Class Notes: Introduction to the Finite Element Methods

Juan David Gómez Cataño
jgomezc1@eafit.edu.co
Nicolás Guarín-Zapata
nicoguarin@gmail.com

Grupo de Investigación en Mecánica Aplicada
Civil Engineering Department
School of Engineering
Universidad EAFIT
Medellín, Colombia
2015

Contents

Todo list

Summary

Class Notes for the Course Introduction to the Finite Element Methods

Keywords: Scientific Computing, Computational Mechanics, Finite Element Methods, Numerical Methods.

Chapter 1

Introduction

In an introductory graduate course in the Finite Element Method (FEM), the purpose is to develop a basic understanding of the discrete or numerical strategy of Finite Element-(FE) Algorithms when a closed form solution to a Boundary Value Problem (BVP) is not possible due to complexities existing probably: in the boundary conditions, material behavior or in the kinematic description used in the model. In such a course, in order to master and identify the main mathematical and algorithmic aspects of the method at the beginners level, the studied problem is kept lineal. In this sense 4 key aspects make the core of the introductory course (at least for the case of the linearized theory of elasticity boundary value problem):

- Formulation and identification of the strong form of the Boundary Value Problem (BVP) where the continuum mechanics governing equations are revisited and particularized to the case of linear elastic material behavior. The BVP is completed by identifying the correct prescription of boundary conditions for a well posed mathematical problem.
- Formulation and identification of the weak form of the BVP where the governing equations and natural boundary conditions representing equilibrium at the material point level are replaced by an equivalent but weaker form of equilibrium now valid at the global level. In the case of a Continuum Mechanics problem this so-called weak form can be shown to be equivalent to the principle of virtual power and lends itself for the partition of the problem into sub-domains or finite size elements.
- Introduction of the idea of discretization dividing the problem into subdomains and using interpolation theory within each subdomain-In the context of the FE method this is the subject of shape functions. Once the computational specimen has been divided into subdomains (or a mesh of finite elements) the selected primary variable is approximated within each element via interpolation of the known response at selected predefined points or nodes.
- Computational aspects grouped also into 5 points:
  - Formulation of elemental matrices in the physical space.

- Formulation of elemental matrices in the natural domain-Isoparametric transformation.
- Numerical integration: Gauss quadrature.
- Assembly of system matrices and imposition of boundary conditions.
- Solution of the discrete equilibrium statement and calculation of elemental results.

These 4 key points are perfectly well documented in numerous nicely written textbooks and it could be argued that is not even worth to register for an introductory course since a moderately dedicated student can accomplish the task via self-study. Unfortunately, linearity is scarce–although useful to grasp the basic understanding of the problem–and the real world is full of non-linear behavior and understanding the needed algorithms can be easily justified. In this Advanced Finite Element Methods course the goal is then to understand the basic aspects of non-linear finite element analysis. Like in the linear case there is also a vast amount of literature for the non-linear problem. However, in the non-linear case the kinematic problem itself may take different routes leading to a wide variety of FEM formulations that will difficult a self-study strategy. Considering the above this brief set of class notes is intended as a guide for self-study and more important represents a help towards the implementation of the algorithm for the consideration of Material and Geometric Non-linearities in Solids.

Check name of the "course"

The basic reference is Professor Bathe's textbook [1] but we will also follow closely ABAQUS Theory Manual [2]. ABAQUS is a robust, multi-physics oriented commercially available finite element analysis tool. Its strength resides in the effective non-linear algorithms and on the capability of taking user subroutines written in good old Fortran or in C++. The possibility of implementing user subroutines makes it a very powerful research tool. In the particular case of a stress/displacement analysis problem with non-linearities these are considered through the kinematics contribution or through the material contribution. In the first case the non-linear behavior must be considered at the element level while in the second it corresponds to the response of a material point which in the context of the FEM algorithm corresponds to an integration point. In ABAQUS those two sources of non-linearity can be independently controlled by the user via user subroutines UEL and UMAT. In both cases the non-linearity is primarily solved by the classical Newton-Raphson scheme and that will be the approached followed herein. Although the notes are mainly written for an advanced course the specific problem of a linear solid usually studied in the introductory course can be derived like a particular case of the most general non-linear algorithm.

The current set of Class Notes is organized as follows. First and since we will be dealing with history dependent non-linear problems the most powerful (at least when it works) solution algorithm, namely the Newton-Raphson iteration is studied. The technique is first illustrated for the simple 1D-case and then generalized into the multi-degree of freedom system. In both cases pseudo-codes will be presented preparing the way for the Finite Element Algorithm. The presentation however is not exhaustive in mathematical terms and the reader is referred to excellent treatments like Burden [3] and Press et al. [4] for the mathematical aspects of the Newton method.

