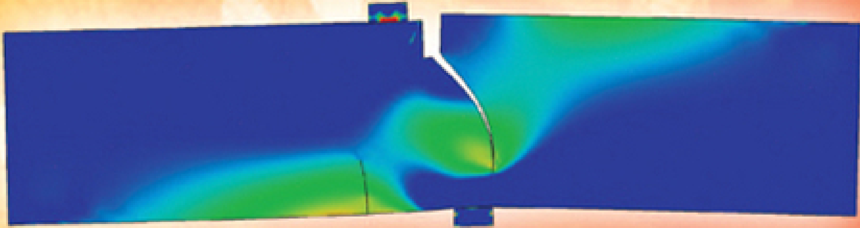
WILEY SERIES IN COMPUTATIONAL MECHANICS

Second Edition

Non-linear Finite Element Analysis of Solids and Structures

René de Borst, Mike A. Crisfield, Joris J. C. Remmers and Clemens V. Verhoosel



WILEY

NON-LINEAR FINITE ELEMENT ANALYSIS OF SOLIDS AND STRUCTURES

WILEY SERIES IN COMPUTATIONAL MECHANICS

Series Advisors:

René de Borst

Perumal Nithiarasu

Tayfun E. Tezduyar

Genki Yagawa

Tarek Zohdi

Introduction to Finite Strain Theory for Continuum Elasto-Plasticity	Hashiguchi and Yamakawa	September 2012
Non-linear Finite Element Analysis of Solids and Structures: Second edition	De Borst, Crisfield, Remmers and Verhoosel	August 2012
An Introduction to Mathematical Modeling: A Course in Mechanics	Oden	November 2011
Computational Mechanics of Discontinua	Munjiza, Knight and Rougier	November 2011
Introduction to Finite Element Analysis: Formulation, Verification and Validation	Szabó and Babuška	March 2011

NON-LINEAR FINITE ELEMENT ANALYSIS OF SOLIDS AND STRUCTURES

SECOND EDITION

René de Borst
School of Engineering, University of Glasgow, UK

Mike A. Crisfield
Imperial College of Science, Technology and Medicine, UK

Joris J.C. Remmers
Eindhoven University of Technology, The Netherlands

Clemens V. Verhoosel
Eindhoven University of Technology, The Netherlands

WILEY

A John Wiley & Sons, Ltd., Publication

This edition first published 2012

Registered office
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, United Kingdom

For details of our global editorial offices, for customer services and for information about how to apply for permission to reuse the copyright material in this book please see our website at www.wiley.com.

Library of Congress Cataloging-in-Publication Data

Non-linear finite element analysis of solids and structures. – 2nd ed. / R. de Borst ... [et al.].
p. cm.
Rev. ed. of: Non-linear finite element analysis of solids and structures / M.A. Crisfield. c1991-c1997. (2 v.)
Includes bibliographical references and index.
ISBN 978-0-470-66644-9 (hardback)
1. Structural analysis (Engineering)–Data processing. 2. Finite element method–Data processing. I. Borst, Ren? de. II. Crisfield, M. A. Non-linear finite element analysis of solids and structures.
TA647.C75 2012
624.1′71–dc23

2012011741

A catalogue record for this book is available from the British Library.

Print ISBN: 9780470666449

Typeset in 10/12pt Times-Roman by Thomson Digital, Noida, India

Contents

PART V ADVANCED DISCRETISATION CONCEPTS

Preface

When the first author was approached by John Wiley & Sons, Ltd to write a new edition of the celebrated two-volume book of Mike Crisfield, *Non-linear Finite Element Analysis of Solids and Structures*, he was initially very hesitant. The task would of course constitute a formidable amount of work. But it would also be impossible to maintain Mike's writing style, a feature which has so much contributed to the success of the books. On the other hand, it would be rewarding to provide the engineering community with a book that is as accessible as possible, that gives a broad introduction into non-linear finite element analysis, with an outlook on the newest developments, and that maintains the engineering spirit which Mike emphasised in his books. This is the philosophy behind this second edition. Indeed, although much has been changed in terms of content, it has been the intention not to change the engineering orientation with an emphasis on practical solutions.

