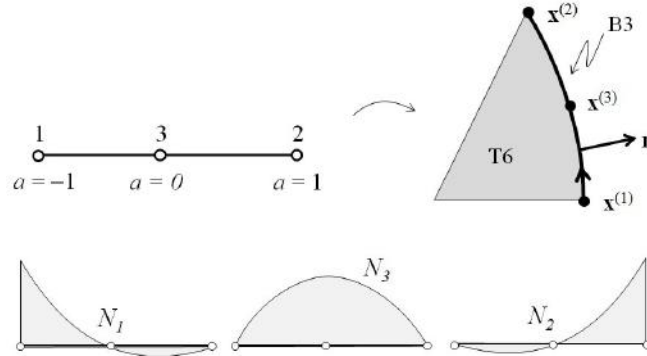
For any  $\mathbf{x}$  within the element one has:

$$\frac{\mathbf{dx}}{\mathbf{da}} = \sum_{k=1}^3 \frac{dN_k}{\mathbf{da}} \mathbf{x}^{(k)},$$



which directly yields the Jacobian:

```
D=[a-.5  a+.5 -2*a];
J=D*X;
detJ=sqrt(J(1)^2+J(2)^2);
```



**Figure 3.1:** Tractions enforced on an edge of a T6 element treated as a B3 line element

The product  $J=D*X$  contains the two components of the vector  $dx/(da)$  which is tangent to the line. In the MATLAB routine, apart from computing the Jacobian,  $J$  is employed when the traction is applied in a direction orthogonal to the surface in order to define the vector  $1/\det J * [J(2) \ -J(1)]$  holding the two components of the outward unit normal (with correct normal orientation requiring that the edge nodes be numbered as shown on Fig. 3.1). Letting the vector  $TD$  hold the two components of the load (see the conventions explained in Section 3.2.4), one has

$$w(a) \cdot T^D(a) J(a) = \{W_e\}^T [N(a)]^T \{T^D(a)\} J(a)$$

so that the list  $[N(a)]^T \{T^D(a)\} J(a)$  is computed as  $N' * TD * \det J$ , where  $N$  is the matrix of shape functions in (2.23). The complete MATLAB function is shown below.

```
function Fe=B3_2A_solid_Fe_surf(X,val,idir)

a_gauss=[-1/sqrt(3) 1/sqrt(3)];           % gauss-legendre two point rule
w_gauss=[1 1];
Fe=zeros(6,1);
for g=1:2                                  % loop over gauss points
    a=a_gauss(g);
    D=[a-.5  a+.5 -2*a];                  % derivative of shape functions
    J=D*X;
    detJ=sqrt(J(1)^2+J(2)^2);             % jacobian
    TD=zeros(2,1);
    if idir>0,                             % sets traction vector
        TD(idir)=val;
    else
        TD=val/detJ*[J(2) -J(1)]';      % traction along normal vector
    end
end
```

```

NL=[.5*a*(a-1) .5*a*(1+a) 1-a^2];          % shape functions
N=[NL(1) 0 NL(2) 0 NL(3) 0;
   0 NL(1) 0 NL(2) 0 NL(3)];
Fe=Fe+N'*TD*detJ*w_gauss(g);
end

```

### 3.2.9 Exercises

**Exercise 3.2** (stiffness matrix for the Q8 element). *Develop the MATLAB function for computing the element stiffness matrix of a Q8 element (Table 2.2).*

**Exercise 3.3** (nodal forces for the Q4 and Q8 elements). *Show that nodal forces due to surface tractions applied on the side of a Q8 element or of a Q4 element can be computed employing two of the functions developed in this chapter.*

**Exercise 3.4** (stiffness matrix and nodal forces for the P10 element). *Consider a quadratic tetrahedral element P10 (see Section 2.3.1). Develop the MATLAB functions for computing the element stiffness matrix and the nodal forces due to surface tractions.*

## 3.3 Assemblage

### 3.3.1 Stiffness matrix and nodal forces arising from given displacements

As discussed in Sections 3.1.3 and 3.2.1, the element stiffness matrix is associated with internal forces depending on either known or unknown displacement nodal values. A first task of the assemblage phase is to identify and separate the contributions to  $[\mathbb{K}]$  and  $\{\mathbb{F}^u\}$ .

Let us consider a generic finite element with  $n_e$  nodes and  $D = 2$  or  $3$ . If displacements are prescribed on a given portion of the boundary of  $E_e$ , we introduce a partition of the  $Dn_e$  degrees of freedom of the element so as to separate the free nodal values (the number of which is denoted  $N_e$ ) from those prescribed by the boundary conditions. Before proceeding we recall how the nodal values of the  $e$ -th element are ordered in the arrays  $\{U_e\}$  and  $\{W_e\}$  (see the conventions set in (2.22)): all the degrees of freedom of the first node in the element connectivity are placed first, followed by all the degrees of freedom of the second node, and so on. The *local numbering* of a nodal value is simply the position of that nodal value in these arrays. On the contrary the *global numbering* of an unknown nodal value  $u_j^{(n)}$  gives the position of  $u_j^{(n)}$  in the global vector of unknowns  $\{U\}$ . As described in Section 2.2.8, the *global numbering* is stored in  $\text{nodes}(n).\text{dof}(j)$ . Moreover we adopt the convention that, if  $u_j^{(n)}$  is prescribed by boundary conditions,  $\text{nodes}(n).\text{dof}(j)$  is set to  $-1$ . The set of all the global numberings in an element, in the sequence defined by the local ordering, is denoted by  $\mathcal{G}_e$ . In other words, if  $n_G$  is the global number of the  $n$ -th local node:

$$\mathcal{G}_e((n-1)D+1:nD) = \text{nodes}(n_G).\text{dof}(1:D)$$

We now create two lists  $\mathcal{L}_e^{(0)}$  and  $\mathcal{L}_e^{(D)}$  such that

$$\mathcal{L}_e^{(0)} = \{i : \mathcal{G}_e(i) > 0, 1 \leq i \leq Dn_e\}, \quad \text{of length } N_e,$$

contains the local numbering of the unknown nodal values, and

$$\mathcal{L}_e^{(D)} = \{i : \mathcal{G}_e(i) < 0, 1 \leq i \leq Dn_e\}, \quad \text{of length } \bar{N}_e = Dn_e - N_e,$$

contains the local numbering of nodal values prescribed by boundary conditions. It should be stressed that the lists  $\mathcal{L}_e^{(0)}$  (of positive integers) and  $\mathcal{L}_e^{(D)}$  (of negative integers) do not necessarily contain consecutive numbers. Accordingly we introduce the sublists of nodal values:

$$\{U_e^{(0)}\} = \{U_e(\mathcal{L}_e^{(0)})\}, \quad \{W_e^{(0)}\} = \{W_e(\mathcal{L}_e^{(0)})\} \quad \text{and} \quad \{U_e^{(D)}\} = \{U_e(\mathcal{L}_e^{(D)})\}$$

where the vectors  $\{U_e^{(0)}\}$  of the unknowns in the element and  $\{W_e^{(0)}\}$  of the nodal values of the virtual field both have length  $N_e$ , while the vector  $\{U_e^{(D)}\}$  of the prescribed nodal values has length  $\bar{N}_e$ . Similarly we define four submatrices

$$\begin{aligned} [K_e^{(00)}] &= [K_e(\mathcal{L}_e^{(0)}, \mathcal{L}_e^{(0)})], & [K_e^{(0D)}] &= [K_e(\mathcal{L}_e^{(0)}, \mathcal{L}_e^{(D)})], \\ [K_e^{(D0)}] &= [K_e(\mathcal{L}_e^{(D)}, \mathcal{L}_e^{(0)})], & [K_e^{(DD)}] &= [K_e(\mathcal{L}_e^{(D)}, \mathcal{L}_e^{(D)})]. \end{aligned}$$

of dimensions  $N_e \times N_e$ ,  $N_e \times \bar{N}_e$ ,  $\bar{N}_e \times N_e$  and  $\bar{N}_e \times \bar{N}_e$ , respectively. With these definitions:

$$\begin{aligned} \{W_e\}^T [K_e] \{U_e\} &= \begin{Bmatrix} W_e^{(0)} \\ 0 \end{Bmatrix}^T \begin{bmatrix} K_e^{(00)} & K_e^{(0D)} \\ K_e^{(D0)} & K_e^{(DD)} \end{bmatrix} \begin{Bmatrix} U_e^{(0)} \\ U_e^{(D)} \end{Bmatrix} \\ &= \{W_e^{(0)}\}^T \left( [K_e^{(00)}] \{U_e^{(0)}\} + [K_e^{(0D)}] \{U_e^{(D)}\} \right). \end{aligned} \quad (3.29)$$

In other terms, following (3.7a),(3.8), the element contribution to the global stiffness matrix reduces to the sub-matrix  $[K_e^{(00)}]$  of dimensions  $N_e \times N_e$ , while the contribution to the generalized nodal forces associated with prescribed displacements is associated with  $[K_e^{(0D)}] \{U_e^{(D)}\}$ .

Employing the notations introduced above, the case of an element without prescribed displacements corresponds to

$$N_e = Dn_e, \quad \bar{N}_e = 0, \quad [K_e^{(00)}] = [K_e], \quad \{F_e^u\} = \{0\}$$

The assemblage procedure of the global stiffness matrix and of the vector of nodal forces associated with the prescribed displacements can be formally expressed, with the notation just introduced, as

$$\{\mathbb{W}\}^T [\mathbb{K}] \{\mathbb{U}\} = \sum_{e=1}^{N_E} \{W_e^{(0)}\}^T [K_e^{(00)}] \{U_e^{(0)}\}, \quad (3.30a)$$

$$\{\mathbb{W}\}^T \{\mathbb{F}^u\} = - \sum_{e=1}^{N_E} \{W_e^{(0)}\}^T [K_e^{(0D)}] \{U_e^{(D)}\}, \quad (3.30b)$$

resulting from the combination of (3.7a), (3.7b), (3.15) and (3.29)

**Assemblage: practical procedure.** The relationships (3.30a)-(3.30b) cannot be applied directly. As a matter of fact, every element matrix  $[K_e]$ , once computed following the procedure described in the previous section, is defined according to the *local* numbering on the element, while the global stiffness matrix  $[\mathbb{K}]$  and the vector  $\{\mathbb{F}^u\}$  are defined in terms of the *global* numbering. The correct procedure employs the correspondence between local and global numberings of the unknowns in  $E_e$ , provided by the list  $\mathcal{G}_e$ .

Let us start by recasting the assemblage formula (3.30a) in a slightly different form employing the  $\mathcal{L}_e^{(0)}$  list and operating on the whole  $[K_e]$ :

$$\sum_{I,J=1}^N \{\mathbb{W}\}_I [\mathbb{K}]_{IJ} \{\mathbb{U}\}_J = \sum_{e=1}^{N_E} \left( \sum_{p,q \in \mathcal{L}_e^{(0)}} \{W_e\}_p [K_e]_{pq} \{U_e\}_q \right) \quad (3.31)$$

Next, let us focus on a specific coefficient  $[\mathbb{K}]_{IJ}$  of the global stiffness matrix. The coefficient  $[K_e]_{pq}$  in the  $e$ -th element will contribute to  $[\mathbb{K}]_{IJ}$  if and only if the nodal values multiplying  $[\mathbb{K}]_{IJ}$  and  $[K_e]_{pq}$  in (3.31) coincide, i.e.:

$$\{\mathbb{W}\}_I = \{W_e\}_p \text{ and } \{\mathbb{U}\}_J = \{U_e\}_q$$

which occurs when  $I = \mathcal{G}_e(p)$  and  $J = \mathcal{G}_e(q)$ . In practice one proceeds element by element following this scheme. Initially  $[\mathbb{K}]$  is set to zero and, for every element:

$$\text{for } p, q \in \mathcal{L}_e^{(0)} \quad [\mathbb{K}]_{\mathcal{G}_e(p), \mathcal{G}_e(q)} = [\mathbb{K}]_{\mathcal{G}_e(p), \mathcal{G}_e(q)} + [K_e]_{pq} \quad (3.32)$$

Similarly one proceeds for the right hand side with minor modifications

$$\text{for } p \in \mathcal{L}_e^{(0)}, q \in \mathcal{L}_e^{(D)} \quad \{\mathbb{F}^u\}_{\mathcal{G}_e(p)} = \{\mathbb{F}^u\}_{\mathcal{G}_e(p)} - [K_e]_{pq} \{U_e\}_q \quad (3.33)$$

### 3.3.2 Assemblage of the vector of nodal forces

This assemblage procedure is operated in analogy to  $[\mathbb{K}]$ , employing the relationship

$$\begin{aligned} \{\mathbb{W}\}^T \{\mathbb{F}^{\text{ext}}\} &= \{\mathbb{W}\}^T (\{\mathbb{F}^{\text{vol}}\} + \{\mathbb{F}^{\text{surf}}\}) \\ &= \sum_{e=1}^{N_E} \{W_e\}^T (\{F_e^{\text{vol}}\} + \{F_e^{\text{surf}}\}). \end{aligned} \quad (3.34)$$

Clearly, the sum is limited to those elements where the loading is actually exerted. The vector of nodal forces is initially set to 0; for every element concerned, the vector of element nodal forces  $\{F_e\}$  is computed as described in Section 3.2.3 and the assemblage is performed according to the formal scheme:

$$\text{for } p \in \mathcal{L}_e^{(0)} \quad \{\mathbb{F}\}_{\mathcal{G}_e(p)} = \{\mathbb{F}\}_{\mathcal{G}_e(p)} + \{F_e\}_p \quad (3.35)$$

Nodal forces can be defined for other types of loadings: initial strains (of thermal or plastic nature), pre-stresses, etc. For instance, the computation of nodal forces of thermal origin will be discussed in Section 4.3.3.

### 3.4 Numerical solution of the linear system

We address now the numerical solution of the system of linear equations (3.5) governing the equilibrium of an elastic solid:

$$[\mathbb{K}]\{\mathbf{U}\} = \{\mathbf{F}\}. \quad (3.36)$$

#### 3.4.1 Properties of the stiffness matrix

**The stiffness matrix is symmetric and positive definite.** The stiffness matrix  $[\mathbb{K}]$  is symmetric by construction. Moreover, if rigid body motions are constrained,  $[\mathbb{K}]$  is also positive definite (Section 1.1.3):

$$[\mathbb{K}] = [\mathbb{K}]^T, \quad \{\mathbf{U}\}^T [\mathbb{K}] \{\mathbf{U}\} > 0 \text{ for any } \{\mathbf{U}\} \neq \{0\}$$

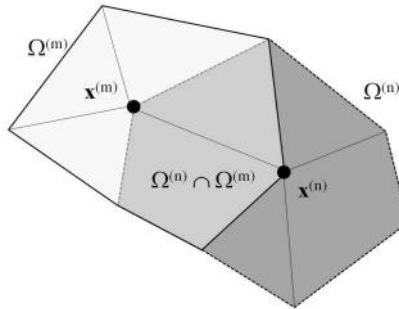
These properties hold for any Galerkin-type discretization and are not limited to finite elements.

**The stiffness matrix is sparse.** One fundamental consequence of the finite element discretization is that  $[\mathbb{K}]$  is *sparse*, which means that most of the matrix entries are zeros. As a matter of fact, if  $I$  and  $J$  denote the global numbering of two degrees of freedom, the coefficient  $[\mathbb{K}]_{IJ}$  can be formally rewritten in terms of the global representation of displacements (2.26):

$$[\mathbb{K}]_{IJ} = \int_{\Omega^{(m)} \cap \Omega^{(n)}} \boldsymbol{\varepsilon}[\tilde{N}_m(\mathbf{x})\mathbf{e}_i] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\tilde{N}_n(\mathbf{x})\mathbf{e}_j] dV(\mathbf{x})$$

$$I = \text{nodes}(m).\text{dof}(i), \quad J = \text{nodes}(n).\text{dof}(j), \quad (3.37)$$

where  $\Omega^{(m)}$  and  $\Omega^{(n)}$  are the geometrical supports of the global shape functions  $\tilde{N}_m(\mathbf{x})$  and  $\tilde{N}_n(\mathbf{x})$ . Since we have seen (Sec. 2.2.7) that every  $\Omega^{(m)}$  is the union of all the finite elements containing the node  $\mathbf{x}^{(m)}$ ,  $[\mathbb{K}]_{IJ}$  may be nonzero only if there exist a finite element containing both nodes  $\mathbf{x}^{(m)}$  and  $\mathbf{x}^{(n)}$  (Fig. 3.2). This implies that the matrix sparsity, defined as the ratio of non-zero entries over the total number  $N^2$  of entries, increases under mesh refinement.



**Figure 3.2:** Intersection  $\Omega^{(m)} \cap \Omega^{(n)}$  of the supports of the global shape functions  $\tilde{N}_m$  and  $\tilde{N}_n$  associated with the nodes  $\mathbf{x}^{(m)}$  and  $\mathbf{x}^{(n)}$ .

**Band and skyline structure of  $[\mathbb{K}]$ .** Thanks to symmetry only half of the matrix needs to be stored. Moreover, if the mesh is such that the global numbers of nodes and of the unknown nodal values in a single element are rather “close” with respect to the total number of nodes, the non-zero entries of  $[\mathbb{K}]$  will be concentrated near its diagonal. Hence it makes sense to define the *bandwidth*  $L_B < N$  such that

$$|I - J| > L_B \Rightarrow [\mathbb{K}]_{IJ} = 0$$

(with this definition a diagonal matrix has a zero bandwidth). The bandwidth decreases as the mesh is refined:

$$L_B/N \rightarrow 0 \quad (N \rightarrow \infty).$$

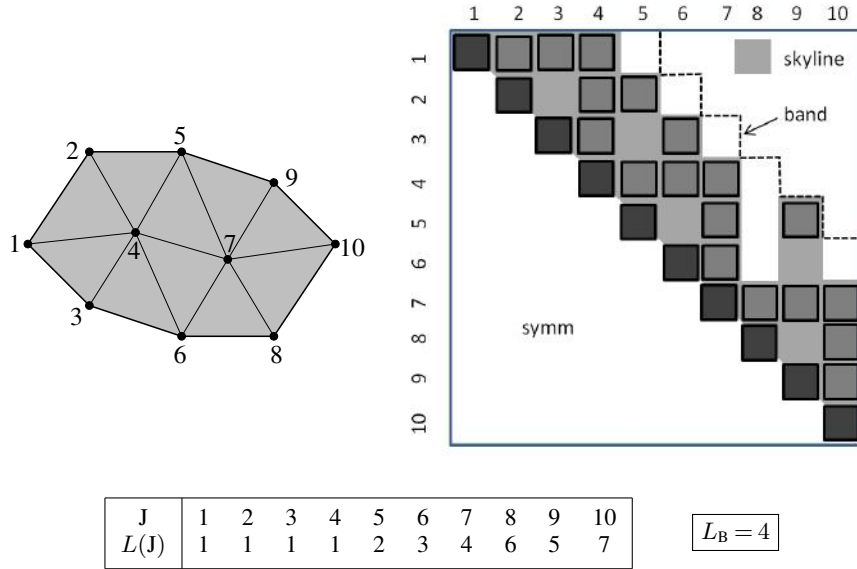
As it happens, the population of  $[\mathbb{K}]$  is such that the band is not filled uniformly. For every column  $J$  of  $[\mathbb{K}]$ , let us define  $L(J)$  ( $1 \leq L(J) \leq J$ ) as

$$L(J) = \min\{I \mid [\mathbb{K}]_{IJ} \neq 0\}. \quad (3.38)$$

We call *skyline* the function  $L(J)$  defined by (3.38) for a given matrix  $[\mathbb{K}]$ . The bandwidth and the skyline are linked by the condition

$$L_B = \max_{1 \leq J \leq N} (J - L(J)).$$

The band and skyline structure are illustrated in Figure 3.3 for a simple mesh made of linear triangles T3 (10 nodes, 10 elements, one scalar unknown per node).



**Figure 3.3:** Example of mesh (left) and population of a stiffness matrix built on this mesh, with scalar nodal unknowns (right). The squares denote the non-zero entries of  $[\mathbb{K}]$ , and the band and skyline of the matrix are outlined.

### 3.4.2 Direct solution by matrix factorization

Since the matrix  $[\mathbb{K}]$  is symmetric and positive definite, it is possible to recast it into a factorized form, called “ $LDL^T$  form”:

$$[\mathbb{K}] = [\mathbb{L}][\mathbb{D}][\mathbb{L}]^T, \quad (3.39)$$

where  $[\mathbb{D}]$  is a positive definite diagonal matrix (i.e.  $[\mathbb{D}]_{II} > 0$  for any  $I$ ) and  $[\mathbb{L}]$  is a lower triangular matrix with unit diagonal ( $[\mathbb{L}]_{II} = 1$  for any  $I$ ). Once the linear system is rewritten as (3.5)

$$[\mathbb{L}][\mathbb{D}][\mathbb{L}]^T \{\mathbb{U}\} = \{\mathbb{F}\},$$

$\{\mathbb{U}\}$  can be obtained by the easy computational task of successively solving two *triangular* systems:

$$(i) \quad [\mathbb{L}]\{\mathbb{Z}\} = \{\mathbb{F}\}, \quad (ii) \quad [\mathbb{L}]^T \{\mathbb{U}\} = [\mathbb{D}]^{-1} \{\mathbb{Z}\}.$$

**$LDL^T$  decomposition.** The decomposition (3.39) can be computed applying an algorithm which flows directly from imposing (3.39) for every coefficient of  $[\mathbb{K}]$ , account taken of the assumptions made on  $[\mathbb{L}]$ :

$$[\mathbb{K}]_{JJ} = [\mathbb{D}]_{JJ} + \sum_{K=1}^{J-1} [\mathbb{L}]_{JK}^2 [\mathbb{D}]_{KK} \quad (1 \leq J \leq N), \quad (a)$$

$$[\mathbb{K}]_{IJ} = [\mathbb{L}]_{JI} [\mathbb{D}]_{II} + \sum_{K=1}^{I-1} [\mathbb{L}]_{IK} [\mathbb{L}]_{JK} [\mathbb{D}]_{KK} \quad (1 \leq I \leq J-1, 2 \leq J \leq N), \quad (b)$$

where the symmetry of  $[\mathbb{K}]$  allows to restrict ourselves to  $I \leq J$ . We consider in a sequence the cases  $J = 1, J = 2, \dots, J = N$ .

- $J = 1$  (initialisation): equation (a) provides  $\mathbb{D}_{11}$  explicitly:

$$[\mathbb{D}]_{11} = [\mathbb{K}]_{11}; \quad (c)$$

- $J = 2, \dots, N$ : equation (b), written for every  $I$  ( $1 \leq I \leq J-1$ ), gives  $\mathbb{L}_{JI}$ :

$$\mathbb{L}_{JI} = \frac{1}{[\mathbb{D}]_{II}} \left[ [\mathbb{K}]_{IJ} - \sum_{K=1}^{I-1} [\mathbb{L}]_{IK} [\mathbb{L}]_{JK} [\mathbb{D}]_{KK} \right] \quad (1 \leq I \leq J-1) \quad (d)$$

and thus allows to sequentially compute all the entries of the  $J$ -th row of  $[\mathbb{L}]$  knowing the  $J-1$  previous rows and the first  $J-1$  entries of  $\text{Diag}([\mathbb{D}])$ . Next, equation (a) gives  $[\mathbb{D}]_{JJ}$  from known quantities:

$$[\mathbb{D}]_{JJ} = [\mathbb{K}]_{JJ} - \sum_{K=1}^{J-1} [\mathbb{L}]_{JK}^2 [\mathbb{D}]_{KK}. \quad (e)$$

Using this algorithm, all the entries of  $[\mathbb{L}]$  and  $\text{Diag}([\mathbb{D}])$  can be rapidly computed.

**Preservation of the band and skyline structure.** A careful analysis of the decomposition algorithm reveals the following properties:

- If  $[\mathbb{K}]$  has a skyline (defined by  $L(J)$ ), then  $[\mathbb{L}]^T$  has the same skyline;
- If  $[\mathbb{K}]$  has a bandwidth  $L_B$ , then  $[\mathbb{L}]^T$  has the same bandwidth.

These properties are very important from a practical viewpoint since they allow to store the entries of  $[\mathbb{L}]$  and  $[\mathbb{D}]$  in the memory space initially allocated for  $[\mathbb{K}]$ .

### 3.4.3 Storage of the stiffness matrix

The properties of  $[\mathbb{K}]$  illustrated in the previous sections show that it is not necessary to store the whole  $[\mathbb{K}]$  (*a priori*  $N^2$  entries):

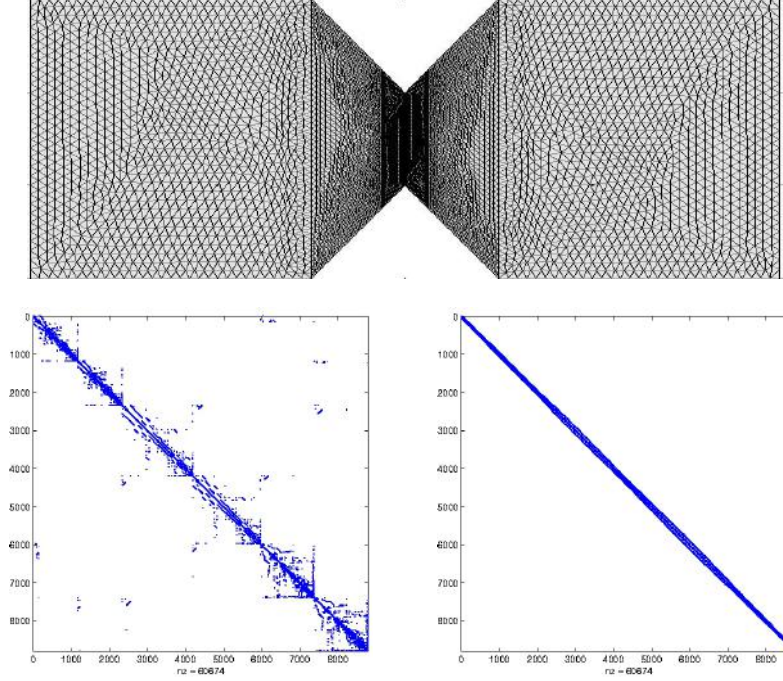
- (a)  $[\mathbb{K}]$ , being symmetric, contains at most  $N(N+1)/2$  independent entries (e.g. those in the upper triangular part)
- (b) The band structure of  $[\mathbb{K}]$  reduces the storage requirement to  $N \times (L_B + 1)$  entries approximately, with  $L_B \ll N$  if  $N$  is large
- (c) This estimate can be further reduced by adopting a skyline storage technique consisting in saving only the entries below the skyline  $L(J)$ .
- (d) In many cases it is convenient to store only the non-zero entries of  $[\mathbb{K}]$ . This can be achieved, for instance, by means of the Morse storage, which involves three one-dimensional arrays:
  - A list  $C$  of real numbers  $\{[\mathbb{K}]_{IJ} \neq 0, 1 \leq I \leq N, L(I) \leq J \leq I\}$ . The matrix  $[\mathbb{K}]$  is scanned row-wise, with the nonzero entries  $[\mathbb{K}]_{IJ}$  of the  $I$ -th row stored for increasing  $J$ .
  - A list  $J$  of integers with the column number  $J$  of every nonzero entry  $[\mathbb{K}]_{IJ}$  stored in  $C$ ;
  - A list  $I$  of  $N$  integers such that  $I(I)$  gives the position in  $[\mathbb{K}]$  of the last nonzero entry of row  $I$ .

The  $LDL^T$  decomposition normally fills all the positions below the skyline, meaning that zeros entries of  $[\mathbb{K}]$  located below the skyline are not preserved in  $[\mathbb{L}]$ . For a given mesh, the skyline depends on the global numbering of unknowns. It is desirable to optimise it by renumbering the nodes by means of a well-chosen permutation. An example is provided by the Cuthill-McKee algorithm Cuthill and McKee (1969) and implemented in MATLAB, together with many variants like the Approximate Minimum Degree (AMD) algorithm. For illustration purposes, Figure 3.4 shows the population of the stiffness matrix associated with the mesh of a wedged specimen (plane elasticity, 17150 T3 elements, 8762 nodes) before and after applying the MATLAB symrcm renumbering algorithm.

### 3.4.4 Iterative solver based on the Conjugate Gradient Method

For a model involving a large number  $N$  of unknowns (millions of degrees of freedom are nowadays standard practice), storing and factoring  $[\mathbb{K}]$  in memory becomes problematic even if skyline techniques with renumbering are applied.





**Figure 3.4:** Top: mesh of a wedged specimen (plane elasticity, 17150 T3 elements, 8762 nodes). Bottom: population of the stiffness matrix assembled on this mesh, before (left) and after (right) applying the MATLAB symrcm renumbering function. The proportion of nonzero entries of  $[\mathbb{K}]$  is  $\approx 7,9 \cdot 10^{-4}$ , and one has  $L_B/N \approx 0.0137$  after renumbering.

To remedy this issue, several approaches have been developed, including domain decomposition methods (Farhat and Roux, 1994; Gosselet and Rey, 2006) and iterative solvers (Greenbaum, 1997; Saad, 2003). These are technical topics which are still the subject of intensive research. Here we will limit ourselves to presenting a simple iterative algorithm well adapted to symmetric positive definite matrices: the *conjugate gradient method*.

The conjugate gradient method was originally developed for solving unconstrained minimization problems involving differentiable functions of  $N$  variables (the search space thus being the whole  $\mathbb{R}^N$ ). Since solving an elastic equilibrium problem amounts to finding the displacement field  $\mathbf{v}$  minimizing the total potential energy  $\mathcal{P}(\mathbf{v})$ , this can be treated as an optimization problem. From a practical standpoint, the matrix equation  $[\mathbb{K}]\{\mathbf{U}\} = \{\mathbf{F}\}$  can be viewed as the first-order necessary optimality condition for the discretized potential energy  $P(\{\mathbf{V}\})$ :

$$P(\{\mathbf{V}\}) = \frac{1}{2}\{\mathbf{V}\}[\mathbb{K}]\{\mathbf{V}\} - \{\mathbf{V}\}^T\{\mathbf{F}\} + P(\{\mathbf{U}^{(D)}\}), \quad (3.40)$$

where the constant vector  $\{\mathbf{U}^{(D)}\}$  is associated with the interpolation of prescribed

displacements on the nodes of  $S_u$ . Since  $[\mathbb{K}]$  is positive definite, the solution of  $[\mathbb{K}]\{\mathbf{U}\} = \{\mathbf{F}\}$  yields the minimum of  $P$  among the kinematically admissible displacements  $\{\mathbf{V}\}$ . In general, iterative methods for the solution of linear systems proceed by generating a minimising sequence  $\{\mathbf{U}^{[k]}\}$  (where  $k \geq 0$  is the iteration number), starting from an initial estimate  $\{\mathbf{U}^{[0]}\}$  and based on the computation of the *residual*

$$\{\mathbf{R}^{[k]}\} = \{\mathbf{F}\} - [\mathbb{K}]\{\mathbf{U}^{[k]}\} \quad (k \geq 0) \quad (3.41)$$

It should be stressed that  $\{\mathbf{R}^{[k]}\}$  can be in principle obtained computing the coefficients of  $[\mathbb{K}]$  on the fly at every iteration  $k$  without storing them.

The conjugate gradient method creates, at each iteration, a descent direction, i.e. a vector  $\{\mathbf{D}^{[k]}\} \in \mathbb{R}^N$  such that

$$\frac{d}{d\delta} P(\{\mathbf{U}^{[k]}\} + \delta\{\mathbf{D}^{[k]}\}) \big|_{\delta=0} < 0.$$

This guarantees the existence of a value  $\delta^{[k]} > 0$  of  $\delta$  such that the potential energy is decreased during the iteration:  $P(\{\mathbf{U}^{[k]}\} + \delta^{[k]}\{\mathbf{D}^{[k]}\}) < P(\{\mathbf{U}^{[k]}\})$ . Moreover, the directions  $\{\mathbf{D}^{[k]}\} \in \mathbb{R}^N$  are  $[\mathbb{K}]$ -conjugate, i.e. orthogonal in the sense of the scalar product associated with  $[\mathbb{K}]$ . As a result, the successive descent directions generated by the algorithm are linearly independent.

The conjugate gradient method applied to the potential energy (3.40) can be implemented as described in Box 3.1 (where  $\epsilon$  denotes a *tolerance* set *a priori* by the user).

We remark in particular that each iteration requires only one matrix-vector product  $\{\mathbf{Z}^{[k]}\} = [\mathbb{K}]\{\mathbf{R}^{[k-1]}\}$  (step 2-(i)), while the other steps have a much smaller computational cost. The conjugate gradient algorithm, when applied to quadratic functions of the type (3.40) has a remarkable property:

- *In exact arithmetic (i.e. neglecting the effect of machine truncation errors), the algorithm converges (with  $\epsilon = 0$ ) in at most  $N$  iterations.*

In practice one is rather interested in an approximate convergence in terms of fulfilling a small, but nonzero, tolerance  $\epsilon$  within a number of iterations much smaller than the number of unknowns. The speed of convergence greatly depends on the use of a suitable *preconditioning* of the system. The latter modification formally consists in solving the modified form

$$([\mathbb{P}]^T[\mathbb{K}][\mathbb{P}])\{\mathbf{Y}\} = [\mathbb{P}]^T\{\mathbf{F}\} \quad \text{with} \quad [\mathbb{P}]\{\mathbf{Y}\} = \{\mathbf{U}\}$$

where the matrix  $[\mathbb{P}]$  is chosen in such a way that  $[\mathbb{P}]^T[\mathbb{K}][\mathbb{P}]$  is better conditioned than  $[\mathbb{K}]$ . A very simple, but still useful, choice for the *preconditioner*  $[\mathbb{P}]$  is:

$$[\mathbb{P}]_{II} = ([\mathbb{K}]_{II})^{-1/2}, \quad [\mathbb{P}]_{IJ} = 0 \quad (I \neq J)$$

where, since  $[\mathbb{K}]$  is definite positive, all the diagonal terms are strictly positive.

**Box 3.1:** Scheme of the conjugate gradient method

## 1. Initialisation:

- (i) Choice of  $\{\mathbb{U}^{[0]}\} \in \mathbb{R}^N$  (often,  $\{\mathbb{U}^{[0]}\} = \{0\}$ ) and of the tolerance  $\epsilon$ ;
- (ii) Initial residuum:  $\{\mathbb{R}^{[0]}\} = \{\mathbb{F}\} - [\mathbb{K}]\{\mathbb{U}^{[0]}\}$ ;
- (iii) Initial descent direction:  $\{\mathbb{D}^{[0]}\} = \{\mathbb{R}^{[0]}\}$  (gradient change of sign).

2. For  $k = 1, 2, \dots$  and until  $\|\{\mathbb{R}^{[k]}\}\| > \epsilon$ :

- (i) Compute  $\{\mathbb{Z}^{[k]}\} = [\mathbb{K}]\{\mathbb{R}^{[k-1]}\}$ ;
- (ii) Compute 
$$\delta^{[k]} = \frac{\{\mathbb{D}^{[k-1]}\}^T \{\mathbb{R}^{[k-1]}\}}{\{\mathbb{D}^{[k-1]}\}^T \{\mathbb{Z}^{[k]}\}}$$
- (iii) Update solution:  $\{\mathbb{U}^{[k]}\} = \{\mathbb{U}^{[k-1]}\} + \delta^{[k]}\{\mathbb{D}^{[k-1]}\}$ ;
- (iv) Update residuum:  $\{\mathbb{R}^{[k]}\} = \{\mathbb{R}^{[k-1]}\} + \delta^{[k]}\{\mathbb{Z}^{[k]}\}$ ;
- (v) Convergence test: if  $\|\{\mathbb{R}^{[k]}\}\| \leq \epsilon$ , set  $\{U\} = \{U^{[k]}\}$ , END;
- (vi) Compute 
$$\beta^{[k]} = \frac{\{\mathbb{R}^{[k]}\}^T \{\mathbb{R}^{[k]}\}}{\{\mathbb{R}^{[k-1]}\}^T \{\mathbb{R}^{[k-1]}\}};$$
- (vii) Update descent direction:  $\{\mathbb{D}^{[k]}\} = \{\mathbb{R}^{[k]}\} + \beta^{[k]}\{\mathbb{D}^{[k-1]}\}$ ;
- (viii) Go to next iteration:  $k \leftarrow k + 1$  and restart from 2-(i).

**3.4.5 Post-processing phase**

Once the linear system (3.36) has been solved for the nodal displacements  $\{\mathbb{U}\}$ , the displacement field  $\mathbf{u}_h(\mathbf{x})$  interpolating  $\{U_e\}$  according to (2.21) can be built in each element  $E_e$ . Next, the approximate strain  $\{\varepsilon[\mathbf{u}_h]\}$  and stress  $\{\sigma[\mathbf{u}_h]\}$  fields flow from (3.12) and (3.10) everywhere in  $\Omega_h$ . However,  $\{\sigma[\mathbf{u}_h]\}$  is not statically admissible and, in general, is discontinuous across elements. Without entering into details, stresses computed at Gauss points are known to be more accurate (super-convergent):

$$\{\sigma^{(g)}\} := \{\sigma(\mathbf{x}_g)\} = [A][B(\mathbf{a}_g)]\{U_e\} \quad \text{at Gauss point of element } E_e$$

where  $e$  and  $g$  run over the number of elements and of Gauss points over each element, respectively. As a consequence, most post-processing tools build the visualization of the stress field in  $\Omega_h$  starting from the set of values  $\{\sigma^{(g)}\}$ . First, nodal stresses are extrapolated from their values at Gauss points; next, a global continuous interpolation is built according to (2.21). Such techniques (see e.g. Zienkiewicz and Taylor, 2000a, Chap. 13), which are also often applied to estimate the convergence of the method to the exact solution and to define improved evaluation of stresses, are beyond the scope of this book and will not be addressed in detail.

### 3.5 Convergence

The finite element method, applied to an equilibrium problem of linear elasticity, allows to compute an approximate displacement field  $\mathbf{u}_h$  for a given mesh and a given choice of shape functions. A fundamental issue consists then in analysing the convergence of the approximate solution  $\mathbf{u}_h$  towards the exact solution  $\mathbf{u}$ , as well as the convergence of strains and stresses. The convergence analysis of the finite element method has been studied as part of the mathematical theory of the finite element methods (e.g. Ciarlet, 1980; Strang and Fix, 1973; Babuska and Strouboulis, 2001). We will here only mention a few basic results.

#### 3.5.1 Basic theoretical results from convergence analysis

The properties discussed in Section 1.4.2 apply here, since the displacement-based FEM studied in this chapter is indeed a Galerkin approach. In particular  $\mathbf{u}_h$  is the best approximation of the exact solution  $\mathbf{u}$ , in the energy norm sense, among all  $\mathbf{v} \in \mathcal{C}_h(\mathbf{u}^D)$ . However, this does not provide any hint about the convergence rate of  $\mathbf{u}_h$  towards  $\mathbf{u}$  as the mesh is refined or the finite element space is enriched. The energy norm

$$\|\mathbf{v}\|_{\mathbb{E}}^2 := 2\mathcal{W}(\mathbf{v}) = \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{v}] : \mathcal{A} : \boldsymbol{\varepsilon}[\mathbf{v}] \, dV \quad (3.42)$$

is actually a seminorm, since for any infinitesimal rigid-body displacement  $\mathbf{w}$  (i.e. any  $\mathbf{w}$  of the form (1.2)) we have  $\|\mathbf{w}\|_{\mathbb{E}} = 0$  and  $\|\mathbf{v}\|_{\mathbb{E}} = \|\mathbf{v} + \mathbf{w}\|_{\mathbb{E}}$ . However, we will here assume that suitable boundary conditions block all rigid body motions. In this case the energy norm and the Sobolev  $H^1$  norm (defined in the sequel) are equivalent norms. Sobolev spaces define the proper mathematical setting for the convergence analysis of the finite element method. We recall that vector field  $\mathbf{v} \in \mathbb{R}^3$  belongs to the order  $k$  Sobolev space  $H^k$  if all the multi-index derivatives  $D^{\alpha}$  of its components:

$$D^{\alpha} \mathbf{v} = \frac{\partial^{|\alpha|} \mathbf{v}(\mathbf{x})}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \partial x_3^{\alpha_3}} \quad \text{with} \quad |\alpha| = \alpha_1 + \alpha_2 + \alpha_3 \quad (3.43)$$

(defined in the distributional sense) are square-integrable for  $|\alpha| \leq k$ . The natural norm  $\|\mathbf{v}\|_{H^k}$  in a Sobolev space is expressed in terms of the seminorms  $|\mathbf{v}|_{H^p}$ :

$$\|\mathbf{v}\|_{H^k} = \left( \sum_{p=0}^k |\mathbf{v}|_{H^p}^2 \right)^{1/2}, \quad |\mathbf{v}|_{H^p} := \left( \sum_{|\alpha|=p} \|D^{\alpha} \mathbf{v}\|_{L^2}^2 \right)^{1/2} \quad (3.44)$$

(where  $H^0$  coincides with the space  $L^2$  of square-integrable vector fields). For instance, in a 2D vector problem:

$$\|\mathbf{v}\|_{H^0}^2 = \int_{\Omega} \mathbf{v} \cdot \mathbf{v} \, dV, \quad \|\mathbf{v}\|_{H^1}^2 = \|\mathbf{v}\|_{L^2}^2 + \int_{\Omega} \sum_{i=1}^2 \left[ \left( \frac{\partial v_i}{\partial x_1} \right)^2 + \left( \frac{\partial v_i}{\partial x_2} \right)^2 \right] dV$$

Let us now suppose that the exact solution  $\mathbf{u} \in H^{r+1}$ . Then, if the finite elements can represent exactly (over each element) any polynomial function of degree  $\leq p$

in  $\mathbf{x}$ , the following error estimates hold:

$$\|\mathbf{u} - \mathbf{u}_h\|_{H^0} = \|\mathbf{u} - \mathbf{u}_h\|_{L^2} \leq C_0 h^{s+1} \|\mathbf{u}\|_{H^{s+1}} \quad s = \min\{p, r\} \quad (3.45)$$

$$\|\mathbf{u} - \mathbf{u}_h\|_{H^1} \leq C_1 h^s \|\mathbf{u}\|_{H^{s+1}} \quad s = \min\{p, r\} \quad (3.46)$$

where the positive constants  $C_0, C_1$  do not depend on  $\mathbf{u}$  and  $h$  (the typical element size). Without entering into details, an intuitive justification can be provided as follows. If the solution is sufficiently smooth, the order  $p$  Taylor expansion  $\mathbf{u}_{T,p}(\mathbf{x})$  can be computed, with

$$\begin{aligned} \mathbf{u}(\mathbf{x}) &= \mathbf{u}(\mathbf{x}^0) + \nabla \mathbf{u}(\mathbf{x}^0) \cdot \Delta \mathbf{x} + \nabla \nabla \mathbf{u}(\mathbf{x}^0) : (\Delta \mathbf{x} \otimes \Delta \mathbf{x}) + \dots + O(\|\Delta \mathbf{x}\|^{p+1}) \\ &= \mathbf{u}_{T,p}(\mathbf{x}) + O(\|\Delta \mathbf{x}\|^{p+1}) \quad (\mathbf{x}, \mathbf{x}^0 \in E_e) \end{aligned}$$

where  $\Delta \mathbf{x} = \mathbf{x} - \mathbf{x}^0$  and  $\mathbf{x}, \mathbf{x}^0$  belong to the same element. Hence, if  $\|\Delta \mathbf{x}\| \leq h$ :

$$\|\mathbf{u}(\mathbf{x}) - \mathbf{u}_{T,p}(\mathbf{x})\|_{H^0} = O(h^{p+1}) \quad \mathbf{x} \in E_e. \quad (3.47)$$

It also follows that the errors made on strains and (linear elastic) stresses are such that

$$\begin{aligned} \|\varepsilon[\mathbf{u}](\mathbf{x}) - \varepsilon[\mathbf{u}_{T,p}](\mathbf{x})\|_{H^0} &= O(h^p) \\ \|\boldsymbol{\sigma}[\mathbf{u}](\mathbf{x}) - \boldsymbol{\sigma}[\mathbf{u}_{T,p}](\mathbf{x})\|_{H^0} &= O(h^p) \end{aligned} \quad (3.48)$$

If the selected finite elements can represent exactly any polynomial function of degree  $\leq p$  in  $\mathbf{x}$ , then they can reproduce exactly the order  $p$  Taylor expansion  $\mathbf{u}_{T,p}$  of the exact solution. This remark, combined with the best approximation property and the equivalence of energy and Sobolev norms, can intuitively explain estimates (3.45)-(3.46).

Let us now consider (3.45)-(3.46) and start assuming that the solution is smooth enough to have  $p < r$ . Then, the factor limiting the convergence rate is  $p$ . It is worth stressing that any isoparametric element can represent exactly every linear field in  $\mathbf{x}$  (Section 2.2.6). As a consequence we always have  $p \geq 1$ . Focusing on plane elements, the simplest possible choices are the linear triangle T3 and the bilinear quadrilateral Q4. In both cases one has  $p = 1$  (neither element permitting exact representation of higher-degree polynomials), implying for both elements that (3.45)-(3.46) predict quadratic and linear convergence in the  $H^0$  and  $H^1$  norms, respectively. This means that displacements converge quadratically while strains converge only linearly with  $h$ .

Moving to higher order elements, we now consider the T6 triangle and Q8 quadrangle. These elements can represent exactly quadratic polynomials in the parameter  $\mathbf{a}$  (see Table 2.2), but this property also holds in the physical space only if the geometrical transformation (2.13) is affine. The latter holds in general only if mid-side nodes are placed exactly at the middle of the segment joining the vertices.

In this case the predicted convergence for strains is quadratic in  $h$  and the performance is largely superior with respect to T3 and Q4. However, quadratic convergence is in general lost when sides are curved. Moreover, even initially

straight sides may become distorted in geometrically non-linear analyses. In such cases, only linear convergence is guaranteed for strains.

It is anyway apparent from (3.46) that the smoothness of the exact solution plays a key role in the rate of convergence. In dynamical or non-linear problems with specific constitutive laws, large deformations, contact or fracture, the performance degrades substantially since  $r$  becomes the limiting factor independently of  $p$ , making the use of higher order elements less attractive.

### 3.5.2 Example: pressurized spherical shell

We now revisit the example of Section 1.5, focusing on the convergence properties of finite element solutions.

**A priori estimate of convergence in  $H^1$  norm.** In this 1D problem, the mathematical tools required to prove (3.46) are simple enough to permit some insight. We will make frequent use of the 1D Cauchy-Schwartz inequality, which for any square-integrable functions  $f, g$ , states that:

$$\int_a^b f(s)g(s) \, ds \leq \left( \int_a^b f^2(s) \, ds \right)^{1/2} \left( \int_a^b g^2(s) \, ds \right)^{1/2} \quad (3.49)$$

Now assume that  $f \in \mathcal{C}(0)$  (and therefore vanishes at  $r = R_2$ ):

$$|f(r)| = \left| \int_r^{R_2} f'(s) \, ds \right| \leq \int_{R_1}^{R_2} |f'(s)| \, ds \leq L^{1/2} \left( \int_{R_1}^{R_2} |f'(s)|^2 \, ds \right)^{1/2}$$

where  $L = R_2 - R_1$  and the Cauchy-Schwartz inequality has been applied with  $g = 1$ . Squaring and integrating over the radius, one obtains

$$\int_{R_1}^{R_2} |f(s)|^2 \, ds \leq L^2 \int_{R_1}^{R_2} |f'(s)|^2 \, ds \leq \frac{L^2}{R_1^2} \int_{R_1}^{R_2} |f'(s)|^2 s^2 \, ds, \quad (3.50)$$

which is a particular form of the Poincaré inequality. It is left as an exercise to show that, using (3.49) and (3.50) and assuming  $\lambda > 0, \mu > 0$ , the seminorm

$$|f|_{H^1}^2 := \int_{\Omega} |f'|^2 \, dV = 4\pi \int_{R_1}^{R_2} |f'(s)|^2 s^2 \, ds$$

is actually a norm which is equivalent to the energy norm

$$\begin{aligned} \|f\|_E^2 = 2\mathcal{W}(f) &= \int_{R_1}^{R_2} \left( (\lambda + 2\mu) f'^2 + 4\lambda \frac{f}{s} f' + 4(\lambda + \mu) \frac{f^2}{s^2} \right) s^2 \, ds \\ &= \int_{R_1}^{R_2} \left( \lambda (s f' + 2f)^2 + 2\mu f'^2 s^2 + 4\mu f^2 \right) \, ds \end{aligned}$$

i.e. there exist two positive constants  $C_i$  and  $C_s$  such that:

$$C_i |f|_{H^1} \leq \|f\|_E \leq C_s |f|_{H^1} \quad \forall f \in \mathcal{C}(0) \quad (3.51)$$

Having gathered these preliminary results, we now proceed to the analysis of convergence. Basically, since according to the best approximation property the numerical

solution minimizes in  $\mathcal{C}(0)$  the energy norm of the error, an upper bound can be obtained with any suitable element of  $\mathcal{C}(0)$  and in particular with the interpolant  $\Pi_h[u](r)$  of the exact solution:

$$\Pi_h[u](r) := \sum_{n=1}^{N_N} \tilde{N}_n(r) u(r^{(n)})$$

where the global representation of displacements (2.26) has been employed with the nodal values taken from the exact solution  $u$ . Next we define the “error”  $e_\pi$  as the difference between the exact solution and  $\Pi_h[u]$ :

$$e_\pi(r) = u(r) - \Pi_h[u](r)$$

which clearly vanishes at each node of the mesh. Let us now focus on the  $i$ -th element of nodal coordinates  $r^{(i)}$  and  $r^{(i+1)}$ . Since  $e_\pi(r^{(i)}) = e_\pi(r^{(i+1)}) = 0$ , there is a point  $z$  within the element such that  $e'_\pi(z) = 0$ . Moreover,  $\Pi_h''[u](r) = 0$  and

$$e'_\pi(r) = \int_z^r e''_\pi(s) ds = \int_z^r u''(s) ds \quad \forall r \in E_i$$

leading to the following bound on  $|e'_\pi|$ :

$$|e'_\pi(r)| \leq \int_{r^{(i)}}^{r^{(i+1)}} |u''(s)| ds \quad \forall r \in E_i \quad (3.52)$$

Now, applying the Cauchy-Schwarz inequality (with  $f = u''$  and  $g = 1$ )

$$\int_{r^{(i)}}^{r^{(i+1)}} |u''(s)| ds \leq h^{1/2} \left( \int_{r^{(i)}}^{r^{(i+1)}} |u''(s)|^2 ds \right)^{1/2}$$

hence, squaring (3.52) and integrating over one element:

$$\int_{r^{(i)}}^{r^{(i+1)}} |e'_\pi(s)|^2 ds \leq h^2 \int_{r^{(i)}}^{r^{(i+1)}} |u''(s)|^2 ds \quad (3.53)$$

Summing (3.53) over all elements, a (rough!) bound for the  $|e_\pi|_{H^1}$  seminorm is found:

$$\begin{aligned} |e_\pi|_{H^1}^2 &= \int_{R_1}^{R_2} |e'_\pi(s)|^2 s^2 ds \leq R_2^2 \int_{R_1}^{R_2} |e'_\pi(s)|^2 ds \\ &\leq R_2^2 h^2 \int_{R_1}^{R_2} |u''(s)|^2 ds \leq \frac{R_2^2}{R_1^2} h^2 \int_{R_1}^{R_2} |u''(s)|^2 s^2 ds = \frac{R_2^2}{R_1^2} h^2 |u''|_{L^2}^2 \end{aligned}$$

yielding (since  $|u''|_{L^2}^2 \leq \|u\|_{H^2}^2$ ):

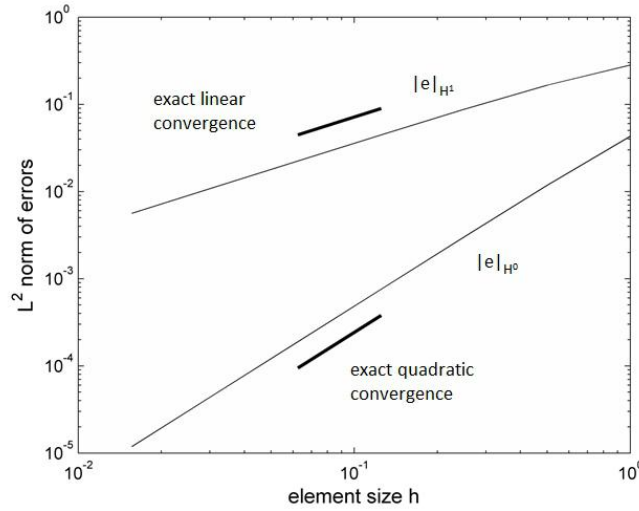
$$|e_\pi|_{H^1} \leq (R_2/R_1) h \|u\|_{H^2} \quad (3.54)$$

Finally, thanks to the approximation property  $\|e\|_E \leq \|e_\pi\|_E$  of the Galerkin approach and using the equivalence of norms (3.51)

$$|e|_{H^1} \leq \frac{1}{C_i} \|e\|_E \leq \frac{1}{C_i} \|e_\pi\|_E \leq \frac{C_s}{C_i} |e_\pi|_{H^1} \leq \frac{C_s R_2}{C_i R_1} h \|u\|_{H^2}$$

which corresponds to (3.46).

**Numerical experiments on convergence.** The two-node line element employed in the code developed in Section 1.5.6 coincides with the isoparametric B2 element of Table 2.2. We now proceed to the empirical verification of the convergence rate. Since displacements are linear and strains constant within an element the computation of  $|e|_{H^1} = \|e'\|_{L^2}$  and of  $|e|_{H^0} = \|e\|_{L^2}$  can be performed analytically, as done at the end of `sphere_B2_1S_solid.m`. The errors are plotted in LogLog scale in Figure 3.5 where the numerical results (thin lines) are seen to match perfectly the predicted convergence rates (thick lines).



**Figure 3.5:** Log-Log error plots of radial displacement  $u$  ( $\|e\|_{H^0}$ ) and radial strain  $u'$  ( $|e|_{H^1}$ ) for the sphere modelled with B2 elements. Predicted convergence rate (thick lines) and numerical results (thin lines)

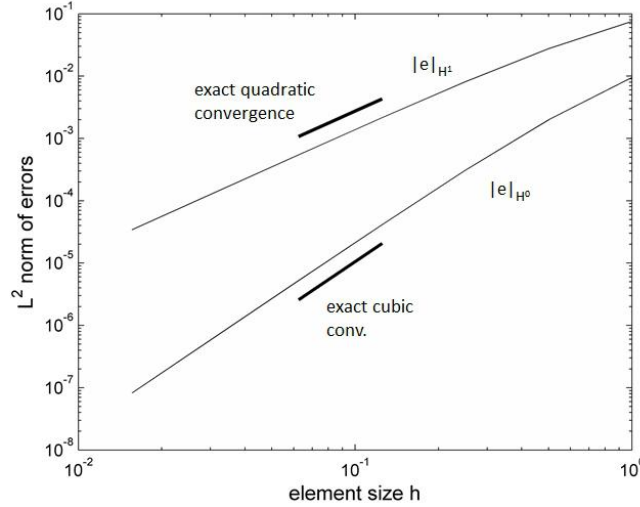
An extension of the B2 code to quadratic B3 elements has been proposed as an exercise in Section 2.1. Let us now introduce (in file `sphere_B3_1S_solid.m`), a distortion parameter  $-1 < \alpha < 1$  such that, when  $\alpha = 0$  the nodes are uniformly spaced (the third node of each B3 elements hence being placed exactly at the element midpoint).

```
h=(R2-R1)/(2*NE); % size of one element
NN=2*NE+1; % number of nodes
coor=[R1:h:R2]'; % uniform discretization
alpha=0.0; % distortion parameter
for n=2:2:NN-1
    coor(n)=coor(n)+alpha/2*(coor(n+1)-coor(n-1));
end
```

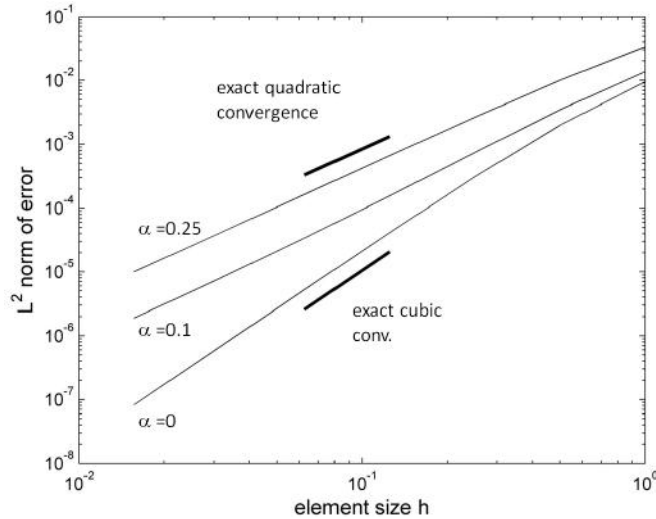
This implies that, when  $\alpha = 0$ , the geometrical transformation between the parametric and physical spaces is affine, each B3 element hence being capable of representing any quadratic polynomial in the physical coordinate. According to (3.45)-(3.46) the convergence of the radial strain and displacement should be quadratic and cubic, respectively, which is confirmed by the plots in Figure 3.6. On the contrary, when  $|\alpha|$  increases the middle node approaches one of the two endpoints and the transformation



is no longer affine. Only a complete linear polynomial in the physical coordinate can be exactly represented. The predicted convergence rate degrades, and this is verified in Figure 3.7, where the  $\|e\|_{L^2}$  error is plotted for three different increasing values of  $\alpha$ . It should be remarked that the cubic convergence rate for  $u$  is lost even for small distortions of the mesh.



**Figure 3.6:** Log-Log error plots of radial displacement  $u$  ( $\|e\|_{H^0}$ ) and radial strain  $u'$  ( $\|e\|_{H^1}$ ) for the sphere modelled with undistorted B3 elements. Predicted convergence rate (thick lines) and numerical results (thin lines)



**Figure 3.7:** Log-Log error plots of radial displacement  $u$  for the sphere modelled with B3 elements and artificial geometric distortion. Predicted convergence rate (thick lines) and numerical results (thin lines)

### 3.6 Code genlin

The code `genlin.m` implements the concepts discussed in previous sections for vector (solid mechanics) and scalar linear problems both in two and three dimensions. It rests on the GMSH code for pre- and post-processing purposes and the interface with GMSH is developed in separate files which are analysed in Appendix B. Even though this choice clearly influences some of the conventions adopted, the structure in `genlin.m` is fairly general and could be easily adapted to different mesh-generation and visualization tools. It should be stressed that the code has only teaching purposes, is not aimed at large-scale problems and is very rudimentary in many respects. However, the open-form of the code guarantees a great versatility and adaptability to different projects and applications which can be easily and rapidly developed: putting the user in the condition of fully exploiting this potentiality is the primary aim of this section.

A mandatory prerequisite is a basic knowledge of the MATLAB programming language, together with an operative understanding of the concepts developed in previous chapters (in particular a revision of Sections 2.2.3 and 2.2.8 is recommended) and of element-level routines described in the first part of this chapter. Moreover each analysis requires input files (mesh and analysis files) which must be prepared by the user according to the procedures presented through examples in Appendix B.

In the sequel the file `genlin.m` is described following the different phases of a standard analysis.

#### 3.6.1 Input phase

Initially all the sub-directories are added to the MATLAB search path and memory is cleared. Next the user is prompted to select the desired analysis file in the input directory which is then read at once.

```
addpath(genpath('.') % adds to path all subdirectories
clear all % clear all existing variables
[name,pname]=uigetfile('input/*.m',... % gets name of analysis file
    'Choose input file');
eval(['run input/',name]); % input file is read at once
```

It is worth recalling that, when encountering a MATLAB instruction which is not familiar to the user, typing on the command line `help` followed by the name of the instruction will help clarify the procedure.

In the final part of the input phase information concerning analysis type, geometry, materials and boundary conditions are extracted by the `readgmsh` procedure which is described in Section B.3. A detailed analysis of `readgmsh` is not required at this stage; it is sufficient to say that it elaborates the main variables analysis, nodes, elements, loads and materials which are defined hereafter.

**Variable analysis.** It is a structure (in the MATLAB sense) with many fields, including those listed in Table 3.1.

The available types of analysis are described in Table 3.2, where they are associated with tags. The use of tags in the code, to be explained in the sequel, is connected with element functions. According to the value of `analysis.type`, some parameters are fixed: `ndof`, the number of degrees of freedom per node; `DG`, the geometric dimen-

sion D. It is worth stressing that  $\text{ndof}=D$  for a problem of solid mechanics; the use of  $\text{ndof}$  is a generalisation which allows to run also thermal problems with  $\text{ndof}=1$ .

<code>analysis.type</code>	type of the analysis
<code>analysis.NN</code>	number of nodes
<code>analysis.NE</code>	number of elements
<code>analysis.neq</code>	number of unknowns
<code>analysis.NL</code>	number of elements with prescribed tractions

**Table 3.1:** Fields of analysis of interest for the assemblage

<code>analysis.type</code>	tag	type of the analysis
1	2D_therm_	2D thermal analysis
2	AX_therm_	axi-symmetric thermal analysis
3	3D_therm_	3D thermal analysis
4	2A_solid_	plane-strain solid analysis
5	2S_solid_	plane-stress solid analysis
6	AX_solid_	axi-symmetric solid analysis
7	3D_solid_	3D solid analysis

**Table 3.2:** Tags for the various types of analysis

The tags and the values of  $DG$  and  $\text{ndof}$  are defined in the list of structures `La`, at the beginning of the code.

```
La(1)=struct('tag','2D_therm_', 'ndof',1,'DG',2,'Sdim',2); % thermal 2D
La(2)=struct('tag','AX_therm_', 'ndof',1,'DG',2,'Sdim',3); % thermal AXI
La(3)=struct('tag','3D_therm_', 'ndof',1,'DG',3,'Sdim',3); % thermal 3D
La(4)=struct('tag','2A_solid_', 'ndof',2,'DG',2,'Sdim',3); % solid 2D
La(5)=struct('tag','2S_solid_', 'ndof',2,'DG',2,'Sdim',3); % solid 2D
La(6)=struct('tag','AX_solid_', 'ndof',2,'DG',2,'Sdim',4); % solid AXI
La(7)=struct('tag','3D_solid_', 'ndof',3,'DG',3,'Sdim',6); % solid 3D

Atag=La(analysis.type).tag;
ndof=La(analysis.type).ndof;
DG=La(analysis.type).DG;
```

The field `Sdim` is associated with the post-processing phase and will be explained in Section 3.6.6.

**Variable nodes.** The list of structures `nodes(1:analysis.NN)` collects nodal information (see Section 2.2.3 and Table 3.3). In particular, the field `nodes.U` contains the values of the nodal degrees of freedom (e.g. the  $D$  displacement components in a problem of solid mechanics and the temperature for a scalar heat conduction analysis). At the beginning of the analysis, nodal values prescribed by boundary conditions are set to their prescribed value and the corresponding entries in `nodes.dof` is set to  $-1$ . The other values `nodes.U` and `nodes.dof` are initialised to zero.

**Variable elements.** Each item of the list of structures `elements(1:analysis.NE)` contains the fields defined in Table 3.4.

The type is an integer code defining the element according to GMSH conventions. In the current release, GMSH can generate more than 30 different types of elements, including lines, triangles, quadrangles, hexahedra, tetrahedra, prisms and pyramids of different orders. Only few of these are implemented in `genlin` and are listed in Table 3.5. Moreover not all these elements have been developed for all the types of analysis: the user should perform the required checks and possibly develop new element-level routines. Several examples along these guidelines are presented in Section 3.7.

<code>nodes.coor(1:DG)</code>	nodal coordinates
<code>nodes.dof(1:ndof)</code>	global numbering of nodal dof
<code>nodes.U(1:ndof)</code>	values of nodal dof

**Table 3.3:** Fields for the list nodes

<code>elements.type</code>	element type
<code>elements.nodes(1:ne)</code>	connectivity of the element
<code>elements.mat</code>	number of the element material

**Table 3.4:** Fields for the list elements

The elements employed in the code are defined in the list of structures `Le` which sets the tag, the number  $n_e \rightarrow ne$  of nodes and the `c` field which will be analysed in the sequel.

```
Le(1)=struct('tag','B2_','ne',2,'c',1);
Le(2)=struct('tag','T3_','ne',3,'c',1);
Le(3)=struct('tag','Q4_','ne',4,'c',2);
Le(4)=struct('tag','P4_','ne',4,'c',1);
Le(8)=struct('tag','B3_','ne',3,'c',2);
Le(9)=struct('tag','T6_','ne',6,'c',3);
```

The structure adopted is flexible, in the sense that the user can add new elements by developing the required element-level subroutines and by filling, if necessary, new lines of `Le` according the GMSH conventions. An example of new element addition is provided in Section 3.7.6.

element.type	tag	type of element
1	B2_	2-node line B2
2	T3_	3-node triangle T3
3	Q4_	4-node quadrangle Q4
4	P4_	4-node tetrahedron P4
8	B3_	3-node line B3
9	T6_	6-node triangle T6

**Table 3.5:** Element types and tags

**Variable materials.** It is a matrix having as many lines as materials employed in the analysis. In the case of linear isotropic elasticity each line contains the Young modulus and the Poisson coefficient, but can be enriched for different applications (for instance plasticity, dynamics, thermoleasticity) by simply adding the new required coefficients

to each line. The variable is directly provided by the user in the analysis file which is read at the very beginning of the input phase. For instance

```
% MATERIAL: Young, Poisson and alpha
material= [2 .3 .01;
           1 .2 .001];
```

will generate a  $2 \times 3$  matrix defining two materials each having three coefficients which, in a thermoelastic analysis, will have the role of Young modulus, Poisson coefficient and thermal expansion.

**Variable loads.** The list of structures `loads(1:analysis.NL)` defines the line (in 2D) or surface (in 3D) elements which are subjected to applied tractions. The fields of each structure are listed in Table 3.6. It is worth recalling that the elements on which the external tractions are enforced are treated as independent isoparametric entities. However, for each element  $E_e$  in `loads`, there exist one and only one “parent” surface (in 2D) or volume (in 3D) element of the list `elements` which reduces to  $E_e$  on a boundary feature (edge or face). The fields `type` and `nodes` have the same meaning as for `elements`; `dir` defines the traction direction (its value being 1, 2 or 3 if that direction is one of the cartesian axes, or 0 to indicate that the traction is applied along the normal direction to the boundary).

<code>loads.type</code>	type of the element with tractions enforced
<code>loads.nodes(1:ne)</code>	connectivity of the element
<code>loads.dir</code>	direction of the applied tractions
<code>loads.val</code>	intensity of the applied tractions

**Table 3.6:** Fields of the list `loads` of loaded elements

### 3.6.2 Global numbering of unknowns

At the end of the execution of `readgmsh` the control is returned to the main file `genlin` which proceeds to the global numbering of the unknowns. Since some of the nodes may not be attached to material elements, the numbering phase is implemented exploiting a loop over the `elements` list. The counter `neq` for the global numbering is set to zero. Then, for each element, the code gets the `type`, the number `ne` of nodes in its connectivity and their list `convec`:

```
neq=0;
for e=1:analysis.NE,                                % assign equation numbers
    type=elements(e).type;
    ne=Le(type).ne;
    convec=elements(e).nodes;

    ... loop over the connectivity nodes ...

end
analysis.neq=neq;
```

The numbering is assigned in a second inner loop over the nodes of the connectivity. The global number `node` of the  $n$ -th local node is picked, and all the degrees of freedom of the node are screened through a third and last loop. If `nodes(node).dof(d)==0`, the  $d$ -th degree of freedom of node has not been prescribed by boundary conditions

and hence is unknown; the counter neq is incremented and the new value is assigned to the dof in analysis.

```

for n=1:ne
    node=connec(n);
    for d=1:ndof,                % loop over dofs
        if nodes(node).dof(d)==0, % if no dbc are enforced on node
            neq=neq+1;           % increments equation number
            nodes(node).dof(d)=neq; % assigns glob num to current dof
        end
    end
end
end

```

### 3.6.3 Allocation of the sparse stiffness matrix

MATLAB allows to manipulate sparse matrices. This is the approach adopted in the code genlin. At present, however, symmetry cannot be directly exploited and the whole matrix must be stored.

To allocate sufficient memory space for the sparse representation of  $[\mathbb{K}]$ , an upper bound of the number of nonzero entries of  $[\mathbb{K}]$  is required. Following the discussion of Section 3.4.1, the coefficient  $[\mathbb{K}]_{IJ}$  in (3.37) is non-zero only if the unknowns with global numbers I and J are associated with global nodes  $m$  and  $n$  belonging to the same element. If  $n_m$  denotes the number of nodes in  $\Omega^{(m)}$  (see Fig. 3.2), an upper bound for the number of non-zero entries in row I is hence ndof  $n_m$ . Since at most ndof unknowns are associated with node  $m$ , the equations associated with node  $m$  have at most ndof<sup>2</sup> $n_m$  non-zero entries. Summing over  $m$ , the global estimate ncoeffs is obtained.

The problem is now shifted to computing  $n_m$ . Let us start by considering the specific case of T3 elements. Taking the elements of  $\Omega^{(m)}$  one by one, the first element introduces  $n_e = 3$  new nodes, but each subsequent element only adds (at most) one new node. With T6 elements, the first element of  $\Omega^{(m)}$  introduces  $n_e = 6$  new nodes, the other ones three each (at most). In general the number of nodes added when a new element is included depends on the element type and is saved in the field c of the list Le. In order to compute the estimate ncoeffs, we adopt the following procedure. The list nm(1:analysis.NN), to be filled with  $n_m$  for all the nodes of the mesh, is initially set to zero.

```

nm=zeros(analysis.NN,1);          % initialisation of nm
for e=1:analysis.NE,              % loop over elements
    type=elements(e).type;
    connec=elements(e).nodes;     % gets connectivity
    for n=1:Le(type).ne
        node=connec(n);
        if nm(node)==0
            nm(node)=nm(node)+Le(type).ne; % first time: adds num of nodes
        else
            nm(node)=nm(node)+Le(type).c; % otherwise adds .c
        end
    end
end
ncoeffs=ndof^2*sum(nm);           % sum of all the terms in nm
K=spalloc(analysis.neq,analysis.neq,ncoeffs); % allocates sparse matrix
F=zeros(analysis.neq,1);          % allocates rhs vector

```

A loop over the elements is started. Whenever a node is found in the element connectivity, either `Le(type).ne` or `Le(type).c` is added to `nm` according to whether or not that node is encountered for the first time. The final estimate for `ncoeffs` is obtained as `ndof^2*sum(nm)`. At this point the command `spalloc` allocates a sparse matrix with `analysis.neq` lines and columns and at most `ncoeffs` non-zero coefficients. During the assemblage and solution phase the matrix is treated as sparse, even if this is totally transparent to the user.

### 3.6.4 Assemblage of the stiffness matrix and of nodal forces due to imposed displacements

Before proceeding to the description of the assemblage phase in `genlin` we need to address one technical detail associated with the use of tags in the code. In order to treat at once different types of analyses employing various 2D or 3D elements, the functions computing element quantities are called by the main program using a special syntax. For instance, in the case of the stiffness matrix:

```
Ke=eval([Etag Atag 'Ke(Xe,material(mat,:))']);
```

The command `eval` takes as input a string of characters and executes the MATLAB instruction defined by that string. Here the string concatenates three substrings: the element tag `Etag`, the analysis tag `Atag` and `'Ke(Xe,material(mat,:))'`. As an example, for a plane-strain analysis and a T3 element, the command above is equivalent to:

```
Ke=T3_2A_solid_Ke(Xe,,material(mat,:));
```

This procedure will be employed several times in the code.

The assemblage phase in `genlin` basically consists of a loop over elements in which first the element stiffness matrix is computed and then assembled:

```
for e=1:analysis.NE,
    ... computation of element quantities ...
    ... assemblage ...
end
```

We now examine in detail the procedure followed for each element, which begins with the definition of some variables: `type` (element type), `Dne` (number of degrees of freedom in the element, and hence also length of e.g.  $\{U_e\}$ ,  $\{W_e\}$  and  $\mathcal{G}_e$ ); `mat` (element material)

```
type=elements(e).type;           % type of the element
Etag=Le(type).tag;               % element tag
ne=Le(type).ne;                  % number of nodes in one element
Dne=ndof*ne;                     % number of nodal values in one el.
mat=elements(e).mat;             % element material
```

Recalling that `DG` stands for the geometrical dimension `D` of the analysis, the code proceeds to the definition of: `Xe`, a matrix collecting the nodal coordinates, a node per line;  $\mathcal{G}_e \rightarrow \mathcal{G}_e$ , a list providing the correspondence between local and global numbering;  $\{U_e\} \rightarrow U_e$  an array containing the value of the degrees of freedom in the element in local numbering – actually only boundary conditions, the rest being set to zero at

this stage.

```

Xe=zeros(ne,DG); % variables are initialised
Ge=zeros(Dne,1);
Ue=zeros(Dne,1);
pos=1;
for n=1:ne % and then filled using...
    node=elements(e).nodes(n); % a loop over nodes
    Xe(n,:)=nodes(node).coor; % creates element
    Ge(pos:pos+ndof-1)=nodes(node).dof;
    Ue(pos:pos+ndof-1)=nodes(node).U;
    pos=pos+ndof;
end

```

The code then launches the computation of the element stiffness matrix:

```
Ke=eval([Etag Atag 'Ke(Xe,material(mat,:))']);
```

The assemblage phase itself is now addressed, resorting to the formulas (3.32) and 3.33. In particular the list  $\mathcal{L}_e^{(0)} \rightarrow \text{Le0}$  can be extracted from  $\mathcal{G}_e$  as  $\text{Le0} = \text{find}(\text{Ge}>0)$ , since the MATLAB command `find` saves in  $\text{Le0}$  the position of the coefficients in  $\text{Ge}$  which are strictly positive. Similarly  $\mathcal{L}_e^{(D)} \rightarrow \text{LeD}$  can be obtained as  $\text{LeD} = \text{find}(\text{Ge}<0)$ . For the purpose of simplifying the notation in the section and in the codes we finally define the restriction of  $\text{Ge}$  to the positions  $\text{Le0}$ :  $\text{Ie} = \text{Ge}(\text{Le0})$ . It should be remarked that  $\text{Ie}$  contains the list of all the global unknowns in the element.

MATLAB provides an easy way to apply formula 3.32 at once for all the admissible indices: the submatrix of  $[K_e]$  of indices  $p, q \in \mathcal{L}^{(0)}$  is  $\text{Ke}(\text{Le0}, \text{Le0})$  while the submatrix of  $[\mathbb{K}]$  with indices  $\text{I} = \mathcal{G}^{(0)}(p), \text{J} = \mathcal{G}^{(0)}(q)$  is  $\text{K}(\text{Ie}, \text{Ie})$ .

```

Le0=find(Ge>0); % local numbering of unknowns
Ie=Ge(Le0); % global numbering
K(Ie,Ie)=K(Ie,Ie)+Ke(Le0,Le0); % matrix assemblage
LeD=find(Ge<0); % local numbering of prescr. b.c
F(Ie)=F(Ie)-Ke(Le0,LeD)*Ue(LeD); % assemblage of rhs

```

An analogous procedure allows to perform the assemblage of  $\{\mathbb{F}^u\}$  according to (3.33).

The great simplicity of this procedure has however a major drawback for problems of large dimensions. Since it is not a priori defined which coefficients will be filled during the assemblage phase, MATLAB has to perform internal searches over the whole stiffness matrix each time the command  $\text{K}(\text{Ie}, \text{Ie}) = \text{K}(\text{Ie}, \text{Ie}) + \text{Ke}(\text{Le0}, \text{Le0})$  is called in order to avoid to duplicate coefficients. This considerably slows down the assemblage phase especially for vector problems when fine meshes are employed. As an alternative, a second version of the assemblage has been developed in the code `genlin_fast`, where a different storage technique is employed. The details of this procedure are presented in Appendix B.6.

### 3.6.5 Assemblage of nodal forces due to applied tractions

The assemblage of the list of nodal forces due to given tractions is obtained by analogy with the procedure explained in the previous section. It consists of a loop over the



elements of loads:

```

for e=1:analysis.NL,                % for each element in loads
    type=loads(e).type;              % type of the element
    Etag=Le(type).tag;               % element tag
    ne=Le(type).ne;                  % number of nodes
    Dne=ndof*ne;                     % num of nodal values in one el
    idir=loads(e).dir;               % traction direction
    val=loads(e).val;                % traction value
    Xe=zeros(ne,DG);
    Ge=zeros(Dne,1);
    pos=1;
    for n=1:ne
        node=loads(e).nodes(n);
        Xe(n,:)=nodes(node).coor;    % matrix of nodal coords
        Ge(pos:pos+ndof-1)=nodes(node).dof; % global numbering
        pos=pos+ndof;
    end

    ... creates element vector ...
    ... performs the assemblage ...

end

```

The first set of instructions is formally identical to the one employed for the assemblage of the global stiffness matrix. The element vector  $\{F_e\}$  is created resorting again to the eval command

```
Fe=eval([Etag Atag 'Fe_surf(Xe,val,idir)']);
```

As an example, for a plane-strain analysis and a B2 element, the string input to eval becomes B2\_2A\_Fe\_surf(Xe,val,idir). The assemblage simply translates into MATLAB language the formula (3.35)

```

Le0=find(Ge>0);                    % local num. of unknown nodal values
Ie=Ge(Le0);                         % global num. of unknown nodal values
F(Ie)=F(Ie)+Fe(Le0);                % adds to global rhs

```

### 3.6.6 System solution and post-processing

MATLAB has highly efficient internal routines and the simplest choice is to employ the operator “\” (*backslash*), which lets MATLAB choose the best solver:

```
U=K\F;                               % solution of linear system
```

The field nodes.U of nodal values is first filled with the solution:

```

for n=1:analysis.NN                % loop over the nodes
    for dir=1:ndof                  % loop over all nodal values
        dof=nodes(n).dof(dir);      % gets the global unknown num.
        if dof>0                    % if it is unknown
            nodes(n).U(dir)=U(dof); % fills nodal value from sol vector
        end
    end
end

```

Next, stresses (for mechanical problems) or gradients (for potential problems) are computed at nodes. Since the number of components Sdim depends on the analysis.type,

as anticipated in Section 3.6.1,  $S_{dim}$  is first extracted from  $La$  and the output matrix  $S_n$  is allocated.

```
Sdim=La(analysis.type).Sdim;
Sn=zeros(analysis.NN,Sdim);          % initializes output matrix
```

For every element, output quantities are computed in terms of  $X_e$  (the usual matrix of nodal coordinates), the material  $material(mat,:)$  and the element vector of nodal values  $U_e$  (filled according to 2.28):

```
type=elements(e).type;
ne=Le(type).ne;                      % number of nodes in element
Etag=Le(type).tag;
Dne=ndof*ne;                          % number of nodal val in element
Xe=zeros(ne,DG);                     % init matrix of coordinates
Ue=zeros(Dne,1);                     % init vector of nodal values
connec=elements(e).nodes;            % gets element connectivity
pos=1;
for n=1:ne
    node=connec(n);
    Xe(n,:)=nodes(node).coor;         % fills with coordinates of node n
    Ue(pos:pos+ndof-1)=nodes(node).U; % fills nodal values of node n
    pos=pos+ndof;
end
mat=elements(e).mat;                  % gets element material
```

The `eval` command launches the correct routine which returns  $S_g$ . For instance, in linear elasticity  $S_g(g,:)$  is the stress at the  $g$ -th Gauss point

```
Sg=eval([Etag Atag 'Sg(Xe,material(mat,:),Ue)']); % at gauss points
```

The stresses computed at the Gauss points are extrapolated with the nodes and stored in  $S_n$ , where the vector  $S_n(n,:)$  holds the components evaluated at the  $n$ -th local node.

```
Sne=eval([Etag 'g2n(Xe,Sg)']); % extrapolates at nodes
```

The extrapolation routine depends on the element type. In the case of T3 elements, where stresses are constant over the element, the extrapolation reduces to the trivial identity operator. For a T6 element, stresses at the three Gauss points are interpolated by means of the linear functions

$$M_1(\mathbf{a}) = \frac{1}{3}(5a_1 - a_2 - a_3) \quad M_2(\mathbf{a}) = \frac{1}{3}(-a_1 + 5a_2 - a_3) \\ M_3(\mathbf{a}) = \frac{1}{3}(-a_1 - a_2 + 5a_3),$$

such that  $M_i$  has a unit value at the  $i$ -th Gauss point and vanish at the other Gauss points ( $M_i(\mathbf{a}_j) = \delta_{ij}$ ). The stresses extrapolated over the element are hence given by

$$\{\sigma(\mathbf{a})\} = \sum_{g=1}^3 M_g(\mathbf{a}) \{\sigma^{(g)}\}$$

As an additional example, in the case of a Q4 element, the shape function satisfying the conditions  $M_i(\mathbf{a}_j) = \delta_{ij}$  are

$$M_1(\mathbf{a}) = \frac{1}{4}(1 - \sqrt{3}a_1)(1 - \sqrt{3}a_2) \quad M_2(\mathbf{a}) = \frac{1}{4}(1 + \sqrt{3}a_1)(1 - \sqrt{3}a_2) \\ M_3(\mathbf{a}) = \frac{1}{4}(1 + \sqrt{3}a_1)(1 + \sqrt{3}a_2) \quad M_4(\mathbf{a}) = \frac{1}{4}(1 - \sqrt{3}a_1)(1 + \sqrt{3}a_2)$$

and the extrapolation can again be expressed as:

$$\{\sigma(\mathbf{a})\} = \sum_{g=1}^4 M_g(\mathbf{a}) \{\sigma^{(g)}\}.$$

The analysis of the Sg and g2n routines for each element type is left as an exercise. The output matrix Sne(ne, Sdim) contains, for a given element, the nodal values of the post-processing quantities.

The final nodal values are computed by simply taking the average of all the nodal quantities computed for each element in  $\Omega^{(k)}$

```

counter=zeros(analysis.NN,1);           % initializes "counter"
for e=1:analysis.NE,                     % for each element

    ... computes Sg and Sne ...

    Sn(convec,:)=Sn(convec,:)+Sne;        % adds to connec nodes
    counter(convec)=counter(convec)+1;    % increments counter
end
for icomp=1:Sdim
    Sn(:,icomp)=Sn(:,icomp)./counter;     % average of stresses
end

```

Finally, the routine genlin\_postgmsh prepares the output files containing displacements and stresses which are next given as input to GMSH. The details of the routine are examined in Section B.4.

### 3.7 Extensions and exercises

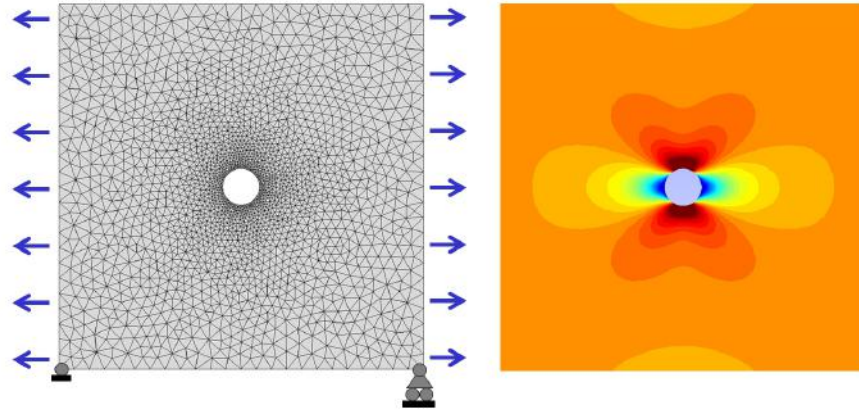
In the following sections the formulation discussed is extended to various applications, often presented in the form of exercises. In order to acquire an operative knowledge, the reader is strongly advised to become familiar with the codes and the pre- post-processor GMSH employed, discussed in Appendix B.

#### 3.7.1 Plate with a circular hole under plane strain conditions

The plate with a hole of Figure 3.8 is loaded with uniform tractions. This is a classical problem of elasticity theory, allowing to estimate the stress concentration caused by the hole. In the limiting case of a hole of radius  $R$  in an infinite plate, the closed form solution is available:

$$\begin{aligned}\sigma_{rr} &= \frac{\sigma}{2} \left(1 - \frac{R^2}{r^2}\right) + \frac{\sigma}{2} \left(1 + 3\frac{R^4}{r^4} - 4\frac{R^2}{r^2}\right) \cos 2\theta \\ \sigma_{\theta\theta} &= \frac{\sigma}{2} \left(1 + \frac{R^2}{r^2}\right) - \frac{\sigma}{2} \left(1 + 3\frac{R^4}{r^4}\right) \cos 2\theta \\ \sigma_{r\theta} &= -\frac{\sigma}{2} \left(1 - 3\frac{R^4}{r^4} + 2\frac{R^2}{r^2}\right) \sin 2\theta\end{aligned}$$

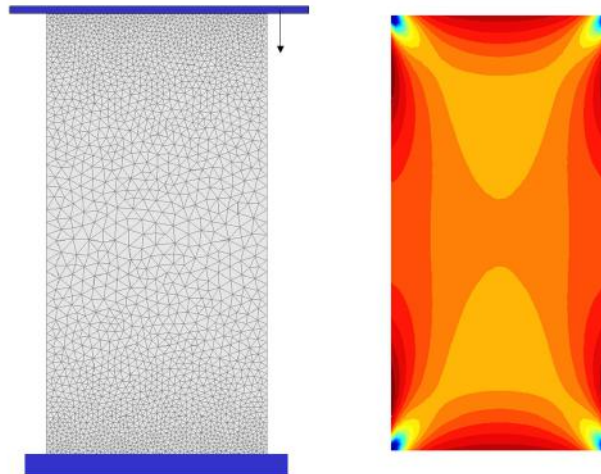
where  $\sigma$  is the traction applied at infinity in direction  $\mathbf{e}_1$ , and  $\theta$  is the angular polar coordinate ( $\theta = 0$  along the  $\mathbf{e}_1$  direction). For instance, the maximum hoop stress  $\sigma_{\theta\theta}$  near the hole boundary occurs for  $\theta = \pi/2$  and is  $\sigma_{\theta\theta} = 3\sigma$ .



**Figure 3.8:** Plate with a circular hole: mesh (left), stress  $\sigma_{11}$  (right)

**Exercise 3.5** (convergence analysis). *Using the files Chap3\_plate\_hole.\* available in the distribution, and modifying the radius, the element type and the level of mesh refinement, verify the convergence towards the analytical solution.*

### 3.7.2 Computation of reaction forces



**Figure 3.9:** Plate in compression and stress  $\sigma_{22}$

The rectangular bar of Figure 3.9 is compressed, under plane strain conditions, between two rigid plates. Self-weight is neglected and the lateral surface is traction-free. The lower plate is blocked while the upper one undergoes a vertical displacement  $-de_y$ . Friction between the plates and the bar prevents any tangential slip. The input file Chap3\_plate\_compress.\* are provided in the MATLAB distribution.

**Exercise 3.6** (reaction force). *The force exerted on the bar by the upper plate can be accurately computed resorting to the form (1.19) of the principle of virtual work. Show first that the choice  $\mathbf{w} = \mathbf{u}_h$  is admissible, and then obtain from (1.19) an explicit expression relating the reaction force to the strain energy.*

**Exercise 3.7** (potential energy and reaction force). *Analyse the procedure of Table 3.7 and show how the quantity work is related to the force exerted. Place the piece of code within `genlin.m` at the correct location and analyse the convergence rate as the mesh is refined. Next consider the total potential energy of the specimen and comment on how it is related to the quantity*

$$\frac{1}{2} \{\mathbf{U}\} [\mathbb{K}] \{\mathbf{U}\}$$

where  $\{\mathbf{U}\}$  is the vector of unknowns and  $\mathbb{K}$  is the global stiffness matrix.

```

impdisp=0.1; % imposed displacement (input required)
work=0.d0;
for e=1:analysis.NE, % stiffness matrix assemblage
    type=elements(e).type;
    ne=Le(type).ne; % number of nodes in element
    Etag=Le(type).tag;
    Dne=ndof*ne; % number of nodal values in one el.
    Ue=zeros(Dne,1);
    pos=1;
    for n=1:ne
        node=elements(e).nodes(n);
        Xe(n,:)=nodes(node).coor; % creates element
        Ue(pos:pos+ndof-1)=nodes(node).U;
        pos=pos+ndof;
    end
    mat=elements(e).mat;
    Ke=eval([Etag Atag 'Ke(Xe,material(mat,:))']);
    work=work+Ue'*Ke*Ue; % adds contrib to work
end
force=work/impdisp % estimate of force

```

**Table 3.7:** Procedure for computing the reaction force in exercise 3.7

### 3.7.3 Axisymmetric problems in linear elasticity

Let us now consider an axisymmetric structure like the scuba-tank depicted in Figure 3.10. A hollow cylinder with internal and external radii  $R_1$  and  $R_2$ , respectively, is closed by two hollow half-spherical shells having the same radii. The bottle is subjected to an internal pressure  $p$ . Since both the geometry and the loading are axisymmetric and the material is assumed to be isotropic, the solution maintains the same type of symmetry.

Working in cylindrical coordinates  $r, \theta, z$ ,  $u_\theta = 0$  and the displacement  $\mathbf{u}(r, z) = u_r(r, z)\mathbf{e}_r + u_z(r, z)\mathbf{e}_z$  generates a strain of the form:

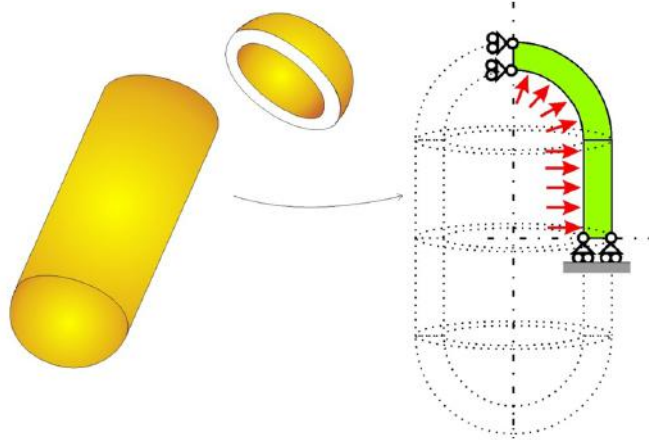
$$\begin{aligned}
 \boldsymbol{\varepsilon}(r, z) = & \frac{\partial u_r}{\partial r} \mathbf{e}_r \otimes \mathbf{e}_r + \frac{\partial u_z}{\partial z} \mathbf{e}_z \otimes \mathbf{e}_z + \frac{u_r}{r} \mathbf{e}_\theta \otimes \mathbf{e}_\theta \\
 & + \frac{1}{2} \left( \frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r} \right) (\mathbf{e}_r \otimes \mathbf{e}_z + \mathbf{e}_z \otimes \mathbf{e}_r)
 \end{aligned} \quad (3.55)$$

associated, through the linear constitutive law, with a stress tensor having the components  $\sigma_{rr}, \sigma_{\theta\theta}, \sigma_{zz}$  and  $\sigma_{z\theta}$  which all depend only on  $r$  and  $z$ . We now consider the weak formulation (3.3) and perform an analytical integration with respect to  $\theta$ . If  $S_h$  denotes the discretized generating 2D surface (i.e. the surface which generates the tank by a rotation of  $2\pi$  about the  $z$  axis), the problem can be formulated as follows (body forces being neglected for simplicity)

$$\text{Find } \mathbf{u}_h^{(0)} \in \mathcal{C}_h(\mathbf{0}) \text{ such that: } \forall \mathbf{w}(r, z) = w_r \mathbf{e}_r + w_z \mathbf{e}_z \in \mathcal{C}_h(\mathbf{0}), \quad (3.56)$$

$$\begin{aligned} \int_{S_h} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(0)}] : \mathcal{A} : \boldsymbol{\varepsilon}[\mathbf{w}] r \, dr \, dz = \\ - \int_{S_h} \boldsymbol{\varepsilon}[\mathbf{u}_h^{(D)}] : \mathcal{A} : \boldsymbol{\varepsilon}[\mathbf{w}] r \, dr \, dz + \int_{L_{T,h}} \mathbf{T}^D \cdot \mathbf{w} \, r \, ds \end{aligned}$$

where both sides have been divided by  $2\pi$  and  $L_{T,h}$  denotes the portion of the boundary curve where tractions are enforced. The integrals in (3.56) are now defined on a 2D surface in the  $r, z$  plane. This implies that, apart from the computation of element contributions which will be analysed in the sequel, the overall discretization/assembly/solution procedure is identical to what discussed in this chapter for 2D elasticity if the radial and axial coordinates and components (of indices  $r, z$ ) are interpreted as “generalized” coordinates and components of indices 1, 2. Axisymmetric analyses can be run with the general linear code `genlin` by setting the appropriate options as explained in Section 3.6.



**Figure 3.10:** Scuba tank: problem geometry and generating surface with symmetry conditions

In particular, employing isoparametric elements, the approximation of geometry and displacements is still given by (2.13) and (2.21). For instance, the radial coordinate is expressed in terms of its nodal values as:

$$r = \sum_{k=1}^{n_e} N_k(\mathbf{a}) r^{(k)} \quad \mathbf{x} \in E_e \quad (3.57)$$

Let us now address the virtual power of internal stresses which, as usual, is computed element by element exploiting additivity. Analogously to (3.9), the components of the

stress and strain tensor are collected in two arrays which now have four components:

$$\{\sigma\} = \{\sigma_{rr} \ \sigma_{zz} \ \sigma_{\theta\theta} \ \sigma_{rz}\}^T, \quad \{\varepsilon\} = \{\varepsilon_{rr} \ \varepsilon_{zz} \ \varepsilon_{\theta\theta} \ 2\varepsilon_{rz}\}^T. \quad (3.58)$$

with

$$\{\sigma\} = [A]\{\varepsilon\}, \quad (3.59)$$

where  $[A]$  is the  $4 \times 4$  symmetric matrix:

$$[A] = \begin{bmatrix} \lambda + 2\mu & \lambda & \lambda & 0 \\ \lambda & \lambda + 2\mu & \lambda & 0 \\ \lambda & \lambda & \lambda + 2\mu & 0 \\ 0 & 0 & 0 & \mu \end{bmatrix} \quad (3.60)$$

Next, on a given element,  $\{\varepsilon\}$  has to be defined in terms of the vector  $\{U_e\}$  collecting the radial and axial nodal displacements ordered according to the convention (2.22):

$$\{\varepsilon[u_h](x)\} = [B(\mathbf{a})]\{U_e\}, \quad (3.61)$$

The specific expression of  $[B(\mathbf{a})]$  depends on the choice of element type, and we will here limit ourselves to the quadratic triangle based on the T6 element addressed in Section 2.2.9. According to the definitions in (3.55), the lines of  $[B(\mathbf{a})]$  yielding the strain components  $\varepsilon_{rr}, \varepsilon_{zz}, 2\varepsilon_{rz}$  are exactly the same as those providing  $\varepsilon_{11}, \varepsilon_{22}, 2\varepsilon_{12}$  in Section 3.2.6. The additional component  $\varepsilon_{\theta\theta}$  can be expressed as

$$\varepsilon_{\theta\theta} = \frac{u_r}{r} = \left\{ \frac{N_1}{r}, 0, \frac{N_2}{r}, 0, \frac{N_3}{r}, 0, \frac{N_4}{r}, 0, \frac{N_5}{r}, 0, \frac{N_6}{r}, 0 \right\} \{U_e\}$$

from which the corresponding line of  $[B(\mathbf{a})]$  can be easily identified. The MATLAB function evaluating the stiffness matrix for the axisymmetric element is presented in Table 3.8.

Two remarks are worth emphasizing. Firstly, the requirement to integrate terms of the form  $N_i N_j / r$  might a priori raise singularity issues in elements having a geometric feature on the rotation axis  $r = 0$ . However, if the condition  $u_r = 0$  is explicitly enforced at the nodes with  $r = 0$ , the columns of  $[B(\mathbf{a})]$  associated with these nodal values are disregarded in the assemblage. Moreover it can be shown that the terms  $N_i / r$  multiplying other nodal values always have a finite limit on the rotation axis. Secondly, the three-point Gauss rule employed in 2D elasticity is no longer complete. Indeed, on a non-distorted element the jacobian  $r$  is linear in  $\mathbf{a}$ , the  $N_i N_j$  terms are polynomials of the fourth order but the occurrence of the  $1/r$  non-polynomial terms makes the notion of *complete integration* no longer applicable. A seven-point rule, which integrates exactly polynomials of order 5, could instead be implemented in the element-level routines. Even though it is expected to improve accuracy, it cannot evaluate exactly the entries of the stiffness matrix.

The analysis of the B3\_AX\_solid\_Fe\_surf function providing the element contribution to the right hand side vector is left as an exercise.

In order to obtain a benchmark solution, we recall two analytical results. The first concerns a hollow sphere subjected to an inner pressure  $p$  which undergoes a purely radial displacement (in a spherical reference system):

$$u_r = \frac{pR_1^3}{R_2^3 - R_1^3} \left( \frac{1 - 2\nu}{E} r + \frac{1 + \nu}{E} \frac{R_2^3}{2r^2} \right) \quad (3.62)$$

```

function Ke=T6_AX_solid_Ke(X,mate)

E=mate(1);
nu=mate(2);
A=E/((1+nu)*(1-2*nu))*[1-nu,nu,nu,0;           % stiffness tensor
                        nu,1-nu,nu,0;
                        nu,nu,1-nu,0;
                        0,0,0,(1-2*nu)/2];
a_gauss=1/6*[4 1 1; 1 4 1; 1 1 4];           % Gauss abscissae
w_gauss=[1/6 1/6 1/6];                       % Gauss weights
Ke=zeros(12,12);
for g=1:3,                                     % loop over Gauss points
    a=a_gauss(g,:);                           % param. coor. for gauss point

    ... G and detJ computed as in T6.2A.solid_Ke function ..

    N=[a(1)*(2*a(1)-1) a(2)*(2*a(2)-1) ...      % list of the 6 shape functions
        a(3)*(2*a(3)-1) 4*a(1)*a(2) ...
        4*a(2)*a(3) 4*a(1)*a(3)];
    r=N*X(:,1);                                % radial coordinate
    B=[G(1,1) 0 G(2,1) 0 G(3,1) 0 ...          % first line as in T6
        G(4,1) 0 G(5,1) 0 G(6,1) 0;
        0 G(1,2) 0 G(2,2) 0 G(3,2) ...        % second line as in T6
        0 G(4,2) 0 G(5,2) 0 G(6,2);
        N(1)/r 0 N(2)/r 0 N(3)/r 0 ...        % new line for eps_tt
        N(4)/r 0 N(5)/r 0 N(6)/r 0;
        G(1,2) G(1,1) G(2,2) G(2,1) G(3,2) G(3,1)... % last line as in T6
        G(4,2) G(4,1) G(5,2) G(5,1) G(6,2) G(6,1)];
    Ke=Ke+B'*A*B*detJ*w_gauss(g)*r;           % contrib to stiffness matrix
end

```

**Table 3.8:** Computation of the element stiffness matrix for axisymmetric analyses

where  $R_1$  and  $R_2$  are the inner and outer radii, respectively. The second concerns a hollow cylinder of height  $2H$  subjected to an inner pressure  $p$  and to a uniform traction  $\pm T e_z$  on the upper and lower surfaces, respectively. The closed form solution in a cylindrical reference system is

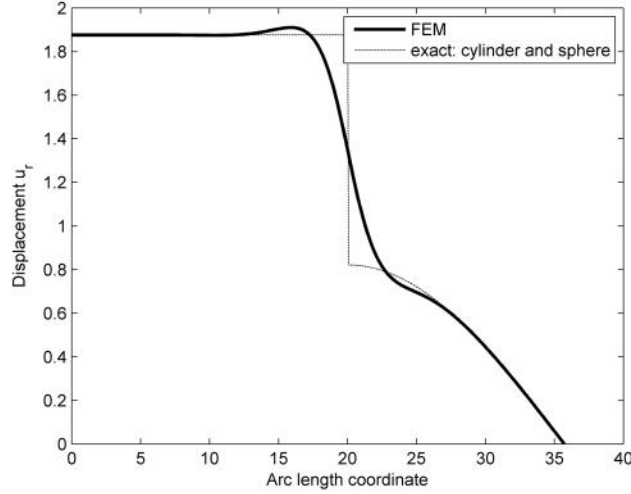
$$u_r = \frac{pR_1^2}{R_2^2 - R_1^2} \left( \frac{1-\nu}{E} r + \frac{1+\nu}{E} \frac{R_2^2}{r} \right) - \nu \frac{T}{E} r, \quad u_z = \frac{T}{E} z \quad (3.63)$$

where  $R_1$  and  $R_2$  are again the inner and outer radii. The comparison of the numerical results with these analytical solutions in the case of an isolated sphere or of an isolated cylinder is proposed as an exercise.

Let us now consider the complete scuba tank. Each cap exerts on the cylinder a force with resultant  $pR_1^2$  and we fix  $T = pR_1^2/(R_2^2 - R_1^2)$  in (3.63) in order to respect global equilibrium. However, the analytical solutions for the radial displacement in the cylinder and in the caps differ at the interface. In the tank, a self-equilibrated distribution of tractions will arise at the interface in order to restore compatibility. However, thanks to the Saint Venant principle, the numerical solution should converge to (3.62)-(3.63) at a sufficient distance from the interface,



The code `gen1in` with input files `Chap3.scubatank.*` ( $R_1 = 10$ ,  $R_2 = 10.5$ ,  $H = 20$ ) is employed to compare the radial displacement of the inner surface of the bottle with the prediction of (3.62)-(3.63) with  $T = pR_1^2/(R_2^2 - R_1^2)$ . The results are presented in Figure 3.11 where the radial displacement in a cylindrical reference system is plotted versus an arc-length coordinate running on the inner surface of the bottle from  $z = 0$  (symmetry plane of the cylinder) up to the  $r = 0$  axis.



**Figure 3.11:** Scuba tank under internal pressure: comparison of FEM solution with the exact solutions obtained for an isolated sphere and an isolated cylinder with equivalent loadings

### 3.7.4 Bulk modulus of an heterogeneous material in axisymmetric conditions

A solid of volume  $V$  is made of a linear elastic isotropic heterogeneous material. A pressure  $p$  is applied everywhere on the external surface and induces the volume variation  $\Delta V$ . The bulk modulus is defined as the ratio  $\kappa_a = -p/(\Delta V/V)$ . In particular, if the material is homogeneous:

$$\kappa_a = \kappa = \frac{E}{3(1-2\nu)} = \frac{3\lambda + 2\mu}{3} \quad (3.64)$$

In the case of an heterogeneous material let us denote by  $\lambda(\mathbf{x})$ ,  $\mu(\mathbf{x})$ ,  $E(\mathbf{x})$ ,  $\nu(\mathbf{x})$  the elastic coefficients, with

$$3\kappa(\mathbf{x}) = 3\lambda(\mathbf{x}) + 2\mu(\mathbf{x}) = \frac{E(\mathbf{x})}{1-2\nu(\mathbf{x})}$$

An upper bound for the bulk modulus can be obtained analytically using classical variational approaches. The displacement  $\mathbf{v} = \alpha \mathbf{x}$  is kinematically admissible and the optimal value of  $\alpha$  can be obtained by minimising the functional of total potential energy:

$$\mathcal{W}(\mathbf{v}) - \mathcal{F}(\mathbf{v}) = \frac{9}{2} \alpha^2 V \langle \kappa(\mathbf{x}) \rangle + 3\alpha p V \quad (3.65)$$

The minimisation with respect to  $\alpha$  gives

$$\alpha = -\frac{p}{3 \langle \kappa(\mathbf{x}) \rangle} \quad \text{where} \quad \langle \kappa(\mathbf{x}) \rangle = \frac{1}{V} \int_{\Omega} \kappa(\mathbf{x}) \, dV \quad (3.66)$$

Finally, using the weak form of the balance equation (1.19) with  $\mathbf{w} = \mathbf{u}$  (the exact solution), one obtains (Clapeyron theorem)

$$\mathcal{W}(\mathbf{u}) = \frac{1}{2} \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{u}] \, dV = \frac{1}{2} \int_{\partial\Omega} \mathbf{T} \cdot \mathbf{u} \, dS = \frac{1}{2} \int_{S_T} \mathbf{T}^D \cdot \mathbf{u} \, dS = \frac{1}{2} \mathcal{F}(\mathbf{u}).$$

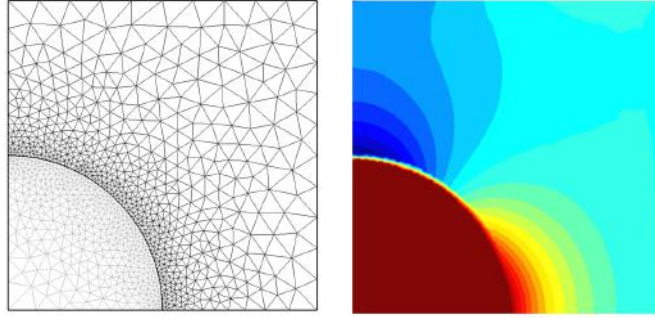
The total potential energy at solution then reads,

$$\mathcal{W}(\mathbf{u}) - \mathcal{F}(\mathbf{u}) = -\frac{p}{2} \int_{\partial\Omega} \mathbf{u} \cdot \mathbf{n} \, dS = -\frac{p}{2} \Delta V = -p^2 \frac{V}{2\kappa_a},$$

leading to the following upper bound:

$$\kappa_a \leq \langle \kappa(\mathbf{x}) \rangle \quad (3.67)$$

In the case of a homogeneous material of bulk modulus  $\kappa$  with inclusions of a different material of bulk modulus  $\kappa_f$ , the upper bound becomes  $\langle \kappa(\mathbf{x}) \rangle = (1 - f)\kappa + f\kappa_f$ , where  $f$  is the volume percentage of the inclusions.