In the next section the briefly introduced Newton-Raphson technique is contextualized to the case of a system of equations representing equilibrium between internal and external forces as

typically found in a finite element model. Moreover, the non-linearities come into play through a dependence of the internal forces into displacements. At this stage the details of the formulation of the finite element equations via discretization into nodal variables is not presented but emphasis is laid down into the solution algorithm. Interest is then given to the particular form taken by the Newton-Raphson algorithm into the commercial finite element code ABAQUS. That code can be used as a powerful non-linear equation solver where the coefficient matrix and the excitation can be directly controlled by the user. Moreover the solver can be used into a multi-physics context in terms of generalized forces and fluxes. In the particular case of the stress analysis finite element method the user can control the elemental contribution to the coefficient matrix and the contribution of each material point to obtain that element contribution. This is achieved through the so-called user subroutines UEL for element and UMAT for material. Having introduced the Newton-Raphson method the notes concentrate next on general discretization aspects starting from the physical strong form of the equations in the deformed configuration and passing to an arbitrary weak form in the reference configuration. The resulting algorithm is therefore a Total Lagrangian (TL) method. Once the general equations are introduced a particular work conjugate stress-strain pair is chosen and the discrete equations, including kinematic interpolators, are described.

Notation

In a non-linear algorithm the bookkeeping is involved since we have to simultaneously record 4 different fields as follows:

- The physical fields in terms of tensorial descriptions.
- The time field since the problem is solved incrementally and time may appear as an artificial chronological variable or as a real quantity in a dynamic problem.
- The interpolation field. Since all the involved variables will be interpolated we will need to keep track of the way this interpolation is being performed.
- The iterations field needed in the solution of the non-linear problem.

In order to keep this bookkeeping simple we use the following indicial notation with subscripts and superscripts

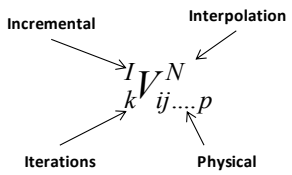


Figure 1.1. General notation to study non-linear finite element problems

Capital superscripts will be reserved for the incremental time description and for the interpolation scheme. For instance an expression like refers to the time instant while refers to . Similarly a variable refers to interpolation over the node. Left and right subscripts will be used to make reference to the iteration being performed and the order of the tensor variable. For instance refers to a second order tensor corresponding to the iteration.

Review notation.

Chapter 2

The Finite Element Method

2.1 Brief review of the linearized theory of elasticity

The balance of momentum, stress symmetry and traction definition on the boundary are given by

$$\begin{aligned} &\sigma_{ij,j} + f_i = 0 \quad \forall\ \vec{x} \in V \\ &\sigma_{ij} = \sigma_{ji} \\ &t_i^{\hat{n}} = \sigma_{ij}\hat{n}_{ij} \quad \forall\ \vec{x} \in S \end{aligned} \quad . \tag{2.1}$$

The simplest constitutive equation in solid mechanics is to consider a linear dependence between stress and strains, and it is called Hooke's law[1]

$$\sigma_{ij} = 2\mu\varepsilon_{ij} + \lambda\varepsilon_{kk}\delta_{ij} \quad . \tag{2.2}$$

Strain-displacement (kinematic relationship)

$$\varepsilon_{ij} = \frac{1}{2}\left(u_{i,j} + u_{j,i}\right) \tag{2.3}$$

eq. (2.3) in eq. (2.2) and the result in eq. (2.1) yields

$$\begin{aligned} \sigma_{ij} &= \mu\left(u_{i,j} + u_{j,i}\right) + \lambda u_{k,k}\delta_{ij} \\ \sigma_{ij,j} &= \mu u_{i,jj} + \mu u_{j,ij} + \lambda u_{k,kj}\delta_{ij} \\ \sigma_{ij,j} &= \mu u_{i,jj} + (\lambda + \mu)\, u_{j,ij} \end{aligned}$$

from which

$$\begin{aligned} &(\lambda + \mu)\, u_{j,ij} + \mu u_{i,jj} + f_i = 0 \quad \forall \vec{x} \in V \\ &t_i^{\hat{n}} = \sigma_{ij}\hat{n}_{ij} \quad \forall\ \vec{x} \in S_t \\ &u_i = \bar{u}_i \quad \forall \vec{x} \in S_u \end{aligned} \tag{2.4}$$

[1] Despite the name of *law* used, this relation is not always valid, but is a good approximation for small strains.

and where $S_t \cup S_u = S$ and $S_t \cap S_u = \emptyset$.

Notice that

$$t_i^{\hat{n}} == \mu \left(u_{i,j} + u_{j,i} \right) \hat{n}_j + \lambda u_{k,k} \delta_{ij} \hat{n}_j \ \ ,$$

is really a Neumann boundary condition on u_i.

Simple wedge under self-equilibrated loads

Consider the double wedge of side ℓ and internal angle 2ϕ shown in fig. 2.1. It is assumed to be contained in the $X - Y$ plane, with loading conditions satisfying a plane strain (or plane stress) idealization. The material is elastic with Lame constants λ and μ. The wedge is loaded by uniform tractions of intensity S applied over its four faces in such a way that the wedge is self-equilibrated. We wish to find the closed-form elasticity solution for the stress, strain and displacement fields throughout the problem domain.