One of the aims of the original two-volume set was to provide the user of advanced non-linear finite element packages with sufficient background knowledge, which is a prerequisite to judiciously handle modern finite element packages. A closely related aim is to make the user of such packages aware of their possibilities, but also of their limitations and pitfalls. Major developments have taken place in computational technology since Mike Crisfield wrote about the danger of the 'black-box syndrome' in the Preface to Volume 1. Therefore, his warning has gained even more strength, and provides a further justification for the publication of a second edition.

Unlike the first edition, the second edition comes as a single volume. The reduction has been achieved by omitting or reducing the discussion on developments now considered to be less central in computational mechanics, by a more compact and focused treatment, and by a removal of all Fortran code from the book. Instead, a small finite element code has been developed, written in Python, which is available through a companion website. The main purpose of the code is to illustrate the models presented in the book, and to show how abstract concepts can be translated into finite element software. To this end, the theory of the book is first transformed into algorithms, mostly listed in boxes that accompany the text. Subsequently, using ideas of literate programming, it is explained how these algorithms have been implemented in the PyFEM code, which contains the basic numerical tools needed to build a finite element code. Some of the solution techniques, element formulations, and material models treated in this book have been added. These tools are used in a series of example programs with increasing complexity.

The book comes in five parts. Part I discusses basic knowledge in mathematics and in continuum mechanics, as well as solution techniques for non-linear problems in static and dynamic analysis, and provides a first introduction into geometrical non-linearity. Some notions and concepts will be familiar, but not all, and the first chapters also serve to provide a common basis for the subsequent parts of the book. Part II contains major chapters on damage, plasticity and time-dependent non-linearities, such as creep. It contains all the material non-linearity that is treated in this book. Shell plasticity forms an exception, since it is treated in Part III, which focuses on structural elements: beams, arches and shells. Starting from a basic shallow arch formulation the discussion extends to cover modern concepts like solid-like shell theories. In Part IV first some additional continuum mechanics is provided that is needed in the remainder of this part, which focuses on large-strain elastic and elastoplastic finite element analysis. Part V, finally, gives an introduction into discretisation concepts that have become popular during the past 20 years: interface elements, discontinuous Galerkin methods, meshless methods, partition-of-unity methods, and isogeometric analysis. Particular reference is made to their potential to solve problems that arise in non-linear analysis, such as locking phenomena, damage and fracture, and non-linear shell analysis.

René de Borst
Joris Remmers
Clemens Verhoosel

Glasgow and Eindhoven

A Personal Note

Like many colleagues and friends in the community I treasure wonderful memories of my meetings and discussions with Mike. I will never forget the times that I visited him at the Transport and Road Research Laboratory, and later, at Imperial College of Science, Technology and Medicine. After a full day of intense discussions on cracking, strain softening, stability and solution techniques we normally went to his home, where Kiki, his wife, joined in and discussions broadened over a good meal.

Mike was a real scientist, and a gentleman. I hope that this Second Edition will properly preserve his legacy, and will help to keep the engineering approach alive in computational mechanics, to which he has so much contributed.

René

Series Preface

The series on *Computational Mechanics* is a conveniently identifiable set of books covering interrelated subjects that have been receiving much attention in recent years and need to have a place in senior undergraduate and graduate school curricula, and in engineering practice. The subjects will cover applications and methods categories. They will range from biomechanics to fluid-structure interactions to multiscale mechanics and from computational geometry to meshfree techniques to parallel and iterative computing methods. Application areas will be across the board in a wide range of industries, including civil, mechanical, aerospace, automotive, environmental and biomedical engineering. Practicing engineers, researchers and software developers at universities, industry and government laboratories, and graduate students will find this book series to be an indispensible source for new engineering approaches, interdisciplinary research, and a comprehensive learning experience in computational mechanics.

Non-linear Finite Element Analysis of Solids and Structures, Second Edition is based on the two original volumes by the late Mike Crisfield, who was a remarkable scholar in computational mechanics. This new edition is a greatly enriched version, written by an author team led by René de Borst, an outstanding scholar in computational mechanics, solids, and structures. The enrichments include the major developments in computational mechanics since the original version was written, such as new numerical discretization techniques, with emphasis on meshless methods and isogeometric analysis. This new edition still retains the “engineering spirit” that was emphasized by the original author, and the algorithmic explanations, which are only part of the enrichments, make it even easier to follow and more valuable in a practical context.