**Figure 3.12:** Bulk modulus of a cylinder with a spherical inclusion: axisymmetric mesh (exploiting symmetry with respect to  $z$  and contour plot of  $\sigma_{rr}$  when  $E_f = 0.1E$ )

In this exercise we aim at improving the analytical upper bound using the finite element method in the specific case of a cylinder with a spherical inclusion centred at the origin. Geometry and analysis files `Chap3_bulkmodulus.*` are available in the distribution. Axial symmetry is exploited and moreover only half of the generating surface is described, as depicted in Figure 3.12. The code `gen1.in` can be employed to obtain an estimate of  $\Delta V/V$ . Indeed using (1.21) with  $\mathbf{w} = \mathbf{u}_h$

$$\int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}_h] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{u}_h] \, dV = \int_{S_T} \mathbf{T}^D \cdot \mathbf{u}_h \, dS = -p \int_{S_T} \mathbf{u}_h \cdot \mathbf{n} \, dS = -p \Delta V_h \quad (3.68)$$

which is indeed very similar to the formula implemented for the computation of reaction forces in Section 3.7.2.

**Exercise 3.8** (effective bulk modulus for a heterogeneous material). *Implement formula (3.68) in the code. How can the computed  $\Delta V_h$  be employed to obtain an upper bound for  $\kappa_a$ ? How does this compare with the analytical estimate (3.67) for a generic mesh? Plot the bounds for  $\kappa_f \rightarrow 0$ .*

### 3.7.5 Equivalent Young modulus for a cylinder under compression

A cylinder of length  $L$  with a circular cross section of area  $S$  is made of a linear elastic isotropic and homogeneous material. The specimen is subjected to the same boundary conditions as for the 2D rectangular case of Section 3.7.2. Let us define the equivalent Young modulus  $E_a = QL/S\delta$ , where  $Q$  is the force exerted on the specimen by the rigid plates and  $\delta$  is the vertical displacement imposed on the upper surface.

Using the variational tools recalled in Chapter 1, and selecting the kinematically admissible displacement  $\mathbf{v} = -\delta(z/L)\mathbf{e}_z$ , one obtains the upper bound:

$$E_a \leq E \frac{(1 - \nu)}{(1 - 2\nu)(1 + \nu)}$$

The compression of the cylinder can be simulated in `genlin` exploiting axial symmetry and the total force exerted on the upper surface can be obtained in a post-processing phase by slightly modifying the procedure of Section 3.7.2.

**Exercise 3.9** (effective Young's modulus for a cylinder under compression). *Obtain a numerical estimate for the equivalent Young modulus and show that is an upper bound for  $E_a$  and that it necessarily improves the analytical one. Plot the upper bounds for different  $L/R$  values, where  $R$  is the radius of the cross section*

### 3.7.6 Cook's wing combining T6 and Q8 elements. Adding a new element to `genlin`

We consider the classical benchmark problem (Cook's wing) depicted in the top of Figure 3.13. The membrane, under plane stress, is clamped on the left and subjected to a uniform traction on the right. The details of the geometry and loading are collected in the input files `Chap3_wing.*`. Several reference solutions are available. They predict a vertical displacement  $u_2 = 23.91$  (in units consistent with the data in the input files) at the middle of the right edge.

Proceeding with standard options, GMSH produces a mesh of triangular elements as in Figure 3.13 (bottom left). Quadrangles can be introduced in several ways. A possibility is to modify mesh options from the *Tools* → *Options* → *Mesh* → *General* window. If *Delaunay for quads* is selected as *2D algorithm* and the *Recombine all triangular meshes* option is activated, the mesh on the left in Figure 3.13 is obtained. Both triangular and quadrangular elements are employed and `genlin` handles mixed elements in a mesh. It is worth stressing that a mesh completely made of quadrangles can be forced by selecting the *All Quads* option as *Subdivision algorithm* (Fig. 3.13, bottom right). Both linear and quadratic elements can be created by setting the appropriate choices in the *Module Menu*. By default Q9 quadrangles are selected, but if Q8 quadrangles are preferred, the option *Use incomplete elements* must be activated. Neither Q8 nor Q9 elements are available in the `genlin` distribution. We propose here, as an exercise, to develop the steps required to add the Q8 element.

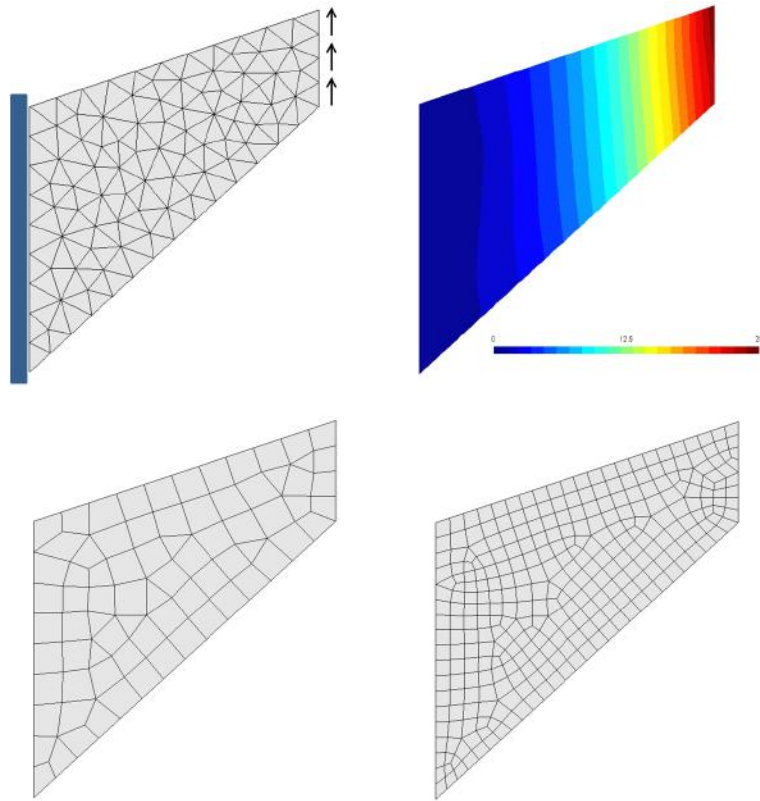
The only required modification to the main code `genlin.m` consist in adding a line to the `Le` list of element definitions:

```
Le(16)=struct('tag','Q8_','ne',8,'c',5);
```

It is worth recalling that the list `Le` is filled in an order which reflects the numbering assigned by GMSH to the different element types and the Q8 quadrangle is indeed the 16th element.

Next the element-level subroutines must be created and placed in the directory subQ8. The required functions are Q8\_2S\_solid\_Ke.m, computing the element stiffness matrix (already proposed as an exercise in Section 3.2.9); Q8\_2S\_solid\_Sg.m, computing stresses at Gauss points; Q8\_g2n.m, extrapolating to nodes the stresses computed at Gauss points.

No action needs to be taken for the subroutines computing nodal loads due to imposed boundary tractions. Since the edge of a Q8 element as well as a T6 element is a B3 element, the same B3\_2S\_solid\_Fe\_surf function of Sec. 3.2.8 can be employed.



**Figure 3.13:** Cook's wing benchmark problem. Geometry and contour plot of  $u_2$  (top); meshes with mixed triangles and quadrangles (bottom left) or quadrangles only (bottom right)

### 3.7.7 Example of scalar problem: warping of a beam section in torsion

All the procedures developed so far can be easily adapted to scalar problems. An interesting example is the analysis of the warping problem associated with the torsion of a beam with arbitrary cross section.

The problem can be formulated over the cross section of the beam using a stress approach or a displacement approach. We will develop here the former one, leaving the latter as an exercise. Focusing on a simply connected cross section  $S$  in the  $x - y$

plane, a stress function  $\varphi$  can be defined such that:

$$\sigma_{zx} = \frac{\partial \varphi}{\partial y} \quad \sigma_{zy} = -\frac{\partial \varphi}{\partial x}, \quad \mathbf{x} \in S$$

The strong form of the problem, stemming from the geometrical compatibility conditions, reads:

$$\operatorname{div} \nabla \varphi = -c = -2\mu\beta, \quad \mathbf{x} \in S, \quad \varphi = 0 \quad \mathbf{x} \in \partial S \quad (3.69)$$

where  $\beta$  denotes the rotation per unit length that the beam undergoes due to the applied torque. Introducing a scalar test function  $w \in \mathcal{C}(0)$  (which vanishes on  $\partial S$ ) the problem can be formulated in weak form after integrating by parts and applying the divergence theorem:

$$\text{find } \varphi \in \mathcal{C}(\varphi^D) \text{ such that, } \forall w \in \mathcal{C}(0): \quad \int_S \nabla \varphi \cdot \nabla w \, dS = c \int_S w \, dS \quad (3.70)$$

The finite-dimensional approximation space for (3.70) is defined in terms of isoparametric finite elements similar to those used for linear elasticity, the only difference being that nodal values employed in the interpolation of  $\varphi$  and  $w$  are scalar quantities. For every element  $E_e$ :

$$\varphi_h(\mathbf{x}) = \sum_{k=1}^{n_e} N_k(\mathbf{a}) \varphi^{(k)}, \quad w(\mathbf{x}) = \sum_{k=1}^{n_e} N_k(\mathbf{a}) w^{(k)}, \quad \mathbf{x} \in E_e$$

so that, exploiting additivity, the different integrals in (3.70) are evaluated by adding the element contributions.

$$\sum_{e=1}^{N_E} \int_{E_e} \nabla \varphi_h \cdot \nabla w \, dS = \sum_{e=1}^{N_E} \int_{E_e} c w \, dS \quad (3.71)$$

Indeed, on a given element mapped onto the parametric space, the terms on the right hand side of (3.70) are analogous to spatially constant body forces:

$$\int_{E_e} c w \, dS = \{W_e\}^T \left( \int_{\Delta_e} c [N(\mathbf{a})]^T J(\mathbf{a}) \, da_1 \, da_2 \right) = \{W_e\}^T \{F_e\}$$

For a T6 element, and with the help of the usual techniques of numerical integration, this gives

```
function Fe=T6_2D_therm_Fe(X, val)

a_gauss=1/6*[4 1 1; 1 4 1; 1 1 4];           % Gauss abscissae
w_gauss=[1/6 1/6 1/6];                       % Gauss weights
Fe=zeros(6,1);
for g=1:3,                                    % loop over Gauss points

    .. evaluate detJ as for Ke ...

    N=[a(1)*(2*a(1)-1) a(2)*(2*a(2)-1) ...    % list of shape functions
        a(3)*(2*a(3)-1) 4*a(1)*a(2) ...
        4*a(2)*a(3) 4*a(1)*a(3)]';
    Fe=Fe+val*detJ*N*w_gauss(g);              % contribution to rhs
end
```

Similarly, employing the notation of (2.37), the left-hand side of (3.70) becomes

$$\begin{aligned} \int_{E_e} \nabla \varphi_h \cdot \nabla w \, dS &= \{W_e\}^T \left( \int_{\Delta_e} [G(\mathbf{a})]^T [G(\mathbf{a})] J(\mathbf{a}) \, da_1 \, da_2 \right) \{U_e\} \\ &= \{W_e\}^T [K_e] \{U_e\} \end{aligned}$$

which can also be evaluated with numerical integration. For instance, the element stiffness matrix for a T6 element is a simplified version of the stiffness matrix in linear elasticity.

```
function Ke=T6_2D_therm_Ke(X,mate)

k=mate(1);
a_gauss=1/6*[4 1 1; 1 4 1; 1 1 4]; % Gauss abscissae
w_gauss=[1/6 1/6 1/6]; % Gauss weights
Ke=zeros(6,6);
for g=1:3, % loop over Gauss points
    a=a_gauss(g,:); % param. coordinates for gauss point
    D=[4*a(1)-1 0 -4*a(3)+1 4*a(2)... % derivative of shape functions...
        -4*a(2) 4*(a(3)-a(1)); % w.r.t. a_1,a_2
        0 4*a(2)-1 -4*a(3)+1 4*a(1) ...
        4*(a(3)-a(2)) -4*a(1)]';
    J=X'*D; % jacobian matrix
    detJ=J(1,1)*J(2,2)-J(1,2)*J(2,1); % jacobian
    invJ=1/detJ*[ J(2,2) -J(1,2); ... % inverse jacobian matrix
        -J(2,1) J(1,1)];
    G=(D*invJ)'; % gradient of shape functions
    Ke=Ke+k*G'*G*detJ*w_gauss(g); % contribution to stiffness matrix
end
```

It is worth emphasizing that T6\_2D\_therm\_Ke also introduces a “conductivity” coefficient  $k$ , in view of its use for heat conduction problems (with  $k = 1$  for the application at hand).

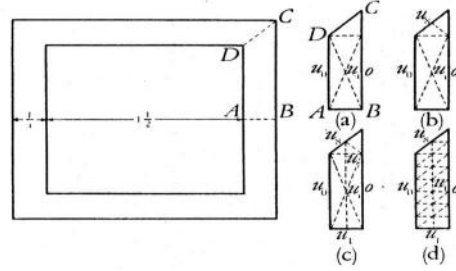
While the assemblage of the stiffness matrix does not require any modification with respect to what described in Section 3.6.4, the body force terms are not accounted for in `genlin` but can be assembled with the following two lines positioned within the loop for the stiffness matrix.

```
Fe=eval([Etag Atag 'Fe(Xe,bf)']);
F(Ie)=F(Ie)+Fe(Le0);
```

Two input files, differing by the shape of the beam cross section, are provided with the codes, namely Chap3\_warp\_ellipse.\* and Chap3\_warp\_square.\*. It is left as an exercise to compare the results of the code (both with T3 and T6 elements) with the analytical solution of the former case, which reads, when  $c = 1$ :

$$\varphi = \frac{a^2 b^2}{2(a^2 + b^2)} \left( 1 - \frac{x^2}{a^2} - \frac{y^2}{b^2} \right)$$

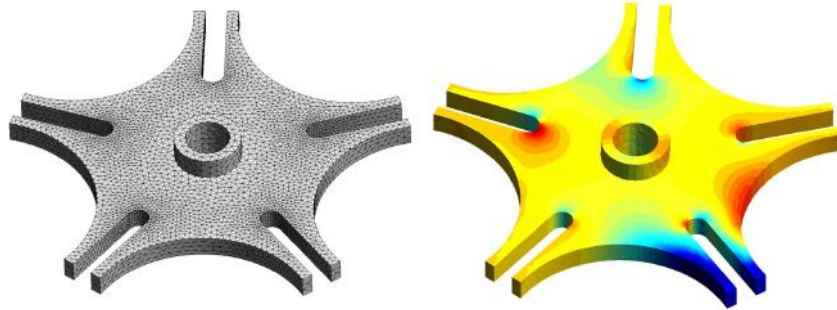
**Remark.** The idea of employing a mesh made of triangles with linear interpolation of the unknown field dates back to a paper by Courant (1943). This is one of the earliest references in which the concept of finite element appears (Fig. 3.14).



**Figure 3.14:** Numerical solution of the warping problem associated with the torsion of a beam with arbitrary cross section: the cross section is partitioned into triangles and the warping function is assumed linear over each element (picture taken from Courant (1943)).

### 3.7.8 Exercise: heat conduction in 3D

The mechanical part depicted in Figure 3.15 is created using Chap3\_heat3D.geo a geometry file taken from the GMSH distribution. Linear tetrahedra P4 are employed in the mesh. The analysis file Chap3\_heat3D.m sets `analysis.type=3`, which denotes a 3D heat conduction case. The analysis of the input files and of the required element-level routines is left as an exercise as well as the exploration of the post-processing capabilities of GMSH in 3D (e.g., clipping, cutting surfaces).



**Figure 3.15:** Thermal analysis of a mechanical part. FEM mesh adopted and contour plot of one component of the gradient

### 3.7.9 Project: capacitance extraction in electrostatics

The code `gen1n` lends itself to a variety of customizations. As an example we develop herein a simple project for the evaluation of the capacitance matrix in electrostatics.

A piezoelectric specimen has a rectangular cross section  $S_P$  as represented in Figure 3.16. Metallic tracks (electrodes) have been deposited on the upper surface of the piezo and can be treated as perfect conductors. A perfect conductor by definition is characterized by a uniform potential. A potential  $\varphi = \varphi_1$  is imposed on odd electrodes (light grey in Figure 3.16), while even electrodes are grounded with  $\varphi = \varphi_2$  (dark grey

in Figure 3.16). Let  $S_1$  and  $S_2$  denote the surfaces of two sets of electrodes. The surrounding air  $S_A$  must be included in the simulation. In order to simplify the treatment of conditions at infinity, we will admit that only a portion of air is considered by introducing, at a sufficient distance from the piezo, a closed truncation line  $\Gamma_A$  where the natural condition  $\nabla\varphi \cdot \mathbf{n} = 0$  is assumed.

The differential problem governing the electrostatic potential  $\varphi$  is:

$$\operatorname{div}(\varepsilon_r(\mathbf{x})\varepsilon_0\nabla\varphi(\mathbf{x})) = 0, \quad \mathbf{x} \in S = S_A \cup S_P \quad (3.72a)$$

$$\varphi(\mathbf{x}) = \varphi_1, \quad \mathbf{x} \in \partial S_1 \quad (3.72b)$$

$$\varphi(\mathbf{x}) = \varphi_2, \quad \mathbf{x} \in \partial S_2 \quad (3.72c)$$

$$\nabla\varphi(\mathbf{x}) \cdot \mathbf{n} = 0, \quad \mathbf{x} \in \Gamma_A \quad (3.72d)$$

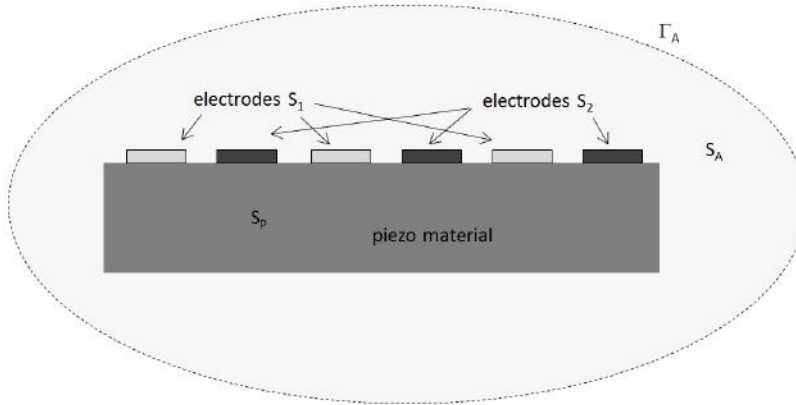
$\varepsilon_0$  is the vacuum permittivity and  $\varepsilon_r$  is the relative permittivity which has unit value in air and a constant value  $\bar{\varepsilon}_r$  in the piezo. Recalling Section 1.6, the problem can be formulated in the weak form:

$$\text{find } \phi \in \mathcal{C}(\varphi^D) \text{ such that, } \forall w \in \mathcal{C}(0): \int_S \varepsilon_r(\mathbf{x}) \nabla\varphi(\mathbf{x}) \cdot \nabla w(\mathbf{x}) \, dS = 0 \quad (3.73)$$

where the space  $\mathcal{C}(\varphi^D)$  is the set of continuous functions which satisfy conditions (3.72b,c). The space  $\mathcal{C}(0)$  is the set of continuous functions  $w(\mathbf{x})$  such that:

$$w(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial S_1 \cup \partial S_2$$

It is worth stressing that the problem is totally analogous to a heat conduction analysis and can be hence treated in the context of the `genlin` code with no modifications. Let us suppose that the domain is meshed with T3 or T6 triangles. All the required element-level subroutines are already available in the standard distribution.



**Figure 3.16:** Cross section of the piezo material

**Capacitance extraction.** Electrical charges  $Q_1, Q_2$  accumulate on each family of electrodes. The total charge on a conductor due to a generic potential distribution can



```

cap12=0.0;d0;
for e=1:analysis.NE, % stiffness matrix assemblage
    type=elements(e).type;
    ne=Le(type).ne; % number of nodes in element
    Etag=Le(type).tag;
    Dne=ndof*ne; % number of nod. val. in one el.
    Ue=zeros(Dne,1);
    We=zeros(Dne,1);
    pos=1;
    for n=1:ne
        node=elements(e).nodes(n);
        Xe(n,:)=nodes(node).coor; % creates element
        Ue(pos:pos+ndof-1)=nodes(node).U; % creates Ue
        if ( nodes(node).dof(1)<0 & nodes(node).U(1)==0 )
            We(pos:pos+ndof-1)=1; % creates We
        end
        pos=pos+ndof;
    end
    mat=elements(e).mat;
    Ke=eval(['Etag Atag 'Ke(Xe,material(mat,:))']);
    cap12=cap12+We'*Ke*Ue; % adds to total capacitance
end

```

For every element the procedure builds  $U_e$  collecting nodal values of  $\varphi_h^{(2)}$  and  $W_e$  collecting nodal values of  $w^{(2)}$ . The nodal values of  $w^{(1)}$  are set to zero everywhere but for the nodes of  $\partial S_1$  where  $w^{(1)} = 1$ .

**Exercise 3.10** (computation of a capacitance component). *Modify the procedure above so as to evaluate  $C_{22}$*

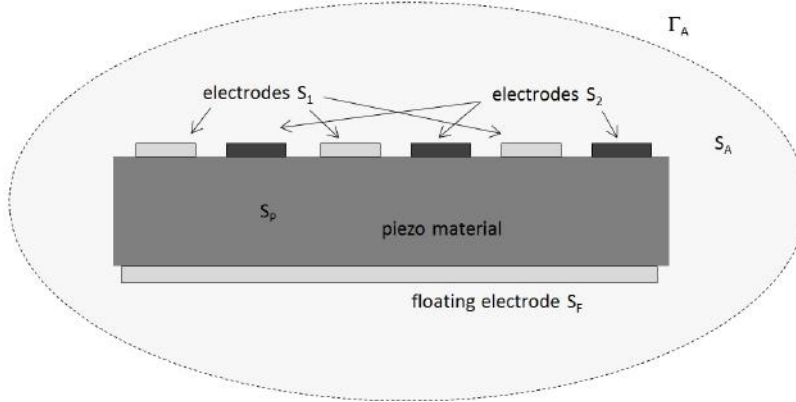
Now, let  $\varphi_h^{(1)}(\mathbf{x})$  denote the solution of problem (3.73) when the electrode potentials are  $\varphi_1 = 1$  and  $\varphi_2 = 0$ . From the above relationships, and remarking that one can chose  $w^{(1)} = \varphi_h^{(1)}$  and  $w^{(2)} = \varphi_h^{(2)}$ :

$$\begin{aligned} C_{12} &= \int_{\partial S_1} \varepsilon_r(\mathbf{x}) \nabla \varphi_h^{(2)}(\mathbf{x}) \cdot \mathbf{n} \, ds = \int_S \varepsilon_r(\mathbf{x}) \nabla \varphi_h^{(1)}(\mathbf{x}) \cdot \nabla \varphi_h^{(2)}(\mathbf{x}) \, dS \quad (3.76) \\ &= \int_{\partial S_2} \varepsilon_r(\mathbf{x}) \nabla \varphi_h^{(1)}(\mathbf{x}) \cdot \mathbf{n} \, ds = C_{21} \end{aligned}$$

Hence the capacitance matrix is symmetric.

**Exercise 3.11** (capacitance matrix). *Complete the computation of the capacitance matrix.*

**Floating electrode.** In many configurations a further electrode (of surface  $S_F$ ) is deposited on the bottom of the piezo material (see Figure 3.17).



**Figure 3.17:** Cross section of the piezo material with floating electrode

The electrode is said to be floating, in the sense that it is not connected to voltage sources and its voltage  $\varphi(\mathbf{x}) = \varphi_F$  is unknown. Equations (3.72a)-(3.72d) still hold. Every function  $v$  of  $\mathcal{C}(\varphi^D)$  and  $\mathcal{C}(0)$  must respect the condition:

$$v(\mathbf{x}) = v_F \quad \forall \mathbf{x} \in \partial S_F$$

with  $v_F$  unknown, which simply states that all the points on the contour of the floating electrode are connected to the same unknown potential. In particular it is worth stressing that a  $w \in \mathcal{C}(0)$  is not forced to vanish on  $\partial S_F$ .

**Exercise 3.12.** Show that the weak form of the problem is still given by (3.73), provided that we assume the further condition that the total charge on  $S_F$  vanishes

We will now show how to rapidly implement some *ad-hoc* (not general) changes which allow to account for the presence of the floating electrode. In order to be able to identify the nodes belonging to  $\partial S_F$ , the GMSH file Chap3\_capacitance.geo defines the boundary of the floating electrode as a physical set (physical set number 1). In the file Chap3\_capacitance.m, a part from standard definitions, the variable `constr = [1]` is introduced to define the constraint that all the nodes belonging to the physical set `constr(1)` will have the degree of freedom associated with the same global unknown.

A fast (though not general) way to enforce this is to impose, in the input-reading phase, that all the nodes on  $\partial S_F$  will be associated with the first global unknown. This can be done by adding the following lines at the end of the first loop over elements in `readgmsh`:

```

for i=1:num,                                % loop over the elements
    tline = fgets(fmid);
    h=sscanf(tline,'%d');
    physet=h(4);                            % physical set
    pos=find(solid(:,1)==physet);
    ...
    pos=find(tbc(:,1)==physet);
    ...
    pos=find(dbc(:,1)==physet);
    ...
    pos=find(constr(:)==physet);            % if the physet appear in constr
    for il=1:length(pos)
        for n=6:length(h)
            nodes(h(n)).dof(1)=1;          % all the nodes of the element
            % have dof=1
        end
    end
end
end

```

It is worth recalling that in standard analyses with no constraints, in `readgmsh` sets `nodes(k).dof(m)` to 0 if the  $m$ -th degree of freedom is unknown and to -1 otherwise. After closing the input reading phase, `genlin`, before defining the global numbering of other unknowns, sets the initial value of `neq` to one if the `constr` variable is active:

```

neq=0; % floating electrode
if exist('constr')
    neq=1;
end

```

The procedure next continues as usual a part from a necessary change in the assemblage phase since the command line

```
K(Ie,Ie)=K(Ie,Ie)+Ke(Le0,Le0);
```

fails if the same integer appears more than once in `Ie`. All the elements having two nodes on the boundary of the floating electrode will generate a list `Ie` with multiple entries of the global number of the unknown associated with the floating electrode (i.e.

1). The loops in the assemblage must be hence performed explicitly:

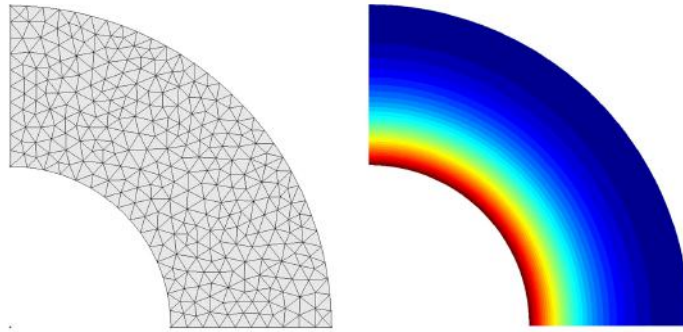
```

for i1=1:length(Ie)
    p=pe(i1);                % local number of unknown
    I=Ie(i1);                % global number of unknown
    for i2=1:length(Ie)      % loop over all columns in row I
        q=pe(i2);           % loc num of unknown in col i2
        J=Ie(i2);           % glob num of unknown in col i2
        K(I,J)=K(I,J)+Ke(p,q);
    end
end
end

```

### 3.7.10 Exercise: numerical experiments on convergence

Let us consider a hollow cylinder with axis parallel to  $z$  with inner and outer radii  $R_1$  and  $R_2$  subjected to the internal pressure  $p$ . Since the exact analytical solution predicts  $\sigma_{zz} = 0$ , the problem is solved as a 2D case formulated on the cross-section of the cylinder under the assumption of plane stress.



**Figure 3.18:** Hollow cylinder: problem geometry and radial displacement plot

An example of input file which exploits symmetry and models only one quarter of the cross section is provided in Chap3\_hollowcyl.\* where the mesh is uniform and the element size is governed by the parameter  $h$ . This is a classical problem of linear elasticity which can be solved analytically in polar coordinates. The solution is provided by (3.63) with  $T = 0$ . Employing both T3 and T6 elements obtain convergence plots in terms of  $h$  for the  $L^2$  error norms of the radial displacement and of the radial stress and compare with the predictions in (3.45) and (3.46).

## Transient heat conduction and linear thermoelasticity

---

This chapter addresses the finite element solution of transient heat conduction and uncoupled linear thermoelasticity. Thermal loads and responses are time-dependent due to heat diffusion in the solid. Much of the chapter is devoted to solving evolutive heat diffusion problems. More generally, the methods presented herein pertain to solving many other evolutive problems whose governing equations involve first-order time derivatives. The case of thermal equilibrium, where the thermal analysis reduces to the Laplace equation with appropriate conditions, can be solved using formulations that are very similar to (and in fact simpler than) elastic equilibrium (see Section 1.6); it is therefore left out of this chapter.

The second part of this chapter considers the numerical solution of uncoupled linear thermoelastic problems, where the temperature at any given time is used as input for a mechanical equilibrium problem. This typically corresponds to cases where the mechanical response of a solid (here assumed to fall within the small deformation assumption) results from the incompatibility of strains that arise due to thermal dilatation. Since heat diffusion usually proceeds in slow and progressive fashion, it is often reasonable to assume, as here, a quasi-static mechanical response even though the heat conduction is considered as transient.

The main components of solution methods for transient heat conduction problems, namely the spatial semi-discretization using finite elements and the integration in time of the system of differential equations thus obtained are first addressed (Sections 4.1 and 4.2). Then, linear thermoelastic problems, their variational formulation and numerical solution are described in Section 4.3. Illustrations, including hands-on examples and an application to the analysis of a car motor part, are shown along the way.

### 4.1 Transient heat conduction: finite element semi-discretization

This section focuses on the numerical solution of time-dependent heat conduction problems. In addition to its usefulness for thermomechanical analyses, this topic is obviously important in its own right. For a comprehensive exposition of heat transfer in solids, see the well-known monograph by Carslaw and Jaeger (1959).

In addition, this class of problems conveniently allows to introduce the reader to numerical solution methods for evolution equations. The case of thermal equilibrium, which is similar to (and in fact simpler than) linear elastostatics, was addressed in Section 1.6 and is not included in this chapter.

#### 4.1.1 Governing equations of heat conduction

**Field equations.** Consider a heat conducting material occupying the domain  $\Omega$ , characterized by its mass density  $\rho$ , specific heat  $c$  and isotropic thermal conductivity  $k$ . The heat conduction model consists of the transient heat equation for the temperature field  $T(\mathbf{x}, t)$ :

$$\rho c \frac{\partial T}{\partial t}(\mathbf{x}, t) - \operatorname{div}(k \nabla T)(\mathbf{x}, t) = g(\mathbf{x}, t) \quad (\mathbf{x} \in \Omega, t \in [0, t^F]), \quad (4.1)$$

where  $[0, t^F]$  is the time interval over which the temperature evolution is sought and  $g(\mathbf{x}, t)$  is an internal heat source which is here considered as given (in coupled thermomechanical models, heat sources may have mechanical origin, e.g. plastic dissipation). Equation (4.1) results from combining the balance of internal energy

$$\operatorname{div} \mathbf{q}(\mathbf{x}, t) + \rho c \frac{\partial T}{\partial t}(\mathbf{x}, t) = g(\mathbf{x}, t) \quad (\mathbf{x} \in \Omega, t \in [0, t^F]) \quad (4.2)$$

(with  $\mathbf{q}$  denoting the inward heat flux) and the Fourier law (thermal constitutive relation)

$$\mathbf{q}(\mathbf{x}, t) = -k \nabla T(\mathbf{x}, t). \quad (4.3)$$

**Boundary conditions.** Equation (4.1) must be supplemented with a boundary condition at each point of the boundary  $\partial\Omega$ . Only two types of boundary conditions are considered here, where either the heat flux or the temperature is prescribed at any given boundary point (other types of boundary conditions exist, such as a heat transfer condition with the external medium postulating a given, e.g. linear, dependence of the heat flux on the difference between the body and external temperatures). These boundary conditions have the form

$$-k \nabla T \cdot \mathbf{n} = q^D \quad \text{on } S_q \times [0, t^F] \quad T = T^D \quad \text{on } S_{th} \times [0, t^F]. \quad (4.4)$$

with  $S_{th} \cap S_q = \emptyset$ ,  $S_{th} \cup S_q = \partial\Omega$  and where  $T^D$  and  $q^D$  are prescribed values of temperature and flux, respectively.

**Initial conditions.** Finally, initial conditions must be given, by prescribing the temperature field at initial time  $t = 0$ :

$$T(\mathbf{x}, 0) = T_0(\mathbf{x}) \quad (\mathbf{x} \in \Omega) \quad (4.5)$$

**Thermal equilibrium.** The governing equations for thermal equilibrium follow directly from equations (4.1) and (4.4) by assuming all quantities to be time-independent. Initial conditions are then no longer needed, and the remaining local equations, of the kind already presented in Section 1.6, are

$$\operatorname{div} k \nabla T + g = 0 \quad (\text{in } \Omega), \quad T = T^D \quad (\text{on } S_{th}), \quad -k \nabla T \cdot \mathbf{n} = q^D \quad (\text{on } S_q).$$

#### 4.1.2 Weak formulation of the heat conduction problem

In analogy with the mechanical analyses addressed in the previous chapters, the numerical solution of the heat conduction problem, whose primary unknown, namely the temperature  $T(\mathbf{x}, t)$ , is governed by the heat equation (4.1), the boundary conditions (4.4) and the initial condition (4.5), is based on a weak formulation. Following again the approach developed in Chapter 1 for elastic structures, let  $\mathcal{T}(T^D)$  and  $\mathcal{T}(0)$  define spaces of temperature fields that are admissible with prescribed temperatures equal to  $T^D$  or zero, respectively:

$$\begin{aligned}\mathcal{T}(T^D) &= \{w \mid w \text{ sufficiently regular in } \Omega \text{ and } w = T^D \text{ on } S_{\text{th}}\}, \\ \mathcal{T}(0) &= \{w \mid w \text{ sufficiently regular in } \Omega \text{ and } w = 0 \text{ on } S_{\text{th}}\}.\end{aligned}\quad (4.6)$$

To establish the weak formulation verified by the temperature field  $T(\cdot, t)$  at a given time instant  $t \in [0, t^F]$ , equation (4.1) is multiplied by a scalar virtual field  $w \in \mathcal{T}(0)$  and integrated over  $\Omega$ , which yields

$$\int_{\Omega} \rho c \frac{\partial T}{\partial t}(\mathbf{x}, t) w(\mathbf{x}) \, dV - \int_{\Omega} \operatorname{div}(k \nabla T)(\mathbf{x}, t) w(\mathbf{x}) \, dV = \int_{\Omega} g(\mathbf{x}, t) w(\mathbf{x}) \, dV.$$

Then, applying the divergence theorem to the second integral, one obtains

$$\begin{aligned}& \int_{\Omega} \operatorname{div}(k \nabla T)(\mathbf{x}, t) w(\mathbf{x}) \, dV \\ &= \int_{\partial\Omega} k \nabla T(\mathbf{x}, t) \cdot \mathbf{n}(\mathbf{x}) w(\mathbf{x}) \, dS - \int_{\Omega} k \nabla T(\mathbf{x}, t) \cdot \nabla w(\mathbf{x}) \, dV \\ &= - \int_{S_q} q^D(\mathbf{x}, t) w(\mathbf{x}) \, dS - \int_{\Omega} k \nabla T(\mathbf{x}, t) \cdot \nabla w(\mathbf{x}) \, dV,\end{aligned}$$

with the last equality stemming from the boundary condition (4.4b) and the assumption  $w \in \mathcal{T}(0)$ . Combining the two previous identities leads to the weak formulation

$$\begin{aligned}& \text{find } T(\mathbf{x}, t) \in \mathcal{T}(T^D) \text{ such that} \\ & \int_{\Omega} \rho c \frac{\partial T}{\partial t} w \, dV + \int_{\Omega} k \nabla T \cdot \nabla w \, dV = \int_{\Omega} g w \, dV - \int_{S_q} q^D w \, dS \\ & \quad (\forall t \in [0, t^F], \forall w \in \mathcal{T}(0)), \\ & T(\mathbf{x}, 0) = T_0(\mathbf{x}) \quad (\mathbf{x} \in \Omega).\end{aligned}\quad (4.7)$$

#### 4.1.3 Finite element semi-discretization

An approximation of the domain  $\Omega$  is introduced by means of a finite element mesh, following the approach described in Chapter 2. Any point  $\mathbf{x}$  in a finite element  $E_e$  is represented in parametric form by (2.13), i.e.

$$\mathbf{x} = \sum_{k=1}^{n_e} N_k(\mathbf{a}) \mathbf{x}^{(k)} \quad \mathbf{a} \in \Delta_e \quad (4.8)$$

where the type of finite element used is implicitly defined through the number  $n_e$  of nodes and the choice of the  $n_e$  shape functions  $N_k(\mathbf{a})$ .

The temperature field  $T(\mathbf{x}, t)$  is then approximated using a *semi-discretization in space* based on the isoparametric interpolation. In its global form, introduced in Section 2.2.7, this interpolation has the form

$$T_h(\mathbf{x}, t) = T_h^{(D)}(\mathbf{x}, t) + T_h^{(0)}(\mathbf{x}, t), \quad (4.9)$$

with

$$\begin{aligned} T_h^{(D)}(\mathbf{x}, t) &= \sum_{n | \text{nodes}(n).dof < 0} \tilde{N}_n(\mathbf{x}) T^D(\mathbf{x}^{(n)}, t) \in \mathcal{T}(T^D), \\ T_h^{(0)}(\mathbf{x}, t) &= \sum_{n | \text{nodes}(n).dof > 0} \tilde{N}_n(\mathbf{x}) T^{(n)}(t) \in \mathcal{T}(0). \end{aligned}$$

where the  $\tilde{N}_n(\mathbf{x})$  are the global shape functions derived from the local shape functions used in interpolation (4.8), while  $\mathcal{T}_h(T^D)$  and  $\mathcal{T}_h(0)$  denote the spaces of admissible temperatures (relative to prescribed values  $T^D$  or zero on  $S_{th}$ , respectively) *in the finite element discretization sense*. Following the approach of Section 2.2.7, each node  $\mathbf{x}^{(n)}$  of the mesh is associated with a *thermal unknown number*  $\text{nodes}(n).dof$  according to the convention

$$\begin{aligned} \text{nodes}(n).dof &> 0 \quad (T^{(n)} \text{ free}), \\ \text{nodes}(n).dof &< 0 \quad (T^{(n)} \text{ prescribed}). \end{aligned}$$

The heat conduction problem being scalar, the numbering of thermal unknowns is normally different from that of mechanical unknowns (Section 2.2.7).

The weak formulation (4.7) restricted to virtual fields  $w$  of the form

$$w(\mathbf{x}) = \sum_{n | \text{nodes}(n).dof > 0} \tilde{N}_n(\mathbf{x}) w^{(n)} \in \mathcal{T}_h(0)$$

leads, after an assembly process involving element integrals and similar to that described for linear elasticity in Chapter 2, to the finite-dimensional system of equations

$$\{\mathbb{W}\}^T \left( [\mathbb{C}] \{\mathbb{T}(t)\} + [\mathbb{M}] \{\dot{\mathbb{T}}(t)\} - \{\mathbb{F}(t)\} \right) = 0 \quad \forall \{\mathbb{W}\} \in \mathbb{R}^N \quad (4.10)$$

where  $\{\mathbb{T}(t)\} = \{T^{(n)}(t) : \text{nodes}(n).dof > 0\}$  is the vector of unknown nodal temperatures (each nodal value being time-dependent at this stage), with  $\{\dot{\mathbb{T}}(t)\}$  denoting the time derivative of  $\{\mathbb{T}(t)\}$ . The conductivity and capacity matrices  $[\mathbb{C}]$ ,  $[\mathbb{M}]$  are defined from the weak formulation (4.7) according to

$$\begin{aligned} \int_{\Omega} k \nabla T_h^{(0)}(\cdot, t) \cdot \nabla w \, dV &= \{\mathbb{W}\}^T [\mathbb{C}] \{\mathbb{T}(t)\}, \\ \int_{\Omega} \rho c \frac{\partial T_h^{(0)}}{\partial t}(\cdot, t) w \, dV &= \{\mathbb{W}\}^T [\mathbb{M}] \{\dot{\mathbb{T}}(t)\}. \end{aligned}$$



Likewise, the thermal load vector  $\{\mathbb{F}(t)\}$  is defined through the equality

$$\int_{\Omega} [g(\cdot, t)w - k\nabla T_h^{(D)}(\cdot, t) \cdot \nabla w] dV - \int_{S_q} q^D(\cdot, t)w dS = \{\mathbb{W}\}^T \{\mathbb{F}(t)\}.$$

The following system of semi-discretized equations is thus obtained:

$$[\mathbb{C}]\{\mathbb{T}(t)\} + [\mathbb{M}]\{\dot{\mathbb{T}}(t)\} - \{\mathbb{F}(t)\} = 0 \quad (t \in [0, t^F]). \quad (4.11)$$

The conductivity matrix  $[\mathbb{C}]$  is mathematically similar to a stiffness matrix, while the capacity matrix  $[\mathbb{M}]$  is similar to a *mass matrix* arising in structural dynamics (Chapter 5); in particular, it is always, by construction, positive definite. Since the nodal unknowns are time-dependent, (4.11) is a (linear, first-order) system of ordinary differential equations (ODEs).

## 4.2 Transient heat conduction: discrete time integration

### 4.2.1 Space-time discrete formulation

The transient heat conduction problem defined by the heat equation (4.1), the boundary conditions (4.4) and the initial condition (4.5) leads to solving the linear first-order system of ordinary differential equations (4.11) governing the time-dependent nodal unknowns. A plethora of methods is available for the approximate (in time) solution of such ODE system. Similarly to the elastic-plastic evolution problems (Chapter 9), they are based on sampling the time interval  $[0, t^F]$  using a discrete sequence of time instants. For simplicity,  $M + 1$  regularly-spaced time instants are considered, the discrete sequence thus being of the form

$$t_0 = 0, \quad t_1 = \Delta t, \dots, t_m = m\Delta t, \dots, t_M = M\Delta t = t^F \quad (4.12)$$

with the time step  $\Delta t$  given by  $\Delta t = t^F/M$ . The corresponding sequence of vectors of unknown nodal temperatures is then defined as

$$\{\mathbb{T}_n\} = \{\mathbb{T}(t_n)\} \quad (0 \leq n \leq M), \quad (4.13)$$

with the initial vector  $\{\mathbb{T}_0\}$  containing the nodal values of the initial temperature field  $T_0$ . A non-constant time step might be considered as well, but this case is not addressed here. A time stepping approach is then formulated for solving (4.11) for the nodal temperature evolutions: starting from the given initial temperature  $\{\mathbb{T}_0\}$ , one finds  $\{\mathbb{T}_1\}$ , then  $\{\mathbb{T}_2\}$ , then  $\{\mathbb{T}_3\}$ ... The most important component of such time-stepping procedure is the method allowing the evaluation of  $\{\mathbb{T}_{n+1}\}$  given  $\{\mathbb{T}_0\}, \dots, \{\mathbb{T}_n\}$ .

This presentation will focus on a family of algorithms based on approximating the time derivative  $\{\dot{\mathbb{T}}(t)\}$  by a finite difference:

$$\theta\{\dot{\mathbb{T}}_{n+1}\} + (1 - \theta)\{\dot{\mathbb{T}}_n\} \approx \frac{\{\mathbb{T}_{n+1} - \mathbb{T}_n\}}{\Delta t}, \quad \theta \in [0, 1]. \quad (4.14)$$

The finite difference appearing in the right-hand side is interpreted as the average temperature rate in the time interval  $[t_n, t_{n+1}]$ . This average rate is considered as

the approximation of a linear combination of the time derivatives of the temperature at the initial and final time instants  $t_n, t_{n+1}$  of the current time step, with weights defined in terms of the parameter  $\theta$ . For example, the extreme values  $\theta = 0$  and  $\theta = 1$  amount to assign the finite difference to the initial and final temperature derivative, respectively. One thus has the opportunity of choosing an optimal value of  $\theta$  once the influence of  $\theta$  on the properties of the time-stepping algorithm are determined.

The differential system (4.11) written for the initial and final time instants  $t_n, t_{n+1}$  reads

$$\begin{aligned} [\mathbb{C}]\{\mathbb{T}_n\} + [\mathbb{M}]\{\dot{\mathbb{T}}_n\} - \{\mathbb{F}_n\} &= 0, \\ [\mathbb{C}]\{\mathbb{T}_{n+1}\} + [\mathbb{M}]\{\dot{\mathbb{T}}_{n+1}\} - \{\mathbb{F}_{n+1}\} &= 0. \end{aligned}$$

A linear combination of the above two equations, respectively weighted by  $\theta$  and  $(1-\theta)$ , leads in view of (4.14) to the following linear relationship between  $\{\mathbb{T}_{n+1}\}$  to  $\{\mathbb{T}_n\}$ :

$$[\mathbb{C}]\{\theta\mathbb{T}_{n+1} + (1-\theta)\mathbb{T}_n\} + \frac{1}{\Delta t}[\mathbb{M}]\{\mathbb{T}_{n+1} - \mathbb{T}_n\} = \{\theta\mathbb{F}_{n+1} + (1-\theta)\mathbb{F}_n\}.$$

One may therefore find  $\{\mathbb{T}_{n+1}\}$  for given  $\{\mathbb{T}_n\}$  by solving the linear system of equations

$$\begin{aligned} &\left(\frac{1}{\Delta t}[\mathbb{M}] + \theta[\mathbb{C}]\right)\{\mathbb{T}_{n+1}\} \\ &- \left(\frac{1}{\Delta t}[\mathbb{M}] + (\theta-1)[\mathbb{C}]\right)\{\mathbb{T}_n\} - \{\theta\mathbb{F}_{n+1} + (1-\theta)\mathbb{F}_n\} = \{0\}. \end{aligned} \quad (4.15)$$

Under very general conditions, a time-stepping method such as (4.15) is known to be convergent if and only if it is consistent and stable (this is the Lax equivalence theorem, see e.g. Lax and Richtmyer, 1956). It must be emphasized that only the effect of time discretization is considered in this result, which treats the differential system (4.11) produced by semi-discretization in space as exact.

**Explicit and implicit schemes.** The family of time-stepping schemes defined by (4.15) contains one purely explicit scheme (for  $\theta = 0$ ) and one purely implicit scheme (for  $\theta = 1$ ). For these particular cases, the transition operation  $\{\mathbb{T}_n\} \rightarrow \{\mathbb{T}_{n+1}\}$  defined through (4.15) takes the form

$$[\mathbb{M}]\{\mathbb{T}_{n+1}\} = \Delta t\{\mathbb{F}_n\} + ([\mathbb{M}] - \Delta t[\mathbb{C}])\{\mathbb{T}_n\} \quad (\theta = 0, \text{ explicit}), \quad (4.16)$$

$$([\mathbb{M}] + \Delta t[\mathbb{C}])\{\mathbb{T}_{n+1}\} = \Delta t\{\mathbb{F}_{n+1}\} + [\mathbb{M}]\{\mathbb{T}_n\} \quad (\theta = 1, \text{ implicit}). \quad (4.17)$$

The "explicit" qualifier used for version (4.16) stems from the fact that it is often possible (and legitimate) to replace the matrix  $[\mathbb{M}]$  by a diagonal approximation  $[\bar{\mathbb{M}}]$ , leading to an explicit expression of  $\{\mathbb{T}_{n+1}\}$  (this notion will be revisited in Chapter 5). Time stepping algorithms based on (4.16) or (4.17) are respectively known as explicit or implicit Euler methods.

### 4.2.2 Stability of the time-stepping algorithm

Solving the system (4.15) for  $\{\mathbb{T}_{n+1}\}$  yields

$$\{\mathbb{T}_{n+1}\} = [\mathbb{Q}]\{\mathbb{T}_n\} + \{\mathbb{Y}_{n+1}\}, \quad (4.18)$$

where

$$\begin{aligned} [\mathbb{Q}] &= ([\mathbb{M}] + \theta\Delta t[\mathbb{C}])^{-1}([\mathbb{M}] + (\theta - 1)\Delta t[\mathbb{C}]) \\ \{\mathbb{Y}_{n+1}\} &= \Delta t ([\mathbb{M}] + \theta\Delta t[\mathbb{C}])^{-1} \{\theta\mathbb{F}_{n+1} + (1 - \theta)\mathbb{F}_n\}. \end{aligned}$$

A simple induction on  $n$  then allows to express  $\{\mathbb{T}_{n+1}\}$  in terms of the initial condition  $\{\mathbb{T}_0\}$ , the resolvent matrix  $[\mathbb{Q}]$  and the right-hand sides  $\{\mathbb{Y}_{k+1}\}$ :

$$\{\mathbb{T}_{n+1}\} = [\mathbb{Q}]^{n+1}\{\mathbb{T}_0\} + \sum_{k=0}^n [\mathbb{Q}]^{n-k}\{\mathbb{Y}_{k+1}\}$$

This expression is not suitable for numerical computations, but helps to understand and analyse the stability properties of the time-stepping scheme. The latter is dictated by whether or not errors on successive solutions  $\{\mathbb{T}_n\}$  may be amplified. The time stepping scheme is thus considered as unstable if the matrix  $[\mathbb{Q}]$  of (4.18) is such that the possibility of an error on  $\{\mathbb{T}_n\}$  being amplified by the product  $[\mathbb{Q}]\{\mathbb{T}_n\}$  exists. Should such error be present, for example as a perturbation to the initial data  $\{\mathbb{T}_0\}$ , the evaluation of  $[\mathbb{Q}]^{n+1}\{\mathbb{T}_0\}$  will then undergo an exponentially-increasing error as the discrete time marching progresses. It is therefore essential to determine the values of  $\Delta t$  and  $\theta$  for which the scheme is guaranteed to be stable, i.e. such that for any error  $\{\delta\mathbb{T}\}$  one has

$$\|[\mathbb{Q}]\{\delta\mathbb{T}\}\| < \|\{\delta\mathbb{T}\}\|. \quad (4.19)$$

The analysis of stability is facilitated by a diagonalization of  $[\mathbb{Q}]$ , leading to scalar amplification coefficients. To this aim, let  $\kappa^I$  and  $\{\mathbb{X}^I\}$  ( $1 \leq I \leq N$ ) denote the eigenvalues and eigenvectors solving the generalized eigenvalue problem<sup>1</sup>

$$[\mathbb{C}]\{\mathbb{X}\} - \kappa[\mathbb{M}]\{\mathbb{X}\} = \{0\}. \quad (4.20)$$

All eigenvalues are positive:  $\kappa^I > 0$ . The eigenvectors  $\{\mathbb{X}^I\}$  may be chosen so as to define a  $[\mathbb{M}]$ -orthonormal, and  $[\mathbb{C}]$ -orthogonal, basis of  $\mathbb{R}^N$ , i.e.:

$$\begin{aligned} \{\mathbb{X}^I\}^T[\mathbb{M}]\{\mathbb{X}^I\} &= 1, & \{\mathbb{X}^I\}^T[\mathbb{M}]\{\mathbb{X}^J\} &= 0 \quad (J \neq I), \\ \{\mathbb{X}^I\}^T[\mathbb{C}]\{\mathbb{X}^I\} &= \kappa^I, & \{\mathbb{X}^I\}^T[\mathbb{C}]\{\mathbb{X}^J\} &= 0 \quad (J \neq I). \end{aligned}$$

Expanding  $\{\mathbb{T}_n\}$  and  $\{\mathbb{T}_{n+1}\}$  on this basis, one may set

$$\{\mathbb{T}_n\} = \sum_{J=1}^N \alpha_n^J \{\mathbb{X}^J\}, \quad \{\mathbb{T}_{n+1}\} = \sum_{J=1}^N \alpha_{n+1}^J \{\mathbb{X}^J\}.$$

<sup>1</sup>Since the matrix  $[\mathbb{M}]$  is positive definite (and therefore, in particular, invertible) and the matrices  $[\mathbb{M}]$ ,  $[\mathbb{C}]$  are symmetric, the generalized eigenvalue problem (4.20) is well-posed and yields a simultaneous diagonalization of  $[\mathbb{M}]$  and  $[\mathbb{C}]$ . Moreover,  $[\mathbb{C}]$  is positive, implying  $\kappa^I \geq 0$  for any  $I$ .

and substitute these expressions into (4.15). Upon left-multiplying the resulting equality by  $\{\mathbb{X}^I\}^T$  and using the  $[\mathbb{M}]$ -orthonormality and  $[\mathbb{C}]$ -orthogonality of the eigenvectors, (4.15) finally yields  $N$  scalar uncoupled equations:

$$\left(\frac{1}{\Delta t} + \theta \kappa^I\right) \alpha_{n+1}^I = \left(\frac{1}{\Delta t} + (\theta - 1) \kappa^I\right) \alpha_n^I + \{\mathbb{X}^I\}^T \{\theta \mathbb{F}_{n+1} + (1 - \theta) \mathbb{F}_n\},$$

i.e.

$$\alpha_{n+1}^I = q^I \alpha_n^I + y_{n+1}^I \quad (4.21)$$

with

$$q^I = \frac{1 + (\theta - 1) \kappa^I \Delta t}{1 + \theta \kappa^I \Delta t}, \quad y_{n+1}^I = \frac{\{\mathbb{X}^I\}^T \{\theta \mathbb{F}_{n+1} + (1 - \theta) \mathbb{F}_n\} \Delta t}{1 + \theta \kappa^I \Delta t}.$$

The conditions ensuring stability of the time-stepping scheme may then be formulated in terms of the scalar amplification coefficients  $q^I$ :

$$-1 < q^I < 1 \quad \text{for any } I, 1 \leq I \leq N. \quad (4.22)$$

In view of expression (4.21) of  $q^I$ , the values of  $\theta$  and  $\Delta t$  must therefore be selected so as to satisfy the inequalities

$$-2 < -\frac{\kappa^I \Delta t}{1 + \theta \kappa^I \Delta t} < 0 \quad \text{pour tout } I, 1 \leq I \leq N.$$

Since  $\theta$ ,  $\kappa^I$  and  $\Delta t$  are positive, the second inequality above always holds. The first inequality becomes

$$2 + (2\theta - 1) \kappa^I \Delta t > 0 \quad \text{for any } I, 1 \leq I \leq N \quad (4.23)$$

and shows that two different cases arise:

- (i)  $1/2 \leq \theta \leq 1$ : inequalities (4.23) hold for any value of  $\Delta t$ . The corresponding time-stepping schemes (4.15) are *unconditionally stable*.
- (ii)  $0 \leq \theta < 1/2$ : inequalities (4.23) hold only if  $\Delta t$  is smaller than the *critical time step*  $(\Delta t)_{\text{stab}}$ :

$$\Delta t < (\Delta t)_{\text{stab}} := \frac{2}{1 - 2\theta} \min_I \left( \frac{1}{\kappa^I} \right). \quad (4.24)$$

The corresponding time-stepping schemes (4.15) are *conditionally stable*. Selecting a value for  $\Delta t$  that violates the criterion (4.24) makes the time-stepping scheme (4.15) unstable.

### 4.2.3 Consistency of the time-stepping algorithm

The time-stepping scheme (4.15) is said to be *consistent* with the system of differential equations (4.11) if the residual of the latter has a zero limit as  $\Delta t \rightarrow 0$  when evaluated with  $\{\mathbb{T}_{n+1}\} := \{\mathbb{T}(t_{n+1})\}$  and  $\{\mathbb{T}_n\} := \{\mathbb{T}(t_n)\}$ ,  $\{\mathbb{T}(t)\}$  being a sufficiently smooth vector-valued function verifying the system of ODEs (4.11). Letting  $\{\mathbb{T}(t)\}$  be such a function, the expansions

$$\{\mathbb{T}(t_{n+1})\} = \{\mathbb{T}(t_n)\} + \Delta t \{\dot{\mathbb{T}}(t_n)\} + \frac{\Delta t^2}{2} \{\ddot{\mathbb{T}}(t_n)\} + \frac{\Delta t^3}{6} \{\dddot{\mathbb{T}}(t_n)\} + o(\Delta t^3),$$

and

$$\{\mathbb{F}_{n+1}\} = \{\mathbb{F}_n\} + \Delta t \{\dot{\mathbb{F}}_n\} + \frac{\Delta t^2}{2} \{\ddot{\mathbb{F}}_n\} + o(\Delta t^2).$$

about  $t = t_n$  hold. Substituting them into the left-hand side of (4.15) evaluated for  $\{\mathbb{T}_n\} = \{\mathbb{T}(t_n)\}$ , one obtains

$$\begin{aligned} & \left( \frac{1}{\Delta t} [\mathbb{M}] + \theta [\mathbb{C}] \right) \{\mathbb{T}(t_{n+1})\} \\ & - \left( \frac{1}{\Delta t} [\mathbb{M}] + (\theta - 1) [\mathbb{C}] \right) \{\mathbb{T}(t_n)\} - \{\theta \mathbb{F}_{n+1} + (1 - \theta) \mathbb{F}_n\} = \{\mathbb{R}_n\}. \end{aligned} \quad (4.25)$$

with the definition

$$\begin{aligned} \{\mathbb{R}_n\} &:= ([\mathbb{C}]\{\mathbb{T}(t_n)\} + [\mathbb{M}]\{\dot{\mathbb{T}}(t_n)\} - \mathbb{F}_n) \\ &+ \Delta t (\theta [\mathbb{C}]\{\dot{\mathbb{T}}(t_n)\} + \frac{1}{2} [\mathbb{M}]\{\ddot{\mathbb{T}}(t_n)\} - \theta \dot{\mathbb{F}}_n) \\ &+ \frac{\Delta t^2}{2} (\theta [\mathbb{C}]\{\ddot{\mathbb{T}}(t_n)\} + \frac{1}{3} [\mathbb{M}]\{\dddot{\mathbb{T}}(t_n)\} - \theta \ddot{\mathbb{F}}_n) + o(\Delta t^2). \end{aligned} \quad (4.26)$$

Besides, since  $\{\mathbb{T}(t)\}$  satisfies (4.11) by assumption, one has

$$\begin{aligned} [\mathbb{C}]\{\mathbb{T}(t_n)\} + [\mathbb{M}]\{\dot{\mathbb{T}}(t_n)\} - \mathbb{F}_n &= \{0\}, \\ [\mathbb{C}]\{\dot{\mathbb{T}}(t_n)\} + [\mathbb{M}]\{\ddot{\mathbb{T}}(t_n)\} - \dot{\mathbb{F}}_n &= \{0\}, \\ [\mathbb{C}]\{\ddot{\mathbb{T}}(t_n)\} + [\mathbb{M}]\{\dddot{\mathbb{T}}(t_n)\} - \ddot{\mathbb{F}}_n &= \{0\}, \end{aligned}$$

Substituting these identities into (4.26) then yields

$$\{\mathbb{R}_n\} = \Delta t \left( \theta - \frac{1}{2} \right) [\mathbb{M}]\{\ddot{\mathbb{T}}_n\} + \frac{\Delta t^2}{2} \left( \theta - \frac{1}{3} \right) [\mathbb{M}]\{\dddot{\mathbb{T}}_n\} + o(\Delta t^2). \quad (4.27)$$

For any value of  $\theta$ , the residual  $\{\mathbb{R}_n\}$  of the system (4.15) is of order  $O(\Delta t)$ , and hence vanishes as  $\Delta t \rightarrow 0$  when  $\{\mathbb{T}_{n+1}\}$  and  $\{\mathbb{T}_n\}$  are values of a sufficiently smooth vector function  $\{\mathbb{T}(t)\}$  that satisfies the system (4.11). This completes the proof of consistency for all time-stepping schemes defined by (4.15).

#### 4.2.4 Accuracy of the time-stepping algorithm

The foregoing analysis also allows to estimate the accuracy of the time-stepping algorithm. From expansion (4.27), the schemes defined by (4.15) have in general a  $O(\Delta t)$  accuracy. One notes, however, that choosing  $\theta = 1/2$  removes the  $O(\Delta t)$  contribution to the residual, making that particular scheme  $O(\Delta t^2)$  accurate:

$$\left( \theta = \frac{1}{2} \right) : \quad \{\mathbb{R}_n\} = \frac{\Delta t^2}{12} [\mathbb{M}]\{\dddot{\mathbb{T}}_n\} + o(\Delta t^2) \quad (4.28)$$

Moreover, since no choice of  $\theta$  permits simultaneous cancellation of the  $O(\Delta t)$  and  $O(\Delta t^2)$  contributions to (4.27), one sees this family of schemes cannot achieve an accuracy better than  $O(\Delta t^2)$ . The time-stepping scheme corresponding to  $\theta = 1/2$  is known as the Crank-Nicolson scheme.

Moreover, the accuracy of the solution can be estimated as well. Let  $\{\mathbb{E}_n\} := \{\mathbb{T}(t_n)\} - \{\mathbb{T}_n\}$  denote the solution error at time  $t_n$ . Subtracting (4.15) from (4.25), the sequence of errors is found to satisfy

$$\left(\frac{1}{\Delta t}[\mathbb{M}] + \theta[\mathbb{C}]\right)\{\mathbb{E}_{n+1}\} - \left(\frac{1}{\Delta t}[\mathbb{M}] + (\theta - 1)[\mathbb{C}]\right)\{\mathbb{E}_n\} = \{\mathbb{R}_n\},$$

with  $\{\mathbb{E}_0\} = \{0\}$  since the same initial conditions hold for the exact and approximate solution. Therefore, one finds by induction (with  $[\mathbb{Q}]$  defined as in (4.18))

$$\{\mathbb{E}_{n+1}\} = \Delta t \sum_{k=0}^n [\mathbb{Q}]^{n-k} \{\mathbb{Z}_k\}, \quad \{\mathbb{Z}_n\} := ([\mathbb{M}] + \Delta t \theta [\mathbb{C}])^{-1} \{\mathbb{R}_n\}$$

Invoking the known limiting behavior (4.27) of  $\{\mathbb{R}_n\}$  for small time steps, applying the triangle inequality to the above sum and using that  $\|[\mathbb{Q}]\| \leq 1$  by virtue of assumed stability, one finds

$$\|\{\mathbb{E}_{n+1}\}\| \leq (n+1)C_{n+1}(\Delta t)^{m+1} = C_{n+1}t_{n+1}\Delta t^m$$

where  $m = 1$  (if  $\theta \neq 1/2$ ) or  $m = 2$  (if  $\theta = 1/2$ ) and having set

$$C_{n+1} = \max_{0 \leq k \leq n} \|([\mathbb{M}] + \Delta t \theta [\mathbb{C}])^{-1} \{\mathbb{R}_k\}\|.$$

This establishes the order  $m$  convergence of the solution of the  $\theta$  scheme, and also highlights that the solution error grows linearly with time.

#### 4.2.5 Example: heat diffusion in a spherical domain

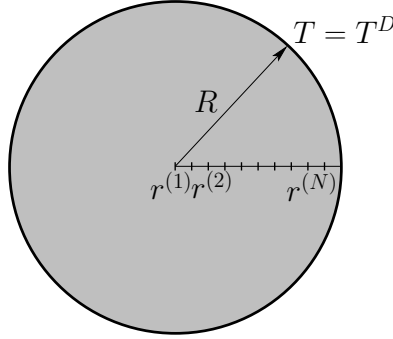
The behavior of explicit and implicit time stepping schemes is now illustrated on a simple heat conduction problem for which an exact solution is known.

A thermally conductive medium, with thermal characteristics  $\rho, c, k$ , fills a sphere of radius  $R$ . The initial temperature is  $T = 0$ . A constant temperature  $T^D$  is applied on the external surface  $r = R$  during the time interval  $[0, t^F]$  (with  $r$  denoting the radial coordinate), in the absence of any body heat source (i.e.  $g = 0$ ). Under these conditions, the temperature has radial symmetry:  $T = T(r, t)$ . Using the well-known expression of the Laplacian in spherical coordinates, the governing equations for  $T(r, t)$  are:

$$\begin{aligned} \frac{1}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial T}{\partial r} \right) - \frac{\rho c}{k} \frac{\partial T}{\partial t} &= 0 & (0 \leq r \leq R, 0 \leq t \leq t^F) \\ T(r, 0) &= 0 & (0 \leq r \leq R) \\ T(R, t) &= T^D & (0 \leq t \leq t^F) \end{aligned} \quad (4.29)$$

This set of equations can be solved exactly using classical separation of variables, to obtain

$$T(r, t)/T^D = 1 + \sum_{m \geq 1} \frac{(-1)^m}{m} \frac{2R}{\pi r} \sin \frac{m\pi r}{R} \exp \left( - \left( \frac{m\pi}{R} \right)^2 \frac{k}{\rho c} t \right). \quad (4.30)$$



**Figure 4.1:** Sphere with prescribed external temperature: notation.

The accuracy of numerically computed solutions will be evaluated against this reference solution.

The semi-discretization in space here simply consists in cutting the interval  $0 \leq r \leq R$  into  $N$  segments. The latter are assumed to be of uniform length  $\Delta r = R/N$ , so that the endpoint radii are given by  $r^{(1)} = 0, \dots, r^{(n)} = (n-1)\Delta r, \dots, r^{(N)} = (N-1)\Delta r$ . The unknown temperature is approximated by a continuous, piecewise-linear, interpolation of the nodal values  $\{\mathbb{T}(t)\} = \{T^{(1)}(t), \dots, T^{(N)}(t)\}$  (with  $T^{(n)}(t) = T(r^{(n)}, t)$ ), taken as unknowns. Problem (4.29) can then easily be cast in the semi-discretized form (4.10).

### Numerical implementation.

This example shares many common characteristics with that of Section 1.5, which the reader should read before going through what follows.

**Weak form.** The weak form of equation (4.29) is found by integrating it over the sphere after multiplication by a radially-symmetric test function  $w \in \mathcal{T}(0)$ :

$$\int_0^R w \left[ \frac{k}{r^2} \frac{\partial}{\partial r} \left( r^2 \frac{\partial T}{\partial r} \right) - \rho c \dot{T} \right] 4\pi r^2 dr = 0, \quad \forall w \in \mathcal{T}(0).$$

On integrating by parts and using the boundary conditions, one obtains (dropping the unnecessary constant factor  $4\pi$ ):

$$\int_0^R \left( \rho c w \dot{T} r^2 + k r^2 \frac{\partial w}{\partial r} \frac{\partial T}{\partial r} \right) dr = 0, \quad \forall w \in \mathcal{T}(0). \quad (4.31)$$

Recall that all functions of  $\mathcal{T}(0)$  vanish on the boundary portion supporting a prescribed temperature, so that  $w(R) = 0$  here.

**Solution of the weak formulation using finite elements.** The interval  $0 \leq r \leq R$  is cut into elements. Similarly to the case of the sphere in linear elasticity, the approximation space for the temperature consists of all continuous and piecewise-linear

functions, which are completely defined from their nodal values. On the generic element  $E_e$  of length  $h$ , one thus again has

$$T_h = N_1(a)T^{(1)} + N_2(a)T^{(2)} = \{N\}^T \{T_e\}$$

where the shape functions have been defined in Section 2.1.4 and table 2.1 where  $\{T_e\}$  is the vector of nodal temperatures for the element and  $N_i$  are the shape functions. The evaluation of (4.31) then, as usual, consists of element-wise integrations and assembly of resulting element matrices into the global matrices. This task is implemented in routines as explained next.

**Element conductivity matrix.** For a given element of nodal coordinates  $r^{(1)}, r^{(2)}$  one has:

$$\int_{r^{(1)}}^{r^{(2)}} k r^2 \frac{\partial w}{\partial r} \frac{\partial T_h}{\partial r} dr = \{W_e\}^T \underbrace{\left( \int_0^L \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \frac{k r^2}{L^2} ds \right)}_{[C_e]} \{T_e\}$$

The corresponding MATLAB code reads

```
function Ke=B2_1S_therm_Ke(X,k)

r1=X(1); r2=X(2); % node coordinates
h=r2-r1; % element length
Ke=k/(3*h)*(r1^2+r1*r2+r2^2)*[1,-1; -1,1]; % element conductivity matrix
```

It takes as input the matrix  $X$  (whose two lines store the radial coordinates of the two element nodes) and the conductivity  $k$ , and outputs the element conductivity matrix  $[C_e] \rightarrow K_e$ .

**Element capacity matrix.** Similarly, one has

$$\int_{r^{(1)}}^{r^{(2)}} \rho c w \frac{\partial T_h}{\partial t} r^2 dr = \{W_e\}^T \underbrace{\left( \int_0^h \rho c \begin{bmatrix} N_1^2 & N_1 N_2 \\ N_2 N_1 & N_2^2 \end{bmatrix} r^2 ds \right)}_{[M_e]} \{\dot{T}_e\} \quad (4.32)$$

which, upon effecting the necessary integrations, translates into MATLAB code as:

```
function Me=B2_1S_therm_Me(X,c,lumped)

r1=X(1); r2=X(2); % radial coordinates of nodes
h=r2-r1; % element length
Me=c*h/60*[12*r1^2+6*r1*r2+2*r2^2, ... % elementary mass matrix
            3*r1^2+4*r1*r2+3*r2^2;
            3*r1^2+4*r1*r2+3*r2^2, ...
            2*r1^2+6*r1*r2+12*r2^2];

if lumped==1
    Me=diag(sum(Me)); % diagonal matrix approx.
end
```

The function first computes the “consistent” capacity matrix according to (4.32). However, one often prefers to use a diagonal “lumped” approximation for the latter, in order to generate a diagonal global conductivity matrix that allows completely explicit time stepping schemes.



The function `B2_1S_therm.Me` implements a simple lumping technique synthesized by the command `Me=diag(sum(Me))` which creates a column vector filled with the sum of all the columns of the consistent matrix. This procedure can be justified on the basis of the following remarks. The total capacity of an element:

$$C_e = \int_{E_e} \rho c \, dV$$

must be preserved by the diagonal approximation. Now, with  $\partial T / \partial t = w = 1$  one has:

$$\int_{E_e} \rho c \frac{\partial T}{\partial t} w \, dV = \int_{E_e} \rho c \, dV = C_e$$

In every isoparametric element the unit fields  $\partial T / \partial t = w = 1$  can be represented exactly setting all entries of  $\{W_e\}$  and  $\{\dot{T}_e\}$  to 1, leading to:

$$\{W_e\} [M_e] \{\dot{T}_e\} = \sum_{i,j} [M_e]_{ij} = \int_{E_e} \rho c \, dV$$

Hence the element capacity  $C_e$  is given by the sum of all the entries of the consistent element capacity matrix. Now, if  $[M_e]$  is replaced by a diagonal approximation where the  $i$ -th diagonal coefficient is the row sum  $\sum_j [M_e]_{ij}$ , the total capacity is preserved. In specific types of elements some rows of  $[M_e]$  sum to zero generating a singular mass matrix. A more robust choice is, in these cases, the HRZ method which creates a diagonal element capacity matrix such that: i) the total element capacity is preserved; ii) the ratio between diagonal entries is the same as that of the consistent matrix. The above conditions lead to a uniquely defined diagonal matrix  $[M_e]$ . Its implementation is left as an exercise.

It is then easy to show that the global matrix  $[\mathbb{M}]$  is diagonal if each element matrix  $[M_e]$  is diagonal.

**Setup and time integration.** The element matrices are now assembled into the global matrices, so as to implement the discrete formulation (4.15). How the choice of parameter  $\theta$  affects the computation is examined by considering two cases. Choosing  $\theta = 1$  (backward finite differences, also referred to as Euler implicit) leads to the integration scheme defined by

$$\left( \frac{1}{\Delta t} [\mathbb{M}] + [\mathbb{C}] \right) \{\mathbb{T}_{n+1}\} = \frac{1}{\Delta t} [\mathbb{M}] \{\mathbb{T}_n\} + \{\mathbb{F}_{n+1}\}, \quad (4.33)$$

which is unconditionally stable. Alternatively, choosing  $\theta = 0$  (forward finite differences, also referred to as Euler explicit) leads to the integration scheme defined by

$$\frac{1}{\Delta t} [\mathbb{M}] \{\mathbb{T}_{n+1}\} = \left( \frac{1}{\Delta t} [\mathbb{M}] - [\mathbb{C}] \right) \{\mathbb{T}_n\} + \{\mathbb{F}_n\} \quad (4.34)$$

which is conditionally stable, with  $\Delta t < 2\min(1/\kappa^I)$ .

**MATLAB implementation.** The foregoing formulation is implemented in the code `sphere_B2_1S_diff` in both implicit ( $\theta = 1$ ) and explicit ( $\theta = 0$ ) versions. The latter formulation may be very efficient if  $[\mathbb{M}]$  is diagonal (which can for example be achieved by assembling the element matrices output by the function `B2_1S_therm.Me`), with `lumped=1`), since no matrix inversion or factorization is required in that case.

The problem parameters are set and, among them, the value of `implicit` specifies the choice of the integration scheme (implicit or explicit, respectively employing the consistent capacity matrix or a diagonal approximation of it).

```
R=1; % outer radius
NE=10; % number of elements
rhoc=10.; % capacity
k=1.; % conductivity
TD=1; % value of temperature in  $r=R$ 
tf=1; % final time
Dt=.001; % time step
implicit=1; % implicit or not?
if implicit==1, lumped=0;
else lumped=1; % if explicit lumped cap matrix
end
```

The code effects a first preprocessing step from the user-provided data, fixing the temperature on the last node.

```
h=R/NE; % size of one element
NN=NE+1; % number of nodes
coor=[0:h:R]'; % nodal coordinates: uniform discr.
neq=NN-1; % number of equations
U=zeros(NN,1); % initialisation of temperatures
U(NN)=TD; % boundary conditions
```

The matrix setup phase is now prepared by reserving adequate memory space for the capacity and conductivity matrices.

```
C=zeros(neq,neq); % allocates conductivity matrix
M=zeros(neq,neq); % allocates capacity matrix
F=zeros(neq,1); % allocates rhs side
```

The global matrices are now assembled. The prescribed temperature at  $r = R$  contributes to the thermal load vector  $\{\mathbb{F}\}$ . The following code is a mere variation on methods used on several previous occasions.

```
for e=1:NE, % loop over elements
    Xe=[coor(e) coor(e+1)]; % creates segment
    Me=B2_1S_therm_Me(Xe,rhoc,lumped); % element capacity
    Ke=B2_1S_therm_Ke(Xe,k); % element conductivity
    Ie=[e e+1]; % sets nodal degrees of freedom
    if e<NE % if not last element
        M(Ie,Ie)=M(Ie,Ie)+Me; % the whole Me goes into M
        K(Ie,Ie)=K(Ie,Ie)+Ke; % the whole Ce goes into C
    else
        M(neq,neq)=M(neq,neq)+Me(1,1); % else only the first coefficient
        K(neq,neq)=K(neq,neq)+Ke(1,1); % else only the first coefficient
        F(neq)=-Ke(1,2)*T(NN); % and Ce also contributes to rhs
    end
end
```

Before starting the time-stepping procedure, the critical time step is determined if the explicit version is used (`implicit=0`) to allow a comparison with the time step chosen

by the user.

```
if implicit==0, % computes stable Dt: expl. option
    uu=eigs(K,M,1);
    stable_time_increment=2/uu
end
```

On the other hand, if the implicit version is used (`implicit=1`), the  $LDL^T$  decomposition of the capacity matrix is computed and stored:

```
if implicit==1, % prepares LDLT decomp.
    K=K+M/Dt;
    KC=chol(K);
    clear K
end
```

allowing a faster computation of the temperature solution for each time step. The vector of unknown nodal temperatures is stored, and  $\{\mathbb{T}\} \rightarrow T$  is initially set to zero (i.e. zero initial temperature is assumed):

```
T=zeros(neq,1);
nstep=floor(tf/Dt); % number of time steps assuming constant dt
```

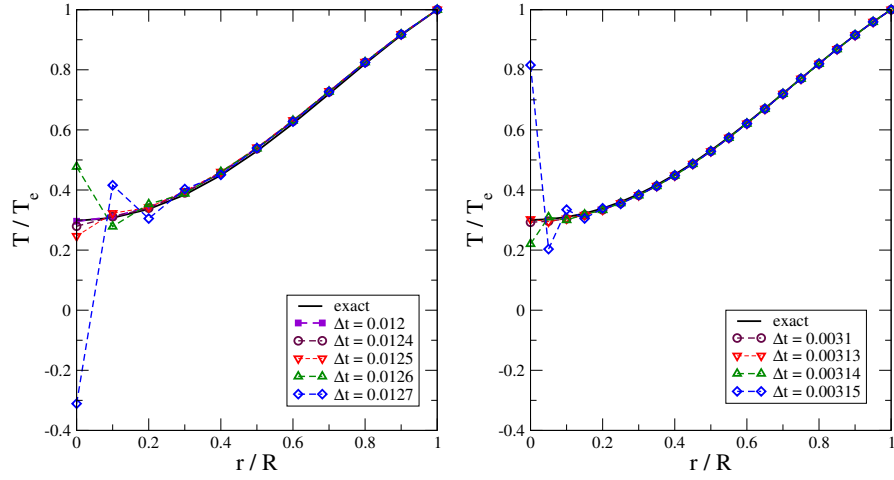
Finally, the computation of the temperature history proceeds using a simple transcription of (4.16) or (4.17):

```
for step=1:nstep, % loop over time steps
    if implicit==1,
        Fg=F+1/Dt*M*T; % implicit case
        T=KC\'KC\Fg;
    else
        Fg=F-K*T; % explicit case
        T=T+Dt*Fg./diag(M);
    end

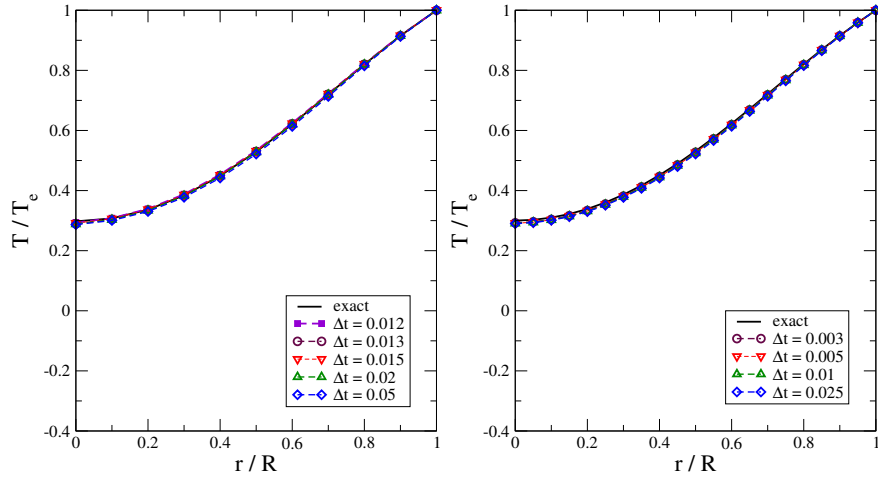
    U(1:NN-1)=T; % temps over all nodes
    vec_time(step)=step*Dt;
    vec_temperature(step)=U(1);
end
```

The last step consists of post-processing the solution and comparing it with the known exact solution of the sphere problem.

**Numerical results.** Figures 4.2 and 4.3 show numerical results for the temperature along a radial line inside the sphere at the final time  $T(r, t^F)$ , obtained using either the explicit or implicit time stepping scheme. When using the explicit scheme, the critical time step  $(\Delta t)_{\text{stab}}$  has been evaluated by solving the generalized eigenvalue problem (4.20). The following parameters were used:  $R = 1$  m,  $k/\rho c = 0, 1 \text{ m}^2 \text{s}^{-1}$ ,  $t^F = 1$  s. Figure 4.2 shows results achieved using explicit time stepping and either  $N = 10$  or  $N = 20$  elements, respectively. Several values of the time step  $\Delta t$ , chosen very close to the critical value  $(\Delta t)_{\text{stab}}$ , were used. In both cases, instabilities occur near the sphere centre  $r = 0$  whenever  $\Delta t > (\Delta t)_{\text{stab}}$ . Still larger values of  $\Delta t$  lead to computed solutions  $T(r, t^F)$  that deviate markedly from the exact solution. This effect is further illustrated in Figure 4.4, which

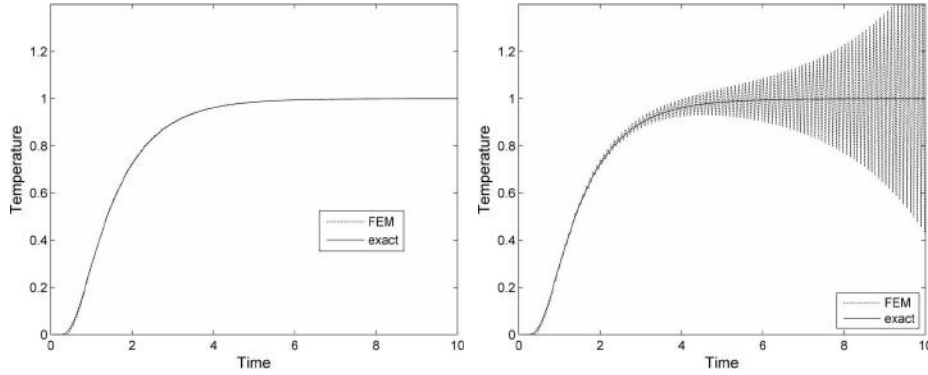


**Figure 4.2:** Sphere with prescribed external temperature: final temperature  $T(r, t^F)$  obtained using explicit time stepping for several values of  $\Delta t$  close to  $(\Delta t)_{stab}$ , and comparison to the exact solution. Left:  $N = 10$  spatial unknowns,  $(\Delta t)_{stab} \approx 1.2433 \cdot 10^{-2}$  s. Right:  $N = 20$  spatial unknowns,  $(\Delta t)_{stab} \approx 3.1309 \cdot 10^{-3}$  s.



**Figure 4.3:** Sphere with prescribed external temperature: final temperature  $T(r, t^F)$  obtained using implicit time stepping for several values of  $\Delta t$  close to the explicit stability threshold  $(\Delta t)_{stab}$ , and comparison to the exact solution. The choice of  $\Delta t$  does not affect the stability of the time stepping. Left:  $N = 10$  spatial unknowns,  $(\Delta t)_{stab} \approx 1.2433 \cdot 10^{-2}$  s. Right:  $N = 20$  spatial unknowns,  $(\Delta t)_{stab} \approx 3.1309 \cdot 10^{-3}$  s.

shows the temperature history at the sphere centre for  $\Delta t = 0.99(\Delta t)_{stab}$  (stable computation) or  $\Delta t = 1.01(\Delta t)_{stab}$  (unstable computation). In contrast, fig-

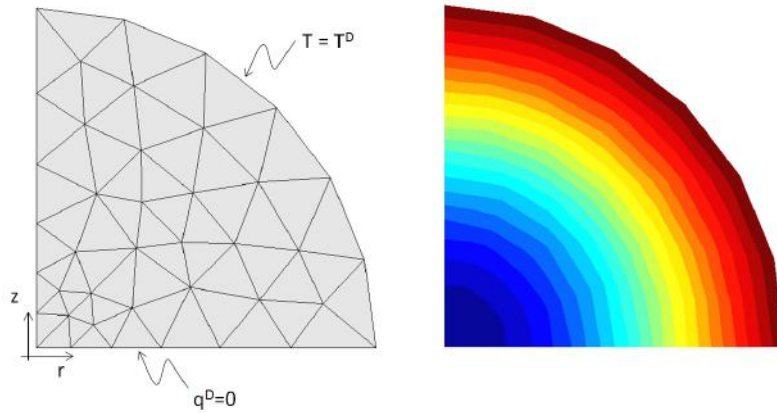


**Figure 4.4:** Sphere with prescribed external temperature: temperature history  $T(r=0, t)$  at the sphere centre, obtained using explicit time stepping for  $\Delta t = 0.99(\Delta t)_{stab}$  (left) or  $\Delta t = 1.01(\Delta t)_{stab}$  (right).

ure 4.3, corresponding to values of  $\Delta t$  chosen either close to  $(\Delta t)_{stab}$  or substantially larger than  $(\Delta t)_{stab}$ , experimentally confirm the predicted unconditional stability of the implicit time-stepping scheme.

#### 4.2.6 Heat diffusion for axisymmetric problems

The code `genlin` (see Chapter 3 and Appendix B) can be rapidly adapted to the analysis of heat diffusion. The file `heatdiffusion.m` presents an example of an implicit approach ( $\theta = 1$ ) employing a consistent capacity matrix.



**Figure 4.5:** Heated sphere in axisymmetric reference system: mesh employed and temperature distribution at a given time value

As an application we focus on the same problem as in Section 4.2.5, but the sphere is herein analysed in an axisymmetric reference system. As illustrated in figure 4.5 only one half of the generating surface is discretized, exploiting symmetry ( $q^D = 0$  on

the axis of symmetry). The mesh can be generated using the Chap4\_sphere\_diff\_axi file which follows the same guidelines as for the genlin code. The analysis file Chap4\_sphere\_diff\_axi contains as usual the definition of materials, solids and boundary conditions, but also introduces parameters related to the time history analysis, namely the total duration  $t_f$  and the time step  $\Delta t$ .

```
file='Chap4_sphere_diff_axi.msh';

% ANALYSIS TYPE: thermal AXI
analysis.type = 2;

% MATERIAL: k and rhoc
material = [1 10];

% SOLID: associates materials to sets
solid = [2 1];

% DBC: physical set, dir. (for scalar prob 1 by default), val
dbc = [1 1 1];

% parameters for time history
tf = 1;
Dt = 0.001;
```

It is worth stressing that material now contains, in addition to the conductivity  $k$ , the heat capacity  $\rho c$  which is added after  $k$  so as not to affect genlin (which employs the same routines for element conductivity matrices). To run the analysis, the user must provide the element-level routines for the selected element. For the quadratic triangle selected herein, the MATLAB functions have already been developed and are: T6\_AX\_therm\_Me (computing the consistent mass matrix), T6\_AX\_therm\_Ke (generating the element conductivity matrix), T6\_AX\_solid\_Sg (used in the post-processing phase to evaluate gradients at the Gauss points), B3\_AX\_therm\_Fe\_surf (providing the equivalent nodal loads due to the imposed fluxes). Reading these routines is recommended as an exercise on isoparametric elements.

The structure of heatdiffusion is identical to genlin, the differences being limited to few lines. First of all, in the assemblage phase the standard procedure for  $[\mathbb{K}]$  is replicated also for  $[\mathbb{M}]$ :

```
for e=1:analysis.NE,                                % stiffness matrix assemblage
...
Ke=eval([Etag Atag 'Ke(Xe,material(mat,:))']);
Me=eval([Etag Atag 'Me(Xe,material(mat,:))']);
Le0=find(Ge>0);                                     % local numbering of unknowns
Ie=Ge(Le0);                                          % global numbering
K(Ie,Ie)=K(Ie,Ie)+Ke(Le0,Le0);                     % conductivity matrix assemblage
M(Ie,Ie)=M(Ie,Ie)+Me(Le0,Le0);                     % capacity matrix assemblage
...
end
```

Next, the time history closely follows what developed in Section 4.2.5, with the difference that only the implicit case has been coded. The  $LDL^T$  decomposition is com-

puted once for all at the beginning of the analysis

```
K=K+M/Dt;
KC=chol(K);
```

and the time marching analysis is performed:

```
U=zeros(neq,1);
nstep=floor(tf/Dt); % number of steps with const. dt
for step=1:nstep, % loop over time steps
    Fg=F+1/Dt*M*U; % implicit case
    U=KC\KC'\Fg;
end
```

The final values of the temperature nodal values are eventually post-processed with GMSH.

**Exercise 4.1.** *Using heatdiffusion, simulate the heating of the sphere subjected to Dirichelet boundary conditions and compare with the analytical solution (4.30)*

### 4.3 Linear thermoelasticity

Temperature variations in materials usually induce volume changes through thermal dilatation. The corresponding thermal strains are in general incompatible (i.e. cannot be linked to a displacement field). The actual strain taking place in heated or cooled materials must therefore "correct" this incompatibility. This gives rise to stresses of thermal origin, even in the absence of any mechanical loading.

The simplest thermomechanical, to which this chapter is restricted, is that of uncoupled thermoelasticity. In this approach, the temperature field  $T(\mathbf{x}, t)$  at any time  $t$  results from external (i.e. other than mechanical) causes and is not influenced by the mechanical state of the material. The thermoelastic constitutive relation is then of the form (see e.g. Maugin, 1992)

$$\boldsymbol{\sigma}(\mathbf{x}, t) = \mathcal{A} : [\boldsymbol{\varepsilon}(\mathbf{x}, t) - \tilde{T}(\mathbf{x}, t)\boldsymbol{\alpha}], \quad \tilde{T}(\mathbf{x}, t) := T(\mathbf{x}, t) - T_0 \quad (4.35)$$

where  $\boldsymbol{\varepsilon}(\mathbf{x}, t)$  is the linearized strain,  $\boldsymbol{\alpha}$  is the anisotropic thermal dilatation tensor, and  $T_0$  is the reference temperature for the solid in undeformed state. If the material is thermally isotropic, the temperature variation  $\tilde{T}$  induces isotropic thermal strains and  $\boldsymbol{\alpha} = \alpha \mathbf{I}$  in terms of the scalar thermal dilatation coefficient  $\alpha$ . In this uncoupled approach, the thermomechanical state of a structure is determined in two steps:

- (i) Compute the evolutive temperature field  $T(\mathbf{x}, t)$  by solving the heat conduction equations;
- (ii) Compute the thermoelastic solution by solving the mechanical equations, using the constitutive relation (4.35).

In this chapter, only quasi-static mechanical states are considered, whereas the thermal state is allowed to be time-dependent (a justification for this assumption is given before Section 4.3.1). Put differently, thermal evolutions are assumed to be slow enough to allow neglecting inertia effects in the mechanical response.

Moreover, attention is focused on situations where the mechanical response results from only thermal excitation; this does not entail any loss of generality as the contribution of other types of loading can be computed separately and (by virtue of the adopted linear thermoelastic behavior) included using the superposition principle. Hence all prescribed body forces, surface tractions or applied displacements will be assumed to be zero in this chapter. Under these conditions, the evolution over a time interval  $[0, t^F]$  of the considered solid is governed by the local field equations

$$\varepsilon = \frac{1}{2}(\nabla \mathbf{u} + \nabla^T \mathbf{u}) \quad \text{in } \Omega \times [0, t^F] \quad (\text{compatibility}), \quad (4.36a)$$

$$\operatorname{div} \boldsymbol{\sigma} = \mathbf{0} \quad \text{in } \Omega \times [0, t^F] \quad (\text{equilibrium}), \quad (4.36b)$$

$$\boldsymbol{\sigma} = \mathcal{A}:(\varepsilon - \tilde{T}\boldsymbol{\alpha}) \quad \text{in } \Omega \times [0, t^F] \quad (\text{constitutive relation}), \quad (4.36c)$$

where the temperature field  $T(\mathbf{x}, t)$  is assumed to be known from a preliminary heat conduction analysis, and the boundary conditions

$$\mathbf{u} = \mathbf{0} \quad \text{on } S_u \times [0, t^F] \quad (\text{prescribed displacement}), \quad (4.36d)$$

$$\boldsymbol{\sigma}[\mathbf{u}] \cdot \mathbf{n} = \mathbf{0} \quad \text{on } S_T \times [0, t^F] \quad (\text{prescribed traction}). \quad (4.36e)$$

As a result of the quasistatic assumption, equations (4.36a) to (4.36e) define for any given time  $t$  an equilibrium problem.

**Validity of the quasi-static assumption for the mechanical response.** Since a thermal evolution induces a mechanical evolution (even in the absence of any other excitation), the quasi-static assumption adopted for the mechanical response requires validation. This can here be achieved by comparing the characteristic times associated with the dynamical response and the heat diffusion. The former can be estimated by noticing that the travel time of an elastic wave over the characteristic diameter  $L$  of  $\Omega$  is of the order  $T_{\text{prop}} \approx L(\rho/E)^{1/2}$  (see Sec. 5.1.2), whereas the heat diffusion over a distance  $L$  is  $T_{\text{diff}} \approx \rho c L^2 / 4k$  (Carslaw and Jaeger, 1959). For many materials, one has  $T_{\text{diff}}/T_{\text{prop}} = O(10^2 - 10^4)$ . The heat diffusion is then much slower than the propagation of mechanical waves, which makes neglecting dynamical effects while accounting for thermal evolution effects a legitimate approximation.

#### 4.3.1 Energy minimization principles for thermoelastic problems

The problem defined by equations (4.36a) to (4.36e) can, equivalently and in keeping with the approach followed in Sections 1.1 and 1.3, be recast in the form

$$\begin{aligned} \text{find } (\mathbf{u}, \boldsymbol{\sigma}) &\in \mathcal{C}(\mathbf{0}) \times \mathcal{S}(\mathbf{0}, \mathbf{0}) \\ \text{such that } \boldsymbol{\sigma} &= \mathcal{A}:(\varepsilon[\mathbf{u}] - \tilde{T}\boldsymbol{\alpha}) \quad \text{in } \Omega \times [0, t^F] \end{aligned} \quad (4.37)$$

Expressing (4.37) in words, one seeks among all displacement fields kinematically admissible to zero and stress fields statically admissible to zero (i.e., self-equilibrated), the unique displacement-stress pair  $(\mathbf{u}, \boldsymbol{\sigma})$  that is linked by the constitutive equation.



Minimum principles for the potential energy and the complementary energy, which are generalizations of (1.29)–(1.31) and (1.30)–(1.32), respectively, may be established using the approach of Chapter 1. To this aim, the definition (1.24) of the error in constitutive relation functional is extended to the linear thermoelastic constitutive model by setting

$$\mathcal{E}_{\text{th}}(\mathbf{v}, \boldsymbol{\tau}) = \frac{1}{2} \int_{\Omega} (\boldsymbol{\tau} - \mathcal{A} : (\boldsymbol{\varepsilon}[\mathbf{v}] - \tilde{T}\boldsymbol{\alpha})) : \mathcal{S} : (\boldsymbol{\tau} - \mathcal{A} : (\boldsymbol{\varepsilon}[\mathbf{v}] - \tilde{T}\boldsymbol{\alpha})) \, dV, \quad (4.38)$$

i.e., upon expanding the squared terms under the integral sign:

$$\begin{aligned} \mathcal{E}_{\text{th}}(\mathbf{v}, \boldsymbol{\tau}) &= \frac{1}{2} \int_{\Omega} (\boldsymbol{\varepsilon}[\mathbf{v}] - \tilde{T}\boldsymbol{\alpha}) : \mathcal{A} : (\boldsymbol{\varepsilon}[\mathbf{v}] - \tilde{T}\boldsymbol{\alpha}) \, dV \\ &\quad + \frac{1}{2} \int_{\Omega} \boldsymbol{\tau} : \mathcal{S} : \boldsymbol{\tau} \, dV - \int_{\Omega} \boldsymbol{\tau} : (\boldsymbol{\varepsilon}[\mathbf{v}] - \tilde{T}\boldsymbol{\alpha}) \, dV. \end{aligned}$$

Then, taking into account the equilibrium equation (4.36c) and boundary conditions (4.36d,e), one obtains

$$(\mathbf{v}, \boldsymbol{\tau}) \in \mathcal{C}(\mathbf{0}) \times \mathcal{S}(\mathbf{0}, \mathbf{0}) \implies \int_{\Omega} (\boldsymbol{\tau} : \boldsymbol{\varepsilon}[\mathbf{v}]) \, dV = 0$$

by virtue of the divergence theorem (or, equivalently, of the virtual work principle). Separating terms involving  $\mathbf{v}$  or  $\boldsymbol{\tau}$ , one arrives at

$$\mathcal{E}_{\text{th}}(\mathbf{v}, \boldsymbol{\tau}) = \mathcal{P}_{\text{th}}(\mathbf{v}) + \mathcal{P}_{\text{th}}^*(\boldsymbol{\tau}), \quad (4.39)$$

where the potential energy  $\mathcal{P}_{\text{th}}(\mathbf{v})$  and the complementary energy  $\mathcal{P}_{\text{th}}^*(\boldsymbol{\tau})$  are defined by

$$\mathcal{P}_{\text{th}}(\mathbf{v}) = \frac{1}{2} \int_{\Omega} (\boldsymbol{\varepsilon}[\mathbf{v}] - \tilde{T}\boldsymbol{\alpha}) : \mathcal{A} : (\boldsymbol{\varepsilon}[\mathbf{v}] - \tilde{T}\boldsymbol{\alpha}) \, dV, \quad (4.40)$$

$$\mathcal{P}_{\text{th}}^*(\boldsymbol{\tau}) = \frac{1}{2} \int_{\Omega} \boldsymbol{\tau} : \mathcal{S} : \boldsymbol{\tau} \, dV + \int_{\Omega} \tilde{T}\boldsymbol{\alpha} : \boldsymbol{\tau} \, dV. \quad (4.41)$$

The displacement solution  $\mathbf{u}$  is then found by minimizing the potential energy among all displacements that are kinematically admissible to zero:

$$\mathbf{u} = \arg \min_{\mathbf{v} \in \mathcal{C}(\mathbf{0})} \mathcal{P}_{\text{th}}(\mathbf{v}). \quad (4.42)$$

For the usual case of isotropic elasticity, and assuming in addition that the thermal dilatation is also isotropic (i.e.  $\boldsymbol{\alpha} = \alpha \mathbf{I}$ ), one can easily show using the formulas of Section 1.1.3 that  $\mathcal{A} : \mathbf{I} = 3\kappa \mathbf{I}$  and  $\mathbf{I} : \mathcal{A} : \mathbf{I} = 9\kappa$  (with the bulk modulus  $\kappa$  given by  $3\kappa = E/(1 - 2\nu)$ ), so that the potential energy is given by

$$\begin{aligned} \mathcal{P}_{\text{th}}(\mathbf{v}) &= \mu \int_{\Omega} \left( \frac{\nu}{1 - 2\nu} (\text{Tr}(\boldsymbol{\varepsilon}[\mathbf{v}]))^2 + \boldsymbol{\varepsilon}[\mathbf{v}] : \boldsymbol{\varepsilon}[\mathbf{v}] \right) \, dV \\ &\quad - 3\kappa\alpha \int_{\Omega} \tilde{T} \text{Tr}(\boldsymbol{\varepsilon}[\mathbf{v}]) \, dV + \frac{9\kappa\alpha^2}{2} \int_{\Omega} \tilde{T}^2 \, dV. \end{aligned} \quad (4.43)$$

### 4.3.2 Variational formulation and finite element approximation

The usual form of the finite element method for solving the quasistatic linear thermoelasticity problem (4.36a–e) rests upon an approximate minimization of the potential energy (4.40) where the infinite-dimensional space of all kinematically admissible displacements is replaced with a finite-dimensional approximation space. The stationarity equation for  $\mathcal{P}_{\text{th}}(\mathbf{v})$ , whose solution is  $\mathbf{v} = \mathbf{u}$ , is established from (4.40) and leads to the variational formulation of the thermoelastic problem (4.36a–e):

find  $\mathbf{u} \in \mathcal{C}(\mathbf{0})$  such that

$$\int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = \int_{\Omega} \tilde{T} \boldsymbol{\alpha} : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV, \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (4.44)$$

Introducing into (4.44) finite element approximations  $\Omega_h$  and  $\mathbf{u}_h$  of  $\Omega$  and  $\mathbf{u}$  and using virtual fields  $\mathbf{w} \in \mathcal{C}_h(\mathbf{0})$  belonging to the same approximation space leads, using methods and notations introduced in Chapters 2 and 3, to the linear system of equations

$$[\mathbb{K}] \{\mathbf{U}(t)\} = \{\mathbb{F}(t)\}, \quad (4.45)$$

where the vector  $\{\mathbb{F}(t)\}$  of generalized nodal forces results from the thermal excitation and is defined through

$$\{\mathbb{W}\}^T \{\mathbb{F}(t)\} = \int_{\Omega} \tilde{T}(\cdot, t) \boldsymbol{\alpha} : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV, \quad (4.46)$$

$\{\mathbb{W}\}$  collecting all nodal values of the virtual field  $\mathbf{w} \in \mathcal{C}_h(\mathbf{0})$ . In the usual case of isotropic elasticity,  $\{\mathbb{F}(t)\}$  is hence defined by

$$\{\mathbb{W}\}^T \{\mathbb{F}(t)\} = 3\alpha\kappa \int_{\Omega} \tilde{T}(\cdot, t) \text{Tr}(\boldsymbol{\varepsilon}[\mathbf{w}]) \, dV. \quad (4.47)$$

To determine the thermomechanical evolution of the structure, the history  $\tilde{T}(\mathbf{x}, t)$  of temperature perturbation must be known. The latter is not normally given *a priori*, and needs to be found by solving the heat transfer equations. These equation may assume various forms depending on the assumed thermal constitutive properties of the material, the heat transfer regime considered (conduction, convection, radiation) and the given thermal excitations. In this Chapter, only the case of heat conduction is considered.

### 4.3.3 Computation of the thermoelastic response

Once the temperature history is computed, using one of the methods presented in Section 4.2, finding the thermoelastic response of a structure essentially requires the evaluation of the generalized load vector defined by (4.46), as explained in Section 4.3, i.e.

$$\{\mathbb{W}\}^T \{\mathbb{F}(t)\} = \int_{\Omega_h} \tilde{T}_h(\cdot, t) \boldsymbol{\alpha} : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV, \quad \mathbf{w} \in \mathcal{C}_h(\mathbf{0}).$$

In practice, the evaluation of  $\{\mathbb{F}(t)\}$  is based on an assembly procedure similar to that, shown in Chapter 3, yielding a stiffness matrix or a nodal force vector. The above integration over  $\Omega_h$  results from summing the corresponding integrals contributed by all elements. Each element integral is evaluated using Gauss points (Section 3.2.5), with tensor variables written using the matrix Voigt notation (Section 3.1.4) and the temperature  $T_h(\cdot, t)$  expressed for each element in terms of nodal values and shape functions:

$$\tilde{T}_h(\mathbf{x}, t) = \{N(\mathbf{a})\}^T \{T_e(t)\} \quad \mathbf{x} \in E_e, \mathbf{x} \text{ and } \mathbf{a} \text{ connected by (4.8);}$$

finally, the strain tensor  $\varepsilon[\mathbf{w}]$  is expressed on each element in terms of the finite element interpolation by means of (3.12). The procedure may thus be summarized by the formula

$$\{\mathbb{W}\}^T \{\mathbb{F}(t)\} = \sum_{e=1}^{N_E} \{W_e\}^T \{F_e(t)\}, \quad (4.48)$$

where the components of element vectors  $\{F_e(t)\}$  and  $\{W_e\}$  are defined using local numbering, each element nodal force vector  $\{F_e(t)\}$  being given by

$$\begin{aligned} \{F_e(t)\} &= \int_{\Delta_e} ([B(\mathbf{a})]^T [A(\mathbf{a})] \{\alpha\}) [N(\mathbf{a})] \{T_e(t)\} J(\mathbf{a}) dV(\mathbf{a}) \\ &\approx \sum_{g=1}^G w_g ([B(\mathbf{a}_g)]^T [A(\mathbf{a}_g)] \{\alpha\}) [N(\mathbf{a}_g)] \{T_e(t)\} J(\mathbf{a}_g) \end{aligned} \quad (4.49)$$

with  $\{\alpha\}$  denoting the six-component vector expressing the tensor  $\alpha$  in Voigt notation, following the convention (3.9) for strains.

In practice, the summation in (4.48) is performed implicitly, by an assembly procedure similar to that introduced in Section 3.3.2 for other types of nodal force vectors.