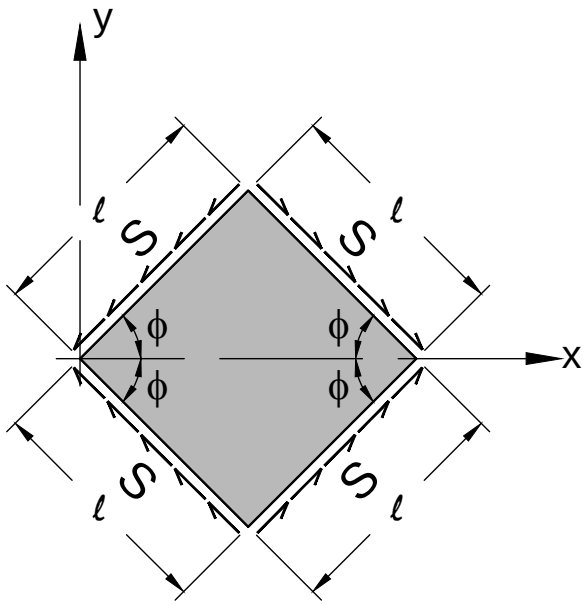


Figure 2.1. 2D Self-equilibrated wedge.

Under plane strain conditions the general 3D stress equilibrium equations reduce to:

$$\begin{aligned} \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} &= 0 \\ \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} &= 0 \end{aligned} \tag{2.5}$$

with strain given by

$$\begin{aligned} \epsilon_{xx} &= \frac{\partial u}{\partial x} \\ \epsilon_{yy} &= \frac{\partial v}{\partial y} \\ \gamma_{xy} &= \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{aligned} \tag{2.6}$$

where u and v are the horizontal and vertical displacements respectively.

Stress field

The stress field can be obtained by simple inspection after identifying the traction boundary conditions. In the current problem these conditions correspond to boundary conditions are of the second type given by

$$t_i = \sigma_{ij}\hat{n}_j \tag{2.7}$$

and prescribed over the inclined surfaces of the wedge.

Simple equilibrium considerations yield

$$\begin{aligned} \sum F_x = 0 &\longrightarrow -\ell S\cos(\phi) + \sigma_{xx}\ell\sin(\phi) = 0 \\ \sum F_y = 0 &\longrightarrow -\ell S\sin(\phi) - \sigma_{yy}\ell\cos(\phi) = 0 \end{aligned}$$

from which the following stress solution results:

$$\begin{aligned} \sigma_{xx} &= S\cot(\phi) \\ \sigma_{xx} &= -S\tan(\phi) \\ \tau_{xy} &= 0 \end{aligned} \tag{2.8}$$

with the condition $\tau_{xy} = 0$ due to symmetry.

Let $\hat{n}^1$, $\hat{n}^2$, $\hat{n}^3$, $\hat{n}^4$ be the outward normal vectors to the exposed faces of the wedge and given by

$$\begin{aligned} \hat{n}^1 &= -\sin(\phi)\hat{e}_x + \cos(\phi)\hat{e}_y \\ \hat{n}^2 &= -\sin(\phi)\hat{e}_x - \cos(\phi)\hat{e}_y \\ \hat{n}^3 &= +\sin(\phi)\hat{e}_x + \cos(\phi)\hat{e}_y \\ \hat{n}^4 &= +\sin(\phi)\hat{e}_x - \cos(\phi)\hat{e}_y \ . \end{aligned}$$

The components of the traction vector follow directly like

$$t_i = \sigma_{ij}\hat{n}_j$$

then over the face with normal $\hat{n}^1$ we have

$$\begin{aligned} t_x &= -S\cos(\phi) \\ t_y &= -S\sin(\phi) \end{aligned}$$

similarly, over the face with normal $\hat{n}^2$

$$\begin{aligned} t_x &= -S\cos(\phi) \\ t_y &= +S\sin(\phi) \end{aligned}$$

over the face with normal $\hat{n}^3$

$$\begin{aligned} t_x &= +S\cos(\phi) \\ t_y &= -S\sin(\phi) \end{aligned}$$

and finally, over the face with normal $\hat{n}^4$;

$$\begin{aligned} t_x &= +S\cos(\phi) \\ t_y &= +S\sin(\phi) \ . \end{aligned}$$

Strain field

Using the stress field found in eq. (2.8) in the constitutive law for a linear elastic material under plane strain conditions gives us

$$\begin{aligned} \epsilon_{xx} &= +\frac{S}{E}\left[\cot(\phi) + \nu\tan(\phi)\right] = +\frac{S}{E}K_1(\nu,\phi) \\ \epsilon_{yy} &= -\frac{S}{E}\left[\tan(\phi) + \nu\cot(\phi)\right] = -\frac{S}{E}K_2(\nu,\phi) \\ \gamma_{xy} &= 0. \end{aligned} \tag{2.9}$$

Displacement field

To determine the displacement field we use the fact that $\omega_{xy} = 0$ from which:

$$\begin{aligned} u &= +\frac{S}{E}K_1(\nu,\phi)x + A \\ v &= -\frac{S}{E}K_2(\nu,\phi)y + B \end{aligned}$$

where A and B are integration constants.