Non-linear Finite Element Analysis of Solids and Structures, Second Edition will serve as an excellent textbook for introductory and advanced courses in non-linear finite element analysis of solids and structures, and will also serve as a very valuable source and guide for research in this field.

Notation

Linear Algebra and Mathematical Operators

$\mathbf{a} \cdot \mathbf{b}$, $a_i b_i$	Dot-product of the vectors **a** and **b**
$\mathbf{a} \otimes \mathbf{b}$, $a_i b_j$	Tensor (or dyadic) product of the vectors **a** and **b**
$\mathbf{a} \times \mathbf{b}$, $e_{ijk} a_j b_k$	Cross-product of the vectors **a** and **b**
$\square^{\mathrm{T}}$	Transpose of matrix $\square$
$\square^{\mathrm{sym}} = (\square)^{\mathrm{sym}}$	Symmetry operator
$\mathrm{tr}(\square)$	Trace of matrix $\square$
$\lVert \square \rVert_2$	Euclidean or L_2-norm of the vector $\square$
δ_{ij}	Kronecker-delta identity
$< \square >$	MacAulay brackets/ramp function
$\nabla \cdot \mathbf{a}$, $\frac{\partial a_{ij}}{\partial x_j}$, $\mathbf{L}^{\mathrm{T}}\mathbf{a}$	Divergence of a (second-order) tensor **a**
$\mathcal{H}(\square)$	Heaviside function
$\delta\square$	Admissable variation of the quantity $\square$

Basic Continuum Mechanics

V	Arbitrary body in the current configuration
S	Boundary of an arbitrary body V in the current configuration
$\mathbf{n}$	Normal vector (to a surface S)
$\mathbf{x} = [x, y, z]^{\mathrm{T}}$	Coordinate in the physical domain
$\mathbf{u} = [u, v, w]^{\mathrm{T}}$	Displacement field
$\gamma_{xy}, \gamma_{xz}, \gamma_{yz}$	Engineering shear strains/elementary square distortions
$\omega_{xy}, \omega_{xz}, \omega_{yz}$	Elementary square rotations
$\mathbf{t}$	Stress vector
$\boldsymbol{\epsilon}$ $[\boldsymbol{\mathcal{E}}]$	Infinitesimal strain tensor [matrix representation]
$\boldsymbol{\sigma}$ $[\boldsymbol{\Sigma}]$	Cauchy stress tensor [matrix representation]
$\boldsymbol{e}$ $[\boldsymbol{E}]$	Deviatoric infinitesimal strain tensor [matrix representation]
$\mathbf{s}$ $[\mathbf{S}]$	Deviatoric stress tensor [matrix representation]
$I_1^{\square}, I_2^{\square}, I_3^{\square}$	Invariants of the tensor $\square$ (Cauchy stress tensor when $\square$ is omitted)
$J_1^{\square}, J_2^{\square}, J_3^{\square}$	Invariants of the tensor $\square$ (deviatoric stress tensor when $\square$ is omitted)
p	Hydrostatic pressure
ϵ_{vol}	Volumetric infinitesimal strain
$\mathbf{T}_{\square}$	Transformation matrix for the tensor $\square$ in Voigt form
$\mathbf{D}$	Tangential stiffness tensor
$\square_{\mathrm{tan}}$	Quantity $\square$ related to the tangent stiffness

$\square^{\mathrm{s}}$	Quantity $\square$ related to the secant stiffness
$\delta\mathcal{W}_{\mathrm{int}}$	Internal virtual work
$\delta\mathcal{W}_{\mathrm{ext}}$	External virtual work
$\mathbf{g}$	Gravity acceleration vector

Elasticity

E	Young's modulus
ν	Poisson's ratio
K	Bulk modulus
λ	Lamé's first parameter
μ, G	Lamé's second parameter/shear modulus
$\mathbf{D}^e$	Elastic stiffness matrix
$\mathbf{C}^e$	Elastic compliance matrix