For the case of isotropic elasticity and thermal dilatation, one has  $\{\alpha\} = \alpha \{1 \ 1 \ 1 \ 0 \ 0 \ 0\}^T$  and formula (4.49) takes the simpler form

$$\{F_e(t)\} \approx 3\alpha\kappa \sum_{g=1}^G w_g \{B_V(\mathbf{a}_g)\} (\{N(\mathbf{a}_g)\} \{T_e(t)\}) J(\mathbf{a}_g) \quad (4.50)$$

with

$$\text{tr} \varepsilon[\mathbf{w}] = \{B_V(\mathbf{a})\}^T \{W_e\} \quad \text{i.e.} \quad \{B_V(\mathbf{a})\} = [B(\mathbf{a})]^T \{1 \ 1 \ 1 \ 0 \ 0 \ 0\}^T.$$

**Remark on the thermal and mechanical interpolations.** All developments presented in this chapter implicitly assume that the meshes used for the mechanical and thermal analyses are such that their elements are geometrically coincident. Two remarks are, however, in order:

- (i) It is also possible to use meshes such that their elements are not geometrically coincident. In that case, the evaluation of the thermal load vector (4.46) requires a "projection" operation whose goal is to define the temperature field

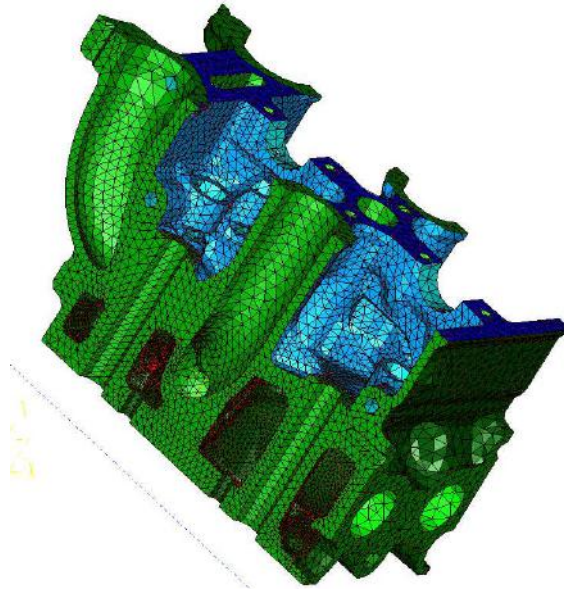
$\tilde{T}_h$  supported by the mechanical mesh which is closest (e.g. in the least-squares sense) to the field  $T_h$  supported by the thermal mesh.

- (ii) It is moreover advisable to employ interpolations for  $\mathbf{u}$  and  $T$  differing by one degree (for instance, considering plane thermoelasticity problems, one might employ triangular elements of type T6 (i.e. quadratic interpolation) for  $\mathbf{u}$  and T3 (i.e. linear interpolation) for  $T$ ), leading to *same-degree* representations for the strain  $\varepsilon[\mathbf{u}]$  and thermal strain  $\varepsilon^{\text{th}}$ .

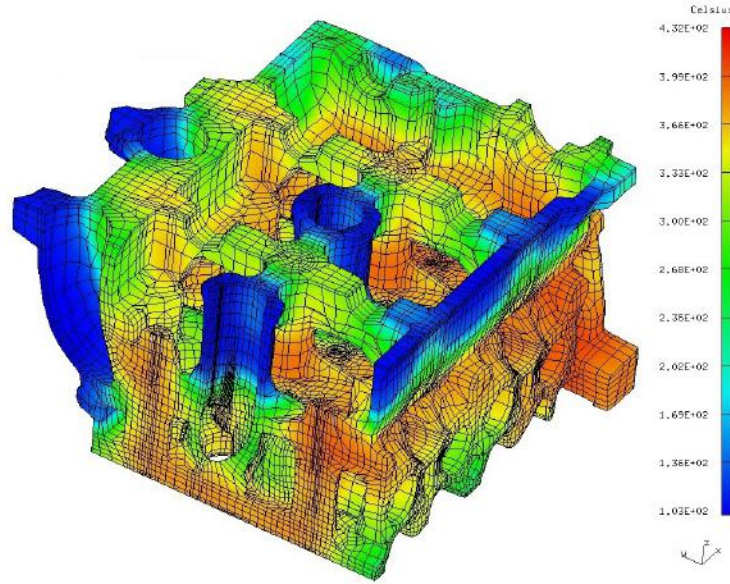
## 4.4 Applications and exercises

### 4.4.1 Industrial example: analysis of a diesel engine part

By way of illustration, an application of thermoelastic FEM analysis to a part of a cylinder head of a diesel automotive engine is now briefly shown (with kind permission of PSA Peugeot Citroën). The analysis is concerned with a zone of the cylinder head where cooling is effected by circulating water. Cracking by polycyclic fatigue is prone to occur in this zone. It is therefore necessary to simulate a design test where the cylinder head undergoes maximal pressure, taking into account the manufacturing process which includes a quenching phase. The mesh created for this analysis, and the simulated temperature field for a quenching stage (non-homogeneous cooling), are shown in Figures 4.6 and 4.7, respectively. The mean volumetric stress induced by the thermal load may then be computed from this thermal load, in order to determine the zones subjected to highest tensile stresses where cracking is most likely to occur.



**Figure 4.6:** Diesel engine part: mesh. With kind permission of PSA Peugeot Citroën.



**Figure 4.7:** Diesel engine part: temperature field during a quenching stage. With kind permission of PSA Peugeot Citroën.

#### 4.4.2 Example: heated bimaterial beam

Let us consider the bi-material beam of length  $L$  illustrated in figure 4.8 in plane strain conditions (Chap4\_beam\_bimat\_thermoel). The two homogeneous half-beams have thickness  $e$  and identical isotropic elastic parameters. Only the (isotropic) thermal dilatation coefficients differs:  $\alpha^+$ , for the upper half, and  $\alpha^-$  for the lower one, with  $\alpha^- > \alpha^+$ . Starting from an initial condition characterized by zero displacements and temperature  $T_0$ , the beam is progressively heated by  $\tilde{T}$  degrees. The analytical solution for the steady state condition is available in the simple case  $\nu = 0$ . Setting  $\Delta\alpha = \alpha^- - \alpha^+$  and  $\alpha_M = (\alpha^+ + \alpha^-)/2$ :

$$\mathbf{u} = \tilde{T}(\alpha_M x_1 - 3\Delta\alpha \frac{x_1 x_2}{4e})\mathbf{e}_1 + \tilde{T}(\alpha^\pm x_2 + 3\Delta\alpha \frac{x_1^2}{8e})\mathbf{e}_2 \quad (4.51)$$

Quadratic T6 triangles are employed to generate the mesh. Initially, only the steady state solution is considered, so that the exercise reduces to the implementation of (4.50) within genlin. The definition of the material variable in the analysis file must include also the  $\alpha$  coefficients which are added in third position after the Young modulus and the Poisson coefficient:

```
% MATERIAL: Young, Poisson and alpha
material= [1 0 2;
          1 0 1];
```

The element-level routine computing the equivalent thermoelastic nodal forces can be built starting from the procedure detailed in Section 3.2.6 and modifying the  $[B]$

matrix definition, as described in Box 4.1. The standard Gauss rule adopted (with three points) guarantees a complete integration only if the temperature distribution is linear.

**Box 4.1:** Equivalent thermoelastic nodal forces for the T6 triangle

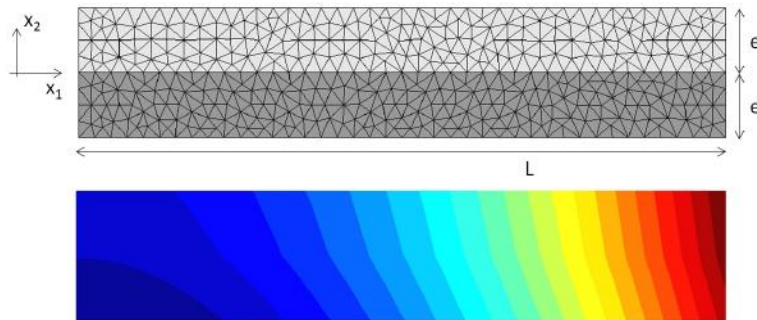
```
function Fe=T6_2A_solid_FTe(X,mate,Te)

E=mate(1);
nu=mate(2);
alpha=mate(3);

a_gauss=1/6*[4 1 1; 1 4 1; 1 1 4];           % Gauss abscissae
w_gauss=[1/6 1/6 1/6];                       % Gauss weights
Fe=zeros(12,1);
for g=1:3,                                     % loop over Gauss points
    a=a_gauss(g,:);                           % coordinates for gauss point
    N=[a(1)*(2*a(1)-1) a(2)*(2*a(2)-1) ...    % list of shape functions
        a(3)*(2*a(3)-1) 4*a(1)*a(2) ...
        4*a(2)*a(3) 4*a(1)*a(3)];
    T=N*Te';                                   % temperature at gauss point
    ... D computation ...
    G=D*invJ;                                 % gradient of shape functions
    B=[G(1,1) G(1,2) G(2,1) G(2,2) G(3,1) G(3,2) ...
        G(4,1) G(4,2) G(5,1) G(5,2) G(6,1) G(6,2)];
    Fe=Fe+alpha*E/(1-2*nu)*B'*T*detJ*w_gauss(g); % nodal forces
end
```

The nodal forces are then added to  $\{\mathbb{F}\}$  together with the other contributions during the assemblage. For a unit  $\bar{T}$  this becomes:

```
Te=ones(1,6);
Fe=eval([Etag Atag 'FThe(Xe,material(mat,:),Te)']);
F(Ie)=F(Ie)+Fe(Le0);
```



**Figure 4.8:** Thermoelastic response of a bi-material beam: mesh (top), contour map of  $u_2$  (bottom)

**Exercise 4.2** (heated bi-material beam: case of a uniform temperature field). *Make the required modifications to genlin and compare the numerical solution with the analytical formula (4.51).*

In a second phase of the exercise we propose to analyse the displacements due to the progressive heating of the beam (i.e. before the steady state has been reached). The left side of the beam is subjected, at time  $t = 0$ , to the Dirichlet boundary condition  $T_h = T_0 + \hat{T}$ , while  $q^D = 0$  is imposed elsewhere on the boundary.

**Exercise 4.3** (heated bi-material beam: case of a non-uniform temperature field). *First apply heatdiffusion for the analysis of the heat diffusion history until the time  $t_F$  of interest. Then save the computed nodal values on a file, using a procedure such as the following:*

```
vecT=zeros(analysis.NN,1)
for n=1:analysis.NN           % loop over the nodes
    vecT(n)=nodes(n).U(1);
end
save('temp.mat', 'vecT');
```

*Next read the temperature vector in genlin with:*

```
load('temp.mat', 'vecT');
```

*and apply the required changes in the assemblage of  $\{\mathbb{F}\}$  in order to account for the non uniform distribution of element nodal temperatures.*

#### 4.4.3 Example: thermoelasticity of underground caverns

We consider the analysis, under very simplified conditions, of a deep-buried cavity used for storing fluid products. The cavity  $\Omega_F$  is spherical with radius  $R_I$ , with the surrounding solid medium occupying a hollow spherical region  $\Omega_S$  bounded by the spheres of radii  $R_I$  and  $R_E > R_I$ . At initial time  $t = 0$ , the cavity is quickly filled with a fluid at temperature  $T_{F0}$ , while the solid is initially at an uniform temperature  $T_0 > T_{F0}$ . The thermoelastic coupled response of the fluid-solid system is to be studied, assuming radial symmetry for all quantities and under the usual small-deformation assumption.

**Coupled heat transfer problem.** In order to simplify the analysis, the temperature is assumed to be uniform in the fluid, i.e. equal to  $T_F(t)$ , while integrating the thermal balance equation (4.1) over the fluid domain  $\Omega_F$  reduces as a result to  $\rho_F c_F V_F \dot{T}_F = Q$ , where  $Q(t)$  is the incoming heat flux through the surface  $\partial\Omega_F$  and  $|\Omega_F| = V_F = 4/3\pi R_I^3$  is the volume of the spherical cavity.

Thermal continuity is assumed between the fluid and the solid, in the following sense: when the fluid gets in contact with the solid at time  $t = 0$ , the solid surface at  $r = R_I$  quickly reaches the fluid temperature, with  $T(R_I, t) = T_F(t)$  for any  $t > 0$ . For the continuous problem, there is a time instant  $t$  close to the initial time, here denoted  $t = 0^+$ , where the temperature distribution in the system is such that  $T_F(0^+) = T_{F0}$ ,  $T(R_I, 0^+) = T_{F0}$ ,  $T(r > R_I, 0^+) = T_0$ . This temperature distribution will be deemed (accepting a slight language inaccuracy) to constitute the initial conditions for the continuous problem.

The surface  $r = R_I$  is considered as subjected to a given heat flux. The thermal constitutive parameters of the (thermally isotropic) solid material are  $\rho, c, k$ . The external surface  $r = R_E$  is assumed to remain at initial temperature  $T_0$ . The thermal

balance for the solid region at a given time  $t > 0$  admits the weak formulation

$$4\pi \int_{R_I}^{R_E} \left( \rho c w \dot{T} + k \frac{\partial w}{\partial r} \frac{\partial T}{\partial r} \right) r^2 dr = -4\pi R_I^2 k \frac{\partial T}{\partial r} \Big|_{r=R_I} w(R_I) = -w(R_I)Q$$

$$\forall w : w(R_E) = 0$$

having set

$$Q = 4\pi R_I^2 k \frac{\partial T}{\partial r} \Big|_{r=R_I}$$

Finally, using the assumed relation between the heat flux and the fluid temperature, the weak formulation is

$$4\pi \int_{R_I}^{R_E} \left( \rho c w \dot{T} + k \frac{\partial w}{\partial r} \frac{\partial T}{\partial r} \right) r^2 dr = -w(R_I) \rho_F c_F V \dot{T}(R_I), \quad \forall w : w(R_E) = 0$$

The numerical solution of this problem is now considered. Thermal continuity between the fluid and solid domains remains assumed for the discretized problem, with  $T^{(1)} = T(R_I) = T_F$  for any  $t > 0$ . To determine a temperature distribution at time  $t = 0^+$  compatible with energy conservation and the discrete scheme, implicit time integration is first applied for the transition between time  $t = 0$  and  $t = 0^+$ , i.e. for a vanishing time step  $\Delta t$ . Under these conditions, only the terms involving capacities remain in the limiting form of the transition equation. Setting all nodal values of the test function  $w$  to zero except for  $w^{(1)} = 1$ , one finds

$$T^{(1)}(0^+) (\rho_F c_F V_F + M^{(1)}) = T_{F0} \rho_F c_F V + T_0 M^{(1)}$$

while all other linearly independent choices for the discrete test function  $w$  yield

$$T^{(i)}(0^+) = T^{(i)}(0), \quad i > 1$$

In the limit  $h \rightarrow 0$ , i.e. using an arbitrarily fine mesh, the conditions of the continuous scheme are recovered since  $M^{(1)} \rightarrow 0$ .

**Exercise 4.4.** Implement the above formulation by adapting the code developed in Section 4.2.5. Next compare with the analytical solution:

$$\frac{T_F(t) - T_0}{T_{F0} - T_0} = \frac{1+b}{2b} e^{(1+b)^2 \tau} \operatorname{erfc}[(1+b)\sqrt{\tau}] - \frac{1-b}{2b} e^{(1-b)^2 \tau} \operatorname{erfc}[(1-b)\sqrt{\tau}]$$

$$b^2 = \frac{\lambda - 4}{\lambda}, \quad \tau = \frac{\lambda^2 k t}{4 \rho c R^2} \quad \lambda = 3 \frac{\rho c}{\rho_F c_F} \quad (4.52)$$

**Coupled thermoelastic problem.** Let the fluid be treated as a fictitious homogeneous elastic solid with a bulk modulus  $\kappa_F$ , in which the stress state is spherical and spatially uniform, i.e.

$$\boldsymbol{\sigma} = -p \mathbf{I}$$

where the pressure  $p$  is constant. The constitutive thermoelastic properties of both the fluid and the solid are assumed linear and isotropic (quantities defined for the fluid will be labelled using the subscript "F", while those for the solid will appear without subscript). Set  $\tilde{T} := T - T_0$ ,  $\tilde{T}_F := T_F - T_{F0}$ . The general thermoelastic isotropic constitutive relation

$$\boldsymbol{\sigma} = \mathcal{A} : [\boldsymbol{\varepsilon} - \alpha \tilde{T} \mathbf{I}] = \lambda \operatorname{tr} \boldsymbol{\varepsilon} \mathbf{I} + 2\mu \boldsymbol{\varepsilon} - 3\kappa \alpha \tilde{T} \mathbf{I}$$



applied to the fluid domain yields

$$p = -\kappa_F \Delta V / V + 3\alpha_F \kappa_F \tilde{T}_F$$

Since, in the framework of small deformations, the volume variation is approximately given by  $\Delta V = 4\pi R_I^3 u(R_I)$ , one obtains

$$p = -\kappa_F \frac{3}{R_I} u(R_I) + 3\alpha_F \kappa_F \tilde{T}_F$$

The final coupled problem hence reads

Find  $u$  such that  $\forall w : w(R_E) = 0$

$$\begin{aligned} \int_{R_1}^{R_2} \left\{ (\lambda + 2\mu) \frac{du}{dr} \frac{dw}{dr} + 2\lambda \left( \frac{u}{r} \frac{dw}{dr} + \frac{w}{r} \frac{du}{dr} \right) + 4(\lambda + \mu) \frac{u}{r} \frac{w}{r} \right\} r^2 dr \\ - 3\alpha_F \kappa_F \int_{R_1}^{R_2} \left( \frac{dw}{dr} + 2\frac{w}{r} \right) \tilde{T} r^2 dr = R_I^2 P w(R_I) \quad (4.53) \\ R_I^2 P w(R_I) = -3\kappa_F R_I u(R_I) w(R_I) + 3R_I^2 \alpha_F \kappa_F \tilde{T}_F w(R_I) \end{aligned}$$

**Exercise 4.5.** Using the developments of Section 1.5 as a starting point, write a MATLAB function implementing (4.53). Assume that displacements are blocked at  $r = R_E$ . Compare with the analytical solution for a hollow sphere subjected to a given pressure on the internal surface.



## Dynamical analysis of elastic solids

---

This chapter is devoted to the numerical solution using finite elements of dynamical problems for linearly elastic solids, within the usual linearization framework based on the small-strain assumption.

Like in the previous chapter, solving the transient (dynamical) problem is based on two discretizations. Since the spatial semi-discretization is based on the finite element method in its "standard" form developed in Chapter 2, suitable for linear mechanical analyses, the main emphasis is placed here on the integration in time of the evolution equations. After reviewing the main concepts of elastodynamics (Section 5.1), the construction of the spatially semi-discretized problem using finite elements (Section 5.2) and modal analysis (Section 5.3), the two-parameter family of Newmark integration schemes, widely used for dynamical analyses, is introduced in Section 5.4. The consistency of these schemes is shown, and the conditions on the parameters and time step guaranteeing stability are established. The "explicit" central-difference scheme is also presented, first independently in view of its quite intuitive character (Section 5.4.2), then as one scheme of the Newmark family (Section 5.4.7). The important issue of estimating the critical time step is examined in Section 5.4.10. Next, Section 5.5 investigates the energy conservation properties of the Newmark schemes, an approach which provides alternative proofs and physical insight on their stability conditions (with instability being associated with divergence in discrete time of the total energy). Many of the foregoing concepts are demonstrated on the 1D example of an elastic rod. The last sections touch upon supplementary topics of interest (the spectral element method, Sec 5.7; the patch test, Sec 5.8.3) and present a numerical example involving dynamical contact (Sec 5.8.4).

### 5.1 General notions on the dynamics of elastic solids

#### 5.1.1 Assumptions and governing equations

A solid body, whose material is assumed to have linearly elastic constitutive properties, occupies the domain  $\Omega \subset \mathbb{R}^3$ . The usual small-deformation framework is assumed. The body is subjected to time-dependent loads applied through a given body force density, prescribed tractions on  $S_T \subset \partial\Omega$  and prescribed displacements on the complementary portion  $S_u = \partial\Omega \setminus S_T$  of the boundary. For simplicity, the boundary portions  $S_u$  and  $S_T$  are assumed not to evolve with time.

Dynamical conditions are assumed, that is, inertial effects are not considered negligible and are taken into account. The transient motion of the structure during the time interval  $t \in [0, t^F]$  is then governed by the following set of equations for the displacement, strain and stress  $\sigma(\mathbf{x}, t)$  fields, respectively denoted  $\mathbf{u}(\mathbf{x}, t)$ ,  $\varepsilon(\mathbf{x}, t)$ ,  $\sigma(\mathbf{x}, t)$ :

(i) Field equations:

$$\varepsilon = \frac{1}{2}(\nabla \mathbf{u} + \nabla^T \mathbf{u}) \quad \text{in } \Omega \times [0, t^F] \quad (\text{compatibility}), \quad (5.1a)$$

$$\operatorname{div} \sigma + \rho \mathbf{f} - \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} = \mathbf{0} \quad \text{in } \Omega \times [0, t^F] \quad (\text{balance}), \quad (5.1b)$$

$$\sigma = \mathcal{A} : \varepsilon \quad (\text{constitutive}), \quad (5.1c)$$

(ii) Boundary conditions:

$$\mathbf{u} = \mathbf{u}^D \quad \text{in } S_u \times [0, t^F] \quad (\text{prescribed displacements}), \quad (5.1d)$$

$$\mathbf{T} = \mathbf{T}^D \quad \text{in } S_T \times [0, t^F] \quad (\text{prescribed tractions}), \quad (5.1e)$$

(iii) Initial conditions:

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{U}_0(\mathbf{x}), \quad \text{in } \Omega \quad (\text{initial displacement}). \quad (5.1f)$$

$$\frac{\partial \mathbf{u}}{\partial t}(\mathbf{x}, 0) = \mathbf{V}_0(\mathbf{x}), \quad \text{in } \Omega \quad (\text{initial velocity}). \quad (5.1g)$$

### 5.1.2 Elastic waves

Equations (5.1a–g) have propagative solutions (i.e. define elastic waves), see e.g. Eringen and Suhubi (1975). The field equations (5.1a–c) may be combined to obtain, after elimination of strains and stresses, the governing equation for the displacement, known as the *Navier equation*:

$$\operatorname{div}(\mathcal{A} : \varepsilon[\mathbf{u}]) - \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} + \rho \mathbf{f} = \mathbf{0},$$

which is readily given the following, more explicit, form in the case of isotropic elasticity:

$$\mu \Delta \mathbf{u} + \frac{\mu}{1 - 2\nu} \nabla \operatorname{div} \mathbf{u} - \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} + \rho \mathbf{f} = \mathbf{0}. \quad (5.2)$$

**Lamé potentials.** Any solution of the elastodynamic Navier equation (5.2) may be represented in the form

$$\mathbf{u} = \nabla \phi_L + \operatorname{rot} \phi_T, \quad \text{with } \operatorname{div} \phi_T = 0, \quad (5.3)$$

in terms of the *Lamé potentials*  $\phi_L$  (scalar) and  $\phi_T$  (vector). In fact, any sufficiently smooth vector field has a representation in terms of Lamé potentials. For example, the body force density may be recast in the form

$$\mathbf{f} = \nabla f_L + \operatorname{rot} \mathbf{f}_T.$$

**Compressional and shear waves.** Introducing the Lamé decomposition (5.3) into the Navier equation (5.2) leads, with the help of identity

$$\Delta \mathbf{w} = \nabla \operatorname{div} \mathbf{w} + \operatorname{rot} \operatorname{rot} \mathbf{w}$$

to the following uncoupled governing equations for the potentials  $\phi_L$  and  $\phi_T$ :

$$\Delta \phi_L - \frac{1-2\nu}{2} \frac{\rho}{\mu} \frac{\partial^2 \phi_L}{\partial t^2} + \mathbf{f}_L = 0, \quad \Delta \phi_T - \frac{\rho}{\mu} \frac{\partial^2 \phi_T}{\partial t^2} + \mathbf{f}_T = 0. \quad (5.4)$$

Each Lamé potential is seen to satisfy a wave equation. Displacements of the form  $\mathbf{u}_L = \nabla \phi_L$  are *compressional waves* (also sometimes referred to as *longitudinal waves*); they propagate with velocity  $c_L$ , with

$$c_L^2 = \frac{2-2\nu}{1-2\nu} \frac{\mu}{\rho}.$$

Displacements of the form  $\mathbf{u}_T = \operatorname{rot} \phi_T$  are *shear waves* (also sometimes referred to as *transversal waves*); they propagate with velocity  $c_T$ , with

$$c_T^2 = \frac{\mu}{\rho}.$$

The compressional and shear velocities verify the inequality  $c_L > c_T$ .

### 5.1.3 Validity conditions for the quasi-static approach

This chapter is concerned with computing the dynamical response of an elastic solid, and hence addresses situations where inertia effects cannot be neglected, i.e. such that quasi-static approximations are not valid.

The above-established wave velocities allow to define more precisely the notion of "(non-)negligible inertia effect". Indeed, letting  $L$  and  $T$  denote characteristic length and time scales for the problem at hand (for instance,  $L$  is a characteristic linear dimension of  $\Omega$  and  $T$  is the time scale for the variation in time of the applied loading, see the comment following Exercise 5.1 in Section 5.3 for an example), the Navier equation (5.2) may be reformulated in terms of non-dimensional coordinates  $\tilde{\mathbf{x}} = \mathbf{x}/L$  and  $\tilde{t} = t/T$ :

$$\operatorname{div}_{\tilde{\mathbf{x}}} \left( \frac{1}{\mu} \mathcal{A} : \varepsilon_{\tilde{\mathbf{x}}}[\mathbf{u}] \right) - \frac{\rho L^2}{\mu T^2} \frac{\partial^2 \mathbf{u}}{\partial \tilde{t}^2} + \frac{\rho L^2}{\mu} \mathbf{f} = 0.$$

Inertia effects may then be neglected (i.e. a quasi-static approximation is reasonable) if

$$\frac{\rho L^2}{\mu T^2} \ll 1.$$

Taking into account the definition of the velocity  $c_T$ , the above condition may be recast in the form

$$\left( \frac{L}{c_T T} \right)^2 \ll 1, \quad (5.5)$$

which is easier to interpret:

- *The quasi-static assumption is applicable when the characteristic length  $L$  is much smaller relative to the distance  $c_T T$  travelled by an elastic wave during the characteristic time  $T$ .*

When (5.5) does not hold, the response of the structure must be treated as dynamical, and falls within the scope of this chapter.

## 5.2 Weak formulation and finite element semi-discretization

### 5.2.1 Weak formulation

The local dynamical balance equation (5.1b) may be recast into an equivalent weak form by means of weighted residuals, i.e. by taking its scalar product with an arbitrary test function  $\mathbf{w} \in \mathcal{C}$  and integrating over  $\Omega$  the resulting identity. One obtains

$$\int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV + \int_{\Omega} \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} \cdot \mathbf{w} \, dV = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{\partial\Omega} [\boldsymbol{\sigma} \cdot \mathbf{n}] \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}, \quad (5.6)$$

which corresponds to the virtual work principle for the virtual field  $\mathbf{w}$ . Using the constitutive equation (5.1c) and the local compatibility (5.1a) in the first integral of (5.6) and invoking the boundary condition (5.1e) in the third, and restricting the resulting identity to virtual fields that are compatible with homogeneous data on  $S_u$ , one arrives at

$$\int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV + \int_{\Omega} \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} \cdot \mathbf{w} \, dV = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (5.7)$$

Finally, accounting for the prescribed displacement (5.1d) and initial conditions (5.1f,g) in (5.7), one arrives at the governing displacement-based weak formulation for the dynamical response of an elastic solid:

$$\begin{aligned} & \text{find } \mathbf{u}(\mathbf{x}, t) \in \mathcal{C}(\mathbf{u}^D) \text{ such that} \\ & \int_{\Omega} \boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV + \int_{\Omega} \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} \cdot \mathbf{w} \, dV \\ & \quad = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} \quad (\forall t \in [0, t^F], \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0})), \quad (5.8) \\ & \mathbf{u}(\mathbf{x}, 0) = \mathbf{U}_0(\mathbf{x}), \quad \frac{\partial \mathbf{u}}{\partial t}(\mathbf{x}, 0) = \mathbf{V}_0(\mathbf{x}) \quad (\mathbf{x} \in \Omega). \end{aligned}$$

### 5.2.2 Finite element semi-discretization

Introducing in the weak formulation (5.8) a finite element approximation  $\Omega_h$  of  $\Omega$  and  $\mathbf{u}_h$  of  $\mathbf{u}$ , and using the virtual fields  $\mathbf{w} \in \mathcal{C}_h(\mathbf{0})$  associated with this discretization, one arrives (with notation as in Chapters 2 and 3) at the following

linear system of ordinary differential equations

$$[\mathbb{K}]\{\mathbb{U}(t)\} + [\mathbb{M}]\{\ddot{\mathbb{U}}(t)\} = \{\mathbb{F}(t)\}, \quad (5.9)$$

where  $\{\mathbb{U}(t)\}$  and  $\{\ddot{\mathbb{U}}(t)\}$  denote N-vectors gathering the nodal displacements and accelerations, respectively, that remain unknown after having accounted for the kinematic boundary conditions (5.1d). The  $N \times N$  stiffness matrix  $[\mathbb{K}]$  and the N-vector of generalized nodal forces  $\{\mathbb{F}(t)\}$  are defined and computed as in Chapter 3, the only difference being that  $\{\mathbb{F}(t)\}$  is now time-dependent.

**Mass matrix.** Relative to the quasistatic case, the linear system (5.9) involves a new object, namely the *mass matrix*  $[\mathbb{M}]$ , which is defined through the equality

$$\{\mathbb{W}\}^T [\mathbb{M}] \{\ddot{\mathbb{U}}(t)\} = \int_{\Omega_h} \rho \frac{\partial^2 \mathbf{u}_h}{\partial t^2} \cdot \mathbf{w} \, dV, \quad \mathbf{w} \in \mathcal{C}_h(\mathbf{0}). \quad (5.10)$$

The mass matrix is associated with the kinetic energy of the structure, given by

$$\mathcal{K}(t) \stackrel{\text{def}}{=} \frac{1}{2} \int_{\Omega} \rho \left\| \frac{\partial \mathbf{u}}{\partial t} \right\|^2 dV,$$

or more precisely to its value as defined for the semi-discretized model:

$$\mathcal{K}_h(t) = \frac{1}{2} \int_{\Omega_h} \rho \left\| \frac{\partial \mathbf{u}_h}{\partial t} \right\|^2 dV = \frac{1}{2} \{\dot{\mathbb{U}}(t)\}^T [\mathbb{M}] \{\dot{\mathbb{U}}(t)\}. \quad (5.11)$$

The mass matrix is clearly positive definite, since any nonzero vector of nodal velocities  $\{\dot{\mathbb{U}}(t)\}$  corresponds to a nonzero velocity field  $\partial \mathbf{u}_h / \partial t$ , and hence to a strictly positive value of the kinetic energy. The matrix  $[\mathbb{M}]$  can moreover easily be shown to be symmetric.

The numerical computation of  $[\mathbb{M}]$  proceeds as usual by computing the element mass matrices:

$$\{\mathbb{W}\}^T [\mathbb{M}] \{\ddot{\mathbb{U}}(t)\} = \sum_{e=1}^{N_E} \{W_e\}^T [M_e] \{\ddot{U}_e(t)\}, \quad (5.12)$$

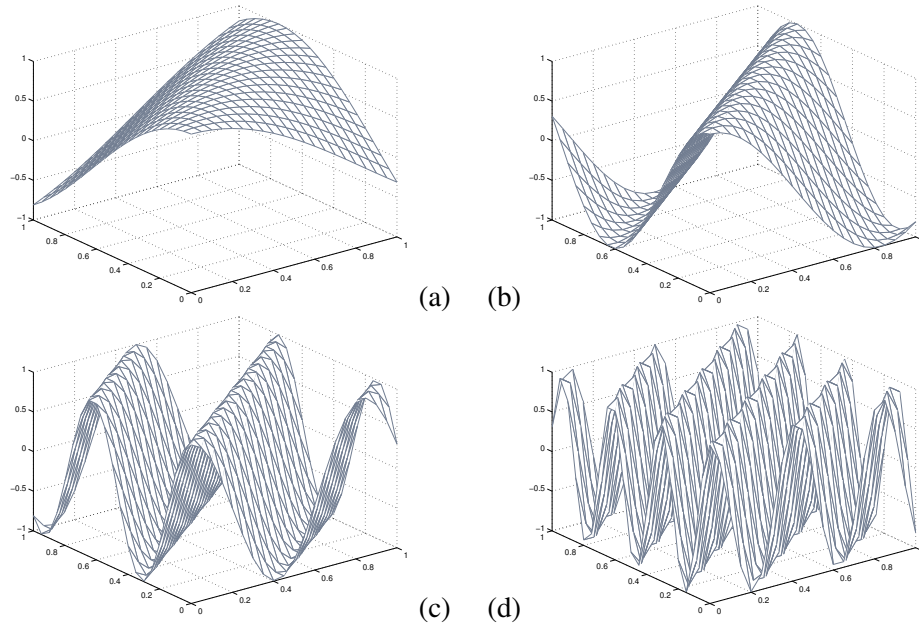
and performing an assemblage operation, similarly to relation (3.30a) for the stiffness matrix  $[\mathbb{K}]$ . Each element mass matrix  $[M_e]$  results from the equality

$$\{W_e\}^T [M_e] \{\ddot{U}_e(t)\} = \{W_e\}^T \left\{ \int_{\Delta_e} \rho [N(\mathbf{a})]^T [N(\mathbf{a})] J(\mathbf{a}) \, dV(\mathbf{a}) \right\} \{\ddot{U}_e(t)\}, \quad (5.13)$$

with the matrix  $[N(\mathbf{a})]$  of element shape functions defined according to the convention (2.23).

### 5.2.3 Mesh size and wavelength

The finite element method implements the same concepts for static and dynamical analyses: mesh, shape functions, setup of a linear system of equations. However,



**Figure 5.1:** Interpolation of a plane wave on a square domain of unit side, using a regular mesh of  $20 \times 20$  square  $Q4$  elements, for four values of the wavelength  $\ell$ :  $\ell = 1$  (a),  $\ell = 2$  (b),  $\ell = 4$  (c),  $\ell = 8$  (d).

criteria for choosing the mesh, and in particular the mesh size  $h$  that is required for achieving a given accuracy, are strongly influenced by the dynamical context.

For (quasi)static analyses, the main criteria for selecting the mesh size  $h$  are the characteristic size of the smallest geometrical features that need accurate representation, and the expected scale of strong spatial variations of fields (e.g. large gradients near high-curvature boundaries, singularities along crack edges...).

The foregoing criteria still apply for dynamical analyses, but are no longer sufficient. Indeed, the known elastic wave velocities  $c_L, c_T$  (Section 5.1.2) imply a new characteristic length  $\ell = c_L T$  or  $\ell = c_T T$ , which does not depend on the structure geometry and represent the distance travelled by a wave during the characteristic time duration  $T$ . For example, if a dynamical loading period  $T$  is applied to the structure,  $c_L T$  et  $c_T T$  are the compressional and shear wavelengths. It is then important that the mesh be designed so as to ensure a sufficiently accurate representation of fields whose spatial variation occurs over a length scale of the order of  $\ell$ . Dividing  $\ell$  by 2 (e.g. by doubling the excitation frequency) requires halving the mesh size  $h$  (and hence 4 or 8 times as many elements, respectively, for 2D or 3D analyses). To illustrate this remark, Figure 5.1 shows the interpolation of a plane wave on a unit square meshed with  $20 \times 20$  four-noded square elements, for four decreasing values  $\ell = 1, 1/2, 1/4, 1/8$  of the wavelength. The representation of the wave by the *fixed* mesh is clearly seen to deteriorate as the wavelength decreases.



### 5.3 Modal analysis

#### 5.3.1 Modes and generalised displacements

Let  $\omega_I^2$  and  $\{\mathbb{X}^I\}$  ( $1 \leq I \leq N$ ) respectively denote the eigenvalues and eigenvectors associated with the generalized eigenvalue problem

$$[\mathbb{K}]\{\mathbb{X}\} - \omega^2[\mathbb{M}]\{\mathbb{X}\} = \{0\} \quad (5.14)$$

which in fact governs the free vibrations of the structure. The mass matrix  $[\mathbb{M}]$  being symmetric positive definite (and hence in particular invertible), this eigenvalue problem, equivalent to  $[\mathbb{M}]^{-1/2}[\mathbb{K}][\mathbb{M}]^{-1/2}\{\mathbb{Y}\} - \omega^2\{\mathbb{Y}\} = \{0\}$ , is well-posed and yields a simultaneous diagonalization of the matrices  $[\mathbb{M}]$  and  $[\mathbb{K}]$ , with  $\omega_I^2 > 0$  for all  $I$ , provided boundary conditions prevent any rigid-body motion. The eigenvalues are conventionally ordered from the smallest to the largest. The eigenvectors  $\{\mathbb{X}^I\}$ , known as *modes* of the structure, can be chosen so as to define a  $[\mathbb{M}]$ -orthonormal and  $[\mathbb{K}]$ -orthogonal basis of  $\mathbb{R}^N$ , that is:

$$\begin{aligned} \{\mathbb{X}^I\}^T[\mathbb{M}]\{\mathbb{X}^I\} &= 1, & \{\mathbb{X}^I\}^T[\mathbb{M}]\{\mathbb{X}^J\} &= 0 \quad (J \neq I), \\ \{\mathbb{X}^I\}^T[\mathbb{K}]\{\mathbb{X}^I\} &= \omega_I^2, & \{\mathbb{X}^I\}^T[\mathbb{K}]\{\mathbb{X}^J\} &= 0 \quad (J \neq I). \end{aligned}$$

Then, let the displacement and acceleration DOF vectors be expanded on the  $\{\mathbb{X}^I\}$  basis according to

$$\{\mathbb{U}(t)\} = \sum_{J=1}^N u^J(t)\{\mathbb{X}^J\}, \quad \{\ddot{\mathbb{U}}(t)\} = \sum_{J=1}^N \ddot{u}^J(t)\{\mathbb{X}^J\} \quad (5.15)$$

where  $u^J(t)$  are often called generalised displacements. Substituting the above expansions into (5.9), left-multiplying the resulting equality by  $\{\mathbb{X}^I\}^T$  for some fixed  $I$  and exploiting the  $[\mathbb{M}]$ -orthonormality and  $[\mathbb{K}]$ -orthogonality properties of the eigenvectors, (5.9) yields  $N$  uncoupled linear equations:

$$\ddot{u}^I(t) + \omega_I^2 u^I(t) = \{\mathbb{X}^I\}^T\{\mathbb{F}(t)\} =: f^I(t) \quad 1 \leq I \leq N \quad (5.16)$$

The general solution of (5.16) is given by the convolution integral:

$$u^I(t) = \frac{1}{\omega_I} \int_0^t f^I(\tau) \sin \omega_I(t - \tau) d\tau + c_I \sin \omega_I t + d_I \cos \omega_I t \quad (5.17)$$

where  $c_I$  and  $d_I$  depend on initial conditions.

**Exercise 5.1** (dynamic vs. quasi-static conditions). *Show that if  $f^I(t) = \sin \omega_0 t$  and  $u^I(0) = \dot{u}^I(0) = 0$  then:*

$$u^I(t) = \frac{1/\omega_I^2}{1 - \omega_0^2/\omega_I^2} \sin \omega_0 t \quad (5.18)$$

*Then  $u^I(t)$  coincides with the quasi-static response when  $\omega_0 \ll \omega_I$  and  $u^I(t)$  vanishes when  $\omega_0 \gg \omega_I$*

Without loss of generality, if the external loading depends everywhere on the same time function then  $\{\mathbb{F}(t)\} = \{\mathbb{F}\}g(t)$  and we set  $f^I = \{\mathbb{X}^I\}^T \{\mathbb{F}\}$ , equation (5.16) simplifies to:

$$\ddot{u}^I(t) + \omega_I^2 u^I(t) = f^I g(t) \quad 1 \leq I \leq N \quad (5.19)$$

Let  $\mathcal{F}(g)$  denote the Fourier transform of  $g(t)$  and assume that the frequency content of  $\mathcal{F}(g)$  is limited, in the sense that

$$|\mathcal{F}(g)| \simeq 0, \quad \omega < \omega_{\min}, \quad \text{and} \quad \omega > \omega_{\max}$$

Then, from the example above it can be concluded that the response will be quasi-static if  $\omega_{\max} \ll \omega_1$ . Let the characteristic time  $T$  be defined as  $T = 2\pi/\omega_{\max}$ , i.e. chosen as the shortest period associated with the frequency content of the loading. Then, criterion (5.5) amounts to assuming that the characteristic length  $L$  is much smaller than the wavelength corresponding to  $T$ .

Otherwise (5.16) is very effective for the solution of the system dynamics. Unfortunately, the computation of the whole set of eigenvalues and eigenvectors is extremely expensive for large  $N$ , so that a trade-off is generally sought. In structural dynamics one often encounters a condition where only few eigenvalues, say  $\bar{N}$ , are in the order of (or smaller than)  $\omega_{\max}$ . Hence (5.16) is applied only for  $1 \leq I \leq \bar{N}$ , while it is assumed that the other modes will respond quasi-statically. The displacement nodal vector is decomposed as follows:

$$\{\mathbb{U}(t)\} = \sum_{J=1}^{\bar{N}} u^J(t) \{\mathbb{X}^J\} + \{\Delta \mathbb{U}(t)\} \quad (5.20)$$

where

$$\{\Delta \mathbb{U}(t)\} := \sum_{J=\bar{N}+1}^N u^J(t) \{\mathbb{X}^J\}$$

represents the *quasi-static response* of higher order modes. Let us define the vectors  $\{\mathbb{Y}^I\} = [\mathbb{M}] \{\mathbb{X}^I\}$  having the property that  $\{\mathbb{Y}^I\}^T \{\mathbb{X}^J\} = \delta_{IJ}$ . The vectors  $\{\mathbb{Y}^I\}$  are the elements of the dual basis of  $\{\mathbb{X}^J\}$  in  $\mathbb{R}^N$  and, since  $f^I$  is the projection of  $\{\mathbb{F}\}$  on  $\{\mathbb{X}^I\}$ :

$$\{\mathbb{F}\} = \sum_{I=1}^{\bar{N}} f^I \{\mathbb{Y}^I\} + \sum_{I=\bar{N}+1}^N f^I \{\mathbb{Y}^I\} = \sum_{I=1}^{\bar{N}} f^I [\mathbb{M}] \{\mathbb{X}^I\} + \{\Delta \mathbb{F}\} \quad (5.21)$$

with the property that  $\{\mathbb{X}^I\}^T \{\Delta \mathbb{F}\} = 0$ ,  $1 \leq I \leq \bar{N}$ . Using (5.20) and (5.21), it is left as an exercise to show that  $u^I(t)$ ,  $1 \leq I \leq \bar{N}$ , is still defined by (5.16), while  $\{\Delta \mathbb{U}(t)\}$  can be directly computed from the solution of the linear system:

$$[\mathbb{K}] \{\Delta \mathbb{U}(t)\} = \{\mathbb{F}(t)\} - \sum_{I=1}^{\bar{N}} f^I(t) [\mathbb{M}] \{\mathbb{X}^I\} \quad (5.22)$$

The benefit of this procedure is that it requires only the computation of the first  $\bar{N}$  modes, which can be done efficiently if  $\bar{N}$  is small. On the contrary, if most of

the modes are activated, it might be more effective to resort to direct integration approaches as discussed later in this chapter.

**Exercise 5.2** (Rayleigh damping). *Real materials and structures normally undergo attenuation, prompting the need to model damping. A model that is frequently used for its ease of implementation, in spite of lacking a clear physical meaning, is the so-called Rayleigh damping where the structural dynamics obeys*

$$[\mathbb{M}]\{\ddot{\mathbf{U}}(t)\} + [\mathbb{C}]\{\dot{\mathbf{U}}(t)\} + [\mathbb{K}]\{\mathbf{U}(t)\} = \{\mathbf{F}(t)\},$$

with the damping matrix  $[\mathbb{C}]$  set in the form  $[\mathbb{C}] = \alpha[\mathbb{M}] + \beta[\mathbb{K}]$  for some  $\alpha, \beta > 0$ . Use diagonalisation to show that the counterpart of (5.19) is

$$\ddot{u}^I(t) + (\alpha + \beta\omega_I^2)\dot{u}^I(t) + \omega_I^2 u^I(t) = f^I g(t) \quad 1 \leq I \leq N.$$

### 5.3.2 Example: elastic rod

To demonstrate the main ideas on an example, consider an elastic rod (length  $L$ , Young modulus  $E$ , cross-section area  $S$ , mass density  $\rho S$ ). The displacement  $u(x, t)$  along the axis direction of the section  $S(x)$  located at abscissa  $x$  is governed, using the elongated elastic rod model (see e.g. G  rardin and Rixen, 1997), by the partial differential equation

$$\frac{\partial^2 u}{\partial x^2}(x, t) - \frac{1}{c^2} \frac{\partial^2 u}{\partial t^2}(x, t) = 0, \quad (5.23)$$

which is in fact the one-dimensional wave equation, with the wave velocity in the elastic rod given by

$$c = \sqrt{E/\rho}. \quad (5.24)$$

The following conditions are assumed: the rod is at initial rest, its left end  $x = 0$  is clamped, and its right end  $x = L$  is subjected to a constant force  $P = T^D S$  applied from the initial time  $t = 0$ :

$$u(x, t) = \frac{\partial u}{\partial t}(x, t) = 0, \quad (0 \leq x \leq L) \quad (\text{initial conditions})$$

$$u(0, t) = 0 \quad E \frac{\partial u}{\partial x}(L, t) = T^D, \quad (t \geq 0) \quad (\text{boundary conditions})$$

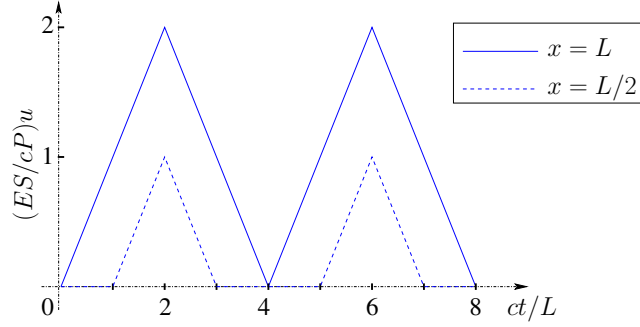
This problem has a known exact solution, shown as a function of time for two values of  $x$  in Figure 5.2.

**Weak formulation.** The weak form of equation (5.23) is found by multiplying it by a test function  $w \in \mathcal{C}(0)$  such that  $w(0) = 0$  and integrating the resulting equality over  $0 \leq x \leq L$ . On performing an integration by parts and invoking the boundary conditions, one finds

$$\int_0^L \left( E \frac{\partial u}{\partial x} \frac{\partial w}{\partial x} + \rho \frac{\partial^2 u}{\partial t^2} w \right) dx = T^D w(L), \quad \forall w \in \mathcal{C}(0), \quad (5.25)$$

The MATLAB implementation, as given in `bar_B2_1D_wave_modal`, follows a structure similar to that previously used for spherically-symmetric examples (Secs. 1.5, 2.1 and 4.2.5). Accordingly, the rod is divided into  $N_E$  finite elements of equal length

$h$  and the contribution of each element to the weak formulation (5.25) is evaluated, the kinematically-admissible approximation space  $\mathcal{C}_h(0)$  being defined as the set of piecewise-linear and continuous functions on  $x \in [0, L]$  that vanish at  $x = 0$ .



**Figure 5.2:** Dynamical response of a rod with a step loading applied to its right end: exact displacement solution  $u(x, t)$  (the curves show  $t \mapsto u(x, t)$  for  $x = L/2$  and  $x = L$ ).

**Element stiffness matrix.** A generic element of length  $h$ , whose nodes are its end-points with axial coordinates  $x^{(1)}, x^{(2)}$  and supporting nodal displacements  $u^{(1)}, u^{(2)}$ , is considered. The element stiffness matrix  $[K_e]$  arises through expressing its contribution to the elastic strain energy:

$$\int_{x^{(1)}}^{x^{(2)}} E \frac{\partial u_h}{\partial x} \frac{\partial w}{\partial x} dx = \{W_e\}^T \underbrace{\frac{E}{h^2} \left( \int_0^h \begin{bmatrix} (N'_1)^2 & N'_1 N'_2 \\ N'_2 N'_1 & (N'_2)^2 \end{bmatrix} ds \right)}_{[K_e]} \{U_e\}, \quad (5.26)$$

Using a linear interpolation over each element,  $[K_e]$  is therefore simply given by

$$[K_e] = \frac{E}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix},$$

and the MATLAB function returning  $Ke$  given the element nodal coordinates and the Young's modulus is:

```
function Ke=B2_1D_wave_Ke(X,E)

x1=X(1); x2=X(2);
h=x2-x1;                                     % element length
Ke=E/h*[1,-1;-1,1];                         % element stiff. matrix
```

**Element mass matrix.** The consistent element mass matrix  $[M_e]$  likewise arises through expressing its contribution to the generalized inertial forces:

$$\int_{x^{(1)}}^{x^{(2)}} \rho w \frac{\partial^2 u_h}{\partial t^2} ds = \{W_e\}^T \underbrace{\left( \int_0^h \begin{bmatrix} N_1^2 & N_1 N_2 \\ N_2 N_1 & N_2^2 \end{bmatrix} ds \right)}_{[M_e]} \{\ddot{U}_e\}. \quad (5.27)$$

The MATLAB function returning the element mass matrix  $M_e$  is then:

```
function Me=B2_1D_wave_Ke(X,rho)

x1=X(1); x2=X(2);
h=x2-x1;
Me=rho*h/6*[2,1;1,2];
```

For simplicity, only the computation of the consistent mass matrix is presented. An example of mass lumping is shown later on a 2D example (Section 5.8.4).

**Modal decomposition and time integration.** We now implement formula (5.20), commenting the file `bar_B2_1D_wave_modal` of the distribution. Let fictitious parameters be defined as

```
L=1; % bar length
NE=20; % number of elements
rho=1.; % density
E=1.; % Young modulus
TD=1; % value of traction x=L
tf=10; % final time
```

Since assembling the mass and stiffness matrices and the vector of nodal forces does not require any novel idea, this step is skipped. We fix the number  $\bar{N} \rightarrow bN$  of retained modes and solve the generalised eigenvalue problem (5.14) with the MATLAB function `eigs`. The eigenvectors are stored in the columns of  $V$  and the eigenvalues are the diagonal elements of  $D$ . Next the coefficients  $f^I$  are computed by projection

```
bN=4;
[V,D]=eigs(K,M,bN,'SM'); % eigenvalues and eigenvectors
f=V'*F;
```

The convolution integral in (5.17) can be performed analytically, since  $f^I(t) = f^I H(t)$ , and initial conditions fix  $c_I = d_I = 0$ :

$$u^I = (1 - \cos(\omega_I t)) f^I / \omega_I^2$$

and the static correction (5.22) does not depend on time and can be computed once for all. The displacement history is finally computed at the point of interest for a selection of time knots

```
DF=F-M*V*f;
Du=K\DF;

Dt=0.1;
nstep=floor(tf/Dt); % number of time steps with constant dt

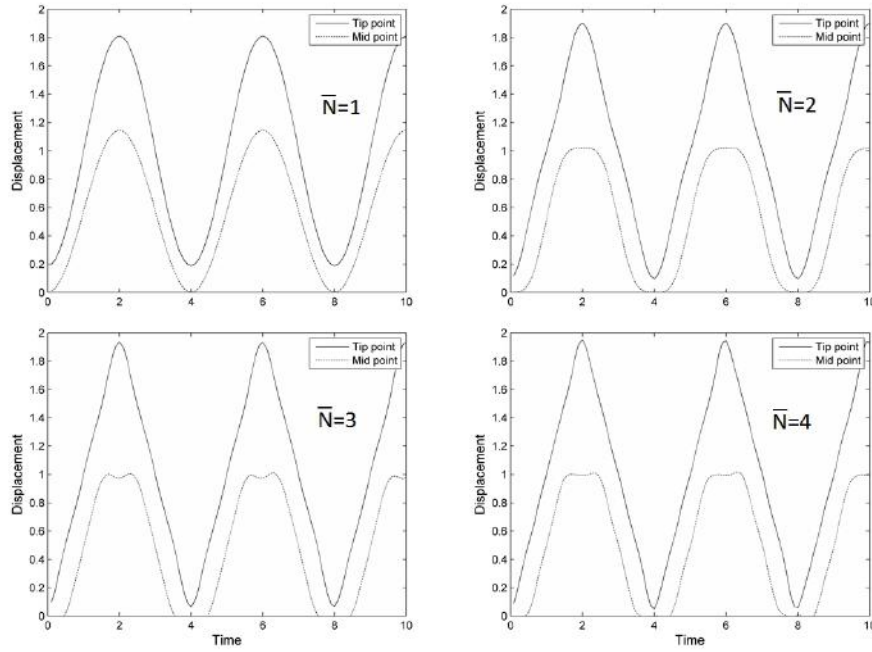
D=diag(D);
U=zeros(bN,1);

for step=1:nstep, % loop over time steps
    t=Dt*step;
    U=f./D.*(1-cos(sqrt(D)*t)); % analytical integration
    tipdisp=V(neq,:)*U+Du(neq); % tip displacement
end
```

With the parameters chosen, the first modes have the following eigenvalues:

$$\omega_1 = 1.57, \quad \omega_2 = 4.72, \quad \omega_3 = 7.9, \quad \omega_4 = 11.13, \quad \omega_5 = 14.43, \quad \dots$$

while the Fourier transform of  $H(t)$  decays as rapidly as  $1/\omega$ . In figure 5.3 it can be appreciated that the displacement history rapidly converges as a function of the number  $\bar{N}$  of modes considered, which means that the modes neglected respond quasi-statically up to a very good approximation.



**Figure 5.3:** Displacements at the middle and at the tip of the rod computed from modal decomposition and static correction using an increasing number of modes

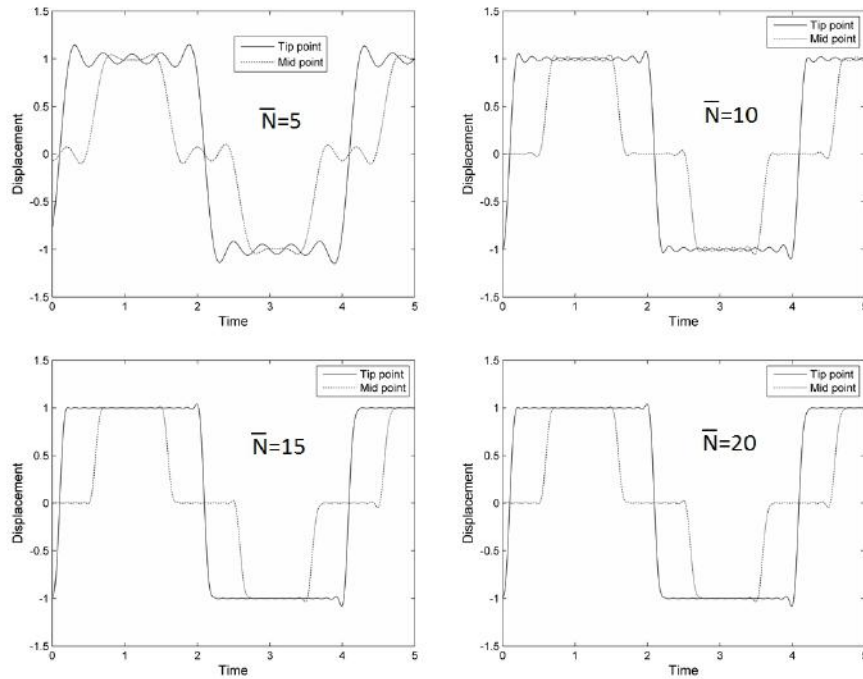
**Blast loading.** Let us now replace the rather smooth loading  $T^D H(t)$  with a blast impulse loading  $T^D \delta(t)$ , numerically simulated as a continuous hat shaped function of height  $\Delta$  and support  $2\Delta$ . Fourier transforms suggest that we can fix  $\omega_{\max} \simeq 2\pi/\Delta$ , as a first approximation. The choice of the size of the elements depends on the frequency content of the applied traction. The maximum meaningful frequency activated in the structure will hence be of the order of  $\omega_{\max}$ . Now, following the discussion of Section 5.2.3, we require that the minimum characteristic length  $\ell \simeq c2\pi/\omega_M$  be meshed with at least  $N_L$  elements, yielding:

$$h \leq \frac{2\pi c}{N_L \omega_{\max}} = \frac{2\pi}{N_L \omega_{\max}} \sqrt{E/\rho} \quad (5.28)$$

If we set  $N_L = 10$  (ten linear elements for each wave), then the maximum element length in our case should be  $h \simeq \Delta/10$ . Moreover, the time convolution yields:

$$u^I = \frac{f^I}{\omega_1^3 \Delta^2} (\sin(\omega_1 t) - 2\sin(\omega_1(t - \Delta)) + \sin(\omega_1(t - 2\Delta)))$$

The results are collected in figure 5.4 for  $\Delta = 0.1, h = 1/100$ , and show, as expected, that the number of modes required to properly represent the structure response increases with respect to the Heaviside type of loading. In order to better appreciate the results it is worth recalling that  $\omega_{10} \simeq 30$ . In such a case the direct integration approaches discussed in the following sections should probably be preferred.



**Figure 5.4:** Blast loading: Displacements at the middle and at the tip of the rod computed from modal decomposition using an increasing number of modes

## 5.4 Direct integration algorithms: the Newmark family

To complete the finite element approximation of transient dynamics, an algorithmic treatment of the integration in time of equations (5.1a-g) is required. This treatment is of a time-stepping nature, similarly to that used for solving heat conduction problems in Chapter 4.

### 5.4.1 Time discretization

The computation of the dynamical response governed by equations (5.1a) to (5.1g) is based on solving the linear system of ordinary differential equations (5.9), wherein each nodal unknown is time-dependent. A vast array of algorithms are available for the approximate solution of equations (5.9). Like their counterparts used for other evolution problems in Chapters 4 and 8, they are based on sampling the time interval of interest  $[0, t^F]$  by means of a discrete sequence of  $M + 1$  time instants  $t_m$   $0 \leq m \leq M$ :

$$t_0 = 0, \quad t_1 = \Delta t, \dots, t_m = m\Delta t, \dots, \quad t_M = M\Delta t = t^F. \quad (5.29)$$

The constant  $\Delta t$  is thus  $\Delta t = t^F/M$  (more sophisticated methods using a variable time step are also available, but are not considered in this introductory text). Time integration algorithms then proceed in time-stepping fashion: starting from initial data  $(\{\mathbb{U}_0\}, \{\dot{\mathbb{U}}_0\})$ , the quantities  $(\{\mathbb{U}_n\}, \{\dot{\mathbb{U}}_n\}, \{\ddot{\mathbb{U}}_n\})$  are incrementally computed for each discrete time instant  $t_n$  ( $n = 1, 2, \dots$ ). The essential component of such time-stepping methods is therefore the procedure achieving the transition

$$(\{\mathbb{U}_n\}, \{\dot{\mathbb{U}}_n\}, \{\ddot{\mathbb{U}}_n\}) \longrightarrow (\{\mathbb{U}_{n+1}\}, \{\dot{\mathbb{U}}_{n+1}\}, \{\ddot{\mathbb{U}}_{n+1}\}). \quad (5.30)$$

A comprehensive presentation of the many available numerical time integration schemes for dynamical analyses is beyond the scope of this text. Instead, the exposition will focus on the Newmark family of schemes, which are widely used in the present context (Sections 5.4.3 to 5.4.7). The Newmark schemes involves two tunable parameters, which lend some flexibility regarding the fulfillment of properties that are either necessary (stability) ou desirable (accuracy, numerical damping, ...). Before presenting the Newmark schemes, an explicit scheme whose derivation is quite intuitive, namely the central difference method, is introduced in Section 5.4.2. The latter will be found later (in Section 5.4.7) to actually belong to the Newmark family.

### 5.4.2 An explicit time-stepping scheme: the central difference method

An intuitive approach consists in defining a time-stepping scheme by means of a central finite difference approximation of the velocity  $\{\dot{\mathbb{U}}\}$ :

$$\{\dot{\mathbb{U}}_{n+1/2}\} := \{\dot{\mathbb{U}}(t_{n+1/2})\} \approx \frac{1}{\Delta t} [\{\mathbb{U}_{n+1}\} - \{\mathbb{U}_n\}] \quad (5.31)$$

and of the acceleration  $\{\ddot{\mathbb{U}}\}$ :

$$\{\ddot{\mathbb{U}}_n\} \approx \frac{1}{\Delta t} [\{\dot{\mathbb{U}}_{n+1/2}\} - \{\dot{\mathbb{U}}_{n-1/2}\}] \quad (5.32)$$

In the above approximation formulas, the intermediary time instant  $t_{n+1/2}$  is defined as

$$t_{n+1/2} := (t_{n+1} + t_n)/2 = t_n + \Delta t/2.$$



The differential system (5.9) is then written at time  $t = t_{n+1}$ :

$$[\mathbb{M}]\{\ddot{\mathbb{U}}_{n+1}\} = \{\mathbb{F}_{n+1}\} - [\mathbb{K}]\{\mathbb{U}_{n+1}\}. \quad (5.33)$$

The central finite difference scheme then proceeds as follows for each time increment:

- (i) Update displacements through  $\{\mathbb{U}_{n+1}\} = \{\mathbb{U}_n\} + \Delta t\{\dot{\mathbb{U}}_{n+1/2}\}$ ;
- (ii) Solve system (5.33) for  $\{\ddot{\mathbb{U}}_{n+1}\}$ ;
- (iii) Update velocities through  $\{\dot{\mathbb{U}}_{n+3/2}\} = \{\dot{\mathbb{U}}_{n+1/2}\} + \Delta t\{\ddot{\mathbb{U}}_{n+1}\}$ .

It is summarized in Box 5.1.

**Box 5.1:** Time-stepping algorithm: the central difference method

Data: mesh, material parameters  $(E, \nu, \rho)$ , initial conditions  $\{\mathbb{U}_0\}$ ,  $\{\dot{\mathbb{U}}_0\}$ , excitations  $\mathbf{u}^D(\mathbf{x}, t)$ ,  $\mathbf{T}^D(\mathbf{x}, t)$ ,  $\mathbf{f}(\mathbf{x}, t)$ .

- 1. Assembly of  $[\mathbb{K}]$  et  $[\mathbb{M}]$ ;
- 2. Initial acceleration: find  $\{\ddot{\mathbb{U}}_0\}$  by solving

$$[\mathbb{M}]\{\ddot{\mathbb{U}}_0\} = \{\mathbb{F}_0\} - [\mathbb{K}]\{\mathbb{U}_0\}$$

- 3. Initial velocity  $\{\dot{\mathbb{U}}_{1/2}\}$ :

$$\{\dot{\mathbb{U}}_{1/2}\} = \{\dot{\mathbb{U}}_0\} + \frac{1}{2}\Delta t\{\ddot{\mathbb{U}}_0\}$$

- 4. For  $n = 0, 1, 2, \dots, M - 1$  (time-stepping loop):

- (a) Displacement update:

$$\{\mathbb{U}_{n+1}\} = \{\mathbb{U}_n\} + \Delta t\{\dot{\mathbb{U}}_{n+1/2}\}$$

- (b) Current acceleration:

$$[\mathbb{M}]\{\ddot{\mathbb{U}}_{n+1}\} = \{\mathbb{F}_{n+1}\} - [\mathbb{K}]\{\mathbb{U}_{n+1}\}$$

- (c) Velocity update:

$$\{\dot{\mathbb{U}}_{n+3/2}\} = \{\dot{\mathbb{U}}_{n+1/2}\} + \Delta t\{\ddot{\mathbb{U}}_{n+1}\}$$

One notes that combining approximations (5.31) of  $\{\dot{\mathbb{U}}\}$  and (5.32) of  $\{\ddot{\mathbb{U}}\}$  leads to the well-known approximation formula for  $\{\ddot{\mathbb{U}}\}$  using a second-order central finite difference:

$$\{\ddot{\mathbb{U}}_n\} \approx \frac{1}{\Delta t^2} [\{\mathbb{U}_{n+1}\} - 2\{\mathbb{U}_n\} + \{\mathbb{U}_{n-1}\}].$$

**Explicit nature of the central difference method.** The central difference time-stepping algorithm is said to be explicit because it relies on solving the system (5.33), governed by the mass matrix. Rigorously speaking, one needs to resort to a linear solver (of either direct or iterative type), and the solution  $\{\ddot{\mathbb{U}}_n\}$

of (5.33) is not given by an explicit formula. However, the mass matrix  $[\mathbb{M}]$  can often be legitimately be replaced with a *diagonal* approximation  $[\tilde{\mathbb{M}}]$ , known as a *lumped mass matrix*. Such diagonal approximation  $[\tilde{\mathbb{M}}]$  is easily set up in practice, using any of several approaches, see e.g. Hughes (1987). For example, one may sum all entries in a row of  $[\mathbb{M}]$  and affect the result to the diagonal entry of that row. Replacing  $[\mathbb{M}]$  with a diagonal approximation  $[\tilde{\mathbb{M}}]$  in system (5.33) yields an explicit expression for  $\{\ddot{\mathbf{U}}_n\}$  whose evaluation is computationally very inexpensive.

The central difference method will be shown in Section 5.4.7 to be conditionally stable, the time step  $\Delta t$  needing to be set to a value lower than a critical time step  $(\Delta t)_{\text{stab}}$ . This conditionally-stable character is sometimes detrimental, especially since accurately evaluating the critical time step  $(\Delta t)_{\text{stab}}$  is neither easy nor economical (see Sec. 5.4.10). This explains the usefulness of other time-stepping schemes that are implicit but unconditionally stable. The Newmark family (among others) fulfills this need, as some of its members are unconditionally stable. It is thus going to be presented and analysed next. Among other facts, the central difference scheme will be seen to coincide with one of the Newmark time-stepping schemes.

### 5.4.3 The Newmark family of time-stepping schemes

The Newmark algorithm is widely used for computational structural dynamics. It is in fact a two-parameter family of algorithms, based on the following displacement and velocity update formulas:

$$\begin{aligned}\{\mathbf{U}_{n+1}\} &\approx \{\mathbf{U}_n\} + \Delta t \{\dot{\mathbf{U}}_n\} + \Delta t^2 \left[ \left( \frac{1}{2} - \beta \right) \{\ddot{\mathbf{U}}_n\} + \beta \{\ddot{\mathbf{U}}_{n+1}\} \right], \\ \{\dot{\mathbf{U}}_{n+1}\} &\approx \{\dot{\mathbf{U}}_n\} + \Delta t \left[ (1 - \gamma) \{\ddot{\mathbf{U}}_n\} + \gamma \{\ddot{\mathbf{U}}_{n+1}\} \right].\end{aligned}\quad (5.34)$$

where the algorithmic parameters  $\beta$  and  $\gamma$  are constrained by  $0 \leq \beta \leq 1/2$  and  $0 \leq \gamma \leq 1$ . The update formulas may be understood as modified forms of Taylor expansions about  $t = t_n$ , which read

$$\begin{aligned}\{\mathbf{U}_{n+1}\} &= \{\mathbf{U}_n\} + \Delta t \{\dot{\mathbf{U}}_n\} + \frac{1}{2} \Delta t^2 \{\ddot{\mathbf{U}}_n\} + o(\Delta t^2), \\ \{\dot{\mathbf{U}}_{n+1}\} &= \{\dot{\mathbf{U}}_n\} + \Delta t \{\ddot{\mathbf{U}}_n\} + o(\Delta t),\end{aligned}\quad (5.35)$$

with the initial acceleration  $\{\ddot{\mathbf{U}}_n\}$  replaced in each formula by a weighted average of the initial and final accelerations  $\{\ddot{\mathbf{U}}_n\}$ ,  $\{\ddot{\mathbf{U}}_{n+1}\}$ . Values of  $\beta, \gamma$  for which the resulting time-stepping scheme has desirable properties will be investigated later on.

The postulated update relations (5.34) are then introduced into the differential system (5.9) resulting from spatial finite element semi-discretization, written at time  $t = t_{n+1}$ , to obtain

$$\begin{aligned}([\mathbb{M}] + \beta \Delta t^2 [\mathbb{K}]) \{\ddot{\mathbf{U}}_{n+1}\} \\ = \{\mathbb{F}_{n+1}\} - [\mathbb{K}] \left( \{\mathbf{U}_n\} + \Delta t \{\dot{\mathbf{U}}_n\} + \Delta t^2 \left( \frac{1}{2} - \beta \right) \{\ddot{\mathbf{U}}_n\} \right)\end{aligned}\quad (5.36)$$

Assuming all mechanical quantities to be known at time  $t_n$ , the time-stepping scheme then consists in

- (i) Solving system (5.36) for  $\{\ddot{\mathbf{U}}_{n+1}\}$ ;
- (ii) Updating  $\{\mathbf{U}_{n+1}\}$  and  $\{\dot{\mathbf{U}}_{n+1}\}$  using equations (5.34).

It is summarized in Box 5.2.

**Convergence.** By virtue of the Lax convergence theorem (Lax and Richtmyer, 1956), also invoked previously in Chapter 4 for the heat conduction problem, the Newmark algorithm converges (to the solution  $\{\mathbf{U}(t)\}$  of the time-continuous and spatially semi-discretized problem) if and only if it is *stable* and *consistent*. It is therefore essential to determine whether, and if yes how, these conditions constrain the possible choices for parameters  $\beta, \gamma$ .

#### 5.4.4 Stability analysis

As explained in Chapter 4, a time-stepping scheme is unstable whenever error amplification may occur between two consecutive iterates. To investigate whether

##### Box 5.2: The Newmark time-stepping algorithm

Data: mesh, material parameters  $(E, \nu, \rho)$ , initial conditions  $\{\mathbf{U}_0\}$ ,  $\{\dot{\mathbf{U}}_0\}$ , excitations  $\mathbf{u}^D(\mathbf{x}, t)$ ,  $\mathbf{T}^D(\mathbf{x}, t)$ ,  $\mathbf{f}(\mathbf{x}, t)$ .

1. Set up matrices  $[\mathbb{K}]$  and  $[\mathbb{M}]$ ;
2. Initialization: compute  $\{\ddot{\mathbf{U}}_0\}$  by solving

$$[\mathbb{M}]\{\ddot{\mathbf{U}}_0\} = \{\mathbf{F}_0\} - [\mathbb{K}]\{\mathbf{U}_0\}$$

3. Compute and factor the matrix  $[\mathbb{S}] = [\mathbb{M}] + \beta\Delta t^2[\mathbb{K}]$ ;
4. For  $n = 0, 1, 2, \dots, M - 1$  (time stepping loop):

(a) Prediction:

$$\begin{aligned}\{\mathbf{U}_{n+1}^{\text{pred}}\} &= \{\mathbf{U}_n\} + \Delta t\{\dot{\mathbf{U}}_n\} + \Delta t^2\left(\frac{1}{2} - \beta\right)\{\ddot{\mathbf{U}}_n\} \\ \{\dot{\mathbf{U}}_{n+1}^{\text{pred}}\} &= \{\dot{\mathbf{U}}_n\} + \Delta t(1 - \gamma)\{\ddot{\mathbf{U}}_n\}\end{aligned}$$

(b) Compute acceleration by solving

$$[\mathbb{S}]\{\ddot{\mathbf{U}}_{n+1}\} = \{\mathbf{F}_{n+1}\} - [\mathbb{K}]\{\mathbf{U}_{n+1}^{\text{pred}}\}$$

(c) Correction and update:

$$\begin{aligned}\{\mathbf{U}_{n+1}\} &= \{\mathbf{U}_{n+1}^{\text{pred}}\} + \Delta t^2\beta\{\ddot{\mathbf{U}}_{n+1}\} \\ \{\dot{\mathbf{U}}_{n+1}\} &= \{\dot{\mathbf{U}}_{n+1}^{\text{pred}}\} + \Delta t\gamma\{\ddot{\mathbf{U}}_{n+1}\}\end{aligned}$$

such amplification is possible for a Newmark algorithm, it is convenient to reformulate it as

$$(\{\mathbb{U}_n\}, \{\dot{\mathbb{U}}_n\}) \longrightarrow (\{\mathbb{U}_{n+1}\}, \{\dot{\mathbb{U}}_{n+1}\}), \quad (5.37)$$

so as to represent it as a sequence of transitions effected from the initial data  $\{\mathbb{U}_0\}, \{\dot{\mathbb{U}}_0\}$ . To this aim, relations (5.34) are left-multiplied by the mass matrix, to obtain

$$\begin{aligned} [\mathbb{M}] (\{\mathbb{U}_{n+1}\} - \beta \Delta t^2 \{\ddot{\mathbb{U}}_{n+1}\}) &= [\mathbb{M}] (\{\mathbb{U}_n\} + \Delta t \{\dot{\mathbb{U}}_n\} + \Delta t^2 (\tfrac{1}{2} - \beta) \{\ddot{\mathbb{U}}_n\}), \\ [\mathbb{M}] (\{\dot{\mathbb{U}}_{n+1}\} - \gamma \Delta t \{\ddot{\mathbb{U}}_{n+1}\}) &= [\mathbb{M}] (\{\dot{\mathbb{U}}_n\} + \Delta t (1 - \gamma) \{\ddot{\mathbb{U}}_n\}). \end{aligned}$$

Then, the differential system (5.9) taken at time  $t = t_{n+1}$  is used for eliminating  $[\mathbb{M}] \{\ddot{\mathbb{U}}_n\}$  and  $[\mathbb{M}] \{\ddot{\mathbb{U}}_{n+1}\}$  from each equation, leading to the recursion

$$\begin{aligned} \begin{bmatrix} [\mathbb{M}] + \beta \Delta t^2 [\mathbb{K}] & 0 \\ \gamma \Delta t [\mathbb{K}] & [\mathbb{M}] \end{bmatrix} \begin{Bmatrix} \mathbb{U}_{n+1} \\ \dot{\mathbb{U}}_{n+1} \end{Bmatrix} - \begin{bmatrix} (\beta - \tfrac{1}{2}) \Delta t^2 [\mathbb{K}] - [\mathbb{M}] & \Delta t [\mathbb{M}] \\ (\gamma - 1) \Delta t [\mathbb{K}] & [\mathbb{M}] \end{bmatrix} \begin{Bmatrix} \mathbb{U}_n \\ \dot{\mathbb{U}}_n \end{Bmatrix} \\ - \begin{Bmatrix} (\tfrac{1}{2} - \beta) \Delta t^2 \mathbb{F}_n + \beta \Delta t^2 \mathbb{F}_{n+1} \\ (1 - \gamma) \Delta t \mathbb{F}_n + \gamma \Delta t \mathbb{F}_{n+1} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}, \quad (5.38) \end{aligned}$$

which can be recast in the more compact form

$$\begin{Bmatrix} \mathbb{U}_{n+1} \\ \dot{\mathbb{U}}_{n+1} \end{Bmatrix} = [\mathbb{Q}] \begin{Bmatrix} \mathbb{U}_n \\ \dot{\mathbb{U}}_n \end{Bmatrix} + \{\mathbb{Y}_n\}.$$

Like in Chapter 4, the stability analysis is made easier by using a diagonalization of  $[\mathbb{Q}]$ . To this end, let  $\omega_I^2$  and  $\{\mathbb{X}^I\}$  ( $1 \leq I \leq N$ ) respectively denote the eigenvalues and eigenvectors associated with the generalized eigenvalue problem (5.14). Then, let the displacement and velocity DOF vectors be expanded on the  $\{\mathbb{X}^I\}$  basis according to

$$\begin{aligned} \{\mathbb{U}_n\} &= \sum_{J=1}^N u_n^J \{\mathbb{X}^J\}, & \{\dot{\mathbb{U}}_n\} &= \sum_{J=1}^N v_n^J \{\mathbb{X}^J\}, \\ \{\mathbb{U}_{n+1}\} &= \sum_{J=1}^N u_{n+1}^J \{\mathbb{X}^J\}, & \{\dot{\mathbb{U}}_{n+1}\} &= \sum_{J=1}^N v_{n+1}^J \{\mathbb{X}^J\}. \end{aligned}$$

Substituting the above expansions into (5.38), left-multiplying the resulting equality by  $\{(\mathbb{X}^I)^T (\mathbb{X}^I)^T\}$  for some fixed  $I$  and exploiting the  $[\mathbb{M}]$ -orthonormality and  $[\mathbb{K}]$ -orthogonality properties of the eigenvectors, (5.38) yields  $N$  uncoupled  $2 \times 2$  linear systems:

$$\begin{aligned} \begin{bmatrix} 1 + \beta \Delta t^2 \omega_J^2 & 0 \\ \gamma \Delta t \omega_J^2 & 1 \end{bmatrix} \begin{Bmatrix} u_{n+1}^J \\ v_{n+1}^J \end{Bmatrix} &= \begin{bmatrix} (\beta - \tfrac{1}{2}) \Delta t^2 \omega_J^2 - 1 & \Delta t \\ (\gamma - 1) \Delta t \omega_J^2 & 1 \end{bmatrix} \begin{Bmatrix} u_n^J \\ v_n^J \end{Bmatrix} \\ &+ \begin{Bmatrix} (\tfrac{1}{2} - \beta) \Delta t^2 f_n^J + \beta \Delta t^2 f_{n+1}^J \\ (1 - \gamma) \Delta t f_n^J + \gamma \Delta t f_{n+1}^J \end{Bmatrix}, \quad (5.39) \end{aligned}$$

which thus have the form

$$\begin{Bmatrix} u_{n+1}^J \\ v_{n+1}^J \end{Bmatrix} = [Q_J] \begin{Bmatrix} u_n^J \\ v_n^J \end{Bmatrix} + \{Y_{n+1}^J\} \quad (1 \leq J \leq N). \quad (5.40)$$

The stability condition therefore consists in ensuring that, for any  $J$ , the matrix  $[Q_J]$  does not result in amplification of perturbations, e.g. on initial conditions:

$$\left\| \begin{Bmatrix} \delta u_{n+1}^J \\ \delta v_{n+1}^J \end{Bmatrix} \right\| = \left\| [Q_J]^n \begin{Bmatrix} \delta u_0^J \\ \delta v_0^J \end{Bmatrix} \right\| \leq \left\| \begin{Bmatrix} \delta u_0^J \\ \delta v_0^J \end{Bmatrix} \right\| \quad (\forall J, 1 \leq J \leq N).$$

Let  $\lambda_1^J, \lambda_2^J$  be the eigenvalues of  $[Q_J]$ . According to the theory of Jordan canonical forms, for any unsymmetric real-valued matrix  $[Q_J]$  there exist an invertible matrix  $[\mathbb{V}]$  such that, if  $\lambda_1^J \neq \lambda_2^J$ :

$$[Q_J] = [\mathbb{V}]^{-1} \begin{bmatrix} \lambda_1^J & 0 \\ 0 & \lambda_2^J \end{bmatrix} [\mathbb{V}] \rightarrow [Q_J]^n = [\mathbb{V}]^{-1} \begin{bmatrix} (\lambda_1^J)^n & 0 \\ 0 & (\lambda_2^J)^n \end{bmatrix} [\mathbb{V}]$$

while, if  $\lambda_1^J = \lambda_2^J = \lambda^J$ :

$$[Q_J] = [\mathbb{V}]^{-1} \begin{bmatrix} \lambda^J & 1 \\ 0 & \lambda^J \end{bmatrix} [\mathbb{V}] \rightarrow [Q_J]^n = [\mathbb{V}]^{-1} \begin{bmatrix} (\lambda^J)^n & n(\lambda^J)^{n-1} \\ 0 & (\lambda^J)^n \end{bmatrix} [\mathbb{V}]$$

The stability condition can be hence expressed as follows:

$$\begin{cases} \text{If } \lambda_1^J \neq \lambda_2^J: & \text{one must have } |\lambda_1^J| \leq 1, |\lambda_2^J| \leq 1 \\ \text{If } \lambda_1^J = \lambda_2^J: & \text{one must have } |\lambda_1^J| = |\lambda_2^J| < 1. \end{cases} \quad (\forall J, 1 \leq J \leq N). \quad (5.41)$$

The matrix  $[Q_J]$  being defined by equations (5.39) and (5.40), its eigenvalues solve the generalized eigenvalue problem

$$\left( \begin{bmatrix} (\beta - \frac{1}{2})\Delta t^2 \omega_J^2 - 1 & \Delta t \\ (\gamma - 1)\Delta t \omega_J^2 & 1 \end{bmatrix} - \lambda \begin{bmatrix} 1 + \beta \Delta t^2 \omega_J^2 & 0 \\ \gamma \Delta t \omega_J^2 & 1 \end{bmatrix} \right) \begin{Bmatrix} u \\ v \end{Bmatrix} = 0.$$

This avoids an explicit calculation of the inverse of the matrix involved in the left-hand side of equation (5.39). After some straightforward manipulation, the associated characteristic equation is

$$\text{Det}([Q_J] - \lambda[I]) = 0 \Leftrightarrow \lambda^2 - 2A\lambda + B = 0 \quad (5.42)$$

$$\text{with } 2A = 2 - \left(\frac{1}{2} + \gamma\right)\zeta^2, \quad B = 1 + \left(\frac{1}{2} - \gamma\right)\zeta^2, \quad \zeta^2 = \frac{\omega_J^2 \Delta t^2}{1 + \beta \omega_J^2 \Delta t^2}.$$

The stability conditions (5.41), expressed in terms of  $(A, B)$ , then are as follows:

(i) If  $A^2 - B > 0$  (two distinct real eigenvalues):

$$-1 \leq A \leq 1 \quad \text{and} \quad \begin{cases} 1 - 2A + B \geq 0, \\ 1 + 2A + B \geq 0; \end{cases}$$

(ii) If  $A^2 - B = 0$  (one eigenvalue with multiplicity 2):

$$-1 < A < 1;$$

(iii) If  $A^2 - B < 0$  (two mutually conjugate distinct complex eigenvalues):

$$B \leq 1.$$

Conditions (i), (ii), (iii) are geometrically depicted in Figure 5.5. This shows that the stability conditions of the Newmark algorithm, expressed using  $A, B$ , are

$$B \leq 1, \quad 1 - 2A + B \geq 0, \quad 1 + 2A + B \geq 0 \quad A \neq \pm 1.$$

They correspond geometrically to the closed triangle with points P, Q removed (Fig. 5.5). Expressing these conditions in terms of parameters  $\beta, \gamma$  and the time step  $\Delta t$ , the above conditions become

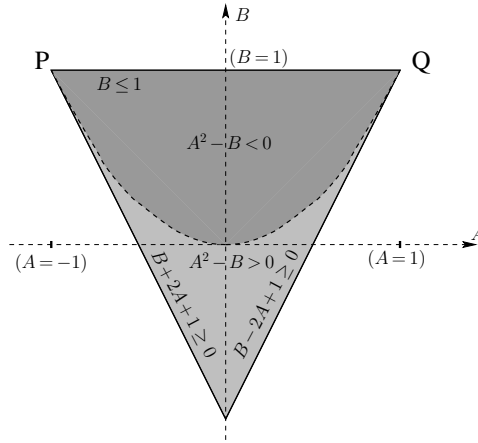
$$(a) \zeta^2 \geq 0, \quad (b) 2\gamma - 1 \geq 0, \quad (c) 4 - 2\gamma\zeta^2 \geq 0. \quad (5.43)$$

Condition (5.43a) is automatically satisfied for all values of  $\beta, \gamma$  because of definition (5.42). Condition (5.43b) enforces  $\gamma \geq 1/2$ . Finally, condition (5.43c) can, in view of definition (5.42) of  $\zeta^2$ , be recast in the form

$$4 + 2(2\beta - \gamma)\omega_f^2 \Delta t^2 \geq 0.$$

Consequently, two possibilities arise:

(i) If  $2\beta - \gamma \geq 0$ , condition (5.43b) is satisfied for any choice of  $\Delta t$ ;



**Figure 5.5:** Stability domain of the Newmark scheme, in terms of the coefficients  $A, B$  of the characteristic equation (5.42).

(ii) If  $2\beta - \gamma < 0$ , condition (5.43b) implies a restriction on  $\Delta t$ :

$$\omega_J^2 \Delta t^2 \leq 2/(\gamma - 2\beta) \quad \forall J.$$

The time step  $\Delta t$  must therefore be chosen smaller than a critical value  $\Delta t_{\text{stab}}$ :

$$\Delta t < (\Delta t)_{\text{stab}} := \min_J \frac{1}{\omega_J} \frac{2}{\sqrt{2\gamma - 4\beta}}. \quad (5.44)$$

### 5.4.5 Consistency analysis

According to the definition introduced in Chapter 4, the time-stepping algorithm (here, the Newmark scheme) is *consistent* with the time-continuous differential system (5.9) if the residuals of the equations defining the time-stepping scheme vanish in the limit  $\Delta t \rightarrow 0$  when  $\{\mathbb{U}_{n+1}\}$  and  $\{\mathbb{U}_n\}$  are the values of a sufficiently smooth vector function  $t \mapsto \{\mathbb{U}(t)\}$  that solves the differential system (5.9).

Here, the action of the Newmark scheme may be formulated as the matrix equation (5.38) expressing the time-discrete transition (5.37). It is therefore of interest to evaluate the residual of the system (5.38) if  $\{\mathbb{U}_{n+1}\}$  and  $\{\mathbb{U}_n\}$  are time-sampled values of a vector function  $t \mapsto \{\mathbb{U}(t)\}$  solving the time-continuous differential system (5.9). On substituting Taylor expansions about  $t = t_n$  for  $\{\mathbb{U}_{n+1}\}$ ,  $\{\dot{\mathbb{U}}_{n+1}\}$  and  $\{\mathbb{F}_{n+1}\}$  in system (5.38) and using equations

$$[\mathbb{K}]\{\mathbb{U}_n\} + [\mathbb{M}]\{\ddot{\mathbb{U}}_n\} = \{\mathbb{F}_n\}, \quad [\mathbb{K}]\{\dot{\mathbb{U}}_n\} + [\mathbb{M}]\{\ddot{\mathbb{U}}_n\} = \{\dot{\mathbb{F}}_n\}$$

expressing satisfaction of (5.9) at time  $t = t_n$  whenever possible, one arrives after some manipulation at:

$$\text{Left-hand side of (5.38)} = \begin{bmatrix} \mathbb{M} & 0 \\ 0 & \mathbb{M} \end{bmatrix} \left\{ \Delta t^3 \left( \frac{1}{6} - \beta \right) \ddot{\mathbb{U}}_n \right\} + \left\{ o(\Delta t^3) \right\} + \left\{ \Delta t^2 \left( \frac{1}{2} - \gamma \right) \ddot{\mathbb{U}}_n \right\} + \left\{ o(\Delta t^2) \right\}. \quad (5.45)$$

This shows in particular that, for any values of the Newmark parameters  $\beta, \gamma$ , the residual of system (5.38) vanishes as  $\Delta t \rightarrow 0$ . The Newmark algorithm is therefore consistent for any choice of its parameters.

#### Box 5.3: Stability of the Newmark scheme.

1. If  $\gamma \geq 1/2$  and  $2\beta - \gamma \geq 0$ , the Newmark scheme is *unconditionally stable*;
2. If  $\gamma \geq 1/2$  and  $2\beta - \gamma < 0$ , the Newmark scheme is *conditionally stable*, the time step being constrained by

$$\Delta t < (\Delta t)_{\text{stab}} := \min_J \frac{1}{\omega_J} \frac{2}{\sqrt{2\gamma - 4\beta}}$$

3. If  $\gamma < 1/2$ , the Newmark scheme is *unstable*.

### 5.4.6 Accuracy

The result (5.45) suggests that the accuracy of the Newmark algorithm may be optimized by setting its parameters to

$$\beta = 1/6, \quad \gamma = 1/2. \quad (5.46)$$

However, this choice defines a scheme that is only conditionally stable, since the condition  $2\beta - \gamma \geq 0$  for unconditional stability is not met. For that reason, one often prefers setting its parameters to

$$\beta = 1/4, \quad \gamma = 1/2, \quad (5.47)$$

which yield the best accuracy consistent with unconditional stability.

### 5.4.7 Revisiting the central difference explicit scheme

Consider the Newmark scheme with  $\beta = 0, \gamma = 1/2$ . On writing relation (5.34a) at times  $t_{n+1}$  and  $t_n$  and subtracting the resulting identities, one arrives at

$$\{\mathbb{U}_{n+1}\} - 2\{\mathbb{U}_n\} + \{\mathbb{U}_{n-1}\} = \Delta t(\{\dot{\mathbb{U}}_n\} - \{\dot{\mathbb{U}}_{n-1}\}) + \frac{\Delta t^2}{2}(\{\ddot{\mathbb{U}}_n\} - \{\ddot{\mathbb{U}}_{n-1}\}).$$

Moreover, relation (5.34b) at time  $t_n$  yields

$$\{\dot{\mathbb{U}}_n\} - \{\dot{\mathbb{U}}_{n-1}\} = \frac{\Delta t}{2}(\{\ddot{\mathbb{U}}_n\} + \{\ddot{\mathbb{U}}_{n-1}\}).$$

Combining the above two identities, one obtains

$$\{\mathbb{U}_{n+1}\} - 2\{\mathbb{U}_n\} + \{\mathbb{U}_{n-1}\} = \Delta t^2\{\ddot{\mathbb{U}}_n\}, \quad (5.48)$$

which is none other than the second-order central difference approximation (5.32) of the acceleration.

The Newmark scheme with  $\beta = 0, \gamma = 1/2$  therefore coincides with the central difference explicit scheme presented in Section 5.4.2. In particular, this remark and (5.44) imply that this scheme is *conditionally stable*, with

$$\Delta t < (\Delta t)_{\text{stab}} = \min_J \frac{2}{\omega_J}. \quad (5.49)$$

### 5.4.8 Example: elastic rod

**Time-stepping algorithm.** We focus on the elastic rod already discussed in Section 5.3.2 and implement in `bar_B2_1D_wave_direct.m` the Newmark time-stepping algorithm as described in Section 5.4.3. The three arrays `U`, `Ud`, `Udd` respectively hold the nodal displacements, velocities and accelerations at all unconstrained nodes. Initial displacements and velocities are set to zero, while initial accelerations are determined as in step 2 of Box 5.2.

```

U=zeros(neq,1);           % initial value of displ.
Ud=zeros(neq,1);          % initial value of vel.
Udd=M\F;                   % initial value of acc.

```



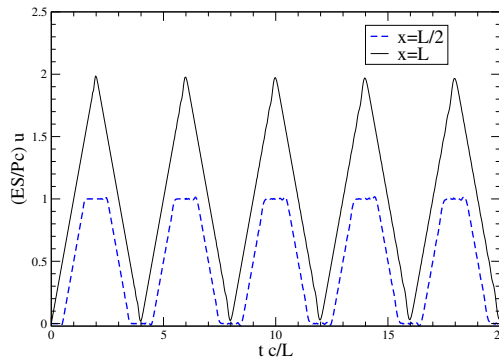
The matrix  $[S]$  is set up, and its  $LDL^T$  decomposition computed, allowing the subsequent time-stepping phase to perform faster.

```
S=M+beta*Dt^2*K;
CS=chol(S); % LDLT decomp. of S
```

Finally, the preparation work being completed, the time-stepping procedure is launched. Each step consists in (i) evaluating a displacement and velocity prediction, (ii) computing the new acceleration, and (iii) correcting and updating the displacement and velocity (steps 4(a), 4(b) and 4(c) of the box in page 160).

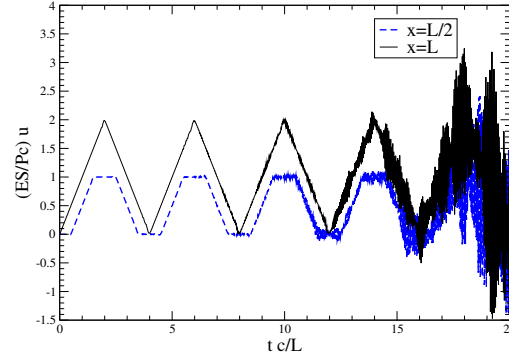
```
nstep=floor(tf/Dt); % number of time steps
for step=1:nstep, % loop over time steps
    U=U+Dt*Ud+0.5*Dt^2*(1-2*beta)*Udd; % prediction phase for displ
    Ud=Ud+Dt*(1-gamma)*Udd; % prediction phase for vel
    Fg=F-K*U;
    Udd=CS\((CS'\Fg)); % new accelerations
    U=U+Dt^2*beta*Udd; % actualisation of displ
    Ud=Ud+Dt*gamma*Udd; % actualisation of vel
end
```

The code ends with a post-processing of the displacement at  $x = L/2$  and  $x = L$ . The response of the rod has been computed for several choices of Newmark schemes and time steps, using the previously-described MATLAB code. For all cases, the spatial semi-discretization uses  $N_E = 50$  straight two-noded elements of uniform length (linear interpolation of  $u$  on each element). Figure 5.6 shows the response computed using the unconditionally stable Newmark scheme ( $\beta = 1/4$ ,  $\gamma = 1/2$ ) with  $c\Delta t/L = 0, 1$ , which is seen to be satisfactory. In contrast, the choice  $\gamma = 99/200$ , which slightly violates the stability condition  $\gamma \geq 1/2$ , produces an unstable solution after just a few back-and-forth wave propagations in the rod (Figure 5.7). Finally, Figure 5.8 shows the application of the (explicit) central difference version of the Newmark scheme ( $\beta = 1/4$ ,  $\gamma = 1/2$ ), which is conditionally stable with condition (5.49). The latter results here in  $c(\Delta t)_{\text{stab}}/L \approx 0,01155$  (the numerical value being found by solving numerically the generalized eigenvalue problem (5.14). Four values of  $\Delta t$  very close

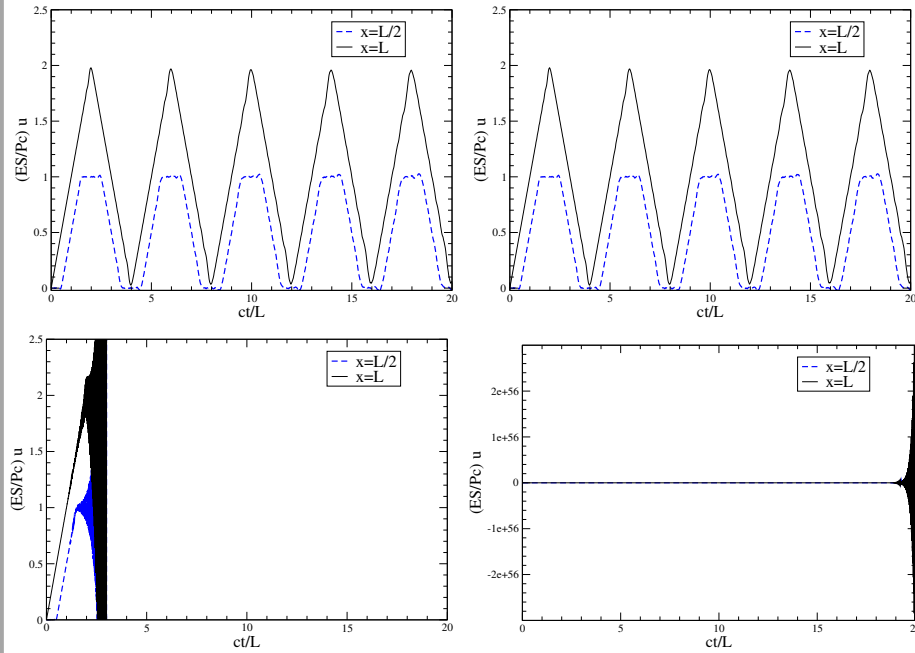


**Figure 5.6:** Rod with a step loading applied to its right end: response computed using the unconditionally stable Newmark scheme ( $\beta = 1/4$ ,  $\gamma = 1/2$ ) and  $c\Delta t/L = 0, 1$  (the curves show  $t \mapsto u(x, t)$  for  $x = L/2$  and  $x = L$ ).

to the critical time step  $\Delta t_{\text{stab}}$  have been used, and major instability is seen to occur even for very slight violations of the stability condition.



**Figure 5.7:** Rod with a step loading applied to its right end: response computed using the unstable Newmark scheme ( $\beta = 1/4$ ,  $\gamma = 99/200$ ) and  $c\Delta t/L = 0, 1$  (the curves show  $t \mapsto u(x, t)$  for  $x = L/2$  and  $x = L$ ).

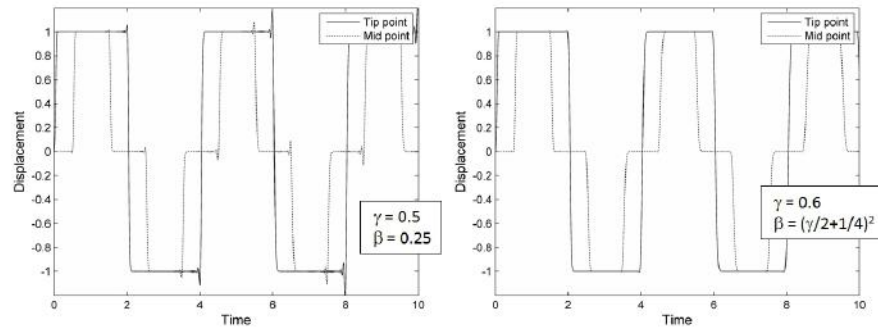


**Figure 5.8:** Rod with a step loading applied to its right end: response computed using the (explicit, conditionally stable) central difference scheme ( $\beta = 0$ ,  $\gamma = 1/2$ ). Each plot shows  $t \mapsto u(x, t)$  for  $x = L/2$  and  $x = L$ , computed using  $\Delta t = 0, 99083(\Delta t)_{\text{stab}}$  (top left),  $\Delta t = 0, 99966(\Delta t)_{\text{stab}}$  (top right),  $\Delta t = 1, 00237(\Delta t)_{\text{stab}}$  (bottom left),  $\Delta t = 1, 00815(\Delta t)_{\text{stab}}$  (bottom right).

**Blast loading.** The analysis is then repeated for the blast loading  $T^D\delta(t)$  already analysed in Section 5.3.2. As in that case we assume  $\omega_{\max} = 2\pi/\Delta$  and fix  $h = c\Delta/N_L$ . If we require that every cycle of frequency  $\omega_{\max}$  should be integrated with at least  $N_T$  time steps, this poses the following upper bound on  $\Delta t$ :

$$\Delta t \leq \frac{2\pi}{N_T\omega_{\max}} \quad (5.50)$$

The simulations of Figure 5.9 have been run with  $N_T = 10$  and  $\Delta = 0.1$ . The striking difference between the two rather close choices of  $\beta$  and  $\gamma$  is analysed in the next section.



**Figure 5.9:** Blast loading: Displacements at the middle and at the tip of the rod computed using the Newmark scheme and two different choices of parameters

#### 5.4.9 Dissipation of high frequency modes in direct integration

Even though in many situations the structural response is actually governed by low or average frequency modes, in direct integration approaches high frequency modes unavoidably play a major role. This is evident if conditionally stable approaches are employed, since the maximum allowable  $\Delta t$  is fixed by the highest eigenvalue. Moreover, even if unconditionally stable techniques are adopted, high frequency modes may induce unwanted oscillations, as observed in the examples of Section 5.4.8 and as analysed in the sequel. Let us consider the Newmark scheme with different choices of the parameters  $\beta$  and  $\gamma$  preserving unconditional stability. We fix  $\Delta t$  and consider the limit case when  $\omega_I \rightarrow \infty$ , which corresponds ideally (in the limit) to the highest frequency modes. Let  $\lambda_\infty$  denote the largest absolute value of the eigenvalues of the transition matrix  $[Q_I]$  in this case. The fastest decay of high order modes occurs when  $\lambda_\infty$  is minimised. According to (5.42), the two eigenvalues solve the characteristic equation with, if  $\omega_I \rightarrow \infty$ :

$$A = 1 - \frac{1}{2\beta} \left( \gamma + \frac{1}{2} \right), \quad B = 1 - \frac{1}{\beta} \left( \gamma - \frac{1}{2} \right)$$

In particular, if  $2\beta = \gamma$ :

$$A = -\frac{1}{2\gamma}, \quad B = \frac{1-\gamma}{\gamma}$$

and  $\lambda_1 = -1$ ,  $\lambda_2 = 1 - 1/\gamma$ . Since  $\gamma \geq 1/2$  in any stable algorithm, one always has  $\lambda_\infty = 1$ . This implies that perturbations do not decay with time and therefore pollute the analysis. As a consequence the choice  $2\beta = \gamma$  is not recommended if the dissipation of higher order modes is a priority in the analysis.

Let us consider the specific choice  $\beta = \theta^2$  and  $\gamma = 2\theta - 1/2$  ( $\theta > 1/2$ ). One has  $\lambda_1 = \lambda_2$  and, in the limit  $\omega_1 \rightarrow \infty$ :

$$\lambda_{1,2} = 1 - \frac{\theta}{\beta} \pm \frac{1}{\beta} \sqrt{\theta^2 - \beta}$$

Hence  $\lambda_\infty = \lambda_1 = \lambda_2 = 1 - 1/\theta$  with  $0 \leq 1/\theta \leq 2$  since  $\gamma \geq 1/2$ . It can be shown that this condition yields the minimum value of  $\lambda_\infty$  for a given  $\gamma$ . Clearly  $\beta = 1, \gamma = 3/2$  would guarantee  $\lambda_\infty = 0$ . However values of  $\gamma$  closer to 1 are generally preferred, since they permit better accuracy for the most important low-frequency modes.

**Exercise 5.3.** Repeat the analyses of the blast loading of the rod in Section 5.4.8 with different choices of  $\theta$  and analyse the behaviour of high frequency modes.

#### 5.4.10 Estimation of the critical time step

The value (5.44) of the critical time step  $(\Delta t)_{\text{stab}}$  depends on the largest eigenvalue  $\omega_{\text{max}}$  of the generalized eigenvalue problem (5.14). Accurate numerical determination of  $\omega_{\text{max}}$  requires solving problem (5.14), a computationally expensive task.

The eigenvalue  $\omega_{\text{max}}$  can be characterized in terms of the *Rayleigh quotient*, which will prove helpful in defining a computationally lighter procedure for estimating  $(\Delta t)_{\text{stab}}$ . The Rayleigh quotient  $\mathcal{R}(\{\mathbb{V}\})$  associated with matrices  $[\mathbb{M}]$  and  $[\mathbb{K}]$  is defined as:

$$\mathcal{R}(\{\mathbb{V}\}) = \frac{\{\mathbb{V}\}^T [\mathbb{K}] \{\mathbb{V}\}}{\{\mathbb{V}\}^T [\mathbb{M}] \{\mathbb{V}\}}$$

Since the matrices  $[\mathbb{M}]$  and  $[\mathbb{K}]$  are respectively positive definite and positive semi-definite<sup>1</sup>, one has  $\mathcal{R}(\{\mathbb{V}\}) \geq 0$  for any non-vanishing  $\{\mathbb{V}\}$ . Moreover, one has

$$\mathcal{R}(\{\mathbb{V}\}) \leq \omega_{\text{max}}^2 \quad (5.51)$$

To prove this inequality, let  $\{\mathbb{V}\}$  be expanded on the eigenvector basis  $\{\mathbb{X}_J\}$  defined by problem (5.14):

$$\{\mathbb{V}\} = \sum_{J=1}^N v_J \{\mathbb{X}_J\}$$

<sup>1</sup>  $[\mathbb{K}]$  is in fact positive definite whenever kinematical constraints preventing any rigid-body motion are present.

Using the assumed  $[\mathbb{M}]$ -orthonormality and  $[\mathbb{K}]$ -orthogonality of the  $\{\mathbb{X}_J\}$ , one obtains

$$\mathcal{R}(\{\mathbb{V}\}) = \frac{(\sum_{J=1}^N v_J \{\mathbb{X}_J\}^T) \{\mathbb{K}\} (\sum_{K=1}^N v_K \{\mathbb{X}_K\})}{(\sum_{J=1}^N v_J \{\mathbb{X}_J\}^T) \{\mathbb{M}\} (\sum_{K=1}^N v_K \{\mathbb{X}_K\})} = \frac{\sum_{K=1}^N \omega_K^2 v_K^2}{\sum_{K=1}^N v_K^2}$$

The claimed inequality (5.51) then readily follows from exploiting the inequality  $\sum_{K=1}^N \omega_K^2 v_K^2 \leq \omega_{\max}^2 \sum_{K=1}^N v_K^2$  in the numerator of  $\mathcal{R}(\{\mathbb{V}\})$  as given above.