From the condition $u = 0$ at $x = l\cos(\phi)$ we have that $A = -\frac{S}{E}K_1(\nu,\phi)l\cos(\phi)$ then it follows that

$$u = \frac{S}{E}K_1(\nu,\phi)(x - l\cos(\phi)).$$

Similarly, from the condition $v = 0$ at $y = 0$ we have that $B = 0$ from which

$$v = -\frac{S}{E} K_2(\nu, \phi) y$$

2.2 Formal definition of strong and weak forms

Strong form

The strong form of the problem, denoted by $\{S\}$ reads:

Given f_i, $t_i^{\hat{n}}$ and $\bar{u}_i$ find $u_i : V \to \Re$ such:

$$\begin{aligned} &(\lambda + \mu)\, u_{j,ij} + \mu u_{i,jj} + f_i = 0 \quad \forall \vec{x} \in V \\ &t_i^{\hat{n}} = \sigma_{ij} \hat{n}_{ij} \quad \text{for } \vec{x} \text{ in } S_t \\ &u_i = \bar{u}_i \quad \text{for } \vec{x} \text{ in } S_u \end{aligned} \tag{2.10}$$

In eq. (2.10) the boundary conditions specified by the traction vector $t_i^{\hat{n}}$ correspond to the natural boundary conditions, while those specified in terms of the displacements vector $\bar{u}_i$ represent the essential boundary conditions.

- We are interested in developing methods to obtain approximate solutions to $\{S\}$.
- The FEM is formulated starting from a statement equivalent to $\{S\}$ in which we use trial functions until certain prescribed conditions are met.
- We will look for solutions u_i subject to the following conditions:

$$\begin{aligned} u_i &= \bar{u}_i \\ \int_S \left(\frac{\partial u_i}{\partial x_j} \right)^2 dS &< \infty \end{aligned}$$

The first condition corresponds to the satisfaction of the essential boundary condition, while the second corresponds to the functions being square integrable. The space of functions satisfying the above two conditions is denoted by ς and formally defined like;

$$\varsigma = \{ u_i \,|\, u_i \in H, u_i = \bar{u}_i \quad in \quad S_u \}$$

.

On the other hand, in order to validate the introduced trial functions we also need testing (or weighting and distribution) functions w_i. Such functions are arbitrary apart form having to satisfy the following conditions:

$$w_i = 0 \quad in \quad S_u$$

$$\int_S \left(\frac{\partial w_i}{\partial x_j}\right)^2 dS < \infty$$

In what follows we formally denote the space of these functions by V and formally define it like

$$V = \{w_i \,|w_i \in H, u_i = w_i = 0 \quad in \quad S_u\}$$

Weak form

Having defined the strong form $\{S\}$ and the spaces ς and V we will state an alternative form (equivalent to $\{S\}$) but with advantages when thinking about a numerical solution. Since in that form the continuity requirements of the trial functions is weaker than in the strong form such formulation is commonly referred to like the weak form $\{W\}$ and defined as follows.

Given f_i, $t_i^{\hat{n}}$ and $\bar{u}_i$ find $u_i : V \to \Re$ and $\forall w_i \in V$ such:

$$\int_V \sigma_{ij,j} w_{i,j} dV - \int_V f_i w_i dV - \int_{S_t} t_i^{\hat{n}} w_i dS = 0$$

Proof 1:

Let $u_i \in \varsigma$ be a solution to $\{S\}$ and let $w_i \in V$. Forming the inner product of the equilibrium statement with w_i and forcing the integral over the domain to be zero we have:

$$\int_V \left(\sigma_{ij,j} + f_i\right) w_i dV = 0$$

expanding the terms in the integrand yields;

$$\int_V \sigma_{ij,j} w_i dV + \int_V f_i w_i dV = 0$$

Now integrating by parts the first term on the left we have:

$$-\int_V w_{i,j}\sigma_{ij} dV + \int_S \sigma_{ij}\hat{n}_j w_i dS + \int_V w_i f_i dV = 0$$

since $w_i \in V$ it follows that $w_i = 0$ in S_u from which

$$\int_V \sigma_{ij,j} w_{i,j} dV - \int_V f_i w_i dV - \int_{S_t} t_i^{\hat{n}} w_i dS = 0 \tag{2.11}$$

Now, considering that u_i is solution of the strong form $\{S\}$ it must satisfy $u_i = \bar{u}_i \quad in \quad S_u$ and as a result $u_i \in \varsigma$. On the other hand, since u_i satisfies eq. (2.11) $\forall w_i \in V$ we have that u_i satisfies the definition of weak solution specified in $\{W\}$.

Proof 2:

Let u_i be a solution of $\{W\}$ and thus $u_i \in \varsigma$ which means that:

$$u_i = \bar{u}_i \quad in \quad S_u$$

and that it satisfies

$$\int_V \sigma_{ij} w_{i,j} dV - \int_V f_i w_i dV - \int_{S_t} t_i^n w_i dS = 0$$

integrating by parts,

$$-\int_V \sigma_{ij,j} w_i dV + \int_S \sigma_{ij} n_j w_i - \int_V f_i w_i dV - \int_{S_t} t_i^n w_i dS = 0$$

Since $w_i \in V$ we have that $w_i = 0$ in S_u and therefore:

$$\int_V w_i \left(\sigma_{ij,j} + f_i\right) dV + \int_{S_t} w_i \left(\sigma_{ij} n_j - t_i^n\right) dS = 0$$

from which:

$$\begin{aligned} &\sigma_{ij,j} + f_i = 0 \quad \vec{x} \in V \\ &t_i^n = \sigma_{ij} n_j \quad for \quad \vec{x} \quad in \quad S_t \\ &u_i = \bar{u}_i \quad for \quad \vec{x} \quad in \quad S_u \end{aligned} \tag{2.12}$$