Finite Element Data Structures

$\square_e$, $\square_{\mathrm{elem}}$	Quantity $\square$ related to the element e
$\mathbf{Z}_e$	Element incidence (or location) matrix
$\boldsymbol{\xi} = [\xi, \eta, \zeta]^{\mathrm{T}}$	Parent element coordinates
$\mathbf{J}$	Jacobian matrix
w_i	Weight factor of parent element integration point i
$\mathbf{h}$, h_i	Finite element shape functions
$\mathbf{H}$	Displacement field interpolation matrix
$\mathbf{B}$	Strain field interpolation matrix
$\mathbf{a}$	Nodal displacement vector
$\mathbf{f}_{\mathrm{int}}$	Internal force vector
$\mathbf{f}_{\mathrm{ext}}$	External force vector
$\mathbf{K}$	Stiffness matrix
$\square_f$	Quantity $\square$ related to an unconstrained degree of freedom
$\square_p$	Quantity $\square$ related to a constrained/prescribed degree of freedom

Geometrically Non-linear Analysis

$\square_0$	Quantity $\square$ related to the reference configuration
$\mathbf{F}$	Deformation gradient
$\boldsymbol{l}$	Velocity gradient
$\mathbf{U}$, $\mathbf{V}$	Right/left pure deformation tensor/stretch tensor
$\mathbf{R}$	Rotation matrix
$\boldsymbol{\Omega}$	Rotation rate matrix
$\boldsymbol{e}$	Linear strain contribution
$\boldsymbol{\eta}$	Quadratic/non-linear strain contribution
$\square_L$	Quantity $\square$ related to linear contributions
$\square_{NL}$	Quantity $\square$ related to non-linear contributions

$\square^{\mathrm{cr}}$	Corotational contribution to quantity $\square$
$\bar{\square}$	Quantity $\square$ related to the corotational coordinate system
$\mathbf{C}$, $\mathbf{B}$	Right/left Cauchy–Green deformation tensor
$\boldsymbol{\gamma}$	Green–Lagrange strain tensor
$\mathbf{p}$	Nominal stress tensor
$\boldsymbol{\kappa}$	Kirchhoff stress tensor
$\boldsymbol{\tau}$	Second Piola–Kirchhoff stress tensor
$\boldsymbol{T}$	Biot stress tensor
$\square_i$	Principal values of the tensor $\square$
λ	Stretch ratio
$\overset{\diamond}{\square}$	Objective derivative of a vector $\square$/Green–Naghdi rate of $\square$
$\overset{\star}{\square}$	Truesdell rate of the tensor $\square$
$\overset{\circ}{\square}$	Jaumann rate of the tensor $\square$
$\boldsymbol{w}$	Spin tensor
$\square_{\mathrm{vol}}$	Volumetric part of quantity $\square$
$\square_{\mathrm{iso}}$, $\tilde{\square}$	Isochoric part of quantity $\square$
$\mathcal{E}$	Total deformation energy
e	Strain energy density
$\mathcal{W}$	Strain energy function
$\mathcal{W}^*$	Volumetric part of the strain energy function
f_p	Deviatoric part of the strain energy function
$\mathbf{T}_\sigma$	Back-transformation matrix
$\square^{\mathrm{JK}}$	Quantity $\square$ related to the Jaumann derivatives of the Kirchhoff stress
$\square^{\mathrm{JC}}$	Quantity $\square$ related to the Jaumann derivatives of the Cauchy stress
$\square^{\mathrm{TK}}$	Quantity $\square$ related to the Truesdell derivatives of the Kirchhoff stress
$\square^{\mathrm{TC}}$	Quantity $\square$ related to the Truesdell derivatives of the Cauchy stress

Incremental Iterative Analysis and Solution Techniques

$\square_0$, $\square^t$	Quantity $\square$ at the previous converged load step
$\Delta\square = \square^{t+\Delta t} - \square^t$	Incremental value of quantity $\square$
$\Delta\square_i$	Approximate incremental value of quantity $\square$ after i iterations
$\mathrm{d}\square_{i+1}$	Correction to the approximate incremental value $\Delta\square_i$
$\mathbf{r}$	Residual vector
$\mathbf{A}$	Constrained degrees of freedom selection matrix
λ	Scalar-valued load parameter
$\hat{\mathbf{f}}_{\mathrm{ext}}$	Unit external force vector
g	Constraint equation
Δl	Path length increment
η	Iterative procedure tolerance
N^t	Number of iterations required for convergence at time t
$\square^I$, $\square^{II}$	Quantity $\square$ related to a two-stage solution procedure
λ_k, $\mathbf{v}_k$	Eigenvalue/vector of the tangential stiffness matrix