**Dependence of  $(\Delta t)_{\text{stab}}$  on mesh size.** It is first useful to investigate the dependence of  $(\Delta t)_{\text{stab}}$  on the characteristic element diameter  $h$  (see Sec. 2.1). To this purpose, the effect of using a non-dimensional coordinate  $\bar{x} := x/h$  in the generalized eigenvalue problem (5.14) is examined. Considering a generic element  $E$  and its scaled version  $\bar{E} := E/h$ , this coordinate scaling induces the following relationships between the initial and rescaled versions of the element mass and stiffness matrices:

$$\begin{aligned} \{\mathbb{W}\}^T [M_e] \{\mathbb{U}\} &= h^3 \rho \{\mathbb{W}\}^T [\bar{M}_e] \{\mathbb{U}\}, & [\bar{M}_e] &:= \int_{\bar{E}} \mathbf{u}_h \cdot \mathbf{w}_h \, d\bar{V} \\ \{\mathbb{W}\}^T [K_e] \{\mathbb{U}\} &= h\mu \{\mathbb{W}\}^T [\bar{K}_e] \{\mathbb{U}\}, & [\bar{K}_e] &:= \int_{\bar{E}} \bar{\varepsilon}[\mathbf{u}_h] : \bar{\mathcal{A}} : \bar{\varepsilon}[\mathbf{w}_h] \, d\bar{V} \end{aligned}$$

having also set  $\bar{\mathcal{A}} = \mu \bar{\mathcal{A}}$  for some elastic modulus  $\mu$ . This implies, through assemblage, that the global matrices follows similar scaling relationships

$$[\mathbb{M}] = h^3 \rho [\bar{\mathbb{M}}], \quad [\mathbb{K}] = h\mu [\bar{\mathbb{K}}].$$

Letting  $\bar{\omega}_{\max}$  denote the largest eigenvalue of the generalized eigenvalue problem  $[\bar{\mathbb{K}}]\{\mathbb{X}\} - \bar{\omega}^2 [\bar{\mathbb{M}}]\{\mathbb{X}\} = \{0\}$ , one then finds by comparison to (5.14) that

$$\bar{\omega}_{\max} = \omega_{\max} h/c,$$

where  $c = \sqrt{\mu/\rho}$  is a characteristic wave velocity for the elastic material considered. Using this result in the value (5.44) of  $(\Delta t)_{\text{stab}}$  yields

$$c(\Delta t)_{\text{stab}}/h = C \quad \text{with} \quad C = \frac{1}{\bar{\omega}_{\max}} \frac{2}{\sqrt{2\gamma - 4\beta}}$$

Two important remarks can be made about the above non-dimensional formulation of the critical time step:

- (a) Since  $(\Delta t)_{\text{stab}}$  is proportional to the characteristic mesh size  $h$ , a spatial refinement of the mesh requires a proportional reduction of the time step  $\Delta t$  (for instance, halving the element size implies halving  $\Delta t$ , i.e. doubling the number of time steps for computing a dynamical response over a given time interval);
- (b) The constant  $C$  measures the fraction of element length travelled by a wave over the duration  $(\Delta t)_{\text{stab}}$ .

**Approximate evaluation of  $(\Delta t)_{stab}$ .** Let us now consider the generalized eigenvalue problem associated with a generic element  $e$  of the mesh, of element matrices  $[M_e]$  and  $[K_e]$  (which are respectively positive definite and positive semi-definite):

$$[K_e]\{X_e\} - \omega_e^2[M_e]\{X_e\} = \{0\} \quad (5.52)$$

This problem is of size  $n_{ne}$ , where  $n_{ne}$  is the number of element degrees of freedom; its solution is therefore inexpensive. We may hence assume that all element-level eigenvalue problems (5.52) have been solved, so that in particular we know the maximum local eigenvalue  $\omega_{\max}^E = \max_e \omega_e$ .

Our aim is to show that:

$$\omega_{\max} \leq \omega_{\max}^E \quad (5.53)$$

which means that  $\omega_{\max}^E$  can be employed as an upper bound for the original eigenvalue problem (5.14).

We now define new matrices  $[\tilde{\mathbb{K}}]$  and  $[\tilde{\mathbb{M}}]$  which are block diagonal matrices, with each diagonal block set equal to an element matrix:

$$[\tilde{\mathbb{K}}] = \begin{bmatrix} [K_1] & [0] & \cdots & [0] \\ [0] & [K_2] & \cdots & [0] \\ \vdots & \vdots & \ddots & \vdots \\ [0] & [0] & \cdots & [K_{N_E}] \end{bmatrix}, \quad [\tilde{\mathbb{M}}] = \begin{bmatrix} [M_1] & [0] & \cdots & [0] \\ [0] & [M_2] & \cdots & [0] \\ \vdots & \vdots & \ddots & \vdots \\ [0] & [0] & \cdots & [M_{N_E}] \end{bmatrix}$$

where  $N_E$  is the number of elements in the mesh. Let us now consider the new generalized eigenvalue problem:

$$[\tilde{\mathbb{K}}]\{\tilde{\mathbb{X}}\} - \tilde{\lambda}[\tilde{\mathbb{M}}]\{\tilde{\mathbb{X}}\} = \{0\} \quad (5.54)$$

whose size  $\tilde{N} = n_{ne}N_E$  is clearly larger than  $N$ . The set of eigenvalues for problem (5.54) is the collection of the eigenvalues of problems (5.52) considered for all elements  $1 \leq e \leq N_E$ , as a result of the block-diagonal structure of  $[\tilde{\mathbb{K}}]$  and  $[\tilde{\mathbb{M}}]$ .

The sign properties of all element matrices  $[M_e]$ ,  $[K_e]$  imply that  $[\tilde{\mathbb{M}}]$  and  $[\tilde{\mathbb{K}}]$  are respectively positive definite and positive semi-definite. Moreover, applying inequality (5.51) with  $[\mathbb{M}] = [\tilde{\mathbb{M}}]$  and  $[\mathbb{K}] = [\tilde{\mathbb{K}}]$ , one obtains

$$\tilde{\mathcal{R}}(\{\tilde{\mathbb{V}}\}) := \frac{\{\tilde{\mathbb{V}}\}^T [\tilde{\mathbb{K}}] \{\tilde{\mathbb{V}}\}}{\{\tilde{\mathbb{V}}\}^T [\tilde{\mathbb{M}}] \{\tilde{\mathbb{V}}\}} \leq (\omega_{\max}^E)^2$$

Let  $\{\mathbb{V}^{\max}\} \in \mathbb{R}^N$  be a maximizer of  $\mathcal{R}(\{\mathbb{V}\})$ . From  $\{\mathbb{V}^{\max}\}$ , we build the element vectors  $\{V_e^{\max}\}$  which, for each element, contain the relevant nodal values of  $\{\mathbb{V}^{\max}\}$  (in particular setting to zero all entries corresponding to homogeneous boundary conditions), and the vector  $\{\tilde{\mathbb{V}}\} \in \mathbb{R}^{\tilde{N}}$  by concatenating the  $\{V_e^{\max}\}$ :

$$\{\tilde{\mathbb{V}}\}^T = \left\{ \{V_1^{\max}\}^T \quad \{V_2^{\max}\}^T \quad \dots \quad \{V_{N_E}^{\max}\}^T \right\}$$

Now, exploiting again the block-diagonal structure of  $[\tilde{\mathbf{M}}]$ ,  $[\tilde{\mathbf{K}}]$  and by additivity:

$$\begin{aligned}\{\tilde{\mathbf{V}}\}^T [\tilde{\mathbf{M}}] \{\tilde{\mathbf{V}}\} &= \sum_{e=1}^{N_E} \{V_e^{\max}\}^T [M_e] \{V_e^{\max}\} = \{\mathbf{V}^{\max}\}^T [\mathbf{M}] \{\mathbf{V}^{\max}\} \\ \{\tilde{\mathbf{V}}\}^T [\tilde{\mathbf{K}}] \{\tilde{\mathbf{V}}\} &= \sum_{e=1}^{N_E} \{V_e^{\max}\}^T [K_e] \{V_e^{\max}\} = \{\mathbf{V}^{\max}\}^T [\mathbf{K}] \{\mathbf{V}^{\max}\}\end{aligned}$$

This concludes the proof since

$$(\omega_{\max}^E)^2 \geq \tilde{\mathcal{R}}(\{\tilde{\mathbf{V}}\}) = \frac{\{\tilde{\mathbf{V}}\}^T [\tilde{\mathbf{K}}] \{\tilde{\mathbf{V}}\}}{\{\tilde{\mathbf{V}}\}^T [\tilde{\mathbf{M}}] \{\tilde{\mathbf{V}}\}} = \frac{\{\mathbf{X}^{\max}\}^T [\mathbf{K}] \{\mathbf{X}^{\max}\}}{\{\mathbf{X}^{\max}\}^T [\mathbf{M}] \{\mathbf{X}^{\max}\}} = \omega_{\max}^2$$

Going back to the original problem of estimating  $(\Delta t)_{\text{stab}}$ , using the established inequality (5.53) in (5.44) yields

$$(\Delta t)_{\text{stab}} \geq (\Delta t)_{\text{stab}}^E := \frac{1}{\omega_{\max}^E} \frac{2}{\sqrt{2\gamma - 4\beta}} \quad (5.55)$$

In other words, the estimate  $(\Delta t)_{\text{stab}}^E$  of  $(\Delta t)_{\text{stab}}$  found by solving the element eigenvalue problems (5.52) is conservative in that the Newmark scheme is guaranteed to be stable for any  $\Delta t \leq (\Delta t)_{\text{stab}}^E$ .

**Exercise 5.4.** Determine analytically the estimate  $(\Delta t)_{\text{stab}}^E$  of  $(\Delta t)_{\text{stab}}$  for the elastic rod (Sec. 5.3.2), if using elements of equal length  $h$ . Consider the cases of 2-node elements (piecewise linear spatial interpolation) and 3-node elements (piecewise quadratic spatial interpolation). Consider also, for both cases, the variant where mass condensation is used.

## 5.5 Conservation of mechanical energy

Another approach for analyzing the properties of time-stepping algorithms for the semi-discretized equations of structural dynamics consists in evaluating the variation of mechanical energy between two consecutive discrete time instants. Consider the homogeneous differential system

$$[\mathbf{K}]\{\mathbf{U}\} + [\mathbf{M}]\{\ddot{\mathbf{U}}\} = \{0\} \quad (5.56)$$

with nonzero initial conditions  $\{\mathbf{U}_0\}, \{\dot{\mathbf{U}}_0\}$ : this corresponds to evaluating the dynamical response of a structure undergoing an initial perturbation relative to a natural (rest) configuration and then left to evolve without further excitation.

Left-multiplying (5.56) by  $\{\dot{\mathbf{U}}\}^T$  gives

$$\{\dot{\mathbf{U}}\}^T [\mathbf{K}] \{\mathbf{U}\} + \{\dot{\mathbf{U}}\}^T [\mathbf{M}] \{\ddot{\mathbf{U}}\} = 0,$$

a relation that can be recast as an exact time derivative and in fact expresses the conservation of the total mechanical energy:

$$\frac{d}{dt} [\mathcal{W}_h(t) + \mathcal{K}_h(t)] = 0 \quad (5.57)$$

where  $\mathcal{K}_h(t)$  and  $\mathcal{W}_h(t)$  denote the kinetic and strain energies of the semi-discretized structure at time  $t$ :

$$\mathcal{K}_h(t) := \frac{1}{2} \{\dot{\mathbf{U}}\}^T [\mathbf{M}] \{\dot{\mathbf{U}}\} \quad \mathcal{W}_h(t) := \frac{1}{2} \{\mathbf{U}\}^T [\mathbf{K}] \{\mathbf{U}\} \quad (5.58)$$

For a given time-stepping scheme, it is then interesting to determine its properties according to the evolution of the total mechanical energy between two consecutive discrete time instants. For instance, regarding the homogeneous evolution equation (5.56) considered here, the possible behaviors of the time-stepping scheme may be categorized as follows:

- (i) If  $\mathcal{W}_h(t_{n+1}) + \mathcal{K}_h(t_{n+1}) - \mathcal{W}_h(t_n) - \mathcal{K}_h(t_n) < 0$ , *numerical damping* occurs since the time-discrete total energy decreases while the original problem is conservative;
- (ii) If  $\mathcal{W}_h(t_{n+1}) + \mathcal{K}_h(t_{n+1}) - \mathcal{W}_h(t_n) - \mathcal{K}_h(t_n) > 0$ , *amplification* occurs since the time-discrete total energy increases: this means the time-stepping scheme is unstable;
- (iii) Si  $\mathcal{W}_h(t_{n+1}) + \mathcal{K}_h(t_{n+1}) - \mathcal{W}_h(t_n) - \mathcal{K}_h(t_n) = 0$ , the time-discrete total energy is preserved by the time-stepping scheme (neither amplification nor numerical damping occurs).

The energy-preservation properties of the Newmark schemes are now investigated. To this aim, one needs to evaluate

$$\mathcal{W}_h(t_{n+1}) - \mathcal{W}_h(t_n) + \mathcal{K}_h(t_{n+1}) - \mathcal{K}_h(t_n),$$

with the solution at time  $t_{n+1}$  deduced from that at time  $t_n$  by solving (5.36) and using update formulas (5.34). Definitions (5.58) allow to set the energy variations in the form

$$\begin{aligned} \mathcal{W}_h(t_{n+1}) - \mathcal{W}_h(t_n) &= 2 \{\mathbf{U}_n\}_s^T [\mathbf{K}] \{\mathbf{U}_n\}_d \\ \mathcal{K}_h(t_{n+1}) - \mathcal{K}_h(t_n) &= 2 \{\dot{\mathbf{U}}_n\}_s^T [\mathbf{M}] \{\dot{\mathbf{U}}_n\}_d, \end{aligned}$$

having used, for a generic vector  $\{\mathbf{X}\}$  of nodal values, the definitions

$$\begin{cases} \{\mathbf{X}_n\}_s = (\mathbf{X}_{n+1} + \mathbf{X}_n)/2 \\ \{\mathbf{X}_n\}_d = (\mathbf{X}_{n+1} - \mathbf{X}_n)/2 \end{cases} \quad \text{i.e.} \quad \begin{cases} \mathbf{X}_n = \{\mathbf{X}_n\}_s - \{\mathbf{X}_n\}_d \\ \mathbf{X}_{n+1} = \{\mathbf{X}_n\}_s + \{\mathbf{X}_n\}_d \end{cases} \quad (5.59)$$

On using the above notations, the Newmark relations (5.34) take the form

$$\begin{aligned} 0 &= -2\{\mathbf{U}_n\}_d + \Delta t \{\dot{\mathbf{U}}_n\}_s - \Delta t \{\ddot{\mathbf{U}}_n\}_d + \frac{1}{2} \Delta t^2 [\{\ddot{\mathbf{U}}_n\}_s + (4\beta - 1)\{\ddot{\mathbf{U}}_n\}_d] \\ 0 &= -2\{\dot{\mathbf{U}}_n\}_d + \Delta t [\{\ddot{\mathbf{U}}_n\}_s + (2\gamma - 1)\{\ddot{\mathbf{U}}_n\}_d], \end{aligned}$$

i.e., upon eliminating  $\Delta t \{\dot{\mathbf{U}}_n\}_d$  from the first equation by using the second equation:

$$\begin{aligned} \{\dot{\mathbf{U}}_n\}_s &= \frac{2}{\Delta t} \{\mathbf{U}_n\}_d + \Delta t (\gamma - 2\beta) \{\ddot{\mathbf{U}}_n\}_d \\ 2\{\dot{\mathbf{U}}_n\}_d &= \Delta t [\{\ddot{\mathbf{U}}_n\}_s + (2\gamma - 1)\{\ddot{\mathbf{U}}_n\}_d] \end{aligned} \quad (5.60)$$



The energy variation can now be evaluated, to obtain

$$\mathcal{W}_h(t_{n+1}) - \mathcal{W}_h(t_n) + \mathcal{K}_h(t_{n+1}) - \mathcal{K}_h(t_n) = 2\{\mathbb{U}_n\}_s^T [\mathbb{K}] \{\mathbb{U}_n\}_d + 2\{\dot{\mathbb{U}}_n\}_s^T [\mathbb{M}] \{\dot{\mathbb{U}}_n\}_d$$

On substituting expressions (5.60) into  $2\{\dot{\mathbb{U}}_n\}_s^T [\mathbb{M}] \{\ddot{\mathbb{U}}_n\}_d$  and reordering the resulting terms, exploiting the relations

$$[\mathbb{K}] \{\mathbb{U}_n\}_d + [\mathbb{M}] \{\ddot{\mathbb{U}}_n\}_d = \{0\} \quad [\mathbb{K}] \{\mathbb{U}_n\}_s + [\mathbb{M}] \{\ddot{\mathbb{U}}_n\}_s = \{0\}$$

deduced from (5.56), and using equality

$$2\{\ddot{\mathbb{U}}_n\}_s^T [\mathbb{M}] \{\ddot{\mathbb{U}}_n\}_d = \frac{1}{2}\{\ddot{\mathbb{U}}_{n+1}\}_s^T [\mathbb{M}] \{\ddot{\mathbb{U}}_{n+1}\}_s - \frac{1}{2}\{\ddot{\mathbb{U}}_n\}_s^T [\mathbb{M}] \{\ddot{\mathbb{U}}_n\}_s,$$

one finally arrives at the result

$$\begin{aligned} \mathcal{E}_h(t_{n+1}) - \mathcal{E}_h(t_n) &= 2(1 - 2\gamma)\{\mathbb{U}_n\}_d^T [\mathbb{K}] \{\mathbb{U}_n\}_d \\ &\quad + \Delta t^2(\gamma - 2\beta)(2\gamma - 1)\{\ddot{\mathbb{U}}_n\}_d^T [\mathbb{M}] \{\ddot{\mathbb{U}}_n\}_d, \end{aligned} \quad (5.61)$$

where  $\mathcal{E}_h(t)$  is defined by

$$\mathcal{E}_h(t) := \mathcal{W}_h(t) + \mathcal{K}_h(t) + \frac{\Delta t^2}{2} \left( \beta - \frac{\gamma}{2} \right) \{\ddot{\mathbb{U}}(t)\}_s^T [\mathbb{M}] \{\ddot{\mathbb{U}}(t)\}_s$$

The result (5.61) constitutes the energy conservation equation associated with the Newmark scheme.

The energy conservation identity (5.61) allows to characterize stability and numerical damping properties of the scheme. In particular:

- (a) If  $\gamma = 1/2$ , the right-hand side of (5.61) vanishes and  $\mathcal{E}_h(t)$  is preserved;
- (b) If  $\gamma \geq 1/2$  and  $2\beta - \gamma = 0$ , identity (5.61) implies that the total energy decreases during a time step:

$$\mathcal{W}_h(t_{n+1}) + \mathcal{K}_h(t_{n+1}) - \mathcal{W}_h(t_n) - \mathcal{K}_h(t_n) = 2(1 - 2\gamma)\{\mathbb{U}_n\}_d^T [\mathbb{K}] \{\mathbb{U}_n\}_d \leq 0$$

- (c) If  $\gamma = 1/2$  and  $\beta = 1/4$  (a special case of (b) corresponding to the optimally-accurate unconditionally stable scheme (5.47)), the total energy is preserved during a time step:

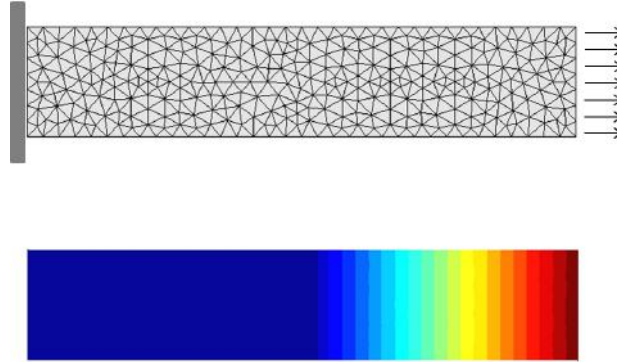
$$\mathcal{W}_h(t_{n+1}) + \mathcal{K}_h(t_{n+1}) - \mathcal{W}_h(t_n) - \mathcal{K}_h(t_n) = 0$$

- (d) If  $\gamma > 1/2$  and  $2\beta > \gamma$ ,  $\mathcal{E}_h(t)$  decreases during a time step. One notes in particular that the third term of  $\mathcal{E}_h(t)$  is positive (since  $2\beta - \gamma > 0$ ), preventing the total energy from diverging.
- (e) If  $\gamma < 1/2$  (unstability), the right-hand side of identity (5.61) is strictly positive (i) for any  $\Delta t$  if  $\gamma - 2\beta \leq 0$ , or (ii) for  $\Delta t$  sufficiently small otherwise.  $\mathcal{E}_h(t)$  may therefore diverge.

Cases (a) to (d) satisfy the stability criteria (conditionally on  $\Delta t$  for (a), unconditionally for (b) to (d)). The stability of the time-stepping scheme is thus seen to be linked to energy preservation properties.

## 5.6 The code dynamics for 2D and 3D applications

We propose here to develop the code dynamics by adapting `genlin` to the analysis of problems in elastodynamics integrated with the central difference scheme and lumped mass matrix. The revision of Section 4.2.6, which contains a similar exercise, is recommended.



**Figure 5.10:** 2D bar under Heaviside loading: mesh adopted and axial displacement at a given time  $t$  denoting a travelling wave

As for the code `heatdiffusion`, only minimal changes are required compared to the reference code `genlin`, the main effort being limited to the development of the element mass matrices required. Even if we work in full generality, we will limit our attention, for the purpose of explanation, to the example of figure 5.10, where a 2D bar in plane-strain conditions is subject to boundary conditions corresponding to the 1D example analysed in the previous sections. We assume that the domain is meshed with linear triangles (T3). The problem geometry is based on the GMSH file `Chap5_bar2D.geo`, while the analysis file is `Chap5_bar2D.m`. The only differences with respect to standard analysis files for `genlin` consist in the definition of `material`, which now contains the density  $\rho$  in the last position, and in the introduction of the parameters `tf`, denoting the total duration of the analysis and of `Dt` fixing the time step value, assumed constant over the whole analysis:

```
% MATERIAL: Young, Poisson, rho
material = [10 0.2 1];

...

% parameters for time history
tf = 5.;
Dt = 0.001;
```

Most of the element matrices required are the same as for `genlin`: `T3_2A_solid.Ke`, for the stiffness matrix, `T3_2A_solid.Fe_surf`, for the nodal forces due to surface tractions, and the routines for post-processing stresses. Actually, only the function providing the element mass matrix must be included in the `subT3` directory. Since we are addressing an explicit scheme, we will provide a lumped mass matrix in the form of a 1D list containing the diagonal coefficients. Since the shape functions are linear,

each entry of the list will be equal to one third of the total mass  $S \cdot \rho$ :

```
function Me=T3_2A_solid_Me(X,mate)

rho=mate(3);
x11=X(1,1); x21=X(2,1); x31=X(3,1);           % nodal coordinates
x12=X(1,2); x22=X(2,2); x32=X(3,2);
S=.5*((x21-x11)*(x32-x12)-...
      (x31-x11)*(x22-x12));                     % element area
Me=1/3*S*rho*ones(6,1);
```

It is easy to verify that the sum of all the coefficients in  $Me$  is equal to twice the total mass of the element.

**Exercise 5.5** (lumped mass matrix with mass preservation). *Starting from 5.13, show that the sum of all the coefficients of a consistent mass matrix are equal to twice the total mass  $M_e$  of the element. Next show that if mass preservation is required, any scheme for lumped mass must generate a mass vector which sums to  $2M_e$*

Only two sections of `genlin` must be adapted to run the analysis at hand. The former is the assemblage phase, in which the mass vector  $M$  must be generated:

```
Me=eval([Etag Atag 'Me(Xe,material(mat,:))']);
...
M(Ie)=M(Ie)+Me(Le0);
```

The latter is the time integration history, which closely follows the scheme in Box 5.1. Assuming homogeneous initial conditions,

```
U=zeros(neq,1);           % initial displ
Ud=zeros(neq,1);          % initial velocity

Udd=M.\F;                  % initial value of accel.
Ud=Ud+Dt/2*Udd;           % and of velocities

nstep=floor(tf/Dt);        % number of time steps
for step=1:nstep,          % loop over time steps
    U=U+Dt*Ud;              % actualisation of displ
    Fg=F-K*U;               % rhs without contact
    Udd=M.\Fg;              % acceleration
    Ud=Ud+Dt*Udd;           % velocity
end
```

At the end of the time integration the final values of displacements are post-processed as in `genlin`

**Exercise 5.6** (stability properties: numerical verification). *Analyse the 2D bar applying the code dynamics using different time steps and different levels of mesh refinement to verify the stability properties.*

**Exercise 5.7** (Newmark scheme - consistent mass matrix). *Modify dynamics introducing the Newmark scheme to integrate the time history with a consistent mass matrix.*

**Exercise 5.8** (dynamics with T6 elements). *Mesh the bar with quadratic triangles T6 and repeat the analysis after taking the necessary provisions.*

## 5.7 Introduction to the spectral element method

The spectral element method is a particular version of the finite element method that is well-suited to explicit time-stepping methods, for reasons that will become clear. It is used in particular for applications in large-scale computational geophysics (SEM, see e.g. Komatitsch and Vilotte, 1998). Here, we briefly explain the underlying ideas on the 1-D equation (5.23) and its weak formulation (5.25).

The spectral element method rests, in the 1-D case, upon a particular choice of the quadrature rule (3.22):

$$\int_{-1}^1 f(a) da \approx \sum_{g=1}^G w_g f(a_g)$$

(repeated for convenience), used for the approximate computation of element stiffness and mass matrices (Sec. 3.2.5). Specifically, the usual choice of a Gauss-Legendre quadrature rule is dropped in favor of a Gauss-Lobatto rule. Rules of the latter family are such that

- (a) The interval endpoints are quadrature points, i.e. (with appropriate numbering):  $a_1 = -1$ ,  $a_G = 1$  (whereas Gauss-Legendre points always lie inside the interval);
- (b) The  $G$ -point Gauss-Lobatto rule integrates exactly all polynomials of degree at most  $2G - 3$  (whereas the maximum degree is  $2G - 1$  for the  $G$ -point Gauss-Legendre rule).

Letting any linear element  $E = [x^{(1)}, x^{(2)}]$  be defined from the reference element  $\Delta = [-1, 1]$  by means of an *affine* mapping

$$x \in E = x_E(a) \quad \text{with} \quad x_E(a) := \frac{1}{2}(x^{(2)} + x^{(1)}) + \frac{1}{2}ha \quad (a \in \Delta),$$

(with  $h := x^{(2)} - x^{(1)}$  denoting as usual the element size) the second main idea behind the SEM is that  $E$  has  $G$  nodes  $x^{(1)}, \dots, x^{(G)}$ , which moreover are identical to the quadrature points:

$$x^{(g)} = x_E(a_g), \quad 1 \leq g \leq G.$$

The interpolation functions  $N_p(a)$  ( $1 \leq p \leq G$ ) are then taken as the usual Lagrange polynomials of degree  $G - 1$ , i.e.:

$$N_p(a) = \frac{\prod_{g \neq p} (a - a_g)}{\prod_{g \neq p} (a_p - a_g)}$$

They are such that  $N_p(a_g) = \delta_{pg}$  ( $1 \leq p, g \leq G$ ). Note that these definitions imply that the element  $E$  is *not* isoparametric (since the geometry representation uses an affine mapping).

The element stiffness matrix is computed from (5.26), by applying the Gauss-Lobatto quadrature. Note that the interpolation function derivatives have degree  $G - 2$ , implying that the quadrature evaluates  $[K_e]$  exactly.

The element mass matrix is computed using (5.27). However, the property  $N_p(a_g) = \delta_{pg}$  implied by the choice of nodal coordinates implies that the Gauss-Lobatto quadrature yields a *diagonal* mass matrix:

$$[M_e] = \frac{\rho h}{2} \text{Diag}(w_1, \dots, w_G)$$

The global mass matrix is therefore diagonal as well, so that the SEM naturally achieves mass lumping. That makes it very suitable for explicit time-stepping schemes, since no solution of linear systems is required. Note however that the adopted quadrature does not permit exact evaluation of  $[M_e]$  since the interpolation functions have degree  $G - 1$ .

Spectral element methods for 2D or 3D problems use box-shaped elements that are obtained by an affine transformations of the unit square or cube, together with Cartesian products of 1-D Gauss-Lobatto quadrature rules. This approach preserves the diagonal character of the mass matrix, while severely constraining the element shapes (for 3-D, to rectangular boxes with planar faces). This method thus is suitable for modelling e.g. seismic waves propagating in large domains, but less so for the analysis of structures with complex geometry.

The SEM is usually applied using high-degree interpolation. The denomination "spectral" is indeed related with the notion of refining the approximation by increasing the degree  $G$  rather than decreasing the element size  $h$ . The mathematical analysis of convergence for the SEM shows that the solution error rapidly decreases with  $G$  for problems having a very smooth exact solution (which again is not normally expected of structural analyses due to geometry or material irregularities).

## 5.8 Additional examples and exercises

### 5.8.1 Explosion in a spherical cavity

Let us consider a spherical cavity of radius  $R_1$  in an infinite, linear elastic isotropic and homogeneous medium. The inner surface is subjected to a given pressure history  $p(t)$ . The analytical solution is available in the following convolutive form:

$$u_r = \frac{1}{c_L} \frac{1}{r} f'(\tau) - \frac{1}{r^2} f(\tau), \quad \tau = t - \frac{1}{c_L}(r - R_1) \quad (5.62)$$

$$f(\tau) = \frac{R_1}{\rho \beta} \int_0^\tau p(\xi) e^{-\gamma(\tau-\xi)} \sin(\beta(\tau-\xi)) d\xi \quad (5.63)$$

where:

$$\gamma = \frac{1-2\nu}{1-\nu} \frac{c_L}{R_1}, \quad \beta = \frac{\sqrt{1-2\nu}}{1-\nu} \frac{c_L}{R_1}$$

Since we are interested in the displacement history immediately after the initial application of the pressure on the inner surface, we will truncate the analysis domain in  $r = R_2$ , where we enforce  $u_r = 0$ . Therefore (see also Section 5.8.2), the validity of the analysis is limited to  $t < (R_2 - R_1)/c_L$ .

The weak form of equilibrium can be obtained by adding to the static counterpart (1.67) the contribution of inertia forces:

$$\int_{R_1}^{R_2} \rho \frac{\partial^2 u_r}{\partial t^2} w r^2 dr + \int_{R_1}^{R_2} \left( \sigma_{rr} \frac{dw}{dr} + (\sigma_{\theta\theta} + \sigma_{\varphi\varphi}) \frac{w}{r} \right) r^2 dr = p R_1^2 w(R_1) \quad \forall w \in \mathcal{C}(0). \quad (5.64)$$

Let the radial line  $[R_1, R_2]$  be discretized via quadratic B3 elements (see Exercise 2.1). The computation of the element mass matrix requires the numerical integration of terms like  $\{N(\mathbf{a})\} \{N(\mathbf{a})\}^T r^2$ , where  $\{N\}$  is the column vector collecting the three shape functions. When the elements are not distorted  $r$  depends linearly on  $a$  and hence we need to integrate a polynomial of order 6. A complete integration thus requires at least 4 Gauss points:

```
function Me=B3_1S_solid_Me(X,rho,lumped);

a1=sqrt((3-2*sqrt(6/5))/7); a2=sqrt((3+2*sqrt(6/5))/7);
a_gauss=[-a2 -a1 a1 a2];
w1=1/36*(18-sqrt(30)); w2=1/36*(18+sqrt(30));
w_gauss=[w1 w2 w2 w1];
Me=zeros(3,3);
for g=1:length(a_gauss),
    a=a_gauss(g);
    N=[.5*a*(a-1) (1-a^2) .5*a*(1+a)]';
    r=N'*X;
    D=[a-.5 -2*a a+.5]';
    J=X'*D;
    Me=Me+rho*r^2*N*N'*J*w_gauss(g);
end
if lumped==1
    Me=diag(sum(Me));
end
```

As for the analysis of the elastic rod, two different loadings are proposed. In the former case  $p(t) = p_0 H(t)$ , corresponding to a constant Heaviside pressure. From (5.62) we have:

$$f(\tau) = -\frac{p_0 R^2}{2\rho c_L} \left( \frac{1}{\gamma} - e^{-\gamma\tau} \left( \frac{1}{\gamma} \cos(\beta\tau) + \frac{1}{\beta} \sin(\beta\tau) \right) \right)$$

The latter, more stringent, test is performed with the choice  $p(t) = p_0 \delta(t)$ , simulating a blast pressure, for which:

$$f(\tau) = \frac{p_0 R}{\rho \beta} e^{-\gamma\tau} \sin(\beta\tau)$$

**Exercise 5.9** (explosion in a spherical cavity). *Simulate the two loadings proposed using both modal and direct integration approaches; compare with the analytical solution.*

### 5.8.2 Simulation of infinite domains with Absorbing Boundary Conditions

Let us consider a semi-infinite elastic bar placed along the  $x$  axis ( $-\infty < x \leq L$ ). For this 1D problem the dynamic equilibrium equation (5.23) again holds, where  $u$  is the horizontal displacement and  $c = \sqrt{E/\rho}$  the wave velocity.

The bar is initially at rest. A traction  $T^D(t)$  (force per unit area) is applied at the end point on the right ( $x = L$ ). The exact solution can be expressed as  $u(x, t) = f(x + ct)$  (for  $t > (L - x)/c$ ), where  $f$  depends on the loading and on material parameters and represents a wave travelling towards the left.

In order to define a bounded computational domain for the numerical simulation of the displacement history, the bar is truncated at  $x = 0$ , the finite domain  $x \in [0, L]$  discretized with finite elements, and a Newmark time stepping scheme employed.

Prescribing a zero displacement at the artificial boundary  $x = 0$  only works for the time duration  $t < L/c$ , i.e. before any wave arrives at  $x = 0$ . To permit longer analysis durations, the absorbing boundary condition (ABC)

$$c \frac{\partial u}{\partial x}(0, t) - \frac{\partial u}{\partial t}(0, t) = 0 \quad (t \geq 0) \quad (5.65)$$

is prescribed instead. A simple verification shows that any left-travelling wave  $f(x + ct)$  exactly respects (5.65). As a consequence no reflected waves arise and the impinging waves are “absorbed”. Moreover, if  $T^D(t) = TH(t)$  (where  $T$  is a constant and  $H$  the Heaviside function), the tip displacement  $u(L, t)$  is expected to increase linearly with time.

The weak formulation for the dynamical problem incorporating the ABC (5.65) reads

$$\int_0^L \left( E \frac{\partial u}{\partial x} \frac{\partial w}{\partial x} + \rho \frac{\partial^2 u}{\partial t^2} w \right) dx = - \frac{E}{c} \frac{\partial u}{\partial t}(0, t) w(0) + T^D(t) w(L), \quad \forall w \in \mathcal{C}, \quad (5.66)$$

since  $\mathcal{C}(0)$  coincides now with  $\mathcal{C}$  (no constraints at  $x = 0$ ).

To solve the transition problem  $t_n \rightarrow t_{n+1}$ , equation (5.66) is imposed at time  $t_{n+1}$ , as usual. A time discretization must be applied also to the new term on the right-hand side arising from the ABC:

$$\frac{E}{c} \frac{\partial u}{\partial t}(0, t_{n+1}) w(0) \approx \frac{E}{c} \dot{U}_{n+1}^{(1)} W^{(1)}$$

where  $U^{(1)}$  denotes the nodal displacement at  $x = 0$ . This is done in accordance with the Newmark scheme (5.34) adopted, i.e.:

$$\dot{U}_{n+1}^{(1)} = \dot{U}_n^{(1)} + \Delta t \left( (1 - \gamma) \ddot{U}_n^{(1)} + \gamma \ddot{U}_{n+1}^{(1)} \right) \quad (5.67)$$

Relative to the code `bar_B2_1D_wave_direct`, since also the displacement in  $x = 0$  is unknown, the number of unknowns and the dof list must be modified:

```
neq=NN;
dof=[1:1:NN];
```

The assemblage is the same for all elements (no distinction between the first and the others). The term  $(E/c)\Delta t\gamma\ddot{U}_{n+1}^{(1)}$  coming from eq.(5.67) is added to M.

```

for e=1:NE, % loop over elements to assemble matrices
    X=[coor(e) coor(e+1)]; % creates segment
    dofe=dof(e:e+1);
    pe=find(dofe>0);
    Ie=dofe(pe); % gets value of associated DOFs
    Me=B2_1D_wave_Me(X,rho); % element mass consistent matrix
    M(Ie,Ie)=M(Ie,Ie)+Me(pe,pe); % assemblage of mass matrix
    Ke=B2_1D_wave_Ke(X,E); % element stiffness matrix
    K(Ie,Ie)=K(Ie,Ie)+Ke(pe,pe); % assemblage of stiffness matrix
end
F(neq)=TD; % rhs
M(1,1)=M(1,1)+E/c*gamma*Dt; % contrib from ABC

```

Finally, at every step of the time history, the new contribution

$$-(E/c) \left( \dot{U}_n^{(1)} + \Delta t(1-\gamma)\ddot{U}_n^{(1)} \right),$$

which also comes from (5.67), is added to the right-hand-side term:

```

for step=1:nstep,
    U=U+Dt*Ud+0.5*Dt^2*(1-2*beta)*Udd;
    Ud=Ud+Dt*(1-gamma)*Udd;
    F(1)=-E/c*Ud(1); % contrib to rhs from ABC
    Fg=F-K*U;
    Udd=CS\((CS'\Fg);
    U=U+Dt^2*beta*Udd;
    Ud=Ud+Dt*gamma*Udd;
end

```

**Exercise 5.10** (Absorbing boundary condition for a 1D elastic bar problem). *Apply the required changes to bar\_B2\_1D\_wave\_direct and compare the numerical solution with the analytical prediction for the displacement at the tip and at the base of the bar*

### 5.8.3 The patch test

The *patch test* is employed to test the correct implementation in a code of a new element. It exploits the fact that all isoparametric elements are capable of representing exactly any displacement field with constant gradient (Sec. 2.2.6).

We consider as an example a patch of four quadrilateral isoparametric elements, whose boundary has straight edges (Fig. 5.11).

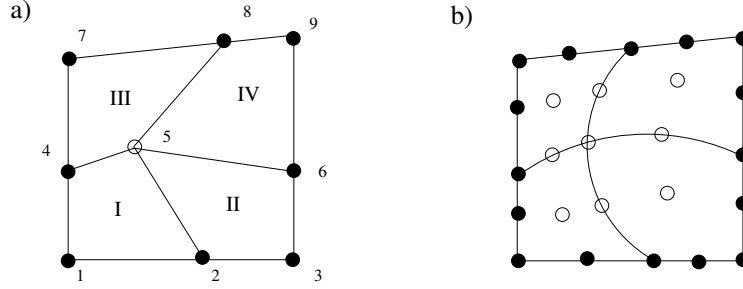
**Static patch test.** For the static patch test, the displacement is prescribed at all boundary nodes (filled symbols) according to

$$u(x) = A.x + b \quad (5.68)$$

(where  $A$  is a constant  $D \times D$  matrix,  $b$  is a constant  $D$ -vector and  $x$  is any of the nodes on the patch boundary). Assuming equilibrium without body forces, the field (5.68) then is in fact the displacement solution in the whole patch, and in particular at all



internal nodes (empty symbols) of e.g. the patches of Figure 5.11. The patch test then consists in numerically solving this problem and checking that the displacement at the internal nodes is indeed equal to (5.68).



**Figure 5.11:** Two examples of a quadrangular patch made of four quadrilateral elements. The displacement or velocity is prescribed at all boundary nodes (filled symbols), and the solution is sought at the internal nodes (empty symbols).

**Dynamical patch test.** For the dynamical patch test, the spatially affine field (5.68) is used for defining prescribed velocities instead of displacements:

$$\dot{\mathbf{u}}(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x} + \mathbf{b}$$

Assuming zero initial displacements and initial velocities given by the above formula, the elastodynamic displacement field is then given in the patch and at any time by

$$\mathbf{u}(\mathbf{x}, t) = (\mathbf{A} \cdot \mathbf{x} + \mathbf{b})t$$

Consider now the application of the Newmark time-stepping algorithm (Box 5.2) under the above conditions:

- Step 2:  $\{\ddot{\mathbf{U}}_0\} = 0$  since  $\{\mathbf{U}_0\} = \{\mathbf{F}_0\} = 0$ ;
- Step 4a ( $n = 0$ ): one finds  $\{\dot{\mathbf{U}}_1^{\text{pred}}\} = \{\dot{\mathbf{U}}_0\}$  and  $\{\mathbf{U}_1^{\text{pred}}\} = \Delta t \{\dot{\mathbf{U}}_0\}$ , which are the values of the exact solution at the end of this time step;
- Step 4b ( $n = 0$ ): since  $\{\mathbf{U}_1^{\text{pred}}\}$  coincides with the static solution with prescribed boundary displacement  $\Delta t \{\dot{\mathbf{U}}_0\}$ , one has  $\{\mathbf{F}\}_1 - [\mathbb{K}] \{\mathbf{U}_1^{\text{pred}}\} = 0$ , and consequently  $\{\ddot{\mathbf{U}}_1\} = 0$ ;
- Step 4c: the updated displacements and velocities are therefore  $\{\mathbf{U}_1^{\text{pred}}\} = \{\mathbf{U}_1\}$  and  $\{\dot{\mathbf{U}}_1^{\text{pred}}\} = \{\dot{\mathbf{U}}_1\}$ ;
- The above reasoning for Steps 4a-4c can be checked (exercise) to hold again for any subsequent time step, with  $\{\ddot{\mathbf{U}}_n\} = 0$  for all  $n$ .

The Newmark time-stepping therefore must reproduce exactly the spatially-affine, constant-velocity solution. The dynamical patch test then consists in numerically solving this dynamical problem and checking that the expected solution is indeed found at all internal nodes of the patch.

### 5.8.4 Contact between a tire and a rigid surface

Explicit time-stepping algorithms are often used for strongly nonlinear dynamical analyses, in which cases convergence issues may arise if implicit algorithms are used instead. A typical example is the simulation of dynamical contact between elastic solids. Even if non-linear solid mechanics is the topic of Chapter 8, in what follows we develop an explicit scheme on a very simplified example. Starting from the code `dynamics` (Section 5.6) implementing an explicit central difference scheme, simple provisions are taken in order to simulate the contact of a deformable solid with a rigid horizontal surface. The results of this application are collected in the code `dynamics_tire`.

Consider a tire located above the rigid planar horizontal surface  $x_2 = 0$ , with homogeneous initial conditions and otherwise subjected only to gravity  $\mathbf{f} = -\rho g \mathbf{e}_2$  (data files for this example have names of the form `Chap5_tire.*`). Let  $\mathbf{T}$  denote contact forces exerted by the rigid surface on the tire. Contact is for simplicity assumed to be frictionless, so that  $T_1 = 0$ . The non-penetration conditions (8.1) discussed in Section 8.1.1 take the simple form

$$T_2 \geq 0 \quad (x_2 + u_2) \geq 0 \quad T_2(x_2 + u_2) = 0,$$

where  $x_2$  denotes the initial coordinate and  $u_2$  the vertical displacement. Instead of being enforced exactly, these conditions are often handled via a penalty approach when using explicit time-stepping: once displacements are updated in the algorithm (step 4(a) in Box 5.1), the distance between each point of the tire to the rigid surface is evaluated. At any point where this evaluation predicts a nonzero penetration depth  $\delta$ , a vertical force  $\lambda\delta$  is exerted along the opposite direction on the tire ( $\lambda$  being a positive multiplier). This amounts to replace the rigid surface by a fictitious bed of springs with stiffness  $\lambda$ , the case of a rigid surface corresponding to the limiting case  $\lambda \rightarrow +\infty$ . This force is involved in the dynamical balance equation from which the accelerations  $\{\ddot{\mathbf{U}}_{n+1}\}$  are found. Assume the stiffness and (lumped) mass matrices  $\mathbf{K}$ ,  $\mathbf{M}$ , as well as the generalized load vector  $\mathbf{F}$ , have been assembled. In particular, the element lumped mass matrix for a T3 finite element is diagonal, given by the identity matrix times  $(1/3)\rho S$ , while the element generalized load vector associated to gravity loads is an array where entries associated with the vertical components of nodal displacements are given by  $(1/3)\rho g S$ . Note the absence of any kinematical boundary condition for this example.

The analysis file `Chap5_tire.m` specifies the body force vector and the penalty coefficient  $\lambda$  as additional parameters:

```
% body forces val
bf = [0 -.5];

% parameters for contact
penalty = 20;
```

The structure of the time integration scheme is identical to what discussed in Section 5.6. Each time step begins with a displacement update, followed by an evaluation of the right-hand side where contact forces are ignored. Current nodal locations are found by adding displacements to initial nodal locations. Nodal forces obtained by multiplying the penetration and the penalty coefficient are then applied at all nodes for which penetration occurs. Note that penetration is checked at all nodes for the sake of code simplicity, resulting in the possibility of penalty forces being also applied to

nodes that are inside the tire. Of course, a sufficiently large penalty coefficient will in practice allow to eliminate such occurrences (with code alterations left to the reader as an exercise). The ensuing steps are as specified in the following lines of code

```

for step=1:nstep,                                % loop over time steps
    U=U+Dt*Ud;                                    % actualisation of displacements
    Fg=F-K*U;                                      % rhs without contact

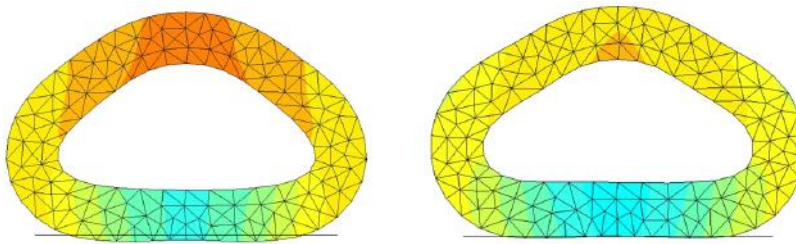
    for n=1:analysis.NN                          % loop over the nodes
        dof=nodes(n).dof(2);
        if dof>0                                  % if active dof
            y=nodes(n).coord(2)+U(dof);
            if y<0,                               % and if compenetrations is present
                Fg(dof)=Fg(dof)-penalty*y;        % adds contact force
            end
        end
    end

    Udd=M.\Fg;                                    % computes acceleration
    Ud=Ud+Dt*Udd;                                % updates velocity

end
    
```

It is again emphasized that this example is but a simple application to a specific situation. With a minimal effort, the code can be extended to other situations (this is left as an exercise). In fact, this analysis is constrained by other assumptions:

1. Large displacements but small rotations are assumed (using the non-linear Green-Lagrange strain tensor (8.16) would otherwise be necessary);
2. The proposed evaluation of nodal contact forces is reasonable only for meshes that are both uniform and sufficiently fine. Failing that, nodal forces proportional to the length of the element portion subject to penetration should be specified on each element.



**Figure 5.12:** Snapshots of the deformed tire over the rigid surface: effect of different penalty coefficients compared at the same time value

The reader will notice that the presented approach, while very simple to implement, is also very severely limited by the small value of the largest permissible time step (which the code computes accurately for comparison purposes). When the critical time step is reached, the instability manifests itself strongly and is easily made graphically visible.



## **Part II**

# **Advanced topics**



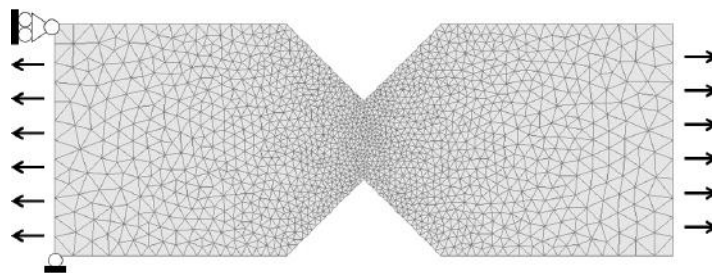
## Where isoparametric formulations fail. Penalty approaches, mixed formulations

---

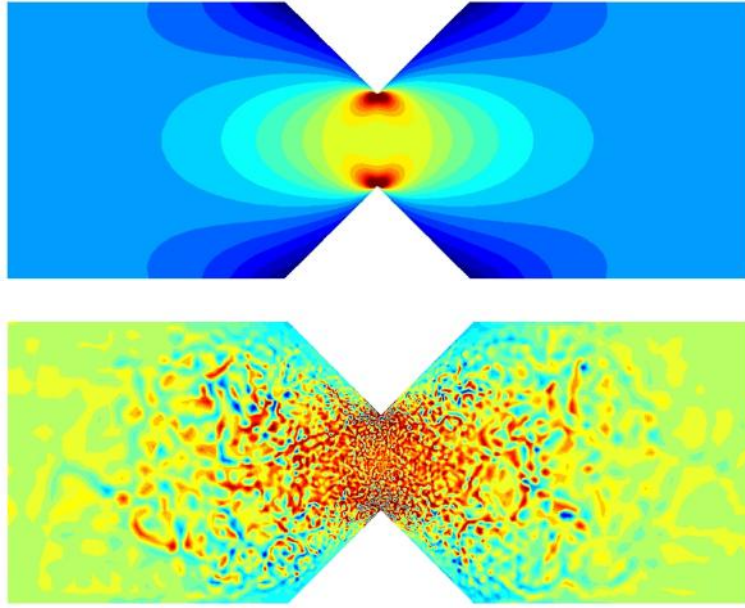
The displacement based Finite Element Method employing isoparametric elements is by far the most popular numerical technique in the field of solid and structural mechanics. However, severe difficulties may arise in specific *constrained* problems. Several examples are available in engineering applications.

The kinematic constraint of incompressibility, implying that strains are purely deviatoric, appears associated, e.g., with rubber, slow- or micro-flows and predominantly plastic strains in metals. Let us consider the specimen with a wedge in figure 6.1 created using the input file Chap6\_wedge. If  $\nu = 0.3$  an accurate pattern of stress concentration is obtained around the edges even if the linear T3 elements are employed (Fig. 6.2, top). However, if the same analysis is repeated for an almost incompressible material with  $\nu = 0.4999$  (i.e. very close to  $\nu = 1/2$ ), the stress map appears to be completely meaningless (Fig. 6.2, bottom).

A second example of constrained problem is provided by slender beams in bending, for which shear strains vanish; a similar phenomenon occurs in thin plate and shell models. If a slender cantilever beam subjected to a tip load is analysed using standard FEM, the tip deflection converges very slowly to the exact value predicted by the theory of Euler-Bernoulli beams, as illustrated in Figure 6.3. A mesh made of 200 Q4 elements (input files Chap6\_cantilever\_Q.\*) still produces an unacceptable error of 5%. The situation drastically improves with T6 elements (input files Chap6\_cantilever\_T.\*).



**Figure 6.1:** Specimen with a wedge: geometry and mesh



**Figure 6.2:** Specimen with a wedge: contour plot of  $\sigma_{11}$  with  $\nu = 0.3$  (top) or  $\nu = 0.4999$  (near-incompressible case, bottom)

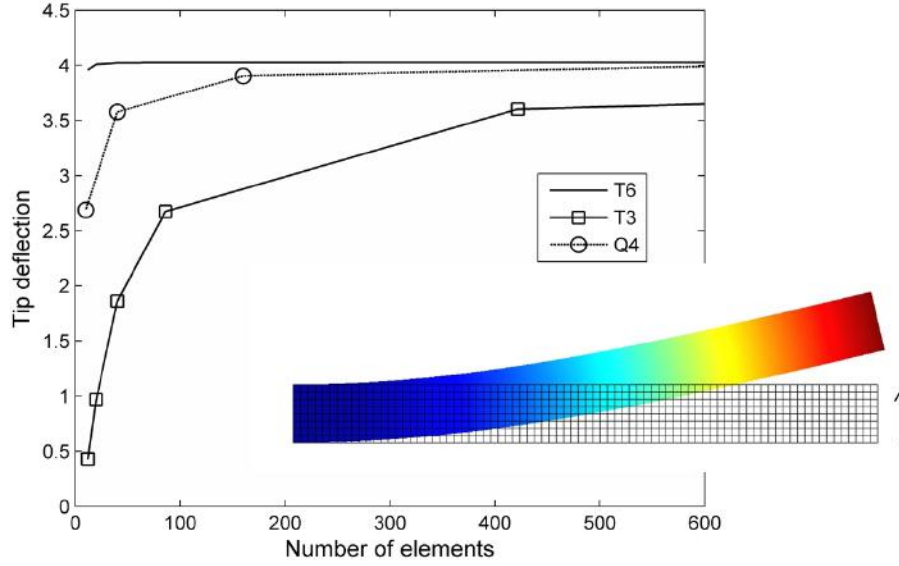
We will show that if the numerical model fails to decouple exactly the different deformation modes (e.g. deviatoric vs. volumetric for the constraint of incompressibility, or shear vs. bending for beams) the model may become exceedingly stiff and induce the phenomenon of *locking*. The incompressibility constraint and the difficulties arising from it are studied in Section 6.1, together with existing heuristic remedies (penalty approaches, selective integration). A more rigorous and systematic approach based on mixed variational formulations, which are specifically designed to overcome such locking phenomena, is then presented in Section 6.2. Section 6.3 introduces the shear locking phenomenon for the bending of a slender Timoshenko beam, and demonstrates reduced integration and a mixed formulation in that context.

However, the deficiencies of isoparametric displacement finite elements are not limited to locking problems. Several examples can be found, for instance, in the complex world of numerical fluid mechanics. We will only discuss a simple example (Sec. 6.4) where the diffusion-transport equation is introduced, and show that when the transport terms dominate over the diffusion ones, boundary layers with steep gradients appear and cannot be resolved with the standard FE approach.

## 6.1 The incompressibility constraint

We will start by focusing, as a model problem, on the specific case of a specimen made of incompressible linear elastic isotropic material subjected to a given





**Figure 6.3:** Slender cantilever beam: convergence of the tip deflection employing uniform meshes and different type of elements (T3,T6,Q4)

external quasi-static loading (inertia forces are neglected). We will assume that stresses are bounded, which means that there exists a bounded coefficient  $M$  such that  $|\sigma_{ij}| < M$  in  $\Omega$ .

When the Poisson coefficient  $\nu$  approaches the limiting value  $1/2$ , the material becomes incompressible and  $\text{div } \mathbf{u}$  tends to vanish. Indeed, let the constitutive law be expressed in terms of the Lamé constants  $\lambda = 2\nu\mu/(1 - 2\nu)$  and  $2\mu = E/(1 + \nu)$ :

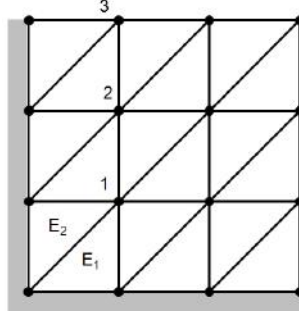
$$\boldsymbol{\sigma} = \lambda(\text{Tr} \boldsymbol{\varepsilon}) \mathbf{I} + 2\mu \boldsymbol{\varepsilon} = \lambda(\text{div } \mathbf{u}) \mathbf{I} + 2\mu \boldsymbol{\varepsilon}.$$

Letting  $\nu \rightarrow 1/2$  necessarily implies  $\text{div } \mathbf{u} \rightarrow 0$  since  $\lambda \rightarrow \infty$  and  $\boldsymbol{\sigma}$  is bounded. If the classical displacement-based FEM introduced in previous chapters is adopted, severe difficulties arise, as shown by means of examples in the following two sections devoted to linear triangles T3 and bilinear quadrangles Q4.

### 6.1.1 Incompressibility and T3 elements

Let us consider the specimen of figure 6.4 meshed with  $N_E$  linear triangles T3. Strains  $\boldsymbol{\varepsilon}[\mathbf{u}_h]$  are constant over each triangle and the condition  $\text{div } \mathbf{u}_h = 0$  is satisfied everywhere if and only if the area of each triangle is preserved. We hence have  $N_E$  constraint equations imposed on nodal displacements.

The total number  $N_N$  of nodes can be written as  $N_N = N_N^I + N_N^U + N_N^T$ , where  $N_N^I$ ,  $N_N^U$  and  $N_N^T$  denote the number of interior nodes, of nodes placed on the boundary  $S_u$  and of nodes on the boundary  $\partial\Omega_h \setminus S_u$ , respectively. For a simply connected



**Figure 6.4:** Locked mesh: the nodes of the T3 elements cannot move due to the incompressibility constraint

planar  $\Omega$ , Euler's formula states that  $S + N_E = A + 1$ , where  $S$  is the number of vertices and  $A$  the number of edges. In T3 elements  $S = N_N$  since vertices coincide with nodes. Elaborating on these relationships, the number of elements and hence the number of incompressibility constraints is:

$$N_E = 2N_N^I + N_N^U + N_N^T - 2$$

On the other hand, for a plane strain problem and accounting for the imposed displacements at the  $N_N^U$  nodes of  $S_u$ , the number of unknowns is  $N = 2(N_N^I + N_N^T)$ . The number of independent nodal values finally becomes:

$$N^{\text{incomp}} = N - N_E = N_N^T - N_N^U + 2 \quad (6.1)$$

and might even vanish or become negative. In such cases the problem is said to be *locked* and all displacement nodal values vanish.

Let us consider in greater detail the example of Figure 6.4. If homogeneous boundary conditions are enforced on the left and lower sides,  $N_N^T - N_N^U + 2 = 0$  and this implies that all the nodes are blocked. Indeed, the same conclusion can be reached on the basis of intuitive remarks. It is easy to see that  $u_2^{(1)} = 0$  since the area of element 1 has to remain constant; similarly,  $u_1^{(1)} = 0$  due to the area constraint for element 2. This reasoning can be repeated for nodes 2 and 3, first, and then extended progressively to all the mesh from left to right.

No general and robust solution to the problem is available and T3 elements should definitely be avoided whenever incompressibility is expected.

**Remark.** This conclusion seems at first sight to contradict formula (3.46), which has been presented as a general convergence result. However, a more refined analysis reveals that the constant  $C_1$  in (3.46), while independent of  $\mathbf{u}$  and  $h$ , indeed depends on the elastic parameters of the problem at hand. Letting  $\nu \rightarrow 1/2$  makes this constant diverge (see e.g. Quarteroni and Valli, 1994).

### Exercise: cross diagonal pattern

Volumetric locking with T3 can be mitigated by using a special arrangement of elements, like the well known *cross diagonal* pattern depicted in Figure 6.5, where each square is partitioned into four triangles by inserting a node exactly at the center of the square. Let us focus on the simple case illustrated on the right of Figure 6.5. Formula 6.1 states that no free degrees of freedom remain, i.e. the mesh should be locked. However, the direct analysis of area constraints reveals that node 2 can move along the  $-e_1 + e_2$  dashed direction.

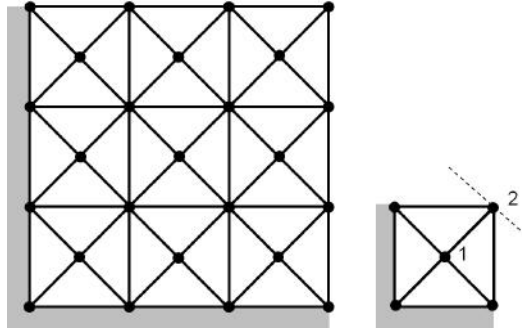


Figure 6.5: Cross-diagonal pattern

In order to better understand this phenomenon, let us analyse the general setting of Figure 6.6, where a quadrangle has been partitioned into triangles by its diagonals. Let  $e_3$  denote the out-of-plane unit vector and  $S_e$  the area of the triangles  $E_e$  ( $1 \leq e \leq 4$ ). We now consider an admissible displacement field  $v_h$  and let  $\dot{S}_e[v_h]$  represent the first order variation of  $S_e$  due to  $v_h$ . Since  $\text{div } v_h$  is constant over  $E_e$ :

$$\dot{S}_e[v_h] = \int_{E_e} \text{div } v_h \, dS = S_e \text{div } v_h \quad (6.2)$$

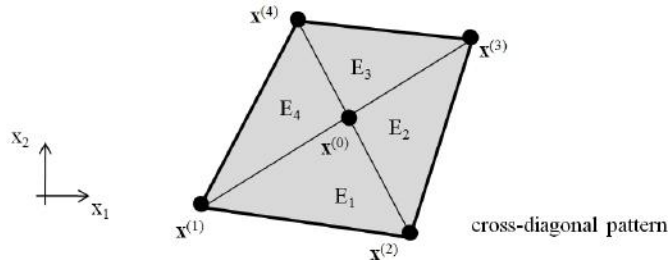


Figure 6.6: Quadrangle partitioned into triangles by its diagonals

Let  $v_e(x)$  be a linear vector field whose restriction to  $E_e$  coincides with  $v_h$  (recall that  $v_h$  is piecewise-linear, while  $v_e$  is globally linear). We now introduce the linear combination:

$$p(x) = v_1(x) - v_2(x) + v_3(x) - v_4(x)$$

If we consider for instance the diagonal from  $x^{(2)}$  to  $x^{(4)}$  we have  $v_1 = v_2$  and  $v_4 = v_3$  due to the continuity of  $v_h$ . Along the second diagonal we have  $v_1 = v_4$  and  $v_2 = v_3$ . As a consequence, the linear vector field  $p$  vanishes along the diagonals and hence vanishes everywhere. Moreover:

$$\operatorname{div} p = \operatorname{div} v_1 - \operatorname{div} v_2 + \operatorname{div} v_3 - \operatorname{div} v_4 = 0$$

Since  $\operatorname{div} v_e$  coincides with  $\operatorname{div} v_h$  over  $E_e$  and  $\dot{S}_e = S_e \operatorname{div} v_h$ , the four area constraints are not linearly independent. The count of  $N^{\text{incomp}}$  in (6.1) is improved since the number of independent incompressibility constraints decreases by a factor 3/4.

### 6.1.2 Incompressibility and Q4 elements

The analysis of a Q4 quadrilateral is more involved but instructive. The components of the generic admissible displacement field  $v_h$

$$v_h(x) = \sum_{k=1}^{n_e} N_k(a) v^{(k)} \quad (6.3)$$

can be expressed in the form:

$$v_h(x) = b_1 + A_1 \cdot x + c_1 h(a) \quad (6.4)$$

where we have introduced the *hourglass* function  $h(a) = a_1 a_2$  and  $b_1, c_1, A_1$  are arbitrary. Indeed, since the Q4 element is isoparametric  $x$  is expressed in terms of the usual shape functions  $N_k$  of Section 2.2, we have, from Table 2.2:

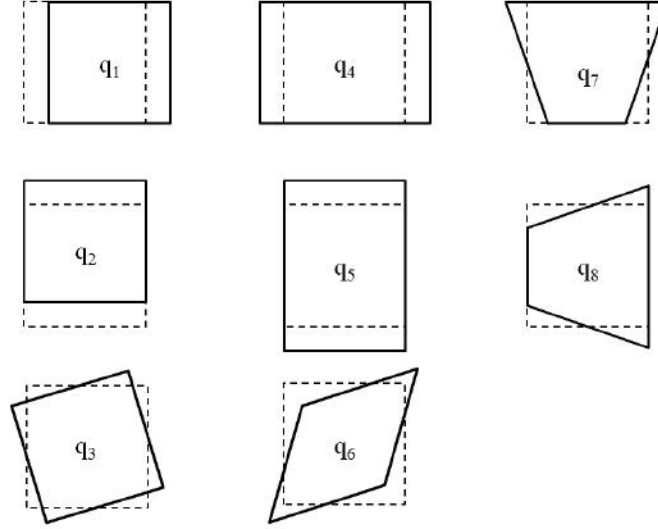
$$x = b_2 + A_2 \cdot a + c_2 h(a) \quad (6.5)$$

with arbitrary  $b_2, c_2, A_2$ . Inverting the linear part of (6.5) in order to express  $a$  in terms of  $x$  and  $h(a)$  and inserting the result in (6.3), equation (6.4) is finally obtained.

Regarding the displacement, an expression which slightly differs from (6.4), called *natural* representation of displacements, is generally preferred:

$$\begin{aligned} v_{h,1} &= (q_1 - q_3 x_2) + q_4 x_1 + q_6 x_2 + q_7 h(a) \\ v_{h,2} &= (q_2 + q_3 x_1) + q_6 x_1 + q_5 x_2 + q_8 h(a) \end{aligned} \quad (6.6)$$

where the  $q_i$  are suitable parameters which depend on nodal coordinates and displacements. In the case of a square element (using  $x_1, x_2$  coordinates with origin at the element centre), a graphical interpretation of (6.6) can be provided as in Figure 6.7, where all the deformation modes are represented. Modes  $q_1, q_2, q_3$  are associated with rigid body motions (translations and rotation) and yield zero strains;  $q_4, q_5, q_6$  represent uniform strain conditions for  $\varepsilon_{11}, \varepsilon_{22}, \varepsilon_{12}$  respectively. Finally,  $q_7$  and  $q_8$  are called *hourglass modes*. The three strain components can be decoupled if and only if  $q_7 = q_8 = 0$ ; in particular, a purely deviatoric field can be



**Figure 6.7:** Graphical interpretation of parameters  $q_i$

represented only if only if  $q_7 = q_8 = 0$ . The hourglass function intrinsically introduces a coupling between volumetric and deviatoric strains, which is responsible of the locking phenomenon analysed next.

Let us now focus on the constraint of incompressibility. The volumetric component of strain is:

$$\operatorname{div} \mathbf{v}_h = q_4 + q_5 + q_7 \frac{\partial h}{\partial x_1} + q_8 \frac{\partial h}{\partial x_2}$$

For a specific element  $E_e$ , an application of the Green divergence theorem to vectors  $h\mathbf{e}_1$  and  $h\mathbf{e}_2$  shows that:

$$\int_{E_e} \frac{\partial h}{\partial x_1} dV = \int_{E_e} \frac{\partial h}{\partial x_2} dV = 0$$

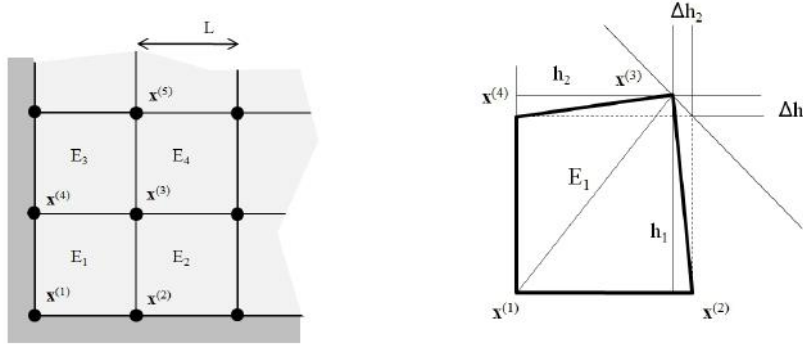
This means that the area of the element is preserved if and only if  $q_4 = -q_5$ , since the hourglass modes do not induce any area change.

It is worth stressing that, since the gradient of  $h$  vanishes at  $\mathbf{a} = \mathbf{0}$ , imposing the total area constraint is equivalent to imposing  $\operatorname{div} \mathbf{v}_h = 0$  at the element centre (see next section on selective integration). However it is clear that (a)  $\operatorname{div} \mathbf{v}_h$  vanishes everywhere in the element only if also  $q_7 = q_8 = 0$ , and (b) simply preserving the element area does not necessarily imply satisfying the incompressibility constraint point-wise.

#### **Exercise: locking for Q4 elements**

The nodes of a mesh like that depicted in Figure 6.8 are fully blocked if the incompressibility constraint is imposed in a *strong* manner.

Let us consider first element  $E_1$ . Only node  $\mathbf{x}^{(3)}$  can move. The global area remains constant if and only if  $v_1^{(3)} = -v_2^{(3)}$ . From Figure 6.8 we see that area  $S_1$  can be computed as the sum of the areas of triangles into which the quadrilateral is partitioned by the diagonal  $\mathbf{x}^{(3)}-\mathbf{x}^{(1)}$ . Since the basis of the triangles remains constant (their nodes are blocked) the variation of their heights  $h_1$  and  $h_2$  must be such that  $\Delta h_1 = -\Delta h_2$ , which gives the desired result, since  $\Delta h_1 = v_2^{(3)}$  and  $\Delta h_2 = v_1^{(3)}$ .



**Figure 6.8:** Mesh of square Q4 elements blocked along the lower and left sides

We remark that enforcing  $\text{div } \mathbf{v} = 0$  at the element center is equivalent to imposing that the area remains constant. On the contrary, even if  $v_1^{(3)} = -v_2^{(3)}$  and the area is preserved, the divergence

$$\text{div } \mathbf{v}_h = \frac{1}{2L} v_1^{(3)} (a_2 - a_1)$$

vanishes only along the line  $a_1 = a_2$  unless  $v_1^{(3)} = 0$ , i.e. if node 3 is blocked. Similar arguments imply that all the mesh nodes are blocked.

### 6.1.3 Introduction to penalty approaches and selective integration

An *engineering* solution to the locking problems, called *selective integration*, is available and widely applied in practice. Though more general, it is presented herein as a specific application to Q4 quadrilaterals and requires first to formulate the incompressible problem in the framework of penalty approaches.

Let us consider the potential energy functional  $\mathcal{P}(\mathbf{v}; E, \nu)$  for an isotropic material ( $\mathcal{P}$  is a function of  $\mathbf{v}$  with  $E, \nu$  as parameters), where  $\mathbf{v}$  is a generic kinematically admissible field (see (1.29)):

$$\mathcal{P}(\mathbf{v}; E, \nu) = \mathcal{W}(\mathbf{v}; E, \nu) - \mathcal{F}(\mathbf{v}).$$

$\mathcal{F}$  is the potential of external loads while  $\mathcal{W}$  represent the strain energy stored in  $\Omega$ :

$$\mathcal{W}(\mathbf{v}; E, \nu) = \frac{1}{2} \int_{\Omega} (\lambda (\text{Tr } \boldsymbol{\varepsilon})^2 + 2\mu \boldsymbol{\varepsilon} : \boldsymbol{\varepsilon}) \, dV = \mathcal{W}^{\mu}(\mathbf{v}; \mu) + \lambda \mathcal{W}^{\lambda}(\mathbf{v}), \quad (6.7)$$

having set

$$\mathcal{W}^\mu(\mathbf{v}; \mu) = \mu \int_{\Omega} \boldsymbol{\varepsilon} : \boldsymbol{\varepsilon} \, dV, \quad \mathcal{W}^\lambda(\mathbf{v}) = \frac{1}{2} \int_{\Omega} (\text{Tr } \boldsymbol{\varepsilon})^2 \, dV$$

It is worth stressing that an alternative (and possibly more elegant) procedure would fully decouple deviatoric and volumetric strain components, but is not adopted herein for the sake of consistency with subsequent developments (see the Herrmann functional, Sec. 6.2.1). If  $\nu \rightarrow 1/2$  the  $\lambda \mathcal{W}^\lambda$  term vanishes even though  $\lambda \rightarrow \infty$ . Indeed,  $\lambda(\text{Tr } \boldsymbol{\varepsilon})$  is bounded by virtue of the assumption of finite stresses, while  $\text{Tr } (\boldsymbol{\varepsilon}) \rightarrow 0$ . Since  $\mu \rightarrow E/3$  for  $\nu \rightarrow 1/2$ , one can formulate a fully incompressible problem as:

$$\min_{\mathbf{v}} \mathcal{W}^\mu(\mathbf{v}; E/3) - \mathcal{F}(\mathbf{v}), \quad \text{under the constraint } \text{div } \mathbf{v} = 0 \text{ in } \Omega. \quad (6.8)$$

However the space of admissible  $\mathbf{v}$  has to be limited *a priori* to the space of kinematically admissible displacements such that  $\text{div } \mathbf{v} = 0$ .

For this reason, an alternative technique is often preferred which consists in introducing a slight *fictitious* compressibility setting  $\nu$  very close, but not exactly equal, to  $1/2$ . If one neglects the variation of  $\mathcal{W}^\mu$  due to the fictitious compressibility, the problem becomes

$$\min_{\mathbf{v}} \left( \mathcal{W}^\mu(\mathbf{v}; E/3) - \mathcal{F}(\mathbf{v}) + \lambda \mathcal{W}^\lambda(\mathbf{v}) \right) \quad \text{unconstrained} \quad (6.9)$$

and this can be interpreted as a penalty approach for the approximate solution of Problem (6.8). We recall here for convenience that the minimisation of a functional  $\mathcal{P}(\mathbf{v})$  subjected to constraints:

$$\min_{\mathbf{v}} \mathcal{P}(\mathbf{v}), \quad \text{with the constraints } c_i(\mathbf{v}) = 0, \quad i = 1, \dots, N \quad (6.10)$$

can be performed with an approximate *penalty* technique, where the augmented functional

$$\mathcal{P}(\mathbf{v}) + \lambda \sum_{i=1}^N (c_i(\mathbf{v}))^2 \quad (6.11)$$

is minimized, with  $\lambda$  a large positive penalty coefficient; constraints are not explicitly accounted for. The two problems (6.10) and (6.11) are rigorously equivalent only in the limit  $\lambda \rightarrow \infty$ . However, in any numerical approach  $\lambda$  will always be bounded, making the penalty technique approximate.

Going back to the problem at hand, a solution to (or mitigation of) the locking problem which has encountered great favor in practice is called *selective integration* and is justified here in an intuitive manner. Let us now reconsider problem (6.9) and introduce a standard discretization using Q4 isoparametric elements. For a square Q4 element, a numerical integration rule with  $2 \times 2$  Gauss points allows to exactly integrate the two terms  $\mathcal{W}^\mu(\mathbf{v}_h; E/3)$ ,  $\mathcal{W}^\lambda(\mathbf{v}_h)$ . Indeed, the Jacobian of the geometrical transformation is a constant and the strain components are linear in  $a_1, a_2$  while a  $2 \times 2$  integration rule allows to integrate exactly a polynomial of order 3 with respect to each single coordinate  $a_1$  and  $a_2$ .

Let us now imagine that, in the process of evaluating the element stiffness matrix, the term  $\mathcal{W}^\mu(\mathbf{v}_h; E/3)$  is integrated correctly, while a 1-point Gauss rule is applied to the term  $\mathcal{W}^\lambda(\mathbf{v}_h)$ . This procedure is an example of *selective integration* and yields, for a given element:

$$\int_{E_e} (\text{Tr } \varepsilon_h(\mathbf{a}))^2 dV \simeq S_e (\text{Tr } \varepsilon_h(\mathbf{0}))^2$$

where  $S_e$  denotes the element area and  $\varepsilon_h = \varepsilon[\mathbf{v}_h]$ . As a consequence

$$\mathcal{W}^\lambda(\mathbf{v}_h; \lambda) \simeq \frac{\lambda}{2} \sum_{e=1}^{N_E} S_e (\text{Tr } \varepsilon_h(\mathbf{0}))^2$$

In this case problem (6.9) becomes:

$$\min_{\mathbf{v}_h} \left( \mathcal{W}^\mu(\mathbf{v}_h; E/3) - \mathcal{F}(\mathbf{v}_h) + \frac{\lambda}{2} \sum_{e=1}^{N_E} S_e (\text{Tr } \varepsilon_h(\mathbf{0}))^2 \right) \quad (6.12)$$

where the sum is performed over the elements and  $\varepsilon_h(\mathbf{0})$  is the strain tensor  $\varepsilon_h$  evaluated in  $a_1 = a_2 = 0$  for every element.

Problem (6.12) can be indeed interpreted as a penalty technique for the approximate solution of problem (6.8) in which the constraint  $\text{div } \mathbf{v}_h = 0$ ,  $\forall \mathbf{x}$  has been replaced by the  $N_E$  constraints  $\text{Tr } \varepsilon_h(\mathbf{0}) = 0$ , (one constraint per element) which express the requirement that the element areas do not change. As discussed in Section 6.1.2, this is a much weaker constraint than enforcing exact incompressibility everywhere in the elements. As a consequence some nodal degrees of freedom are released and the mesh does not *lock*.

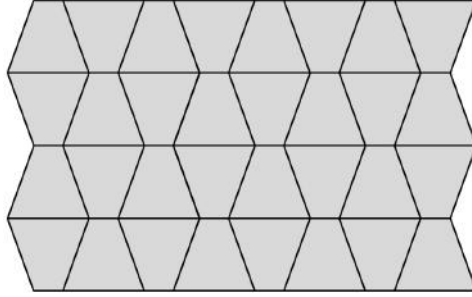
It is now clear that if, on the contrary, a 4-point Gauss rule is employed even for the volumetric term, then this corresponds to introducing the  $4N_E$  constraints that, for every element and every Gauss point  $\text{Tr } \varepsilon_h(\mathbf{a}_g) = 0$ . The number of constraints imposed thus increases four-fold relative to the selective integration case, causing the problem to be severely locked.

This approach of selective integration has strong connections with mixed approaches which will be discussed in the sequel.

**Remark on reduced integration schemes.** The technique described in the previous paragraph is called selective since it integrates different terms of the strain energy with different integration rules, and specifically under-integrates the locking term. By contrast, *reduced integration* generally means that all the terms of the strain energy are under-integrated. This technique is sometimes applied in explicit approaches in dynamics (see Chapter 5) since it is extremely fast; however, it might induce degeneracies associated with the fact that the numerically integrated element stiffness matrix  $[K_e]$  does not have the appropriate rank.

Indeed, employing a 1-point Gauss integration rule (whose Gauss point is  $\mathbf{a} = \mathbf{0}$ ), since the gradient of the hourglass function vanishes at  $\mathbf{a} = \mathbf{0}$ , the  $q_7$





**Figure 6.9:** Hourglass spurious mode propagates in the mesh

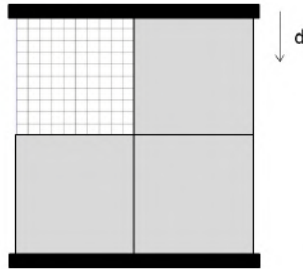
and  $q_8$  deformation modes will contribute neither to the strains nor to the strain energy.

This means that instabilities like the one depicted in Figure 6.9, where all the elements are deforming according to a hourglass mode, can freely propagate in the mesh if not prevented by suitable boundary conditions or specific arrangements of the mesh layout.

**Exercise 6.1.** Show that the correct rank of a fully integrated element stiffness matrix  $[K_e]$  for a Q4 is 5. Explain why, in the case of under-integration with one Gauss point, this rank reduces to 3.

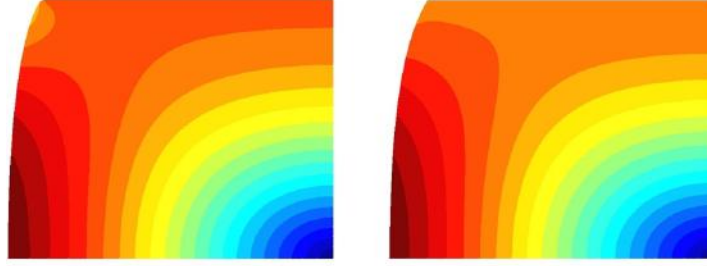
#### 6.1.4 Exercise: selective integration

Let us consider the square specimen of figure 6.10 in plane-strain conditions. The lower surface is blocked, while the upper one undergoes a vertical displacement  $-de_y$ . Friction prevents sliding between the specimen and the rigid plates. By virtue of symmetry, only one-fourth of the structure is meshed, using  $N \times N$  square Q4 elements.



**Figure 6.10:** Compression of a specimen in plane-strain

The aim of the exercise is to provide an estimate of the force exerted by the upper plate on the specimen and, in particular, verify the convergence of the force with respect to  $N$  in an almost incompressible condition ( $\nu = 0.4999$ ). If standard fully integrated elements are employed, the convergence is painfully slow, as shown in



**Figure 6.11:** Compression of a specimen in plane-strain: deformed specimen using selective (left) and full (right) integration

Table 6.1. On the contrary, selective integration dramatically improves that behaviour. The locking issue is mitigated, as can be appreciated from the deformation pattern of the upper left corner for the two deformed configurations in Figure 6.11 for selective (left) and full (right) integration.

Number of elements	$2 \times 2$	$4 \times 4$	$8 \times 8$	$16 \times 16$	$32 \times 32$
Full integration	10.5944	2.7933	0.8428	0.3539	0.2289
Selective integration	0.1864	0.1801	0.1772	0.1759	0.1754

**Table 6.1:** Convergence of applied force with mesh refinement

From a practical point of view, the implementation of selective integration is straightforward. Indeed, recalling the definition of the element stiffness matrix (3.15) and the decomposition (1.11) of the stiffness tensor:

$$\{W_e\}^T [K_e] \{U_e\} = \int_{E_e} \varepsilon[\mathbf{u}_h] : \mathcal{A} : \varepsilon[\mathbf{w}] dV = \int_{E_e} \varepsilon[\mathbf{u}_h] : (3\lambda \mathcal{J} + 2\mu \mathcal{I}) : \varepsilon[\mathbf{w}] dV$$

with

$$[K_e] = \int_{\Delta_e} [B(\mathbf{a})]^T ([A_J] + [A_I]) [B(\mathbf{a})] J(\mathbf{a}) dV(\mathbf{a}) \quad (6.13)$$

The part associated with the  $\mu \varepsilon : \varepsilon$  term is fully integrated employing a complete Gauss rule, while the contribution of the terms arising from  $\lambda(\text{Tr } \varepsilon)^2$  is evaluated with a 1-point Gauss rule:

$$[K_e] \approx \sum_{g=1}^G w_g [B(\mathbf{a}_g)]^T [A_I] [B(\mathbf{a}_g)] J(\mathbf{a}_g) + [B(\mathbf{0})]^T [A_J] [B(\mathbf{0})] 4J(\mathbf{0}). \quad (6.14)$$

This translates into the following MATLAB code (contained in the file Q4\_2A\_solid.KeS) which starts considering the contribution from  $[A_I]$ :

```
mu=E/(2*(1+nu));
AI=[2*mu,0,0;
    0,2*mu,0;
    0,0,mu];
a_gauss=1/sqrt(3)*[-1 1];
w_gauss=[1 1];
```

% Gauss abscissae  
% Gauss weights

```

for g1=1:2, % loop over Gauss points
    a1=a_gauss(g1); % param. coord. of gauss point
    for g2=1:2, % loop over Gauss points
        a2=a_gauss(g2); % param. coord. for gauss point

        ... same as for a standard stiffness matrix ...

        Ke=Ke+B'*AI*B*detJ*w_gauss(g1)*w_gauss(g2); % contrib. to stiffness matrix
    end
end

```

and then adds the terms associated with  $[A_J]$ :

```

lambda=2*mu*nu/(1-2*nu);
AJ=lambda*[1,1,0;
            1,1,0;
            0,0,0];
D=1/4*[-1 1 1 -1; % D evaluated at a_1=a_2=0
        -1 -1 1 1]';
J=X'*D; % jacobian matrix
detJ=J(1,1)*J(2,2)-J(1,2)*J(2,1); % jacobian
invJ=1/detJ*[ J(2,2) -J(1,2); % inverse jacobian matrix
              -J(2,1) J(1,1)];
G=D*invJ; % gradient of shape functions
B=[G(1,1) 0 G(2,1) 0 G(3,1) 0 G(4,1) 0 ;
    0 G(1,2) 0 G(2,2) 0 G(3,2) 0 G(4,2) ;
    G(1,2) G(1,1) G(2,2) G(2,1)...
    G(3,2) G(3,1) G(4,2) G(4,1)];
Ke=Ke+B'*AJ*B*detJ*4; % contrib. to stiffness matrix

```

The force exerted by the upper plate on the specimen can be computed as suggested in Section 3.7.2 by means of the Principle of Virtual Work.

## 6.2 Mixed functionals

While analyzing Q4 elements it has been argued that inconsistencies associated with the incompressibility constraint are due to the fact that the interpolation of displacements induces an intrinsic coupling between deformation modes which can prove harmful when these contribute in a significantly different manner to the strain energy. Indeed, in the finite element method described so far, displacements are the primary unknowns and the virtual work of internal stresses in the weak equilibrium equation (principle of virtual work) is actually expressed in terms of strains and stresses

$$\varepsilon[\mathbf{u}](\mathbf{x}) = \frac{1}{2}(\nabla \mathbf{u}(\mathbf{x}) + \nabla^T \mathbf{u}(\mathbf{x})), \quad (6.15a)$$

$$\boldsymbol{\sigma}[\mathbf{u}](\mathbf{x}) = \mathcal{A}:\varepsilon[\mathbf{u}](\mathbf{x}) \quad (6.15b)$$

where compatibility and constitutive equations are imposed in a *strong* manner. It is thus natural to investigate different options allowing for a decoupling between the interpolations of displacements, strains and stresses. These alternative approaches are typically based on *mixed* functionals, which extend the notions of

total potential energy and total complementary potential energy. A complete and detailed treatment of mixed formulations is out of the scope of this introductory book, and only some basic ideas will be presented, often without adequate mathematical rigour. Moreover, we will not address the general cases, based for instance on the Hu-Washizu or Hellinger-Reissner functionals, but will confine ourselves to the specific case of incompressibility.

### 6.2.1 Herrmann functional

A specific cure to the incompressibility issue can be obtained from the so called Herrmann functional.

In the classical representation of the stress tensor in terms of Lamé constants  $\boldsymbol{\sigma} = 2\mu\boldsymbol{\varepsilon}[\mathbf{u}] + \lambda(\text{Tr}\boldsymbol{\varepsilon})\mathbf{I}$ ,  $\lambda \rightarrow \infty$  and  $\text{Tr}\boldsymbol{\varepsilon} \rightarrow 0$  for  $\nu \rightarrow 1/2$ . However, their product  $\lambda(\text{Tr}\boldsymbol{\varepsilon})$  remains well defined if global stresses are bounded. Hence, the idea is to represent the stress tensor as  $\boldsymbol{\sigma} = 2\mu\boldsymbol{\varepsilon}[\mathbf{u}] + p\mathbf{I}$  where  $p = \lambda\text{Tr}\boldsymbol{\varepsilon}$  is treated as an independent field. The function  $p$ , improperly termed pressure, corresponds to the volumetric component  $(1/3)\text{Tr}(\boldsymbol{\sigma})$  of the stress tensor only when  $\nu = 1/2$ , i.e. in the case of exact incompressibility. The Herrmann functional  $\mathcal{H}$  is defined for admissible displacements  $\mathbf{v} \in \mathcal{C}(\mathbf{u}^{\text{D}})$  and admissible stresses  $\boldsymbol{\tau}$  which are assumed to have the form  $\boldsymbol{\tau} = 2\mu\boldsymbol{\varepsilon}[\mathbf{v}] + g\mathbf{I}$ , where  $g \in \mathcal{Q}$  plays the role of an admissible pressure which we will here only require to be square-integrable (i.e.  $\mathcal{Q} = L^2(\Omega)$ ). It is given by:

$$\mathcal{H}[\mathbf{v}, g] = \int_{\Omega} \left\{ \mu\boldsymbol{\varepsilon}[\mathbf{v}] : \boldsymbol{\varepsilon}[\mathbf{v}] + g \text{div } \mathbf{v} - \frac{1}{2\lambda} g^2 - \rho \mathbf{f} \cdot \mathbf{v} \right\} dV - \int_{S_T} \mathbf{T}^{\text{D}} \cdot \mathbf{v} dS \quad (6.16)$$

and remains well defined even if  $\nu = 1/2$ , while this would not be the case for the total potential energy functional. The stationarity condition with respect to  $\mathbf{v}$  and  $g$  is:

$$\begin{aligned} 0 = \int_{\Omega} \left\{ 2\mu\boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\varepsilon}[\mathbf{w}] + p \text{div } \mathbf{w} \right\} dV - \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} dV - \int_{S_T} \mathbf{T}^{\text{D}} \cdot \mathbf{w} dS \\ + \left( \text{div } \mathbf{u} - \frac{p}{\lambda} \right) q dV \quad \forall \mathbf{w} \in \mathcal{C}(0), \forall q \in \mathcal{Q} \end{aligned}$$

Since  $\mathbf{w}$  and  $q$  are independent test fields, the above condition is equivalent to

find  $\mathbf{u} \in \mathcal{C}(\mathbf{u}^{\text{D}})$  and  $p \in \mathcal{Q}$  such that

$$\begin{aligned} \int_{\Omega} \left\{ 2\mu\boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\varepsilon}[\mathbf{w}] + p \text{div } \mathbf{w} \right\} dV \\ = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} dV + \int_{S_T} \mathbf{T}^{\text{D}} \cdot \mathbf{w} dS \quad \forall \mathbf{w} \in \mathcal{C}(0) \end{aligned} \quad (6.17a)$$

$$\int_{\Omega} \left( \text{div } \mathbf{u} - \frac{p}{\lambda} \right) q dV = 0 \quad \forall q \in \mathcal{Q} \quad (6.17b)$$

Considering that:

$$2\mu\boldsymbol{\varepsilon}[\mathbf{u}]:\boldsymbol{\varepsilon}[\mathbf{w}] + p \operatorname{div} \mathbf{w} = (2\mu\boldsymbol{\varepsilon}[\mathbf{u}] + p\mathbf{I}):\boldsymbol{\varepsilon}[\mathbf{w}] = \boldsymbol{\sigma}:\boldsymbol{\varepsilon}[\mathbf{w}]$$

equations (6.17a) and (6.17b) correspond to the weak enforcement of equilibrium and incompressibility ( $p/\lambda = \operatorname{Tr}\boldsymbol{\varepsilon}[\mathbf{u}] = \operatorname{div} \mathbf{u}$ ), respectively.

### 6.2.2 Mixed finite element models: pressure-displacement elements

Let us consider the variational approach associated with the Herrmann functional and assume that a standard mesh has been created for the domain  $\Omega$ . The geometry and the displacement field in each element are represented as in classical isoparametric elements, for simplicity. In addition, in order to obtain a fully discretized version of eqs.(6.17a)-(6.17b), an approximation space  $\mathcal{Q}_h$  for the pressure field has to be selected. Since pressure has only to be square-integrable, with continuity between elements not required (although clearly admissible), a wealth of options are in principle available. Since no boundary conditions are imposed on the pressure field, the two approximation spaces (for the unknown and test pressures) will coincide.

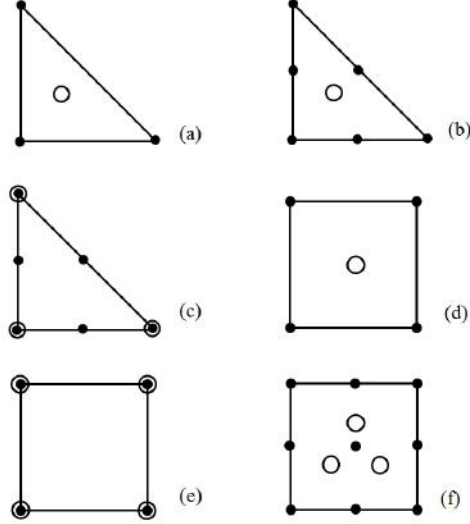
We will generically refer to *pressure nodes* as geometrical entities, possibly independent of the mesh itself, where pressure nodal values are defined, similarly to what is done for displacements. The local interpolation in an element  $E_e$  endowed with  $n_p$  pressure nodes is given as:

$$p_h(\mathbf{x}) = \sum_{k=1}^{n_p} N_k^P(\mathbf{a}) p^{(k)} \quad q(\mathbf{x}) = \sum_{k=1}^{n_p} N_k^P(\mathbf{a}) q^{(k)} \quad (6.18)$$

where  $N_k^P(\mathbf{a})$  are pressure shape functions.

Several examples are presented in Figure 6.12, which is by no means exhaustive. A simple choice is to enrich the linear triangle (T3) with a constant pressure (T3P1 element, case a); here  $N_p = 1$ , the pressure node is arbitrarily placed at the centre of the element and the associated shape function is constant. In cases (b) and (c), respectively, the T6 quadratic triangle is associated with a constant (T6P1) and a linear (T6P3C) pressure. Typically, if pressure nodes are placed within the element, the interpolation will be discontinuous across elements (this is the case for T3P1 and T6P1 elements); if on the contrary the pressure nodes coincide with some mesh nodes and are shared by neighbouring elements (as in T6P3C, case c), the pressure field will be continuous (or at least *can* be made continuous).

Similar alternatives are also available for quadrilaterals. In the case of the classical bilinear Q4, typical choices for the pressure field are: constant discontinuous (case d, Q4P1 element), linear discontinuous (Q4P3 element, not shown) and bilinear continuous (case e, Q4P4C element). Furthermore, higher-order elements can be designed, like the Q9P3 element of case (f) which actually proves to be quite effective. However, as it will be discussed in the sequel, not every combination of interpolation for displacements and pressure leads to stable and convergent formulations and, even worse, it is not always an easy matter to get intuitive guidelines.



**Figure 6.12:** Examples of continuous (c,e) or discontinuous (a,b,d,f) pressure-displacement elements. Black dots denote displacement nodes, and circles denote pressure nodes

In any case the global interpolation of the pressure  $p_h$  and of the test field  $q$  can formally be represented as:

$$p_h(\mathbf{x}) = \sum_{n=1}^{N_P} \tilde{N}_n^P(\mathbf{x}) p^{(n)} \quad q(\mathbf{x}) = \sum_{n=1}^{N_P} \tilde{N}_n^P(\mathbf{x}) q^{(n)} \quad (6.19)$$

where  $N_P$  represents the global number of pressure nodes and  $\tilde{N}_n^P(\mathbf{x})$  are global shape functions.

Inserting the adopted interpolations for the displacement and pressure fields in the variational equations eqs.(6.17a)-(6.17b), the following final discretized form is obtained:

find  $\mathbf{u}_h \in \mathcal{C}_h(\mathbf{u}^D)$  and  $p_h \in \mathcal{Q}_h$  such that

$$\begin{aligned} \int_{\Omega} \left\{ 2\mu \varepsilon[\mathbf{u}_h^{(0)}] : \varepsilon[\mathbf{w}] + p_h \operatorname{div} \mathbf{w} \right\} dV &= - \int_{\Omega} 2\mu \varepsilon[\mathbf{u}_h^{(D)}] : \varepsilon[\mathbf{w}] dV \\ &+ \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} dS \quad \forall \mathbf{w} \in \mathcal{C}_h(\mathbf{0}) \end{aligned} \quad (6.20a)$$

$$\int_{\Omega} \operatorname{div} \mathbf{u}_h^{(0)} q dV - \int_{\Omega} \frac{1}{\lambda} p_h q dV = - \int_{\Omega} \operatorname{div} \mathbf{u}_h^{(D)} q dV \quad \forall q \in \mathcal{Q}_h \quad (6.20b)$$

Like in the classical displacement-based FEM, global matrices and vectors are computed by an assemblage procedure which gathers the contribution of each

element obtained by means of element routines. If  $\{\mathbb{P}\}$  and  $\{\mathbb{Q}\}$  collect all the nodal values of the  $p_h$  and  $q_h$  pressure fields and  $\mathbb{U}$  and  $\mathbb{W}$  contain the free nodal values of the displacement fields, one has

$$\begin{aligned} \int_{\Omega} 2\mu \varepsilon[\mathbf{u}_h^{(0)}] : \varepsilon[\mathbf{w}] \, dV &= \{\mathbb{W}\}^T [\mathbb{K}] \{\mathbb{U}\} \\ \int_{\Omega} p_h (\operatorname{div} \mathbf{w}) \, dV &= \{\mathbb{W}\}^T [\mathbb{G}] \{\mathbb{P}\} \\ \int_{\Omega} \left\{ -2\mu \varepsilon[\mathbf{u}_h^{(D)}] : \varepsilon[\mathbf{w}] + \rho \mathbf{f} \cdot \mathbf{w} \right\} dV + \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} \, dS &= \{\mathbb{W}\}^T \{\mathbb{F}\} \end{aligned}$$

and

$$\begin{aligned} \int_{\Omega} \operatorname{div} \mathbf{u}_h^{(0)} q \, dV &= \{\mathbb{Q}\}^T [\mathbb{G}]^T \{\mathbb{U}\} \\ \int_{\Omega} \frac{1}{\lambda} p_h q \, dV &= \{\mathbb{Q}\}^T [\mathbb{H}] \{\mathbb{P}\} \\ - \int_{\Omega} \operatorname{div} \mathbf{u}_h^{(D)} q \, dV &= \{\mathbb{Q}\}^T \{\mathbb{F}^P\} \end{aligned}$$

Finally, the discretized variational form becomes:

Find  $\mathbb{U}$  and  $\mathbb{P}$  such that:

$$\begin{Bmatrix} \mathbb{W} \\ \mathbb{Q} \end{Bmatrix}^T \begin{bmatrix} \mathbb{K} & \mathbb{G} \\ \mathbb{G}^T & -\mathbb{H} \end{bmatrix} \begin{Bmatrix} \mathbb{U} \\ \mathbb{P} \end{Bmatrix} = \begin{Bmatrix} \mathbb{W} \\ \mathbb{Q} \end{Bmatrix}^T \begin{Bmatrix} \mathbb{F} \\ \mathbb{F}^P \end{Bmatrix}, \quad \forall \mathbb{W}, \forall \mathbb{Q} \quad (6.21)$$

which is equivalent to the linear system:

$$\begin{bmatrix} \mathbb{K} & \mathbb{G} \\ \mathbb{G}^T & -\mathbb{H} \end{bmatrix} \begin{Bmatrix} \mathbb{U} \\ \mathbb{P} \end{Bmatrix} = \begin{Bmatrix} \mathbb{F} \\ \mathbb{F}^P \end{Bmatrix}, \quad (6.22)$$

If the problem is *exactly* incompressible,  $\lambda \rightarrow \infty$  and  $\mathbb{H} = 0$ . It is worth stressing that the system matrix is, in that case, symmetric but no longer positive definite. The  $LDL^T$  decomposition hence fails and different solvers, based e.g. on Gaussian  $LU$  decomposition, should be applied.

One benefit of a discontinuous pressure interpolation is that, for each element, pressure nodal values can be expressed in terms of the displacement nodal values of the same element with simple local operations and this greatly reduces the burden of the system solution. For this reason discontinuous pressure elements tend to be somehow in practice.

**Remarks.** We strongly emphasize that mixed methods in general should not be expected to improve the performance of the classical displacement FEM in the absence of pathologies like incompressibility and shear locking (Belytschko et al.,

2002). Also, in many cases, mixed formulations give results which coincide with the classical FEM (see the Principle of Limitation discussed in the sequel) and strong links can be proven with the selective integration schemes which have been heuristically discussed for the Q4 element. Finally, mixed methods may display a number of negative byproducts (like e.g. spurious pressure modes) which require a much deeper understanding of the underlying element technology. Some of these issues will be highlighted by means of specific examples.

### 6.2.3 Convergence and constraint count

The mathematical theory of convergence of mixed approaches, and in particular for the pressure-displacement formulation, is provided by the Babuska-Brezzi, or LBB, stability condition (see Brezzi and Fortin, 1991; Girault and Raviart, 1986; Quarteroni and Valli, 1994)). The LBB stability condition is met if

$\forall q_h \in \mathcal{Q}_h$  there exists a  $\mathbf{v}_h \in \mathcal{C}_h(\mathbf{u}^D)$  and a  $\beta > 0$  such that

$$\int_{\Omega} q_h (\operatorname{div} \mathbf{v}_h) \, dV \geq \beta \|\mathbf{v}_h\|_{H^1(\Omega)} \|q_h\|_{H^0(\Omega)}$$

If an element satisfies the LBB condition, then the solution exists unique and the following error estimates holds (Brezzi and Fortin, 1991; Hughes, 1987):

$$\|\mathbf{u} - \mathbf{u}_h\|_{H^1(\Omega)} + \|p - p_h\|_{H^0(\Omega)} = O(h^\alpha)$$

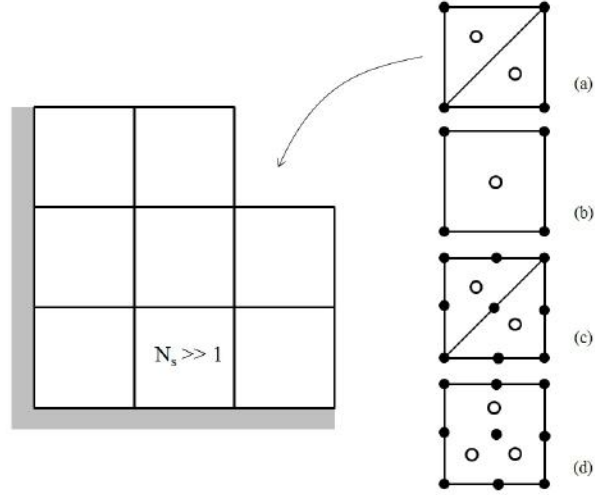
with  $\alpha = \min(k, \ell + 1)$  and convergence is guaranteed as  $h \rightarrow 0$ . Integers  $k$  and  $\ell$  are the degree of the maximum order polynomials representable by the interpolation spaces  $\mathcal{C}_h$  and  $\mathcal{Q}_h$ , respectively. However, the derivation and analysis of the LBB condition is beyond the scope of this course. Even just assessing whether a given element satisfies or not the LBB condition is by no means a trivial task. Heuristic guidelines, like the so called *constraint count* described in the sequel, are often helpful.

Let us consider the case of 2D elasticity. We first remark is that the continuum formulation presents two scalar equilibrium equations with two components of unknown displacements and one scalar constraint equation, with a ratio  $r := \text{unknowns}/\text{constraints}$  equal to 2. The idea is that the ratio  $r_h$  between the numbers of unknown nodal values and constraints in the discrete model should mimic  $r$  as close as possible.

To obtain an estimate of  $r_h$ , we focus on a regular mesh like the one depicted in Figure 6.13, made of identical squares ( $N_s$  elements per side, with  $N_s \gg 1$ ), each of them either filled with one quadrilateral element or two triangular elements. The left and lower sides are blocked. When the upper right corner is filled with the same type of elements used for the rest of the mesh, the number of unknown nodal displacements inserted (i.e. twice the number of new nodes) divided by the number of constraints added provides an estimate for  $r_h$ .

In the simple case of T3P1 elements (case a, linear displacement, constant pressure), only one node is added to the mesh (two scalar unknowns) and two





**Figure 6.13:** Examples of constraint count

elements, which corresponds to two additional constraints (one for each of the two pressure functions). This yields a discrete ratio  $r_h = 1$ , which denotes an overconstrained problem, as expected (see the example at the end of the section). If on the contrary the Q4P1 element (case b) is employed, the count gives  $r_h = 2$  as in the continuous case and should be optimal. However, this element suffers from spurious pressure modes. Both these elements violate the LBB condition, even if the Q4P1 element is often employed in analyses adopting specific *engineering* tricks like pressure smoothing (Hughes, 1987). Another example is provided by the T6P1 element, for which  $r_h = 4$ . This element satisfies the LBB condition, but the high  $r_h$  value indicates that the incompressibility condition is poorly enforced in a global analysis. Finally, the Q9P3 elements (case d) gives  $r_h = 8/3$ . This element, which also satisfies the LBB condition, is generally considered to be a very robust choice for incompressible analyses.

#### 6.2.4 T3P1 element and the Principle of Limitation

We now consider in greater detail the T3P1 element, with the aim of assessing its performance in the case of the analysis of Figure 6.4 which has already been considered in the context of the classical FEM. We focus on the variational equation (6.20b) in the case of exact incompressibility:

$$\int_{\Omega} (\operatorname{div} \mathbf{u}_h) q \, dV = 0 \quad \forall q \in \mathcal{Q}_h$$

Since the pressure test field is piecewise constant (constant over each element), this is equivalent to enforcing:

$$\int_{E_e} \operatorname{div} \mathbf{u}_h \, dV = 0 \quad \forall e$$

i.e. the area of each element must remain constant. But this is the very same condition encountered in Section 6.1.1 where it was shown to generate locking. In this case, the mixed approach therefore gives exactly the same results as the classical one.

This remark, which has been presented with reference to a specific example, has however a general validity for any mixed approach. Let us consider a given isoparametric element and let us call  $\mathcal{Q}_v$  the space of pressures  $\lambda \text{Tr} \varepsilon[v_h]$ ,  $v_h \in \mathcal{C}_h(\mathbf{u}^D)$  computed from the displacements via the compatibility and constitutive equations. Now, let us create a mixed element starting from an isoparametric element enriched with a pressure interpolation space  $\mathcal{Q}_h$ . Then, the *Principle of Limitation* (Fraeijs de Veubeke, 1965) states that, if  $\mathcal{Q}_h \supseteq \mathcal{Q}_v$ , the mixed approach is equivalent to the displacement FEM in the sense that the same values of nodal displacements will be obtained with both formulations and there is no advantage in using a mixed formulation.

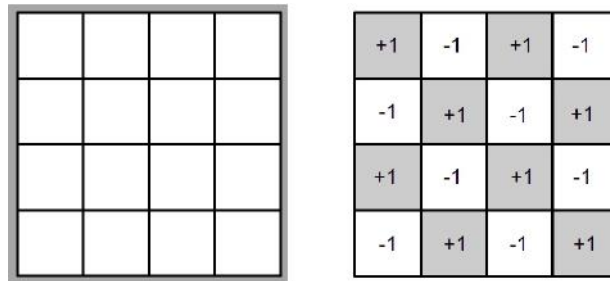
**Exercise 6.2.** Discuss the equivalence between T3 and T3P1 in the case of the mesh in Figure 6.5.

### 6.2.5 Q4P1 element and Pressure Modes

Let us consider the square specimen of Figure 6.14, made of perfectly incompressible material and blocked everywhere on the boundary (homogeneous displacement boundary conditions). Independently of the loading applied within the specimen, if the mesh adopted is made of  $N_s \times N_s$  regular Q4P1 square elements, with  $N_s$  even, it is left as an exercise to verify that the *checkerboard* pressure mode (illustrated on the right of Figure 6.14) is associated with a zero eigenvalue and will pollute the results of any analysis or even induce the failure of the linear system solver. More generally, a *pressure mode* for a perfectly incompressible material is any function  $p_h \in \mathcal{Q}_h$  such that:

$$\int_{\Omega} p_h (\text{div } \mathbf{w}) \, dV = 0 \quad \forall \mathbf{w} \in \mathcal{C}_h(0).$$

In the discrete system 6.22 with  $[\mathbb{H}] = 0$ , a pressure mode is such that  $\{\mathbb{U}\} = \{0\}$  and  $[\mathbb{G}]\{\mathbb{P}\} = \{0\}$ . Actually this pathology is common to all mixed elements



**Figure 6.14:** Example of spurious pressure mode: checkerboard mode

in the sense that, if an excessive number of displacement boundary conditions is imposed and the number of rows in  $[\mathbb{G}]$  is smaller than the number of columns, pressure modes will unavoidably show up.