2.2.1 The principle of virtual work-Discretization into local interpolating functions

We now discretize the principle of virtual work repeated below for completeness:

$$\int_V \sigma_{ij} \delta u_{i,j} dV - \int_V f_i \delta u_i dV - \int_{S_t} t_i^n \delta u_i dS = 0. \tag{2.13}$$

For that purpose we will divide the complete domain V into N-finite non-overlapping subdomains over each one of which we will approximate the solution in terms of local interpolating functions. Since the PVW (or weak form of the BVP) has been casted into an integral representation, it is possible to build the total integral considering the contribution of the N-sub-domains as follows;

$$\sum_{e=1}^{NEL} \int_{V^e} \sigma_{ij} \delta u_{i,j} dV^e - \int_V f_i \delta u_i dV^e - \int_{S_t} t_i^n \delta u_i dS^e = 0 \tag{2.14}$$

For easiness consider a single subdomain

$$\int_V \sigma_{ij} \delta u_{i,j} dV - \int_V f_i \delta u_i dV - \int_{S_t} t_i^n \delta u_i dS = 0. \tag{2.15}$$

The involved functions (e.g., displacements, strain, stresses) will be approximated via interpolation of the solution over a determined number of points termed in what follows nodes. Assume

for instance that over element e containing n such nodes we know the displacements vector u_i. Furthermore, let the displacements for the p-node $u^P = [u^P, v^P, w^P]$. Using ideas from interpolation theory it is now possible to approximate the displacements vector over an arbitrary point $\vec{x}$ inside the element as follows;

$$u_i(\vec{x}) = N_i^1(\vec{x})u^1 + N_i^2(\vec{x})u^2 + ... + N_i^P(\vec{x})u^P + ... + N_i^n(\vec{x})u^n$$

or in more general form;

$$u_i(\vec{x}) = N_i^Q(\vec{x})u^Q \tag{2.16}$$

and where the caption superscripts indicate summation over the number of nodes of the element while the subscript refers to the physical character of the variable being interpolated.

$$\varepsilon_{ij}(\vec{x}) = B_{ij}^Q(\vec{x})u^Q$$

$$\varepsilon_{ij}(\vec{x}) = \frac{1}{2}\left(u_{i,j} + u_{j,i}\right)$$

$$\varepsilon_{ij}(\vec{x}) = \frac{1}{2}\left(\frac{\partial N_i^Q}{\partial x_j} + \frac{\partial N_j^Q}{\partial x_i}\right)u^Q$$

$$B_{ij}^Q = \frac{1}{2}\left(\frac{\partial N_i^Q}{\partial x_j} + \frac{\partial N_j^Q}{\partial x_i}\right)$$

$$\delta u_i = N_i^Q(\vec{x})\delta u^Q$$

$$\int_V C_{ijkl}B_{kl}^P u^P B_{ij}^Q \delta u^Q dV - \int_V f_i N_i^Q \delta u^Q dV - \int_{S_t} t_i^n N_i^Q \delta u^Q dS = 0$$

$$\delta u^Q \int_V B_{ij}^Q C_{ijkl} B_{kl}^P dV u^P - \delta u^Q \int_V N_i^Q f_i dV - \delta u^Q \int_{S_t} N_i^Q t_i^n dS = 0$$

$$\delta u^Q f_\sigma^Q - \delta u^Q f_V^Q - \delta u^Q f_c^Q = 0$$

$$f_\sigma^Q - f_V^Q - f_c^Q = 0$$

$$f_\sigma^Q = \int_V B_{ij}^Q C_{ijkl} B_{kl}^P dV u^P \equiv K^{QP} u^P$$

$$f_V^Q = \int_V N_i^Q f_i dV$$

$$f_c^Q = \int_{S_t} N_i^Q t_i^n dS$$

$$K^{QP} u^P = f_V^Q + f_c^Q$$

2.3 Discretization of the PVW via the FEM

Application to a simple spring-mass system

Algorithm 1: Springs Algorithm

Data: Problem paramters; NUMNP, NUMEL, NMATP
Result: Displacements and spring forces
Create DME operator;
Assemble K^G, F^G;
while $j \leq 1, NUMEL$ **do**
    $K^G \leftarrow K^G + K^i$
    $F^G \leftarrow F^G + F^i$
end
Impose BCs;
Solve $[K^G]U = F^G$
Find internal forces

2.4 Variational formulations

According to Wikipedia [5]

Add Variational formulation

A variational principle is a scientific principle used within the calculus of variations, which develops general methods for finding functions which minimize or maximize the value of quantities that depend upon those functions. For example, to answer this question: "What is the shape of a chain suspended at both ends?" we can use the variational principle that the shape must minimize the gravitational potential energy.

According to Cornelius Lanczos, any physical law which can be expressed as a variational principle describes an expression which is self-adjoint. These expressions are also called Hermitian. Such an expression describes an invariant under a Hermitian transformation.