Dynamics and Time-dependent Material Models

t	Time
$\dot{\Box}$	First-order temporal derivative
$\ddot{\Box}$	Second-order temporal derivative
ρ	Mass density
$\mathbf{M}$	Mass matrix
$\Box^0$	Quantity $\Box$ at the initial state
$\underline{\Box}$	Quantity $\Box$ evaluated at the time interval mid-point
θ	Generalised mid-point rule parameter
β, γ	Newmark integration parameters
α	HHT α-method integration parameter
$\mathbf{q}$	Pseudo-load vector
τ	Relaxation time
$E(t-\tilde{t})$	Response function
$J(t-\tilde{t})$	Creep function
h	Hardening/softening modulus
s	Strain-rate sensitivity
$\omega^{\max}$	Maximum natural frequency of a system
l	Internal length scale

Damage and Fracture

S_d	Discontinuity surface
$\Box_d$, $\Box_{S_d}$	Quantity $\Box$ related to the discontinuity surface S_d
$\Box_n$, $\Box_s$, $\Box_t$	Normal and shear components of quantity $\Box$
$\Box^+$, $\Box^-$	Quantity $\Box$ related to the positive or negative side of a discontinuity
$[\![\Box]\!] = \Box^+ - \Box^-$	Jump operator
$\mathbf{v}$	Relative displacement across a discontinuity/crack opening
$\mathcal{D}_{S_d}$	Distance function related to the discontinuity surface S_d
$\mathcal{H}_{S_d}$	Heaviside function related to the discontinuity surface S_d
δ_{S_d}	Dirac-delta function related to the discontinuity surface S_d
$\Box_{\text{eff}}$, $\hat{\Box}$	Effective part of quantity $\Box$
ω , $\boldsymbol{\omega}$, $\boldsymbol{\Omega}$	Scalar, second-order tensor, and fourth-order tensor damage parameter
$\underline{\tilde{\Box}}$	Scalar-valued function of the tensor $\underline{\Box}$
$\overline{\Box}$	Spatially averaged scalar-valued function $\tilde{\Box}$
l	Failure process zone length scale
$\psi(\mathbf{x}, \mathbf{y})$	Spatial averaging weight function, $\Psi(\mathbf{x}) = \int_V \psi(\mathbf{x}, \mathbf{y}) \mathrm{d}V$
c_1, c_2, c_3	(Higher-order) gradient damage parameters
f_t	Fracture strength
$\mathcal{G}_c$	Fracture energy
h	Softening modulus
f	Loading–unloading function
κ	History parameter
β	Shear retention factor

μ	Tensile stiffness damage factor
$\underline{\mathbf{A}}$	Acoustic tensor
$\square + \sum \psi_l(\mathbf{x})\tilde{\square}_l$	Partition-of-unity decomposition of the quantity $\square$
$\boldsymbol{\lambda}$	Lagrange multiplier
$\square^{\mathrm{con}}$	Concrete part of the quantity $\square$
$\square^{\mathrm{cr}}$	Cracking part of the quantity $\square$
$\square^{\mathrm{re}}$	Quantity $\square$ related to a reinforcement
$\square^{\mathrm{rc}}$	Quantity $\square$ related to reinforced concrete
$\square^{\mathrm{ia}}$	Quantity $\square$ related to concrete-reinforcement interaction