### 6.2.6 Equivalence between selective integration and mixed approaches

A general theorem, established by Malkus and Hughes (1978), guarantees the equivalence between several mixed elements and specific (complete, selective or reduced) integration rules in the sense that they generate the same stiffness matrix and yield the same nodal values for the displacement field. The equivalence requires that the problem be nearly, but not exactly, incompressible, in the sense discussed in Section 6.1.2; indeed, selective integration can only be applied in the presence of a fictitious compressibility.

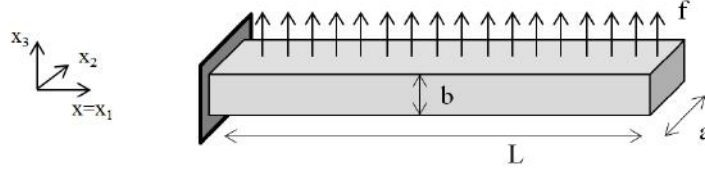
The easiest example is represented by the T3P1 mixed element which turns out to be fully equivalent to the T3 element with one integration point (full integration). This naturally follows from the remarks made in Section 6.2.4.

Similarly, the Q4P1 element corresponds to the selective integration discussed in Section 6.1.2; the Q8P4 corresponds to the Q8 element with full  $3 \times 3$  integration points for the deviatoric part and a reduced  $2 \times 2$  for the volumetric part. An exhaustive list of the equivalence conditions can be found in Hughes (1987).

## 6.3 Bending of slender structures

As illustrated in Figure 6.3, the bending of slender structure is an other typical example where standard formulations fail. While that example employs usual finite elements, one often prefers to use alternative theories to that of three-dimensional continuum mechanics when one or two linear dimensions of a structure are small compared to the largest one. For example, the bending of slender beams is classically associated with the Euler-Bernoulli beam theory for which transverse shear strains (but not the shear forces) are assumed to vanish. Its use is usually limited to cases where the axial dimension is at least 10 times larger than cross-section dimensions. The Timoshenko beam theory is another classical beam theory, in which a transverse shear strain is added to others strains (axial, torsion, bending); it is thus applicable to moderately slender (e.g. short) beams. Unfortunately, when the axial dimension becomes large (making them slender enough to apply the Euler-Bernoulli theory), using the Timoshenko model with finite elements can lead to locking phenomena (shear locking). Similar issues also arise when models valid for thick plates or shells are applied to very thin structures. Structural models (beams, plates, shells) and the associated finite element concepts are the topic of specialized books (Bathe (1996)). They are therefore out of the scope of this section, which only aims at illustrating the shear locking phenomena and introducing possible remedies.

Consider a cantilever beam (Fig. 6.15) with the axis in the  $x = x_1$  direction (neutral axis) and cross-section in the  $x_2$ - $x_3$  plane. The beam is loaded by a uniform load  $f e_3$ . We only consider the contribution of bending in the  $x_1$ - $x_3$



**Figure 6.15:** Cantilever beam: geometry and other notations

plane in the Timoshenko beam theory. Accordingly, the problem is expressed in term of the transverse displacement  $u$  and the rotation  $\phi$  of a beam section around the  $x_2$ -axis, so that the displacement field has the form

$$\mathbf{u}(\mathbf{x}) = x_3 \phi(x) \mathbf{e}_1 + u(x) \mathbf{e}_3,$$

The strains are given by  $d\phi/dx$  (curvature) and  $\gamma := du/dx + \phi$  (shear strain), with associated forces  $M$  (bending moment) and  $T$  (shear force). The equilibrium equations read

$$(a) \quad \frac{dT}{dx} + f = 0 \quad ; \quad (b) \quad \frac{dM}{dx} - T = 0. \quad (6.23)$$

Boundary conditions for a cantilever beam are  $u(0) = \phi(0) = 0$  (clamped left end) and  $T(L) = M(L) = 0$  (free right end). Lastly, for an isotropic beam, the relevant constitutive relations read

$$(a) \quad T = k\mu S \gamma \quad ; \quad (b) \quad M = EI \frac{d\phi}{dx}, \quad (6.24)$$

where  $I$  is the area moment of inertia,  $S$  is the cross-section area,  $E$  is the Young modulus,  $\mu$  is the shear modulus and  $k$  is a known shear correction factor.

The weak enforcement of the equilibrium conditions (6.23) then reads

find  $(u, \phi) \in \mathcal{C}(0, 0)$  such that (6.25)

$$\int_0^L \left( M \frac{d\psi}{dx} + T \psi \right) dx = 0, \quad \int_0^L \left( T \frac{dw}{dx} - f w \right) dx = 0 \quad \forall (\psi, w) \in \mathcal{C}(0, 0)$$

where  $\mathcal{C}(0, 0)$  is the space of sufficiently regular functions such that  $w(0) = \psi(0) = 0$ . The weak form can alternatively be found by minimizing the functional of potential energy

$$\mathcal{P}(v, \theta) = \mathcal{W}(v, \theta) - \mathcal{F}(v, \theta)$$

among all kinematically admissible displacements  $\mathcal{C}(0, 0)$ . For the cantilever beam of Figure 6.15, the potential of external loads  $\mathcal{F}(v, \theta)$  is

$$\mathcal{F}(v, \theta) = \int_0^L f v \, dx \quad (6.26)$$

and the strain energy  $\mathcal{W}(v, \theta)$  for the Timoshenko beam theory (which includes shear strains) reads

$$\mathcal{W}(v, \theta) = \frac{1}{2} \int_0^L \left( EI \left( \frac{d\theta}{dx} \right)^2 + k\mu S \left( \frac{dv}{dx} + \theta \right)^2 \right) dx. \quad (6.27)$$

In order to highlight an analogy with the incompressibility constraint (Section 6.1), let us introduce in (6.27) the non-dimensional quantities  $\tilde{x} = x/L$ ,  $\tilde{v} = v/L$

$$\mathcal{W}(\tilde{v}, \theta) = \frac{EI}{2L} \int_0^1 \left( \left( \frac{d\theta}{d\tilde{x}} \right)^2 + \beta \left( \frac{d\tilde{v}}{d\tilde{x}} + \theta \right)^2 \right) d\tilde{x}. \quad (6.28)$$

with  $\beta = L^2 k \mu S / (EI)$  proportional to the square of the slenderness ratio  $(L/b)^2$ . Hence, for very slender beams  $((L/b) \rightarrow \infty)$ , we have  $\beta \rightarrow \infty$  and the second term (associated with shear strain) of the total potential energy in (6.28) is *finite* only if  $\gamma := \frac{d\tilde{v}}{d\tilde{x}} + \theta \rightarrow 0$  (that is the Euler-Bernoulli condition). A condition which closely resembles that of (6.7). The same locking issues are expected to arise but the remedy or reduced integration should be effective in alleviating the constraints.

### 6.3.1 Beam bending: a displacement approach

Let us now introduce a finite element discretization for (6.27) using linear shape functions for  $v_h$  and  $\theta_h$ . The beam of length  $L$  is divided into  $N_E$  elements of length  $h = L/N_E$ . If  $s$  denotes the distance from the first node on the physical element, the two shape functions are

$$N_1(s) = 1 - s/h, \quad N_2(s) = s/h.$$

The nodal values  $v^{(e)}$  and  $\theta^{(e)}$  for each node  $x^{(e)}$  are numbered in ascending order from left to right ( $e = 1, \dots, N_E + 1$ ). Accordingly

$$\{V_e\} = \{v^{(e)} \quad \theta^{(e)} \quad v^{(e+1)} \quad \theta^{(e+1)}\}^T.$$

denotes the vector of nodal values of the element  $e$  and the finite element approximations  $v_h$  and  $\theta_h$  writes,

$$v_h = \{N_u\}^T \{V_e\}, \quad \text{with} \quad \{N_u\} = \{N_1 \quad 0 \quad N_2 \quad 0\}^T \quad (6.29)$$

$$\theta_h = \{N_\phi\}^T \{V_e\}, \quad \text{with} \quad \{N_\phi\} = \{0 \quad N_1 \quad 0 \quad N_2\}^T \quad (6.30)$$

and their derivative with respect to  $x = x^{(e)} + s$ ,

$$\frac{dv_h}{dx} = \{B_u\}^T \{V_e\}, \quad \text{with} \quad \{B_u\} = \left\{ -\frac{1}{h} \quad 0 \quad \frac{1}{h} \quad 0 \right\}^T \quad (6.31)$$

$$\frac{d\theta_h}{dx} = \{B_\phi\}^T \{V_e\}, \quad \text{with} \quad \{B_\phi\} = \left\{ 0 \quad -\frac{1}{h} \quad 0 \quad \frac{1}{h} \right\}^T \quad (6.32)$$

Using (6.30-6.32) in (6.27) then defines the element stiffness matrix:

$$\begin{aligned} \mathcal{W}_e(v_h, \theta_h) &= \frac{1}{2} \int_{x^{(e)}}^{x^{(e+1)}} \left( EI \left( \frac{d\theta_h}{dx} \right)^2 + k \mu S \left( \frac{dv_h}{dx} + \theta_h \right)^2 \right) dx \\ &= \frac{1}{2} \{V_e\}^T [K_e] \{V_e\}, \end{aligned} \quad (6.33)$$

which can be conveniently written  $[K_e] = [K_e^1] + [K_e^2] + [K_e^3]$ , with

$$[K_e^1] = EI \int_{x^{(e)}}^{x^{(e+1)}} \{B_\phi\} \{B_\phi\}^T dx \quad (6.34)$$

$$[K_e^2] = k\mu S \int_{x^{(e)}}^{x^{(e+1)}} \{B_u\} \{B_u\}^T + \{B_u\} \{N_\phi\}^T + \{N_\phi\} \{B_u\}^T dx \quad (6.35)$$

$$[K_e^3] = k\mu S \int_{x^{(e)}}^{x^{(e+1)}} \{N_\phi\} \{N_\phi\}^T dx \quad (6.36)$$

The exact evaluation of the integrals in (6.34) and (6.35) can be done analytically or (equivalently) using 1-point Gaussian quadrature, to obtain

$$[K_e^1] = \frac{EI}{h} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} ; \quad [K_e^2] = \frac{k\mu S}{2h} \begin{bmatrix} 2 & -h & -2 & -h \\ -h & 0 & h & 0 \\ -2 & h & 2 & h \\ -h & 0 & h & 0 \end{bmatrix}.$$

whereas the evaluation of the integrals in (6.36) require 2-point Gaussian quadrature (or analytic evaluation), to obtain

$$[K_e^3] = \frac{k\mu Sh}{6} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 \end{bmatrix}. \quad (6.37)$$

The MATLAB function returning the exact evaluation of the element stiffness matrix ( $[K_e] = [K_e^1] + [K_e^2] + [K_e^3]$ ) given the element nodal coordinates and the material parameters ( $EI \rightarrow EI$ ,  $k\mu S \rightarrow k\mu S$ ) simply reads

```
function Ke=B2_P1P1_beam_Ke(X,EI,kmuS);
r1=X(1); r2=X(2); % radial coordinates of nodes
h=r2-r1; % length of element
Ke=(EI/h)*[0, 0, 0, 0; 0, 1, 0, -1; ...
0, 0, 0, 0; 0, -1, 0, 1];
Ke=Ke+kmuS/(6*h)* [ 6, -3*h, -6, -3*h; -3*h, 2*h*h, 3*h, h*h; ...
-6, 3*h, 6, 3*h; -3*h, h*h, 3*h, 2*h*h];
% element stiffness
```

Using instead a 1-point Gaussian quadrature (i.e. applying a reduced integration technique) for  $[K_e^3]$  instead yields

$$[K_e^3] \approx [K_e^3]^{\text{red}} = \frac{k\mu Sh}{4} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}. \quad (6.38)$$

Similarly, the function B2\_P1P1R\_beam\_Ke returns the modified element stiffness matrix with reduced integration ( $[K_e]^{\text{red}} = [K_e^1] + [K_e^2] + [K_e^3]^{\text{red}}$ ). It is worth noting that both  $[K_e]$  and  $[K_e]^{\text{red}}$  have rank 2, i.e. have no spurious modes. Using (6.29) in (6.26) defines the element right-hand side:

$$\int_{x^{(e)}}^{x^{(e+1)}} f v_h dx = \{V_e\}^T \{F_e\} \quad (6.39)$$

with

$$\{F_e\} = \int_{x^{(e)}}^{x^{(e+1)}} \{N_u\} dx = f \frac{h}{2} \{1 \ 0 \ 1 \ 0\}^T \quad (6.40)$$

The function B2\_P1P1\_beam\_Fe performs the evaluation of the element right-hand side.

We now solve the cantilever beam problem described in Figure 6.15 with the program beambending\_P1P1 provided in the MATLAB distribution (its reading being left as an exercise). We use NE=10 or 100 finite elements, with the element stiffness matrix taken as either its exact version  $[K_e] = [K_e^1] + [K_e^2] + [K_e^3]$  or its version modified by reduced integration  $[K_e]^{\text{red}}$ . Figure 6.16 plots the ratio of the tip deflection to that given by the Euler-Bernoulli slender beam theory against the slenderness ratio  $L/b$ . When using first order shape functions for both transverse displacement  $u$  and rotation  $\phi$ , the locking effect clearly appears, as the tip deflection in the limit of infinite slenderness ( $L/b \rightarrow \infty$ ) disagrees with the value predicted by the Euler-Bernoulli theory. In contrast, the use of reduced integration scheme makes the Timoshenko model behaving correctly in the Euler-Bernoulli limit  $L/b \rightarrow \infty$ .

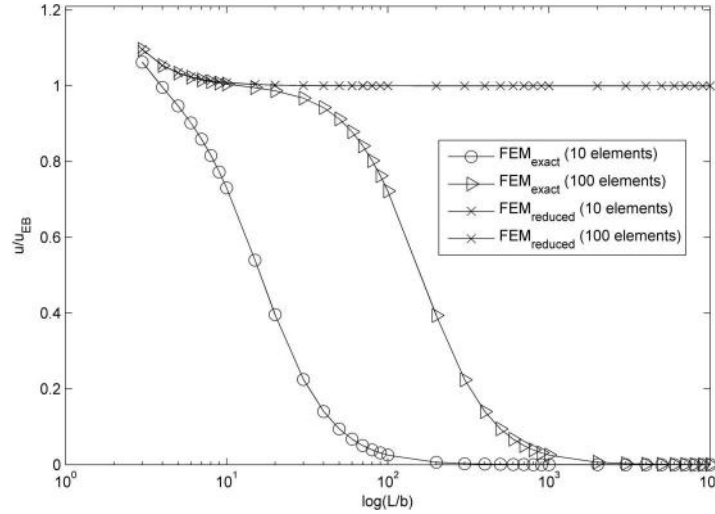


Figure 6.16: Bending of a beam: shear locking in the slender limit

### 6.3.2 Beam bending: a mixed approach

Similarly to what has been done for the incompressibility constraint in Section 6.2, a mixed approach can be obtained by introducing the shear force  $T = k\mu S\gamma$  as an independent quantity. The associated Herrmann-like functional reads

$$\mathcal{H}[v, \theta, G] = \int_0^L \left( \frac{1}{2} EI \left( \frac{d\theta}{dx} \right)^2 + G \left( \frac{dv}{dx} + \theta \right) - \frac{1}{2k\mu S} G^2 \right) dx - \int_0^L f v dx$$

and is defined for admissible displacements  $(v, \theta) \in \mathcal{C}(0, 0)$  and admissible shear forces  $G$ , which here are only required to be square-integrable. The stationarity

conditions with respect to  $(v, \theta, G)$  write:

$$0 = \int_0^L \left( EI \frac{d\phi}{dx} \frac{d\psi}{dx} + T \left( \frac{dw}{dx} + \psi \right) \right) dx - \int_0^L f w dx \\ + \int_0^L \left( \left( \frac{du}{dx} + \phi \right) - \frac{T}{k\mu S} \right) Q dx \quad \forall (w, \psi) \in \mathcal{C}(0, 0), \forall Q \in \mathcal{Q}$$

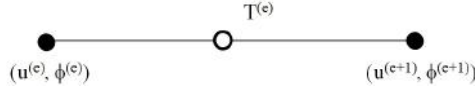
or, equivalently,

Find  $(u, \phi) \in \mathcal{C}(0, 0)$  and  $T \in \mathcal{Q}$  such that

$$\int_0^L \left( \left( EI \frac{d\phi}{dx} \frac{d\psi}{dx} + T \left( \frac{dw}{dx} + \psi \right) \right) \right) dx = \int_0^L f w dx \quad \forall (w, \psi) \in \mathcal{C}(0, 0) \quad (6.41a)$$

$$\int_0^L \left( \left( \frac{du}{dx} + \phi \right) - \frac{T}{k\mu S} \right) Q dx = 0 \quad \forall Q \in \mathcal{Q} \quad (6.41b)$$

**Implementation.** Assume now that a piecewise-constant approximation (hence discontinuous between elements) is used for  $T_h$  and the associated test functions  $Q_h$ , while  $u_h$  and  $\phi_h$  (and the associated test functions  $w_h, \psi_h$ ) are defined in terms of piecewise-linear continuous interpolations (Fig. 6.17).



**Figure 6.17:** A mixed element: displacement nodes and shear node

Setting  $Q = Q_e$  in (6.41b) yields the relation

$$Q_e \left( \{G_e\}^T \{U_e\} - H_e T_e \right) = 0 \quad (6.42)$$

with

$$\{G_e\} = \int_{x^{(e)}}^{x^{(e+1)}} (\{B_u\} + \{N_\phi\}) dx = \left\{ -1 \quad \frac{h}{2} \quad 1 \quad \frac{h}{2} \right\}^T ; \quad H_e = \frac{h}{k\mu S}$$

which involves only one element  $E_e$ . Equation (6.42) can then be used to express  $T_e$  in terms of  $\{U_e\}$ . The contribution of the element  $E_e$  to equation (6.41a), given by

$$\{W_e\}^T \left( [K_e^1] \{U_e\} + \{G_e\} T_e - \{F_e\} \right),$$

can then be recast, upon eliminating  $T_e$  as explained above, in the form

$$\{W_e\}^T \left( [K_e]^{\text{mixed}} \{U_e\} - \{F_e\} \right)$$

with

$$[K_e]^{\text{mixed}} = [K_e^1] + \{G_e\} H_e^{-1} \{G_e\}^T, \\ \{G_e\} H_e^{-1} \{G_e\}^T = \frac{k\mu S}{4h} \begin{bmatrix} 4 & -2h & -4 & -2h \\ -2h & h^2 & 2h & h^2 \\ -4 & 2h & 4 & 2h \\ -2h & h^2 & 2h & h^2 \end{bmatrix}$$



It is worth noting that  $\{G_e\}H_e^{-1}\{G_e\}^T = [K_e^2] + [K_e^3]^{\text{red}}$ , implying that  $[K_e]^{\text{mixed}} = [K_e]^{\text{red}}$ . The mixed approach after elimination of the shear forces is therefore found to be identical to the displacement approach with reduced integration (section 6.3.1).

**Exercise 6.3** (mixed finite element). *Develop a mixed finite element with continuous quadratic interpolation for  $\phi_h$ , continuous linear interpolation for  $u_h$  and constant (discontinuous) interpolation for  $T_h$  and compare.*

**Exercise 6.4** (another mixed finite element). *Develop a mixed finite element with continuous quadratic interpolation for  $\phi_h$ , continuous linear interpolation for  $u_h$  and linear discontinuous interpolation for  $T_h$ . Compare and comment.*

**Exercise 6.5** (Beam bending: reduced integration scheme for Q4 element). *Using the insight gained from the theory of beams, develop a suitable reduced integration scheme for Q4 elements in order to alleviate the locking phenomena described in Figure 6.3 (input files Chap6\_cantilever\_Q.\*).*

## 6.4 Diffusion-transport equation

Many physical problems (like the steady state diffusion of a given species of concentration  $\phi$  transported by a fluid flow of given velocity  $\mathbf{u}$ ) can be described by the so called diffusion-transport equation of the type

$$-\text{div}(\mu \nabla \phi) + \mathbf{u} \cdot \nabla \phi = f \quad (\text{in } \Omega), \quad \phi = \phi^D \quad (\text{on } \partial\Omega) \quad (6.43)$$

When the diffusion term  $\text{div}(\mu \nabla \phi)$  is dominated by the convective term  $\mathbf{u} \cdot \nabla \phi$ , strong gradients arise in very thin boundary layers. The standard Finite Element Method then fails to predict the correct behaviour and alternative approaches should be preferred, as discussed in the sequel by means of a 1D example.

### 1D diffusion transport equation

In order to describe the difficulties and introduce possible remedies, we consider the equivalent 1D model on the unit segment  $0 < x < 1$ :

$$-\mu \phi''(x) + u \phi'(x) = 0 \quad (6.44)$$

where  $\phi(x)$  is the unknown distribution with boundary conditions  $\phi(0) = 0$ ,  $\phi(1) = 1$  and  $u, \mu$  are strictly positive given parameters. The general solution is:

$$\phi(x) = A \exp((u/\mu)x) + B$$

Imposing the boundary conditions  $\phi(0) = 0$ ,  $\phi(1) = 1$  one obtains the analytical solution to the given problem:

$$\phi(x) = \frac{\exp((u/\mu)x) - 1}{\exp(u/\mu) - 1} \quad (6.45)$$

The qualitative plot of Figure 6.18 shows that a thin boundary layer with strong gradients develops near  $x = 1$  for  $u/\mu \gg 1$ .

Multiplying (6.44) by  $w$  and integrating on  $0 < x < 1$ , we have the identity:

$$\int_0^1 (-\mu \phi''(x)w(x) + u \phi'(x)w(x)) dx = 0$$

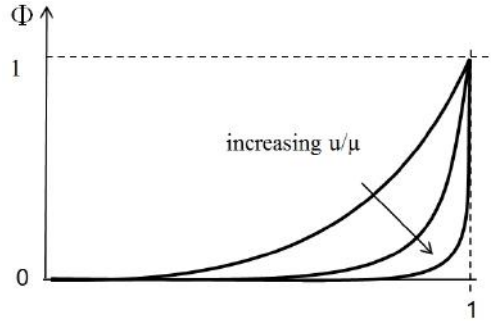
Then integrating by parts the first term:

$$\int_0^1 \mu \phi''(x) w(x) dx = \mu \phi'(1) w(1) - \mu \phi'(0) w(0) - \int_0^1 \mu \phi'(x) w'(x) dx$$

and applying boundary conditions on  $w$ , the weak form of (6.44) can be finally expressed as

$$\text{Find } \phi \in \mathcal{C}(\phi^D), \quad \int_0^1 (\mu \phi'(x) w'(x) + u \phi'(x) w(x)) dx = 0 \quad \forall w \in \mathcal{C}(0) \quad (6.46)$$

where  $\mathcal{C}(\phi^D)$  is the space of sufficiently regular functions  $f$  such that  $f(0) = 0, f(1) = 1$  and  $\mathcal{C}(0)$  is the space of sufficiently regular functions  $f$  such that  $f(0) = f(1) = 0$ .



**Figure 6.18:** Exact solution for different values of  $u/\mu$

Let us now discretise (6.46) in the spirit of the Galerkin isoparametric FEM using linear shape functions both for  $\phi$  and  $w$ . The unit segment is partitioned in  $M - 1$  elements of length  $h$ . Unknown nodal values are numbered in ascending order from left to right. If  $s$  denotes the distance from the first node on the physical element, the two shape functions are:

$$N_1(s) = 1 - \frac{s}{h}, \quad N_2(s) = \frac{s}{h}$$

Hence:

$$[K_e] = \frac{\mu}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{u}{2} \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$$

Let us now assemble the  $(i - 1)th$  row (the row left-multiplied by  $w^{(i)}$ ). The terms from the second row of the element matrix  $[K_{i-1}]$ :

$$\frac{\mu}{h} (-\phi^{(i-1)} + \phi^{(i)}) + \frac{u}{2} (-\phi^{(i-1)} + \phi^{(i)})$$

have to be added to the terms from the first row of the element matrix  $[K_i]$ :

$$\frac{\mu}{h} (\phi^{(i)} - \phi^{(i+1)}) + \frac{u}{2} (-\phi^{(i)} + \phi^{(i+1)})$$

Finally one has:

$$(\alpha - 1)\phi^{(i+1)} + 2\phi^{(i)} - (\alpha + 1)\phi^{(i-1)} = 0 \quad (6.47)$$

with  $\alpha = uh/(2\mu)$ .

Eq.(6.47) admits solutions in the form  $\phi^{(i)} = \rho^i$  provided  $\rho$  is a root of the characteristic polynomial:

$$(\alpha - 1)\rho^2 + 2\rho - (\alpha + 1) = 0 \quad (6.48)$$

From the theory of linear finite difference equations it is known that, if the two roots  $\rho_1$  and  $\rho_2$  are distinct, then the general solution is a linear combination of  $\rho_1^i$  and  $\rho_2^i$ . Since the two roots of (6.48) are:

$$\rho_1 = (1 + \alpha)/(1 - \alpha), \quad \rho_2 = 1 \quad (6.49)$$

we have  $\phi^{(i)} = A_1\rho_1^i + A_2\rho_2^i$ , and hence, after enforcing boundary conditions  $\phi^{(1)} = 0$ ,  $\phi^{(M)} = 1$ :

$$\phi^{(i)} = \left[1 - \left(\frac{1 + \alpha}{1 - \alpha}\right)^i\right] / \left[1 - \left(\frac{1 + \alpha}{1 - \alpha}\right)^M\right] \quad (6.50)$$

If  $\alpha > 1$  the power in the numerator has a negative base and hence  $\phi^{(i)}$  oscillates.

Since  $\alpha = uh/(2\mu)$ ,  $h$  can always be made small enough to get  $\alpha < 1$ . In many cases, however, this will not be viable since it would require to employ excessively refined meshes.

**Upwinding and modified viscosity.** Equation (6.47) is identical to the following finite difference approximation of 6.44:

$$-\mu \frac{\phi^{(i+1)} - 2\phi^{(i)} + \phi^{(i-1)}}{h^2} + u \frac{\phi^{(i+1)} - \phi^{(i-1)}}{2h} = 0 \quad (6.51)$$

However, it is known that a slight modification of (6.51) eliminates spurious oscillations, as explained in the sequel. If central differences for the transport term are replaced by backward differences one gets:

$$-\mu \frac{\phi^{(i+1)} - 2\phi^{(i)} + \phi^{(i-1)}}{h^2} + u \frac{\phi^{(i)} - \phi^{(i-1)}}{h} = 0 \quad (6.52)$$

This technique is called "upwinding" and has deep physical roots. Indeed, the material derivative of the concentration  $\phi$  in a steady state 1D problem with velocity  $u$  is  $u\phi'(x)$ . In the absence of diffusion, a material particle transports its concentration unchanged along the flux from  $x^{(i-1)}$  to  $x^{(i)}$  after a suitable time lag. It is thus rather natural to express the  $u\phi'(x)$  term with finite differences along the "upwind" (against the wind) direction.

Using the identity:

$$\frac{\phi^{(i)} - \phi^{(i-1)}}{h} = \frac{\phi^{(i+1)} - \phi^{(i-1)}}{2h} - \frac{h}{2} \frac{\phi^{(i+1)} - 2\phi^{(i)} + \phi^{(i-1)}}{h^2} \quad (6.53)$$

one can show that (6.52) is equivalent to:

$$-\mu_h \frac{\phi^{(i+1)} - 2\phi^{(i)} + \phi^{(i-1)}}{h^2} + u \frac{\phi^{(i+1)} - \phi^{(i-1)}}{2h} = 0 \quad (6.54)$$

with  $\mu_h = \mu(1 + \alpha)$ . The solution is thus still given by (6.50) provided that  $\alpha$  is replaced by  $\alpha_h = uh/(2\mu_h) = \alpha/(1 + \alpha)$ . Since  $\alpha_h < 1$  oscillations disappear.

In general, the stabilisation of oscillations can be achieved introducing a modified diffusivity of the form:

$$\mu_h = \mu(1 + \chi(\alpha)), \quad \lim_{\alpha \rightarrow 0} \chi(\alpha) = 0 \quad (6.55)$$

The choice  $\chi(\alpha) = \alpha$  corresponds to the upwinding technique; another choice commonly employed is the "exponential fitting"  $\chi(\alpha) = \alpha - 1 + 2\alpha/(\exp(2\alpha) - 1)$ .

The concepts of modified viscosity and upwinding can be extended to finite element applications in higher dimensions.

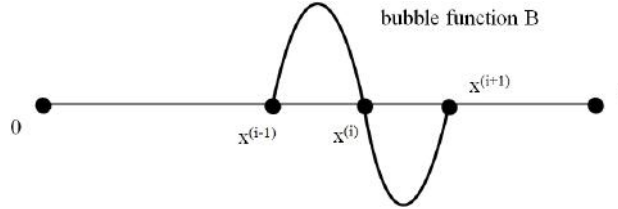
**Petrov-Galerkin approaches and bubble functions.** Let us now consider a different space for the test functions. Let the test field be defined as:

$$w(x) = \sum_{i=2}^{M-1} \tilde{M}_i(x) w^{(i)} \quad (6.56)$$

where the global shape functions  $\tilde{M}_i$  are:

$$\tilde{M}_i(x) = \tilde{N}_i(x) + \begin{cases} B((x - x^{(i-1)})/h) & x^{(i-1)} \leq x \leq x^{(i)} \\ -B((x - x^{(i)})/h) & x^{(i)} \leq x \leq x^{(i+1)} \end{cases}$$

and  $B(\eta) = 3\eta(1 - \eta)$  is called "bubble" function (see Figure 6.19).



**Figure 6.19:** Bubble function

Functions  $\tilde{N}_i(x)$  are the standard linear hat-shaped functions. On the contrary the concentration  $\phi$  is interpolated in the standard way:

$$\phi(x) = \sum_{i=1}^M \tilde{N}_i(x) \phi^{(i)} \quad (6.57)$$

The approaches in which the functional space for the test function  $w$  differs from that of  $\phi$  are called Galerkin-Petrov techniques and are the topic of specialised books. Here we will just show a simple result which clearly indicates their potentialities. Indeed, since  $d\tilde{N}_i/dx$  is piecewise constant, using the identities:

$$\int_0^1 B(\eta) d\eta = \frac{1}{2}, \quad \int_0^1 (dB/d\eta) d\eta = 0 \quad (6.58)$$

it can be shown that the equation associated with  $w^{(i)}$  is the same as (6.52) and is thus stable.

## Application to linear elastic fracture mechanics

---

The linear elastic fracture mechanics, also called brittle fracture mechanics, aims at evaluating the effect of the presence of cracks in solids on mechanical quantities such as stresses, in order to assess their nocivity. Cracks are dangerous because they may propagate (i.e. grow), either progressively or suddenly (depending on the loading and materials involved). Moreover, they facilitate the deterioration of structures through other mechanisms such as corrosion.

The linear elastic fracture mechanics is based on the framework of linear elasticity under the small-deformation assumption (SDA). This framework is, in this introductory chapter, further restricted by assuming quasi-static evolutions (i.e. neglecting all inertia effects) and plane-strain conditions.

This chapter aims at presenting, and comparing, computational approaches based on finite elements that allow to assess, for arbitrary configurations, the propensity of a crack to grow. These approaches work by either exploiting the displacements at nodes in the vicinity of the crack tip (Section 7.3) or numerically evaluating the energy release rate (Section 7.4). They are both demonstrated on the same example, for illustration and comparison purposes. As a preliminary, essential concepts of fracture mechanics are reviewed in Section 7.1.

### 7.1 Linear elastic fracture mechanics

This section concisely reviews some classical and essential facts about linear elastic fracture mechanics (LEFM), so as to make this chapter self-sufficient. Complete expositions of this subject are available in the course Suquet (2004) as well as many monographs, such as those by Broberg (1999) or Leblond (2002).

#### 7.1.1 Crack, opening mode

In the LEFM framework to which this chapter is confined, a crack is a material separation that occurs on an arbitrarily-shaped surface in a solid with elastic constitutive behavior. A crack is thus modelled by two open surfaces  $F^+$  and  $F^-$  (the *faces* of the crack), which are identical but with opposite orientation (the crack being therefore assumed, in the absence of loading, to have zero thickness). Letting

$F$  denote the common geometrical locus of surfaces  $F^+$  and  $F^-$  in the reference configuration, the displacement undergoes a discontinuity (jump) across  $F$ :

$$[[u(x)]] = u^+(x) - u^-(x) \quad x \in F. \quad (7.1)$$

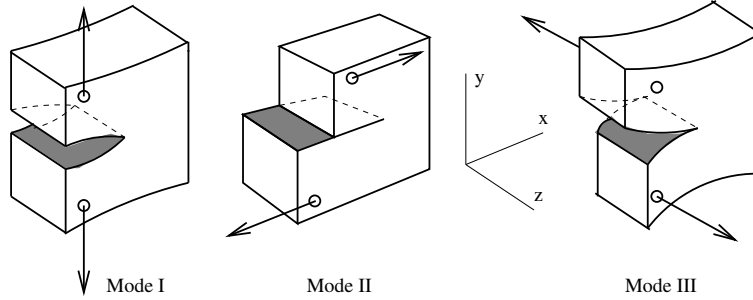
This discontinuity reflects the opening of the crack when sole loading is applied to the solid. The LEFM framework assumes the crack faces to be traction-free, neglecting possible contact or friction between faces.

This introductory chapter on computational fracture mechanics is further restricted to 2D, plane-strain configurations. In this setting, a crack is modelled by an open arc  $F$  lying inside the planar domain  $\Omega$  occupied by the cross-section of the solid. A straight crack, modelled by a rectilinear segment, thus corresponds to an infinite slit in a three-dimensional translationally-invariant solid.

**Opening modes.** When the solid is subjected to some excitation, the crack faces undergo a relative motion. Following a well-established terminology, such relative motions occur in three possible *modes* (Figure 7.1):

- (i) Mode I (opening): the direction of relative motion is normal to the crack;
- (ii) Mode II (in-plane shear): the direction of relative motion is tangential to the crack and in-plane;
- (iii) Mode III (antiplane shear): the direction of relative motion is tangential to the crack and perpendicular to the reference plane.

In plane-strain configurations, modes I and II only are activated.



**Figure 7.1:** The three crack opening modes.

**Equilibrium of a cracked solid.** The geometrical configuration of the solid body is described using a *two-dimensional* domain  $\Omega \subset \mathbb{R}^2$  with boundary  $\partial\Omega = (S_u \cup S_T) \cup (F^+ \cup F^-)$ . The notation  $\Omega(F)$  will sometimes be used for emphasizing the fact that the geometry of the solid depends on the crack locus  $F$ . Vanishing body forces ( $\mathbf{f} = \mathbf{0}$ ) are assumed for simplicity. The equilibrium of the thus-defined solid body is governed by the local equations

$$\operatorname{div} \boldsymbol{\sigma} = \mathbf{0}, \quad \boldsymbol{\sigma} = \mathcal{A}:\boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \mathbf{u} + \nabla^T \mathbf{u}) \quad (\text{in } \Omega), \quad (7.2a)$$

the boundary conditions on the external surface (externally applied loading)

$$\mathbf{u} = \mathbf{u}^D \quad (\text{on } S_u), \quad \boldsymbol{\sigma} \cdot \mathbf{n} = \mathbf{T}^D \quad (\text{on } S_T), \quad (7.2b)$$

and the traction-free condition on the crack faces

$$\boldsymbol{\sigma} \cdot \mathbf{n} = \mathbf{0} \quad (\text{on } F^+ \cup F^-). \quad (7.2c)$$

All the fields involved in (7.2a–c) depend on coordinates  $(x_1, x_2)$  only; moreover, the components  $u_3$  of the displacement,  $\varepsilon_{31}, \varepsilon_{32}, \varepsilon_{33}$  of the strain tensor and  $\sigma_{32}, \sigma_{33}$  of the stress tensor are all zero.

### 7.1.2 Energy release rate

In view of the set of assumptions that define the LEFM framework, the energy balance for a crack propagation problem, which results from an application of the first two laws of thermodynamics, reads (Germain et al., 1983; Maugin, 1992)

$$P_{\text{ext}} = \dot{W} + \mathcal{D}, \quad \mathcal{D} \geq 0, \quad (7.3)$$

where  $P_{\text{ext}}$  is the power supplied to the solid from external sources,  $\dot{W}$  is the variation of the elastic strain energy and  $\mathcal{D}$  is the power dissipated (as heat) by the crack extension. In particular, the crack propagation is irreversible (the dissipation  $\mathcal{D}$  being non-negative).

Let  $P(F, \mathbf{u}^D, \mathbf{T}^D)$  denote the value reached by the potential energy  $\mathcal{P}(\mathbf{u})$ , defined by (1.29), when  $\mathbf{u}$  solves the equilibrium problem (7.2a–c), i.e. minimizes  $\mathcal{P}$ , for a given crack configuration  $F$  and loading  $(\mathbf{u}^D, \mathbf{T}^D)$ :

$$P(F, \mathbf{u}^D, \mathbf{T}^D) = \frac{1}{2} \int_{\Omega(F)} \boldsymbol{\varepsilon}[\mathbf{u}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{u}] \, dV - \int_{S_T} \mathbf{T}^D \cdot \mathbf{u} \, dS. \quad (7.4)$$

The following remarkable result, linking the power dissipated by crack propagation to the derivative of the potential energy with respect to the crack configuration, with the loading  $(\mathbf{u}^D, \mathbf{T}^D)$  kept fixed:

$$\mathcal{D} = - \left\langle \frac{\partial P}{\partial F}(F, \mathbf{u}^D, \mathbf{T}^D), \dot{F} \right\rangle. \quad (7.5)$$

The mathematical concept of a "derivative with respect to the crack  $F$ ", which is involved in the above expression of  $\mathcal{D}$ , arises from evaluating the perturbation of a scalar quantity depending on  $F$  (here, the potential energy (7.4)) induced by perturbations of the shape of  $F$ :

$$\begin{aligned} \delta P &= P(F + \delta F, \mathbf{u}^D, \mathbf{T}^D) - P(F, \mathbf{u}^D, \mathbf{T}^D) \\ &= \left\langle \frac{\partial P}{\partial F}(F, \mathbf{u}^D, \mathbf{T}^D), \delta F \right\rangle + o(\delta F) \end{aligned}$$

In other words, and as suggested by the notation  $\langle \cdot \rangle$ ,  $\partial P / \partial F$  is a linear mapping that, for any perturbation  $\delta F$  of  $F$ , yields the corresponding perturbation  $\delta P$  of  $P$  to first order in  $\delta F$ . We do not attempt here to present a rigorous formulation of the underlying mathematical concept of *domain derivative* (Murat and Simon, 1976; Sokolowski and Zolesio, 1992), contenting ourselves with the above intuitive formulation. Expression (7.5) can however be given a more explicit, and easier to

grasp, form for the case of a straight crack. In that case,  $F$  is a rectilinear segment  $[A, B]$  and can, after a suitable choice of coordinate system, be represented as

$$F = F_{a,b} = [A, B] = \{-a \leq x_1 \leq b, x_2 = 0\},$$

A straight crack propagation along the line  $(AB)$  defined by its initial configuration is then entirely characterized by the time-dependent abscissae  $-a(t)$  and  $b(t)$  of its tips  $A$  and  $B$  (this convention being chosen so that both tip velocities are positive, i.e.  $\dot{a}, \dot{b} \geq 0$ , for a crack *extension*). In particular, the potential energy at equilibrium for a given crack configuration  $F_{a,b}$  has the form  $P(a, b, \mathbf{u}^D, \mathbf{T}^D)$ , and (7.5) becomes

$$\mathcal{D} = -\frac{\partial P}{\partial a}(a, b, \mathbf{u}^D, \mathbf{T}^D)\dot{a} - \frac{\partial P}{\partial b}(a, b, \mathbf{u}^D, \mathbf{T}^D)\dot{b}. \quad (7.6)$$

For each crack tip  $A, B$ , an *energy release rate*  $G_A, G_B$  is defined according to

$$G_A = -\frac{\partial P}{\partial a}(a, b, \mathbf{u}^D, \mathbf{T}^D), \quad G_B = -\frac{\partial P}{\partial b}(a, b, \mathbf{u}^D, \mathbf{T}^D) \quad (7.7)$$

so the power instantaneously dissipated by the crack extension is given by

$$\mathcal{D} = G_A\dot{a} + G_B\dot{b}.$$

In the framework of linear fracture mechanics, the propensity of a crack to propagate in a given material is commonly characterized by the *critical energy release rate*  $G_c$  of that material. The possible propagation of the crack tip  $x_1 = a$  is then decided by the criterion:

$$\begin{cases} \text{if } G_A < G_c & \text{then } \dot{a} = 0 & \text{(the crack does not propagate),} \\ \text{if } G_A = G_c & \text{then } \dot{a} \geq 0 & \text{(the crack may propagate)} \end{cases} \quad (7.8)$$

and similarly for the other tip  $x_1 = b$ . In this context, the objective of computational fracture mechanics is to evaluate  $G_A$  and  $G_B$  so as to predict the conditions and occurrence of crack propagation (by comparison with the critical value  $G_c$  of the material), to simulate the subsequent crack evolution, or to determine conditions on the loading, material or geometry that forbid crack propagation.

### 7.1.3 Stress singularity at the crack tips, material toughness

**Singular behavior at the crack tips and stress intensity factors.** Let  $(r, \theta)$  denote polar coordinates emanating from a crack tip (with  $\theta = 0$  defining the direction tangent to the crack and pointing away from it), see Figure 7.2. Assuming the constitutive material to be linearly elastic, the stress components at a point  $(r, \theta)$  close to the crack tip (i.e. such that  $r$  is small relative to the crack length  $\ell$ ), has (assuming a state of plane strain) the following asymptotic form:

$$\begin{aligned} \sigma_{rr} = & \frac{K_I}{4\sqrt{2\pi r}} \left[ 5 \cos\left(\frac{\theta}{2}\right) - \cos\left(\frac{3\theta}{2}\right) \right] \\ & + \frac{K_{II}}{4\sqrt{2\pi r}} \left[ -5 \sin\left(\frac{\theta}{2}\right) + 3 \sin\left(\frac{3\theta}{2}\right) \right] + O(1), \end{aligned} \quad (7.9a)$$



$$\begin{aligned} \sigma_{\theta\theta} = & \frac{K_I}{4\sqrt{2\pi r}} \left[ 3 \cos\left(\frac{\theta}{2}\right) + \cos\left(\frac{3\theta}{2}\right) \right] \\ & + \frac{K_{II}}{4\sqrt{2\pi r}} \left[ -3 \sin\left(\frac{\theta}{2}\right) - 3 \sin\left(\frac{3\theta}{2}\right) \right] + O(1), \end{aligned} \quad (7.9b)$$

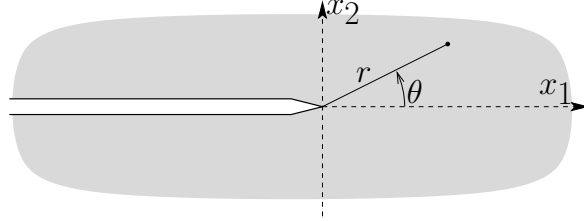
$$\begin{aligned} \sigma_{r\theta} = & \frac{K_I}{4\sqrt{2\pi r}} \left[ \sin\left(\frac{\theta}{2}\right) + \sin\left(\frac{3\theta}{2}\right) \right] \\ & + \frac{K_{II}}{4\sqrt{2\pi r}} \left[ \cos\left(\frac{\theta}{2}\right) + 3 \cos\left(\frac{3\theta}{2}\right) \right] + O(1). \end{aligned} \quad (7.9c)$$

They have in particular a  $O(1/\sqrt{r})$  singular behavior. The *stress intensity factors*  $K_I$ ,  $K_{II}$ ,  $K_{III}$  featured in the above expansions depend on the domain and crack geometry, and also on the nature and magnitude of the applied loading.

Likewise, the (vector-valued) displacement discontinuity across the crack admits the following asymptotic expansion in the vicinity of the crack tip:

$$[\mathbf{u}] = \frac{4}{\mu} \sqrt{\frac{r}{2\pi}} \{ (1-\nu)[K_I \mathbf{e}_1 + K_{II} \mathbf{e}_2] + K_{III} \mathbf{e}_3 \} + O(r). \quad (7.10)$$

As usual, the notation  $O(r^\alpha)$  in the remainder of expansions (7.9a-c) and (7.10) stands for unspecified terms that are equivalent to  $r^\alpha$  in the limit  $r \rightarrow 0$ .



**Figure 7.2:** Local geometry in the crack tip vicinity; notations.

**Material toughness.** Considering a crack propagation along a single mode (for example in opening mode I, often deemed to be the most dangerous), the propagation criterion introduced in 1957 by Irwin is expressed in terms of a material threshold  $K_{Ic}$  on  $K_I$ , called *material toughness*:

$$\begin{cases} \text{if } K_I < K_{Ic} & \text{then } \dot{a} = 0, \\ \text{if } K_I = K_{Ic} & \text{then } \dot{a} \geq 0 \end{cases} \quad (7.11)$$

The toughness  $K_{Ic}$  is a material characteristic, which can be determined on the basis of traction or bending experiments on fracture specimens where crack propagation initiales from a preexisting notch. The concept of material toughness, while debatable from a theoretical standpoint as the singular behavior of stresses at the crack tips implies the local onset of plastic zones in their vicinity, is nevertheless corroborated for certain classes of materials (in particular metals) by experimental evidence.

### 7.1.4 Link between energy approach (global) and singularity (local)

In the framework of quasi-static linear elasticity and small deformations, the energy release rate  $G$  and the stress intensity factors associated with a crack tip are related through *Irwin's formula* (here restricted to the plane strain case):

$$G(s) = \frac{1 - \nu^2}{E} (K_I^2 + K_{II}^2). \quad (7.12)$$

This relationship establishes in particular a link between the material toughness  $K_{Ic}$  associated with pure (opening) mode I propagation and the critical energy release rate  $G_c$ :

$$G_c = \frac{1 - \nu^2}{E} K_{Ic}^2. \quad (7.13)$$

## 7.2 Purpose of computational fracture mechanics

Analysing the nocivity of cracks largely rests upon comparing the values of characteristic quantities (such as the energy release or stress intensity factors) for given geometry and loading conditions to the corresponding critical values  $G_c$  or  $K_{Ic}$  of the material. This type of comparison is in particular effected repetitively in algorithms that incrementally compute crack propagation under a given loading history.

Such comparisons require the ability to accurately evaluate  $G$  or  $K_I, K_{II}$  for a given configuration. On the one hand, many analytical or semi-analytical solutions for cracked solids that are either unbounded or have simple shapes are available in compendia (e.g. Tada, Paris, and Irwin, 2000). However, applying these solutions to cracked structures with complex geometry only yields rough approximations, and better accuracy is only attainable by means of numerical computation. The remainder of this chapter is devoted to the exposition of finite element-based approaches for the numerical evaluation of stress intensity factors (Section 7.3) and of the energy release rate (Section 7.4). A broader overview of the methods of computational fracture mechanics is available in e.g. Aliabadi and Rooke (1991), Ingraffea and Wawrzynek (2003) or Pommier et al. (2010).

Within the restrictive LEFM framework adopted in this chapter, the energy release rate may be deduced from the stress intensity factors using Irwin's formula (7.12). However, the concept of stress intensity factors is meaningful only in association with a linear elastic constitutive behavior, whereas the concept of energy release rate has broader generality and may be used in situations that fall outside the LEFM framework (e.g. crack propagation in elastic-plastic materials).

For this reason, a substantial part of this chapter is devoted to the numerical evaluation of the energy release rate using an approach known as the *G-θ method*, which directly implements the definition of  $G$  as derivative of the potential energy at equilibrium under crack shape changes and does not involve the stress intensity factors. Generalizations of this approach have in particular been proposed for crack propagation in elastic-plastic solids (Lorentz, Wadier, and Debruyne, 2000).

### 7.3 Numerical evaluation of stress intensity factors

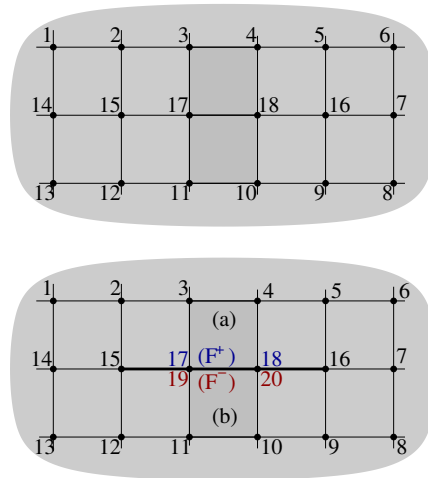
#### 7.3.1 The finite element method for cracked solids

Modelling cracked solids using the finite element method follows the general principles laid out in Chapters 2 and 3. There are however important issues specific to this class of problems that require elaboration.

**Modelling of the crack opening displacement.** In the LEFM framework, a crack is modelled (for the 2D configurations considered in this chapter) as an arc  $F$  across which the displacement field is discontinuous. Any finite element model of a cracked solid must account for that displacement discontinuity.

The basis functions defining the finite element approximation space are continuous over the whole approximate domain  $\Omega_h$  (Chapter 2). In this framework, a crack therefore cannot cross elements, and the discretized  $F_h$  must consist of a set of finite element boundaries. Moreover, nodes located on  $F_h$  must be treated as "double nodes", as suggested by Fig. 7.3. This amounts to modeling the cracked solid as the limiting case of a solid containing a traction-free cavity that becomes infinitely thin. Nodes located on the two crack faces, and thus possibly geometrically identical, must be given distinct numbers. The basis functions implicitly defined by such a mesh are then by construction discontinuous across  $F_h$ .

The above approach to crack modelling fits the classical finite element framework. The partition of unity property of finite elements can however be used so as



**Figure 7.3:** Mesh with "double nodes": uncracked solid and mesh achieving displacement continuity (top), portion of solid containing a crack  $F$  and mesh achieving displacement discontinuity across  $F$  (bottom). In the connectivity tables of the respective meshes, elements (a) and (b) are defined by the lists of nodes  $\{17, 18, 4, 3\}$  and  $\{11, 10, 18, 17\}$  (uncracked solid) or  $\{17, 18, 4, 3\}$  and  $\{11, 10, 20, 19\}$  (cracked solid).

to enrich the approximation space Babuška and Melenk (1997). The extended finite element method (XFEM) introduced by Moës et al. (1999) implements this alternative framework by introducing displacement discontinuities and known crack front singularities through suitable enrichment of FEM approximation spaces.

**Convergence, mesh refinement near the crack tips.** The singular character of the strain and stress fields under the LEFM assumption alters (and in fact deteriorates) the rate of convergence of the approximate solution towards the exact solution as the characteristic element size  $h$  approaches zero. For an elastic solid without cracks, one has (Section 3.5):

$$\|\mathbf{u} - \mathbf{u}_h\|_E = O(h^p) \quad (\text{with } \mathbf{u} \text{ assumed to have } H^{p+1}(\Omega) \text{ regularity}), \quad (7.14)$$

where  $p \geq 1$  denotes the largest degree of polynomials in  $\mathbf{x}$  that can be exactly reproduced by the interpolation functions and  $\|\cdot\|_E$  is the energy norm defined by (3.42), with the notations of Section 3.5. In contrast, when considering a cracked solid,  $\varepsilon[\mathbf{u}]$  is square-integrable in a neighbourhood of the crack tip (i.e. the strain energy of  $\mathbf{u}$  is finite despite the crack tip singularity) but higher-order derivatives of  $\mathbf{u}$  are not. The error estimate (7.14) therefore does not hold in this case, and one has the weaker estimate

$$\|\mathbf{u} - \mathbf{u}_h\|_E = O(h^{1/2-\eta}) \quad (\eta > 0) \quad (7.15)$$

where  $\eta$  is arbitrarily small but strictly positive. As a consequence, modelling a cracked solid and an uncracked solid using the same mesh (up to the necessary introduction of double nodes on  $F$  in the former case), the approximate solutions for the cracked solid is less accurate, in particular in the vicinity of the crack tip. Obtaining satisfactorily accurate solutions for fracture mechanics problems using the finite element method hence entails specific treatments, such as:

- (a) Highly refined meshes in the neighbourhood of the crack tip, so as to improve the representation of fields having singularities of the form (7.9a-c) by the usual finite element approximation space;
- (b) Designing new finite elements that allow to represent strain fields having  $1/\sqrt{r}$  singularities at crack tips.

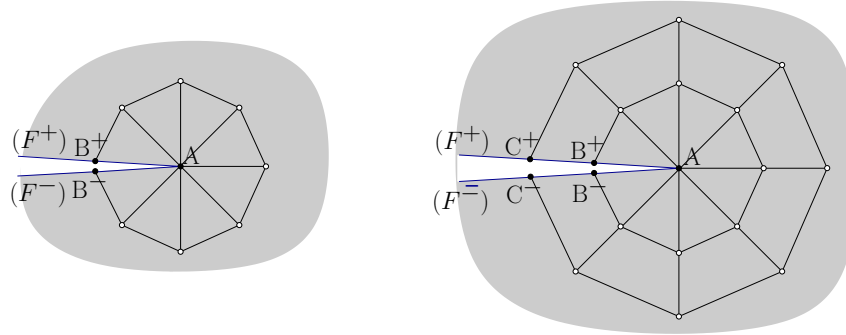
### 7.3.2 Evaluation of stress intensity factors using extrapolation

This approach, while not very accurate, is simple to implement. It consists of exploiting directly the displacements at nodes of  $F$  that are closest to a crack tip, comparing their value to the asymptotic behavior (7.10) of the crack opening displacement across the crack.

For instance (with notations as in Figure 7.4), choosing on the crack faces  $F_h^\pm$  the nodes  $B^+$  and  $B^-$  that are closest to the tip  $A$ , the asymptotic formula (7.10) suggests to write

$$K_I \approx \frac{E}{8(1-\nu^2)} \sqrt{\frac{2\pi}{d}} [\![\mathbf{u}(B)]\!] \cdot \mathbf{n}, \quad K_{II} \approx \frac{E}{8(1-\nu^2)} \sqrt{\frac{2\pi}{d}} [\![\mathbf{u}(B)]\!] \cdot \mathbf{t}, \quad (7.16)$$

where  $\mathbf{t}$  and  $\mathbf{n}$  respectively denote the tangent and normal directions to  $F$  at  $A$ ,  $B$  is the common geometrical locus of  $B^+$  and  $B^-$ , and  $d$  is the distance of  $B$  to  $A$ . The practical application of formulas (7.16) necessitates highly refined meshes near the crack tip (see Section 7.3.3 for an example).



**Figure 7.4:** Notation for the evaluation of  $K_I$ ,  $K_{II}$  using extrapolation at the double node  $(B^+, B^-)$  closest to the crack tip (left) or at several successive double nodes  $(B^+, B^-)$ ,  $(C^+, C^-)$ , ... (right).

A variation on this approach consists in using several double nodes taken along  $F_h$  in direct succession from a crack tip, and considering an approximation of the crack opening displacement  $[[\hat{\mathbf{u}}]]$  of the form

$$[[\hat{\mathbf{u}}]] = \sqrt{r}[\alpha \mathbf{n} + \beta \mathbf{t}].$$

The coefficients  $\alpha, \beta$  are then determined so that the function  $[[\hat{\mathbf{u}}]]$  evaluated at the selected double nodes is closest in the least squares sense to the corresponding nodal values. The resulting approximate values of the stress intensity factors  $K_I, K_{II}$  are then obtained by identification with the leading  $O(\sqrt{r})$  term of (7.10):

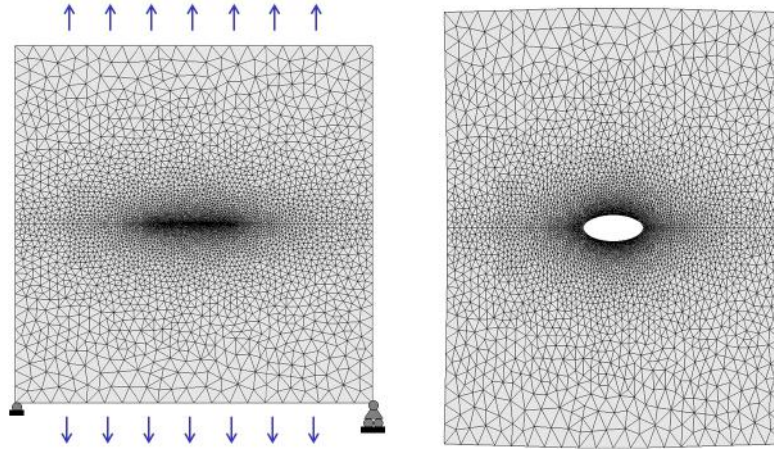
$$K_I \approx \frac{\sqrt{2\pi}E}{8(1-\nu^2)}\alpha \quad K_{II} \approx \frac{\sqrt{2\pi}E}{8(1-\nu^2)}\beta. \quad (7.17)$$

### 7.3.3 Example: analysis of a cracked solid using T3 elements and extrapolation

In this example, a rectangular solid of width  $2H$  and height  $2V$  is considered, under the plane strain assumption (Figure 7.5). The rectangle is aligned with a Cartesian frame  $(O, x_1, x_2)$  and centered at its origin  $O$ . The elastic solid contains a straight crack, described by the line segment  $F = \{-a \leq x_1 \leq a, x_2 = 0\}$ . A uniform tensile load density  $\sigma$  is applied along the  $x_2$  direction  $x_2$  on the edges  $x_2 = \pm V$ , causing a pure mode I crack opening.

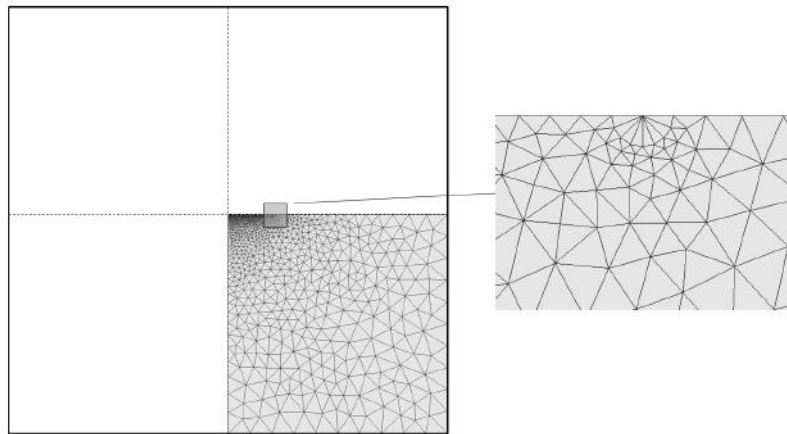
An exact solution for  $K_I$  is available when the solid has finite width  $2H$  but arbitrary large height  $V \rightarrow \infty$ :

$$K_I^H = \sigma \sqrt{\pi a} \left( \cos \frac{\pi a}{2H} \right)^{-1/2} \quad (7.18)$$



**Figure 7.5:** Square cracked plate: undeformed mesh Chap7\_fract.geo (left) and deformed mesh under the tensile load (right).

A procedure for creating a mesh with double nodes, suitable for modelling a cracked plate, is presented in Appendix B. Meshes that are *structured* near the crack tips, such as that of Figure 7.6, yield more accurate results for fracture-related quantities. In such meshes, crack tips are surrounded by elements of identical shape that emanate from them. Such meshes facilitate the application of extrapolation approaches for the evaluation of stress intensity factors.



**Figure 7.6:** Square cracked plate: mesh Chap7\_fract\_ros.geo

Finite element analyses are performed for three plate geometries with increasing heights  $V_1 = 5$ ,  $V_2 = 10$ ,  $V_3 = 20$ ; the other relevant dimensions used for this example are  $H = 5$ ,  $a = 1$ . Numerical results for  $K_I$  using extrapolation will be compared to the analytical value (7.18) for a plate of infinite height, and their convergence as  $V$  increases examined.

The computational procedure implemented in the provided MATLAB code is as follows. First, the approximate elastic solution for the cracked plate is computed using the module `genlin_fast`, with T3 elements and the data as defined in the file `Chap7_fract_ros`. This analysis yields in particular the nodal displacements on both crack faces. The double node closest to a given tip is then sought, and the normal displacement jump  $\llbracket u_2 \rrbracket$  evaluated at that location. The stress intensity factor  $K_I$  is finally evaluated using the extrapolation formula (7.16).

```
a=1.; % crack semilength
H=5.; % plate halfwidth
coor_K=0;
node_K=0;
for i=1:analysis.NN,
    coor=nodes(i).coor(:);
    if (abs(coor(2))<1.d-6)& ... % finds nodes on x_2=0 ...
        (coor(1)<(a-1.d-5))&(coor(1)>coor_K)
        node_K=i;
        coor_K=coor(1);
    end
end

delta_disp=2*nodes(node_K).U(2);
d=a-coor_K;
K_H=sqrt(pi*a)* ... % analyt. K for V->infty
    sqrt(sec(pi*a/(2*H)))
K_disp=material(1,1)/(8*(1-material(1,2)^2))... % numerical K from displ.
    *sqrt(2*pi/d)*abs(delta_disp)
```

This procedure is organised as a post-processing of the displacement solution at the mesh nodes, performed by the module `post_fract` which is called after completion of `genlin_fast`. The crack tip  $A$  considered in this example is located at  $x_1 = a, x_2 = 0$ . A loop over all the nodes is launched and, among the nodes located on the line  $x_2 = 0$  at the left of  $A$ , the one closest to the crack tip is stored in `node_K` with the corresponding nodal coordinate  $x_1$  saved in `coor_K`. The coordinate  $x_1$  of the node closest to  $A$  is sought, and the displacement jump at that (double) node evaluated.

Numerical results obtained using data in `fract_ros` and mesh refinement parameters as defined in Appendix B are compared to the analytical solution (7.18) in the following table:

$K_I^H$	$K_I^{V_1}$	$K_I^{V_2}$	$K_I^{V_3}$
1.8175	1.6985	1.6503	1.6529

As expected, the comparison reveals unsatisfactory accuracy due to the inability of T3 elements to represent the square-root behavior of the displacement field near the crack tip. However, on seeing that  $K_I^{V_2}$  and  $K_I^{V_3}$  are almost equal, one may also conclude that the third plate reasonably approximates the the plate of finite width and infinite height assumed for the analytical value  $K_I^H$ . The perturbation (relative to a uniform-strain solution for the uncracked solid under the same tensile load) induced by the presence of the crack is seen to decrease rapidly away from the crack tip.

### 7.3.4 Special-purpose finite elements

The extrapolation technique is simple to implement; in particular it works with standard types of finite elements (such as the isoparametric elements introduced in Chapter 2). On the other hand, reasonably accurate analysis results require meshes to be very fine (i.e. made of elements of very small size  $h$ ) in regions containing crack tips. This requirement results from the fact that the approximation spaces associated with standard finite elements do not adequately represent displacement fields that lead to strain or stress fields that are singular at crack tips.

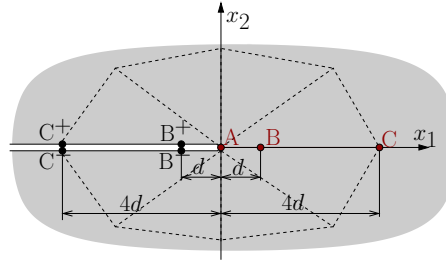
One way of addressing this shortcoming consists in defining new, special-purpose finite elements that are specifically designed for the representation of such fields. Such elements can be defined in many ways. Attention will here be restricted to the so-called *quarter-node element*, initially proposed by Barsoum (1976), as this approach is quite simple to formulate and implement, with only minimal modifications compared to the case of standard finite elements.

**One-dimensional interpolation with quarter-node element.** In its simplest form, the quarter-node interpolation involves just one space dimension. Consider three nodes  $\{A, B, C\}$  that are aligned along the direction of a straight crack (Fig. 7.7), with  $A$  located at the crack tip. The line segment  $\Gamma = [A, C]$  corresponds to one edge of a finite element based on shape functions of degree 2 with respect to each parametric coordinate. For the two-dimensional problems considered in this chapter, that element is therefore either triangular (six-noded) or quadrangular (eight- or nine-noded), as described in Table 2.2). The traces on  $\Gamma$  of the shape functions associated to the nodes  $\{A, B, C\}$  are then, for any such adjacent element, the Lagrange interpolation polynomials of degree 2 :

$$N_A(a) = a(a - 1)/2, \quad N_B(a) = 1 - a^2, \quad N_C(a) = a(a + 1)/2, \quad (7.19)$$

where  $a \in [-1, 1]$  is the parametric coordinate allowing a parametric representation of  $\Gamma$ , with nodes  $(A, B, C)$  respectively corresponding to  $a = (-1, 0, 1)$ .

The interpolation with quarter-node element simply consists in choosing as the middle node  $B$  the point of  $\Gamma$  such that  $AC = 4AB$  (i.e situated at one-quarter the length of  $\Gamma$  from the crack tip) and using the standard quadratic interpolation (7.19). Choosing  $A$  as the coordinate origin, the nodal abscissae are thus



**Figure 7.7:** One-dimensional interpolation using three nodes  $(\{A, B, C\})$  on  $\{A, B^\pm, C^\pm\}$ , with the quarter node located at  $B$  (or  $B^\pm$ ).



$x_A = 0, x_B = d, x_C = 4d$ , with  $d = AB = AC/4$ . The physical coordinate  $x_1$  of any point of  $\Gamma$  is then linked to the parametric coordinate  $a$  by (2.13), i.e.:

$$\begin{aligned} x_1 &= N_A(a)x_A + N_B(a)x_B + N_C(a)x_C \\ &= 0 + d(1 - a^2) + 2da(a + 1) \\ &= d(a + 1)^2, \end{aligned}$$

i.e.

$$a = \sqrt{x_1/d} - 1 \quad (0 \leq x_1 \leq 4d). \quad (7.20)$$

Then, interpolating the nodal values  $\mathbf{u}_A$ ,  $\mathbf{u}_B$  and  $\mathbf{u}_C$  at nodes of  $\Gamma$  yields the following interpolated displacement:

$$\begin{aligned} \mathbf{u}_h &= N_A(a)\mathbf{u}_A + N_B(a)\mathbf{u}_B + N_C(a)\mathbf{u}_C \\ &= \frac{1}{2}(\mathbf{u}_A - 2\mathbf{u}_B + \mathbf{u}_C)a^2 + \frac{1}{2}(\mathbf{u}_C - \mathbf{u}_A)a + \mathbf{u}_B. \end{aligned}$$

Upon expressing  $\mathbf{u}_h$  as a function of  $x_1$  with the help of (7.20), one obtains

$$\begin{aligned} \mathbf{u}_h(x_1) &= \mathbf{u}_A + \left[ 2(\mathbf{u}_B - \mathbf{u}_A) - \frac{1}{2}(\mathbf{u}_C - \mathbf{u}_A) \right] \sqrt{\frac{x_1}{d}} \\ &\quad + \left[ \frac{1}{2}(\mathbf{u}_C - \mathbf{u}_A) - (\mathbf{u}_B - \mathbf{u}_A) \right] \frac{x_1}{d}. \end{aligned} \quad (7.21)$$

Selecting the node  $B$  as the point located at one-quarter of the length of  $\Gamma$  thus has the following remarkable consequence:

- *The gradient of the corresponding interpolated displacement (7.21) has a  $1/\sqrt{x_1}$  singularity at the crack tip.*

A similar analysis applied to the case of two sets of aligned nodes  $\{A, B^\pm, C^\pm\}$  located on the adjacent crack faces  $F^\pm$  yields the quarter-node interpolation of the crack opening displacement along a line segment adjacent to a crack tip:

$$[\![\mathbf{u}_h]\!](x_1) = \left( 2[\![\mathbf{u}_B]\!] - \frac{1}{2}[\![\mathbf{u}_C]\!] \right) \sqrt{\frac{x_1}{d}} + \left( \frac{1}{2}[\![\mathbf{u}_C]\!] - [\![\mathbf{u}_B]\!] \right) \frac{x_1}{d} \quad (7.22)$$

(with of course  $[\![\mathbf{u}_A]\!] = \mathbf{0}$  at the crack tip).

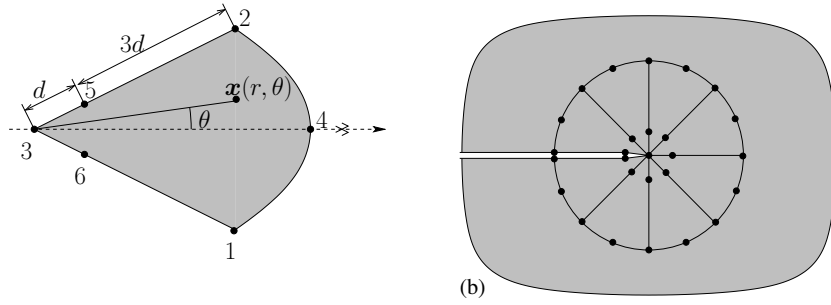
The numerical evaluation of  $K_I, K_{II}$  is then based on a comparison of the coefficient of  $\sqrt{r}$  in (7.21) or (7.22) to the asymptotic expression of  $\mathbf{u}$  or  $[\![\mathbf{u}]\!]$  for small  $r$ . For example, comparing (7.22) and (7.10) yields

$$\begin{aligned} K_I &\approx \frac{E\sqrt{2\pi}}{8(1-\nu^2)} \left[ 2[\![\mathbf{u}_B]\!] - \frac{1}{2}[\![\mathbf{u}_C]\!] \right] \cdot \mathbf{n} , \\ K_{II} &\approx \frac{E\sqrt{2\pi}}{8(1-\nu^2)} \left[ 2[\![\mathbf{u}_B]\!] - \frac{1}{2}[\![\mathbf{u}_C]\!] \right] \cdot \mathbf{t} . \end{aligned} \quad (7.23)$$

**Multi-dimensional interpolation with quarter-node element.** The concept of quarter-node element has been generalised to many two- and three-dimensional finite elements. For instance, Freese and Tracey (1976) have shown that triangular elements whose six nodes *in the physical space* are arranged according to Figure 7.8a are such that the interpolated displacement based on the usual shape functions (given in Table 2.2, page 42) has the form

$$\mathbf{u}_h(\mathbf{x}) = \mathbf{u}_0(\theta) + \mathbf{u}_1(\theta)\sqrt{r} + \mathbf{u}_2(\theta)r$$

along a straight line emanating from node 1 with inclination  $\theta$  (Fig. 7.8a). Such elements are then arranged so as to fill a disk centered at the crack tip (Fig. 7.8b). The six-noded quarter-node triangular element is described next in more detail.



**Figure 7.8:** Six-noded triangular element: (a) spatial arrangement of the nodes in physical space allowing the isoparametric interpolation to represent  $O(1/\sqrt{r})$  strain singularities; (b) typical arrangement of such elements around a crack tip.

### 7.3.5 Example: analysis of a cracked solid using T6 elements

**T6-QP quarter-node element.** Consider the element of Figure 7.9. The coordinate origin coincides with  $\mathbf{x}^{(3)}$ , and the  $x_1$ -direction is chosen aligned with the element edge with endpoints  $\mathbf{x}^{(3)}$  and  $\mathbf{x}^{(1)}$ . Moreover, the nodal locations obey the following relations:

$$\begin{aligned} \mathbf{x}^{(6)} &= \frac{1}{4}\mathbf{x}^{(1)}, & \mathbf{x}^{(4)} &= \frac{1}{2}(\mathbf{x}^{(1)} + \mathbf{x}^{(2)}), & \mathbf{x}^{(5)} &= \frac{1}{4}\mathbf{x}^{(2)}, \\ \mathbf{x}^{(1)} &= b_1\mathbf{e}_1, & \mathbf{x}^{(2)} &= b_2\mathbf{e}_1 + c_2\mathbf{e}_2 \end{aligned}$$

Using the shape functions of the T6 element (Table 2.2, page 42), the approximate displacement field along the dashed line of Figure 7.9, defined by  $a_2 = \alpha a_1$  in terms of the parametric coordinates of the reference element (with  $\alpha \geq 0$ ), is given by

$$\begin{aligned} x_1 &= b_1 a_1 (2a_1 - 1) + 2(b_1 + b_2)a_1 a_2 + b_2 a_2 (2a_2 - 1) + (b_2 a_2 + b_1 a_1)(1 - a_1 - a_2) \\ &= a_1^2(1 + \alpha)(b_1 + \alpha b_2) \end{aligned} \quad (7.24)$$

It is important to note that the above expression does not involve any linear term in  $a_1$ , a feature which plays a key role towards representing a square-root behavior near  $\mathbf{x}^{(3)}$ . Equivalently, one has

$$x_2 = a_1^2(1 + \alpha)\alpha c_2 \quad (7.25)$$

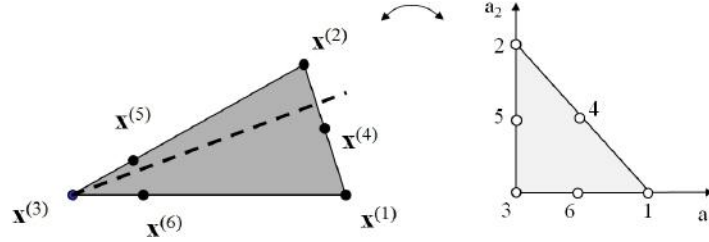


Figure 7.9: Special (quarter-node) T6 element

The important implication of equations (7.24) and (7.25) is that

$$r = a_1^2 c \quad \rightarrow \quad a_1 = \sqrt{r/c},$$

where  $r$  denotes the distance (in physical space) of a point of the element to the crack tip  $\mathbf{x}^{(3)}$  and the coefficient  $c$  depends only of the triangle geometry.

One is now in a position to analyse the interpolation  $v_h(\mathbf{x})$  of a displacement field. Since the interpolation is quadratic, one must have:

$$v_h(\mathbf{x}) = v^{(3)} + p a_1 + q a_1^2$$

where  $v^{(3)}$  is the displacement at node  $\mathbf{x}^{(3)}$ , where the coefficients  $p, q$  depend on the nodal displacements. This implies

$$v_h(\mathbf{x}) = v^{(3)} + p \sqrt{r/c} + q(r/c).$$

The asymptotic behavior of the interpolated displacement near the crack tip is thus identical with that predicted by the LEFM theory.

**Remark.** The above results remain valid for other configurations of the triangular element. The perhaps most often used configuration is such that nodes  $\mathbf{x}^{(1)}, \mathbf{x}^{(4)}, \mathbf{x}^{(2)}$  are arranged on a circle of radius  $d$  (with respective angular coordinates  $\theta = 0, \beta, 2\beta$ ) and nodes  $\mathbf{x}^{(6)}, \mathbf{x}^{(5)}$  on a circle of radius  $d/4$  (with  $\theta = 0$  and  $2\beta$ , respectively), with both circles centered at  $\mathbf{x}^{(3)}$ .

**Numerical results.** The comparison performed in Section 7.3.3 is revisited by using the previously-defined T6-QP elements as a basis for extrapolating  $K_I$  from the computed displacement field. The mesh used is as defined in the file `fract_ros.geo`.

Numerical values of  $K_I$  for the plate  $V_3$ , computed using either the standard T6 element or its quarter-node version, are shown in Table 7.1 together with the exact value of  $K_I$  corresponding to (7.18). The mesh created from the data of file `fract_ros.geo` is made of standard T6 elements only (i.e. all nodes with local numbering 4, 5, 6 are the mid-point of the edge they belong to). The user thus needs to shift the position of the mid-nodes of all edges emanating from a crack tip to the required quarter-point location. This preliminary alteration must obviously be done (in the module `genlinfast`) prior to setting up the stiffness matrix of the structure, for example by using the code provided in `pre_fract`. During a loop over all the nodes the code identifies all the

nodes having a distance from the crack tip smaller than the rosette radius and shifts them to the required quarter-node position:

```

a=1.d0;                                % crack semilength
rad=0.02;                               % radius of rosette

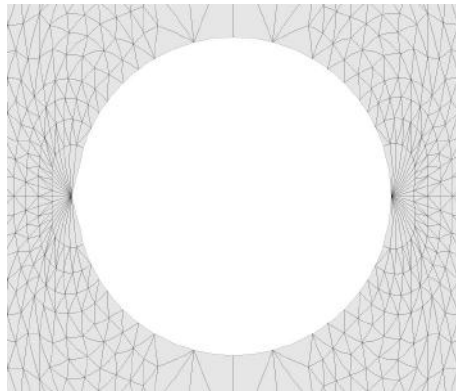
for i=1:analysis.NN,
    dist=sqrt(nodes(i).coor(2)^2+ ...
              (nodes(i).coor(1)-a)^2); % distance of node from x_B
    if dist<.6*rad
        nodes(i).coor(2)=.5*nodes(i).coor(2); % modifies x_1
        nodes(i).coor(1)=a+.5*(nodes(i).coor(1)-a); % modifies x_2
    end
end
end

```

(i) $K_I^H$	(ii) $K_I^{V_3}$	(iii) $K_I^{V_3-QP}$
1.8175	1.7078	1.8172

**Table 7.1:** Cracked plate, evaluation of  $K_I$ : (i) exact; (ii) numerical, with standard T6 elements; (iii) numerical, with quarter-node T6-QP elements.

Figure 7.10 shows a blow-up of the deformed mesh in the vicinity of the crack tip (obtained using a visually suitable magnification of the computed displacement and slight modifications in the available codes). The elements surrounding the crack tip are standard T6 triangles for the left crack tip  $x_1 = -a$ , and quarter-node T6-QP triangles for the right crack tip  $x_1 = a$ . The choice of element is seen to substantially affect the displacement solution close to the crack tip; for instance, the vertical slope at the right crack tip is consistent with the expected square-root behavior while the discontinuous, not vertical, slopes at the left crack tip is not. Once the analysis performed using `genlin_fast` is performed,  $K_I$  is again estimated using the method previously used in conjunction with the T3 element, coded in the post-processing module `post_fract`.



**Figure 7.10:** Crack opening displacement, using either standard T6 elements (left) or quarter-node T6-QP elements (right).

## 7.4 Numerical computation of energy release rates

The concept of stress intensity factor is intrinsically linked to the LEFM framework, whereas that of energy release rate is susceptible of generalization. Energy-based fracture criteria are also more appealing in terms of physical intuition. It is therefore useful to formulate computational strategies for evaluating energy release rates without explicit reference or reliance to the local properties of the solution such as strain or stress crack tip singularities. The main idea consists in going back to the primary definition (7.5) of  $G$  as the derivative of the potential energy reached at equilibrium in a virtual crack extension. This approach has originally been proposed by Destuynder, Djaoua, and Lescure (1983).

To avoid lengthy and technical developments, the idea is presented here only for the simplest situation where  $G$  is to be evaluated for both tips of a straight crack  $F = F_{a,b} = [A, B] = \{-a \leq x_1 \leq b, x_2 = 0\}$ , so that one has

$$G_A = -\frac{\partial P}{\partial a}(a, b, \mathbf{u}^D, \mathbf{T}^D), \quad G_B = -\frac{\partial P}{\partial b}(a, b, \mathbf{u}^D, \mathbf{T}^D)$$

(i.e. equations (7.7), repeated here for convenience), the potential energy at equilibrium  $P(a, b, \mathbf{u}^D, \mathbf{T}^D)$  being given by (7.4). Therein, the displacement  $\mathbf{u}$  of the elastic equilibrium state for the chosen crack configuration and loading is governed by the weak formulation (1.22), which in the present case reads

$$\text{find } \mathbf{u} \in \mathcal{C}(\mathbf{u}^D) \text{ such that } \int_{\Omega(F)} \varepsilon[\mathbf{u}] : \mathcal{A} : \varepsilon[\mathbf{w}] \, dV = \int_{S_T} \mathbf{T}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}) \quad (7.26)$$

The restricted setting adopted here notwithstanding, the methodology for computing  $G$  presented next remains valid for more complex configurations.

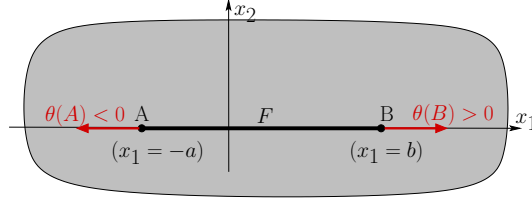
### 7.4.1 Representation of crack extensions by means of domain transformations

The formulation and evaluation of the derivatives involved in the definition of  $G_A, G_B$  is not straightforward. The potential energy  $P(a, b, \mathbf{u}^D, \mathbf{T}^D)$  depends on  $a, b$  (or, more generally, on the crack configuration  $F$ ) in two ways:

- (a) Explicit dependency: the domain  $\Omega$  in (7.4) or (7.26) depends on  $F = [-a, b]$ ;
- (b) Implicit dependency: the displacement solution  $\mathbf{u}$  of (7.26) depends on  $F = [-a, b]$ .

To differentiate  $P(a, b, \mathbf{u}^D, \mathbf{T}^D)$  with respect to the crack tip positions, one needs to consider perturbations of the crack configuration induced by variations of the crack tip abscissae  $-a$  and  $b$ . To take into account the effect of crack perturbations on  $P(a, b, \mathbf{u}^D, \mathbf{T}^D)$ , which is a global quantity as it is expressed as an integral over the structure, a fictitious transformation of the whole domain  $\Omega$  is introduced. It is defined in Lagrangian fashion by

$$\mathbf{y} = \Phi(\mathbf{x}, \tau) = \mathbf{x} + \tau \boldsymbol{\theta}(\mathbf{x}) \quad \mathbf{x} \in \Omega(F), \quad (7.27)$$



**Figure 7.11:** Transformation velocity for a straight crack.

where  $\tau$  is a fictitious time (with  $\tau = 0$  corresponding to the unperturbed configuration of  $\Omega$  on which  $G$  is to be evaluated) and  $\theta(\mathbf{x})$  is a *initial transformation velocity*. The relation (7.27) is thus the Lagrangian description of a transformation of  $\Omega(F)$  such that a point initially located at  $\mathbf{x} \in \Omega$  moves at any time  $\tau \geq 0$  with a constant velocity  $\theta(\mathbf{x})$ . Moreover, letting  $A$  and  $B$  denote the crack tip locations for the unperturbed crack, the transformation velocity  $\theta(\mathbf{x})$  must be chosen so that the transformation satisfies the following requirements:

- (a) The external boundary of the cracked solid does not change under the transformation;
- (b) The shape (here, straight) of the pre-existing part of the crack is left unaltered by the transformation;
- (c) The transformation must represent a crack *extension* (Fig. 7.11);

and hence is subject to the requirements

$$\begin{aligned} \theta(\mathbf{x}) &= \mathbf{0} \quad (\mathbf{x} \in S_u \cup S_T) & (a), \\ \theta(\mathbf{x}) \cdot \mathbf{e}_2 &= 0 \quad (\mathbf{x} \in F) & (b), \\ \dot{a} = -\theta(A) \cdot \mathbf{e}_1 &\geq 0 \quad \text{et} \quad \dot{b} = \theta(B) \cdot \mathbf{e}_1 \geq 0 & (c). \end{aligned} \quad (7.28)$$

The virtual crack extension under the transformation, characterized by the tangent transformation velocities at the crack tip, is thus defined by

$$a(\tau) = a - \theta_1(A)\tau, \quad b(\tau) = b + \theta_1(B)\tau.$$

#### 7.4.2 The $G$ -theta method

**Lagrangian differentiation of potential energy: the  $G$ -theta method.** The  $G - \theta$  method consists in evaluating  $G$  by differentiating  $P(a(\tau), b(\tau), \mathbf{u}^D, \mathbf{T}^D)$  with respect to the fictitious time  $\tau$ . The derivative is taken for  $\tau = 0$ , with the boundary data  $(\mathbf{u}^D, \mathbf{T}^D)$  kept fixed (i.e. independent on  $\tau$ ). This approach exploits the concept of material (i.e. Lagrangian) derivative and is based on the following classical results of continuum kinematics (Malvern, 1969; Salençon, 2001):

- Initial Lagrangian derivative  $\dot{g}$  of a scalar or tensor field  $g(\mathbf{y}, \tau)$ :

$$\dot{g}(\mathbf{x}) = \lim_{\tau \rightarrow 0} \frac{1}{\tau} [g(\mathbf{x} + \tau\theta(\mathbf{x}), \tau) - g(\mathbf{x}, 0)] = \frac{\partial g}{\partial \tau}(\mathbf{x}, 0) + \nabla g(\mathbf{x}, 0) \cdot \theta(\mathbf{x}); \quad (7.29)$$

- Initial Lagrangian derivative of the gradient of a scalar or tensor field  $g(\mathbf{y}, \tau)$  (easily deduced from (7.29)):

$$\overbrace{(\nabla \mathbf{w})}(\mathbf{x}) = \nabla \dot{\mathbf{w}}(\mathbf{x}) - \nabla \mathbf{w}(\mathbf{x}, 0) \cdot \nabla \boldsymbol{\theta}(\mathbf{x}); \quad (7.30)$$

- Lagrangian form of the initial derivative of a domain integral ( $\Omega(\tau) := \Phi(\Omega, \tau)$  denoting the image of  $\Omega = \Omega(0)$  at "time"  $\tau$  under the transformation (7.27)):

$$I(\tau) = \int_{\Omega(\tau)} g(\mathbf{x}, \tau) dV_x \quad \frac{dI}{d\tau} \Big|_{\tau=0} = \int_{\Omega} \{ \dot{g}(\mathbf{x}) + g(\mathbf{x}) \operatorname{div} \boldsymbol{\theta}(\mathbf{x}) \} dV_x. \quad (7.31)$$

Since the loading is assumed to be kept fixed, consistently with the definition (7.5) of  $G$ , as the domain is modified under the transformation, the potential energy at equilibrium depends on  $\tau$  only through  $a(\tau)$  and  $b(\tau)$ . Its Lagrangian derivative thus depends only on the transformation velocities at the crack tips:

$$\begin{aligned} \dot{P} &= \frac{d}{d\tau} P(a(\tau), b(\tau), \mathbf{u}^D, \mathbf{T}^D) \Big|_{\tau=0} \\ &= \frac{\partial P}{\partial a} \dot{a} + \frac{\partial P}{\partial b} \dot{b} = -\frac{\partial P}{\partial a} \theta_1(A) + \frac{\partial P}{\partial b} \theta_1(B) \end{aligned} \quad (7.32)$$

The main ingredient on which the  $G - \theta$  method is based is thus the computation of the Lagrangian derivative of  $P$  in a transformation (7.27) whose transformation velocity satisfies requirements (7.28). The energy release rates  $G_A, G_B$  are then identified by comparing (7.32) with (7.7), i.e. by exploiting the relationship

$$\frac{dP}{d\tau} \Big|_{\tau=0} = G_A \theta_1(A) - G_B \theta_1(B). \quad (7.33)$$

**Computation of the Lagrangian derivative of the potential energy.** By exploiting identities (7.29), (7.30) and (7.31) and the symmetry properties (1.9) of the elasticity tensor, the following identity, which holds for any displacement field  $\mathbf{v}$ , is readily established:

$$\frac{d}{d\tau} \frac{1}{2} \int_{\Omega(\tau)} \boldsymbol{\varepsilon}[\mathbf{v}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{v}] dV \Big|_{\tau=0} = \int_{\Omega} \boldsymbol{\varepsilon}[\dot{\mathbf{v}}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{v}] dV + \mathcal{G}(\mathbf{v}; \boldsymbol{\theta}), \quad (7.34)$$

with

$$\mathcal{G}(\mathbf{v}; \boldsymbol{\theta}) = \int_{\Omega} \left\{ \frac{1}{2} (\nabla \mathbf{v} : \boldsymbol{\mathcal{A}} : \nabla \mathbf{v}) \operatorname{div} \boldsymbol{\theta} - \nabla \mathbf{v} : \boldsymbol{\mathcal{A}} : (\nabla \mathbf{v} \cdot \nabla \boldsymbol{\theta}) \right\} dV. \quad (7.35)$$

With the help of this identity, the Lagrangian derivative of the potential energy (7.4) is found to be given by

$$\begin{aligned} \frac{d}{d\tau} P(a(\tau), b(\tau), \mathbf{u}^D, \mathbf{T}^D) \Big|_{\tau=0} \\ = \int_{\Omega} \boldsymbol{\varepsilon}[\dot{\mathbf{u}}] : \boldsymbol{\mathcal{A}} : \boldsymbol{\varepsilon}[\mathbf{u}] dV - \int_{S_T} \mathbf{T}^D \cdot \dot{\mathbf{u}} dS + \mathcal{G}(\mathbf{u}; \boldsymbol{\theta}) \end{aligned} \quad (7.36)$$

where the "time"-independent character  $\dot{T}^{(D)} = \mathbf{0}$  of the loading assumed in the definition of  $G$  has been taken into account. The above result involves the Lagrangian derivative  $\dot{\mathbf{u}}$  of the displacement solution for the reference cracked configuration. An evaluation of (7.36) therefore seems to require that  $\dot{\mathbf{u}}$  be computed beforehand. It turns out that the latter task can be avoided by means of the following argument. Indeed, since the definition of  $G$  also stipulates that the prescribed displacement does not depend on  $\tau$ , one has

$$\dot{\mathbf{u}} = \mathbf{0} \text{ on } S_u, \text{ i.e. } \dot{\mathbf{u}} \in \mathcal{C}(\mathbf{0}).$$

Consequently, the weak formulation (7.26) may be applied with the specific virtual field  $\mathbf{v} = \dot{\mathbf{u}}$ . The identity thus obtained allows to eliminate the two integrals containing  $\dot{\mathbf{u}}$  from (7.36). The derivative of the potential energy is thus formulated, in terms of the equilibrium solution  $\mathbf{u}$  for the reference configuration and the transformation velocity, as

$$\left. \frac{d}{d\tau} P(a(\tau), b(\tau), \mathbf{u}^D, \mathbf{T}^D) \right|_{\tau=0} = \frac{1}{2} \mathcal{G}(\mathbf{u}; \boldsymbol{\theta}) \quad (7.37)$$

with  $\mathcal{D}(\mathbf{u}; \boldsymbol{\theta})$  defined by (7.35).

**Evaluation of  $G_A$  and  $G_B$ .** In apparent contradiction of the fact that  $dP/d\tau$  was expected to depend only on the crack tip extension velocities  $\theta_1(A), \theta_1(B)$ , the value (7.37) of  $dP/d\tau$  apparently depends on the complete transformation velocity field  $\boldsymbol{\theta}$  in  $\Omega$ .

In fact, considering two transformation velocity fields  $\boldsymbol{\theta}'$  et  $\boldsymbol{\theta}''$  that verify conditions (7.28) and coincide at the crack tips (i.e.  $\boldsymbol{\theta}'(A) = \boldsymbol{\theta}''(A)$  and  $\boldsymbol{\theta}'(B) = \boldsymbol{\theta}''(B)$ ), one can in fact show (see footnote<sup>1</sup>) that formula (7.37) evaluated with either  $\boldsymbol{\theta} = \boldsymbol{\theta}'$  or  $\boldsymbol{\theta} = \boldsymbol{\theta}''$  yields the same value of  $dP/d\tau$ .

Consider now two transformation velocity fields  $\boldsymbol{\theta}_A$  and  $\boldsymbol{\theta}_B$ , such that

$$\begin{aligned} \boldsymbol{\theta}_A(A) &= -\mathbf{e}_1 & \boldsymbol{\theta}_A(B) &= \mathbf{0} & \boldsymbol{\theta}_A & \text{satisfies conditions (7.28),} \\ \boldsymbol{\theta}_B(A) &= \mathbf{0} & \boldsymbol{\theta}_B(B) &= \mathbf{e}_1 & \boldsymbol{\theta}_B & \text{satisfies conditions (7.28).} \end{aligned} \quad (7.38)$$

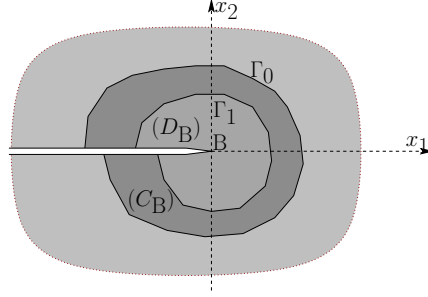
<sup>1</sup>A sketchy proof of this result is as follows. Letting  $(dP/d\tau)'$  and  $(dP/d\tau)''$  denote the respective values of  $dP/d\tau$  as given by (7.37), one has

$$\begin{aligned} \left( \frac{dP}{d\tau} \right)' - \left( \frac{dP}{d\tau} \right)'' &= \int_{\Omega} \left\{ \frac{1}{2} (\nabla \mathbf{u} : \boldsymbol{\mathcal{A}} : \nabla \mathbf{u}) \operatorname{div} (\boldsymbol{\theta}' - \boldsymbol{\theta}'') - \nabla \mathbf{u} : \boldsymbol{\mathcal{A}} : (\nabla \mathbf{u} \cdot \nabla (\boldsymbol{\theta}' - \boldsymbol{\theta}'')) \right\} dV \\ &= \int_{\partial\Omega} \left( \mathbf{T} \cdot \nabla \mathbf{u} \cdot (\boldsymbol{\theta}' - \boldsymbol{\theta}'') \right) - \frac{1}{2} (\nabla \mathbf{u} : \boldsymbol{\mathcal{A}} : \nabla \mathbf{u}) (\boldsymbol{\theta}' - \boldsymbol{\theta}'') \cdot \mathbf{n} \, ds = 0, \end{aligned}$$

where the second equality results from invoking identity (7.43) and the third exploits conditions (7.28).

One must notice the fact that identity (7.43) could not be directly applied to the expression (7.37) of  $dP/d\tau$  because of the  $1/r$  (integrable) singularity of the strain energy density  $(\nabla \mathbf{u} : \boldsymbol{\mathcal{A}} : \nabla \mathbf{u})/2$  at the crack tips: the divergence of the latter quantity then has a non-integrable singularity. In contrast, this approach becomes valid when applied, as above, to  $(dP/d\tau)' - (dP/d\tau)''$ , by virtue of the fact that the difference  $(\boldsymbol{\theta}' - \boldsymbol{\theta}'')$  vanishes at the crack tip, thus compensating the singular behaviour of the strain energy density.





**Figure 7.12:** Annulus  $C_B$  for the computation of  $G_B$  using (7.37); notation.

By virtue of (7.33) and (7.37), the energy release rates  $G_A$ ,  $G_B$  are then given by

$$G_A = -\frac{1}{2}\mathcal{G}(\mathbf{u}, \boldsymbol{\theta}_A), \quad G_B = -\frac{1}{2}\mathcal{G}(\mathbf{u}, \boldsymbol{\theta}_B). \quad (7.39)$$

The transformation velocity fields  $\boldsymbol{\theta}_A$  and  $\boldsymbol{\theta}_B$  can still be chosen and designed in many ways. Since the integral (7.35) defining  $\mathcal{G}(\mathbf{u}, \boldsymbol{\theta})$  involves  $\nabla \mathbf{u}$ , which is singular at the crack tips, it is useful to define velocities  $\boldsymbol{\theta}$  that allow to avoid the evaluation of integrals whose support contains a crack tip. This amounts to require that  $\boldsymbol{\theta}$  vanish in a neighbourhood of any crack tip.

A simple approach for fulfilling this criterion is as follows. Considering for definiteness the velocity field  $\boldsymbol{\theta}_B$ , let  $\Gamma_0$  and  $\Gamma_1$  denote two closed curves without self-intersection, such that  $\Gamma_0$  surrounds  $\Gamma_1$  and  $B$  is interior to  $\Gamma_1$  (Fig. 7.12). Let  $C_B$  be the open annulus-shaped region bounded by the two curves  $\Gamma_0$ ,  $\Gamma_1$  and the crack  $F$ , and  $D_B$  the region bounded by  $\Gamma_1$  and  $F$  ( $D$  is then a neighbourhood of  $B$  relative to  $\Omega$ ). The transformation velocity field  $\boldsymbol{\theta}_B$  is then defined by:

$$\boldsymbol{\theta}_B(\mathbf{x}) = \theta_B(\mathbf{x})\mathbf{e}_1 \quad \text{with} \quad \begin{cases} \theta_B(\mathbf{x}) = 1 & (\mathbf{x} \in D_B), \\ \theta_B(\mathbf{x}) = 0 & (\mathbf{x} \in \Omega \setminus (C_B \cup D_B)), \\ \theta_B(\mathbf{x}) \text{ continuous over } \Omega. \end{cases} \quad (7.40)$$

The definition of  $\theta_B$  in  $C_B$  is arbitrary, subject only to achieve a continuous transition between  $\theta_B = 0$  on  $\Gamma_0$  and  $\theta_B = 1$  on  $\Gamma_1$ . The function  $\theta_B$  may then be defined in  $C_B$  by interpolating nodal values in  $C_B$ . The latter are then set to 0 or 1, respectively, at the nodes of  $\Gamma_0$  and  $\Gamma_1$  and arbitrarily chosen at all internal nodes of  $C_B$ . The continuity of  $\theta_B$  thus defined is of course guaranteed by the fact that isoparametric interpolation is used. In practice, internal nodal values for  $\theta_B$  may be set using a variety of methods, including:

- In case  $C$  is a circular annulus bounded by concentric circles with radii  $r_0$  and  $r_1$ , affine interpolation:

$$\theta_B(\mathbf{x}) = \frac{r - r_1}{r_0 - r_1};$$

- Use finite elements for solving, on the same mesh, a Dirichlet problem for the Laplace equation:

$$\Delta \theta_B = 0 \text{ (in } C_B),$$

$$\theta_B = 0 \text{ (on } \Gamma_0), \quad \theta_B = 1 \text{ (on } \Gamma_1), \quad \frac{\partial \theta_B}{\partial x_2} = 0 \text{ (on } C_B \cap F^\pm).$$

Using this construction, the energy release rate  $G_B$  is then numerically evaluated by means of the formula

$$G_B = \int_{C_B} \left\{ (\nabla \mathbf{u} \cdot \nabla \theta_B) : \mathcal{A} : \nabla \mathbf{u} - \frac{1}{2} (\nabla \mathbf{u} : \mathcal{A} : \nabla \mathbf{u}) \operatorname{div} \theta_B \right\} dV, \quad (7.41)$$

whose result depends neither on the choice of annular region  $C_B$  surrounding  $B$  nor on the definition of  $\theta_B$  in  $C$ . Obviously, a velocity field  $\theta_A$  may be defined using the same approach, and  $G_A$  computed as an integral over an annular region  $C_A$ .

**Numerical implementation of the  $G - \theta$  method.** From a practical standpoint, the  $G - \theta$  method is a post-processing step applied to the elastic solution for the analysed cracked solid, which consists in an evaluation of the integral (7.41) based on the finite element mesh used for the analysis:

$$G_B \approx \sum_{E_e \in C_B} \int_{E_e} \left\{ (\nabla \mathbf{u}_h \cdot \nabla \theta_B) : \mathcal{A} : \nabla \mathbf{u}_h - \frac{1}{2} (\nabla \mathbf{u}_h : \mathcal{A} : \nabla \mathbf{u}_h) \operatorname{div} \theta_B \right\} dV \quad (7.42)$$

Each element integral is then computed using the Gauss points previously introduced for the evaluation of the element stiffness matrices (Section 3.2). This procedure is explained in more detail in Section 3.2.2 for the case of plane-strain analyses based on the T3 triangular element.

### 7.4.3 Expression of $G$ as an invariant contour integral: the $J$ integral.

Assuming the absence of body forces, any displacement field  $\mathbf{v}$  at equilibrium satisfies the identity

$$\begin{aligned} \nabla \mathbf{v} : \mathcal{A} : (\nabla \mathbf{v} \cdot \nabla \theta) - \frac{1}{2} (\nabla \mathbf{v} : \mathcal{A} : \nabla \mathbf{v}) \operatorname{div} \theta \\ = \operatorname{div} \left( \nabla \mathbf{v} : \mathcal{A} : (\nabla \mathbf{v} \cdot \theta) - \frac{1}{2} (\nabla \mathbf{v} : \mathcal{A} : \nabla \mathbf{v}) \theta \right). \end{aligned} \quad (7.43)$$

Upon exploiting it in expression (7.41) of  $G_B$ , the domain integral over the annulus  $C_B$  is converted into a contour integral by means of the divergence theorem. Since in addition  $\theta_B = \mathbf{0}$  on  $\Gamma_0$ ,  $\theta_B \cdot \mathbf{n} = 0$  on  $F^\pm$  and  $\theta_B = \mathbf{e}_1$  on  $\Gamma_1$ , one obtains

$$G_B = \int_{\Gamma_1} \left\{ \frac{\partial \mathbf{u}}{\partial x_1} \cdot \boldsymbol{\sigma} \cdot \mathbf{n} - \frac{1}{2} (\boldsymbol{\sigma} : \boldsymbol{\varepsilon}[\mathbf{u}]) n_1 \right\} ds, \quad (7.44)$$

where  $\sigma = \mathcal{A}:\varepsilon[u]$  is the stress tensor associated with the displacement solution  $u$ . The contour integral over  $\Gamma_1$  is independent on the choice of contour surrounding  $B$ : this invariant contour integral is known as the  $J$  integral of Rice.

One notes in passing that one of the available proofs of Irwin's formula (7.12) linking  $G$  to the stress intensity factors, consists in finding the limiting value of the invariant contour integral (7.44) as the contour  $\Gamma_1$  becomes arbitrarily close to  $B$  with the help of the asymptotic expressions (7.9a-c).

#### 7.4.4 Advantages of energy-based approaches

The computation of  $G$  using the  $G - \theta$  method is more involved to implement, and computationally more intensive, than local extrapolation approaches (irrespective of whether the latter use standard or special-purpose elements). This approach also has, however, significant conceptual and practical advantages:

- (a) The approach, being based on general principles involving energy conservation, lends itself to generalizations reaching far beyond the LEFM framework considered in this chapter. In contrast, the concept of stress intensity factor is intrinsic to linear elasticity. For example, an extension of the  $G - \theta$  approach that takes into account plastic energy dissipation has been proposed and implemented by Lorentz, Wadier, and Debruyne (2000).
- (b) The fact of having expressed  $G$  as an integral over an annular region that avoids the vicinity of the crack tip allows to sidestep the computational difficulties entailed by strain and stress singularities at the crack tips. This departs from the extrapolation approaches for the evaluation of stress intensity factors, which exploit the solution locally in the vicinity of the crack tips and are thus more prone to inaccuracy (or more demanding in terms of mesh refinement).

#### 7.4.5 Example: the $G - \theta$ method using T3 elements

The cracked plate of Figure 7.6 is again considered, this time to investigate the application of the  $G - \theta$  method to the evaluation of the energy release rate  $G_B$  associated with the right crack tip. Geometrical symmetry and loading conditions imply that  $K_{II} = 0$ . Irwin's formula (7.12) hence permit an evaluation of  $K_I$  from  $G$ :

$$K_{I,\theta} = \sqrt{\frac{E}{1-\nu^2}} G_B \quad (7.45)$$

The transformation velocity  $\theta_B$  is chosen so that its component normal to the crack vanishes:  $\theta_B = \theta_h e_1$ . Letting  $x_B$  denote the right crack tip, with coordinates  $(x_1, x_2) = (a, 0)$ ,  $\theta_B$  is a piecewise affine and continuous field defined in each element using interpolation of nodal values. The latter are chosen as follows (Fig. 7.13):

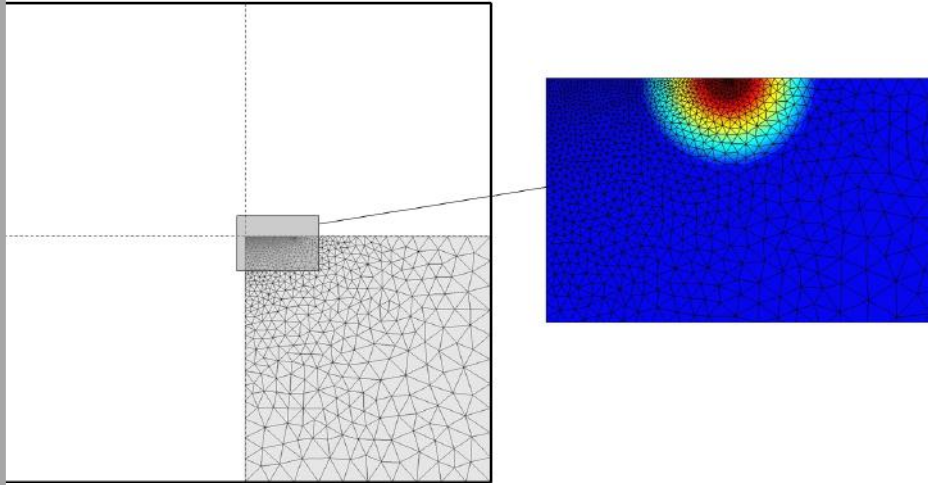
- (a) Set to unity at all nodes of the closed disk  $\|x - x_B\| \leq R_i$ ;
- (b) Linearly decreasing from unity to zero in the annulus  $R_i \leq \|x - x_B\| \leq R_e$ ;
- (c) Set to zero at all remaining nodes (i.e. those outside the closed disk  $\|x - x_B\| \leq R_e$ ).