2.4.1 Principle of minimum potential energy

2.4.2 Hu-Washizu Variational Principle

Consider the following functional

$$\pi^* = \pi - \int_V \lambda_{ij}^{\varepsilon}(\varepsilon_{ij} - L_{ijk}u_k)dV - \int_{S_u} \lambda_i^u(u_i^{S_u} - \bar{u}_i)dS \tag{2.17}$$

where

- π: is the potential energy functional.
- L_{ijk} is a differential operator such $\varepsilon_{ij} = L_{ijk}u_k$.
- S_u surface where essential boundary conditions are prescribed.
- $\lambda_{ij}^{\varepsilon}$ and λ_i^u are Lagrange multipliers.

We want to determine the so-called Euler equations resulting from the condition $\delta\pi^* = 0$. Applying the variational operator we have:

$$\begin{aligned}\delta\pi* = \delta\pi &- \int_V \delta\lambda_{ij}^{\varepsilon}(\varepsilon_{ij} - L_{ijk}u_k)dV - \int_V \lambda_{ij}^{\varepsilon}(\delta\varepsilon_{ij} - L_{ijk}\delta u_k)dV \\ &- \int_V \delta\lambda_i^u(u_i^{S_u} - \bar{u}_i)dS - \int_V \lambda_i^u\delta u_i^{S_u}dS\end{aligned} \tag{2.18}$$

$$\begin{aligned}\delta\pi* = &\int_V C_{ijkl}\varepsilon_{kl}\delta\varepsilon_{ij}dV - \int_{S_t} t_i\delta u_i dS - \int_V f_i\delta u_i dV - \int_V \lambda_{ij}^{\varepsilon}\delta\varepsilon_{ij}dV + \int_V \lambda_{ij}^{\varepsilon}L_{ijk}\delta u_k dV \\ &- \int_V \delta\lambda_{ij}^{\varepsilon}(\varepsilon_{ij} - L_{ijk}u_k)dV - \int_S \delta\lambda_i^u(u_i^{S_u} - \bar{u}_i)dS - \int_{S_u} \lambda_i^u\delta u_i dS = 0\end{aligned} \tag{2.19}$$

using

$$\int_V (\lambda_{ij}^{\varepsilon}\delta u_i)_{,j}dV = \int_V \lambda_{ij}^{\varepsilon}\delta u_{i,j}dV + \int_V \lambda_{ij,j}^{\varepsilon}\delta u_i dV$$

in the above we can write

$$\begin{aligned}\int_V \lambda_{ij}^{\varepsilon}L_{ijk}\delta u_k dV &= \int_V (\lambda_{ij}^{\varepsilon}\delta u_i)_{,j}dV - \int_V \lambda_{ij,j}^{\varepsilon}\delta u_i dV \\ &= \int_{S_t} \lambda_{ij}^{\varepsilon}\delta u_i \hat{n}_j dS - \int_V \lambda_{ij,j}^{\varepsilon}\delta u_i dV\end{aligned}$$

therefore

$$\begin{aligned}\delta\pi* = &\int_V (C_{ijkl}\varepsilon_{kl} - \lambda_{ij}^{\varepsilon})\delta\varepsilon_{ij}dV - \int_{S_t} t_i\delta u_i dS - \int_V f_i\delta u_i dV + \int_{S_t} \lambda_{ij}^{\varepsilon}\delta u_i\hat{n}_j dS - \int_V \lambda_{ij,j}^{\varepsilon}\delta u_i dV \\ &- \int_V \delta\lambda_{ij}^{\varepsilon}(\varepsilon_{ij} - L_{ijk}u_k)dV - \int_{S_u} \delta\lambda_i^u(u_i^{S_u} - \bar{u}_i)dS - \int_{S_u} \lambda_i^u\delta u_i dS = 0\end{aligned}$$

$$\begin{aligned}&\int_V (C_{ijkl}\varepsilon_{kl} - \lambda_{ij}^{\varepsilon})\delta\varepsilon_{ij}dV + \int_{S_t} (\lambda_{ij}^{\varepsilon}\hat{n}_j - t_i)\delta u_i dS - \\ &\int_V (\lambda_{ij,j}^{\varepsilon} + f_i)\delta u_i dV - \int_V (\varepsilon_{ij} - L_{ijk}u_k)\delta\lambda_{ij}^{\varepsilon}dV - \int_{S_u} (u_i^{S_u} - \bar{u}_i)\delta\lambda_i^u dS - \cancel{\int_{S_u} \lambda_i^u\delta u_i^{S_u} dS} = 0\end{aligned}$$

Now, imposing the conditions $\delta\varepsilon_{ij} \neq 0$, $\delta\lambda_{ij}^{\varepsilon} \neq 0$, $\delta u_i \neq 0$ in S_t, $\delta u_i \neq 0$ in V and $\delta\lambda_i^u \neq 0$ in S_u we have

$$\lambda_{ij}^{\varepsilon} = C_{ijkl}\varepsilon_{kl} \tag{2.20}$$

$$\varepsilon_{ij} = L_{ijk}u_k \tag{2.21}$$

$$t_i = \lambda_{ij}^{\varepsilon}\hat{n}_j \tag{2.22}$$

$$\lambda_{ij,j}^{\varepsilon} + f_i = 0 \tag{2.23}$$

$$u_i^{S_u} = \bar{u}_i \tag{2.24}$$

2.5 Weighted residual methods

2.5.1 Application to the Navier equations

2.5.2 Application to the acoustic wave equation

2.6 Basic elements of interpolation theory

2.6.1 Formulation in the physical space

2.6.2 Formulation in the natural space: the continuum mechanics analogy

In typical finite element equilibrium equations we need to perform integration over the reference element domain $V_0(\vec{x})$ corresponding to originally arbitrarily shaped sub-domains as created during the meshing process. In order to proceed with this integration it is useful to consider the following continuum mechanics analogy.