Plasticity

$\square^{\mathrm{p}}$	Plastic part of quantity $\square$
$\square^{\mathrm{e}}$	Elastic part of quantity $\square$
ψ	Dilatancy angle
ϕ	Friction angle
λ	Plastic multiplier
$\mathbf{m}$	Plastic flow direction
f	Yield function
g	Plastic potential function
$\mathbf{n}$	Yield surface normal vector
τ	Shear stress
γ	Shear deformation
c	Adhesion coefficient
$\bar{\sigma}$	Yield strength
h	Hardening modulus
κ, $\boldsymbol{\kappa}$	Scalar hardening parameter/vector of hardening (history) parameters
$\square_e$	Quantity $\square$ related to the trial (elastic) step
$\square_c$	Quantity $\square$ related to the corrector step
$r_\square$, $\mathbf{r}_\square$	Residuals for local scalar- and vector-valued quantities $\square$
$\mathbf{A}$	Stress residual tangent matrix
$\mathbf{H}$	Pseudo-elastic stiffness matrix
q	Modified J_2 stress invariant
$\mathbf{P}$	Projection matrix for the modified J_2 stress
$\mathbf{Q}$	Projection matrix for strain hardening
$\boldsymbol{\pi}$	Projection vector for the hydrostatic pressure
$\underline{\boldsymbol{\alpha}}$	Back-stress tensor
$\square$	Quantity $\square$ represented in the principal stress coordinate system
$\square_{\mathrm{m}}$	Quantity $\square$ evaluated at the time interval mid-point
α, k, β	Drucker–Prager model parameters
θ	Lode's angle
M, p_c	Cam-clay model parameters
κ^*	Modified swelling index
λ^*	Modified compression index
ϕ^*	Void volume fraction

Structural Members

$\square_0$	Quantity $\square$ in the undeformed state
$\square_l$	Quantity $\square$ related to the centre line/mid plane
ξ, η	Centre line/mid plane parametric coordinates
ζ	Out-of-plane parametric coordinate
l	Length of the structural member
h, t	Thickness of the structural member
b	Width of the structural member
A	Cross-sectional area of the structural member
I	Moment of inertia of the structural member
$\mathbf{d}$	Director
ϕ, ψ	Rotations of the structural member
$\theta, \boldsymbol{\theta}, \boldsymbol{\Theta}$	Centre line/mid plane rotations
$\chi, \boldsymbol{\chi}$	Centre line/mid plane curvature
N	Normal force
M	Bending moment
G	Shear force
$\mathbf{a}$	Nodal variables related to the centre line/mid plane deformation
$\mathbf{w}$	Nodal variables related to the out-of-plane deformation
$\boldsymbol{\theta}$	Nodal variables related to the centre line/mid plane rotations
$\square_a$	Quantity $\square$ related to the centre line/mid plane nodal variables
$\square_w$	Quantity $\square$ related to the out-of-plane deformation nodal variables
$\square_\theta$	Quantity $\square$ related to the centre line/mid plane rotation nodal variables
$\square_c$	Quantity $\square$ related to an hierarchical mid-side node
k	Shear stiffness correction factor
w	Solid-like shell internal stretch parameter

Isogeometric Analysis

d_p	Dimension of the parameter domain
d_s	Dimension of the physical domain
$\widehat{V}$	Parameter domain
$\boldsymbol{\xi} = [\xi, \eta, \zeta]^{\mathrm{T}}$	Parametric coordinate
$\boldsymbol{\Xi}_\square$	Knot vector corresponding to $\square$
$\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]^{\mathrm{T}}$	Control net/control points
W_i	Control point weights
$w(\boldsymbol{\xi})$	Weight function
$\mathbf{h}, h_i$	B-spline basis functions
$\mathbf{r}, r_i$	NURBS basis functions
$\mathbf{B}$	Bernstein basis functions
$\boldsymbol{\mathcal{C}}_e$	Element extraction operator

About the Code

A number of models and algorithms that are discussed in this book, have been implemented in a small finite element code named PyFEM, which is available for a free download from the website that accompanies this book. The code has been written in Python, an object-oriented, interpreted, and interactive programming language. Its clear syntax allows for the development of small, yet powerful programs. A wide range of Python packages are available, which are dedicated towards numerical simulations. Many numerical libraries and software tools have been equipped with a Python interface and can be integrated within a Python program seamlessly.

In PyFEM we restrict ourselves to the use of the packages `NumPy`, `SciPy` and `Matplotlib`. The `NumPy` package contains array objects and a collection of linear algebra operations. The `SciPy` package is an extension to this package and contains additional linear algebra tools, such as solvers and sparse arrays. The `Matplotlib` package allows the user to make graphs and plots. Python and the three aforementioned packages are standard components of most Linux distributions. The most recent versions of Python for various Windows operating systems and Mac OS X can be downloaded from `www.python.org`.