This definition is implemented in the following code segment:

```

Ri=.1; % outer circle for theta=1 region
Re=.5; % inner circle for theta=0 region
theta=zeros(analysis.NN,2);
for i=1:analysis.NN,
    coor=nodes(i).coor(:);
    dist=sqrt((coor(1)-a)^2+coor(2)^2); % distance from crack tip
    if dist<(Ri-1.d-5),
        theta(i,1)=1.;
    elseif dist<(Re-1.d-5),
        theta(i,1)=1-(dist-Ri)/(Re-Ri); % imposes linear theta field
    end
end
end

```



**Figure 7.13:** Transformation velocity  $\theta$  (red:  $\theta = 1$ ; blue:  $\theta = 0$ ).

The evaluation of (7.42) then proceeds elementwise, as usual. For each element, the vectors  $U_e$  and  $G_e$ , both of size (6,1) for elements T3, are created; they store the nodal values of  $u_h$  and  $\theta$ , respectively. The element contribution is then computed by calling function T3\_2A\_solid\_gtheta or T6\_2A\_solid\_gtheta (depending on whether T3 or T6 elements are used, which this exposition stresses the T3 case for simplicity) and accumulated. The stress intensity factor is finally estimated using (7.45).

```

GB=0.;
for e=1:analysis.NE,
    type=elements(e).type;
    ne=Le(type).ne; % number of nodes in element
    Etag=Le(type).tag;
    Dne=ndof*ne; % number of nodal values
    Ue=zeros(Dne,1);
    Ge=zeros(Dne,1);
    pos=1;

```

```

for n=1:ne
    node=elements(e).nodes(n);
    Xe(n,:)=nodes(node).coor;           % creates element
    Ue(pos:pos+ndof-1)=nodes(node).U;
    Ge(pos:pos+ndof-1)=theta(node,:);
    pos=pos+ndof;
end
mat=elements(e).mat;
GB=eval([Etag Atag 'gtheta(Xe,material(mat,:),Ue,Ge)']);
GB=GB+GB;
end
K_GB=sqrt(material(1,1)/...           % extrapolation of K_GB
(1-material(1,2)^2)*2*GB)

```

The function  $GB=T3\_2A\_solid\_gtheta(X, mate, Ue, Ge)$  is now examined more closely. First, stresses in the element are evaluated:

```
sigma=T3_2A_solid_Sg(X, mate, Ue);           % computes stresses in element
```

The gradients  $\nabla \mathbf{u}_h$  and  $\nabla \boldsymbol{\theta}$  are also needed. Recall that, for example

$$\frac{\partial \mathbf{u}_h}{\partial x_k} = \frac{\partial}{\partial x_k} (a_1 \mathbf{u}^{(1)} + a_2 \mathbf{u}^{(2)} + a_3 \mathbf{u}^{(3)})$$

which implies (see also (2.37))

$$\{\nabla \mathbf{u}_h\} = \begin{Bmatrix} \frac{\partial u_{h1}}{\partial x_1} \\ \frac{\partial u_{h1}}{\partial x_2} \\ \frac{\partial u_{h2}}{\partial x_1} \\ \frac{\partial u_{h2}}{\partial x_2} \end{Bmatrix} = \begin{bmatrix} \frac{\partial a_1}{\partial x_1} & 0 & \frac{\partial a_2}{\partial x_1} & 0 & \frac{\partial a_3}{\partial x_1} & 0 \\ \frac{\partial a_1}{\partial x_2} & 0 & \frac{\partial a_2}{\partial x_2} & 0 & \frac{\partial a_3}{\partial x_2} & 0 \\ 0 & \frac{\partial a_1}{\partial x_1} & 0 & \frac{\partial a_2}{\partial x_1} & 0 & \frac{\partial a_3}{\partial x_1} \\ 0 & \frac{\partial a_1}{\partial x_2} & 0 & \frac{\partial a_2}{\partial x_2} & 0 & \frac{\partial a_3}{\partial x_2} \end{bmatrix} \{U_e\} \quad (7.46)$$

where  $\{\nabla \mathbf{u}\}$  is a column array holding the four components of  $\nabla \mathbf{u}_h$ , arranged according to the convention defined by (7.46). Denoting by  $G\_d$  and  $G\_t$  the MATLAB arrays holding  $\nabla \mathbf{u}_h$  and  $\nabla \boldsymbol{\theta}$ , respectively, one then has:

```

x11=X(1,1); x12=X(1,2);           % coordinates of first node
x21=X(2,1); x22=X(2,2);           % coordinates of second node
x31=X(3,1); x32=X(3,2);           % coordinates of third node
S=((x21-x11)*(x32-x12)-...         % area of element
(x31-x11)*(x22-x12))/2;
G=[x22-x32,0,x32-x12,0,x12-x22,0;   % gradient matrix
x31-x21,0,x11-x31,0,x21-x11,0;
0,x22-x32,0,x32-x12,0,x12-x22;
0,x31-x21,0,x11-x31,0,x21-x11]/(2*S);
G_d=(G*Ue);                       % displacement gradient
G_t=(G*Ge);                       % theta gradient

```

Then the contraction  $\nabla \mathbf{u}_h \cdot \nabla \boldsymbol{\theta}$ , being also a second-order tensor, can be stored in the equivalent line array  $prod\_G$ , having four entries, again following the convention used

in (7.46):

```
prod_G=[G_d(1)*G_t(1)+G_d(2)*G_t(3) ... % contraction of the two gradients
        G_d(1)*G_t(2)+G_d(2)*G_t(4) ...
        G_d(3)*G_t(1)+G_d(4)*G_t(3) ...
        G_d(3)*G_t(2)+G_d(4)*G_t(4)];
```

Then, setting up arrays  $\text{eps}$  (entries of  $\varepsilon[\mathbf{u}_h]$ ),  $\text{div\_t}$  (divergence of the field  $\theta_B$ ) and  $\text{symprod\_G}$  (symmetrized version of  $\nabla \mathbf{u}_h \cdot \nabla \theta_B$ ) allows to determine the contribution of the current element to  $G_B$ :

```
div_t=G_t(1)+G_t(2);
eps=[G_d(1) G_d(4) G_d(2)+G_d(3)]; % strains in engineering notation
symprod_G=[prod_G(1) prod_G(4) ... % symmetric part of prod_G
           prod_G(2)+prod_G(3)];
GBe=(symprod_G*sigma'-.5*eps*sigma'*div_t)*S; % sigma: stresses in element
```

Results obtained for the cracked plate example are shown in Table 7.2. For the plate with height  $V_3$ , the estimated value  $K_I^{V_3-G}$  of  $K_I$  is very close to the reference value (7.18) despite the fact that the computation of  $G_B$  is based on T3 elements only.

$K_I^H$	$K_I^{V_1-G}$	$K_I^{V_2-G}$	$K_I^{V_3-G}$
1.8175	1.8597	1.8078	1.8085

**Table 7.2:** Cracked plate, evaluation of  $K_I$ : reference solution and numerical estimations provided by the  $G$ - $\theta$  method with T3 elements.

**Exercise 7.1.** Write the MATLAB function `GBE=T6_2A-solid_gtheta` and apply the  $G$  -  $\theta$  method with standard T6 elements.

## Introduction to nonlinear solid mechanics. Nonlinear elasticity

---

This chapter provides an introduction to the numerical simulation of solids under nonlinear conditions. First, an overview of some major classes of nonlinear behavior of deformable structures is presented in Section 8.1. It is followed, in Section 8.2, by a general presentation of the Newton method for solving nonlinear differentiable equations. This iterative solution approach is then applied to nonlinear elasticity in Section 8.3, where the associated finite element implementation is fully developed. Section 8.4 ends this chapter with a more specific attention to the geometrically nonlinear problems. The iterative solution approach for the analysis of nonlinear structures developed in this chapter is fundamental because of its generality. In addition to being applied in this chapter to nonlinear elasticity, it will be used again in Chapter 9 as a foundation for numerical solution methodologies for elastoplastic solids.

### **8.1 Overview and illustrations of nonlinear structural behaviors**

This section is devoted to a brief presentation of some major classes of nonlinear behavior of deformable structures. Some of these behaviors may involve deformable structures whose constitutive material is described by a linear elastic model: contact (Section 8.1.1), fracture (Section 8.1.2), wear ... Other models involve material nonlinearities associated with an inelastic behavior of materials (Section 8.1.3): damage, plasticity ... or, when the assumption of small deformations is no longer valid (Section 8.1.4). Of course, analysis of a structure can mobilize more than one type of non-linearities (Section 8.1.5).

#### **8.1.1 Unilateral contact**

If a deformable solid is in contact with a rigid solid, or if a system is composed of deformable solids in contact with each other, no interpenetration should occur. As a consequence of the non- interpenetration constraint, the response of the system depends non-linearly on the applied load, even if its constitutive materials are assumed to be elastic linear. As such, the case of unilateral contact between solids

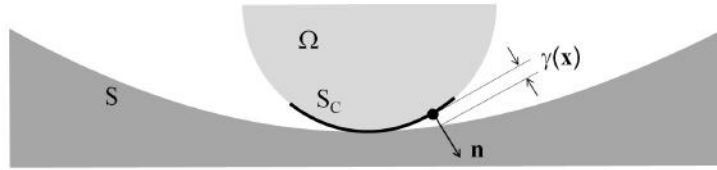
is a good example for introducing nonlinear problems. The simplest case is that of frictionless contact, where the solids in contact develop only *normal* reactions, mutually opposite, at contact points.

Consider the frictionless contact problem between a rigid body occupying the domain  $\Omega \subset \mathbb{R}^3$  with a deformable support  $S$ , as illustrated on Figure 8.1. Denote by  $S_C \subset \partial\Omega$  the *potential contact surface*, that is, the part of  $\partial\Omega$  outside of which no contact with the rigid support is expected.

The displacement  $\mathbf{u}$  and the traction vector  $\mathbf{T} := \boldsymbol{\sigma} \cdot \mathbf{n}$  then satisfy the following boundary conditions on  $S_C$ ,

$$\begin{aligned} \mathbf{T}(\mathbf{x}) - T_n(\mathbf{x})\mathbf{n}(\mathbf{x}) &= \mathbf{0} & (a) \\ T_n(\mathbf{x}) &\leq 0 & (b) \\ u_n(\mathbf{x}) - \gamma(\mathbf{x}) &\leq 0 & (c) \\ (u_n(\mathbf{x}) - \gamma(\mathbf{x}))T_n(\mathbf{x}) &= 0 & (d) \end{aligned} \quad (\mathbf{x} \in S_C), \quad (8.1)$$

where  $\gamma(\mathbf{x}) \geq 0$  denotes the initial gap (for the solid  $\Omega$  in its undeformed state) between  $S_C$  and  $S$ , measured along the normal direction to  $S_C$ , between  $\mathbf{x} \in S_C$  and the rigid support  $S$ , while  $u_n = \mathbf{u} \cdot \mathbf{n}$  and  $T_n = \mathbf{T} \cdot \mathbf{n}$  are the normal components of  $\mathbf{u}$  and  $\mathbf{T}$ . The condition of frictionless contact, the contact force sign constraint and the kinematic no-penetration requirement respectively correspond to equations (8.1a, b, c). Equation (8.1d) is the complementarity condition. The latter summarizes the two possible situations at point  $\mathbf{x} \in S_C$ : contact ( $u_n - \gamma = 0$ ) or non-contact ( $T_n = 0$ ), so that conditions (8.1) hold at any point of  $S_C$ . For a given loading, the set of points in contact is usually a subset of  $S_C$ , namely the *effective contact surface*  $S_C^{\text{eff}}$ .

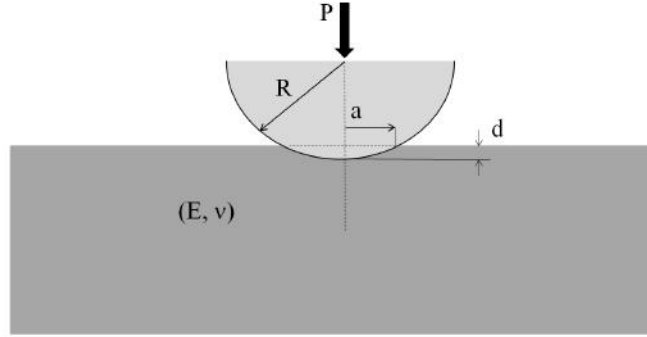


**Figure 8.1:** Potential contact surface  $S_C$  and distance  $\gamma(\mathbf{x})$  between  $S_C$  and the rigid support  $S$ .

In the case of contact with friction, very useful in practice, equations (8.1) must be complemented with an additional law for the tangential part  $\mathbf{T}_t$  of the reaction. For example, the Coulomb friction law writes, at any point  $\mathbf{x} \in S_C^{\text{eff}}$

$$\begin{aligned} \text{if } T_t < fT_n, & \text{ there is adhesion at } \mathbf{x} \\ \text{if } T_t = fT_n, & \text{ there is slipping at } \mathbf{x} \end{aligned}$$





**Figure 8.2:** Hertz contact problem definition.

where  $T_t = \|\mathbf{T} - T_n \mathbf{n}\|$  and  $f$  is the friction coefficient. In particular, the Coulomb friction law rules out situations such that  $T_t > f T_n$ .

As an illustration, consider the frictionless contact problem between an indenter (rigid solid having rotational symmetry about a vertical axis  $\Delta$ ) and an infinite elastic horizontal support, so that the initial contact point is located on  $\Delta$  as shown in Figure 8.2. The solution of this problem, known as the Hertz problem, gives the analytical expression of the penetration depth  $d$  and the radius  $a$  of the effective contact area as a function of the elastic constants  $(E, \nu)$  and the radius  $R$  of the indenter at the point of initial contact:

$$d = \left( \frac{3(1 - \nu^2)}{4E\sqrt{R}} \right)^{2/3} P^{2/3}, \quad a = \left( \frac{3R(1 - \nu^2)}{4E} \right)^{1/3} P^{1/3}, \quad (8.2)$$

The relation between  $P$  and  $d$  is clearly nonlinear.

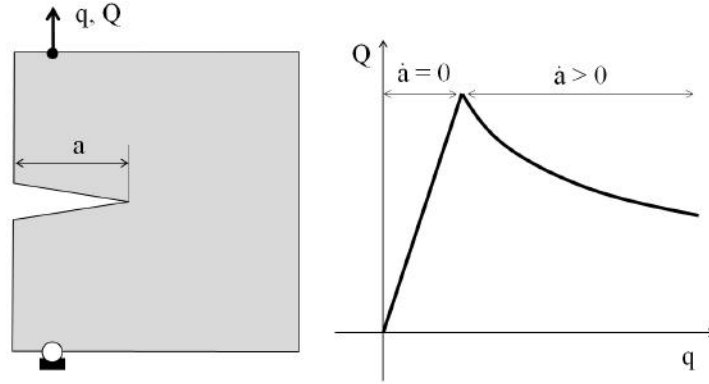
### 8.1.2 Crack propagation

Chapter 7 focused on various numerical tools for fracture mechanics under linear conditions: small-deformation assumption, linear elastic materials and fixed crack configurations. These assumptions lead to a linear load-response relation as the crack  $F$  remains fixed. However, under time-dependent loading, e.g.

$$\begin{aligned} \mathbf{u} &= \mathbf{u}^D(\mathbf{x}, t) \quad (\mathbf{x} \in S_u, t \in [0, T]), \\ \mathbf{T} &= \mathbf{T}^D(\mathbf{x}, t) \quad (\mathbf{x} \in S_T, t \in [0, T]), \end{aligned}$$

the crack propagates. The response of the cracked solid  $\Omega(F)$  then depends nonlinearly on the load (even assuming quasi-static conditions and a linear elastic material), as the crack  $F = F(t)$  depends on time, in a way that is not known *a priori*. Indeed, the propagation usually occurs according to criteria involving thresholds, such as (7.8) or (7.11). Moreover, for a given load, the elastic response of the solid also depends on the configuration  $F$  of the crack, as shown schematically in Figure 8.3.

The evolution of the crack configuration (length and direction) being *a priori* unknown, the numerical computation of the crack propagation generally requires a time-stepping approach.



**Figure 8.3:** Specimen for fracture testing: schematic (left), force-displacement diagram ( $Q, q$ ) (right). The non-linear part of the curve corresponds to crack propagation (here unstable if the test is force-driven)

### 8.1.3 Material nonlinearities

There is a great diversity of behaviors of solid materials, and of models to describe them. Many of these models assume a linear elastic response when stresses belong to some *elasticity domain* specified by the model, and a nonlinear response otherwise. The linear elastic model, both simplest and easiest to implement, has practical relevance for a substantial proportion of the applications. There are also, however, many situations where it is necessary to take into account other (usually nonlinear) types of material response.

No attempt is made in this book to describe, even briefly, the variety of known nonlinear behaviors that mobilize phenomena such as plasticity, damage or damping. The reader wishing to explore these concepts may refer, for example, to the books by Besson et al. (2009) or Lemaître and Chaboche (1990). Chapter 9 discusses in detail the case of quasi-static elastoplastic computations under the small-deformation assumption.

**Elastic behavior with damage.** The properties of a material at the macroscopic scale, modelled as a continuous medium, may be adversely affected by the presence of micro-defects (cracks, cavities), individually of very small size but in large numbers. Similarly to a macroscopic crack, these micro-defects are likely to grow as a result of the loading history applied to the structure. The tools developed in linear fracture mechanics are not suited to taking into account such sets of micro-defects. This has prompted the development of *damage theory*.

As an illustration, the *brittle damage* still consider the material behavior as linear elastic but describe the presence of micro-defects by a modification (weakening) of the macroscopic elastic properties. The elastic tensor is then function of a internal variable  $\alpha$  (either scalar or tensor-valued depending on models), called the *damage variable*:

$$\mathcal{A} = \mathcal{A}(\alpha). \quad (8.3)$$

The simplest damage model has the form

$$\mathcal{A} = \mathcal{A}_0(1 - \alpha),$$

where  $\mathcal{A}_0$  is the elastic tensor of the undamaged material. The damage is assumed irreversible (i.e.  $\alpha$  is a non-decreasing function of time) and its possible evolution is described by a law involving a energy density threshold  $w_{\text{critical}}$ :

$$\begin{cases} \text{if } \varepsilon : \mathcal{A}(\alpha) : \varepsilon < w_{\text{critical}} & \text{so } \dot{\alpha} = 0, \\ \text{if } \varepsilon : \mathcal{A}(\alpha) : \varepsilon = w_{\text{critical}} & \text{so } \dot{\alpha} \geq 0. \end{cases}$$

For a solid, the damage  $\alpha$  is space-dependent, and hence is a field. Given the progressive and irreversible nature of the damage, the response of a damaged solid is a nonlinear function of the applied load and depends on the loading history.

**Elastoplastic behavior.** The elastoplastic behavior has the following features:

- Existence of an elasticity domain: the material behavior is linear elastic as long as the stress state does not reach a threshold called yield limit
- The yield limit may be reached, but not exceeded. Upon reaching it, incompatible plastic strains may appear and evolve under subsequent load increases.
- The appearance or evolution of plastic strain is accompanied by energy dissipated as heat, and this behavior is *irreversible* (e.g. a loading-unloading cycle produces a final mechanical state with residual stresses and strains whenever the loading phase generates plastic strains)

The key concepts for this class of constitutive models, under three-dimensional conditions and the small-deformation assumption, are recalled in Chapter 9.

#### 8.1.4 Geometric nonlinearities

The small-deformation assumption is at the basis of the linearization of the kinematic description of the motion of a continuum. By allowing to treat the initial configuration  $\Omega$  and its current (deformed) counterpart  $\Omega_t$  as identical for writing local equations and boundary conditions, it achieves a major mathematical simplification. The error induced by the approximation  $\Omega_t \approx \Omega$  is of order  $O(\|\nabla \mathbf{u}\|^2)$ .

When the small-deformation assumption is not acceptable, the analysis of a solid becomes necessarily nonlinear, if only because of the now-nonlinear strain-displacement relations.

- The strain cannot be approximated by its linearized form  $\varepsilon$ , and one has to use the Green-Lagrange tensor  $\mathbf{E}$ , nonlinear in  $\nabla \mathbf{u}$ :

$$\mathbf{E} = \frac{1}{2} (\nabla \mathbf{u} + \nabla^T \mathbf{u} + \nabla^T \mathbf{u} \cdot \nabla \mathbf{u}). \quad (8.4)$$

- The weak form (1.19) of the balance equation has to be written on the *current configuration*  $\Omega_t$  and its boundary, which are generally not known *a priori*.

The geometrically nonlinear version of elasticity, its formulation and finite element treatment will be presented in more detail in Section 8.3. Section 8.4 will then describe a simple algorithm for computing the buckling of a beam.

### 8.1.5 Combination of several types of non-linearities

Many situations present simultaneously several types of nonlinearities. For example, the non-linearities associated with cracking, damage, contact or wear, previously introduced for linear elastic solids and hence under the small-deformation assumption, may also occur for solids whose material has a nonlinear behavior, or in association with large displacements and strains.

### 8.1.6 Solution methods: preliminary remarks

The foregoing overview has emphasized the vast variety of non-linear behaviors that can be involved in the analysis of solids and structures. Consequently, there is also a great variety of numerical solution methods, which are hardly possible to present in a unified way. The common feature of most numerical techniques dedicated to solving balance equations involving nonlinearities is their iterative nature. Indeed, they are often based on the construction of a sequence of solution candidates which converge to a solution (not necessarily unique) of the system of equations arising from a finite element discretization.

## 8.2 Newton-type iterative methods

Many computational structure problems lead, after a finite element discretization, to seeking the N-vector of unknown nodal displacements  $\{\mathbf{U}\}$  by solving a system of equations of the form

$$\{\mathbb{R}(\{\mathbf{U}\})\} = \{0\} \quad (8.5)$$

where  $\{\mathbf{U}\} \mapsto \{\mathbb{R}(\{\mathbf{U}\})\} \in \mathbb{R}^N$  is a vector-valued function, often called *residual*, assumed to be differentiable with respect to  $\{\mathbf{U}\}$ <sup>1</sup>.

Several types of iterative algorithms are available for the numerical solution of systems of equations of the form (8.5). In this section, we successively examine the solution of scalar equations (Sec. 8.2.1) and systems of equations (Sec. 8.2.2) by Newton's method (also known as the Newton-Raphson method) and its variants.

### 8.2.1 Case of a scalar equation

Consider the generic problem of finding the roots (zeros) of a scalar function  $r(u)$ :

$$\text{Find } u \text{ such that } r(u) = 0.$$

which corresponds to a simplified version of (8.5). Any iterative algorithm consists in a specific method for constructing a sequence  $u^{[k]} \rightarrow u$  such that  $r(u) = 0$ .

---

<sup>1</sup>The case of unilateral contact does not fall within this category, as it prominently involves non-differentiability.

**Newton method.** It is based on the first-order approximation of  $r(u^{[k+1]})$  about the previous iterate  $u^{[k]}$ , and consists in fact in determining the iterate  $u^{[k+1]}$  by setting that approximation to zero:

$$r^{[k+1]} \approx r^{[k]} + (u^{[k+1]} - u^{[k]})r'^{[k]} = 0,$$

where  $r^{[m]} = r(u^{[m]})$  and  $r'^{[m]} = r'(u^{[m]})$  are the values of the function  $r$  and its derivative  $r'$  at  $u^{[m]}$ . The approximation  $u^{[k+1]}$  is given by,

$$u^{[k+1]} = u^{[k]} - r^{[k]} / r'^{[k]} \quad (8.6)$$

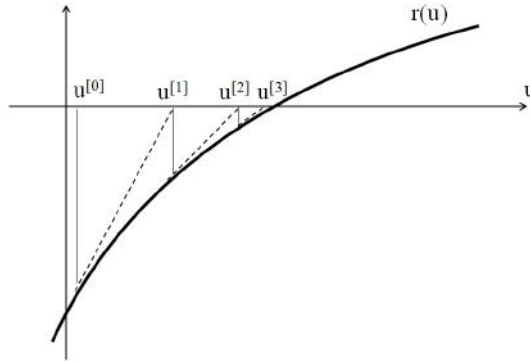
The geometric interpretation of (8.6) is simple: the approximation  $u^{[k+1]}$  is determined by seeking the intersection of the tangent at point  $(u^{[k]}, r^{[k]})$  to the curve  $(u, r(u))$  and the axis  $r = 0$  (Figure 8.4). This tangent defines a search direction that may be termed *consistent* as it results from exact linearization of the residual  $r(u)$  about the current iterate  $u^{[k]}$ .

The Newton method does not converge unconditionally with respect to the choice of initial iterate  $u^{[0]}$ : a bad choice of  $u^{[0]}$  can indeed cause (8.6) to build a divergent sequence  $u^{[k]}$  (see Fig. 8.5). When convergence occurs, it has the remarkable property of being quadratic. Towards proving this property, let  $e^{[k]} = u^{[k]} - u$  denote the solution error at iteration  $k$ , so that

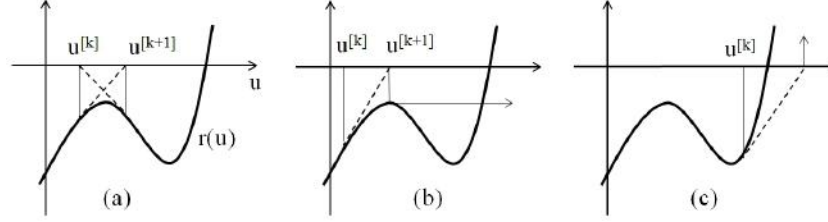
$$e^{[k+1]} - e^{[k]} = u^{[k+1]} - u^{[k]} = -r^{[k]} / r'^{[k]} \quad (8.7)$$

by using (8.6). A first-order Taylor expansion of  $r^{[k]}$  and  $r'^{[k]}$  about the exact solution  $u$ , with its remainder set in integral form, leads to

$$\begin{aligned} r^{[k]} &= r'(u)e^{[k]} + \frac{1}{2}r''(u + \alpha e^{[k]})(e^{[k]})^2 & (0 \leq \alpha \leq 1), \\ r'^{[k]} &= r'(u) + r''(u + \beta e^{[k]})e^{[k]} & (0 \leq \beta \leq 1), \end{aligned}$$



**Figure 8.4:** Scalar nonlinear equation: principle of the Newton method (with consistent search direction obtained by exact linearization).



**Figure 8.5:** Scalar nonlinear equation, Newton method: divergence examples.

where  $\alpha$  and  $\beta$  are scalars depending on  $e^{[k]}$ . Using these formulas, we then express  $e^{[k+1]}$  given by (8.7) and obtain the recurrence relation linking successive errors  $e^{[k+1]}$  and  $e^{[k]}$ :

$$\begin{aligned} e^{[k+1]} &= \frac{2r''(u + \beta e^{[k]}) - r''(u + \alpha e^{[k]})}{2r'(u) + 2r''(u + \beta e^{[k]})e^{[k]}} (e^{[k]})^2 \\ &= \frac{r''(u)}{2r'(u)} (e^{[k]})^2 + o(|e^{[k]}|^2) = O(|e^{[k]}|^2). \end{aligned} \quad (8.8)$$

This expression highlights the *quadratic convergence* of Newton's method in a neighborhood of the solution: for a small enough value of the error  $e^{[k]}$ , the next error  $e^{[k+1]}$  is proportional to the square of  $e^{[k]}$ . Once "close" to the solution, the algorithm, defined by the recurrence (8.6), has a fast convergence.

**Modified Newton method: constant search direction.** This variant consists in replacing  $r'^{[k]}$  in (8.6) by a constant  $K$  (e.g.  $r'^{[0]}$ ), which leads to the recurrence

$$u^{[k+1]} = u^{[k]} - r^{[k]}/K. \quad (8.9)$$

Geometrically speaking, the iterate  $u^{[k+1]}$  is determined by seeking the intersection between the line of slope  $K$  passing through the point  $(u^{[k]}, r^{[k]})$  and the axis  $r = 0$  (Fig. 8.6).

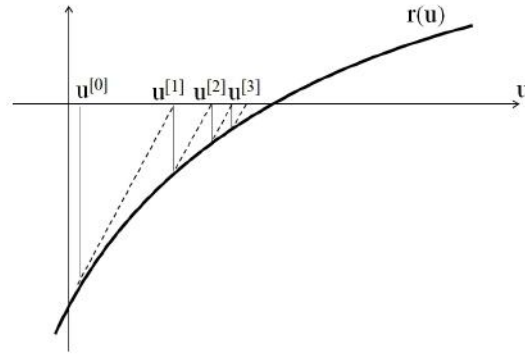
The convergence of this modified version is only linear in the neighborhood of the solution, as successive errors are found to be connected by

$$e^{[k+1]} = (1 - r'(u)/K)e^{[k]} + o(|e^{[k]}|) = O(|e^{[k]}|).$$

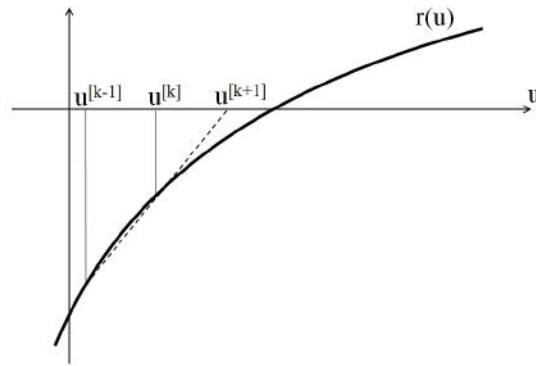
The modified Newton method is not of great interest for solving scalar equations, but we will see that its extension to systems of nonlinear equations in  $\mathbb{R}^N$  can be useful because of computational cost considerations.

**Modified Newton method: secant method.** This variant consists in replacing  $r'^{[k]}$  by the slope of the secant line passing through the points  $(u^{(k-1)}, r(u^{(k-1)}))$  and  $(u^{[k]}, r^{[k]})$  which correspond to the two previous iterates (Fig. 8.7), leading to the recurrence

$$u^{[k+1]} = u^{[k]} - r^{[k]} \frac{u^{[k]} - u^{(k-1)}}{r^{[k]} - r(u^{(k-1)})}. \quad (8.10)$$



**Figure 8.6:** Scalar nonlinear equation: principle of the modified Newton method with constant search direction.



**Figure 8.7:** Scalar nonlinear equation: principle of the secant method.

One can prove that the convergence of this variant is such that

$$|e^{[k+1]}| = O(|e^{[k]}|^\alpha), \quad \alpha = (1 + \sqrt{5})/2.$$

(incidentally, the exponent  $\alpha$  in the above expression is the well-known *golden ratio*). Consequently, the convergence of the recurrence (8.10) is slower than that achieved by the "standard" Newton method ( $\alpha = 2$ , i.e. quadratic convergence), yet faster than using the Newton method with constant search direction ( $\alpha = 1$ , i.e. linear convergence).

### 8.2.2 Systems of equations

Solving a nonlinear system of equations (8.5) is also based on iterative algorithms. These consist in building a sequence  $\{\mathbb{U}^{[k]}\}$  converging to a limit  $\{\mathbb{U}\}$  is the solution of  $\{\mathbb{R}(\{\mathbb{U}\})\} = \{0\}$ .

As for the case of scalar equations, the Newton method is based on the first-order Taylor expansion of the residual function about the previous iterate:

$$\{\mathbb{R}^{[k+1]}\} \approx \{\mathbb{R}^{[k]}\} + [\mathbb{K}^{[k]}]\{\delta\mathbb{U}^{[k]}\}, \quad (8.11)$$

where  $\{\mathbb{R}^{[m]}\} = \{\mathbb{R}(\{\mathbb{U}^{[m]}\})\}$  is the residual vector at the  $m$ -th iteration, and

$$\{\delta\mathbb{U}^{[k]}\} = \{\mathbb{U}^{[k+1]}\} - \{\mathbb{U}^{[k]}\}$$

is the correction that defines the current approximation  $\{\mathbb{U}^{[k+1]}\}$ , and

$$[\mathbb{K}^{[k]}] = [\nabla_{\mathbb{U}}\{\mathbb{R}\}](\{\mathbb{U}^{[k]}\}) \quad \text{i.e.} \quad [\mathbb{K}^{[k]}]_{IJ} = \frac{\partial\{\mathbb{R}\}_I}{\partial\{\mathbb{U}\}_J}(\{\mathbb{U}^{[k]}\}) \quad (8.12)$$

is the *tangent matrix* at  $\{\mathbb{U}^{[k]}\}$ . The correction  $\{\delta\mathbb{U}^{[k]}\}$  is then found by setting the approximation (8.11) of  $\{\mathbb{R}^{[k+1]}\}$  to zero, i.e. by solving the linear system

$$\{\mathbb{R}^{[k]}\} + [\mathbb{K}^{[k]}]\{\delta\mathbb{U}^{[k]}\} = \{0\} \quad (8.13)$$

The implementation of the Newton algorithm dedicated to a system of equations (8.5) is summarized in Box 8.1 (where  $\epsilon$  denotes a preset tolerance). The tangent matrix  $[\mathbb{K}^{[k]}]$  is referred to as *consistent* as it stems from an exact linearization of the residual  $\{\mathbb{R}(\{\mathbb{U}\})\}$  about the current iterate.

**Box 8.1:** Newton algorithm for a nonlinear system of equations

1. Initialization:
  - (i) Choose  $\{\mathbb{U}^{[0]}\}$  (often  $\{\mathbb{U}^{[0]}\} = \{0\}$ ) and tolerance  $\epsilon$
  - (ii) Compute initial residual  $\{\mathbb{R}^{[0]}\} := \{\mathbb{R}(\{\mathbb{U}^{[0]}\})\}$
2. For  $k = 0, 1, \dots$ 
  - (i) Compute global stiffness tangent matrix  $[\mathbb{K}^{[k]}]$ ;
  - (ii) Update the iterate  $\{\mathbb{U}^{[k+1]}\} = \{\mathbb{U}^{[k]}\} + [\mathbb{K}^{[k]}]^{-1}\{\mathbb{R}^{[k]}\}$ ;
  - (iii) Compute new residual  $\{\mathbb{R}^{[k+1]}\} = \{\mathbb{R}(\{\mathbb{U}^{[k+1]}\})\}$ ;
  - (iv) Convergence test:  $\|\{\mathbb{R}^{[k+1]}\}\| \leq \epsilon\|\{\mathbb{R}^{[0]}\}\|$  ?;
    - If yes: STOP,  $\{\mathbb{U}\} = \{\mathbb{U}^{[k+1]}\}$
    - If no, do  $k \leftarrow k + 1$  and return to 2(i).

**Local quadratic convergence near a solution.** As in the case of the scalar equation, the Newton method has a quadratic convergence in a neighborhood of the solution. To show this, we define the solution error  $\{\mathbb{E}^{[k]}\} := \{\mathbb{U}^{[k]}\} - \{\mathbb{U}\}$  at iteration  $k$  and the solution, and we use the relation

$$\{\mathbb{E}^{[k+1]}\} = \{\mathbb{E}^{[k]}\} + \{\delta\mathbb{U}^{[k]}\} = \{\mathbb{E}^{[k]}\} - [\mathbb{K}^{[k]}]^{-1}\{\mathbb{R}^{[k]}\} \quad (8.14)$$

which follows from this definition and the recurrence (8.13). Explicit expressions for the Taylor expansions (with integral remainder) about the solution  $\{\mathbb{U}\}$  of



$\{\mathbb{R}^{[k]}\}$  (to second order) and  $[\mathbb{K}^{[k]}]^{-1}$  (to first order) are then derived, like for the scalar case. Using Einstein's summation convention for repeated indices, these expansions are found as:

$$\begin{aligned}\{\mathbb{R}^{[k]}\}_I &= [\mathbb{K}]_{IJ} \{\mathbb{E}^{[k]}\}_J + \frac{1}{2} \frac{\partial^2 \{\mathbb{R}\}_I}{\partial \{\mathbb{U}\}_J \partial \{\mathbb{U}\}_K} (\{\mathbb{U} + \alpha \mathbb{E}^{[k]}\}) \{\mathbb{E}^{[k]}\}_J \{\mathbb{E}^{[k]}\}_K \\ [\mathbb{K}^{[k]}]_{IJ} &= [\mathbb{K}]_{IJ} + \frac{\partial^2 \{\mathbb{R}\}_I}{\partial \{\mathbb{U}\}_J \partial \{\mathbb{U}\}_K} (\{\mathbb{U} + \beta \mathbb{E}^{[k]}\}) \{\mathbb{E}^{[k]}\}_K \\ [\mathbb{K}^{[k]}]_{IJ}^{-1} &= [\mathbb{K}]_{IJ}^{-1} - [\mathbb{K}]_{IK}^{-1} \frac{\partial^2 \{\mathbb{R}\}_K}{\partial \{\mathbb{U}\}_M \partial \{\mathbb{U}\}_L} (\{\mathbb{U} + \beta \mathbb{E}^{[k]}\}) \{\mathbb{E}^{[k]}\}_M [\mathbb{K}]_{LJ}^{-1}\end{aligned}$$

having set

$$[\mathbb{K}]_{IJ} = \frac{\partial \{\mathbb{R}\}_I}{\partial \{\mathbb{U}\}_J}(\{\mathbb{U}\}).$$

Identity (8.14) then becomes

$$\begin{aligned}\{\mathbb{E}^{[k+1]}\}_I &= \frac{1}{2} [\mathbb{K}]_{IJ}^{-1} \left[ \frac{\partial^2 \{\mathbb{R}\}_J}{\partial \{\mathbb{U}\}_K \partial \{\mathbb{U}\}_L} (\{\mathbb{U} + \beta \mathbb{E}^{[k]}\}) \right. \\ &\quad \left. - 2 \frac{\partial^2 \{\mathbb{R}\}_J}{\partial \{\mathbb{U}\}_K \partial \{\mathbb{U}\}_L} (\{\mathbb{U} + \alpha \mathbb{E}^{[k]}\}) \right] \{\mathbb{E}^{[k]}\}_K \{\mathbb{E}^{[k]}\}_L + o(\|\{\mathbb{E}^{[k]}\}\|^2) \\ &= O(\|\{\mathbb{E}^{[k]}\}\|^2),\end{aligned}\tag{8.15}$$

generalizing (8.8) to systems of equations. This establishes, and highlights, the quadratic convergence of Newton's method in a neighborhood of the solution: when the error  $\|\{\mathbb{E}^{[k]}\}\|$  is small enough, the error  $\|\{\mathbb{E}^{[k+1]}\}\|$  at the next iteration is proportional to the square of  $\|\{\mathbb{E}^{[k]}\}\|$ . As for the scalar case, once in the neighborhood of a solution (whose precise characterization depends on the vector function  $\{\mathbb{R}\}$ ), the algorithm defined by the recursion (8.13) has a very fast convergence.

### 8.3 Nonlinear elasticity

The small-deformation assumption has been consistently assumed throughout this book up to this point. In other words, deformations were assumed to be small enough to lend validity to the linearized kinematic description of the motion of a continuum. This linearized framework, whereby the initial and current (deformed) configurations are treated as identical and linearized strains are employed, permits considerable mathematical simplification, in particular by allowing to write field equations and boundary conditions on the (known) initial configuration. In the remainder of this chapter, we consider the more difficult case where deformations and strains may become large during the loading process. As a consequence, strains can no longer be approximated by their linearized form  $\varepsilon[\mathbf{u}]$ . The exact strain is given by the Green-Lagrange strain tensor  $\mathbf{E}[\mathbf{u}]$ , which is nonlinear in  $\mathbf{u}$ :

$$\mathbf{E}[\mathbf{u}] = \frac{1}{2} (\mathbf{F}^T[\mathbf{u}] \cdot \mathbf{F}[\mathbf{u}] - \mathbf{I}) = \frac{1}{2} (\nabla \mathbf{u} + \nabla^T \mathbf{u} + \nabla^T \mathbf{u} \cdot \nabla \mathbf{u}), \tag{8.16}$$

where

$$\mathbf{F} = \mathbf{F}[\mathbf{u}] = \mathbf{I} + \nabla \mathbf{u}$$

is the deformation gradient. The current configuration  $\Omega_t$ , on which the field equations and boundary conditions are written, must be distinguished from the initial (undeformed) configuration  $\Omega$ , both configurations being linked by  $\Omega_t = \Omega + \mathbf{u}$ . While  $\Omega$  is known, the latter relationship shows that  $\Omega_t$  depends on the unknown displacement  $\mathbf{u}$ , and is therefore unknown at the start of the analysis. The material is assumed to be elastic, with a constitutive relation of the form<sup>2</sup>:

$$\boldsymbol{\sigma} = \rho_t \mathbf{F} \cdot \frac{\partial \phi}{\partial \mathbf{E}} \cdot \mathbf{F}^T = \frac{1}{J} \mathbf{F} \cdot \mathbf{S} \cdot \mathbf{F}^T \quad \text{with} \quad \mathbf{S} = \rho \frac{\partial \phi}{\partial \mathbf{E}}, \quad (8.17)$$

where  $\phi(\mathbf{E})$  is the free energy,  $\rho_t$  and  $\rho$  respectively denote the mass density in the current and initial configuration, and  $J = \det \mathbf{F}$  is the Jacobian of the deformation. The second-order tensor  $\mathbf{S}$  defined in (8.17) is known as the (second) Piola stress tensor. It will be convenient to also introduce the first Piola-Kirchhoff stress tensor  $\mathbf{P}$ :

$$\mathbf{P} = J \boldsymbol{\sigma} \cdot \mathbf{F}^{-T} = \mathbf{F} \cdot \mathbf{S}. \quad (8.18)$$

The constitutive law (8.17) then relates  $\mathbf{P}$  to the deformation through<sup>3</sup>

$$\mathbf{P} = \rho \mathbf{F} \cdot \frac{\partial \phi}{\partial \mathbf{E}} = \rho \frac{\partial \phi}{\partial \mathbf{F}}. \quad (8.19)$$

Note that, by definition, the Piola and Cauchy stress tensors are symmetric tensors while the first Piola-Kirchhoff tensor is not.

The foregoing considerations provide a first example for a detailed presentation of the construction of the associated system of nonlinear equations (8.5) and of the implementation of previously-described solution algorithms for systems of nonlinear equations.

### 8.3.1 Governing equations

Consider a solid body whose initial configuration occupies the domain  $\Omega \subset \mathbb{R}^3$ . The solid material is assumed to have an elastic behavior. Assume that the solid, in its current configuration  $\Omega_t \subset \mathbb{R}^3$ , is subjected to a volume force density  $\mathbf{f}$  and a surface force density  $\mathbf{T}^D$  on a part  $(S_T)_t$  of its boundary  $\partial\Omega_t$ , while the remaining part of  $(S_u)_t = \partial\Omega_t \setminus (S_T)_t$  of the boundary is constrained by prescribed displacements  $\mathbf{u}^D$  applied on the portion  $S_u$  of the reference configuration. The equilibrium state of the solid is then governed by:

$$\operatorname{div}_t \boldsymbol{\sigma} + \rho_t \mathbf{f} = \mathbf{0} \quad \text{in } \Omega_t \quad (\text{equilibrium}), \quad (8.20a)$$

$$\mathbf{T} = \mathbf{T}^D \quad \text{on } (S_T)_t \quad (\text{external loading}), \quad (8.20b)$$

<sup>2</sup>Such a form can be established from the laws of thermodynamics

<sup>3</sup>Using the chain rule and the symmetry of the Green-Lagrange tensor, we have  $d\phi = \frac{\partial \phi}{\partial \mathbf{E}} : d\mathbf{E} = \frac{\partial \phi}{\partial \mathbf{E}} : \frac{1}{2} (d\mathbf{F}^T \cdot \mathbf{F} + \mathbf{F}^T \cdot d\mathbf{F}) = \frac{\partial \phi}{\partial \mathbf{E}} : \mathbf{F}^T \cdot d\mathbf{F} = \mathbf{F} \cdot \frac{\partial \phi}{\partial \mathbf{E}} : d\mathbf{F}$ . By identification,  $\mathbf{F} \cdot \frac{\partial \phi}{\partial \mathbf{E}} = \frac{\partial \phi}{\partial \mathbf{F}}$  follows.

$$\boldsymbol{\sigma} = \rho_t \mathbf{F} \cdot \frac{\partial \phi}{\partial \mathbf{E}} \cdot \mathbf{F}^T \quad \text{in } \Omega_t \quad (\text{elastic constitutive law}), \quad (8.20c)$$

$$\mathbf{u} = \mathbf{u}^D \quad \text{on } S_u \quad (\text{prescribed displacement}), \quad (8.20d)$$

where  $\text{div}_t$  is the divergence operator with respect to the current configuration  $\Omega_t$ .

### 8.3.2 Examples of constitutive models

**Saint-Venant-Kirchhoff model.** The simplest model is the Saint-Venant-Kirchhoff model, which is defined by the free energy function

$$\rho \phi(\mathbf{E}) = \frac{1}{2} \mathbf{E} : \mathcal{A} : \mathbf{E}, \quad (8.21)$$

where  $\mathcal{A}$  is the fourth-order tensor of elastic moduli already introduced in Chapter 1. We can obtain the Piola tensor  $\mathbf{S}$  as

$$\mathbf{S} = \mathcal{A} : \mathbf{E}.$$

In particular, for materials that are isotropic in the reference configuration, the Piola stress tensor reads

$$\mathbf{S} = \lambda(\text{tr} \mathbf{E}) \mathbf{I} + 2\mu \mathbf{E},$$

where  $\lambda$  and  $\mu$  are material parameters. Unfortunately, these models are only valid for the small-strain case<sup>4</sup>.

**Compressible materials isotropic in the reference configuration.** For compressible materials that are isotropic in the reference configuration, one can prove<sup>5</sup> that the free energy function only depends on the three invariants<sup>6</sup> of the dilatation tensor  $\mathbf{C} = \mathbf{F}^T \cdot \mathbf{F} = \mathbf{I} + 2\mathbf{E}$ , defined by

$$I_1 = \text{tr}(\mathbf{C}), \quad I_2 = \frac{1}{2}((\text{tr} \mathbf{C})^2 - \text{tr} \mathbf{C}^2), \quad I_3 = J^2 = \det(\mathbf{C}),$$

whose derivatives with respect to the Green-Lagrange strain tensor are

$$\frac{\partial I_1}{\partial \mathbf{E}} = 2\mathbf{I}, \quad \frac{\partial I_2}{\partial \mathbf{E}} = 2I_1 \mathbf{I} - 2\mathbf{C}, \quad \frac{\partial I_3}{\partial \mathbf{E}} = 2I_3 \mathbf{C}^{-1}.$$

The general expression of the constitutive law for compressible materials isotropic in the reference configuration finally reads:

$$\mathbf{S} = 2\rho \left[ \left( \frac{\partial \phi}{\partial I_1} + I_1 \frac{\partial \phi}{\partial I_2} \right) \mathbf{I} - \frac{\partial \phi}{\partial I_2} \mathbf{C} + \frac{\partial \phi}{\partial I_3} I_3 \mathbf{C}^{-1} \right]. \quad (8.22)$$

Among the large variety of available such models, we mention the compressible neo-Hookean model

$$\rho \phi(\mathbf{E}) = \frac{\mu}{2}(I_1 - 3) - \mu \ln J + \frac{\lambda}{2}(\ln J)^2$$

<sup>4</sup>These models can be shown to reach infinite compression rate under finite compression load

<sup>5</sup>By virtue of the Rivlin-Ericksen theorem (see Ciarlet, 1988, Theorem 3.6.1)

<sup>6</sup>We have  $I_1 = c_1 + c_2 + c_3$ ,  $I_2 = c_1 c_2 + c_2 c_3 + c_3 c_1$  and  $I_3 = c_1 c_2 c_3$ , where  $c_1, c_2, c_3$  are the eigenvalues of  $\mathbf{C}$

where  $\mu, \lambda$  are material parameters, and the compressible Mooney-Rivlin model

$$\rho\phi(\mathbf{E}) = C_1(I_1 - 3) + C_2(I_2 - 3) + A(I_3 - 1) - (C_1 + 2C_2 + A) \ln I_3$$

where  $C_1, C_2$  and  $A$  are material parameters and the dependence in the third invariant is designed so as to satisfy the polyconvexity assumptions of Ball (1977).

### 8.3.3 Principle of virtual work. Weak formulation

The weak form of the equilibrium equation (8.20a), which corresponds to the virtual work principle, has to be written in the (unknown) current configuration  $\Omega_t$ . Taking into account the boundary conditions (8.20b), it yields

$$\int_{\Omega_t} \boldsymbol{\sigma} : \nabla_t \mathbf{w} \, dV_t = \int_{\Omega_t} \rho_t \mathbf{f} \cdot \mathbf{w} \, dV_t + \int_{S_{T,t}} \mathbf{T}^D \cdot \mathbf{w} \, dS_t + \int_{S_{u,t}} [\boldsymbol{\sigma} \cdot \mathbf{n}_t] \cdot \mathbf{w} \, dS_t \quad \forall \mathbf{w} \in \mathcal{C}_t, \quad (8.23)$$

where  $\rho_t$  is the mass density in the current configuration,  $\nabla_t$  is the gradient operator with respect to the current coordinates,  $\mathbf{n}_t$  is the outward unit normal to  $\Omega_t$  and  $\mathcal{C}_t$  is the space of displacement fields that are sufficiently regular in  $\Omega_t$ . Moreover, the convention  $\mathbf{A} : \mathbf{B} = \text{tr}(\mathbf{A}^T \cdot \mathbf{B})$ , i.e.  $\mathbf{A} : \mathbf{B} = A_{ij} B_{ij}$  using component notation, is used in (8.23) and thereafter. Using the transformation relationships

$$\nabla_t \mathbf{w} = \nabla \mathbf{w} \cdot \mathbf{F}^{-1}, \quad dV_t = J \, dV, \quad \mathbf{n}_t \, dS_t = J \mathbf{F}^{-T} \cdot \mathbf{n} \, dS,$$

and the definition (8.18) of the first Piola-Kirchhoff stress tensor, the weak form (8.23) can then be mapped to the (known) reference configuration  $\Omega$ :

$$\int_{\Omega} \mathbf{P} : \nabla \mathbf{w} \, dV = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_T} \bar{\mathbf{T}}^D \cdot \mathbf{w} \, dS + \int_{S_u} [\mathbf{P} \cdot \mathbf{n}] \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}, \quad (8.24)$$

where  $\rho$  ( $\rho_t = J\rho$ ) is the mass density in the reference configuration,  $\mathbf{n}$  is the outward unit normal in the reference configuration, and  $\bar{\mathbf{T}}^D$  denotes imposed tractions in the reference configuration. For instance, for an applied pressure of constant value  $p_0$ , imposed tractions in the reference configuration are given by

$$\bar{\mathbf{T}}^D = -p_0 J \mathbf{F}^{-T} \cdot \mathbf{n}$$

(note that  $\bar{\mathbf{T}}^D$  then depends on  $\mathbf{u}$ ). At last, restricting (8.24) to the virtual fields that are compatible with zero boundary conditions on  $S_u$  (i.e.  $\mathbf{w} \in \mathcal{C}(\mathbf{0})$ ), the *weak formulation* for the equilibrium problem in the reference configuration follows:

$$\int_{\Omega} \mathbf{P} : \nabla \mathbf{w} \, dV = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_T} \bar{\mathbf{T}}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (8.25)$$

We can now substitute the constitutive law (8.19) (or equivalently (8.20c)) into (8.25) and take into account the kinematic condition (8.20d) through the set  $\mathcal{C}(\mathbf{u}^D)$  of admissible displacements among which  $\mathbf{u}$  is sought. This leads to the weak form of balance equations for the nonlinear elasticity problem

$$\text{find } \mathbf{u} \in \mathcal{C}(\mathbf{u}^D) \text{ such that } \mathcal{R}(\mathbf{u}; \mathbf{w}) = 0 \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}), \quad (8.26)$$

where the *residual*  $\mathcal{R}(\mathbf{u}; \mathbf{w})$  is defined by

$$\mathcal{R}(\mathbf{u}; \mathbf{w}) = -\mathcal{F}^{\text{int}}(\mathbf{u}; \mathbf{w}) - \mathcal{F}^{\text{ext}}(\mathbf{u}; \mathbf{w}), \quad (8.27)$$

where

$$\mathcal{F}^{\text{ext}}(\mathbf{u}; \mathbf{w}) = \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV + \int_{S_T} \bar{\mathbf{T}}^D \cdot \mathbf{w} \, dS \quad (8.28)$$

is the virtual work of external forces, which may depend on  $\mathbf{u}$  as already mentioned, and

$$\mathcal{F}^{\text{int}}(\mathbf{u}; \mathbf{w}) = - \int_{\Omega} \mathbf{P} : \nabla \mathbf{w} \, dV = - \int_{\Omega} \left( \rho \frac{\partial \phi}{\partial \mathbf{F}}(\mathbf{F}[\mathbf{u}]) \right) : \nabla \mathbf{w} \, dV \quad (8.29)$$

is the virtual work of internal forces, which is clearly nonlinear in  $\mathbf{u}$ . Using (8.19), the latter can alternatively be expressed by using the derivative of the free energy with respect to the Green-Lagrange tensor:

$$\begin{aligned} \mathcal{F}^{\text{int}}(\mathbf{u}; \mathbf{w}) &= - \int_{\Omega} \left( \rho \mathbf{F}[\mathbf{u}] \cdot \frac{\partial \phi}{\partial \mathbf{E}}(\mathbf{E}[\mathbf{u}]) \right) : \nabla \mathbf{w} \, dV \\ &= - \int_{\Omega} \left( \rho \frac{\partial \phi}{\partial \mathbf{E}}(\mathbf{E}[\mathbf{u}]) \right) : \left( \mathbf{F}[\mathbf{u}]^T \cdot \nabla \mathbf{w} \right) \, dV \\ &= - \int_{\Omega} \mathbf{S}[\mathbf{u}] : \left( \mathbf{F}[\mathbf{u}]^T \cdot \nabla \mathbf{w} \right) \, dV. \end{aligned} \quad (8.30)$$

As no direct numerical solution method is available for problem (8.26), an iterative algorithm has to be developed. Among these, Newton-type methods, whose general features were presented in Section 8.2, are expected to be efficient by virtue of their known quadratic convergence.

**Comments:** The mathematical analysis of problems such as (8.26) is not an easy task. Concerning the conditions under which a solution exists, the interested reader may refer to the work of Ball (1977). Moreover, there are situations (such as the buckling of a beam) where multiple solutions exist. A comprehensive mathematical treatment of nonlinear elasticity is given in Ciarlet (1988).

### 8.3.4 Newton Method: linearized weak formulation

The Newton iterative method applied to the equilibrium problem defined by (8.26), (8.27) rests upon linearizing the residual  $\mathcal{R}(\mathbf{u}; \mathbf{w})$  about the current iterate  $\mathbf{u}^{[k]}$ :

$$\mathcal{R}(\mathbf{u}^{[k+1]}; \mathbf{w}) = \mathcal{R}(\mathbf{u}^{[k]}; \mathbf{w}) + \langle \mathcal{R}'(\mathbf{u}^{[k]}; \mathbf{w}), \delta \mathbf{u}^{[k]} \rangle + o(\|\delta \mathbf{u}^{[k]}\|),$$

where  $\delta \mathbf{u}^{[k]} := \mathbf{u}^{[k+1]} - \mathbf{u}^{[k]} \in \mathcal{C}(\mathbf{0})$  is the correction between two subsequent iterates and the tangent linear operator  $\mathcal{R}'$  of  $\mathcal{R}$  is defined through

$$\mathcal{R}(\mathbf{v} + \mathbf{z}, \mathbf{w}) - \mathcal{R}(\mathbf{v}, \mathbf{w}) = \langle \mathcal{R}'(\mathbf{v}, \mathbf{w}), \mathbf{z} \rangle + o(\|\mathbf{z}\|)$$

The new iterate

$$\mathbf{u}^{[k+1]} = \mathbf{u}^{[k]} + \delta \mathbf{u}^{[k]}$$

is then obtained by solving the weak formulation (8.26) in linearized form, i.e. the linear problem

Find  $\delta \mathbf{u}^{[k]} \in \mathcal{C}(\mathbf{0})$  such that

$$\langle \mathcal{R}'(\mathbf{u}^{[k]}; \mathbf{w}), \delta \mathbf{u}^{[k]} \rangle = -\mathcal{R}(\mathbf{u}^{[k]}; \mathbf{w}) \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (8.31)$$

Note that solving problem (8.26) requires the choice of an initialization  $\mathbf{u}^{[0]}$  that is kinematically admissible ( $\mathbf{u}^{[0]} \in \mathcal{C}(\mathbf{u}^D)$ ). The Newton method for solving problem (8.26) can finally be summarized by the algorithm of Box 8.2:

**Box 8.2:** Newton algorithm for nonlinear elasticity

1. Initialization: choice of  $\mathbf{u}^{[0]} \in \mathcal{C}(\mathbf{u}^D)$

2. Iterate ( $k \rightarrow k + 1$ ):

(i) Solve the linear problem (8.31), i.e.:

Find  $\delta \mathbf{u}^{[k]} \in \mathcal{C}(\mathbf{0})$  such that

$$\langle \mathcal{R}'(\mathbf{u}^{[k]}; \mathbf{w}), \delta \mathbf{u}^{[k]} \rangle = -\mathcal{R}(\mathbf{u}^{[k]}; \mathbf{w}) \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}).$$

(ii) Update the solution:  $\mathbf{u}^{[k+1]} = \mathbf{u}^{[k]} + \delta \mathbf{u}^{[k]}$

### 8.3.5 Explicit form of the tangent linear operator

In the definition (8.27) of the residual,  $\mathcal{F}^{\text{int}}(\mathbf{u}; \mathbf{w})$  is not necessarily the only displacement-dependent quantity. Indeed, the given loads  $\mathbf{f}$  and  $\bar{\mathbf{T}}^D$  may also depend on  $\mathbf{u}$ , if only as a result of the mapping of the weak formulation to the reference configuration. Applied pressure is another typical example of a displacement-dependent *follower force*. Using the definition (8.27) of the residual, the tangent linear operator is thus also split into two parts:

$$\langle \mathcal{R}'(\mathbf{u}; \mathbf{w}), \delta \mathbf{u} \rangle = -\langle \mathcal{F}^{\text{int}'}(\mathbf{u}; \mathbf{w}), \delta \mathbf{u} \rangle - \langle \mathcal{F}^{\text{ext}'}(\mathbf{u}; \mathbf{w}), \delta \mathbf{u} \rangle. \quad (8.32)$$

The explicit form of  $\mathcal{F}^{\text{ext}'}$  depends on prescribed load  $\mathbf{f}$  and  $\bar{\mathbf{T}}^D$ . For simplicity, we restrict attention to the case where  $\mathcal{F}^{\text{ext}}$  does not depend on the displacement.

The explicit form of the tangent linear operator  $\mathcal{F}^{\text{int}'}$  can be established by means of a first-order Taylor expansion with respect to  $\mathbf{z}$  of  $\mathcal{F}^{\text{int}}(\mathbf{v} + \mathbf{z}; \mathbf{w})$ , with  $\mathcal{F}^{\text{int}}$  as defined by (8.29). To begin, the first-order expansion of  $\partial_{\mathbf{F}} \phi(\mathbf{F}[\mathbf{v} + \mathbf{z}])$  about  $\mathbf{v}$  reads:

$$\frac{\partial \phi}{\partial \mathbf{F}}(\mathbf{F}[\mathbf{v} + \mathbf{z}]) = \frac{\partial \phi}{\partial \mathbf{F}}(\mathbf{F}[\mathbf{v}]) + \frac{\partial^2 \phi}{\partial \mathbf{F} \partial \mathbf{F}}(\mathbf{F}[\mathbf{v}]) : \nabla \mathbf{z} + o(\|\nabla \mathbf{z}\|)$$

Using the above expansion in (8.29) with  $\mathbf{v} = \mathbf{u}^{[k]}$  and  $\mathbf{z} = \delta \mathbf{u}^{[k]}$ , one obtains

$$-\langle \mathcal{F}^{\text{int}'}(\mathbf{u}^{[k]}; \mathbf{w}), \delta \mathbf{u}^{[k]} \rangle = \int_{\Omega} \rho \left( \frac{\partial^2 \phi}{\partial \mathbf{F} \partial \mathbf{F}} : \nabla \delta \mathbf{u}^{[k]} \right) : \nabla \mathbf{w} \, dV. \quad (8.33)$$

It can be more convenient to alternatively express  $\mathcal{F}^{\text{int}'}$  in terms of the derivatives of the free energy with respect to the Green-Lagrange tensor, by taking the first-order Taylor expansion with respect to  $\mathbf{z}$  of the expression (8.30) of  $\mathcal{F}^{\text{int}}(\mathbf{v} + \mathbf{z}; \mathbf{w})$ . We start by the first-order expansion of  $\mathbf{E}[\mathbf{v} + \mathbf{z}]$  about  $\mathbf{v}$ :

$$\mathbf{E}[\mathbf{v} + \mathbf{z}] = \mathbf{E}[\mathbf{v}] + \frac{1}{2} \left( \mathbf{F}[\mathbf{v}]^T \cdot \nabla \mathbf{z} + \nabla \mathbf{z}^T \cdot \mathbf{F}[\mathbf{v}] \right) + o(\|\nabla \mathbf{z}\|),$$

from which, using the symmetry of  $\partial_{\mathbf{E}\mathbf{E}}^2 \phi$ , the first-order expansion of  $\partial_{\mathbf{E}} \phi$  about  $\mathbf{v}$  can be obtained as

$$\frac{\partial \phi}{\partial \mathbf{E}}(\mathbf{E}[\mathbf{v} + \mathbf{z}]) = \frac{\partial \phi}{\partial \mathbf{E}}(\mathbf{E}[\mathbf{v}]) + \frac{\partial^2 \phi}{\partial \mathbf{E} \partial \mathbf{E}}(\mathbf{E}[\mathbf{v}]) : \left( \mathbf{F}[\mathbf{v}]^T \cdot \nabla \mathbf{z} \right) + o(\|\nabla \mathbf{z}\|).$$

Lastly, using

$$\mathbf{F}[\mathbf{v} + \mathbf{z}] = \mathbf{F}[\mathbf{v}] + \nabla \mathbf{z},$$

and setting again  $\mathbf{v} = \mathbf{u}^{[k]}$  and  $\mathbf{z} = \delta \mathbf{u}^{[k]}$ , we arrive at

$$\begin{aligned} -\langle \mathcal{F}^{\text{int}'}(\mathbf{u}^{[k]}; \mathbf{w}), \delta \mathbf{u}^{[k]} \rangle &= \int_{\Omega} (\mathcal{A}^{[k]} : (\mathbf{F}^{[k]T} \cdot \nabla \delta \mathbf{u}^{[k]})) : (\mathbf{F}^{[k]T} \cdot \nabla \mathbf{w}) \, dV \\ &\quad + \int_{\Omega} \mathbf{S}^{[k]} : (\nabla \delta \mathbf{u}^{[k]T} \cdot \nabla \mathbf{w}) \, dV \end{aligned} \quad (8.34)$$

where the fourth-order tensor

$$\mathcal{A}^{[k]}(\mathbf{x}) := \rho \frac{\partial^2 \phi}{\partial \mathbf{E} \partial \mathbf{E}}(\mathbf{E}[\mathbf{u}^{[k]}](\mathbf{x})) \quad (8.35)$$

has the same symmetry (1.9) as the elastic tensor  $\mathcal{A}$ , the second-order tensor  $\mathbf{S}^{[k]}(\mathbf{x})$  is the Piola stress tensor associated with the Green-Lagrange strain tensor  $\mathbf{E}[\mathbf{u}^{[k]}](\mathbf{x})$ :

$$\mathbf{S}^{[k]}(\mathbf{x}) = \rho \frac{\partial \phi}{\partial \mathbf{E}}(\mathbf{E}[\mathbf{u}^{[k]}](\mathbf{x})), \quad (8.36)$$

and the second-order tensor  $\mathbf{F}^{[k]}(\mathbf{x})$  is the deformation gradient tensor:

$$\mathbf{F}^{[k]}(\mathbf{x}) = \mathbf{I} + \nabla \mathbf{u}^{[k]}(\mathbf{x}). \quad (8.37)$$

### 8.3.6 Finite element approximation

Following the principles and notations developed in Chapters 2 and 3, a finite element approximation is now introduced for the solid  $\Omega$  (initial configuration), the unknown displacement field  $\mathbf{u}$ , the correction field  $\delta \mathbf{u}$  and the virtual fields  $\mathbf{w}$  so as to define the finite element problem associated with the linearized weak form (8.31). This leads to

$$[\mathbb{K}^{[k]}] \{ \delta \mathbb{U}^{[k]} \} = - \{ \mathbb{R}^{[k]} \}. \quad (8.38)$$

The unknown  $\{\delta\mathbb{U}^{[k]}\}$  is the global nodal displacement vector associated with the finite element approximation of the correction  $\delta\mathbf{u}_h^{[k]} \in \mathcal{C}_h(\mathbf{0})$ . Note that it only contains nodal degrees of freedom that are not prescribed by boundary conditions. The right-hand side  $\{\mathbb{R}^{[k]}\} = \{\mathbb{R}(\{\mathbb{U}^{[k]}\})\}$  is the finite element approximation of the residual  $\mathcal{R}(\mathbf{u}_h^{[k]}; \mathbf{w})$ :

$$\begin{aligned} \mathcal{R}(\mathbf{u}_h^{[k]}; \mathbf{w}) &= \int_{\Omega} \mathbf{S}^{[k]} : (\mathbf{F}^{[k]\top} \cdot \nabla \mathbf{w}) \, dV - \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{w} \, dV - \int_{S_T} \bar{\mathbf{T}}^D \cdot \mathbf{w} \, dS \\ &= \{\mathbb{W}\}^T \{\mathbb{R}^{[k]}\} \end{aligned} \quad (8.39)$$

where  $\{\mathbb{W}\}$  is the global nodal vector associated with the virtual field  $\mathbf{w} \in \mathcal{C}_h(\mathbf{0})$ . It may be conveniently split into two parts according to

$$-\{\mathbb{R}^{[k]}\} = \{\mathbb{F}^{\text{int}[k]}\} + \{\mathbb{F}^{\text{ext}}\}, \quad (8.40)$$

where  $\{\mathbb{F}^{\text{ext}}\} = \{\mathbb{F}^{\text{vol}}\} + \{\mathbb{F}^{\text{surf}}\}$  is the vector of nodal forces associated with given body forces  $\mathbf{f}$  and surface forces  $\bar{\mathbf{T}}^D$ . As we here assume the external forces to be independent on the configuration, their finite element representation is exactly identical to that developed for prescribed body and surface force densities in Chapter 3.

The vector  $\{\mathbb{F}^{\text{int}[k]}\}$  is the nodal vector of internal forces (generalized forces) associated with the Piola stress tensor field  $\mathbf{S}^{[k]}$ . Using the symmetry of  $\mathbf{S}^{[k]}$ , its expression can be modified to

$$-\int_{\Omega} \mathbf{S}^{[k]} : (\mathbf{F}^{[k]\top} \cdot \nabla \mathbf{w})_{\text{sym}} \, dV = \{\mathbb{W}\}^T \{\mathbb{F}^{\text{int}[k]}\}, \quad (8.41)$$

with  $\mathbf{A}_{\text{sym}} := \frac{1}{2}(\mathbf{A} + \mathbf{A}^T)$  denoting the symmetric part of a second-order tensor  $\mathbf{A}$ .

The matrix  $[\mathbb{K}^{[k]}] = [\mathbb{K}(\{\mathbb{U}^{[k]}\})]$  is the tangent stiffness matrix whose expression can also be simplified using symmetry properties of  $\mathcal{A}^{[k]}$  and  $\mathbf{S}^{[k]}$

$$\begin{aligned} -\langle \mathcal{F}^{\text{int}'}(\mathbf{u}_h^{[k]}; \mathbf{w}), \delta\mathbf{u}_h^{[k]} \rangle &= \int_{\Omega} (\mathbf{F}^{[k]\top} \cdot \nabla \delta\mathbf{u}_h^{[k]})_{\text{sym}} : \mathcal{A}^{[k]} : (\mathbf{F}^{[k]\top} \cdot \nabla \mathbf{w})_{\text{sym}} \, dV \\ &\quad + \int_{\Omega} \mathbf{S}^{[k]} : (\nabla^T \delta\mathbf{u}_h^{[k]} \cdot \nabla \mathbf{w})_{\text{sym}} \, dV \\ &= \{\mathbb{W}\}^T [\mathbb{K}^{[k]}] \{\delta\mathbb{U}^{[k]}\}. \end{aligned} \quad (8.42)$$

Note again that if  $\mathcal{F}^{\text{ext}}$  depends on  $\mathbf{u}$ , the "exact" tangent matrix then features an extra additive term, not given here, arising from  $\mathcal{F}^{\text{ext}'}$ .

The tangent matrix  $[\mathbb{K}^{[k]}]$  and the internal force vector  $\{\mathbb{F}^{\text{int}[k]}\}$  obviously have a similar structure as their linear elastic counterparts. Their computation is also based on the assemblage of element-level contributions (see Section 8.3.7). The assemblage procedure in itself is identical to that developed in Chapter 3.



### 8.3.7 Element contributions to tangent matrix and internal force vector

We may again introduce matrix notation based on the engineering notation (3.9) introduced in Chapter 3. At the element level, similarly to what was done in Chapter 3 to express the relationship (3.12) between the strain  $\varepsilon[\mathbf{w}]$  and the nodal displacements on an element, and observing that  $(\mathbf{F}^T \cdot \nabla \mathbf{w})_{\text{sym}}$  is linear in  $\mathbf{w}$ , one can build a list collecting the components of the symmetric tensor  $(\mathbf{F}^T \cdot \nabla \mathbf{w})_{\text{sym}}$  according to the engineering notation adopted for strain tensors and express it as:

$$(\mathbf{F}^T \cdot \nabla \mathbf{w})_{\text{sym}} \rightarrow [B_{\text{nl}}(\mathbf{a})]\{W_e\}. \quad (8.43)$$

Accordingly,

$$\mathbf{S}^{[k]} : (\mathbf{F}^{[k]T} \cdot \nabla \mathbf{w})_{\text{sym}} = \{W_e\}^T [B_{\text{nl}}^{[k]}(\mathbf{a})]^T \{S^{[k]}\},$$

where  $\{S^{[k]}\}$  collects the components of the symmetric Piola stress tensor  $\mathbf{S}^{[k]}$  in engineering notation for stress tensors. The internal force vector at the element level can then be obtained as

$$\begin{aligned} \{F_e^{\text{int}[k]}\} &= - \int_{E_e} [B_{\text{nl}}^{[k]}(\mathbf{a})]^T \{S^{[k]}\} dV \\ &= - \int_{\Delta_e} [B_{\text{nl}}^{[k]}(\mathbf{a})]^T \{S^{[k]}\} J(\mathbf{a}) dV(\mathbf{a}) \\ &\approx - \sum_{g=1}^G w_g [B_{\text{nl}}^{[k]}(\mathbf{a}_g)]^T \{S^{[k]}(\mathbf{a}_g)\} J(\mathbf{a}_g), \end{aligned} \quad (8.44)$$

where the last approximation results from the use of a quadrature rule (3.20).

Similarly, introducing the matrix representation  $[A^{[k]}(\mathbf{a})]$  of the local tangent moduli  $\mathcal{A}^{[k]}$  induced by the adopted engineering notation, we obtain

$$\begin{aligned} (\mathbf{F}^{[k]T} \cdot \nabla \delta \mathbf{u}_h^{[k]})_{\text{sym}} : \mathcal{A}^{[k]} : (\mathbf{F}^{[k]T} \cdot \nabla \mathbf{w})_{\text{sym}} \\ = \{W_e\}^T [B_{\text{nl}}^{[k]}(\mathbf{a})]^T [A^{[k]}(\mathbf{a})] [B_{\text{nl}}^{[k]}(\mathbf{a})] \{\delta \mathbf{u}_h^{[k]}\} \end{aligned}$$

Observing that  $(\mathbf{u}_h, \mathbf{w}) \mapsto \mathbf{S} : (\nabla \mathbf{u}_h^T \cdot \nabla \mathbf{w})_{\text{sym}}$  is a symmetric bilinear form (because of the symmetry of the Piola tensor  $\mathbf{S}$ ), one can write

$$\mathbf{S} : (\nabla \mathbf{u}_h^T \cdot \nabla \mathbf{w})_{\text{sym}} = \{W_e\}^T [K_g^{[k]}(\mathbf{a})] \{U_e\}.$$

The tangent matrix at the element level can then be obtained as

$$\begin{aligned} [K_e^{[k]}] &= \int_{E_e} \left( [B_{\text{nl}}^{[k]}(\mathbf{a})]^T [A^{[k]}(\mathbf{a})] [B_{\text{nl}}^{[k]}(\mathbf{a})] + [K_g^{[k]}(\mathbf{a})] \right) dV \\ &= \int_{\Delta_e} \left( [B_{\text{nl}}^{[k]}(\mathbf{a})]^T [A^{[k]}(\mathbf{a})] [B_{\text{nl}}^{[k]}(\mathbf{a})] + [K_g^{[k]}(\mathbf{a})] \right) J(\mathbf{a}) dV \\ &\approx \sum_{g=1}^G w_g \left( [B_{\text{nl}}^{[k]}(\mathbf{a}_g)]^T [A^{[k]}(\mathbf{a}_g)] [B_{\text{nl}}^{[k]}(\mathbf{a}_g)] + [K_g^{[k]}(\mathbf{a}_g)] \right) J(\mathbf{a}_g). \end{aligned} \quad (8.45)$$

where the last approximation results again from the use of a quadrature rule (3.20).

### 8.3.8 Application: Saint-Venant Kirchhoff model for a T6 element

As an application, we discuss how to compute the element tangent matrix and internal force vector of a T6 element for the Saint-Venant Kirchhoff model (8.21) (plane strain assumption). The constitutive relation reads

$$\mathbf{S}[\mathbf{u}_h](\mathbf{x}) = \rho \frac{\partial \phi}{\partial \mathbf{E}}(\mathbf{E}[\mathbf{u}_h](\mathbf{x})) = \mathcal{A} : \mathbf{E}[\mathbf{u}_h](\mathbf{x}),$$

and the tensor  $\mathcal{A}$  of local tangent elastic moduli, given by

$$\mathcal{A} = \rho \frac{\partial^2 \phi}{\partial \mathbf{E} \partial \mathbf{E}}$$

is here a constant fourth-order tensor. Using engineering notation, the constitutive law becomes

$$\{S\} = [A]\{E\}. \quad (8.46)$$

where  $\{S\}$  (resp.  $\{E\}$ ) collects the components of the symmetric Piola stress tensor  $\mathbf{S}$  (resp. the symmetric Green-Lagrange strain tensor  $\mathbf{E}$ ) according to the engineering notation for stress tensors (resp. strain tensors).

We are now ready to describe the function `T6_2A_solid_KTeNL` performing the computation of the element tangent matrix and the element internal force vector. In addition to the matrix  $\mathbf{X}$  of nodal coordinates and the list `mate` holding materials parameters (Young's modulus, Poisson ratio), input variables contain the nodal values of the displacement  $\{U_e\} \rightarrow \mathbf{U}_e$ . Output variables are the element tangent matrix  $[K_e^{[k]}] \rightarrow \mathbf{K}_e$  and the element internal force vector  $\{F_e^{\text{int}[k]}\} \rightarrow \mathbf{F}_e$ . They are initially set to zero.

```
function [Ke, Fe]=T6_2A_solid_KTeNL(X,mate,Ue)
```

```

E=mate(1);
nu=mate(2);
A=E/((1+nu)*(1-2*nu))*[1-nu,nu,0;
                        nu,1-nu,0;
                        0,0,(1-2*nu)/2];
a_gauss=1/6*[4 1 1; 1 4 1; 1 1 4];           % Gauss abscissae
w_gauss=[1/6 1/6 1/6];                       % Gauss weights
Ke=zeros(12,12);
Fe=zeros(12,1);
```

The loop over Gauss points can now begin<sup>7</sup>. Up to the definition of the  $[G]$  matrix that contains the gradients of shape functions, the code lines are identical to those in the

<sup>7</sup>Due to the nonlinear term in the Green-Lagrange strain tensor (8.16), the three-point Gauss rule employed in 2D elasticity is no longer complete. A six-point rule, which integrates exactly polynomials of order 4, could instead be implemented in the element-level routines

linear elastic counterpart (function T6\_2A.solid\_Ke).

```

for g=1:length(w_gauss),
    a=a_gauss(g,:);
    D=[4*a(1)-1 0 -4*a(3)+1 4*a(2)...
        -4*a(2) 4*(a(3)-a(1));
        0 4*a(2)-1 -4*a(3)+1 4*a(1) ...
        4*(a(3)-a(2)) -4*a(1)]';
    J=X'*D;
    detJ=J(1,1)*J(2,2)-J(1,2)*J(2,1);
    invJ=1/detJ*[ J(2,2) -J(1,2); ...
                  -J(2,1) J(1,1)];
    G=D*invJ;

... see below ...

end

```

We continue by providing the expression of the matrix  $[G_V]$  defined by (see also 2.37)

$$\{\nabla u_h\} = \begin{Bmatrix} \partial u_{h1}/\partial x_1 \\ \partial u_{h1}/\partial x_2 \\ \partial u_{h2}/\partial x_1 \\ \partial u_{h2}/\partial x_2 \end{Bmatrix} = [G_V]\{U_e\}$$

in terms of the matrix  $[G]$  containing the gradient of shape functions:

```

GV=[G(1,1) 0 G(2,1) 0 G(3,1) 0 G(4,1) 0 G(5,1) 0 G(6,1) 0;
    G(1,2) 0 G(2,2) 0 G(3,2) 0 G(4,2) 0 G(5,2) 0 G(6,2) 0;
    0 G(1,1) 0 G(2,1) 0 G(3,1) 0 G(4,1) 0 G(5,1) 0 G(6,1);
    0 G(1,2) 0 G(2,2) 0 G(3,2) 0 G(4,2) 0 G(5,2) 0 G(6,2)];

```

We then compute the associated Green-Lagrange strain tensor and the Piola stress tensor given by the Saint-Venant-Kirchhoff model (8.46):

```

gradU=GV*Ue;
e=[gradU(1)+.5*gradU(1)^2+.5*gradU(3)^2, ...
   gradU(4)+.5*gradU(2)^2+.5*gradU(4)^2, ...
   gradU(2)+gradU(3)+...
   grad(1)*gradU(2)+gradU(3)*gradU(4)]';
SK=A*e;

```

We now express the matrix  $[B_{nl}]$  defined by eq. (8.43). First, observing that

$$(\mathbf{F}^T \cdot \nabla \mathbf{w})_{\text{sym}} = \varepsilon[\mathbf{w}] + (\nabla \mathbf{u}_h^T \cdot \nabla \mathbf{w})_{\text{sym}},$$

from (8.43) and (3.12) it follows that

$$[B_{nl}(\mathbf{a})]\{W_e\} = ([B(\mathbf{a})] + [B_u(\mathbf{a})])\{W_e\}$$

where the expression for  $[B_u]$  is developed hereafter. Noting that we have

$$\frac{\partial w_m}{\partial x_1} \frac{\partial u_{hm}}{\partial x_1} = \frac{\partial u_{h1}}{\partial x_1} \frac{\partial w_1}{\partial x_1} + \frac{\partial u_{h2}}{\partial x_1} \frac{\partial w_2}{\partial x_1} = \left\{ \frac{\partial u_{h1}}{\partial x_1} [G_V]_{1,:} + \frac{\partial u_{h2}}{\partial x_1} [G_V]_{3,:} \right\} \{W_e\}$$

$$\begin{aligned}\frac{\partial w_m}{\partial x_1} \frac{\partial u_m}{\partial x_2} &= \frac{\partial u_{h1}}{\partial x_1} \frac{\partial w_1}{\partial x_2} + \frac{\partial u_{h2}}{\partial x_1} \frac{\partial w_2}{\partial x_2} = \left\{ \frac{\partial u_{h1}}{\partial x_1} [G_V]_{2,:} + \frac{\partial u_{h2}}{\partial x_1} [G_V]_{4,:} \right\} \{W_e\}, \\ \frac{\partial w_m}{\partial x_2} \frac{\partial u_m}{\partial x_1} &= \frac{\partial u_{h1}}{\partial x_2} \frac{\partial w_1}{\partial x_1} + \frac{\partial u_{h2}}{\partial x_2} \frac{\partial w_2}{\partial x_1} = \left\{ \frac{\partial u_{h1}}{\partial x_2} [G_V]_{1,:} + \frac{\partial u_{h2}}{\partial x_2} [G_V]_{3,:} \right\} \{W_e\}, \\ \frac{\partial w_m}{\partial x_2} \frac{\partial u_m}{\partial x_2} &= \frac{\partial u_{h1}}{\partial x_2} \frac{\partial w_1}{\partial x_2} + \frac{\partial u_{h2}}{\partial x_2} \frac{\partial w_2}{\partial x_2} = \left\{ \frac{\partial u_{h1}}{\partial x_2} [G_V]_{2,:} + \frac{\partial u_{h2}}{\partial x_2} [G_V]_{4,:} \right\} \{W_e\},\end{aligned}$$

where  $[G_V]_{i,:}$  denotes the  $i$ th line of  $[G_V]$ . We deduce the corresponding MATLAB code lines

```
B=[GV(1,:); GV(4,:); GV(2,:)+GV(3,:)];
Bu=[gradU(1)*GV(1,:)+gradU(3)*GV(3,:);
    gradU(2)*GV(2,:)+gradU(4)*GV(4,:);
    gradU(1)*GV(2,:)+gradU(3)*GV(4,:)+ ...
    gradU(2)*GV(1,:)+gradU(4)*GV(3,:)];
Bnl=B+Bu;
```

Similarly, upon developing

$$\begin{aligned}S:(\nabla \mathbf{u}_h^T \cdot \nabla \mathbf{w}) &= \frac{\partial w_1}{\partial x_1} (S_{11} \frac{\partial u_{h1}}{\partial x_1} + S_{12} \frac{\partial u_{h1}}{\partial x_2}) + \frac{\partial w_1}{\partial x_2} (S_{22} \frac{\partial u_{h1}}{\partial x_2} + S_{12} \frac{\partial u_{h1}}{\partial x_1}) \\ &\quad + \frac{\partial w_2}{\partial x_1} (S_{11} \frac{\partial u_{h2}}{\partial x_1} + S_{12} \frac{\partial u_{h2}}{\partial x_2}) + \frac{\partial w_2}{\partial x_2} (S_{22} \frac{\partial u_{h2}}{\partial x_2} + S_{12} \frac{\partial u_{h2}}{\partial x_1}),\end{aligned}$$

we deduce the expression of the  $[K_g]$  matrix:

$$S:(\nabla \mathbf{u}_h^T \cdot \nabla \mathbf{w}) = \{\nabla \mathbf{w}\}^T [B_g] \{U_e\} = \{W_e\}^T [G_V]^T [B_g] \{U_e\} = \{W_e\}^T [K_g] \{U_e\}$$

and the MATLAB code lines

```
Bg=[SK(1)*GV(1,:)+SK(3)*GV(2,:);
    SK(3)*GV(1,:)+SK(2)*GV(2,:);
    SK(1)*GV(3,:)+SK(3)*GV(4,:);
    SK(3)*GV(3,:)+SK(2)*GV(4,:)];
Kg=GV'*Bg;
```

It only remains to add contributions to the internal force (resp. the tangent stiffness matrix associated with the Saint-Venant-Kirchhoff model) according to (8.44) (resp. (8.45))

```
Fe=Fe+Bnl'*S*detJ*w_gauss(g); % internal force contribution
Ke=Ke+(Bnl'*A*Bnl+Kg)*detJ*w_gauss(g); % tangent stiff matrix contr.
```

## 8.4 Applications and exercises

In this section the numerical solving of the formulation presented previously (Section 8.3) is developed for different applications. We aim at demonstrating the importance of taking finite deformations into account properly even when the strains remain small. The chosen assumptions (small strain but possibly large deformation, and in particular large rotations) justify adopting the Saint-Venant-Kirchhoff constitutive model (8.21) for the material.

Section 8.4.1 introduces the code `geomNL`, dedicated to solving nonlinear elasticity problems by an *incremental* Newton-Raphson algorithm. Two applications are then presented in the form of exercises. The first one deals with the buckling problem of a beam (Section 8.4.2). The second is devoted to the numerical simulation of a 2D contact problem of a tire rolling down a slope (Section 8.4.3).

#### 8.4.1 The code `geomNL` for geometric nonlinearities

The code `geomNL` is dedicated to solving nonlinear elasticity problems by the Newton-Raphson algorithm developed in Section 8.3.4. It uses the same structures as the linear code `genlin`. The constitutive law implemented is the isotropic Saint-Venant-Kirchhoff model (8.21). As a consequence, `geomNL` can be used for analyses involving large deformations (large rotations) but small strains. The buckling of a beam, presented and analysed in Section 8.4.2, is a typical illustration of such situations. For problems involving large strains, other constitutive models should be implemented and used.

The convergence of the Newton-Raphson algorithm is known to be strongly dependent on the quality of its initialization. As a consequence, we have implemented a simple incremental procedure for computing the final equilibrium position, which consists in progressively increasing the loading (external forces and prescribed displacement). An equilibrium position characterized by displacements  $\mathbf{u}_n$  is found for each load level of an increasing sequence of loadings, by means of the Newton-Raphson algorithm, and serves as an initialization for solving the next load increment. The incremental loading is defined by the sequence  $\lambda_n$  assigned by the user through the array `lambda` in the analysis file. It represents a sequence of multiplicative load parameters starting from zero and with the actual load level of interest corresponding to the final unit value. For example:

```
lambda = [0. 0.1 0.3 0.5 0.6 1.];
```

Note that, for the applications envisaged in this chapter, we choose to increment only the prescribed displacements while the external forces and tractions, if any, are constant through the analysis. To this purpose we express displacements prescribed at the  $n$ -th loading increment as  $\mathbf{u}_n^D = \lambda_n \mathbf{u}^D$ . Hence, given the numerical solution  $\mathbf{u}_{h,n}$  of the  $n$ -th loading increment, the initialisation of the subsequent increment will be

$$\mathbf{u}_{h,n+1}^{[0]} = \mathbf{u}_{h,n} + \Delta\lambda_n \mathbf{u}_h^{(D)} \quad \text{with} \quad \Delta\lambda_n = \lambda_{n+1} - \lambda_n \quad (8.47)$$

It is worth noting that the admissibility of such an initialisation requires  $\Delta\lambda_n$  to be smaller than the size of finite elements connected to  $S_u$ . With the exception of the array `lambda`, the input file is identical to that required for a classical linear elastic analysis (see Appendix B for details). The first part of the code (up to the line `Solution` phase) is similar to the code `genlin`; reading it is left as an exercise. Then, displacement boundary conditions are saved and both the internal force vector and the tangent matrix are set to zero:

```
for n=1:analysis.NN
    nodes(n).UD(1:ndof)=nodes(n).U(1:ndof); % saves prescribed displ
    nodes(n).U(1:ndof)=zeros(1:ndof); % resets displacements
end

Fint=zeros(analysis.neq,1); % init. internal forces
KT=spalloc(analysis.neq,analysis.neq,ncoeffs); % allocates sparse matrix
```

The loop over the load sequence can begin. The external force vector  $F_{ext}$  and the displacement field are initialized.

```

numstep = length(lambda)-1;           % number of load steps
for step=1:numstep,                   % loop over load steps

    Fext=F;                           % Fext due to constant loads

    Dlambda=lambda(step+1)-lambda(step); % inc. of loading factor
    for n=1:analysis.NN               % displacement initialization ...
        nodes(n).U(:)=nodes(n).U(:)+ ... % accounting for increment ...
            Dlambda*nodes(n).UD(:);    % of displ. BC
    end

    iter=0;                           % resets iteration counter
    resid=1;                           % fictitious initial value of res
    toll=1.d-8;                         % tolerance on residuum
    while resid > toll,                 % Newton-Raphson procedure ..
        iter=iter+1;                   % for the nth load increment

        ... Newton-Raphson iterations ...

    end                                % end of Newton algorithm
end                                    % end of load sequence

```

Newton-Raphson iterations can now start. The tangent matrix and the internal force vector are reset to zero.

```

KT(:,:)=0.d0;                         % resets to zero tangent matrix KT
Fint(:)=0;                             % resets to zero internal force Fint

```

This is followed by the computation (assembly process) of the tangent matrix and internal force vector. The assembly process is similar to that used for linear elasticity.

```

for e=1:analysis.NE,                  % loop over elements

    ... standard assemblage procedures...

    Ue=zeros(Dne,1);
    pos=1;
    for n=1:ne
        node=elements(e).nodes(n);
        Xe(n,:)=nodes(node).coor;
        Ge(pos:pos+ndof-1)=nodes(node).dof;
        Ue(pos:pos+ndof-1)=nodes(node).U;    % gets element displacements
        pos=pos+ndof;
    end
    [KTe, Finte]=eval([Etag Atag 'KTeNL'... % element tangent stiff matrix
        '(Xe, material(mat,:), Ue)']); % and vector of internal forces
    Le0=find(Ge>0);                        % local numbering of unknowns
    Ie=Ge(Le0);                             % global numbering
    Fint(Ie)=Fint(Ie)+Finte(Le0);           % assembling internal forces
    KT(Ie,Ie)=KT(Ie,Ie)+KTe(Le0,Le0);      % matrix assemblage
end

```

The final residual (8.40) is computed, and the linear system (8.38) solved. Lastly, the

displacement is updated.

```

R=-Fext-Fint;                % residuum at prev iteration
DU=-KT\R;                    % solves linear problem
resid=sqrt(R'*R);

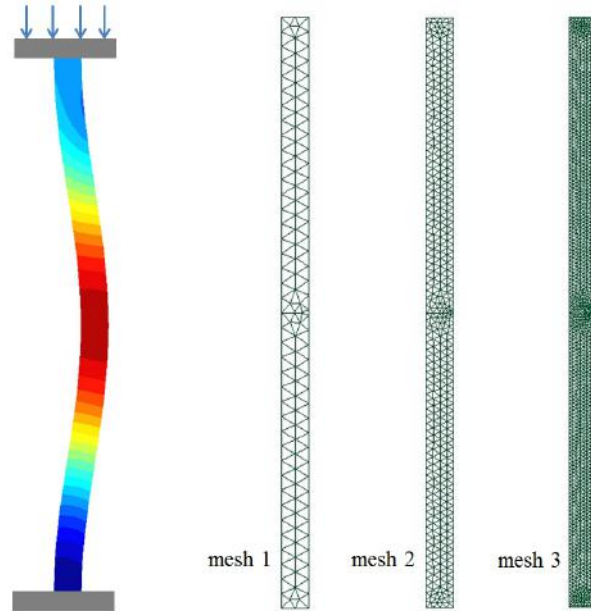
for n=1:analysis.NN          % updates displacements on nodes
    for dir=1:ndof
        dof=nodes(n).dof(dir);
        if dof>0              % finds active dofs
            nodes(n).U(dir)=nodes(n).U(dir)+DU(dof);
        end
    end
end
end

```

The displacement evolution is saved at every step in a history file which can be visualised using GMSH at the end of the analysis.

### 8.4.2 Buckling of a beam

Consider a doubly clamped beam of length  $L$  and thickness  $H$  ( $L \gg H$ ) under an imposed axial compression displacement on its upper part, as shown in Fig. 8.8.



**Figure 8.8:** Buckling of the beam: geometry and meshes

Let  $P$  be the compressive force necessary to impose a vertical displacement  $u^D$ . The theory of stability of slender structures predicts that simple compression without bending is a *stable* solution of the problem for  $P < P_c$ , with:

$$P_c = 4\pi^2 \frac{EI}{L^2}, \quad I = \frac{1}{12} H^3$$

where  $I$  is the flexural inertia of the beam's cross section and  $E$  is the Young modulus.

For  $P = P_c$  this solution still exists but is no longer stable; the beam bends sideways, i.e. buckles. However, we recall that this analytical result is established in the framework of the simplified theory of beams, which assumes the deformation to be such that the cross sections remain orthogonal to the mean fiber. The numerical solution presented here corresponds to a model that is richer than the beam theory.

In the numerical example, we chose  $E = 1$ ,  $H = 1$ ,  $L = 20$  (in consistent units). In addition, we slightly modify the problem by adding a small lateral force  $F$  which can be interpreted as an imperfection of the structure. Using the files Chap8\_buckling.\*, the user can modify the mesh refinement and the load increment.

**Exercise 8.1** (Computation of the compressive force). *Prove that the compressive force  $R$  exerted on the beam by the upper plate can be accurately computed (neglecting a term proportional to the lateral force  $F$  - usually  $F \ll R$  in the instability analysis) by*

$$R \delta \approx \int_{\Omega} \mathbf{S}[\mathbf{u}_h] : \left( \mathbf{F}[\mathbf{u}_h]^T \cdot \nabla \mathbf{u}_h \right)_{\text{sym}} dV \quad (8.48)$$

where  $\delta$  is the prescribed vertical displacement on the upper plate (Hint: resort to the form (8.24) of the virtual work principle, with  $\mathbf{w} = \mathbf{u}_h$  as virtual field).

Analyse the code of Box 8.3 and show how the quantity work is related to the force exerted. Place the code within geomNL.m at the correct location and analyse the convergence as the mesh is refined.

---

**Box 8.3:** Procedure for computing the compressive force in exercise 8.1

---

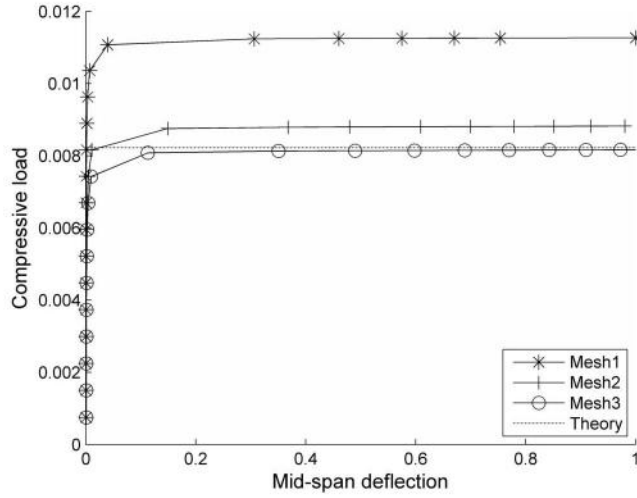
```
work=0.d0;
for e=1:analysis.NE,                                % loop over elements
    type=elements(e).type;
    Etag=Le(type).tag;
    ne=Le(type).ne;                                  % number of nodes in element
    Dne=ndof*ne;                                     % number of nodal values
    Xe=zeros(ne,DG);
    Ue=zeros(Dne,1);
    pos=1;
    for n=1:ne
        node=elements(e).nodes(n);
        Xe(n,:)=nodes(node).coor;                   % creates element
        Ue(pos:pos+ndof-1)=nodes(node).U;           % fills element nodal displ
        pos=pos+ndof;
    end
    mat=elements(e).mat;
    [KTe, Finte]=eval([Etag Atag 'KTeNL'...
        '(Xe, material(mat,:), Ue)']);
    work=work+Ue'*Finte;
end
```

---

Figure 8.9 plots the computed compressive force against the mid-span deflection for three different meshes shown in Fig 8.8. Results show the convergence of the compressive force to the beam theory value as the mesh is refined.



**Exercise 8.2** (Axisymmetric T6 element for Saint-Venant Kirchhoff model). *Develop an axisymmetric T6 element for a Saint-Venant Kirchhoff model.*



**Figure 8.9:** Buckling of a beam: results on three different meshes

### 8.4.3 2D contact problem of a tire rolling down a slope

Consider a tire rolling down a rigid inclined plane of slope  $\theta = \pi/4$  (Fig. 8.10). We assume that the constitutive material of the tire is elastic, isotropic in the reference configuration and that strains remains small (Saint-Venant-Kirchhoff model). Both gravity forces ( $\mathbf{f} = -ge_2$ ) and inertial forces are taken into account. A similar problem has been addressed in Section 5.8.4 (with an horizontal contact surface) by implementing an explicit central difference scheme. As in that case, to model the potential contact forces  $\mathbf{T}_C = T_n \mathbf{n}_\Gamma + T_t \mathbf{t}_\Gamma$  between the tire and the inclined plane, we adopt for simplicity a regularized version of the unilateral contact condition (8.1) as shown in Figure 8.10, where  $\gamma$  denotes the interpenetration between the tire and the rigid surface. Hence

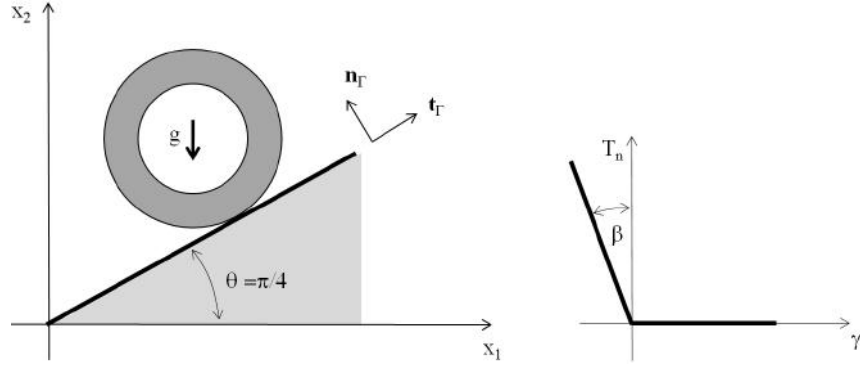
$$T_n = \frac{\gamma}{\sqrt{2}} |x_1 + u_1 - x_2 - u_2| H(x_1 + u_1 - x_2 - u_2) \quad \mathbf{x} \in S_C,$$

where  $H$  is the Heaviside step function ( $H(x) = 1$  if  $x > 0$ ,  $H(x) = 0$  if  $x < 0$ ). In this application, in order to put in evidence the effects of large rotations, we also introduce a simplified friction force exerted by the rigid plane upon the tire, of the form

$$T_t = f|T_n| \quad \text{on } S_C, \quad \text{where } f \text{ is the friction coefficient.}$$

**Exercise 8.3.** *Modify the code dynamics\_tire (section 5.8.4) in order to account for the presence of the inclined surface and of the friction force. Next run the analysis and verify that the numerical results are not realistic.*

Indeed, the presence of large rotations sets the requirement to formulate the problem in the framework of the general theory recalled in Section 8.3.

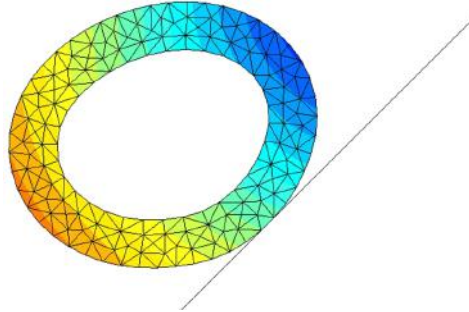


**Figure 8.10:** 2D contact problem of a tire rolling down a slope plane (left); regularization of the unilateral contact condition (right)

**Exercise 8.4** (Weak formulation). Assuming that  $dS_t/dS \approx 1$  (we have  $\mathbf{P} \cdot \mathbf{n} dS = \boldsymbol{\sigma} \cdot \mathbf{n}_t dS_t$ , which implies that  $\mathbf{P} \cdot \mathbf{n} \approx \boldsymbol{\sigma} \cdot \mathbf{n}_t$ ), prove that the weak formulation of the problem of a tire rolling down an inclined plane described above is given by:

$$\int_{\Omega} \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} \cdot \mathbf{w} dV + \int_{\Omega} \mathbf{P}[\mathbf{u}] : \nabla \mathbf{w} dV + \int_{\Omega} \rho g \mathbf{e}_2 \cdot \mathbf{w} dV - \int_{S_C} \mathbf{T}_C \cdot \mathbf{w} dS = 0 \quad \forall \mathbf{w} \in \mathcal{C}, \quad (8.49)$$

where  $\mathbf{T}_C$  is the contact force and  $S_C$  is the contact surface at time  $t$  in the reference configuration.



**Figure 8.11:** Snapshot of the tire rolling down the slope

As a consequence, the major difference with respect to the formulation implemented in Section 5.8.4 is the contribution of the internal forces which now depend in a nonlinear way on actual displacements.

The code `dynamics_tire_rolling.m` (input file `Chap8_tire.m`) implements the formulation (8.49) for T3 elements using an explicit central difference scheme. The

structure is identical to `dynamics_tire`, the only difference being that the contribution to nodal internal forces can no longer be evaluated with the command  $F_g = F - K * U$ . The term  $K * U$  must be replaced, at each time step, by the procedure of Box 8.4. In particular `T3_2A_solid_Fint` is a function that, for a T3 element, adapts the general procedure of Section (8.3.8) to the computation of the only quantity of interest, i.e.  $F_e$ .

**Exercise 8.5** (numerical simulation of a tire rolling down a slope). *Analysis in detail the procedure of Box 8.4. Run the analysis and comment.*

---

**Box 8.4:** Procedure for the numerical computation of internal nodal forces

---

```

for e=1:analysis.NE,
    type=elements(e).type;           % element type
    Etag=Le(type).tag;               % element tag
    ne=Le(type).ne;                  % number of nodes
    Dne=ndof*ne;                     % number of nodal values
    mat=elements(e).mat;              % number of material
    Xe=zeros(ne,DG);
    Ge=zeros(Dne,1);
    Ue=zeros(Dne,1);
    pos=1;
    for n=1:ne
        node=elements(e).nodes(n);  % global num of node
        Xe(n,:)=nodes(node).coor;    % creates element
        Ge(pos:pos+ndof-1)=nodes(node).dof; % global number of unknowns
        pos=pos+ndof;
    end
    Ue=U(Ge);                        % element nodal values
    Fe=T3_2A_solid_Fint(Xe,material(mat,:),Ue); % internal nodal forces
    Fg(Ge)=Fg(Ge)-Fe;                % assemblage of rhs
end

```

---



## Numerical elastoplasticity

---

In this chapter, we discuss the numerical computation of the mechanical state of solids made of elastoplastic materials. Only the simplest elastoplastic model (von Mises criterion with isotropic hardening) used for the plasticity of metals and widely employed in industrial applications is addressed herein, but the main concepts developed can easily be generalized. We adopt the following assumptions:

- Small-deformation assumption;
- Quasi-static process (slowly time-varying loading allowing to neglect inertia effects);

Section 9.1 presents a brief review of the essential concepts of the elastoplastic behavior within this framework. Section 9.2 introduces the governing equations and the overall structure of numerical simulation methods for elastoplastic structures, based on an incremental-iterative procedure. It involves two phases of different natures: a first purely local one (point level), related to the integration of the elastoplastic constitutive law and, a second, related to the global equilibrium of the structure, that requires a global treatment (structure level).

The local component of the solution strategy is first addressed. Section 9.3 is devoted to the algorithmic treatment of the incremental nature of the elastoplastic constitutive relation. This purely local component of the numerical treatment performs the integration of discrete time evolution equations. It rests on the *radial return algorithm*, which is described and then illustrated on a simple example in Section 9.4.

The global component of the solution strategy is then treated. Section 9.5 introduces the global nonlinear spatially-continuous problem, based on a weak formulation of the global equilibrium of the structure on a load step, and its solution by a Newton method. A more specific attention is devoted to the computation of the consistent tangent operator required by the Newton method. Practical aspects of the computation of a finite element approximation are addressed in Sections 9.6 and 9.7. Section 9.8 describes the code plasticity associated with this chapter followed (Sec. 9.9) by an illustration of the influence of the choice of the tangent operator on the efficiency of the algorithm.

## 9.1 Small-strain elastoplastic behavior: a review

This section provides a brief review whose scope is limited to the needs of this book. A few key concepts about the elastoplastic behavior are introduced to make this chapter self-sufficient. In particular, only the simplest elastoplastic model (von Mises criterion with isotropic hardening), used for the plasticity of metals, is described. For further details on nonlinear constitutive models, their physical background and algorithmic treatment, the reader may refer to books such as Simo and Hughes (1998); Besson, Cailletaud, Chaboche, and Forest (2009); Lemaître and Chaboche (1990).

### 9.1.1 Elasticity domain, yield limit, criterion

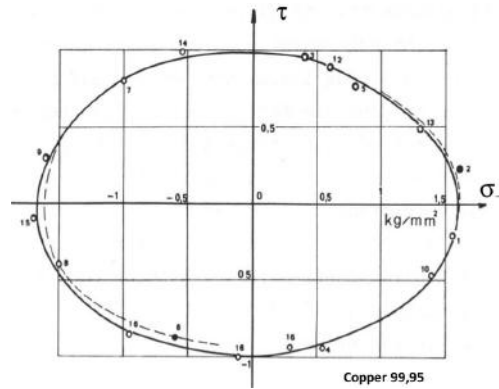
Experiments show that many solid materials have an elastic behavior (which is reversible, the material returning to its original state after a loading/unloading cycle) as long as stresses undergone are low enough. This observation is modeled by the notion of a *yield limit*. For example, in a simple tension-compression test in the  $e_1$  direction, a linear elastic behavior of the form  $\sigma_{11} = E\varepsilon_{11}$  (where  $E$  is the Young's modulus) is observed as long as  $|\sigma_{11}| - R \leq 0$ , wherein  $R$  is the yield limit in tension-compression. The generalisation of this concept is formulated through the definition of an *elasticity domain* in the stress space, described by a *yield criterion*  $f(\boldsymbol{\sigma})$  such that:

- The elasticity domain is the set of stresses such that  $f(\boldsymbol{\sigma}) < 0$ .