First assume that the actual physical domain $V_0(\vec{x})$ is the result of a deformation process imparted upon the natural domain as shown in fig. 2.2. In this analogy, the physical domain $V_0(\vec{x})$ is regarded like a "deformed" configuration at an imaginary time $t = t$, while the natural "undeformed" domain $V(\vec{r})$ is treated like a reference undeformed configurations at time $t = 0$. Both configurations are assumed to be connected through a deformation process;

$$\begin{aligned} \vec{X} &= \vec{X}(\vec{r}) \\ \vec{r} &= \vec{r}(\vec{X}) \end{aligned} \tag{2.25}$$

In eq. (2.25) we can understand $\vec{r}$ like a material (Lagrangian) variable and $\vec{X}$ like a spatial (or Eulerian) variable. Using the continuum mechanics analogy it is clear that the "deformation" process at the continuum level is fully characterized by the "deformation" gradient or Jacobian of the transformation eq. (2.25) and defined according to;

$$dX_i = \frac{\partial X_i}{\partial r_J} dr_J \equiv J_{iJ} dr_J \tag{2.26}$$

where dr_J and dX_i represent material vectors in the original and deformed configuration. From eq. (2.26) it is evident that the Jacobian contains all the information describing the change of the

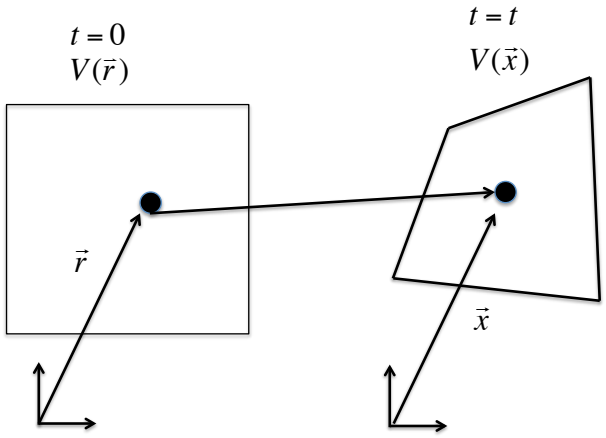


Figure 2.2. Definition of the natural domain

physical sub-domain with respect to the natural element. For the element integration process we will assume that every element $V(\vec{r})$ in the natural domain deforms into the physical element $V_0(\vec{X})$, thus allowing us to write typical terms like the ones in the material stiffness matrix

$$\int_{V(\vec{X})} \hat{B}^K_{ij}(\vec{X}) C_{ijkl} \hat{B}^P_{kl}(\vec{X}) dV(\vec{X}) \equiv \int_{V_0(\vec{r})} \hat{B}^K_{ij}(\vec{r}) C_{ijkl} \hat{B}^P_{kl}(\vec{r}) J dV_0(\vec{r}) \tag{2.27}$$

where we have used $dV(\vec{X}) = JdV(\vec{r})$, with J being the determinant of the deformation gradient and in general we transform functions between the natural and physical space making use of eq. (2.25) according to

$$f(\vec{r}) = F[\vec{X}(\vec{r})] \tag{2.28}$$

Interpolation scheme

Having identified the fact that the integration process will take place in the natural domain, we will approach the interpolation process directly in this natural space. In the case of the displacement based finite element method all the involved variables will then be obtained via interpolation of nodal displacements. For instance, assume that a given problem variable is defined in the physical space by the tensor $\Phi_{ik\ldots p}(\vec{X})$. The interpolated variable is then written like;

$$\Phi_{ij\ldots p}(\vec{X}) = H^K_{ij\ldots p}(\vec{r}) \hat{u}^K \tag{2.29}$$

where $\hat{u}^K$ represents a vector of nodal points displacements, see fig. 2.3, and $H^K_{ij\ldots p}(\vec{r})$ is an interpolator which keeps the tensorial character of the original physical variable $\Phi_{ik\ldots p}(\vec{X})$ and where the super-index makes reference to a nodal identifier (with the summation convention in place).