The PyFEM code contains the basic numerical tools which are needed to build a finite element code. These tools are used in a series of example programs with increasing complexity. The examples that illustrate the numerical techniques presented in the first chapters of this book are basically small scripts that perform a single numerical operation and do not require an input file. These small scripts are developed further, and finally result in a general finite element program which will be presented in Chapter 4: `PyFEM.py`. This program can be considered as a stand-alone program that can carry out a variety of simulations with different element formulations and material models. In the remaining parts of this book the implementation of some solvers, elements and material models is discussed in more detail.

The directory structure of PyFEM is shown in Figure 1. The package contains the following files and directories:

– `PyFEM.py` is the main program. Executing this program requires an input file with the extension `.pro`.
– The directory `doc` contains installation notes and a short user manual of the code.
– The directory `examples` contains a number of small example programs and input files, which are stored in subdirectories `ch01`, `ch02` etc., which refer to the corresponding chapters of this book for easy reference. Some of the programs and files in these directories are discussed in detail.
– The actual finite element tools are stored in the directory `pyfem`. This directory consists of six subdirectories, including `elements`, which contains element implementations, `solvers`, which contains the solvers and `materials`, in which the material formula-

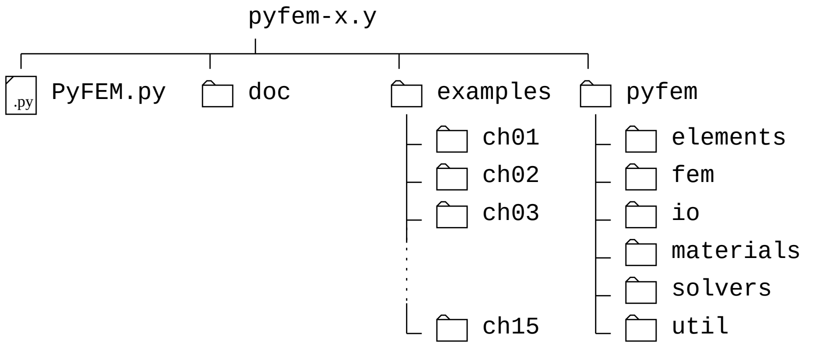


Figure 1 Directory structure of the PyFEM code. The root directory is called `pyfem-x.y`, where `x.y` indicates the version number of the code

tions are stored. A selection of files in these directories is elaborated in the book. The other three directories, `io`, `fem` and `util` contain input parsers and output writers and various finite element utility functions such as a shape function utility, which will be discussed in Chapter 2.

PyFEM is an open source code and is intended for educational and scientific use. It does not contain comprehensive libraries, e.g. of material models, but it has been designed so that it is relatively easy to implement other solvers, elements, and material models, for which the theory and the algorithmic details can be found in this book. A concise user's guide how to implement these can be found at the website.

Instead of giving full listings of classes and functions, we will use a notation that is inspired by literate programming. The main idea behind literate programming is to present a code in such a way that it can be understood by humans and by computers. An important feature of literate programming is that parts of the source code are presented as small fragments, allowing for a detailed discussion of the code. A short overview of the notation, including a system to refer to other fragments, is given in Figure 2.

The name of the fragment. The '≡' symbol indicates that this is a new fragment. When the '+≡' symbol is used, it augments a fragment that has been defined before.

In the case of a new fragment, this number refers to the page where this fragment was announced. When this fragment augments an existing fragment, this number refers to the page where the previous part of the fragment was presented.

⟨*Solution procedure* ⟩≡ 65

```
K = zeros( shape=( totDof , totDof ) )

for elemNodes in elems:
  elemDofs    = getDofs(elemNodes)                    67
  elemCoords = coords[elemNodes,:]

  sData = getElemShapeData( elemCoords )              49

  for iData in sData:
    ⟨Calculate integration point contribution 69⟩
```

Inclusion of another fragment in the code. The number behind the fragment name refers to the page on which the fragment has been presented. If this number is missing, the fragment is not discussed in the book.

Reference to the page on which the function or class that is mentioned in the same line in the code is described. In this case, the function `getElemShapeData` has been explained on page 49.