Moreover, for elastoplastic materials, the criterion  $f(\boldsymbol{\sigma})$  defines a *threshold*:

- All stress states in the material must be such that  $f(\boldsymbol{\sigma}) \leq 0$ .

The manifold defined, in the six-dimensional stress space, by the equation  $f(\boldsymbol{\sigma}) = 0$  is called the *yield surface*.



**Figure 9.1:** Tension-torsion test on copper (Bui, 1970): experimental determination of the initial yield surface (with the solid line showing the yield surface predicted by the von Mises criterion).

**Von Mises criterion.** Often used to describe the plasticity of metals, it reads:

$$\sigma^{\text{eq}} - R \leq 0, \quad (9.1)$$

where  $R$  is the yield limit and the *equivalent stress*  $\sigma^{\text{eq}}$  is defined by

$$\sigma^{\text{eq}} = \sqrt{\frac{3}{2}} \|s\| \quad \text{with } s := \sigma - \frac{1}{3} \text{Tr}(\sigma) \mathbf{I}, \quad (9.2)$$

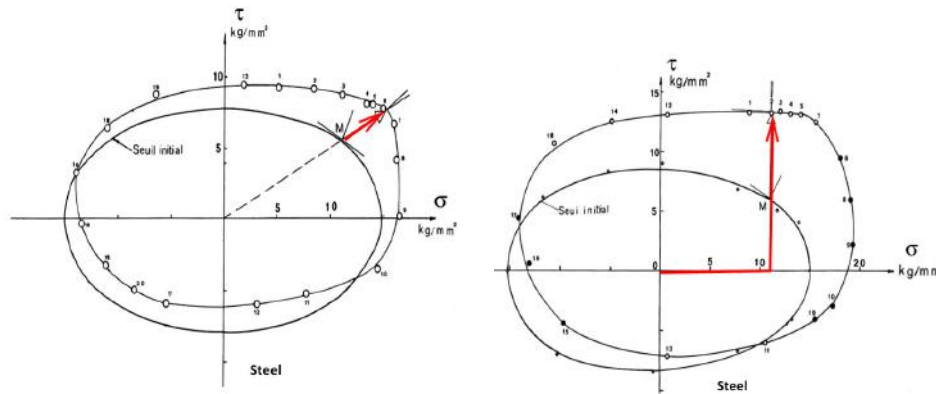
$s$  being the *deviatoric stress tensor*. The norm of a second-order (Euclidean) tensor  $\mathbf{a} \in \mathbb{R}^{3 \times 3}$  is defined by  $\|\mathbf{a}\| = (\mathbf{a} : \mathbf{a})^{1/2}$ , by direct extension of the usual definition  $\|v\| = (v \cdot v)^{1/2}$  of the Euclidean norm of a vector  $v \in \mathbb{R}^3$ . The factor  $\sqrt{3/2}$  in definition (9.2) is chosen so as to satisfy  $\sigma^{\text{eq}} = \sigma$  for a uniaxial stress state  $\sigma = \sigma(e_1 \otimes e_1)$ , allowing a simple interpretation of the equivalent stress. In particular, this interpretation implies that the yield limit  $R$  of von Mises criterion (9.1) is the value of the yield limit (elasticity limit) observed in an uniaxial tension-compression test.

### 9.1.2 Evolution of the yield surface, hardening

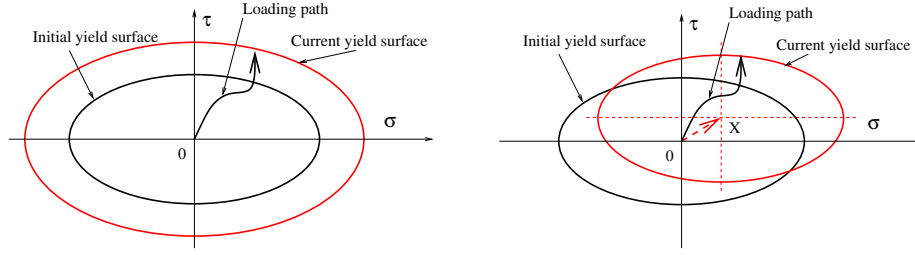
It is observed for many materials that the loading can be increased beyond the initial yield limit. The material then deforms plastically and the yield surface evolves (Fig. 9.2) as plastic strains develop in the material: this phenomenon is called *strain hardening*.

We distinguish in particular the notion of *isotropic* hardening (increase in size of the yield surface, its center and shape remaining unchanged, as shown in Fig. 9.3a), corresponding to a criterion of the form

$$f(\sigma) - R \leq 0 \quad (9.3)$$



**Figure 9.2:** Tension-torsion test on mild steel (Bui, 1970). Evolution of the yield surface for two load paths: radial load path (left), tension below the elasticity limit followed by a torsion with constant tensile stress (right).



**Figure 9.3:** Isotropic (left) or kinematic (right) hardening.

where the yield limit  $R$  depends on the plastic strain. In the following, we will consider only this type of hardening, although other forms of hardening also exist, e.g. *kinematic* hardening (the yield surface evolves through translation in stress space, its shape and size remaining fixed, as shown in Figure 9.3b) and combinations of isotropic and kinematic hardening.

### 9.1.3 Plastic strains

At points where the material undergoes an elastoplastic strain  $\epsilon$ , we define the elastic part  $\epsilon^E$  and the plastic part  $\epsilon^P$  of the strain<sup>1</sup> by

$$\epsilon^E = \mathcal{S} : \sigma, \quad \epsilon^P = \epsilon - \epsilon^E, \quad \text{then} \quad \sigma = \mathcal{A} : \epsilon^E = \mathcal{A} : (\epsilon - \epsilon^P), \quad (9.4)$$

where  $\mathcal{S}$  is the elastic compliance tensor defined in (1.17). According to this definition, a purely elastic evolution of the material implies zero plastic part  $\epsilon^P$  (which justifies the decomposition (9.4) and the terminology). So, the latter only evolves during loading phases beyond the current yield limit (in particular, the plastic strain does not evolve during an elastic unloading).

A given stress level corresponds to infinitely many possible values of the plastic strain. This implies that the current state of the material depends on the loading history. A scalar measure of the history of plastic strain over time is the *cumulative plastic strain*  $p(t)$  defined by

$$\dot{p}(t) = \sqrt{\frac{2}{3}} \int_0^t \|\dot{\epsilon}^P(u)\| du. \quad (9.5)$$

The factor  $\sqrt{2/3}$  in this definition is adjusted so that  $\dot{p}$  satisfies  $\sigma : \dot{\epsilon}^P = \sigma^{\text{eq}} \dot{p}$  in uniaxial tension.

### 9.1.4 Normality rule

Multiaxial tests on metals highlight the fact that, when the loading is continued from a stress state located on the yield surface, the plastic strain rate is normal to the yield surface (Figure 9.4). This leads to postulate that the evolution of the plastic strain is governed by the *normality rule*:

<sup>1</sup>Often termed elastic and plastic strain even though  $\epsilon^E$  and  $\epsilon^P$  do not usually derive from a displacement



- The plastic strain rate is zero whenever stresses are in the elasticity domain ( $f(\boldsymbol{\sigma}) < 0$ ). When the stress is on the yield surface ( $f(\boldsymbol{\sigma}) = 0$ ), the plastic strain rate is normal to the yield surface.

The mathematical formulation of the normality rule, covering both cases, writes

$$\dot{\boldsymbol{\epsilon}}^P = \dot{\gamma} \frac{\partial f}{\partial \boldsymbol{\sigma}}, \quad \dot{\gamma} \geq 0, \quad f(\boldsymbol{\sigma}) \leq 0, \quad \dot{\gamma} f(\boldsymbol{\sigma}) = 0. \quad (9.6)$$

The *plastic multiplier*  $\dot{\gamma}$  is an *a priori* undetermined scalar, and thus is one of the unknowns introduced by the model. The *complementarity condition*  $\dot{\gamma} f(\boldsymbol{\sigma}) = 0$  is used to express the fact that the plastic strain may change only in situations where the current stress state is on the yield surface.

For the von Mises criterion (9.1), we find that

$$\frac{\partial f}{\partial \boldsymbol{\sigma}} = \frac{3}{2\sigma^{\text{eq}}} \mathbf{s}. \quad (9.7)$$

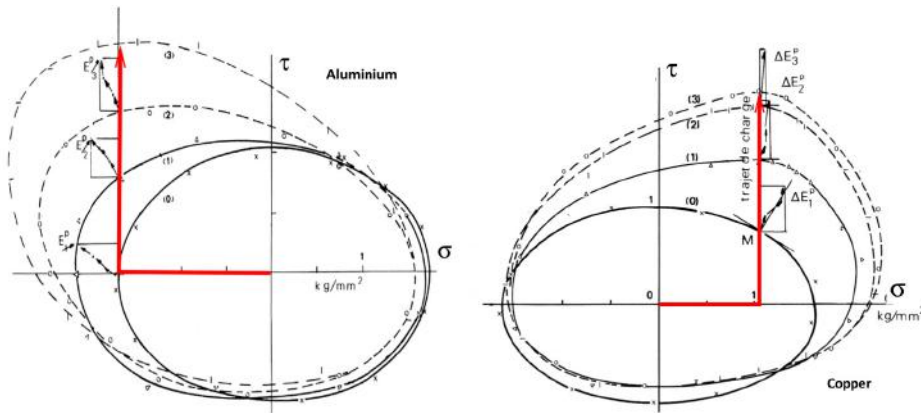
Since  $\|\mathbf{s}\| = \sqrt{2/3}\sigma^{\text{eq}}$  by definition (9.2) of the equivalent stress, we then note that the normality rule (9.6) and the formula (9.7) imply that the cumulative plastic strain rate can be explicitly evaluated:

$$\dot{p} = \sqrt{\frac{2}{3}} \|\dot{\boldsymbol{\epsilon}}^P\| = \sqrt{\frac{2}{3}} \dot{\gamma} \frac{3}{2\sigma^{\text{eq}}} \sqrt{\frac{2}{3}} \sigma^{\text{eq}} = \dot{\gamma}$$

Thus, for the *von Mises criterion*, the plastic multiplier  $\dot{\gamma}$  is equal to the equivalent plastic strain rate  $\dot{p}$ . In the sequel we only consider the von Mises criterion. As a consequence, we will write the constitutive relations in terms of  $\dot{p}$  rather than  $\dot{\gamma}$ .

### 9.1.5 Summary of elastoplastic constitutive model

Forthcoming developments in this chapter are limited to a family of elastoplastic constitutive models often used for metallic materials, based on the following assumptions:



**Figure 9.4:** Experimental evidence of the normality of the plastic strain rate to the yield surface, at different stages of a loading path (Bui, 1970).

- (i) kinematic satisfying small-deformation assumption (SDA);
- (ii) linear isotropic elasticity;
- (iii) von Mises criterion;
- (iv) plastic flow follows the normality rule;
- (v) isotropic hardening.

Under such assumptions, the constitutive relations read:

$$\begin{aligned}\boldsymbol{\sigma} &= \mathcal{A}:(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^P) \\ &= [3\kappa\mathcal{J} + 2\mu\mathcal{K}]:(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^P) \\ &= \kappa\text{tr}(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^P)\mathbf{I} + 2\mu(\boldsymbol{e} - \boldsymbol{e}^P) \quad (\text{elasticity}),\end{aligned}\tag{9.8a}$$

$$f(\boldsymbol{\sigma}, p) = \sigma^{\text{eq}} - R(p) \leq 0 \quad (\text{yield criterion}),\tag{9.8b}$$

$$\dot{\boldsymbol{\varepsilon}}^P = \dot{p} \frac{\partial f}{\partial \boldsymbol{\sigma}}(\boldsymbol{\sigma}, p) = \frac{3\dot{p}}{2\sigma^{\text{eq}}} \boldsymbol{s}, \quad \dot{p} \geq 0, \quad (\text{normality rule}),\tag{9.8c}$$

$$\dot{p}f(\boldsymbol{\sigma}, p) = \dot{p}(\sigma^{\text{eq}} - R(p)) = 0 \quad (\text{complementarity condition}),\tag{9.8d}$$

where  $\boldsymbol{\varepsilon}$  is the strain,  $\boldsymbol{\varepsilon}^P$  the plastic part of the strain,  $\boldsymbol{e} = \mathcal{K}:\boldsymbol{\varepsilon}$  the strain deviator,  $\boldsymbol{e}^P = \mathcal{K}:\boldsymbol{\varepsilon}^P$  the plastic strain deviator,  $\boldsymbol{\sigma}$  the stress,  $\boldsymbol{s} = \mathcal{K}:\boldsymbol{\sigma}$  the stress deviator,  $\sigma^{\text{eq}} = \sqrt{3/2}\|\boldsymbol{s}\|$  the equivalent stress, and  $p$  the cumulative plastic strain. The fourth-order tensors  $\mathcal{J}$  and  $\mathcal{K}$ , respectively associated with the projections onto the subspaces of isotropic and deviatoric second-order symmetric tensors, have been introduced in Chapter 1 and are defined by (1.13) and (1.14). Besides, the formulation (9.8a) of the elastic part of the behavior involves the bulk modulus  $\kappa$ :

$$3\kappa = 2\mu \frac{1+\nu}{1-2\nu} = 3\lambda + 2\mu = \frac{E}{1-2\nu}.$$

We assume the function  $R(p)$  defining the yield limit as a function of the cumulative plastic strain to have the following properties:

$$\begin{aligned}(\text{i}) \quad & R(0) = \sigma_0, \quad (\text{ii}) \quad R'(p) \geq 0, \\ (\text{iii}) \quad & R(\alpha p_1 + (1-\alpha)p_2) \geq \alpha R(p_1) + (1-\alpha)R(p_2) \quad (0 \leq \alpha \leq 1),\end{aligned}\tag{9.9}$$

where (i)  $\sigma_0$  is the initial tensile yield limit, (ii) states that the yield limit increases with the cumulative plastic strain, and (iii) the function  $R(p)$  is *concave*. Since the equivalent stress  $\sigma^{\text{eq}}$  is a convex function of  $\boldsymbol{\sigma}$ , (iii) ensures that the set of plastically admissible pairs  $(\boldsymbol{\sigma}, p)$  is convex. In the standard particular case of linear isotropic hardening, the function  $R(p)$  is of the form:

$$R(p) = \sigma_0 + hp\tag{9.10}$$

(where  $\sigma_0$  is the initial yield limit and  $h$  is the hardening modulus), and clearly satisfies assumptions (9.9).

It is also useful to note that the normality rule (9.8c) may be recast in the form

$$\dot{\varepsilon}^P = \sqrt{\frac{3}{2}} \dot{p} \mathbf{N} \quad \text{with} \quad \mathbf{N} = \sqrt{\frac{2}{3}} \frac{\partial f}{\partial \boldsymbol{\sigma}}(\boldsymbol{\sigma}, p) = \sqrt{\frac{3}{2}} \frac{1}{\sigma^{\text{eq}}} \mathbf{s}, \quad (9.11)$$

where the tensor  $\mathbf{N}$  as defined above is the unit "normal" ( $\|\mathbf{N}\|^2 = (\mathbf{N} : \mathbf{N}) = 1$ ) to the yield surface defined by  $f(\boldsymbol{\sigma}, p) = 0$ . Finally, an immediate consequence of the normality rule, in its form (9.8c) or (9.11), is the plastic incompressibility:

$$\text{Tr}(\varepsilon^P) = 0 \quad \text{or} \quad \mathbf{e}^P = \varepsilon^P$$

which in fact holds for any pressure-independent yield criterion  $f(\boldsymbol{\sigma})$ .

## 9.2 Numerical elasto-plasticity: problem formulation

### 9.2.1 Assumptions and governing equations

Consider a structure occupying the domain  $\Omega$ . The structure is subjected to a volume force density  $\mathbf{f}$  in  $\Omega$  and a surface force density  $\mathbf{T}^D$  on a part  $S_T$  of its boundary  $\partial\Omega$  while being constrained by prescribed displacements  $\mathbf{u}^D$  over the remaining part  $S_u = \partial\Omega \setminus S_T$  of  $\partial\Omega$ . These loadings are assumed to be time-dependent. Surfaces  $S_u$  and  $S_T$  are however assumed to be time-independent for the sake of simplicity. The evolution of the structure is slow enough to make it reasonable to neglect inertia effects (quasi-static analysis). The constitutive material is assumed to have an elastoplastic behavior of the form (9.8a–d). The quasi-static evolution of the structure over the time interval  $t \in [0, T]$  is then governed by equations:

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \mathbf{u} + \nabla^T \mathbf{u}) \quad \text{in } \Omega \times [0, T] \quad (\text{compatibility}), \quad (9.12a)$$

$$\text{div} \boldsymbol{\sigma} + \rho \mathbf{f} = \mathbf{0} \quad \text{in } \Omega \times [0, T] \quad (\text{equilibrium}), \quad (9.12b)$$

$$\boldsymbol{\sigma} = \kappa \text{tr}(\boldsymbol{\varepsilon}) \mathbf{I} + 2\mu(\mathbf{e} - \varepsilon^P) \quad (\text{constitutive law, elastic part}), \quad (9.12c)$$

$$\dot{\varepsilon}^P = \dot{p} \frac{3}{2\sigma^{\text{eq}}} \mathbf{s}, \quad \dot{p} \geq 0, \quad \sigma^{\text{eq}} - R(p) \leq 0, \quad \dot{p}[\sigma^{\text{eq}} - R(p)] = 0 \quad (\text{constitutive law, plastic part}), \quad (9.12d)$$

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{u}^D(\mathbf{x}, t) \quad \text{on } S_u \times [0, T] \quad (\text{prescribed displacement}), \quad (9.12e)$$

$$\mathbf{T}(\mathbf{x}, t) = \mathbf{T}^D(\mathbf{x}, t) \quad \text{on } S_T \times [0, T] \quad (\text{prescribed traction}), \quad (9.12f)$$

$$\varepsilon^P(\mathbf{x}, 0) = \mathbf{0} \quad \text{in } \Omega \quad (\text{initial condition}). \quad (9.12g)$$

It is noted in particular that (9.12g) implies that the loadings  $\mathbf{f}(\mathbf{x}, 0)$ ,  $\mathbf{u}^D(\mathbf{x}, 0)$ ,  $\mathbf{T}^D(\mathbf{x}, 0)$  are such that the initial state of the structure is elastic. Moreover, the adopted von Mises criterion implies plastic incompressibility, i.e. that the plastic strain is purely deviatoric ( $\mathbf{e}^P = \varepsilon^P$ ), see (9.11).

### 9.2.2 Numerical simulation: principle

Chapter 8 has already introduced the concept of iterative algorithm for the computation of equilibrium states of structures with nonlinear behavior. In the case of a non-linear elastic material, a weak formulation for the unknown  $\mathbf{u}$ , nonlinear in  $\mathbf{u}$ , was obtained, the solution algorithm consisting in the construction by means of the Newton method of a sequence  $\mathbf{u}^{[k]}$  converging to  $\mathbf{u}$  (other methods using successive approximations, not discussed in this book, may alternatively be considered).

The computation of an elastoplastic structural response is based on a similar approach but with substantial differences:

- (a) Equations (9.12a-g) define a quasi-static evolution problem, whose solution  $\mathbf{u}(\mathbf{x}, t)$  must thus be sought as a function of space *and* time;
- (b) The constitutive relation (stress-strain relation) is no longer a function of the current state (in contrast with (8.17) for nonlinear elasticity). The stress-strain relation is independent of the loading rate but does depend on the loading path. The stress field  $\boldsymbol{\sigma}(\mathbf{x}, t)$  at point  $\mathbf{x}$  and time  $t$  depends on the whole history ( $\tau \in [0, t]$ ) of other variables ( $\boldsymbol{\varepsilon}(\mathbf{x}, \tau)$ ,  $\boldsymbol{\varepsilon}^P(\mathbf{x}, \tau)$ , ...) at point  $\mathbf{x}$ .

**Time stepping procedure.** To take into account (in an approximate way) the time dependence of the solution, a sequence of  $M + 1$  regularly spaced<sup>2</sup> discrete time instants,  $t_0 = 0, t_1 = \Delta t, \dots, t_M = M\Delta t = t^F$  is introduced (the time step is then  $\Delta t = t^F/M$ ). The goal of the algorithm is then to compute all mechanical fields (displacement  $\mathbf{u}$ , strain  $\boldsymbol{\varepsilon}$ , plastic strain  $\boldsymbol{\varepsilon}^P$ , stress  $\boldsymbol{\sigma}$ , ...) at all discrete time instants  $t = t_n$  ( $0 \leq n \leq M$ ). Denoting by  $f_n(\mathbf{x}) = f(\mathbf{x}, t_n)$  the value of  $f(\mathbf{x}, t)$  at time  $t_n$ , this objective can be summarized as:

$$\text{compute } \mathcal{S}_n = \{\mathbf{u}_n, \boldsymbol{\varepsilon}_n, \boldsymbol{\varepsilon}_n^P, \boldsymbol{\sigma}_n, \dots\} \text{ at every time } t = t_n.$$

An *incremental* approach is then built: the mechanical state of the structure is evaluated *step-by-step*. More precisely, the incremental approach consists in computing the mechanical state  $\mathcal{S}_{n+1} = \{\mathbf{u}_{n+1}, \boldsymbol{\varepsilon}_{n+1}, \boldsymbol{\varepsilon}_{n+1}^P, \boldsymbol{\sigma}_{n+1}\}$  at time  $t_{n+1}$  knowing (i) the mechanical state  $\mathcal{S}_n = \{\mathbf{u}_n, \boldsymbol{\varepsilon}_n, \boldsymbol{\varepsilon}_n^P, \boldsymbol{\sigma}_n\}$  at time  $t_n$  and (ii) the loading  $(\mathbf{f}_{n+1}, \mathbf{u}_{n+1}^D, \mathbf{T}_{n+1}^D)$  at time  $t = t_{n+1}$ . Such a step-by-step procedure can be summarized as:

$$\text{knowing } \mathcal{S}_n \text{ and } (\mathbf{f}_{n+1}, \mathbf{u}_{n+1}^D, \mathbf{T}_{n+1}^D), \quad \text{compute } \mathcal{S}_{n+1}, \quad (9.13)$$

**Displacement-based space discretisation approach.** For each time step, a finite element approximation of  $\Omega$  and the kinematic fields (unknown displacement  $\mathbf{u}$ , virtual fields  $\mathbf{w}$ ) is used. Such approximations are based on the concepts introduced in Chapters 2 and 3.

<sup>2</sup>the assumption of constant time step is not essential but simplifies the analysis and algorithm.

As done previously for linear elasticity (Chapter 3) or nonlinear elasticity (Chapter 8), the approximate problem is constructed on the basis of a weak formulation of the equilibrium conditions (9.12b) and (9.12f) which, at time  $t_{n+1}$ , can be written

$$\int_{\Omega} \boldsymbol{\sigma}_{n+1} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = \int_{\Omega} \rho \mathbf{f}_{n+1} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}_{n+1}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (9.14)$$

The formulation considered here for the finite element approximation, known as a "displacement-based approach", treats the displacement field as the primary unknown, other variables being considered as secondary quantities that can be evaluated once the displacement is known. This leads to treat the stress  $\boldsymbol{\sigma}_{n+1}$ , at any given point  $\mathbf{x} \in \Omega$  as a function of the displacement  $\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta \mathbf{u}_n$  and of all the other mechanical variables  $\mathcal{S}_n$  at time  $t = t_n$ . Accordingly, problem (9.14) can be recast into a nonlinear problem whose unknown is the increment of displacement  $\Delta \mathbf{u}_n$ ; its computation can then be based on an iterative algorithm similar to the one used for nonlinear elasticity (Section 8.3.4).

**Local integration of the constitutive relations.** The foregoing considerations clearly outline the importance and necessity to built an algorithm dedicated to the local integration of elastoplastic behavior. This goal can be summarized by:

$$\text{knowing } \mathcal{S}_n \text{ and } \Delta \mathbf{u}_n, \quad \text{find } \boldsymbol{\sigma}_{n+1}, \quad (9.15)$$

which can be formally represented by an algorithm denoted  $\Sigma$ :

$$(\Delta \mathbf{u}_n, \mathcal{S}_n) \longrightarrow \boldsymbol{\sigma}_{n+1} = \Sigma(\Delta \boldsymbol{\varepsilon}_n; \mathcal{S}_n), \quad \text{with } \Delta \boldsymbol{\varepsilon}_n = \boldsymbol{\varepsilon}[\Delta \mathbf{u}_n] \quad (9.16)$$

**Global equilibrium of the structure.** The weak formulation (9.14) at time  $t_{n+1}$  leads to the nonlinear problem in  $\Delta \mathbf{u}_n$ :

$$\text{find } \Delta \mathbf{u}_n \in \mathcal{C}(\Delta \mathbf{u}_n^D), \quad \mathcal{R}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n) = 0 \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}), \quad (9.17)$$

where the residual  $\mathcal{R}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n)$ , defined by

$$\begin{aligned} \mathcal{R}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n) = & \int_{\Omega} \Sigma(\boldsymbol{\varepsilon}[\Delta \mathbf{u}_n]; \mathcal{S}_n) : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV \\ & - \int_{\Omega} \rho \mathbf{f}_{n+1} \cdot \mathbf{w} \, dV - \int_{S_T} \mathbf{T}_{n+1}^D \cdot \mathbf{w} \, dS, \end{aligned} \quad (9.18)$$

depends on the algorithm  $\Sigma$  which still needs to be specified and the solution  $\Delta \mathbf{u}_n$  has to be kinematically admissible with  $\Delta \mathbf{u}_n^D = \mathbf{u}_{n+1}^D - \mathbf{u}_n^D$ .

Hence, the numerical solution of a time step involves two tasks, respectively defined at the local and structural (global) level:

- The integration of elastoplastic constitutive relations over a time step at any given point, symbolized by (9.16), which has a strictly local character.
- The solution of the equilibrium equation in weak form (9.17), which is a global problem involving the whole structure.

The local integration of the elastoplastic behavior is addressed in Sections 9.3 and 9.4 whereas the global equilibrium problem is discussed in Sections 9.5 and 9.6. We stress for now that the solution strategy will be both *incremental* and *iterative*. The incremental (step-by-step) character of the strategy is related to the time integration, while its iterative character results from the need to solve a nonlinear problem at each time step.

### 9.3 Local integration of elastoplastic constitutive relations

The problem of local integration of elastoplastic constitutive relations on a time step can be formulated as follows:

- Consider a material element whose mechanical state  $\mathcal{S}_n = \{\boldsymbol{\varepsilon}_n, \boldsymbol{\varepsilon}_n^P, \boldsymbol{\sigma}_n\}$  at time  $t_n$  is spatially uniform and known. Prescribe a given strain increment  $\Delta\boldsymbol{\varepsilon}_n = \boldsymbol{\varepsilon}[\Delta\mathbf{u}_n]$ , so that at time  $t_{n+1}$  the strain is  $\boldsymbol{\varepsilon}_{n+1} = \boldsymbol{\varepsilon}_n + \Delta\boldsymbol{\varepsilon}_n$ . Find the stress  $\boldsymbol{\sigma}_{n+1}$  and plastic strain  $\boldsymbol{\varepsilon}_{n+1}^P$  that satisfy the constitutive relations (9.8a–d).

Accordingly, the local integration problem favors strain as the main kinematic variable. This simply reflects the fact that constitutive equations are stress-strain relations. Displacements, rather than strains, appear in the global problem due to the compatibility condition (9.12a), i.e.  $\Delta\boldsymbol{\varepsilon}_n = \boldsymbol{\varepsilon}[\Delta\mathbf{u}_n]$ .

#### 9.3.1 Time integration: implicit approach, "discrete" constitutive relations

Among the constitutive relations (9.8a–d), the normality rule

$$\dot{\boldsymbol{\varepsilon}}^P = \dot{p} \frac{3}{2\sigma^{\text{eq}}} \mathbf{s}$$

is a first-order differential equation in time. The numerical integration of this type of relation is often based on approximating the derivative by *finite difference*:

$$\dot{\boldsymbol{\varepsilon}}^P \approx [\boldsymbol{\varepsilon}_{n+1}^P - \boldsymbol{\varepsilon}_n^P] / \Delta t.$$

This finite difference can be considered as an approximation of  $\dot{\boldsymbol{\varepsilon}}_n^P$  ("forward" or "explicit" scheme), or  $\dot{\boldsymbol{\varepsilon}}_{n+1}^P$  ("backward" or "implicit" scheme), or  $\dot{\boldsymbol{\varepsilon}}^P$  at some intermediate time  $t \in [t_n, t_{n+1}]$ . As previously discussed in Chapter 4, the properties (stability, accuracy) of numerical time integration schemes are likely to depend on this choice. The "implicit" version often leads to schemes that are unconditionally stable, whereas the stability of the "explicit" version is usually achieved only for sufficiently small time steps (and therefore at the potential cost of a larger number of time steps).

For these reasons, we choose the implicit version: constitutive relations (9.8a–d) are written at time  $t_{n+1}$  and finite differences are treated as approximations of time derivatives at this final time. The discrete form of the constitutive relations, which is the basis for the local integration procedure, can be written:

$$\boldsymbol{\sigma}_{n+1} = \boldsymbol{\sigma}_n + \kappa \text{tr}(\Delta\boldsymbol{\varepsilon}_n) \mathbf{I} + 2\mu(\Delta\mathbf{e}_n - \Delta\boldsymbol{\varepsilon}_n^P) \quad (\text{elasticity}), \quad (9.19a)$$

$$\sigma_{n+1}^{\text{eq}} - R(p_n + \Delta p_n) \leq 0 \quad (\text{von Mises}), \quad (9.19b)$$

$$\begin{aligned}\Delta \varepsilon_n^P &= \Delta p_n \frac{3}{2\sigma_{n+1}^{\text{eq}}} \mathbf{s}_{n+1} = \Delta p_n \sqrt{\frac{3}{2}} \mathbf{N}_{n+1}, \\ \Delta p_n &\geq 0, \quad \Delta p_n (\sigma_{n+1}^{\text{eq}} - R(p_n + \Delta p_n)) = 0 \quad (\text{normality rule}),\end{aligned}\quad (9.19c)$$

where  $\Delta \mathbf{e}_n = \mathcal{K} : \Delta \varepsilon_n$  is the deviatoric part of the strain increment  $\Delta \varepsilon_n = \varepsilon_{n+1} - \varepsilon_n$ ,  $\Delta p_n = p_{n+1} - p_n$  the (unknown) increment of cumulative plastic strain, and  $\mathbf{N}_{n+1}$  the (unknown) unit normal to the *final* yield surface. Recall, again, that plastic incompressibility implicit to the von Mises criterion implies that  $\Delta \varepsilon_n^P = \Delta \mathbf{e}_n^P$ .

### 9.3.2 Elastic prediction and correction

Solving the discrete form of the constitutive relations (9.19a-c) depends on whether  $\sigma_{n+1}^{\text{eq}} - R(p_n + \Delta p_n) < 0$  (in which case the evolution is purely elastic and therefore  $\Delta p_n = 0$ ) or  $\sigma_{n+1}^{\text{eq}} - R(p_n + \Delta p_n) = 0$  (in which case plastic strains may evolve and therefore  $\Delta p_n \neq 0$  *a priori*). It is then natural to propose a strategy based on the definition of an *elastic prediction*, followed, if necessary, by an *elastoplastic correction* approach. The elastic prediction  $\sigma_{n+1}^{\text{elas}}$  of the stress  $\sigma_{n+1}$  at time  $t_{n+1}$  is then defined by:

$$\sigma_{n+1}^{\text{elas}} = \sigma_n + \mathcal{A} : \Delta \varepsilon_n = \sigma_n + \kappa \text{tr}(\Delta \varepsilon_n) \mathbf{I} + 2\mu \Delta \mathbf{e}_n \quad (9.20)$$

and its deviatoric counterpart,

$$\mathbf{s}_{n+1}^{\text{elas}} = \mathcal{K} : \sigma_{n+1}^{\text{elas}} = \mathbf{s}_n + 2\mu \Delta \mathbf{e}_n. \quad (9.21)$$

This prediction therefore consists in (temporarily) assuming a purely elastic evolution of stresses over the time interval  $[t_n, t_{n+1}]$  (and in particular such that  $\Delta p_n = 0$ ).

Because of the convexity and differentiability of the yield function  $f(\sigma, p) = \sigma^{\text{eq}} - R(p)$  (see the assumptions in Section 9.1.5), we have,

$$\begin{aligned}f(\sigma_{n+1}^{\text{elas}}, p_n) - f(\sigma_{n+1}, p_n + \Delta p_n) \\ \geq (\sigma_{n+1}^{\text{elas}} - \sigma_{n+1}) : \frac{\partial f}{\partial \sigma}(\sigma_{n+1}) - \Delta p_n \frac{\partial f}{\partial p}(p_n + \Delta p_n),\end{aligned}$$

which results from the direct application of the following inequality, satisfied by any differentiable convex function  $f(z)$ :

$$\text{for all } (z_1, z_2), \quad f(z_2) - f(z_1) \geq \nabla f(z_1) \cdot (z_2 - z_1).$$

In addition, the elastic part (9.19a) of the constitutive relations and the "discrete" form of the normality rule (9.19c) imply that

$$\begin{aligned}\sigma_{n+1}^{\text{elas}} - \sigma_{n+1} &= \mathcal{A} : \Delta \varepsilon_n - \mathcal{A} : (\Delta \varepsilon_n - \Delta \varepsilon_n^P) = \mathcal{A} : \Delta \varepsilon_n^P \\ &= \sqrt{\frac{3}{2}} \Delta p_n (\mathcal{A} : \mathbf{N}_{n+1}),\end{aligned}$$

so that

$$\begin{aligned} (\sigma_{n+1}^{\text{elas}} - \sigma_{n+1}) : \frac{\partial f}{\partial \sigma}(\sigma_{n+1}) - \Delta p_n \frac{\partial f}{\partial p}(p_n + \Delta p_n) \\ = \frac{3}{2} \Delta p_n (N_{n+1} : \mathcal{A} : N_{n+1}) + \Delta p_n R'((p_n + \Delta p_n)) \geq 0 \end{aligned}$$

because of both the assumption (9.9(ii)) on the function  $R$  and the positive definiteness of the quadratic form associated with  $\mathcal{A}$ . One then has proved the inequality

$$f(\sigma_{n+1}^{\text{elas}}, p_n) \geq f(\sigma_{n+1}, p_n + \Delta p_n) \quad (9.22)$$

Two possibilities then arise:

- If  $f(\sigma_{n+1}^{\text{elas}}, p_n) \leq 0$ , inequality (9.22) then ensures  $f(\sigma_{n+1}, p_n + \Delta p_n) \leq 0$ . The elastic prediction is plastically admissible, and the final state  $\sigma_{n+1}$  results from a purely elastic evolution over the time step and is given by

$$\sigma_{n+1} = \sigma_{n+1}^{\text{elas}} \quad \epsilon_{n+1}^{\text{P}} = \epsilon_n^{\text{P}} \quad p_{n+1} = p_n;$$

- If  $f(\sigma_{n+1}^{\text{elas}}, p_n) > 0$ , the elastic prediction  $\sigma_{n+1}^{\text{elas}}$  is not plastically admissible and in particular  $\Delta \epsilon_n \neq \Delta \epsilon_n^{\text{E}}$ . A nonzero variation of the plastic strain during the time step, such that  $\Delta p_n > 0$  holds, must be assumed. The "discrete" form of the complementarity condition  $\Delta p_n f(\sigma_{n+1}, p_n + \Delta p_n) = 0$  then implies  $f(\sigma_{n+1}, p_n + \Delta p_n) = 0$ , that is, the final stress is on the *final* yield surface. In this case, the "discrete" form (9.19a,c) of the constitutive relation reduces to:

$$\sigma_{n+1} = \sigma_{n+1}^{\text{elas}} - 2\mu \Delta \epsilon_n^{\text{P}}, \quad \Delta \epsilon_n^{\text{P}} = \Delta p_n \sqrt{\frac{3}{2}} N_{n+1}, \quad \Delta p_n > 0.$$

It appears that the *correction*  $\sigma_{n+1} - \sigma_{n+1}^{\text{elas}}$  is normal to the final yield surface.

In other words,  $\sigma_{n+1}$  is the orthogonal projection of  $\sigma_{n+1}^{\text{elas}}$  onto the final yield surface (figure 9.5):

$$\sigma_{n+1} = \mathcal{P}(\sigma_{n+1}^{\text{elas}}) \quad (9.23)$$

Due to the fact that the yield surface  $f(\sigma_{n+1}, p_{n+1}) = 0$  depends on the final state through  $p_{n+1} = p_n + \Delta p_n$ , the correction step is of implicit nature. We now go into details of the procedure, and in particular specify the *radial return algorithm* that performs the integration of the constitutive relation on a time step  $\Delta t$ .

### 9.3.3 Radial return algorithm

The deviatoric part of the elastic relation (9.19a) reads

$$s_{n+1} = s_n + 2\mu(\Delta e_n - \Delta \epsilon_n^{\text{P}}) \quad \text{or} \quad s_{n+1} = s_{n+1}^{\text{elas}} - 2\mu \Delta \epsilon_n^{\text{P}}. \quad (9.24)$$

Using the discrete form (9.19c) of the normality rule gives (after elimination of  $\Delta \epsilon_n^{\text{P}}$ )

$$s_{n+1} = s_{n+1}^{\text{elas}} - 2\mu \sqrt{\frac{3}{2}} \Delta p_n N_{n+1}. \quad (9.25)$$



By definition (9.11) of the unit normal to the yield surface for the von Mises criterion, one has

$$\mathbf{s}_{n+1} = \sqrt{\frac{2}{3}} \sigma_{n+1}^{\text{eq}} \mathbf{N}_{n+1},$$

and equation (9.25) finally reads

$$\left( \sqrt{\frac{2}{3}} \sigma_{n+1}^{\text{eq}} + 2\mu \sqrt{\frac{3}{2}} \Delta p_n \right) \mathbf{N}_{n+1} = \mathbf{s}_{n+1}^{\text{elas}}$$

which highlights an important point:

- The normal  $\mathbf{N}_{n+1}$  to the final yield surface is collinear to the deviatoric part of the elastic prediction of the stress  $\mathbf{s}_{n+1}^{\text{elas}}$ :

$$\mathbf{N}_{n+1} = \frac{1}{\|\mathbf{s}_{n+1}^{\text{elas}}\|} \mathbf{s}_{n+1}^{\text{elas}} = \sqrt{\frac{3}{2}} \frac{1}{\sigma_{n+1}^{\text{elas,eq}}} \mathbf{s}_{n+1}^{\text{elas}} = \mathbf{N}_{n+1}^{\text{elas}}. \quad (9.26)$$

This remark explains the terminology *radial return algorithm* given to this algorithm: the correction of the elastic predictor is directed along the radial line connecting the center of the yield surface to the deviator of the elastic prediction  $\mathbf{s}_{n+1}^{\text{elas}}$  (figure 9.5).

One now seeks, using (9.25) and (9.26), the final deviatoric stress in the form

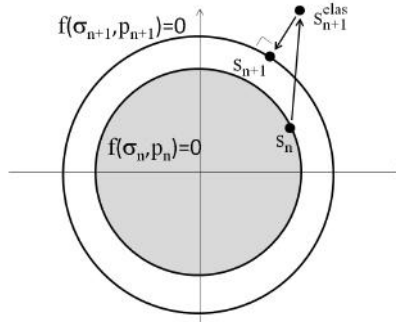
$$\mathbf{s}_{n+1} = \mathbf{s}_{n+1}^{\text{elas}} - 2\mu \sqrt{\frac{3}{2}} \Delta p_n \mathbf{N}_{n+1}^{\text{elas}} = \sqrt{\frac{2}{3}} (\sigma_{n+1}^{\text{elas,eq}} - 3\mu \Delta p_n) \mathbf{N}_{n+1}^{\text{elas}}, \quad (9.27)$$

which yields

$$\sigma_{n+1}^{\text{eq}} = \sigma_{n+1}^{\text{elas,eq}} - 3\mu \Delta p_n.$$

Finally, the evolution of the stress during the time step being not purely elastic by assumption, the final stress must be on the yield surface:  $\sigma_{n+1}^{\text{eq}} - R(p_n + \Delta p_n) = 0$ . This condition reads

$$\sigma_{n+1}^{\text{elas,eq}} - 3\mu \Delta p_n - R(p_n + \Delta p_n) = 0 \quad (9.28)$$



**Figure 9.5:** Geometric interpretation in the stress space of the computation of  $\sigma_{n+1}$  as the projection of  $\sigma_{n+1}^{\text{elas}}$  onto the final yield surface  $f(\sigma_{n+1}, p_{n+1}) = 0$  at time  $t_{n+1}$

and, in fact, constitutes the discrete form of the *consistency condition*. Solving this equality for  $\Delta p_n$  is possible as the other quantities therein are known (either from the initial state  $\mathcal{S}_n$  or the prescribed strain increment  $\Delta \varepsilon_n$ ).

Summarizing the steps described above, we obtain the *radial return algorithm* (Box 9.1) for the incremental integration of the constitutive relations.

**Box 9.1:** Radial return algorithm (von Mises criterion, isotropic hardening)

1. Compute elastic prediction

$$\mathbf{s}_{n+1}^{\text{elas}} = \mathbf{s}_n + 2\mu\mathcal{K} : \Delta \varepsilon_n, \quad \sigma_{n+1}^{\text{elas,eq}} = \sqrt{\frac{3}{2}} \|\mathbf{s}_{n+1}^{\text{elas}}\|$$

2. Compute  $f_{n+1}^{\text{elas}} = f(\sigma_{n+1}^{\text{elas}}, p_n) = \sigma_{n+1}^{\text{elas,eq}} - R(p_n)$  and test:

- If  $f_{n+1}^{\text{elas}} \leq 0$  (elastic evolution), update variables

$$\sigma_{n+1} = \sigma_{n+1}^{\text{elas}} = \kappa \text{Tr}(\varepsilon_n + \Delta \varepsilon_n) \mathbf{I} + \mathbf{s}_{n+1}^{\text{elas}}, \quad \Delta p_n = 0$$

- If  $f_{n+1}^{\text{elas}} > 0$  (elastoplastic evolution):

- (i) Solve with respect to  $\Delta p_n$  the discrete consistency condition

$$\sigma_{n+1}^{\text{elas,eq}} - 3\mu\Delta p_n - R(p_n + \Delta p_n) = 0;$$

- (ii) Compute the increment of plastic strain:

$$\Delta \varepsilon_n^{\text{P}} = \frac{3\Delta p_n}{2\sigma_{n+1}^{\text{elas,eq}}} \mathbf{s}_{n+1}^{\text{elas}};$$

- (iii) Update variables:

$$\sigma_{n+1} = \kappa \text{Tr}(\varepsilon_n + \Delta \varepsilon_n) \mathbf{I} + \mathbf{s}_{n+1}^{\text{elas}} - 2\mu\Delta \varepsilon_n^{\text{P}}.$$

This algorithm outputs  $\sigma_{n+1}$  given (i) the mechanical state at time  $t_n$  and (ii) the strain increment  $\Delta \varepsilon_n = \varepsilon[\Delta \mathbf{u}_n]$  over the time interval  $[t_n, t_{n+1}]$ . Its action can therefore be symbolically denoted

$$\sigma_{n+1} = \Sigma(\Delta \varepsilon_n; \mathcal{S}_n), \quad (9.29)$$

### 9.3.4 Case of linear isotropic hardening

For any isotropic hardening characterized by the function  $p \mapsto R(p)$ , the discrete plastic multiplier  $\Delta p_n$  is found by solving a nonlinear scalar equation, the discrete consistency condition (9.28) of unknown  $\Delta p_n$ . The computation of  $\Delta p_n$  is therefore generally based on an iterative method, such as the Newton method for scalar unknown as defined in Section 8.2.1.

In the particular case of linear isotropic hardening (9.10), equation (9.28) is linear in  $\Delta p_n$ , and its solution is given by:

$$\Delta p_n = \frac{\sigma_{n+1}^{\text{elas,eq}} - \sigma_0 - hp_n}{3\mu + h}. \quad (9.30)$$

This value is then used in step (2-ii) of the radial return algorithm.

### 9.3.5 Radial return algorithm in a plane-strain analysis

For a 2D plane-strain analyses, the radial return algorithm for the von Mises criterion with linear isotropic hardening (plastic part) and isotropic linear behavior (elastic part) is implemented in the function `RR_VonMises_2A_R`. The strain increment tensor  $\Delta\epsilon_n$  is represented using engineering notation (Section 3.1.4) by a list collecting the three nonzero components, of the form

$$\{\Delta\epsilon_n\} = \{\Delta\epsilon_{n,11} \quad \Delta\epsilon_{n,22} \quad 2\Delta\epsilon_{n,12}\}^T.$$

However, since in plane-strain, the 33-component is *a priori* nonzero for stress tensors and deviators, and are compulsorily required for the integration of the plastic part of the constitutive relations, we choose to represent any such tensor  $\mathbf{a}$  by a list collecting the four expected nonzero components:

$$\{a\} = \{a_{11} \quad a_{22} \quad a_{33} \quad a_{12}\}^T.$$

Input variables of the function `RR_VonMises_2A_R` are: (i) the initial stress  $\{\sigma_n\} \rightarrow$  sigma (four-component list), (ii) the initial cumulative plastic strain  $p_n \rightarrow$  p, (iii) a strain increment  $\{\Delta\epsilon_n\} \rightarrow$  Deps (three-component list), and (iv) the list mate with material parameters (Young modulus E, Poisson coefficient nu, hardening coefficient h and yield limit sigma0). The output variables are (i) the increment of cumulative plastic strain  $\Delta p_n \rightarrow$  Dp and (ii) the new estimate of stress  $\{\sigma_{n+1}\} \rightarrow$  sigma\_hat (four-component list).

```
function [Dp,sigma_hat]=RR_VonMises_2A_R(mate,sigma,p,Deps)

E=mate(1); % Young modulus
nu=mate(2); % Poisson coefficient
sigma0=mate(3); % Yield limit
h=mate(4); % hardening coefficient
mu=E/(2*(1+nu)); % Lamé coeff
kappa=E/(3*(1-2*nu)); % bulk modulus
```

We then first compute the deviator  $\{\Delta e_n\} \rightarrow$  De of a strain increment  $\{\Delta\epsilon_n\}$  in the form of a 4-component list

$$\{\Delta e_n\} = \{\Delta\epsilon_{n,11} \quad \Delta\epsilon_{n,22} \quad 0 \quad \Delta\epsilon_{n,12}\} - \frac{1}{3}\text{Tr}[\Delta\epsilon_n]\{1 \ 1 \ 1 \ 0\}^T,$$

with  $\text{Tr}[\Delta\epsilon_n] = (\Delta\epsilon_{n,11} + \Delta\epsilon_{n,22}) \rightarrow \text{trDeps}$ .

```
vl=[1 1 1 0]'; % identity tensor
trDeps=Deps(1)+Deps(2); % trace of incr. of strain
De=[Deps(1:2); 0; Deps(3)/2]-1/3*trDeps*vl; % full 3D deviatoric tensor
```

We are now ready to compute the elastic prediction  $\{\sigma_{n+1}^{\text{elas}}\} \rightarrow$  sigma\_elas using (9.20),

$$\{\sigma_{n+1}^{\text{elas}}\} = \{\sigma_n\} + \kappa \text{Tr}[\Delta\epsilon_n]\{1 \ 1 \ 1 \ 0\}^T + 2\mu\{\Delta e_n\}$$

and then, the trace  $\text{Tr}[\sigma_{n+1}^{\text{elas}}] \rightarrow \text{trsigma}$ , the deviator  $\{s_{n+1}^{\text{elas}}\} \rightarrow \text{s\_elas}$  and the equivalent stress  $\sigma_{n+1}^{\text{elas,eq}} \rightarrow \text{sigeq\_elas}$ ,

$$\sigma_{n+1}^{\text{elas,eq}} = \left( \frac{3}{2} \left[ (s_{n+1,11}^{\text{elas}})^2 + (s_{n+1,22}^{\text{elas}})^2 + (s_{n+1,33}^{\text{elas}})^2 + 2(s_{n+1,12}^{\text{elas}})^2 \right] \right)^{1/2}.$$

Next, the yield function  $f_{n+1}^{\text{elas}} \rightarrow f_{\text{elas}}$  is evaluated.

```
sigma_elas=sigma+kappa*trDeps*vl+2*mu*De;          % elastic pred. (stress)
trsigma=sum(sigma_elas(1:3));                      % volumetric part (stress)
s_elas=sigma_elas-1/3*trsigma*vl;                  % deviatoric elastic pred. (stress)
sigeq_elas=sqrt(1.5*...                             % equivalent deviatoric stress
    (s_elas(1)^2+s_elas(2)^2+...
    s_elas(3)^2+2*s_elas(4)^2));

f_elas=sigeq_elas-h*p-sigma0;
```

If  $f_{\text{elas}} > 0$ , the plastic flow is evaluated by computing the increment of cumulative plastic strain ( $\Delta p_n \rightarrow Dp$ ) according to (9.30), followed by the update of the stress  $\{\sigma_{n+1}\} \rightarrow \text{sigma\_hat}$ . Otherwise, if  $f_{\text{elas}} \leq 0$ , the elastic prediction represents the solution.

```
if(f_elas>0)                                       % if plastic process
    Dp=f_elas/(3*mu+h);                          % increment of plastic eq. strain
    Depsp=3/2*Dp*s_elas/sigeq_elas;              % increment of plastic strain
    sigma_hat=sigma_elas-2*mu*Deps;               % new total stress
else                                              % elseif elastic process
    Dp=0;
    sigma_hat=sigma_elas;                        % new total stress
end
```

### 9.3.6 Generalization

The algorithmic form  $\sigma_{n+1} = \Sigma(\Delta \varepsilon_n, S_n)$  for the local integration of the constitutive relations, and in particular the idea of finding  $\sigma_{n+1}$  by projection of  $\sigma_{n+1}^{\text{elas}}$  onto the current yield surface, is applicable to many other nonlinear constitutive models involving e.g. plasticity, but also viscosity, damage... The concept of radial return, namely the fact that the projection is collinear to a radial direction in the elasticity domain, stems from the hypothesis of isotropy in the stress space of the criterion considered (von Mises). There are criteria that do not assume such isotropy. The reader interested in a more complete presentation of algorithms associated with other constitutive relations may refer to the books by Simo and Hughes (1998) or Besson, Cailletaud, Chaboche, and Forest (2009).

### 9.3.7 Deviation from radiality and error in the time integration

It is interesting to try to evaluate the error due to the time discretization. To do this, we return to the relation between discrete and continuous variables (in time). On the one hand, we have by definition of the cumulative plastic strain  $p(t)$ :

$$\Delta p_n = p_{n+1} - p_n = \int_{t_n}^{t_{n+1}} \dot{p}(u) du.$$

On the other hand, using the normality rule in its continuous version (9.11), we have

$$\Delta \varepsilon_n^P = \varepsilon_{n+1}^P - \varepsilon_n^P = \int_{t_n}^{t_{n+1}} \dot{\varepsilon}^P(u) du = \sqrt{\frac{3}{2}} \int_{t_n}^{t_{n+1}} \dot{p}(u) N(u) du. \quad (9.31)$$

Two cases then arise:

- (i) The normal  $\mathbf{N}$  to the yield surface is constant over the time step  $t_n \leq t \leq t_{n+1}$ . In this case, integration of equation (9.31) is straightforward and gives

$$\Delta \epsilon_n^P = \Delta p_n \sqrt{\frac{3}{2}} \mathbf{N}_{n+1},$$

which is exactly the discrete version of the normality rule (9.19c). Since equations (9.19a,b) do not involve approximation, the time-discrete version of the constitutive model is exact, and the time discretization does not introduce errors. This situation corresponds to the case where the elastic predictor  $\sigma_{n+1}^{\text{elas}}$  is collinear to the initial stress  $\sigma_n$  (Figure 9.5).

- (ii) The normal  $\mathbf{N}$  to the yield surface varies over the time step  $t_n \leq t \leq t_{n+1}$ . In this case, we have

$$\Delta \epsilon_n^P - \Delta p_n \sqrt{\frac{3}{2}} \mathbf{N}_{n+1} = \sqrt{\frac{3}{2}} \int_{t_n}^{t_{n+1}} \dot{p}(u) [\mathbf{N}(u) - \mathbf{N}_{n+1}] du,$$

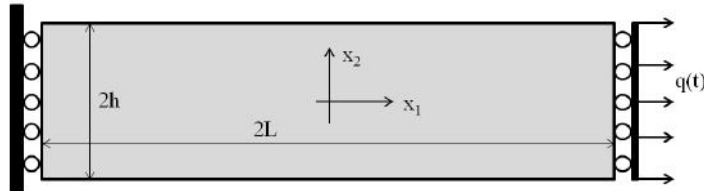
and the right-hand side measures the error arising from time discretization. Note that this error increases with the deviation from radiality  $\mathbf{N}(u) - \mathbf{N}_{n+1}$  over the time step.

#### 9.4 Example: illustration of the radial return algorithm

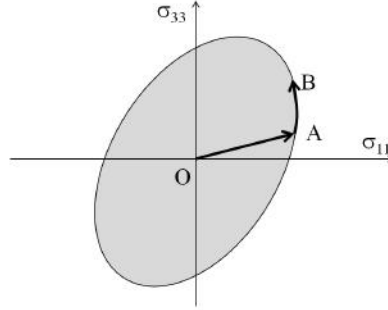
The radial return algorithm is exact over a time step when the stress evolution is radial, i.e. collinear to the initial stress state in the stress space (Sec. 9.3.7). This is the case for the uniaxial tension-compression test of a cylindrical specimen.

In the following example we illustrate the radial return algorithm in a simple situation (involving strains and stresses that are spatially constant) which is however not radial. Under the plane strain assumption, we consider a solid whose planar section occupies the rectangle  $\Omega = \{-L \leq x_1 \leq L, -h \leq x_2 \leq h\}$  (Figure 9.6). A displacement  $q(t)$  along the direction  $x_1$  is prescribed on the side  $x_1 = L$ , while sides  $x_2 = \pm h$  are traction-free boundaries. The boundary conditions are therefore

$$\begin{aligned} u_1(L, x_2) &= q(t) & u_1(-L, x_2) &= 0 & T_1(x_1, \pm h) &= 0 \\ T_2(L, x_2) &= 0 & T_2(-L, x_2) &= 0 & T_2(x_1, \pm h) &= 0 \end{aligned}$$



**Figure 9.6:** Elongation of a bar (plane strain): geometry and notations.



**Figure 9.7:** Elongation of a bar (plane strain): elastic (from  $O$  to  $A$ ) and plastic (from  $A$  to  $B$ ) load path in the  $(\sigma_{11}, \sigma_{33})$ -plane. The path  $(AB)$  is clearly non-radial.

The constitutive material is homogenous, isotropic, elastic-perfectly plastic. The initial state is natural (stress-free). The loading path consists in increasing the prescribed displacement  $q(t)$  from 0. The exact solution, which can be found in Suquet (2004), will be used to demonstrate the deviation from radiality (temporal integration error) of the radial return algorithm (Section 9.3.7). The only nonzero components of the strains and stresses are  $\varepsilon_{11}, \varepsilon_{22}$  and  $\sigma_{11}, \sigma_{33}$ .

The particularity of this example lies in the fact that the load  $q(t)$  may be increased beyond the level  $q_0$  corresponding to the elastic limit, even though (a) the material is perfectly plastic and (b) the stress state is spatially uniform. Stress states for load levels beyond  $q_0$  then move on the curve  $f(\sigma_{11}, \sigma_{33}) = 0$  (von Mises criterion) so that plastic evolutions are not radial (Figure 9.7).

We examine here the discrete-time integration by means of the radial return algorithm. To do this, at each time  $t_n$ , a strain increment  $(\Delta\varepsilon_{11}, \Delta\varepsilon_{22})$  is applied, with  $\Delta\varepsilon_{11} = \Delta q/2L$  so as to satisfy the prescribed displacement conditions. The stresses  $(\sigma_{n+1,11}, \sigma_{n+1,22}, \sigma_{n+1,33})$  are then computed by the radial return algorithm. Equilibrium requires

$$\sigma_{n+1,22} = 0, \quad (9.32)$$

and the value of the strain increment  $\Delta\varepsilon_{22}$  is found (by a Newton method) so that the stress state satisfies this condition.

#### 9.4.1 Uniaxial elastoplastic test: numerical solution by code strip\_plast

We aim at determining the strain and stress histories in the bar as a function of the prescribed displacement  $q(t)$ , assuming spatially-constant strains and stresses and  $\varepsilon_{12} = 0$ . Using these assumptions (which are specific to this example), the global equilibrium problem of the bar reduces to the scalar equation  $\sigma_{22} = 0$ , and the prescribed displacement condition to  $\varepsilon_{11} = q/(2L)$ . Neither relationship involve  $\varepsilon_{22}$  and  $\sigma_{11}$ .

We assume spatially linear displacements, so that the relation  $\varepsilon_{11} = q/(2L)$  holds everywhere. It only remains to satisfy (a) the equilibrium condition  $\sigma_{22} = 0$  and (b) the elastoplastic constitutive relations. The radial return algorithm will be used for meeting

requirement (b). While specific to this example, the following scheme, implemented in the code `strip_plast`, nevertheless bears many commonalities with the general solution strategy to be presented in Sections 9.5- 9.7 (see also exercise 9.10).

The main computational task consists in finding the solution after one load step. Starting from a known mechanical state  $\mathcal{S}_n = (\boldsymbol{\sigma}_n, p_n, \boldsymbol{\varepsilon}_n)$  (such that in particular  $\sigma_{n,22} = 0$ ), the loading is increased by a displacement increment  $\Delta q$ , resulting in an axial strain increment  $\Delta \varepsilon_{n,11}^{[1]} = \Delta q / (2L)$ . The new mechanical state  $\mathcal{S}_{n+1} = (\boldsymbol{\sigma}_{n+1}, p_{n+1}, \boldsymbol{\varepsilon}_{n+1})$  is sought.

Temporarily assuming that the load increment is purely elastic (elastic predictor), the equilibrium condition  $\sigma_{22} = 0$  gives

$$\Delta \varepsilon_{n,22}^{[1]} = -\frac{A_{21}}{A_{22}} \Delta \varepsilon_{n,11}^{[1]} = -\frac{\nu}{(1-\nu)} \Delta \varepsilon_{n,11}^{[1]}.$$

We can then apply the radial return algorithm, which was analysed in Section 9.3.3, with the elastic predictor  $\Delta \varepsilon_n^{[1]}$ . This yields, among other things, an estimate of the stress  $\boldsymbol{\sigma}_{n+1}^{[1]} = \boldsymbol{\Sigma}(\Delta \varepsilon_n^{[1]}, \mathcal{S}_n)$ . If the component  $\sigma_{n+1,22}^{[1]}$  is *a priori* nonzero, the initial assumption (elastic evolution) that enabled us to define  $\Delta \varepsilon_{n,22}^{[1]}$  was incorrect and the procedure is then repeated with

$$\Delta \varepsilon_n^{[2]} = \Delta \varepsilon_n^{[1]} + \delta \varepsilon_n^{[1]},$$

where  $\delta \varepsilon_{n,11}^{[1]} = 0$  so that the prescribed displacement condition  $\Delta \varepsilon_{n,11}^{[2]} = \Delta q / (2L)$  remains satisfied. The only free parameter is  $\delta \varepsilon_{n,22}^{[1]}$ . Let us choose, for instance, the latter so that  $\boldsymbol{\sigma}_{n+1}^{[1]} + \mathcal{A} : \delta \varepsilon_n^{[1]}$  is a tensor with zero 22-component. This yields

$$\delta \varepsilon_{n,22}^{[1]} = -\frac{1}{A_{22}} \sigma_{n+1,22}^{[1]} = -\frac{(1+\nu)(1-2\nu)}{(1-\nu)E} \sigma_{n+1,22}^{[1]}$$

It is noted that this choice of  $\delta \varepsilon_{n,22}^{[1]}$ , although reasonable, is not unique. The radial return algorithm is then restarted and we iterate this process until convergence, i.e. until satisfying equilibrium within a given tolerance ( $|\sigma_{n+1,22}^{[k]}| \leq \varepsilon$ ).

The above steps are implemented in the code `strip_plast` as follows:

- Input parameters are defined: half-length of the rectangle (L), material parameters (mate), history of prescribed displacement (q) (here from q=0 to q=0.1 by step of 0.02):

```
L=1; % halflength of strip
E=1000; % Young modulus
nu=.1; % Poisson coefficient
sigma0=1; % yield limit
H=0; % hardening parameter
mate= [E nu sigma0 H]; % material parameters
q=[0:.02:.1]; % history of prescribed disp.
numstep=length(q); % number of steps
```

- Initialisation of the initial mechanical state, here a natural state (stress-free state)

```
p=0; % plastic strains (initia.)
sigma=zeros(4,1); % initialization of stresses
toll=1.d-4; % user fixed tolerance
```

- The final phase consists of two loops: *Outer loop* (for-loop) which compute the loading path and update the mechanical state,

```

for istep=2:numstep,           % loop over all load steps
    Dq=q(istep)-q(istep-1);    % delta of prescribed displ.
    iter=0;                    % iteration counter
    resid=1;                   % Residual initialisation
    Dp=0;                      % plastic increment in step
    Deps=zeros(3,1);           % initialization strain inc.

    ... Inner loop: compute sigma_hat and Dp ...

    p=p+Dp;                    % Update plastic multiplier
    sigma=sigma_hat;           % Update stress
end                             % end loop over time steps

```

*Inner loop* (while-loop) which execute the iterative method described above for each load step. It leads to the satisfaction of both elastoplastic constitutive relation and equilibrium.

```

while resid > toll,            % Find solution ...
    iter=iter+1;               % ... for the load step
    if iter==1                 % First iteration
        Deps=Dq/(2*L)*[1 -nu/(1-nu) 0]';
    else                       % Generic iteration
        deps2=-sigma_hat(2)*(1+nu)*(1-2*nu)/(E*(1-nu));
        Deps=Deps+[0 deps2 0]';
    end
    [Dp,sigma_hat]=RR_VonMises_2A_R(mate,sigma,p,Deps);
    resid=abs(sigma_hat(2));    % abs value of sig_22 (residual)
end

```

It is worth noting that, as mentioned above, the computation of  $\Delta \epsilon_n^{[k]} \rightarrow \text{Deps}$  is different for the first iteration. Finally, we use the function `RR_VonMises_2A_R` described in Section 9.3.5 to run the radial return algorithm. The code ends by the plot of stresses  $\sigma_{11}$  and  $\sigma_{33}$  in function of the strain  $\epsilon_{11}$  for both the exact solution and the numerical solution (Fig. 9.8).

**Exercise 9.1.** Show that, for this example, the initial displacement  $q_0$  associated with the beginning the plastic flow is given by :

$$q_0 = \frac{(1 - \nu^2)}{\sqrt{1 - \nu} + \nu^2} \frac{\sigma_0 2L}{E}$$

Modify `Strip_plast` so as to compute the initial displacement  $q_0$  and simulate the loading path  $[q_0, 6q_0]$ .

**Exercise 9.2** (3D analysis). Develop a function `RR_VonMises_3D_R` that implement the radial return algorithm for the von Mises criterion with linear isotropic hardening (plastic part) and isotropic linear behavior (elastic part) for a 3D analysis. Use it in `strip_plast` and compare with the exact solution provided for this example.

**Exercise 9.3** (Von Mises and nonlinear isotropic hardening). Modify the function `RR_VonMises_2A_R` so as to consider the case with the nonlinear isotropic hardening  $R(p) = \sigma_0 + hp^{1/M_0}$ , where  $M_0$  is a user-supplied material parameter.



**Exercise 9.4** (Von Mises and kinematic hardening). *Propose a numerical scheme for the local integration of the following constitutive law*

$$\begin{aligned}
 \boldsymbol{\sigma} &= \mathcal{A}:(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^P) && (\text{elasticity}), \\
 f(\boldsymbol{\sigma}, \mathbf{X}) &= (\boldsymbol{\sigma} - \mathbf{X})^{eq} - \sigma_0 \leq 0 && (\text{yield criterion}), \\
 \dot{\boldsymbol{\varepsilon}}^P &= \dot{\gamma} \frac{\partial f}{\partial \boldsymbol{\sigma}}(\boldsymbol{\sigma}, \mathbf{X}), && (\text{normality rule}), \\
 \dot{\mathbf{X}} &= 2C\dot{\boldsymbol{\varepsilon}}^P, \\
 \dot{\gamma} f(\boldsymbol{\sigma}, \mathbf{X}) &= 0 && (\text{complementarity condition}),
 \end{aligned}$$

where  $C$  is a user-supplied material parameter, and develop the associated function `RR_VonMises_2A.X`.

**Exercise 9.5** (Drucker-Prager). *Propose a numerical scheme for the local integration of the following constitutive law*

$$\begin{aligned}
 \boldsymbol{\sigma} &= \mathcal{A}:(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^P) && (\text{elasticity}), \\
 f(\boldsymbol{\sigma}) &= \sigma^{eq} + b \operatorname{tr} \boldsymbol{\sigma} - \sigma_0 \leq 0 && (\text{yield criterion}), \\
 \dot{\boldsymbol{\varepsilon}}^P &= \dot{\gamma} \frac{\partial f}{\partial \boldsymbol{\sigma}}(\boldsymbol{\sigma}), && (\text{normality rule}), \\
 \dot{\gamma} f(\boldsymbol{\sigma}) &= 0 && (\text{complementarity condition}),
 \end{aligned}$$

where  $b$  is a user-supplied material parameter, and develop the associated MATLAB function `RR_DPrager_2A.R`.

### 9.4.2 Numerical results

The results obtained by numerical integration are presented on Figure 9.8 for  $q(t)$  ranging from  $q_0$  to  $6q_0$  (where  $q_0$  corresponds to the beginning of the plastic flow). The stresses  $\sigma_{11}, \sigma_{33}$  are plotted in function of the strain  $\varepsilon_{11}$ .

Three subdivisions of the loading path  $[q_0, 6q_0]$ , which acts as the time interval, are considered: 5 steps ( $\Delta q = q_0$ ), 20 steps ( $\Delta q = q_0/4$ ) and 50 steps ( $\Delta q = q_0/10$ ). The results for 5 load steps appear to be significantly less accurate than the others, reflecting the fact that the deviation from radiality over each load step is too large.

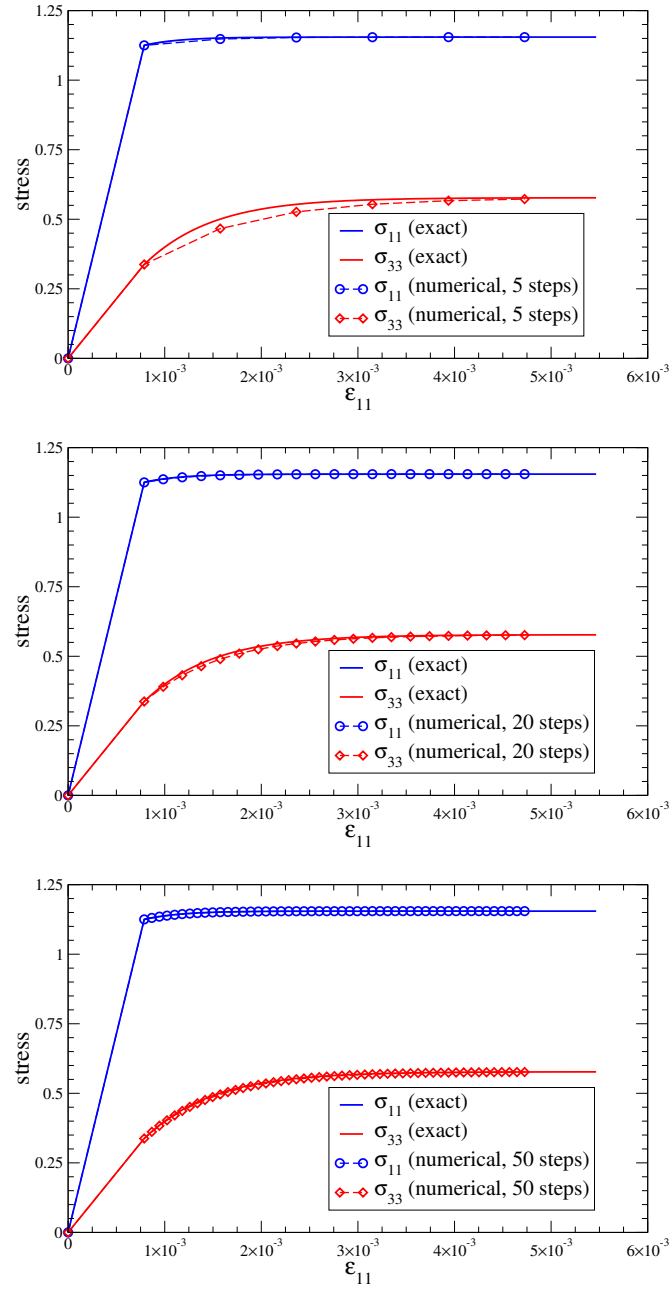
## 9.5 Global equilibrium: linearized weak formulation

The weak formulation of the global equilibrium of an elastoplastic structure at time  $t_{n+1}$  reads (equation (9.14) repeated for convenience):

$$\int_{\Omega} \boldsymbol{\sigma}_{n+1} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = \int_{\Omega} \rho \mathbf{f}_{n+1} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}_{n+1}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}). \quad (9.33)$$

Introducing the constitutive relations in an approximate way by using the algorithm  $\Sigma$  defined in Section 9.3, the elastoplastic problem to be solved is then:

$$\text{find } \Delta \mathbf{u}_n \in \mathcal{C}(\Delta \mathbf{u}_n^D), \quad \mathcal{R}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n) = 0 \quad \forall \mathbf{w} \in \mathcal{C}(\mathbf{0}), \quad (9.34)$$



**Figure 9.8:** Elongation of a bar (plane strain): exact solution, and radial return algorithm solution using  $M$  load steps after first plastic flow, with  $M = 5$  (top),  $M = 20$  (middle) and  $M = 50$  (bottom).

where, setting  $\Delta \epsilon_n = \epsilon[\Delta \mathbf{u}_n]$ , the *residual*  $\mathcal{R}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n)$  is defined by:

$$\mathcal{R}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n) = -\mathcal{F}_{n+1}^{\text{int}}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n) - \mathcal{F}_{n+1}^{\text{ext}}(\mathbf{w}), \quad (9.35)$$

and

$$\mathcal{F}_{n+1}^{\text{ext}}(\mathbf{w}) = \int_{\Omega} \rho \mathbf{f}_{n+1} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}_{n+1}^D \cdot \mathbf{w} \, dS \quad (9.36)$$

is the virtual work of external forces, while

$$\mathcal{F}_{n+1}^{\text{int}}(\Delta \mathbf{u}_n; \mathbf{w}, \mathcal{S}_n) = - \int_{\Omega} \boldsymbol{\Sigma}(\epsilon[\Delta \mathbf{u}_n]; \mathcal{S}_n) : \epsilon[\mathbf{w}] \, dV \quad (9.37)$$

is the virtual work of internal forces, which is clearly nonlinear in  $\Delta \mathbf{u}_n$ . Accordingly, an elastoplastic analysis requires numerical methods for solving problem (9.34). As no direct numerical method is available, an iterative process has to be developed. A natural solution strategy rests upon a Newton-type algorithm, as already done in Chapter 8 for nonlinear elasticity.

The Newton iterative method applied to the global equilibrium problem (9.34) proceeds by generating iteratively a sequence of approximations  $\Delta \mathbf{u}_n^{[k]}$  of  $\Delta \mathbf{u}_n$  ( $k \geq 0$  being the iteration counter). It is in fact equivalent, and more convenient in practice, to think in terms of successive corrections  $\delta \mathbf{u}_n^{[k]}$  defined by

$$\Delta \mathbf{u}_n^{[k+1]} = \Delta \mathbf{u}_n^{[k]} + \delta \mathbf{u}_n^{[k]}. \quad (9.38)$$

where each correction  $\delta \mathbf{u}_n^{[k]}$  is found by solving the equation obtained by linearizing (9.34) about the current iterate  $\Delta \mathbf{u}_n^{[k]}$ , which formally reads

$$\mathcal{R}(\Delta \mathbf{u}_n^{[k]}; \mathbf{w}, \mathcal{S}_n) + \langle \mathcal{R}'(\Delta \mathbf{u}_n^{[k]}; \mathbf{w}, \mathcal{S}_n), \delta \mathbf{u}_n^{[k]} \rangle = 0, \quad (9.39)$$

The iterations are stopped as soon as the norm of the residual is within a predefined relative tolerance  $\epsilon$ . In other words, the approximate solution of the global nonlinear problem is given by

$$\Delta \mathbf{u}_n \approx \Delta \mathbf{u}_n^{[k+1]} = \Delta \mathbf{u}_n^{[k]} + \delta \mathbf{u}_n^{[k]}, \quad (9.40)$$

where  $k$  is the iteration number such that

$$\|\mathcal{R}(\Delta \mathbf{u}_n^{[k]}; \mathbf{w}, \mathcal{S}_n)\| < \epsilon \|\mathcal{R}(\mathbf{0}; \mathbf{w}, \mathcal{S}_n)\|,$$

$\|\cdot\|$  being an appropriate norm that does not need to be specified here.

Building the linearization (9.39) requires the numerical construction of the tangent linear operator  $\mathcal{R}'$ , associated with the residual  $\mathcal{R}$ , defined by

$$\mathcal{R}(\mathbf{v} + \mathbf{z}; \mathbf{w}, \mathcal{S}_n) - \mathcal{R}(\mathbf{v}; \mathbf{w}, \mathcal{S}_n) = \langle \mathcal{R}'(\mathbf{v}; \mathbf{w}, \mathcal{S}_n), \mathbf{z} \rangle + o(\|\mathbf{z}\|).$$

The existence of a linear operator  $\mathcal{R}'$  satisfying the above condition constitutes the condition under which  $\mathcal{R}$  is differentiable. It is therefore important to establish a procedure performing the numerical computation of the tangent linear operator. The latter will of course take the form of a tangent  $N \times N$  matrix, but it is convenient to use the continuous form of the residual as a basis for this treatment.

### 9.5.1 Global and local tangent operators

In the definition (9.35) of the residual, only the first term (bilinear form associated with internal forces (9.37)) depends on  $\Delta\epsilon = \epsilon[\Delta\mathbf{u}]$ . Hence, the construction of the global tangent linear operator  $\mathcal{R}'$  in practice rests on the local tangent linear operator  $\Sigma'$  associated with  $\Sigma$ :

$$\begin{aligned}\Sigma(\Delta\epsilon + \delta\epsilon; \mathcal{S}_n) - \Sigma(\Delta\epsilon; \mathcal{S}_n) &= \langle \Sigma'(\Delta\epsilon; \mathcal{S}_n), \delta\epsilon \rangle + o(\|\delta\epsilon\|) \\ &= \frac{\partial \Sigma}{\partial \Delta\epsilon}(\Delta\epsilon; \mathcal{S}_n) : \delta\epsilon + o(\|\delta\epsilon\|)\end{aligned}$$

We will call *local tangent operator*, or alternatively *tangent elastoplastic moduli*, the fourth-order tensor  $\mathcal{A}^{\text{EP}}$  given by

$$\mathcal{A}^{\text{EP}}(\Delta\epsilon; \mathcal{S}_n) = \frac{\partial \Sigma}{\partial \Delta\epsilon}(\Delta\epsilon; \mathcal{S}_n), \quad (9.41)$$

so that the linear tangent operator is defined through

$$\langle \Sigma'(\Delta\epsilon; \mathcal{S}_n), \delta\epsilon \rangle = \mathcal{A}^{\text{EP}}(\Delta\epsilon; \mathcal{S}_n) : \delta\epsilon. \quad (9.42)$$

In other words, the local tangent operator  $\mathcal{A}^{\text{EP}}$  is the tensor gathering the tangent moduli that link (to first order in  $\delta\epsilon$ ) stress variations about  $\sigma_{n+1} = \Sigma(\Delta\epsilon; \mathcal{S}_n)$  to strain variations about  $\epsilon_{n+1} = \epsilon_n + \Delta\epsilon$ . Lastly, using

$$\epsilon[\Delta\mathbf{u} + \delta\mathbf{u}] = \epsilon[\Delta\mathbf{u}] + \epsilon[\delta\mathbf{u}] = \Delta\epsilon + \delta\epsilon$$

and setting  $\Delta\epsilon = \Delta\epsilon_n^{[k]} = \epsilon[\Delta\mathbf{u}_n^{[k]}]$  and  $\delta\mathbf{u} = \delta\mathbf{u}_n^{[k]}$  and anticipating that  $\mathcal{A}^{\text{EP}}(\Delta\epsilon_n^{[k]}; \mathcal{S}_n)$  has the same symmetries as  $\mathcal{A}$ , the global tangent linear operator  $\mathcal{R}'$  can be written

$$\langle \mathcal{R}'(\Delta\mathbf{u}_n^{[k]}; \mathbf{w}, \mathcal{S}_n), \delta\mathbf{u}_n^{[k]} \rangle = \int_{\Omega} \epsilon[\delta\mathbf{u}_n^{[k]}] : \mathcal{A}^{\text{EP}}(\Delta\epsilon_n^{[k]}; \mathcal{S}_n) : \epsilon[\mathbf{w}] \, dV. \quad (9.43)$$

### 9.5.2 Computation of the local tangent operator

The local tangent operator (9.41) comes from the differentiation of  $\Sigma(\Delta\epsilon_n; \mathcal{S}_n)$  (radial return algorithm) with respect to  $\Delta\epsilon_n$ . Hence, a natural way of finding its explicit expression consists in differentiating with respect to  $\Delta\epsilon_n$  the steps of the radial return algorithm given in Box 9.1.

In both cases of elastic or plastic evolution, the final stress  $\sigma_{n+1} = \Sigma(\Delta\epsilon_n; \mathcal{S}_n)$  given by radial return algorithm can be written using (9.19a), i.e.:

$$\sigma_{n+1} = \sigma_n + \mathcal{A} : \Delta\epsilon_n - 2\mu\Delta\epsilon_n^{\text{P}}. \quad (9.44)$$

**Elastic evolution.** In this case, there is no plastic strain evolution:  $\Delta p_n = 0$ ,  $\Delta\epsilon_n^{\text{P}} = \mathbf{0}$ , and (9.44) directly gives

$$\frac{\partial \Sigma}{\partial \Delta\epsilon_n}(\Delta\epsilon_n; \mathcal{S}_n) = \mathcal{A}. \quad (9.45)$$

**Elastoplastic evolution.** In this case, (9.44) becomes

$$\frac{\partial \Sigma}{\partial \Delta \varepsilon_n}(\Delta \varepsilon_n; \mathcal{S}_n) = \mathcal{A} - 2\mu \frac{\partial \Delta \varepsilon_n^P}{\partial \Delta \varepsilon_n}, \quad (9.46)$$

which then requires the evaluation of the fourth-order tensor  $\mathcal{D}$  defined by

$$\mathcal{D}(\Delta \varepsilon_n; \mathcal{S}_n) = 2\mu \frac{\partial \Delta \varepsilon_n^P}{\partial \Delta \varepsilon_n}, \quad (9.47)$$

or equivalently, using the expression of  $\Delta \varepsilon_n^P$  given in step (2(ii)) of the radial return algorithm (Box 9.1), by:

$$\mathcal{D}(\Delta \varepsilon_n; \mathcal{S}_n) = 3\mu \frac{\partial}{\partial \Delta \varepsilon_n} \left( \frac{\Delta p_n}{\sigma_{n+1}^{\text{elas,eq}}} \mathbf{s}_{n+1}^{\text{elas}} \right). \quad (9.48)$$

where  $\mathbf{s}_{n+1}^{\text{elas}}$  is the deviatoric part of the trial elastic stress  $\sigma_{n+1}^{\text{elas}}$  defined by (9.20). Accordingly, we first need to derive the derivatives of  $\mathbf{s}_{n+1}^{\text{elas}}$ ,  $\sigma_{n+1}^{\text{elas,eq}}$  and  $\Delta p_n$  with respect to  $\Delta \varepsilon_n$ . The first two are easily obtained by a direct calculation: noting that  $\mathbf{s} = \mathcal{K} : \mathbf{s}$  for any deviatoric (i.e. symmetric and traceless) tensor  $\mathbf{s}$ , we get:

$$\frac{\partial}{\partial \Delta \varepsilon_n} \mathbf{s}_{n+1}^{\text{elas}} = \frac{\partial}{\partial \Delta \varepsilon_n} (2\mu \mathcal{K} : \Delta \varepsilon_n) = 2\mu \mathcal{K}, \quad (9.49a)$$

$$\begin{aligned} \frac{\partial}{\partial \Delta \varepsilon_n} \sigma_{n+1}^{\text{elas,eq}} &= \frac{\partial}{\partial \Delta \varepsilon_n} \sqrt{\frac{3}{2} (\mathbf{s}_{n+1}^{\text{elas}} : \mathbf{s}_{n+1}^{\text{elas}})^{1/2}} \\ &= \sqrt{\frac{3}{2}} \frac{2\mu}{(\mathbf{s}_{n+1}^{\text{elas}} : \mathbf{s}_{n+1}^{\text{elas}})^{1/2}} \mathcal{K} : \mathbf{s}_{n+1}^{\text{elas}} = \frac{3\mu}{\sigma_{n+1}^{\text{elas,eq}}} \mathbf{s}_{n+1}^{\text{elas}}. \end{aligned} \quad (9.49b)$$

The derivative of  $\Delta p_n$  is then obtained by differentiating the discrete form of the consistency condition (9.28) with respect to  $\Delta \varepsilon_n$ . Using (9.49b), this yields

$$\frac{3\mu}{\sigma_{n+1}^{\text{elas,eq}}} \mathbf{s}_{n+1}^{\text{elas}} - [3\mu + R'_{n+1}] \frac{\partial}{\partial \Delta \varepsilon_n} \Delta p_n = 0$$

with  $R'_{n+1} = R'(p_{n+1}) = R'(p_n + \Delta p_n)$ , and finally

$$\frac{\partial}{\partial \Delta \varepsilon_n} \Delta p_n = \frac{3\mu}{3\mu + R'_{n+1}} \frac{1}{\sigma_{n+1}^{\text{elas,eq}}} \mathbf{s}_{n+1}^{\text{elas}} \quad (9.50)$$

Performing the partial differentiation in (9.48) and using (9.49a,b) and (9.50), the final expression of  $\mathcal{D}$  is obtained as

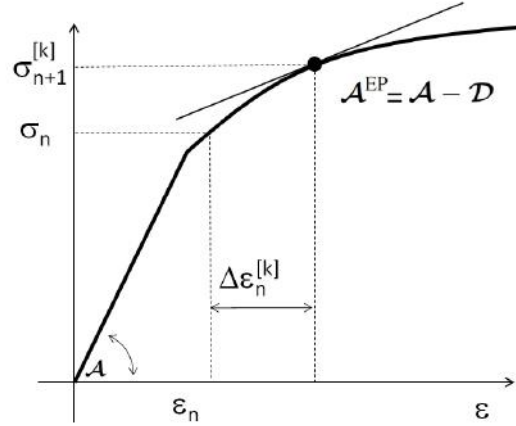
$$\mathcal{D}(\Delta \varepsilon_n; \mathcal{S}_n) = 3\mu(\gamma - \beta) \left( \frac{\mathbf{s}_{n+1}^{\text{elas}}}{\sigma_{n+1}^{\text{elas,eq}}} \otimes \frac{\mathbf{s}_{n+1}^{\text{elas}}}{\sigma_{n+1}^{\text{elas,eq}}} \right) + 2\mu\beta \mathcal{K} \quad (9.51)$$

with

$$\beta = \frac{3\mu \Delta p_n}{\sigma_{n+1}^{\text{elas,eq}}} = 1 - \frac{R_{n+1}}{\sigma_{n+1}^{\text{elas,eq}}}, \quad \gamma = \frac{3\mu}{3\mu + R'_{n+1}} \quad (9.52)$$

where (9.28) is used to express  $\beta$  in terms of  $R_{n+1} := R(p_{n+1}) = R(p_n + \Delta p_n)$ . Note that  $\beta$ ,  $\gamma$ ,  $\mathbf{s}_{n+1}^{\text{elas}}$  et  $\sigma_{n+1}^{\text{elas,eq}}$  depend on  $\mathcal{S}_n$  and therefore as expected, the fourth-

order tensor  $\mathcal{D}$ . It provides the "plastic correction" with the elastic modulus  $\mathcal{A}$  so that  $\mathcal{A} - \mathcal{D}$  is the tangent modulus tensor (Figure 9.9). Moreover,  $\mathcal{D}$  has the same symmetry properties (1.9) as  $\mathcal{A}$  because (i)  $\mathcal{K}$  defined by (1.14) has these symmetries, and (ii)  $\mathbf{s}_{n+1}^{\text{elas}}$  is a symmetric second-order tensor.



**Figure 9.9:** Geometric interpretation of the local tangent operator  $\mathcal{A}^{\text{EP}}$

**Summary.** Definition (9.41) and expressions (9.45), (9.46), (9.47), and (9.51) lead to the following definition of the local tangent operator  $\mathcal{A}^{\text{EP}}$ :

$$\mathcal{A}^{\text{EP}}(\Delta\epsilon_n, \mathcal{S}_n) = \begin{cases} \mathcal{A} & \text{if } f_{n+1}^{\text{elas}} < 0 \text{ (elastic),} \\ \mathcal{A} - \mathcal{D}(\Delta\epsilon_n; \mathcal{S}_n) & \text{if } f_{n+1}^{\text{elas}} > 0 \text{ (elastoplastic),} \end{cases} \quad (9.53)$$

where  $f_{n+1}^{\text{elas}} = f(\sigma_{n+1}^{\text{elas}}) - R(p_n)$  is the yield criterion (von Mises here) evaluated for a purely elastic trial evolution and  $\mathcal{D}$  is given by (9.51).

Note that the radial return mapping  $\Delta\epsilon_n \mapsto \Sigma(\Delta\epsilon_n, \mathcal{S}_n)$  is differentiable with respect to  $\Delta\epsilon_n$  in the cases (i)  $f_{n+1}^{\text{elas}} < 0$  and (ii)  $f_{n+1}^{\text{elas}} > 0$ . Case (i) corresponds to the case where the final stress is within the elasticity domain, and case (ii) to the case of an elastoplastic evolution with non-zero plastic strain rate  $\Delta p_n$  (elastoplastic load). Nonetheless, in the limit situation where  $f_{n+1}^{\text{elas}} = 0$ , which corresponds to a "neutral" evolution (final stress is on the yield surface  $f_{n+1} = 0$  but with a zero plastic strain rate  $\Delta p_n = 0$ ),  $\Sigma(\Delta\epsilon_n, \mathcal{S}_n)$  is not differentiable.

The radial return algorithm (Box 9.1) can now be supplemented with the evaluation of the local tangent moduli. The resulting algorithm is given in Box 9.2.

### 9.5.3 Local tangent operator using engineering notation

To facilitate the implementation of the construction of the local tangent operator  $\mathcal{A}^{\text{EP}}$  in a computer code, the expression of the fourth-order plastic correction tensor  $\mathcal{D}$  (9.51) is recast into engineering notation (Sec. 3.1.4) by using the six-component vector representation (3.9) for stress and strain tensors.

**Box 9.2:** Radial return algorithm with local tangent moduli  
(von Mises criterion with isotropic hardening)

1. Compute  $\mathbf{s}_{n+1}^{\text{elas}} = \mathbf{s}_n + 2\mu\mathcal{K} : \Delta\boldsymbol{\varepsilon}_n$  (elastic pred.) and  $\sigma_{n+1}^{\text{elas,eq}} = \sqrt{\frac{3}{2}} \|\mathbf{s}_{n+1}^{\text{elas}}\|$ ;
2. Compute  $f^{\text{elas}} = f(\boldsymbol{\sigma}_{n+1}^{\text{elas}}, p_n) = \sigma_{n+1}^{\text{elas,eq}} - R(p_n)$  and test:

- If  $f_{n+1}^{\text{elas}} \leq 0$ , update

$$\boldsymbol{\sigma}_{n+1} = \kappa \text{Tr}(\boldsymbol{\varepsilon}_n + \Delta\boldsymbol{\varepsilon}_n) \mathbf{I} + \mathbf{s}_{n+1}^{\text{elas}}, \quad \Delta p_n = 0, \quad \mathcal{A}^{\text{EP}} = \mathcal{A};$$

- If  $f_{n+1}^{\text{elas}} > 0$ :

- (i) Solve with respect to  $\Delta p_n$  the discrete consistency condition:

$$\sigma_{n+1}^{\text{elas,eq}} - 3\mu\Delta p_n - R(p_n + \Delta p_n) = 0;$$

- (ii) Compute constants  $\beta$  and  $\gamma$ :

$$\beta = \frac{3\mu\Delta p_n}{\sigma_{n+1}^{\text{elas,eq}}}, \quad \gamma = \frac{3\mu}{3\mu + R'(p_n + \Delta p_n)};$$

- (iii) Compute stress and plastic strain increment:

$$\boldsymbol{\sigma}_{n+1} = (1 - \beta)\mathbf{s}_{n+1}^{\text{elas}} + \kappa \text{Tr}(\boldsymbol{\varepsilon}_n + \Delta\boldsymbol{\varepsilon}_n) \mathbf{I},$$

- (iv) Compute the elastoplastic stiffness matrix  $\mathcal{A}^{\text{EP}}$ :

$$\mathcal{A}^{\text{EP}} = \mathcal{A} - 3\mu(\gamma - \beta) \left( \frac{\mathbf{s}_{n+1}^{\text{elas}}}{\sigma_{n+1}^{\text{elas,eq}}} \otimes \frac{\mathbf{s}_{n+1}^{\text{elas}}}{\sigma_{n+1}^{\text{elas,eq}}} \right) - 2\mu\beta\mathcal{K}.$$

The extraction of the deviatoric part of a second-order symmetric tensor ( $\boldsymbol{\varepsilon} \mapsto \mathcal{K} : \boldsymbol{\varepsilon}$ ) is then recast in terms of a projection matrix  $[\mathcal{K}]$ :

$$[\mathcal{K}]\{\boldsymbol{\varepsilon}\} \quad \text{with} \quad [\mathcal{K}] = \begin{bmatrix} 2/3 & -1/3 & -1/3 & 0 & 0 & 0 \\ -1/3 & 2/3 & -1/3 & 0 & 0 & 0 \\ -1/3 & -1/3 & 2/3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 \end{bmatrix}. \quad (9.54)$$

Similarly, the stress deviator and equivalent stress associated with a given stress  $\boldsymbol{\sigma}$  are computed using

$$\begin{aligned} \{s\} &= \{\sigma\} - \frac{1}{3}(\sigma_{11} + \sigma_{22} + \sigma_{33})\{1 \ 1 \ 1 \ 0 \ 0 \ 0\}^T \\ \sigma^{\text{eq}} &= \left[ \frac{3}{2}(s_{11}^2 + s_{22}^2 + s_{33}^2) + 3s_{12}^2 + 3s_{13}^2 + 3s_{23}^2 \right]^{1/2}. \end{aligned}$$

Finally, the fourth-order plastic correction tensor  $\mathcal{D}$  defined by (9.51) is recast into the matrix  $[D] \in \mathbb{R}^{6 \times 6}$

$$[D] = \frac{3\mu(\gamma - \beta)}{(\sigma_{n+1}^{\text{elas,eq}})^2} \{s_{n+1}^{\text{elas}}\} \{s_{n+1}^{\text{elas}}\}^T + 2\mu\beta[\mathcal{K}] \quad (9.55)$$

where  $\{s^{\text{elas}}\}$  is the "engineering notation" of  $s^{\text{elas}}$ , and  $\beta, \gamma$  are defined by (9.52). The final matrix representation of the local tangent modulus then reads:

$$[A^{\text{EP}}] = [A] - [D]$$

As already mentioned in Section 9.3.1, discrete time integration schemes of the form  $\Delta \varepsilon_n \rightarrow \Sigma(\Delta \varepsilon_n; \mathcal{S}_n)$  exist for a wide variety of constitutive relations. The concept of consistent tangent operator, presented here for a specific class of elastoplastic behavior, has a much broader scope.

#### 9.5.4 Radial return algorithm with local tangent moduli in plane-strain analyses

The function `RRTM_VonMises_2A_R` provides an implementation of the radial return algorithm that includes the computation of the local tangent moduli for plane strain analyses. We adopt the convention already used for the function `RR_VonMises_2A_R` (section 9.3.5): the strain tensor  $\Delta \varepsilon_n$  is represented by a list of the three nonzero components  $\{\Delta \varepsilon_n\} = \{\Delta \varepsilon_{n,11}, \Delta \varepsilon_{n,22}, 2\Delta \varepsilon_{n,12}\}$  whereas the stress tensor  $\sigma_n$  (as well as all deviatoric tensors) is represented by the list  $\{\sigma_n\} = \{\sigma_{n,11}, \sigma_{n,22}, \sigma_{n,33}, \sigma_{n,12}\}$  of its four nonzero components.

Both functions have the same input variables that is (i) the initial stress  $\{\sigma_n\} \rightarrow \text{sigma}$  (four-component list), (ii) the initial cumulative plastic strain  $p_n \rightarrow p$ , (iii) a strain increment  $\{\Delta \varepsilon_n\} \rightarrow \text{Deps}$  (three-component list), and (iv) the list `mate` with material parameters (Young modulus  $E$ , Poisson coefficient  $\nu$ , hardening coefficient  $h$  and yield limit  $\sigma_0$ ). The output variables are (i) the increment of plastic strain  $\Delta p_n \rightarrow \text{Dp}$ , (ii) the new estimate of stress  $\{\sigma_{n+1}\} \rightarrow \text{sigma\_hat}$  (four-component list) to which we add (iii) the tangent matrix  $[A^{\text{EP}}] \rightarrow \text{AEP}$ .

```
function [AEP,Dp,sigma_hat]=RRTM_VonMises_2A_R(mate,sigma,p,Deps)

E=mate(1); % Young modulus
nu=mate(2); % Poisson coefficient
sigma0=mate(3); % Yield limit
h=mate(4); % hardening coefficient
mu=E/(2*(1+nu)); % Lamé coeff
kappa=E/(3*(1-2*nu)); % bulk modulus
A = E/((1+nu)*(1-2*nu))*[1-nu,nu,0; % elastic tensor (plane strain)
                        nu,1-nu,0;
                        0,0,(1-2*nu)/2];
MK=1/3*[2 -1 -1 0; -1 2 -1 0; ... % deviatoric extractor
        -1 -1 2 0; 0 0 0 3/2];
```

where  $[A] \in \mathbb{R}^{3 \times 3} \rightarrow A$  is the matrix representation of the elasticity tensor  $\mathcal{A}$  in plane strain and  $[K] \in \mathbb{R}^{4 \times 4} \rightarrow MK$  is the projection matrix that extract the deviatoric part of any second-order symmetric tensor (four-component list) in plane-strain analysis defined by

$$[K] = \begin{bmatrix} 2/3 & -1/3 & -1/3 & 0 \\ -1/3 & 2/3 & -1/3 & 0 \\ -1/3 & -1/3 & 2/3 & 0 \\ 0 & 0 & 0 & 1/2 \end{bmatrix}.$$



We then follow the same procedure as in `RR_VonMises_2A_R`, up to the evaluation of the yield function  $f_{n+1}^{\text{elas}}$ . We compute the deviator  $\{\Delta e_n\} \rightarrow \text{De}$  of the strain increment  $\{\Delta \varepsilon_n\} \rightarrow \text{Deps}$ ,

```
v1=[1 1 1 0]'; % identity tensor
trDeps=Deps(1)+Deps(2); % trace of increment of strain
De=[Deps(1:2); 0; Deps(3)/2]-1/3*trDeps*v1;% Deviatoric tensor (4 comp.)
```

follow by the computation of the elastic prediction  $\{\sigma_{n+1}^{\text{elas}}\} \rightarrow \text{sigma\_elas}$  and its deviator  $\{s_{n+1}^{\text{elas}}\} \rightarrow \text{s\_elas}$  and equivalent stress  $\{\sigma_{n+1}^{\text{elas,eq}}\} \rightarrow \text{sigeq\_elas}$ . Next, the yield function  $f_{n+1}^{\text{elas}} \rightarrow \text{f\_elas}$  is evaluated.

```
sigma_elas=sigma+kappa*trDeps*v1+2*mu*De ; % elastic prediction of stresses
trsigma=sum(sigma_elas(1:3)); % volumetric part (stress)
s_elas=sigma_elas-1/3*trsigma*v1; % Deviatoric elastic prediction
sigeq_elas=sqrt(1.5*... % equivalent deviatoric stress
    (s_elas(1)^2+s_elas(2)^2+...
    s_elas(3)^2+2*s_elas(4)^2));
f_elas=sigeq_elas-h*p-sigma0;
```

Note that we choose here to not use the projection matrix  $[K]$  to extract the deviatoric part of  $\{\sigma_{n+1}^{\text{elas}}\}$  and  $\{\Delta \varepsilon_n\}$ .

Now, if  $\text{f\_elas} \leq 0$ , the elastic prediction represents the solution and the tangent moduli are equal to the elastic moduli ( $[A^{\text{EP}}] = [A]$ ). Otherwise, if  $\text{f\_elas} > 0$ , the plastic flow is evaluated by computing the increment of cumulative plastic strain ( $\Delta p_n \rightarrow \text{Dp}$ ) according to (9.30), followed by the update of the stress  $\{\sigma_{n+1}\} \rightarrow \text{sigma\_hat}$ . We then used (9.52) and (9.55) to compute coefficients  $\beta \rightarrow \text{beta}$  and  $\gamma \rightarrow \text{gamma}$ , and the local tangent plastic correction matrix  $[D] \rightarrow \text{D}$ . Lastly, selecting the components of the matrix  $[D] \in \mathbb{R}^{4 \times 4}$ , we obtain matrix representation  $[A^{\text{EP}}]$  of the tangent operator  $\mathcal{A}^{\text{EP}} = \mathcal{A} - \mathcal{D}$  for a plane strain analysis.

```
if(f_elas>0) % if plastic process
    Dp=f_elas/(3*mu+h); % incr. acc. plastic strain
    n_elas=s_elas/sigeq_elas;
    Depsp=3/2*Dp*n_elas; % incr. plastic strain
    sigma_hat=sigma_elas-2*mu*Deps; % new total stress

    beta=3*mu*Dp/sigeq_elas; % coefficients gamma and beta
    gamma=3*mu/(3*mu+h);
    D=3*mu*(gamma-beta)*n_elas*n_elas'+... % D matrix
        2*mu*beta*MK;
    AEP=A-[D(1:2,1:2) D(1:2,4); ... % A-D (select components)
        D(4,1:2) D(4,4)];
else % elseif elastic process
    Dp=0;
    sigma_hat=sigma_elas; % new total stress
    AEP=A;
end
```

**Exercise 9.6** (3D analysis). *Add the computation of the local tangent moduli for the case considered in exercise 9.2.*

**Exercise 9.7** (Von Mises and nonlinear isotropic hardening). *Add the computation of the local tangent moduli for the case considered in exercise 9.3.*

**Exercise 9.8** (Von Mises and kinematic hardening). *Add the computation of the local tangent moduli for the case considered in exercise 9.4.*

**Exercise 9.9** (Drucker-Prager model). *Add the computation of the local tangent moduli for the case considered in exercise 9.5*

**Exercise 9.10** (Newton algorithm for the example of Section 9.4). *Write a Newton algorithm for solving equilibrium equation (9.32). Prove that the iterative scheme introduced Section 9.4.1 is a modified Newton scheme with a constant search direction. Modify the code strip\_plast so as to use a Newton method. Compare the number of iterations of the two iterative procedures.*

## 9.6 Global equilibrium: discretization

A finite element approximation of  $\Omega$  and of the kinematic fields based on the principles developed in Chapter 3 is introduced into the linearized weak formulation (9.39) so as to solve the global equilibrium problem (9.34) by a Newton algorithm.

The initial estimate  $\Delta \mathbf{u}_{h,n}^{[0]}$  of the solution  $\Delta \mathbf{u}_{h,n}$  is chosen such that  $\Delta \mathbf{u}_{h,n}^{[0]} \in \mathcal{C}_h(\Delta \mathbf{u}_n^D)$  and the successive corrections  $\delta \mathbf{u}_{h,n}^{[k]}$  (9.38) are sought in  $\mathcal{C}_h(\mathbf{0})$ . By construction,  $\Delta \mathbf{u}_{h,n}^{[k]}$  will then be kinematically admissible, i.e.  $\Delta \mathbf{u}_{h,n}^{[k]} \in \mathcal{C}_h(\Delta \mathbf{u}_n^D)$ , for any  $k \geq 0$ . The choice of  $\Delta \mathbf{u}_{h,n}^{[0]}$ , which is clearly crucial for the performance of the overall procedure, will be discussed in section 9.7. Each correction  $\delta \mathbf{u}_{h,n}^{[k]}$  is governed by the linearized weak formulation (9.39):

$$\begin{aligned} \text{Find } \delta \mathbf{u}_{h,n}^{[k]} \in \mathcal{C}_h(\mathbf{0}), \quad (9.56) \\ \int_{\Omega} \boldsymbol{\varepsilon}[\delta \mathbf{u}_{h,n}^{[k]}] : \mathcal{A}^{\text{EP}}(\Delta \boldsymbol{\varepsilon}_n^{[k]}; \mathcal{S}_n) : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = \\ - \int_{\Omega} \boldsymbol{\sigma}_{n+1}^{[k]} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV + \int_{\Omega} \rho \mathbf{f}_{n+1} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}_{n+1}^D \cdot \mathbf{w} \, dS \quad \forall \mathbf{w} \in \mathcal{C}_h(\mathbf{0}) \end{aligned}$$

Using notation as in Chapter 3, the finite element approximation of the left hand side of (9.56) can be written

$$\int_{\Omega} \boldsymbol{\varepsilon}[\delta \mathbf{u}_{h,n}^{[k]}] : \mathcal{A}^{\text{EP}}(\Delta \boldsymbol{\varepsilon}_n^{[k]}; \mathcal{S}_n) : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV = \{\mathbb{W}\}^T [\mathbb{K}_{n+1}^{[k]}] \{\delta \mathbb{U}_n^{[k]}\}, \quad (9.57)$$

where  $\{\delta \mathbb{U}_n^{[k]}\}$  and  $\{\mathbb{W}\}$  are the global nodal vectors associated with the virtual fields  $\delta \mathbf{u}_{h,n}^{[k]} \in \mathcal{C}_h(\mathbf{0})$  and  $\mathbf{w} \in \mathcal{C}_h(\mathbf{0})$ , respectively. The matrix  $[\mathbb{K}_{n+1}^{[k]}]$  is known as the *global elastoplastic tangent stiffness matrix*. The right-hand side, corresponding to the residual  $-\mathcal{R}(\Delta \mathbf{u}_{h,n}^{[k]}; \mathbf{w}, \mathcal{S}_n)$ , yields

$$- \int_{\Omega} \boldsymbol{\sigma}_{n+1}^{[k]} : \boldsymbol{\varepsilon}[\mathbf{w}] \, dV + \int_{\Omega} \rho \mathbf{f}_{n+1} \cdot \mathbf{w} \, dV + \int_{S_T} \mathbf{T}_{n+1}^D \cdot \mathbf{w} \, dS = -\{\mathbb{W}\}^T \{\mathbb{R}_{n+1}^{[k]}\}$$

The residual  $\{\mathbb{R}_{n+1}^{[k]}\}$  can also be conveniently split into two parts:

$$-\{\mathbb{R}_{n+1}^{[k]}\} = \{\mathbb{F}_{n+1}^{\text{int}[k]}\} + \{\mathbb{F}_{n+1}^{\text{ext}}\}, \quad (9.58)$$

where  $\{\mathbb{F}_{n+1}^{\text{ext}}\} = \{\mathbb{F}_{n+1}^{\text{vol}}\} + \{\mathbb{F}_{n+1}^{\text{surf}}\}$  is the vector of external nodal forces arising from given body force densities  $\mathbf{f}_{n+1}$  and traction  $\mathbf{T}_{n+1}^{\text{D}}$  (see Sec. 3.2.3 for the definition of generalized nodal forces), while  $\{\mathbb{F}_{n+1}^{\text{int}[k]}\}$  is the vector of internal nodal forces associated with the stress field  $\sigma_{n+1}^{[k]}$ , defined by

$$-\int_{\Omega} \sigma_{n+1}^{[k]} : \varepsilon[\mathbf{w}] \, dV = \{\mathbb{W}\}^T \{\mathbb{F}_{n+1}^{\text{int}[k]}\}. \quad (9.59)$$

Eventually, the finite element approximation of the linearized weak formulation (9.56) can be written

$$[\mathbb{K}_{n+1}^{[k]}] \{\delta \mathbb{U}_n^{[k]}\} = -\{\mathbb{R}_{n+1}^{[k]}\}. \quad (9.60)$$

**Comments.** The global elastoplastic tangent matrix  $[\mathbb{K}_{n+1}^{[k]}]$  defined by (9.57) reduces to the elastic stiffness matrix in the absence of any plastic strain. It is also known under the name of *consistent tangent matrix*, "consistent" referring to the fact that it is the correct tangent matrix associated with the Newton method (and therefore consistent with this method).

### 9.6.1 Element tangent matrix and internal forces

Solving problem (9.60) requires the evaluation of the global tangent operator (9.57) and the global nodal vector of internal forces (9.59). All these quantities are similar in structure to their linear elastic counterparts, and can therefore be set up by assembling element contributions as shown in Chapter 3.

The element contributions to (i) the global nodal vector of internal forces (9.59) and (ii) the global tangent matrix (9.57) are defined by

$$\begin{aligned} \int_{E_e} \sigma_{n+1}^{[k]} : \varepsilon[\mathbf{w}] \, dV &= \{W_e\}^T \{F_{n+1,e}^{\text{int}[k]}\}, \\ \int_{E_e} \varepsilon[\delta \mathbf{u}_{h,n}^{[k]}] : \mathcal{A}^{\text{EP}}(\Delta \varepsilon_n^{[k]}, \mathcal{S}_n) : \varepsilon[\mathbf{w}] \, dV &= \{W_e\}^T [K_{n+1,e}^{[k]}] \{\delta U_{n,e}^{[k]}\}. \end{aligned}$$

or, using the engineering notation (3.12) and a quadrature rule (3.20) for their evaluation,

$$\begin{aligned} \{F_{n+1,e}^{\text{int}[k]}\} &= - \int_{\Delta_e} [B(\mathbf{a})]^T \{\sigma_{n+1}^{[k]}(\mathbf{a})\} J(\mathbf{a}) \, dV(\mathbf{a}), \\ &\approx - \sum_{g=1}^G w_g [B(\mathbf{a}_g)]^T \{\sigma_{n+1}^{[k]}(\mathbf{a}_g)\} J(\mathbf{a}_g), \end{aligned} \quad (9.61)$$

$$\begin{aligned} [K_{n+1,e}^{[k]}] &= \int_{\Delta_e} [B(\mathbf{a})]^T [A^{\text{EP},[k]}] [B(\mathbf{a})] J(\mathbf{a}) \, dV(\mathbf{a}), \\ &\approx \sum_{g=1}^G w_g [B(\mathbf{a}_g)]^T [A^{\text{EP},[k]}] [B(\mathbf{a}_g)] J(\mathbf{a}_g). \end{aligned} \quad (9.62)$$

It should be stressed that  $[A^{\text{EP},[k]}]$  and  $\{\sigma_{n+1}^{[k]}\}$  are space-dependent, through the nodal values of the displacement increment  $\{\Delta U_{n,e}^{[k]}\}$  and the mechanical state  $S_n$  at time  $t_n$  (in particular the stress  $\{\sigma_n\}$  and internal variable  $p_n$ ).

It is worth noting that this procedure is a simple transposition of the one introduced in Section 3.2.5 for the computation of element contributions to the stiffness matrix in linear elasticity (see eq. (3.21)). Similarly, the element tangent matrix stemming from this computation will contribute, through assemblage, to the global tangent matrix.

### 9.6.2 Element tangent matrix and internal forces in plane strain analyses

In plane-strain analysis, we always have  $\sigma_{33}\epsilon_{33} = 0$ . The component associated with  $\sigma_{33}$  then does not contribute to the element tangent matrix. However, as already mentioned for the implementation of the radial return algorithm in plane-strain analyses (Sections 9.3.5 and 9.5.4), the integration of the plastic part of the constitutive relations requires the full stress tensor. We quickly describe the function `T6_2A_solid_KTePlast` that provides the element quantities for plane-strain analyses with the T6 element (Sec. 2.2.9). This function has to be compared with the function `T6_2A_solid_Ke` (Sec. 3.2.6) returning the element elastic stiffness matrix.

In addition to the matrix  $X$  with the nodal coordinates and the list `mate` with materials parameters (Young modulus, Poisson ratio, hardening coefficient, initial yield limit ...), input variables contain the nodal values of the displacement fields  $\{\Delta U_e\} \rightarrow \text{DUE}$  and the four non-zero components of the stress tensor  $\{\sigma_n\} \rightarrow \text{sigma}$  and cumulative plastic strain  $p_n \rightarrow p$  at the element Gauss points.

```
function [KEPe,Finte,sigma_hat, Dp]=...
        T6_2A_solid_KTePlast(T,mate,DUE,sigma,p)

E=mate(1);
nu=mate(2);
A=E/((1+nu)*(1-2*nu))*[1-nu,nu,0;      % Plane-strain linear
                        nu,1-nu,0;      % isotropic elastic matrix
                        0,0,(1-2*nu)/2];
```

Output variables are the element tangent matrix ( $[K_{n+1,e}^{[k]}] \rightarrow \text{KEPe}$ ), the element nodal internal forces ( $\{F_{n+1,e}^{\text{int}[k]}\} \rightarrow \text{Finte}$ ) and the new stresses  $\{\sigma_{n+1}^{[k]}\} \rightarrow \text{sigma\_hat}$  cumulative strain increments  $\Delta p_n^{[k]} \rightarrow \text{Dp}$  at the element Gauss points.

Only a few additions with respect to the "elastic" version of the function are then required: (i) the computation according to (3.12) of the strain components  $\text{Deps} = \text{Be} * \text{DUE}$  associated with  $\text{DUE}$ , (ii) the run of the radial return algorithm `RRTM_VonMises_2A_R` introduced in Section 9.5.4 so as to compute the local tangent modulus  $\text{AEP}$ , the new stresses and the increment of cumulative plastic strain at each Gauss point, and (iii) the computation of element internal forces (resp. tangent matrix) defined by (9.61) (resp. (9.62)).

```
a_gauss=1/6*[4 1 1; 1 4 1; 1 1 4]; % Gauss abscissae
w_gauss=[1/6 1/6 1/6]; % Gauss weights
KEPe=zeros(12,12); %
Finte=zeros(12,1); %
for g=1:3, % loop over Gauss points
```

```

... Compute Be ... (see function T6_2A_solid_Ke)

Deps=Be*DUE; % Increment of strain
[AEP,Dp(g,:),sigma_n(g,:)] = ...
[AEP,Dp,sigma_n(g,:)] = ... % radial return algorithm
    RRTM_VonMises_2A_R...
    (mate,sigma(g,1:4)',p(g),Deps);
KEPe = KEPe+Be'*AEP*Be*detJ*w_gauss(g); % tangent matrix
Finte = Finte - ... % internal nodal forces
    Be'*sigma_n([1:2 4])*detJ*w_gauss(g);
end

```

## 9.7 Algorithmic issues

The full incremental-iterative algorithm is summarized in Box 9.3. It corresponds to the "exact" Newton method with consistent tangent operator.

### 9.7.1 Initialisation of each increment

An important issue concerns the choice of the initialisation  $\Delta \mathbf{u}_{h,n}^{[0]} \in \mathcal{C}_h(\Delta \mathbf{u}_n^D)$ . A "bad" choice can affect drastically the convergence of the algorithm. For instance, in case of non-zero prescribed displacements, setting  $\Delta \mathbf{u}_{h,n}^{[0]} = \Delta \mathbf{u}_{h,n}^{(D)}$  (defined by (2.32)) turns out to be a bad initialisation, since it only affects the nodal values on  $S_u$ . Indeed the radial return algorithm applied to elements with nodes on  $S_u$  would predict totally un-physical plastic strain increments and this would consequently slow down the overall convergence. A "good" choice depends on the problem and in particular on the loading history.

**Handling prescribed displacements: the general case.** Prescribed displacements can be handled according to the following strategy. The initialisation of the displacement increment  $\Delta \mathbf{u}_{h,n}$  is expressed as:

$$\Delta \mathbf{u}_{h,n}^{[0]} = \Delta \mathbf{u}_{h,n}^{(0)} + \Delta \mathbf{u}_{h,n}^{(D)} \quad (\Delta \mathbf{u}_{h,n}^{(0)} \in \mathcal{C}_h(\mathbf{0}), \Delta \mathbf{u}_{h,n}^{(D)} \in \mathcal{C}_h(\Delta \mathbf{u}_n^D))$$

where  $\Delta \mathbf{u}_{h,n}^{(D)}$  is a given arbitrary extension to  $\Omega$  of the prescribed displacement  $\Delta \mathbf{u}_n^D$  on  $S_u$ . The unknown  $\Delta \mathbf{u}_{h,n}^{(0)}$  is obtained by solving:

$$\begin{aligned}
 &\text{Find } \Delta \mathbf{u}_{h,n}^{(0)} \in \mathcal{C}_h(\mathbf{0}), \\
 &\langle \mathcal{R}'(\mathbf{0}; \mathbf{w}, \mathcal{S}_n), \Delta \mathbf{u}_{h,n}^{(0)} \rangle = -\mathcal{R}(\mathbf{0}; \mathbf{w}, \mathcal{S}_n) \\
 &\quad - \langle \mathcal{R}'(\mathbf{0}; \mathbf{w}, \mathcal{S}_n), \Delta \mathbf{u}_{h,n}^{(D)} \rangle, \quad \forall \mathbf{w} \in \mathcal{C}_h(\mathbf{0}),
 \end{aligned}$$

where the residual  $\mathcal{R}(\mathbf{0}; \mathbf{w}, \mathcal{S}_n)$  is expressed in terms of the stress  $\boldsymbol{\sigma}_n$  associated with the initial mechanical state (i.e. the final stress field of the previous

increment) and the tangent operator  $\mathcal{R}'(\mathbf{0}; \mathbf{w}, \mathcal{S}_n)$  coincides with the final tangent operator of the previous increment:

$$\mathcal{R}'(\mathbf{0}; \mathbf{w}, \mathcal{S}_n) = \mathcal{R}'(\Delta \mathbf{u}_{h,n-1}; \mathbf{w}, \mathcal{S}_{n-1}).$$

This initialization strategy must be implemented with care, as some quantities resulting from the previous increment (in particular the converged value of the consistent tangent operator) need to be retained and used.

**Handling prescribed displacements: the case of proportional loading.** A frequently-used subclass of elastoplastic analyses rest on the assumption of proportional loading, whereby all prescribed loads are proportional to given time-independent distributions and modulated by a scalar load factor  $\lambda(t)$ :

$$\mathbf{f}(\mathbf{x}, t) = \lambda(t)\mathbf{f}(\mathbf{x}), \quad \mathbf{T}^D(\mathbf{x}, t) = \lambda(t)\mathbf{T}^D(\mathbf{x}), \quad \mathbf{u}^D(\mathbf{x}, t) = \lambda(t)\mathbf{u}^D(\mathbf{x}). \quad (9.63)$$

Load increments are then proportional to increments  $\Delta \lambda_n := \lambda(t_{n+1}) - \lambda(t_n)$  of the load factor.

Under this assumption, the initialization  $\Delta \mathbf{u}_{h,0}^{[0]}$  for the first load increment is chosen as the purely elastic response to loads corresponding to  $\lambda_1 := \lambda(t_1)$ . Then, for all subsequent load increments, the convergence speed is significantly enhanced by setting

$$\Delta \mathbf{u}_{h,n}^{[0]} = \frac{\Delta \lambda_n}{\Delta \lambda_{n-1}} \Delta \mathbf{u}_{h,n-1} \quad (9.64)$$

which amounts to assuming that the new displacement increment is proportional to the solution for the previous load increment, modulated by the relative increment of load factor.

Proportional loading thus significantly simplifies the handling of prescribed displacements in elastoplastic analyses. It is assumed in the code *plasticity*, presented thereafter in Section 9.8.

### 9.7.2 Simplified versions of the iterative algorithm

The consistent Newton method (using the consistent tangent matrix at each iteration) has quadratic convergence in a neighbourhood of the solution  $\{\Delta \mathbf{U}_n\}$ . On the other hand, it requires the computation of the consistent tangent matrix  $[\mathbb{K}_{n+1}^{[k]}]$  at each iteration, which can be computationally expensive. In terms of iteration count, fast convergence is often obtained.

For reasons of simplicity of implementation and of computational cost of each iteration, a simplified version is sometimes desirable, such as the modified Newton method with constant search direction presented in Section 8.2.1 for a scalar equation. It is based on replacing the consistent tangent matrix  $[\mathbb{K}_{n+1}^{[k]}]$  by a constant "tangent" matrix  $[\hat{\mathbb{K}}]$ . We usually define  $[\hat{\mathbb{K}}]$  as a stiffness matrix constructed on the basis of a choice of local modulus  $\hat{\mathcal{A}}$  (for instance, choosing  $\hat{\mathcal{A}} = \mathcal{A}$  gives the elastic stiffness matrix  $[\hat{\mathbb{K}}] = [\mathbb{K}]$ ). The main advantage of

**Box 9.3:** Incremental and iterative algorithm with the consistent tangent operator.

**Input:** • mesh, temporal discretization  $t_0, t_1, \dots, t_M$   
• materials parameters, loading, tolerance  $\epsilon$ .

1. – **Initialization** at  $t = 0$  :

(i) Initialization (natural state)

$$\{\mathbb{U}_0\} = \{0\}$$

$\sigma_0 = \mathbf{0}$ ,  $p_0 = 0$  at Gauss points.

(ii) Compute elastic solution and reference residual  $r^{\text{ref}} = \|\{\mathbb{F}^{\text{u}}\} + \{\mathbb{F}^{\text{ext}}\}\|$

2. – **Step-by-step solution:** for  $n = 0, 1, 2, \dots, M$ ,

(i) **Newton initialization:**

(a) External nodal forces at  $t_{n+1}$ :  $\{\mathbb{F}_{n+1}^{\text{ext}}\} = \{\mathbb{F}_{n+1}^{\text{vol}}\} + \{\mathbb{F}_{n+1}^{\text{surf}}\}$ ;

(b) Initialisation of displacement increment  $\{\Delta \mathbb{U}_n^{[0]}\} \in \mathcal{C}_h(\Delta \mathbf{u}_n^{\text{D}})$  using elastic solution and/or other strategies (see section 9.7);

(ii) **Newton solver:** while  $r > \epsilon r^{\text{ref}}$  iterate ( $k = 0, 1, 2, \dots$ );

(a) Assemblage:

Internal nodal forces  $\{\mathbb{F}_{n+1}^{\text{int}[k]}\}$ ;

Elastoplastic tangent matrix  $[\mathbb{K}_{n+1}^{[k]}]$ ;

(b) Compute residual:  $\{\mathbb{R}_{n+1}^{[k]}\} := -\{\mathbb{F}_{n+1}^{\text{int}[k]}\} - \{\mathbb{F}_{n+1}^{\text{ext}}\}$ ,

(c) Solve  $[\mathbb{K}_{n+1}^{[k]}\{\delta \mathbb{U}_n^{[k]}\} = -\{\mathbb{R}_{n+1}^{[k]}\}$ ;

(d) Update displacement increment:  $\{\Delta \mathbb{U}_n^{[k+1]}\} := \{\Delta \mathbb{U}_n^{[k]}\} + \{\delta \mathbb{U}_n^{[k]}\}$ ;

(e) Compute norm of current residual:  $r = \|\{\mathbb{R}_{n+1}^{[k]}\}\|$

(iii) **Update solution** at  $t = t_{n+1}$  (displacement, stress, internal variables):

$$\{\mathbb{U}_{n+1}\} = \{\mathbb{U}_n\} + \{\Delta \mathbb{U}_n\},$$

$$\sigma_{n+1} = \Sigma(\Delta \mathbf{u}_n), \quad p_{n+1} = p_n + \Delta p_n, \text{ at Gauss points}$$

such a strategy is that each iteration performs faster. Its main drawback is that the quadratic convergence of the "exact" Newton method is lost (see Section 8.2.1). Intermediate strategies are possible. One may recompute a new operator  $[\hat{\mathbb{K}}]$  for each time increment (e.g.  $[\hat{\mathbb{K}}]$  can be chosen as the consistent tangent operator  $[\mathbb{K}_1^{[0]}]$  evaluated at the first iteration of the modified Newton method). Three strategies ("exact" Newton method, modified Newton method with (i)  $[\hat{\mathbb{K}}] = [\mathbb{K}]$  or (ii)  $[\hat{\mathbb{K}}] = [\mathbb{K}_{n+1}^{[0]}]$ ) are demonstrated in Section 9.9).

### 9.7.3 Difficulties associated with plastic incompressibility

In the classical plastic models such as the one considered in this chapter, the plastic part of the strain is incompressible (Section 9.1.5). This incompressibility results from the fact that plastic strains in metals result from slips in the crystal lattice, and hence without change in volume. When plastic strain increases during a loading path, the total strain may become progressively dominated by its plastic part, and thus close to a state of incompressible strain. Materials characterized by low hardening are especially prone to such situations, since the limiting case of perfect plasticity allows the strain to be arbitrarily large. The numerical treatment of incompressibility has been seen in Chapter 6 to introduce difficulties. The computational difficulties arising from plastic incompressibility have been initially pointed out by Rice Nagtegaal et al. (1974). Suitable finite element techniques have since been proposed, such as the so-called  $\bar{B}$  method introduced e.g. in the book by Simo and Hughes (1998) which is based on a redefinition of the matrix  $[B]$  which connects strain and nodal displacements, see expression (3.12).

## 9.8 The code plasticity for plane-strain elastoplastic analysis

The concepts developed in this chapter are implemented in the code `plasticity` dedicated to the analysis of elastoplastic structures in plane-strain conditions. This code uses the same structures as the linear code `genlin`. The implemented constitutive model assumes the von Mises criterion with linear isotropic hardening, the yield surface thus being defined by  $f(\sigma, p) = \sigma^{\text{eq}} - \sigma_0 - hp$ , together with an isotropic linear elastic behavior. The loading is assumed to be proportional, i.e. of the form (9.63).

### 9.8.1 Input files

Three example problems are provided in the distribution: a notched specimen loaded in tension (`Chap9_wedge.m`), a plate with a circular hole (`Chap9_hole.m`), and the example of Section 9.4 (`Chap9_strip.m`).

The input file is very similar to that used for a linear elastic analysis (see details in Appendix B). More precisely, the only additions (relative to the elastic case) are:

- the value of the plastic material parameters in table `material`,  
`material = [1 .3 0.88 0.];`                      % [Young, Poisson, Sigma0, H, ...]

Note that new material parameters, if needed, can be added at the end of the line. Every line of the matrix defines a material (in this case only one material is given)

- the list of the value of the "history function"  $\lambda(t)$  at discrete time in table `LambdaT`,  
`LambdaT = [0 0.5 1. 2.];`                      % [T\_0=0, T\_1, T\_2, ..., T\_n]

As we choose to systematically start the analysis from a natural state at time  $t_0 = 0$ , the list `LambdaT` must start with zero.

### 9.8.2 Initialisation phase

The first part of the code (up to the "Initial state" line) is devoted to (i) reading the input data; (ii) estimating the number of non-zero coefficients in the matrix; (iii)



assembling (a) the elastic stiffness matrix, (b) the nodal force vector  $F_{Du}$  associated with prescribed displacement for  $\lambda = 1$ , and (c) the nodal force vector  $F_{ext}$  associated with prescribed tractions for  $\lambda = 1$  (it involves the same operations as for the linear elastic case); (iv) computing the elastic solution  $U$  corresponding to  $\lambda = 1$  and the reference residual

```
R=-Fext-FuD; % right hand side
U=-KEP\R; % elastic solution for lambda=1
residref=sqrt(R'*R); % reference residual
```

Reading this set-up phase, and comparing the operations to the linear elastic case, is left as an exercise.

All variables are first initialized, assuming a natural state. The displacement increment field  $nodes(n).DU$  is created and initialized to  $LambdaT(2)*nodes(n).U$  which contains the imposed displacements at time  $t_1$ . Next, the entries of  $nodes(n).DU$  with unknown nodal values are initialised to the elastic prediction  $LambdaT(2)*U$ .

```
for n=1:analysis.NN % loop on nodes
    nodes(n).DU(1:ndof)=LambdaT(2)* ...
        nodes(n).U(1:ndof); % saves prescribed displacement
    for dir=1:ndof
        dof=nodes(n).dof(dir);
        if dof>0
            nodes(n).DU(dir)=LambdaT(2)*U(dof); % elastic solution
        end
    end
    nodes(n).U(1:ndof)=zeros(1:ndof); % resets to zero total displ
end
```

At the element level, an initialization of the stresses  $elements(e).Sg$  and cumulative plastic strains  $elements(e).pg$  is operated for all  $ng$  Gauss points. Intermediate variables (output variables of the radial return algorithm, namely the cumulative plastic strain increment and final stresses) are also set to zero. Lastly, the global vector of internal forces  $\{\mathbb{F}_{n+1}^{int[k]}\} \rightarrow F_{int}$  is initialized to zero.

```
for e=1:analysis.NE % Loop on element
    ng = elements(e).ng; % number of Gauss point
    elements(e).pg = zeros(ng,1); % eq plastic strain (Gauss pts)
    elements(e).Sg = zeros(ng,Sdim); % stresses (Gauss pts)
    elements(e).Dpg= zeros(ng,1); % Inc. c. plastic strain (Gauss pts)
    elements(e).Sghat= zeros(ng,Sdim); % new stresses (Gauss pts)
end
Fint=zeros(analysis.neq,1); % zero internal forces
```

### 9.8.3 The incremental-iterative algorithm

The loop on loading steps can begin.

```
numstep=length(LambdaT)-1; % number of steps
for nstep=0:numstep-2, % loop over all load steps
    iter=0;
    toll=1.d-4;
    resid=1;
```

For every loading step with  $n \rightarrow \text{nstep} > 0$ , the displacement increment is initialized according to (9.64) which automatically satisfies the requirement that  $\Delta \mathbf{u}_n^{[0]} \in \mathcal{C}_h(\Delta \mathbf{u}_n^D)$  and improves a purely elastic initialisation.

```

if nstep>0
    for n=1:analysis.NN                % initialize DU
        nodes(n).DU(:)=nodes(n).DU(:)* ...
            (LambdaT(nstep+2)-LambdaT(nstep+1))/(LambdaT(nstep+1)-LambdaT(nstep));
    end
end

```

All variables being initialized, Newton iterations can start for the current loading step.

```

while resid > toll*residref,
    iter=iter+1;
    KEP(:,:)=0.d0;                    % sets Tangent matrix to zero
    Fint(:)=0;                        % sets Fint to zero

    .... assemblage of KEP and Fint ...

    R=-Fext*LambdaT(nstep+2)-Fint;     % residual vector
    resid=sqrt(R'*R);
    DDu=-KEP\R;                       % global prediction with KEP

    for n=1:analysis.NN                % updates displacement increment
        for dir=1:ndof
            dof=nodes(n).dof(dir);
            if dof>0
                nodes(n).DU(dir)=nodes(n).DU(dir)+DDu(dof);
            end
        end
    end
end
% end of Newton's iterations

```

DDu is exactly the MATLAB translation of the vector  $\{\delta \mathbf{U}_n^{[k]}\}$  in the algorithm of Box 9.3. It should be remarked that the increment  $\Delta \mathbf{u}_{h,n}^{[k]} \in \mathcal{C}_h(\Delta \mathbf{u}_n^D)$  is stored at the nodal level in nodes.DU. The assemblage procedure used is based on the data structure (in MATLAB sense) adopted in the code genlin.

```

for e=1:analysis.NE,                  % assemblage of KEP and Fint

    ... standard procedures ...

    DDe=zeros(Dne,1);
    pos=1;
    for n=1:ne
        node=elements(e).nodes(n);
        Xe(n,:)=nodes(node).coor;
        Ge(pos:pos+ndof-1)=nodes(node).dof;
        DDe(pos:pos+ndof-1)=nodes(node).DU;
        pos=pos+ndof;
    end
end

```

```

Sg=elements(e).Sg; % stresses (Gauss pts)
pg=elements(e).pg; % cumulative pl. strain (Gauss pts)
[KEPe,Finte,elements(e).Sghat, ... % element routine
 elements(e).Dpg]=eval([Etag Atag 'KTePlast'...
 '(Xe, material(mat,:),DUe,Sg,pg)']);
Le0=find(Ge>0); % local numbering of unknowns
Ie=Ge(Le0); % global numbering
Fint(Ie)=Fint(Ie)+Finte(Le0); % internal force vector assemblage
KEP(Ie,Ie)=KEP(Ie,Ie)+KEPe(Le0,Le0); % matrix assemblage
end

```

The assemblage is a simple extension of the code lines introduced in Section 3.2.2 where the call to the function computing the element contribution to the stiffness matrix is replaced by the call to the function computing element contribution to the consistent tangent matrix and internal forces. The latter is, for example, the function `T6_2A_solid_KTePlast` (resp. `T3_2A_solid_KTePlast`) for a plane-strain analysis with T6 elements (resp. T3 elements). These lines perform the assemblage of the consistent tangent matrix `KEP` and the internal force vector `Fint`.

Once converged, variables are updated.

```

for n=1:analysis.NN
    nodes(n).U=nodes(n).U+nodes(n).DU; % updates displacement
end
for e=1:analysis.NE
    elements(e).pg=elements(e).pg... % updates cumulative
        + elements(e).Dpg; % plastic strain (Gauss pts)
    elements(e).Sg=elements(e).Sghat; % updates stress (Gauss pts)
end

```

As in the elastic case, the analysis is followed by a post-processing phase in which the stresses and the equivalent plastic strains at Gauss points are extrapolated to the element nodes before starting a new increment of the loading sequence.

```

DOUT=Sdim+1;
Outn=zeros(analysis.NN,DOUT); % initializes output matrix
counter=zeros(analysis.NN,1); % initializes counter list
for e=1:analysis.NE, % loop over elements
    type=elements(e).type;
    Etag=Le(type).tag;
    connec=elements(e).nodes; % gets element connectivity
    ng=Le(type).ng; % number of Gauss pts in element
    Outg(1:ng,1:Sdim)=elements(e).Sg; % stress components
    Outg(1:ng,Sdim+1)=elements(e).pg(1:ng); % accumulated plastic strain
    Outne=eval([Etag 'g2n(Outg)']); % extrapolates at nodes
    Outn(connec,:)=Outn(connec,:)+Outne; % adds to connec nodes
    counter(connec)=counter(connec)+1;
end
for icomp=1:DOUT
    Outn(:,icomp)=Outn(:,icomp)./counter; % naive average of stresses
end

```

### 9.8.4 Exercises

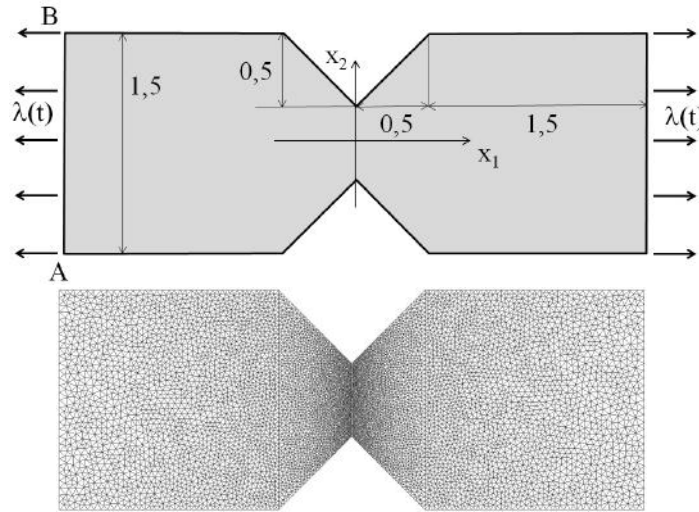
**Exercise 9.11.** Run plasticity with the input file Chap9.strip. This analysis is equivalent to the 1D procedure strip\_plast described in Section 9.4.1. Compare the results of the two codes with the analytical solution available in strip\_plast.

**Exercise 9.12.** Modify plasticity so as to use the constant tangent operator.

## 9.9 Example: illustration of algorithms on a notched specimen

This section aims at illustrating, on a simple example in plane-strain, the typical behavior of the Newton method with a consistent tangent operator or with constant search direction.

**Description of the notched specimen.** We consider the notched specimen under tensile loading sketched in Figure 9.10a. The mesh used and shown in Figure 9.10 has 17,150 linear (T3) elements and 8,792 nodes. The large number of elements used is justified by the stress concentration<sup>3</sup> expected in the vicinity of the notches, resulting in steep stress and strain gradients. The numerical results for this example were obtained using the plasticity code described in Section 9.8 with consistent tangent stiffness or its variants using constant "tangent" matrices.



**Figure 9.10:** Notched specimen loaded in tension: geometry and loading (top), finite element mesh (bottom).

The constitutive material is assumed elastoplastic with von Mises relations yield function (9.12c-d) with linear isotropic hardening,

$$R(p) = \sigma_0 + hp.$$

<sup>3</sup>Linear elasticity predicts a stress *singularity* in any corner between two free edges with an internal angle.

Materials parameters used are  $\nu = 0,3$ ,  $\sigma_0 = 0,88E$ . Two cases of hardening are considered:  $h = 0$  (no hardening, that is to say, perfect plasticity) or  $h = 0,05E$ . With such material parameters and because the loading is here prescribed through tractions (see below), stresses do not depend on Young modulus  $E$  and strains are inversely proportional to  $E$ . The value of  $E$  is here set to unity for convenience.

The specimen is loaded in simple tension in the horizontal direction: uniform tensile tractions  $\pm\lambda(t)e_1$  are applied to the left- and rightmost edges as shown in Figure 9.10a. The load factor  $\lambda(t)$  is slowly increasing from zero. We also prescribed the three constraints

$$u_1(A) = u_2(A) = u_1(B) = 0,$$

so as to prevent any (infinitesimal) rigid-body motion. With the value of  $\sigma_0/E$  adopted here, a pure elastic computation (done for example with  $\lambda = 1$ ) shows that the plastic flow starts at  $\lambda = \lambda_0 \approx 0.0802^4$ .

**Case of perfectly plastic material.** We consider  $M = 10$  loading steps. The loading history is defined by the sequence of values  $\lambda(n) = \lambda(t_n)$  of the loading parameter  $\lambda$  at time  $t_n$ :

$n$	0	1	2	3	4	5	6	7	8	9	10
$\lambda$	0	0,08	0,16	0,24	0,32	0,40	0,46	0,52	0,57	0,59	0,6

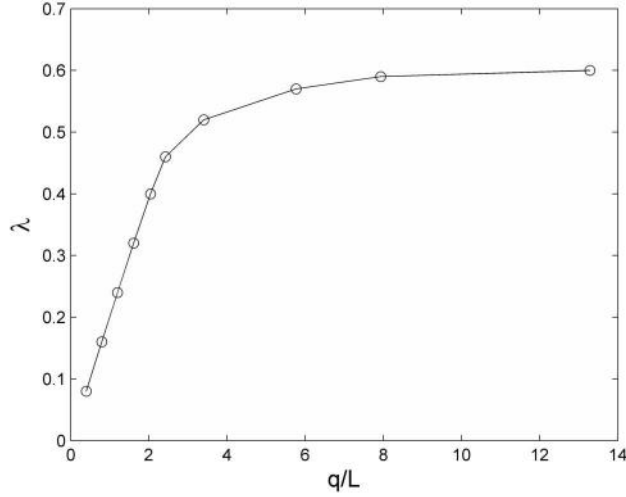
Table 9.1 shows a comparison of the number of iterations required to converge for each time step. The convergence criterion is  $\|\{\mathbb{K}_{n+1}^{[k]}\}\| < \epsilon \|\{\mathbb{K}_{n+1}^{[0]}\}\|$  with a relative tolerance  $\epsilon = 10^{-4}$ .

$n$	1	2	3	4	5	6	7	8	9	10
(a) $[\mathbb{K}_{n+1}^{[k]}]$	1	4	3	3	4	5	5	6	5	5
(b) $[\hat{\mathbb{K}}] = [\mathbb{K}_{n+1}^{[1]}]$	1	14	20	26	23	50	117	146	40	210
(c) $[\hat{\mathbb{K}}] = [\mathbb{K}]$	1	14	28	51	78	127	304	643	981	4972

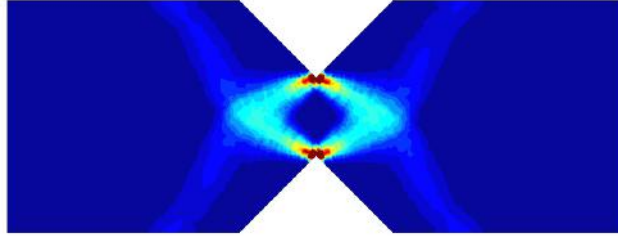
**Table 9.1:** Notched specimen loaded in tension, perfectly plastic material ( $h = 0$ ): iteration count for each load step. Newton method (a) with consistent tangent matrix, (b) modified using the elastoplastic tangent matrix calculated at the first iteration for each time step, and (c) modified using the elastic stiffness matrix.

We note in particular that the use of the elastic stiffness matrix as a search direction in the modified Newton method leads to prohibitive iteration counts for

<sup>4</sup>For this type of configuration with a singularity of the elastic stress, plasticity should rigorously appear at the notch roots for any value  $\lambda \neq 0$ . In fact, with the type of finite element used here (T3 element), the stresses are constant on each element, and  $\lambda_0$  depends on the mesh. . . To put it another way, the plastic zone cannot be "seen" as long as it is much smaller than an element size.



**Figure 9.11:** Notched specimen loaded in tension, perfectly plastic material ( $h = 0$ ): tensile force as a function of elongation (in dimensionless variables).



**Figure 9.12:** Notched specimen loaded in tension, perfectly plastic material ( $h = 0$ ): cumulative plastic strain at final step ( $n = 10$ ).

the last loading steps. This is due to the assumption of perfect plasticity and the fact that the final loading is close to the limit load (the force-elongation curve becoming close to horizontal). The convergence of the "consistent" Newton method (based on the tangent operator  $[\mathbb{K}_{n+1}^{[k]}]$ ) requires very few iterations, even for the last loading steps, but each iteration is more expensive in CPU time due to the computation of  $[\mathbb{K}_{n+1}^{[k]}]$  matrix at each iteration.

Figure 9.11 shows the relation between the tensile traction  $\lambda$  and the longitudinal extension  $q$ . The final value of loading chosen here appears to be near the limit load of the structure. Figure 9.12 shows the distribution of the cumulative plastic strain at the final loading.

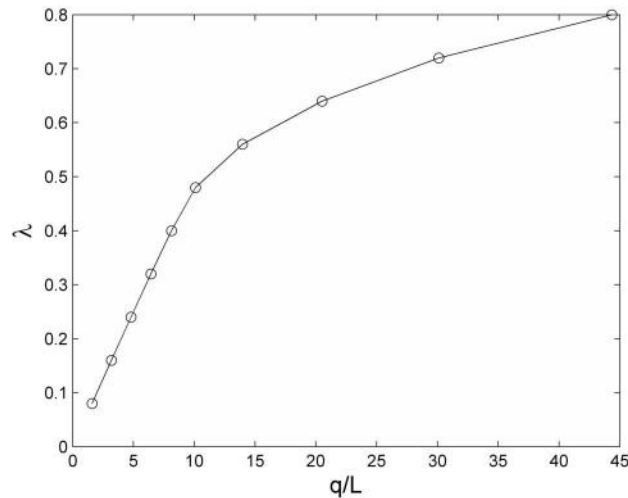
**Case of plastic material with hardening.** We consider again  $M = 10$  loading steps. The loading history is now defined by

$n$	0	1	2	3	4	5	6	7	8	9	10
$\lambda$	0	0,08	0,16	0,24	0,32	0,40	0,48	0,56	0,64	0,72	0,8

Table 9.2 shows a comparison of the iteration counts required to converge for each loading step. Again, the convergence criterion is  $\|\{\mathbb{K}_{n+1}^{[k]}\}\| < \epsilon \|\{\mathbb{K}_{n+1}^{[0]}\}\|$  with  $\epsilon = 10^{-4}$ . The numbers of iterations required by the various methods are smaller than in the case of perfect plasticity, particularly when using the modified Newton method with constant search direction (cases (b) and (c) in Table 9.2). Again, the "exact" Newton method converges within very few iterations. Figure 9.13 shows the tensile traction  $\lambda$  against the longitudinal extension  $q$ .

$n$	1	2	3	4	5	6	7	8	9	10
(a) $[\mathbb{K}_{n+1}^{[k]}]$	1	4	3	3	3	4	6	5	5	4
(b) $[\hat{\mathbb{K}}] = [\mathbb{K}_{n+1}^{[1]}]$	1	12	16	18	22	39	65	58	42	55
(c) $[\hat{\mathbb{K}}] = [\mathbb{K}]$	1	12	24	35	43	65	96	115	133	152

**Table 9.2:** Notched specimen loaded in tension, material with linear isotropic hardening ( $h = 0,05E$ ): iteration count for each load step. Newton method (a) with consistent tangent matrix, (b) modified, using the elastoplastic tangent matrix calculated at the first iteration for each time step, and (c) modified, using the elastic stiffness matrix.



**Figure 9.13:** Notched specimen loaded in tension, material with linear isotropic hardening ( $h = 0,05E$ ): tensile force as a function of elongation (in dimensionless variables).





# **Appendices**



# A

## Gaussian quadrature rules

---

This appendix provides the quadrature points and weights for the most frequently used numerical integration methods, with  $G$  denoting the number of points for a given quadrature rule.

### A.1 Gauss-Legendre points for the segment $[-1, 1]$

Table A.1 provides the abscissas  $a_g$  and weights  $w_g$  for the 1-D Gauss-Legendre rules on the unit segment  $S = [-1, 1]$ . The abscissas are always interior to  $S$ ; they are located symmetrically with respect to the segment center  $a = 0$ , and the weights  $w_g$  obey the same symmetry. The positive abscissas, together with the associated weights, therefore completely define a Gauss-Legendre rule. The

$G$	$\pm a_g$	$w_g$	$p$
1	0.	2.	1
2	0.57735026918962576450	1.	3
3	0. 0.77459666924148337703	0.8888888888888888889 0.5555555555555555556	5
4	0.33998104358485626480 0.86113631159405257522	0.65214515486254614262 0.34785484513745385737	7
5	0. 0.53846931010568309103 0.90617984593866399279	0.5688888888888888889 0.47862867049936646804 0.23692688505618908751	9
6	0.23861918608319690863 0.66120938646626451366 0.93246951420315202781	0.46791393457269104738 0.36076157304813860756 0.17132449237917034504	11

**Table A.1:** Gauss-Legendre abscissas and weights for the reference segment  $-1 \leq a \leq 1$  (from Stroud and Secrest, 1966).

table also provides the maximum degree  $p = 2G - 1$  of polynomials that the rule integrates exactly.

## A.2 Gauss points and weights for the reference triangle

Table A.2 furnishes Gauss points  $(a_{1,g}, a_{2,g})$  and weights  $w_g$  for the reference triangle  $\Delta = \{(a_1, a_2) \mid (a_1, a_2) \geq (0, 0), 1 - a_1 - a_2 \leq 1\}$  (from Lyness and Jespersen, 1975). All points are interior to  $\Delta$ . Moreover, all these rules follow a ternary symmetry: all permutations of the barycentric coordinates  $a_{1,g}^T, a_{2,g}^T, 1 - a_{1,g}^T - a_{2,g}^T$  also define quadrature points, with the same weight  $w_g^T$ . This symmetry allows to compress the necessary information in Table A.2: quadrature points are given with their *multiplicity*  $M$ , which indicates the number of different points obtained from all different permutations of the three barycentric coordinates, and the maximum degree  $p$  of polynomials that the rule integrates exactly.

$G$	$a_{1,g}$	$a_{2,g}$	$w_g$	$M$	$p$
1	0.3333333333333333	0.3333333333333333	0.5	3	1
3	0.1666666666666667	0.1666666666666667	0.1666666666666667	3	2
6	0.445948490915965 0.091576213509771	0.445948490915965 0.091576213509771	0.111690794839005 0.054975871827661	3 3	4
7	0.3333333333333333 0.470142064105115 0.101286507323456	0.3333333333333333 0.470142064105115 0.101286507323456	0.1125000000000000 0.066197076394253 0.062969590272414	1 3 3	5

**Table A.2:** Gauss points, weights and order of accuracy for the reference triangle (from Lyness and Jespersen, 1975).

## A.3 Gauss points and weights for the reference tetrahedron

The Gauss points  $(a_{1,g}, a_{2,g}, a_{3,g})$  and weights  $w_g$  for the reference tetrahedron  $\Delta = \{(a_1, a_2, a_3) \mid (a_1, a_2, a_3) \geq (0, 0, 0), 1 - a_1 - a_2 - a_3 \leq 1\}$  are, similarly, given in Table A.3 (from Hammer et al., 1956). Only the two lowest-order

$G$	$a_{1,g}$	$a_{2,g}$	$a_{3,g}$	$w_g$	$M$	$p$
1	1/4	1/4	1/4	$\frac{1}{6}$	1	1
4	0.585410196624969	0.138196601125011	0.138196601125011	$\frac{1}{24}$	4	2

**Table A.3:** Gauss points, weights and order of accuracy for the reference tetrahedron (from Hammer et al., 1956).

rules are given, the second one being of multiplicity  $M = 4$  due to all different permutations of the four barycentric coordinates.

#### A.4 Gauss-Lobatto points and weights for the segment $[-1, 1]$

Table A.4 provides the abscissas  $a_g$  and weights  $w_g$  for the 1-D Gauss-Lobatto rules on the unit segment  $S = [-1, 1]$ . The quadrature points always include the segment endpoints; they are located symmetrically with respect to the segment center  $a = 0$ , and the weights  $w_g$  obey the same symmetry. The positive abscissas and their associated weights therefore completely define a Gauss-Lobatto rule. The table also provides the maximum degree  $p = 2G - 3$  of polynomials that the rule integrates exactly.

$G$	$\pm a_g$	$w_g$	$p$
3	0. 1.	$4/3$ $1/3$	3
4	$1/\sqrt{5}$ 1.	$5/6$ $1/6$	5
5	0. $\sqrt{3/7}$ 1.	$32/45$ $49/90$ $1/10$	7
6	$\sqrt{(7-2\sqrt{7})/21}$ $\sqrt{(7+2\sqrt{7})/21}$ 1.	$(14+\sqrt{7})/30$ $(14-\sqrt{7})/30$ $1/15$	9

**Table A.4:** Gauss-Lobatto abscissas and weights for the reference segment  $-1 \leq a \leq 1$ .



# B

## Introduction to the use of the codes developed in this book

---

The MATLAB codes employed in the book have been developed by the authors with a pedagogical purpose. They always privilege simplicity and clarity and are very rudimentary with respect to the codes employed in the industrial context. They also have very limited performances in terms of speed and maximum size of the problems which can be analysed. The sources can be downloaded from the website [www.ateneonline.it/bonnet](http://www.ateneonline.it/bonnet) and from the personal pages of the authors.

The main files are collected in the root directory of the distribution, together with a copy of the `Gmsh.exe` file. Other subdirectories collect service files:

- `subB2`, `subT3`, `subT6` ... (one directory per element type) contain subroutines for computing element quantities;
- `all-elements` contains the subroutines which are common to all element types (for instance the function for the radial return algorithm in plasticity);
- `input` contains some examples of input files analysed in the sequel;

The execution of a code can be launched with the standard MATLAB procedures, i.e. either by typing the name of the code on the command line or by pressing the F5 button on the editor window.

Some codes do not require any pre- or post-processing capabilities. These are typically associated with problems which do not involve space discretization or with 1D or 3D problems with spherical symmetry which are defined on a segment. These codes are:

- `sphere_B2_1D_solid.m`, introduced in section 1.5.6 for the analysis of a linear-elastic hollow sphere subjected to a given pressure and displacement boundary conditions.
- `sphere_B2_1D_diff.m`, introduced in section 4.2.5 for the simulation of heat diffusion in a sphere.
- `bar_B2_1D_wave.m`, developed in sections 5.3.2 and 5.4.8. In the two variants `modal` and `direct` it addresses an elastic bar in dynamics employing either modal decomposition or direct integration approaches

- `beambending_P1P1.m`, implementing the displacement approach for a cantilever Timoshenko-beam in section 6.3.1
- `strip_plast.m`, introduced in chapter 7 to demonstrate the implementation of an elastoplastic constitutive law in combination with a simple equilibrium condition

These examples have been created to treat specific examples, but similar applications can be addressed with minor modifications that are suggested as useful exercises.

The other codes developed in the book are

- `genlin`, section 3.6: non-evolutionary linear scalar and vector problems in 2D or 3D.
- `heatdiffusion`, section 4.2.6: heat diffusion problems with the generalised trapezoidal rule
- `dynamics`, section 5.6: problems in elastodynamics with direct integration approaches. Two specific application based on dynamics are `dynamics_tyre` and `dynamics_tyre_rolling` simulating the contact (section 5.8.4) and rolling of a tire down a slope (section 8.4.3)
- `geomNL`, section 8.4.1: large displacement, small strain analysis applied to the buckling of structures
- `plasticity`, section 9.8: Von Mises elastoplasticity with consistent tangent matrix

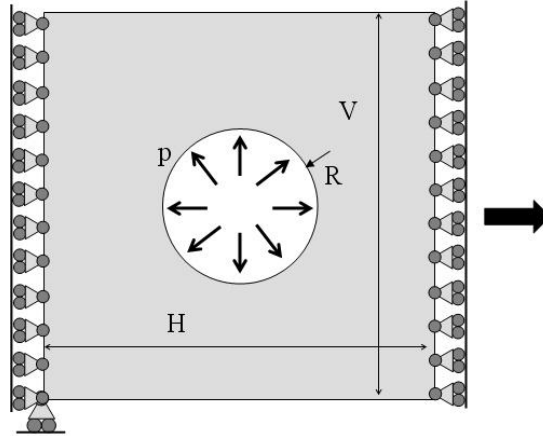
These codes are rather general and can be run with any input file generated by the user according to the rules described in the respective sections. They rest on the pre- and post- processing capabilities provided by the free code Gmsh, thus achieving a certain generality and, most of all, a large flexibility and ease of adaptation to different contexts. All the codes are based on the structure of `genlin`. For this reason the following sections will focus on `genlin` itself, while the peculiarities of the other codes are commented in the sections indicated.

## B.1 Geometry and mesh files

The definition of the problem geometry and of the associated mesh is here performed with the code Gmsh v. 2.8.3. This code is freely available, together with the necessary documentation, at the web-page [www.geuz.org/Gmsh/](http://www.geuz.org/Gmsh/), but the version employed in this book is also distributed together with the codes in order to guarantee compatibility. Gmsh is a rather powerful tool which requires some training in order to be employed at the best of its capabilities. This, though recommended, is left to the initiative of the interested readers and we will here only limit ourselves to presenting a small guide showing how to treat the example of a plate with a hole of Figure B.1.

We first create the file `Appendix_example.geo` collecting some topological informations allowing to define the geometry of the surface. The file starts with the definition of some parameters: the dimensions  $H$  and  $V$  of the plate and the





**Figure B.1:** Example problem: thick plate with a hole

radius  $R$  of the circular hole; next the mesh-size parameters  $lc1$  and  $lc2$  are employed to fix the level of mesh refinement at some points:

```
// global dimensions
```

```
H=10;  
V=10;  
R=2;
```

```
// mesh parameters
```

```
lc1=1;  
lc2=0.5;
```

The geometry is defined following a bottom-up procedure. We start defining the corner points. Each point is specified by the three coordinates and one mesh-size parameter:

```
// point coordinates
```

```
Point(1) = {-H/2, -V/2, 0, lc1};  
Point(2) = { H/2, -V/2, 0, lc1};  
Point(3) = { H/2,  V/2, 0, lc1};  
Point(4) = {-H/2,  V/2, 0, lc1};
```

```
Point(5) = { 0, 0, 0, lc2};  
Point(6) = { R, 0, 0, lc2};  
Point(7) = { 0, R, 0, lc2};  
Point(8) = {-R, 0, 0, lc2};  
Point(9) = { 0, -R, 0, lc2};
```

Next lines are traced. The definition of an oriented straight segment only requires the start and end point; an oriented circular arc (of angle strictly smaller than  $180^\circ$ )

is defined by the start point, the center, and the end points. In order to create a surface one has to define the closed loops representing the outer and internal borders, oriented in counter-clockwise and clockwise directions, respectively. This is done employing the command `Line Loop` and listing the sequence of lines in the loop

```
// lines and line loops

Line(1) = {1,2};
Line(2) = {2,3};
Line(3) = {3,4};
Line(4) = {4,1};
Line Loop(5) = {1,2,3,4};

Circle(6) = {6,5,7};
Circle(7) = {7,5,8};
Circle(8) = {8,5,9};
Circle(9) = {9,5,6};
Line Loop(10) = {-6,-7,-8,-9};
```

The “minus” signs in the list of lines imply that the orientation of the segment has to be inverted. At this point one can define the surface by providing the list of outer and inner borders

```
// surface

Plane Surface(1) = {5,10};
```

The geometrical entities created so far are called *elementary*: points, lines and surfaces (and, in 3D, volumes). The elementary entities of every type cannot have duplicate numbers within the same type. One can also create *physical* entities which are combinations of the elementary entities already introduced. This possibility is exploited for creating more complex meshes (we will see an example in fracture mechanics, for instance) but especially for providing mandatory informations employed in the analysis file (discussed in the sequel). Whenever a boundary condition is enforced on a boundary feature, or a material has to be associated to a solid, these must be defined as a physical entities.

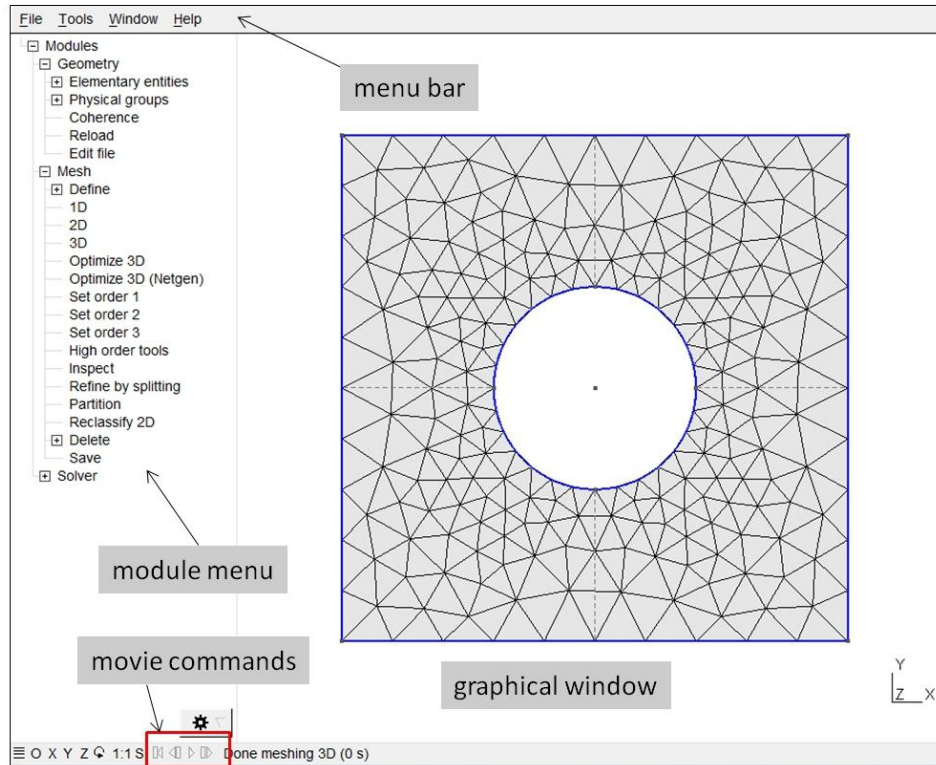
In our specific case, boundary conditions will be enforced on the vertical lines (physical entities number 1 and 2), pressure will be applied on the inner hole (physical entity number 3) and the surface (physical entity 4) will be associated to specific constitutive properties.

```
// physical entities

Physical Line(1) = {2};
Physical Line(2) = {4};
Physical Line(3) = {-6,-7,-8,-9};
Physical Surface(4) = {1};
```

By convention the physical entities are identified by an increasing number irrespective of their type. It should be recalled that, apart from nodes, only physical entities are written in the output file generated by Gmsh.

Once the `Appendix_example.geo` file has been created, in order to generate the mesh one has to run Gmsh. When the graphical window (see Figure B.2) appears, the file `Appendix_example.geo` is opened from *File* → *Open* in the *menu bar*. Next, clicking *2D* in the *Mesh* module of the *module menu*, Gmsh automatically generates a mesh of the plate using linear T3 elements. By clicking *Save* all the informations are saved in the file `Appendix_example.msh`. A mesh of second order T6 elements can be alternatively generated clicking *Set order 2* tab and then saving the file as before. In the specific case of the example, Gmsh creates the geometry and mesh depicted in Figure B.2. It is worth stressing that all the default values of parameters (for instance the background color of the graphical window, or the order of the elements) can be modified with the *Tools* → *Options* menu in the *menu bar* and saved with the *File* → *Save Model Options* command.



**Figure B.2:** Gmsh window with the mesh created for the example problem

We now analyse the file `Appendix_example.msh` which Gmsh produces in the case of a first order mesh. The file contains, after three lines defining the version of the code employed, a first section `$Nodes` providing the list of nodes and their coordinates. First the number  $N_N$  of nodes is given, followed by a series of  $N_N$

lines:

```
$Nodes
236
1 -5 -5 0
2 5 -5 0
3 5 5 0
4 -5 5 0
...
...
234 -2.4999999999996938 4.514062992162897 0
235 2.4999999999996938 -4.514062992162899 0
236 -4.514062992162899 -2.4999999999996938 0
$EndNodes
```

Every line contains the number of the node and its three coordinates. Since in this specific example we are addressing plane problem, the third coordinate is always set to zero. The section is ended by the keyword `$EndNodes`. It is worth remarking that Gmsh does not guarantee that the numbering of the nodes is sequential and this means in particular that, even if there are exactly  $N_N$  nodes in the mesh, the maximum tag is not necessarily  $N_N$ .

The second section, beginning with `$Elements`, defines the elements:

```
$Elements
452
1 1 2 1 2 2 18
2 1 2 1 2 18 19
3 1 2 1 2 19 20
4 1 2 1 2 20 21
5 1 2 1 2 21 22
6 1 2 1 2 22 23
...
...
450 2 2 4 1 125 222 221
451 2 2 4 1 127 226 225
452 2 2 4 1 128 228 227
$EndElements
```

The mesh of this specific example has been created with T3 triangles for the surface, but B2 line elements are automatically created on the physical sets which contain lines and have been introduced to enforce prescribed boundary conditions. The B2 bars are not independent from the T3 triangles since they share the same nodes, but the parents-children correspondence is not explicitly provided. Gmsh associates a code to every element type. The set of codes for the elements employed in the codes is presented in table 3.5. For instance, 1 denotes B2 elements and 2 T3 triangles.

The first line of the `$Elements` section provides the total number of elements which are defined in the section, one per line. Every line contains

1. the element tag
2. the element type
3. a code which is not employed herein

4. the number of the physical entity to which the element belongs
5. the number of the elementary entity to which the element belongs
6. the connectivity of the element (the list of nodes of the element)

The section is closed by the keyword `$EndElements`.

## B.2 Analysis file for `genlin`

In the sequel we will focus on the procedure required to create analysis files for `genlin`. The other codes call for minor modifications which are highlighted in the relevant chapters.

Let us again focus on the structure of Figure B.1. The plate is made of a linear elastic homogeneous material with Young modulus  $E = 1$  and Poisson coefficient  $\nu = 0.3$  and is undergoing infinitesimal transformations in plane-strain conditions.

The hole boundary is subjected to a given pressure  $p = 2$ ; on the left side of the plate  $u_1$  is forced to vanish, while  $u_1 = 0.5$  is imposed on the right vertical side.

Even if the resultant of external forces in the vertical direction vanishes and is hence compatible with equilibrium, the solution in terms of displacements is defined up to an arbitrary constant  $\alpha e_2$ . Since this indeterminacy would lead to a singular stiffness matrix, an additional constraint is added blocking the vertical displacements of the left lower corner of the plate. This is graphically represented by an horizontal roller placed on the corner node. It is a priori guaranteed that the reaction force exerted by the roller on the plate will vanish and that the solution will not be perturbed. This is a general rule: whenever the displacement boundary conditions do not block all rigid body motions, additional well placed concentrated constraints must be introduced in a number equal to the rigid body motions that have to be eliminated.

All the data required to perform a finite element analysis of this problem are collected in the analysis file, discussed in the sequel, `Appendix_example.m` and a mesh file, created according to the guidelines presented in the previous section.

Two remarks are worth stressing: i) conventions adopted in this context are neither universal nor optimal, but only represent a reasonable choice justified in particular by the desire to keep the procedure of reading the input files as simple as possible; ii) “units” are mentioned nowhere in the codes and it is the user’s responsibility to provide input data consistent with a specific (and admissible!) choice of units. The output produced by the code will be consistent with the input itself.

The analysis file `Appendix_example.m` is a standard MATLAB Mfile and first provides the name, `Appendix_example.msh`, of the file (placed in the same directory as `Appendix_example.m`) defining the mesh of the problem. As discussed in Section B.1, `Appendix_example.msh` defines nodes, elements and also physical entities (element sets). An element set (called `elset` for brevity in the sequel) is simply a collection of elements (either line, surface or volume elements) which are employed for the discretization of specific geometrical entities of the structure.

**Box B.1:** File `Appendix_example.m`

```
file='Appendix_example.msh';  
  
analysis.type=4;  
  
material = [1 1/3];  
  
solid = [4 1];  
  
dbc = [1 1 0.5;  
       2 1 0];  
  
dbcn = [1 2 0];  
  
tbc = [3 0 -2];
```

In our specific example we have:

- elset 1: list of line elements on the right vertical side;
- elset 2: list of line elements on the left vertical side;
- elset 3: list of line elements on the inner circle;
- elset 4: list of surface elements on the surface of the plate;
- node 1: node placed on the left lower corner

The variable `analysis.type` (the field type of the structure analysis) defines the type of analysis. For instance `analysis.type=4` declares a linear elastic plane-strain problem in statics (all the available options are listed in table 3.2).

The variable `material` defines the Young modulus and Poisson coefficient of a linear elastic material. Different materials can be defined in different lines of `material`.

The variable `solid` establishes a link between elements and materials. The structure to be analysed is meshed with elements collected in the element set number 4 and is made of material number 1 (defined in the first line of `mat`).

The variable `dbc` gives displacement boundary conditions assigned on elsets, one per every line. The first line states that on elset 1 the displacement in direction 1 is imposed equal to 0.5, while the second line states that on elset 2 the displacement in direction 1 is forced to vanish.

The variable `dbcn` defines the displacement boundary conditions imposed on single nodes, one per line of `dbcn`. In this case node number 1 is blocked in direction 2.

The variable `tbc` describes traction boundary conditions. Every line of `tbc` defines a specific condition imposed on an elset. In our example a traction is imposed on the inner circle (elset 3), in direction 0, with intensity per unit length equal to -2. By convention direction 0 denotes the direction  $\mathbf{n}$  orthogonal to the line. The normal  $\mathbf{n}$  is defined as follows: if  $\mathbf{e}_3$  denotes the out-of-plane direction and  $\mathbf{t}$  is the unit tangent to the line, then  $\mathbf{n} \wedge \mathbf{t} = \mathbf{e}_3$ . The orientation of the line is fixed during the definition of the geometry (see Section B.1) and in this

case is clockwise along the inner circle. As a consequence, the loading actually represents a pressure of intensity 2. For a plane problem the direction of the loading can also be either 1 for  $e_1$  or 2 for  $e_2$ . In the present version of the code only uniform tractions can be imposed on an elset. However, small modifications would allow more flexibility and are left as an exercise. By convention, a portion of the boundary on which no boundary conditions are enforced along a specific direction, is implicitly subjected to zero tractions in that direction.

### B.3 Reading the input files

We assume that the user has compiled the `.m` and `.geo` input files and has saved the mesh in the `.msh` files as detailed in the previous sections. All the definitions contained in the data and mesh files must be accessed by the main code `genlin.m` before proceeding. The input phase should be analysed in parallel with section 3.6, where many definitions of the variables employed are provided. MATLAB reads the whole Mfile at once through command `run`.

```
eval(['run input/',name]);           % input file is read at once
```

At this stage all the variables defined in the `.m` file are loaded in the workspace. In particular the value of `analysis.type` has been fixed (see table 3.2) which allows to select the specific values of the analysis tag `Atag`, of the number of degrees of freedom per node `ndof` and of the geometrical dimensionality  $D \rightarrow DG$  (section 3.6.4). Next, the file `readgmsh` collecting all the input commands is launched

```
readGmsh % input file is read
```

Its content is analysed in what follows. The string `file` contains the name of the mesh file (for instance `Appendix_example.msh`) which can be opened in reading mode:

```
mshfile=['input/' file];
fmid=fopen(mshfile,'r');           % opens input file
```

The first four lines contain informations of no interest for the analysis and are hence read without performing any further action:

```
tline = fgets(fmid);               % reads a series of lines in preamble
tline = fgets(fmid);
tline = fgets(fmid);
tline = fgets(fmid);
```

The fifth line, on the contrary, provides the number of nodes defined in the sequel. The code first reads the whole string and then extracts the integer `NN` by means of the command `sscanf`:

```
tline = fgets(fmid);               % gets string with number of nodes NN
NN=sscanf(tline,'%d');             % reads NN from string
```

Reading the nodal coordinates is complicated by the fact that, as recalled previously, Gmsh does not guarantee a consecutive numbering of the nodes. In order

to solve this issue without increasing the complexity of the core of the code, a non-optimal strategy is adopted. First, the largest tag `maxnodenum` attached to a node is read and then the list of nodal structures nodes defined in table 3.3 of length `maxnodenum` is allocated. This implies that some of the structures of nodes will not be filled with any information since “holes” are present in the Gmsh numbering. This implies a slight waste of memory space. However, the advantage of this choice is the direct correspondence of the position in nodes with the elements connectivity, as will be discussed in the sequel.

In order to put this in practice, the list of nodes in the meshfile must be read twice. For this reason the current position in the file is saved with the command `ftell`. In the first loop over the `NN` nodes the list `nodenum` is filled with the tags of the nodes and `max(nodenum)` directly yields `maxnodenum`.

```
position = ftell(fmid);           % memorizes position in mesh file
nodenum=zeros(NN,1);
for i=1:NN,                       % loop lines of section $Nodes
    tline = fgets(fmid);
    h=sscanf(tline,'%d %f %f %f'); % h contains node number and 3 coords
    nodenum(i)=h(1);              % saves the node number in nodenum
end
maxnodenum=max(nodenum);
fseek(fmid, position, 'bof');
```

The file is next repositioned at the saved position in order to preallocate the nodes list of structures and to perform the second loop during which the nodes are actually read. Each element in nodes is a structure containing several fields defined in table 3.3: `coor` for the nodal coordinates, `dof` for the global numbering of nodal degrees of freedom and `U` for their values. These two latter fields are simply set to zero at this level.

```
nodes=repmat(struct('nodes',[],... % allocates the list nodes
                    'dof',[],...
                    'U',[],1,maxnodenum);
for i=1:NN,                       % loop to read the nodes
    tline = fgets(fmid);
    h=sscanf(tline,'%d %f %f %f'); % reads node number and 3 coordinates
    nodes(h(1)).coor=h(2:1+DG);    % saves coordinates
    nodes(h(1)).dof=zeros(1,ndof); % init to zero
    nodes(h(1)).U=zeros(1,ndof);  % init to zero
end
analysis.NN=NN;                   % saves number of nodes
```

Before starting the input phase for the elements, the code needs to set some variables. The command `numel` extracts the number of rows of the matrix `solid` and assigns it to `nsolid` which hence represents the number of elsets associated with materials in the analysis file. Similarly `ndbc`, `ndbcn`, `ntnbc` are, respectively, the number of displacement boundary conditions imposed on elsets and on nodes and the number of loading conditions. Since one or more of these conditions might be missing from a given analysis, the code first checks with `exist` the existence of the relevant variables. If `dbc` or `tbc` do not exist, they are set to zero in order to



simplify certain search procedures discussed in the sequel.

```

nsolid=numel(solid(:,1));          % number of sets assoc. to materials
ndbc=0; ndbcn=0; ntbc=0;
if exist('dbc')
    ndbc=numel(dbc(:,1));          % number of sets with imposed dbc
else
    dbc=0;
end
if exist('tbc')
    ntbc=numel(tbc(:,1));          % number of sets with imposed tbc
else
    tbc=0;
end
if exist('dbcn')
    ndbcn=numel(dbcn(:,1));        % number of nodes with imposed dbc
end

```

Next all the different types of elements (line, surface and volume elements) are read and stored in suitable lists of structures. Material elements, i.e. those elements which are associated with materials, are stored in the list `elements`, while load elements, i.e. those elements which are associated with loading conditions, are stored in the list `load`. First, the number `NE` of material elements and `NL` of load element are set to zero and the global number of elements is read.

```

NE=0;
NL=0;
tline = fgets(fmid);              % string with the number of elements
num=sscanf(tline,'%d');           % reads the global number of elements

```

Proceeding in a way similar to what done with nodes, during a first loop the numbers `NE` and `NL` of material elements and of loaded elements, respectively, is computed. This is achieved by reading the lines one by one, picking the fourth coefficient which represents the physical set and checking whether any boundary condition is enforced on the `elset`. In order to avoid doing these checks twice the matrix `eltype=zeros(num,2)` is employed to save the results. For every element the physical set `physet=h(4)` is read and different actions are performed according to its value. At the end of the loop the reading position is reset the beginning of the element list.

```

eltype=zeros(num,2);
position = ftell(fmid);            % memorizes position in mesh file
for i=1:num,
    tline = fgets(fmid);
    h=sscanf(tline,'%d');
    physet=h(4);                  % physical set

    ... according to physet value performs diff. actions ...

end
fseek(fmid, position, 'bof');

```

For each element the search `find(solid(:,1)==physet)` is performed in the first

column of `solid` to check if a material is associated to the element. If this is the case `pos`, the material number, is saved in `eltype` together with the tag 1 and the code continues to the next element.

```
pos=find(solid(:,1)==physet);
if ~isempty(pos)                % if physet associated with a mat
    NE=NE+1;
    eltype(i,:)=[1 pos];        % saves in eltype the tag 1 and set
    continue
end
```

Otherwise the same search is performed in the first column of `tbc` to check if a loading condition is associated to the element. Since more loadings can be applied to the same elset, the number of loads is incremented as `NL=NL+length(pos)`. The tag 2 is saved in `eltype` and the code continues to the next element.

```
pos=find(tbc(:,1)==physet);
NL=NL+length(pos);
eltype(i,:)=[2 0];
```

A final search in `dbc` allows to establish whether one or more boundary conditions have been enforced on the element. For all the elsets in `pos`, the field `.dof` of the list nodes is filled with -1 in the direction specified by `dbc(pos(iL),2)`, while the field `.U` is filled with the imposed value of the boundary condition.

```
pos=find(dbc(:,1)==physet);
for iL=1:length(pos)
    for n=6:length(h)
        nodes(h(n)).dof(dbc(pos(iL),2))=-1;
        nodes(h(n)).U(dbc(pos(iL),2))=dbc(pos(iL),3);
    end
end
```

Out of the first loop, the lists of material elements and of load elements are allocated (see tables 3.4-3.6)

```
elements=repmat(struct('type',0,... % allocates list of elements
                      'solid',0,...
                      'nodes',[]),1,NE);
loads=repmat(struct('type',0,... % allocates list of loads
                   'nodes',[],...
                   'dir',0,...
                   'val',0),1,NL);
```

A second loop, identical to the first one, is launched over the elements and different actions are performed according to the tags saved in `eltype`, using the `switch` construct. In any case `pos=eltype(i,2)`; contains the number of the physical set to which the element belongs.

If `eltype(i,1)==1` the element is a material one. Three different data are saved in the fields of the elements list of structures: in `.type` the type of the element; in `.mat` the material number; in `.nodes` the list of nodes representing the connectivity of the element. Remark that the elements list does not preserve

Gmsh numbering.

```
switch eltype(i,1)
case 1,
    NE=NE+1;
    elements(NE).type=h(2);           % element type
    elements(NE).mat=solid(pos,2);    % element material
    elements(NE).nodes=h(6:length(h)); % connectivity of the element
```

If `eltype(i,1)==2` the element is the support of one or more distributed loads. The data concerning the loads applied to the element are stored in the list `load`, which, in addition to the fields `.type` and `.nodes` having the same meaning as for the list `elements`, contains the fields `.dir` and `.val` which give the direction and the value of the loading (see the previous section for the conventions on the direction).

```
case 2,
    pos=find(tbc(:,1)==physet);
    for iL=1:length(pos)
        NL=NL+1;
        loads(NL).type=h(2);
        loads(NL).nodes=h(6:length(h));
        loads(NL).dir=tbc(pos(iL),2);
        loads(NL).val=tbc(pos(iL),3);
    end
```

After saving in the structure `analysis` the number of elements `NE` and `NL`, this last procedure is repeated for the `ndbcn` displacements imposed on isolated nodes:

```
analysis.NE=NE;
analysis.NL=NL;

for j=1:ndbcn
    nodes(dbcn(j,1)).dof(dbcn(j,2))=-1;
    nodes(dbcn(j,1)).U(dbcn(j,2))=dbcn(j,3);
end

fclose(fmid);
clear eltype nodenum
```

The input reading phase is now over, the mesh file is closed (`fclose`) and some variables cleared from memory.

## B.4 Graphical post-processing

A simple interface to Gmsh is included in the codes in order to exploit its post-processing capabilities for the graphical visualization of results. We will discuss here only some basic options and the user is referred to the on-line documentation for a complete guide to Gmsh. The interface can be easily improved and optimized for the specific desired application, especially in 3D. Each of the codes has a different interface with slightly different options. For instance, in the case of `heatdiffusion`, `dynamics`, `geomNL` and `plasticity` nodal values are written for

several values of the time parameter in order to generate a movie of the deformed shape (using the movie commands in figure B.2). The analysis of the different options is left to the user and we will here only focus on the interface for genlin.

The aim of `genlin_postgmsb.m` is to prepare the output file `out.msh` and the option file `opti.geo` and finally launch Gmsb with the command:

```
Gmsb.exe input/Appendix_example.msh input/out.msh input/opti.geo
```

where `Appendix_example.msh` is the mesh file.

The file `out.msh` contains the results of the analysis in terms of the nodal degrees of freedom nodes.U followed by the post-processing quantities extrapolated to nodes and saved in Sn. The file has to be written according to a specific syntax which starts with a conventional definition of the format:

```
$MeshFormat
2.2 0 8
$EndMeshFormat
```

After opening `out.msh` in writing mode, `postGmsb` writes these lines automatically using the `fprintf` command:

```
fmid=fopen('input/out.msh','w');
fprintf(fmid,'%s\n%s\n%s\n',' $MeshFormat','2.2 0 8',' $EndMeshFormat');
```

These lines are followed in `out.msh` by the list of nodal values nodes.U, component by component, node by node. E.g. in the case of the example analysed (two components of displacement per node) with 236 nodes, the file will list the first component (236 values preceded by some conventional codes) followed by the second component.

\$NodeData	start of section
1	conventional code
"U1"	name of the component
1	conventional code
0.0	time value (0 in genlin)
3	conventional code
0	time step (0 in genlin)
1	one component per field
186	number of nodes
1 0.000000E+000	value of first node
2 5.000000E-001	value of second node
...	
...	
186 -1.867125E-001	value of last node
1	conventional code
"U2"	name of the second component
1	conventional code
...	
...	
\$EndNodeData	end of section

The comments on the right have been added here for explanation purposes but are missing in the real file, in general. Since the number of nodal values per node is

ndof, the MATLAB lines required to write this section are

```
for icomp=1:ndof          % loop over components
    fprintf(fmid,'%s\n','$NodeData','1',...
        ['"U',num2str(icmp),'"', '1','0.0','3','0','1']);
    fprintf(fmid,'%d\n',analysis.NN);
    for n=1:analysis.NN    % loop over nodes
        fprintf(fmid,'%d %E\n',n,nodes(n).U(icmp));
    end
    fprintf(fmid,'%s\n','$EndNodeData');
end
```

Next, out.msh is filled with the post-processing quantities in Sn, always one component (186 values) after the other. For instance, in plane elastic analyses one would have the three (Sdim) stresses. Since this section is almost identical to the previous one, its analysis is left as an exercise.

Finally, the out.msh file is closed with fclose(fmid); and MATLAB starts writing the options file opti.geo, which controls the way the charts are visualised in Gmsh. Each of the \$NodeData fields created in out.msh is now a View numbered from 0 to nodf+Sdim-1. In our specific case five views are available (displacement and stress components), starting from View[0] to View[4]. The background color is set to white and the colour charts will be visualized without the superposed mesh (elements and edges).

```
General.BackgroundGradient=0;
Mesh.SurfaceEdges=0;
Mesh.SurfaceFaces=0;
```

By default the first chart is made visible when the postprocessor is launched. All the views will be presented as filled isovalue-charts with 21 levels.

```
View[0].Visible = 1;
View[0].NbIso=21;
View[0].IntervalsType=3;
...
...
View[4].Visible=0;
View[4].NbIso=21;
View[4].IntervalsType=3;
```

The MATLAB code writing these lines is

```
fmid=fopen('input/opti.geo','w');
fprintf(fmid,'%s\n','General.BackgroundGradient=0;');
fprintf(fmid,'%s\n','Mesh.SurfaceEdges=0;');
fprintf(fmid,'%s\n','Mesh.SurfaceFaces=0;');
fprintf(fmid,'%s\n','View[0].Visible = 1;');
fprintf(fmid,'%s\n','View[0].NbIso=20;');
fprintf(fmid,'%s\n','View[0].IntervalsType=3;');
for n=1:ndof+DOUT-1
    fprintf(fmid,'%s%d%s\n','View[' ,n,'].Visible=0;');
    fprintf(fmid,'%s%d%s\n','View[' ,n,'].NbIso=21;');
    fprintf(fmid,'%s%d%s\n','View[' ,n,'].IntervalsType=3;');
end
fclose(fmid);
```

The user can modify any option according to needs. For instance, in order to better appreciate the contour plots, the edges can be toggled on and off (for the selected View) in *Tools* → *Options* → *Visibility* → *Draw element outline*. If strong gradients are present, maximum and minimum values can be clipped setting *Tools* → *Options* → *General* → *Range Mode* to *Custom* and selecting the desired *Custom min* and *Custom max* values (and possibly activating the *Saturate* option).

Starting from this basis, many functionalities can be added, either from the *Tools* → *Options* tab of the *menu bar* or writing directly the instructions in the *opti.geo* file. As an example, one can create a new view with the deformed configuration of the specimen. This can be obtained with the following strategy. First hide the current view by clicking the relevant box in the module menu. Then select *Tools* → *Plugin* → *Scal2Vec* plugin. Now set *View X* to 0, and *View Y* to 1 (the views containing the two displacement components) and then click *Run*. A new view is added with the deformed configuration. Set *Aspect* → *Vector Display* to *Displacement* in order to plot the displacement field of the deformed configuration. The deformation might be exaggerated or too small, but this scale factor can be set modifying the options of the new view from *Tools* → *Options* and selecting the newly created view. Then modify as desired *Displacement factor* in *Aspect*. The same deformed configuration can be built, in the case of plane elasticity problems, adding the following commands to the *opti.geo* file:

```
View[0].Visible=0;
Plugin(Scal2Vec).ViewX=0;
Plugin(Scal2Vec).ViewY=1;
Plugin(Scal2Vec).Run;
View[5].Name = "Def";
View[5].Visible = 1;
View[5].IntervalsType=3;
View[5].VectorType = 5;
View[5].NbIso=21;
```

Similarly one can create new views by cutting or clipping 3D plot with planes or boxes exploiting the numerous plugins developed within Gmsh.

It is worth recalling here that in the interfaces for other codes (e.g. *geomNL*, *plasticity*) all the time history of displacements is written in *out.msh* and the whole displacement vector is defined at once, by specifying the three components for every node:

\$NodeData	start of section
1	conventional code
"U"	name of the component
1	conventional code
0.0	time value
3	conventional code
1	time step
3	three components per field
186	number of nodes
1 2.043334E-001 -1.250000E-002 0	first node
2 -9.622657E-001 -1.250000E-002 0	
...	

```

...
186 -9.62257E-001 -1.25000E-002 0 last node
$EndNodeData end of section

```

This approach is very effective for creating the movie of the deformed structure through the time history.

## B.5 Geometry files for fracture mechanics problems

The mesh files for the examples of chapter 6 are created according to the same guidelines as in the previous paragraphs, the only difference being that “double nodes” must be introduced on the crack faces, that is couples of nodes having different global numbering but sharing the same coordinates. In order to achieve this, the plate surface represented in figure B.3 is generated as the union of two portions, representing the upper and lower parts of the plate, respectively, Physical Surface(3) = {1,2}. The closed loops employed to define the sub-surfaces share the same segments on the line separating the two parts, but for the crack line where the distinct segments Line(4) et Line(10) are utilized. The file Chap7\_fract.geo contains an example of geometry definition:

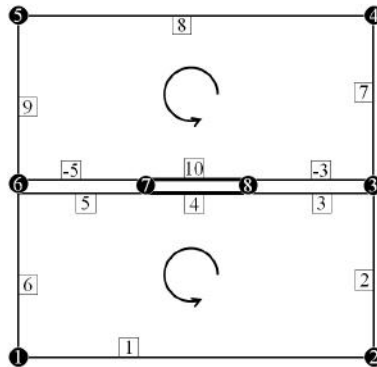
```

H=5;      // semiwidth of plate
V=5;      // semiheight of plate
a=1;      // semilength of crack (center crack is at x1=x2=0)

lc1=.5;
lc2=.2;
lc3=.01; // element size at crack tip

Point(1) = {-H, -V,0.0,lc1};
Point(2) = { H, -V,0.0,lc1};
Point(3) = { H,0.0,0.0,lc2};
Point(4) = { H, V,0.0,lc1};
Point(5) = {-H, V,0.0,lc1};
Point(6) = {-H,0.0,0.0,lc2};
Point(7) = {-a,0.0,0.0,lc3};
Point(8) = { a,0.0,0.0,lc3};

```



**Figure B.3:** Cracked plate: definition of the closed loops for generating double nodes on the crack.

```
Line(1) = {1,2};
Line(2) = {2,3};
Line(3) = {3,8};
Line(4) = {8,7};
Line(5) = {7,6};
Line(6) = {6,1};
Line(7) = {3,4};
Line(8) = {4,5};
Line(9) = {5,6};
Line(10) = {7,8};

Line Loop(11) = {1,2,3, 4, 5, 6}; // border of lower surface
Line Loop(12) = {7,8,9,-5,10,-3}; // border of upper surface

Plane Surface(1) = {11};           // lower surface
Plane Surface(2) = {12};           // upper surface

Physical Line(1) = {1};             // lower line for traction
Physical Line(2) = {8};             // upper line for traction
Physical Surface(3) = {1,2};
```

A different approach is employed in the file `Chap7_fract_ros.geo` which generates the geometry represented in figure 7.6 and exploits symmetry in order to represent only 1/4th of the plate. The mesh is structured in the form of a “rosette” of elements around the crack tip. The reading of the geometry file is suggested as an exercise on Gmsh.

## B.6 Code `genlin_fast`

The code `genlin` becomes particularly slow when employed to analyse elasticity problems with quadratic elements as soon as the mesh grows in size (say when the number of elements is larger than  $10^4$ ). The user can easily verify that this slowness is mainly due to the assembly phase, as already remarked in section 3.6.3. For some reason that we will not investigate herein, the automatic treatment of the matrix sparsity in MATLAB becomes inefficient in this case.

A partial solution to this issue is presented in this appendix, as an example of technical provision which can favour speed at the cost of simplicity and clarity of the code. This solution has been programmed in `genlin_fast.m` which is an exact copy of `genlin.m` but for the assembly phase.

The new procedure rests on the explicit creation of the three arrays  $C_m$ ,  $I_m$ ,  $J_m$  which are closely related to the three arrays  $C$ ,  $I$ ,  $J$  generally employed for the Morse stocking (see section 3.4.3)

- $C_m$  contains the non-zero coefficients  $[\mathbb{K}]_{IJ}$
- $I_m$  contains the row position of each coefficient of  $C_m$  such that, if  $[\mathbb{K}]_{IJ}$  is stored in  $C_m(\text{pos})$ , then  $I_m(\text{pos})=I$
- $J_m$  contains the column position of each coefficient of  $C_m$  such that, if  $[\mathbb{K}]_{IJ}$  is stored in  $C_m(\text{pos})$ , then  $J_m(\text{pos})=J$

The idea is to store the coefficients in  $C_m$  in line order (a line after the other) and save in the array `startrow(1:analysis.neq)` the position of the first co-



efficient in each line of  $[\mathbb{K}]$ , so as to speed up certain search procedures. An estimate of the maximum number of non-zero coefficients associated to each line of the matrix has been obtained in section 3.6.3) as  $\text{ndof} \times \text{nm}(n)$ . The array  $\text{startrow}(1:\text{analysis.neq})$  is filled with an external loop over nodes and a nested loop over degrees of freedom of every node. If the current dof is active (unknown nodal value),  $\text{startrow}(\text{dof})$  is set to  $\text{ncoeffs}+1$ , which is next incremented by  $\text{ndof} \times \text{nm}(n)$

```

startrow=zeros(analysis.neq,1);           % index of row start
ncoeffs=0;
for n=1:analysis.NN,                     % loop over nodes
    for dir=1:ndof                         % loop over directions
        dof=nodes(n).dof(dir);            % DOF associated to node and idir
        if dof>0,                          % only if nodal value is unknown
            startrow(dof)=ncoeffs+1;
            ncoeffs=ncoeffs+ndof*nm(n);    % increments number of coefficients
        end
    end
end
Cm=zeros(ncoeffs,1);                     % allocates Cm
Jm=ones(ncoeffs,1);                      % allocates Jm vector
Im=ones(ncoeffs,1);                      % allocates Im vector
lenrow=zeros(analysis.neq,1);            % length of each row

```

The additional array  $\text{lenrow}=\text{zeros}(\text{analysis.neq},1)$  will be employed to store the number of non zero coefficients in every row.

At the end of the loops  $\text{ncoeffs}$  contains an estimate of non-zero coefficients and  $\text{Cm}$  is initialised to  $\text{zeros}(\text{ncoeffs},1)$ . On the contrary  $\text{Jm}$  and  $\text{Im}$  are initialised to one, which means that at the beginning all the coefficients are associated with  $\text{K}(1,1)$ ; this is only a trick to avoid any reference to  $\text{K}(0,0)$ , which is not defined. Anyway, all the elements of  $\text{Cm}$  that are zero are ignored, along with the corresponding values of row and column.

The assemblage is again performed through a loop over elements which proceeds as in `genlin` until the elemental matrix  $\text{Ke}$  is created and the local and global numbering of unknowns is retrieved.

```

Le0=find(Ge>0);                          % local numbering of unknowns
Ie=Ge(Le0);                              % global numbering

```

The new procedure, which replaces the single line

$$\text{K}(\text{Ie}, \text{Ie}) = \text{K}(\text{Ie}, \text{Ie}) + \text{Ke}(\text{Le0}, \text{Le0})$$

in `genlin`, performs a double nested loop over the element unknowns. The first loop, of counter  $i1$ , runs through the lines of  $\text{Ke}$ , which have to fill the  $I$ -th row of the global matrix, with  $\text{I}=\text{Ie}(i1)$ ;  $\text{inRow}=\text{startrow}(\text{I})$  is the starting position of the portion of  $\text{Cm}$  where the coefficients of the  $I$ -th row are stored;  $\text{dimRow}=\text{lenrow}(\text{I})$  is the current number of coefficients in the  $I$ -th row (which is initially set to zero and then incremented during the process). The second loop, of counter  $i2$ , runs through the columns of the of  $\text{Ke}$ , associated to the  $J$ -th global unknown, with  $\text{J}=\text{Ie}(i2)$ .

The assemblage can be simply expressed as  $K(I, J) = K(I, J) + K_e(p, q)$ . In terms of  $C_m$ , however, the procedure is more complicated. The  $I$ -th row of  $K$  is stored in the  $C_m$  list, and more specifically in the  $C_m(\text{inRow}:\text{inRow}+\text{dimRow}-1)$  sublist. In particular, if a contribution to the coefficient  $K(I, J)$  has already been encountered, then it is stored in  $C_m(\text{ipos})$  such that  $(J_m(\text{ipos}))=J$ .

This event is tested with the  $\text{ipos}=\text{find}(J_m(\text{inRow}:\text{inRow}+\text{dimRow}-1)=J)$  command. If the test is negative then the coefficient has to be created; the length of the row in  $\text{lenrow}$  is incremented and  $K_e(p, q)$  is appended at the end of the row. Otherwise  $\text{ipos}$  contains the position of the coefficient and  $K_e(p, q)$  is added to the coefficient  $C_m(\text{inRow}+\text{ipos}-1)$ .

```
for i1=1:length(Ie)
    p=Le0(i1); % local number of unknown
    I=Ie(i1); % global number of unknown
    inRow=startrow(I); % initial position of row I
    dimRow=lenrow(I); % length of row I
    for i2=1:length(Ie) % loop over all columns in row I
        q=Le0(i2); % loc num of unknown in col i2
        J=Ie(i2); % glob num of unknown in col i2
        ipos=find(Jm(inRow:inRow+dimRow-1)==J); % verifies if coeff already exists
        if isempty(ipos), % if not creates a new coefficient
            dimRow=dimRow+1;
            lenrow(I)=dimRow;
            Jm(inRow+dimRow-1)=J;
            Cm(inRow+dimRow-1)=Ke(p,q);
            count_coeff=count_coeff+1;
        else % otherwise adds the contribution
            Cm(inRow+ipos-1)=Cm(inRow+ipos-1)+Ke(p,q);
        end
    end
end
```

Once  $C_m$ ,  $I_m$ ,  $J_m$  have been generated they are associated to the stiffness matrix  $K$  via the command  $K=\text{sparse}(I_m, J_m, C_m)$ .

**Remark.** The improved performance of this assemblage is due to the fact that the search with `find` is limited to a single row of the matrix and is not extended to the whole structure.

## References

---

- Aliabadi, M. H., Rooke, D. P., 1991. *Numerical fracture mechanics*. Computational Mechanics Publications and Kluwer Academic Press.
- Allaire, G., 2007. *Numerical analysis and optimization*. Oxford.
- Babuška, I., Melenk, J. M., 1997. The partition of unity method. *Int. J. Num. Meth. Eng.*, **40**:727–758.
- Babuska, I., Strouboulis, T., 2001. *The finite element method and its reliability*. Oxford Science Publications.
- Ball, J. M., 1977. Convexity conditions and existence theorems in nonlinear elasticity. *Arch. Rat. Mech. Anal.*, **63**:337–403.
- Baranger, T., Andrieux, S., 2006. An optimization approach for the Cauchy problem in linear elasticity. *Structural and Multidisciplinary Optimization*.
- Barsoum, R. S., 1976. On the use of isoparametric finite element in linear fracture mechanics. *Int. J. Num. Meth. Eng.*, **10**:25–37.
- Bathe, K. J., 1996. *Finite element procedures*. Prentice-Hall, Englewood Cliffs, New Jersey, USA.
- Batoz, J. L., Dhatt, G., 1990. *Modélisation des structures par éléments finis*. Hermes, Paris.
- Belytschko, T., Liu, W. K., Moran, B., 2002. *Nonlinear finite elements for continua and structures*. Wiley.
- Bernardi, C., Maday, Y., 1997. Spectral methods. In *Handbook of numerical analysis*, vol. 5, 209–485. Elsevier.
- Besson, J., Cailletaud, G., Chaboche, J. L., Forest, S., 2009. *Non-Linear Mechanics of Materials*. Springer.
- Bettess, P., 1992. *Infinite elements*. Penshaw Press.
- Bonnet, M., Constantinescu, A., 2005. Inverse problems in elasticity. *Inverse Prob.*, **21**:R1–R50.
- Bonnet, M., Frangi, A., 2006. *Analyse des solides déformables par la méthode des éléments finis*. Editions de l'Ecole polytechnique.
- Brezzi, F., Fortin, M., 1991. *Mixed and hybrid element methods*. Springer.
- Broberg, K. B., 1999. *Cracks and fracture*. Academic Press.
- Bui, H. D., 1970. Evolution de la frontière du domaine élastique des métaux avec l'écrouissage plastique et le comportement élastoplastique d'un agrégat de cristaux cubiques. *Mémorial de l'Artillerie Française*, **1**:141–165.

- Carslaw, H. S., Jaeger, J. C., 1959. *Conduction of heat in solids*. Oxford.
- Ciarlet, P. G., 1980. *The finite element method for elliptic problems*. North Holland.
- Ciarlet, P. G., 1988. *Mathematical elasticity. Volume 1 : Three-dimensional elasticity*. North Holland.
- Courant, R., 1943. Variational methods for the solution of problems of equilibrium and vibration. *Bull. Amer. Mathem. Soc.*, **49**:1–23.
- Cuthill, E., McKee, J., 1969. Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 1969 24th ACM national conference*, 157–172. ACM Press.
- Destuynder, P., Djaoua, M., Lescure, S., 1983. Quelques remarques sur la mécanique de la rupture élastique. *J. Mécan. Théor. Appl.*, **2**:113–135.
- Dhatt, G., Touzot, G., 1984. *Une présentation de la méthode des éléments finis*. Maloine S. A., Paris.
- Eringen, A. C., Suhubi, E. S., 1975. *Elastodynamics (vol II - linear theory)*. Academic Press.
- Farhat, C., Roux, F. X., 1994. *Implicit parallel processing in structural mechanics*, 1–124. Computational Mechanics Advances, vol. 2. North-Holland.
- Fraeijs de Veubeke, B., 1965. Displacement and equilibrium models in the finite element method. In O. C. Zienkiewicz, G. S. Holister, editors, *Stress analysis*, 145–197. John Wiley & Sons.
- Freese, C. E., Tracey, D. M., 1976. The natural isoparametric triangle versus collapsed quadrilateral for elastic crack analysis. *Int. J. Fract.*, **12**:767–770.
- Géradin, M., Rixen, D., 1997. *Mechanical vibrations: theory and application to structural dynamics*. Wiley-Blackwell.
- Germain, P., Nguyen, Q. S., Suquet, P., 1983. Continuum thermodynamics. *ASME J. Appl. Mech.*, **50**:1010–1020.
- Girault, V., Raviart, P. A., 1986. *Finite element methods for Navier-Stokes equations: theory and algorithms*. Springer-Verlag.
- Gosselet, P., Rey, C., 2006. Non-overlapping domain decomposition methods in structural mechanics. *Arch. Comp. Meth. Eng.*, **13**:515–572.
- Greenbaum, A., 1997. *Iterative methods for solving linear systems*. SIAM, Philadelphia, USA.
- Hammer, P. C., Marlowe, O. J., Stroud, A. H., 1956. Numerical integration over simplexes and cones. *Mathematical Tables and Other Aids to Computation*, **10**:130–137.
- Hughes, T. J. R., 1987. *The finite element method – linear static and dynamic finite element analysis*. Prentice Hall, Englewood Cliffs, New Jersey, USA.
- Ingraffea, A. R., Wawrzynek, P. A., 2003. Finite element methods for linear elastic fracture mechanics. In R. de Borst, H. Mang, editors, *Comprehensive Structural Integrity*, chapter 3.1. Elsevier.
- Kempeneers, M., Beckers, P., Debongnie, J., 2004. Eléments finis tétraédriques

- pour l'analyse duale. *Rev. Eur. Elements Finis*, **13**:547–558.
- Komatitsch, D., Vilotte, J. P., 1998. The spectral element method: an efficient tool to simulate the seismic response of 2D and 3D geological structures. *Bull. Seism. Soc. Am.*, **88**:368–392.
- Ladevèze, P., 1999. *Nonlinear computational structural mechanics. New approaches and non-incremental methods of calculations*. Mechanical Engineering series. Springer-Verlag.
- Ladevèze, P., Moës, N., Douchin, B., 1999. Constitutive relation error for (visco)plastic finite element analysis with softening. *Comp. Meth. Appl. Mech. Eng.*, **176**:247–264.
- Lax, P. D., Richtmyer, R. D., 1956. Survey of the stability of linear finite difference equations. *Comm Pure Appl. Math.*, **9**:267–293.
- Leblond, J. B., 2002. *Mécanique de la rupture fragile et ductile*. Hermes.
- Lemaître, J., Chaboche, J. L., 1990. *Mechanics of solid materials*. Cambridge University Press.
- Lorentz, E., Wadier, Y., Debruyne, G., 2000. Brittle fracture in a plastic medium: definition of an energy release rate. *C.R. Acad. Sci. Paris, série II*, **328**:657–662. In French, with an extended English abstract.
- Lyness, J. N., Jespersen, D., 1975. Moderate degree symmetric quadrature rules for the triangle. *J. Inst. Math. Appl.*, **15**:19–32.
- Malkus, D. S., Hughes, T. J. R., 1978. Mixed finite element methods — reduced and selective integration techniques: a unification of concepts. *Comp. Meth. Appl. Mech. Eng.*, **15**:63–81.
- Malvern, L. E., 1969. *Introduction to the mechanics of a continuous medium*. Prentice Hall.
- Maugin, G. A., 1992. *The thermomechanics of plasticity and fracture*. Cambridge University Press.
- Moës, N., Dolbow, J., Belytschko, T., 1999. A finite element method for crack growth without remeshing. *Int. J. Num. Meth. Eng.*, **46**:131–150.
- Murat, F., Simon, J., 1976. Sur le contrôle par un domaine géométrique. Rapport de recherche, Laboratoire d'Analyse Numérique de l'université Paris 6.
- Nagtegaal, J. C., Parks, D. M., Rice, J. R., 1974. On numerically accurate finite element solutions in the fully plastic range. *Comp. Meth. Appl. Mech. Eng.*, **4**:153–177.
- Oden, J. T., Reddy, J. N., 1976. *Variational methods in theoretical mechanics*. Springer-Verlag.
- Pommier, S., Gravouil, A., Combescure, A., Moës, N., 2010. *Extended finite element method for crack propagation*. Wiley-ISTE.
- Quarteroni, A., Valli, A., 1994. *Numerical approximation of partial differential equations*. Springer.
- Reddy, J. N., 1986. *Applied functional analysis and variational methods in engineering*. McGraw-Hill.

- Reddy, J. N., 1993. *An introduction to the finite element method*. McGraw-Hill.
- Saad, Y., 2003. *Iterative methods for sparse linear systems*. SIAM.
- Salençon, J., 2001. *Handbook of continuum mechanics (general concepts, thermoelasticity)*. Springer-Verlag.
- Simo, J. C., Hughes, T. J. R., 1998. *Computational Inelasticity*. Springer-Verlag.
- Sokolowski, J., Zolesio, J. P., 1992. *Introduction to shape optimization. Shape sensitivity analysis*, volume 16 of *Springer series in Computational Mathematics*. Springer-Verlag.
- Strang, G., Fix, G. J., 1973. *An analysis of the finite element method*. Prentice-Hall, Englewood Cliffs, New Jersey, USA.
- Stroud, A. H., Secrest, D., 1966. *Gaussian quadrature formulas*. Prentice-Hall.
- Suquet, P., 2004. *Rupture et plasticité*. Cours de l'Ecole Polytechnique, Palaiseau, France.
- Tada, H., Paris, P. C., Irwin, G. R., 2000. *The stress analysis of cracks handbook*. Professional Engineering Publishing.
- Zienkiewicz, O. C., 1989. *The Finite Element Method: Basic Concepts and Linear Applications*. McGraw Hill, fourth edition.
- Zienkiewicz, O. C., Taylor, R. L., 1989. *The Finite Element Method: Basic Concepts and Linear Applications*. McGraw Hill. See also references therein.
- Zienkiewicz, O. C., Taylor, R. L., 2000a. *The Finite Element Method, volume 1: the basis*. Butterworth Heimemann.
- Zienkiewicz, O. C., Taylor, R. L., 2000b. *The Finite Element Method, volume 2: solid mechanics*. Butterworth Heimemann.

# Index

---

- $G - \theta$  method, 222, 234
- $J$  integral, 239
- MATLAB codes
  - RRTM\_VonMises\_2A\_R, 300, 304
  - RR\_VonMises\_2A\_R, 292
  - T3\_2A\_solid\_Ke, 61
  - T6\_2A\_solid\_KTeNL, 262
  - T6\_2A\_solid\_KTePlast, 304
  - bar\_B2\_1D\_wave\_modal, 151
  - dynamics\_tire\_rolling, 270
  - dynamics\_tire, 182
  - dynamics, 174
  - genlin\_fast, 92, 227
  - genlin, 86, 90
  - geomNL, 265
  - heatdiffusion, 129
  - plasticity, 308, 312
  - sphere\_B2\_1S\_diff, 125
  - sphere\_B2\_1S\_solid, 21
  - strip\_plast, 291, 312
  - T3\_2A\_solid\_gtheta, 241
  - bar\_B2\_1D\_wave\_direct, 164
- accuracy
  - Newmark scheme, 164
  - time-stepping scheme, 121
- admissible
  - kinematically, 3
  - statically, 3
- amplification, 172
- assemblage, 57, 70
  - nodal force vector, 72
- axisymmetric problem, 129
- Babuska-Brezzi (LBB) condition, 204
- behavior
  - elastoplastic, 274
  - local integration of —, 281
- bending of a cantilever beam, 207
- best approximation property, 14
- boundary layer, 213
- buckling, 267
- capacity matrix, 117
- compatibility (kinematical), 3
- complementarity condition, 244, 277
- compliance matrix, 15
- conditionally stable (time-stepping scheme),
  - 120, 163, 164, 166
- conductivity, 27, 114
- conductivity matrix, 117, 124
- conformity (mesh), 42
- consistency
  - Newmark scheme, 163
  - time-stepping scheme, 118, 120
- consistent
  - search direction, 249
  - tangent matrix, 252
- constitutive model
  - brittle damage, 246
  - elastic, linear, 4
  - elastic, nonlinear, 254
  - elastoplastic, 247
  - neo-Hookean, 255
  - Saint-Venant-Kirchhoff, 255
  - thermoelastic, linear, 131
- constraint
  - count, 204
  - incompressibility, 189
- contact, 243
- convergence with mesh refinement, 80, 224
- Coulomb (friction law), 244
- crack, 217
  - opening modes, 218
  - propagation, 245
- cross diagonal pattern, 191
- damage, 246
- damping (numerical), 167
- degree of freedom, 11, 48, 70

- deviatoric stress tensor, 275, 285
- diagonalization, 119, 149
- eigenvalue (generalized — problem), 119, 149
- elastic prediction, 283
- elasticity domain, 274
- elastoplastic
  - behavior, 274
- elastoplastic correction, 283
- elastoplasticity, 247
- element, 43
  - force vector, 61, 68
  - integral, 58, 59
  - matrix, 58
  - stiffness matrix, 59, 66–68
- energy
  - complementary, 8, 133
  - minimization, 9, 11, 15, 132
  - potential, 8, 133
  - preservation, 172
- energy release rate, 220
- engineering (Voigt) notation, 58, 299
- equilibrium, 2
  - cracked solid, 218
  - heat conduction, 27
- error in constitutive relation, 7, 133
- explicit (time-stepping scheme), 118, 282
- finite difference, 117, 156, 282
- finite element
  - cube, 41
  - isoparametric, 37, 46
  - line (B2), 36, 39, 63
  - line (B3), 39, 45, 68
  - quadrangle (Q4), 40, 67, 187, 193
  - quadrangle (Q8), 40
  - tetrahedron (P10), 52
  - tetrahedron (P4), 41, 51
  - transition, 53
  - triangle (T3), 40, 60, 62, 225
  - triangle (T6), 40, 50, 66, 187
- flux, 27
  - heat, 114
- formulation
  - variational, 10, 134
  - weak, 6, 28, 115, 134, 146, 256, 281
    - by kinematic data incorporation, 6
    - by reaction elimination, 6
    - linearised, 295
    - linearized, 258
- Fourier law, 114
- free energy, 254
- Galerkin method
  - for the complementary energy, 15
  - for the potential energy, 11
- Gauss points, 64
- generalized
  - forces, 11, 56
- generalized displacements, 11
- global
  - interpolation, 34, 48
  - numbering, 32, 43
  - shape function, 34
- gms (mesh generator), 38
- hardening
  - isotropic
    - linear, 286
- hardening, 275, 314
  - isotropic, 275
  - kinematic, 276
- heat conduction
  - equilibrium, 27
  - transient, 113
- Herrmann functional, 200
- Hertz contact problem, 245
- hourglass mode, 192
- implicit (scheme), 282
- implicit (time-stepping scheme), 118
- incompressibility constraint, 188
- initial condition, 114, 144
- integration
  - numerical, 64
  - reduced, 196, 211
  - selective, 197
- Irwin's formula, 222
- isoparametric element, 37, 46
- jacobian (matrix, determinant), 45, 49
- Lamé potentials, 144
- Lax-Milgram theorem, 10
- local
  - interpolation, 33
  - ordering, 33
  - tangent operator, 296
- local, ordering, 43



- locking, 188, 190
  - shear, 207
- mass matrix, 117, 147
- material toughness, 221
- matrix
  - capacity, 117
  - compliance, 15
  - conductivity, 117
  - mass, 117
  - resolvent, 119
  - stiffness, 11, 56
- mesh, 32
- mixed
  - finite element formulation, 201, 211
  - functional, 199
- mode
  - crack opening, 218
  - hourglass, 192, 197
  - of a structure, 149
  - pressure, 206
  - spurious, 192
- multiplier
  - plastic, 277
- neo-Hookean model, 255
- Newton (method), 248, 257, 280
  - modified: constant search direction, 250
  - modified: secant method, 250
  - quadratic convergence, 250, 252
- Newton-Raphson, *see* Newton
- nodal displacements, 46
- nodal forces, 56, 61
  - thermal, 134
- node, 32
  - double (crack), 223
  - quarter-node element, 228
- normality rule, 276, 284
- numerical damping, 172
- patch test, 48, 180
- penalty approach, 195
- piecewise-linear interpolation, 32
- plastic incompressibility, 279, 308
- polyconvexity, 256
- potential, 27
  - problems, 28
- quadrature, numerical, 64, 319
  - Gauss, one-dimensional, 64
  - Gauss, product, 65
  - Gauss-Hammer for triangles, 65
- quasi-static (assumption), 132, 145
- radial return algorithm, 284, 286
- radiality, deviation from, 288
- Rayleigh quotient, 168
- reduced integration, 196
- residual, 248, 257, 295, 302
- rigid-body motion, infinitesimal, 3
- rod, 151, 164
- Saint-Venant-Kirchhoff model, 255
- scheme
  - central difference, 164
  - Newmark, 158
- semi-discretization in space, 116, 146
- shape function, 33
  - global, 34, 48
- specimen, notched, 312
- stability
  - central difference scheme, 164
  - Newmark scheme, 163, 165
  - of a time-stepping scheme, 118, 120, 159, 161
- stiffness matrix, 11, 56, 73
  - band structure, 74
  - direct solver ( $LDL^T$  factorization), 75
  - iterative solver (conjugate gradient), 76
  - properties, 73
  - skyline, 74
  - storage, Morse, 76
  - storage, skyline, 76
  - tangent, 252
- stiffness matrix
  - tangent, 260
- strain
  - cumulative plastic, 276
  - elastic, 276
  - Green-Lagrange — tensor, 247, 253
  - linearized, 2
  - plastic, 276
- stress
  - crack tip singularity, 220
  - equivalent, 275
  - Piola — tensor, 254
  - Piola-Kirchhoff — tensor, 254
- stress intensity factors, 221
  - évaluation using extrapolation, 224

- evaluation using special-purpose elements, 228
- tangent
  - moduli, 262, 296
  - operator, 257, 258, 295
  - stiffness matrix, 260
    - consistent, 313, 315
    - constant, 307, 313, 315
    - elastoplastic, global, 302
- tensor
  - conductivity, 27
  - dilatation, 255
  - elastic compliance, 5
  - elastic stiffness, 4
  - engineering notation, 58
  - tangent elastic stiffness, 262
  - thermal dilatation, 131
- thermoelasticity (linear), 131
- time step, 117, 156, 280
  - critical, 120, 126, 127, 158, 163, 166, 168, 183
- time-stepping scheme
  - central difference, 156
  - Crank-Nicolson, 121
  - Euler (explicit), 118, 125
  - Euler (implicit), 118, 125
  - plasticity, 280
- tolerance, 78, 252, 307
- toughness, 221
- unconditionally stable (time-stepping scheme), 120, 163, 165
- unstable (time-stepping scheme), 166
- virtual work, 5, 199, 256, 295
- von Mises criterion, 275
- wave, 145
  - compressional, 145
  - shear, 145
- weak formulation, 10
  - discretised, 55
  - dynamics, 146
  - heat conduction, 115
  - linear elasticity, 6
  - nonlinear elasticity, 256
  - potential problem, 28
- yield criterion, 274