Figure 2 Example of a code fragment with the nomenclature and references to other code fragments

Part I

Basic Concepts and Solution Techniques

1

Preliminaries

This chapter is primarily intended to familiarise the reader with the notation we have adopted throughout this book and to refresh some of the required background in mathematics, especially linear algebra, and applied mechanics. As regards notation, we remark that most developments have been carried out using matrix-vector notation, and tensor notation is less often needed, either in indicial form or in direct form. For the benefit of readers who are less familiar with tensor notation, we have added a small section on this topic. But, first, we will give an example of non-linearity in a structural member. This example involving a simple truss element can be solved analytically, and serves well to illustrate the various procedures that are described in this book for capturing non-linear phenomena in solids and structures, and for accurately solving the ensuing initial/boundary-value problems.

1.1 A Simple Example of Non-linear Behaviour

Many features of solution techniques can be demonstrated for simple truss structures, possibly in combination with springs, where the non-linear structural behaviour can stem from geometrical as well as from material non-linearities. In this section we shall assume that the displacements and rotations can be arbitrarily large, but that the strains remain small, say less than 5%. This limitation will be dropped in Part IV of this book, where the extension will be made to large elastic and inelastic strains.

We consider the shallow truss structure of Figure 1.1. From elementary equilibrium considerations in the deformed configuration, the following expression for the force can be deduced that acts in a symmetric half of the shallow truss:

$$F_{\text{int}} = -A\sigma \sin\phi - F_s \tag{1.1}$$

where σ is the axial stress in the member, F_s is half of the force in the spring, and ϕ is the angle of the truss member with the horizontal plane in the deformed configuration. Owing to the small-strain assumption, the difference between the cross section in the current configuration, A, and that in the original configuration, A_0, is negligible. For the same reason, the difference

Non-linear Finite Element Analysis of Solids and Structures, Second Edition.
René de Borst, Mike A. Crisfield, Joris J.C. Remmers and Clemens V. Verhoosel.

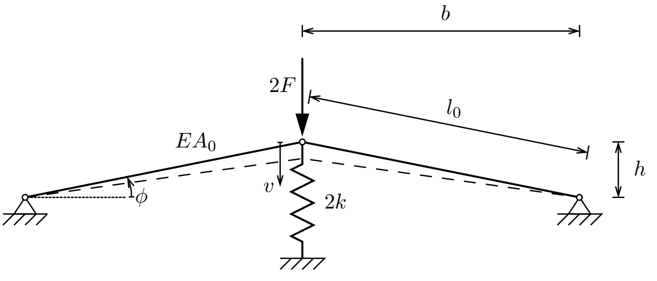


Figure 1.1 Plane shallow truss structure

between the length of the bar in the original configuration,

$$\ell_0 = \sqrt{b^2 + h^2} \tag{1.2}$$

and that in the current configuration,

$$\ell = \sqrt{b^2 + (h - v)^2} \tag{1.3}$$

can be neglected in the denominator of the expression for the strain:

$$\epsilon = \frac{\ell - \ell_0}{\ell_0} \tag{1.4}$$

or when computing the inclination angle ϕ:

$$\sin\phi = \frac{h - v}{\ell} \approx \frac{h - v}{\ell_0} \tag{1.5}$$

The dimensions b and h are defined in Figure 1.1. The vertical displacement v is taken positive in the downward sense. For half of the force in the spring we have

$$F_s = -kv \tag{1.6}$$

with k the spring stiffness, and the axial stress in the bar reads:

$$\sigma = E\epsilon \tag{1.7}$$

with E the Young's modulus. Substitution of the expressions for the stress σ, the force in the spring F_s and the angle ϕ into the equilibrium condition (1.1) yields:

$$F_{\text{int}}(v) = -EA_0 \sin\phi \frac{\ell - \ell_0}{\ell_0} + kv \tag{1.8}$$

Equation (1.8) expresses the internal force that acts in the structure as a non-linear function of the vertical displacement v. Normally, the external force at time $t + \Delta t$, $F_{\text{ext}}^{t+\Delta t}$, is given. The displacement v must then be computed such that

$$F_{\text{ext}}^{t+\Delta t} - F_{\text{int}}^{t+\Delta t} = 0 \tag{1.9}$$