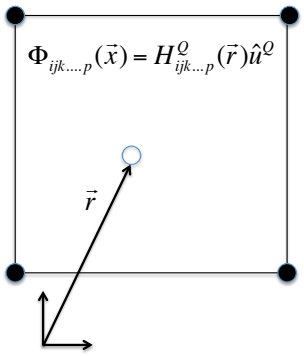


Figure 2.3. General interpolation strategy in the natural domain

Since the primary variable corresponds to displacements it must be kept in mind that $H^K_{ij...p}(\vec{r})$ corresponds to combinations of derivatives (or other arbitrary combinations) of the basic element shape functions defined in;

$$u_i(\vec{X}) = N^K_i(\vec{r})\hat{u}^K \tag{2.30}$$

For the general interpolation process we need two kinds of transformations. First we need to transform integrals over the physical space into integrals into the natural space which corresponds to

$$\int\limits_{V(\vec{X})} F(\vec{X})dV(\vec{X}) \equiv \int\limits_{V_0(\vec{r})} f(\vec{r})JdV_0(\vec{r}) \tag{2.31}$$

Second we need to relate spatial differentiation in both, the physical and spatial domains. Let us define these operators like ∇^X_i and ∇^r_I respectively. It then follows from eq. (2.28) that

$$\frac{\partial F}{\partial X_i} = \frac{\partial f}{\partial r_J}\frac{\partial r_J}{\partial X_i} \tag{2.32}$$

from where we can establish the connection between the two operators like

$$\nabla^X_i = J^{-1}_{iJ}\nabla^r_J \tag{2.33}$$

The fundamental interpolator

We further define the fundamental interpolator giving rise to gradients of the primary displacement variable in the physical space according to

$$u_{i,j}(\vec{X}) = L^K_{ij}(\vec{r})\hat{u}^K \tag{2.34}$$

This fundamental interpolator $L^K_{ik}(\vec{r})$ is derived after using eq. (2.30) and eq. (2.33) in the physical displacement gradient definition as shown next

$$\begin{aligned} u_{i,j}(\vec{X}) &= \nabla^X_j u_i(\vec{X}) \\ u_{i,j}(\vec{X}) &= \nabla^X_j N^K_i(\vec{r})\hat{u}^K \\ u_{i,j}(\vec{X}) &= J^{-1}_{jQ}\nabla^r_Q N^K_i(\vec{r})\hat{u}^K \\ u_{i,j}(\vec{X}) &= J^{-1}_{jQ} N^K_{i,Q}(\vec{r})\hat{u}^K \end{aligned}$$

then

$$L^K_{ij}(\vec{r}) = J^{-1}_{jQ} N^K_{i,Q}(\vec{r}) \tag{2.35}$$

Elemental stiffness matrix

The elemental material stiffness matrix computed in the natural domain of fig. 2.4 reads;

$$K^{KP} = \int\limits_{V_0(\vec{r})} \hat{B}^K_{ij}(\vec{r}) C_{ijkl} \hat{B}^P_{kl}(\vec{r}) J dV_0(\vec{r}) \equiv \int\limits_{r=-1}^{r=+1}\int\limits_{s=-1}^{s=+1} \hat{B}^K_{ij}(r,s) C_{ijkl} \hat{B}^P_{kl}(r,s) J(r,s) \mathrm{d}r\mathrm{d}s \tag{2.36}$$

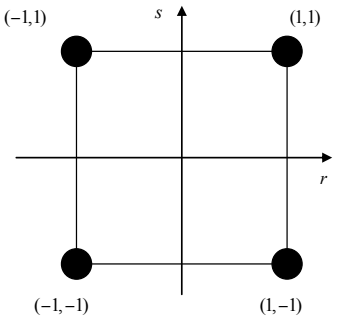


Figure 2.4. Natural domain of integration

Once the interpolator $\hat{B}^K_{ij}(\vec{r})$ has been identified the elemental stiffness matrix is obtained via numerical integration (quadrature) as described in (**??**);

$$\int\limits_{r=-1}^{r=+1}\int\limits_{s=-1}^{s=+1} \hat{B}^K_{ij}(r,s) C_{ijkl} \hat{B}^P_{kl}(r,s) J(r,s) \mathrm{d}r\mathrm{d}s \approx \sum_{i,j=1}^{\mathrm{NGPTS}} \alpha_i \alpha_j \hat{B}^K_{kl}(r_i,s_j) C_{ijkl} \hat{B}^P_{kl}(r_i,s_j) J(r_i,s_j) \tag{2.37}$$

and where NGPTS corresponds to the number of integration points, α_j is a weighting factor and r_i, s_j are the coordinates of a typical point $\vec{r}$ in the natural space of fig. 2.4.

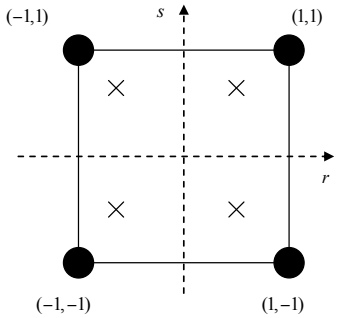


Figure 2.5. Natural integration domain showing quadrature evaluation nodes

One important aspect of the numerical integration that has to be kept in mind is accuracy. Depending on the particularly selected integration scheme, the number of introduced integration points fixes the maximum polynomial order of the considered functions that can be integrated accurately. In the case of the integrand in eq. (2.37), it is clear that this order increases as the distortion of the physical element with respect to the natural element increases. One way of dealing with this dependency of accuracy with element distortion is to make use of adaptative integration techniques which are numerically expensive. What is actually done in standard FEM analysis is to choose the number of quadrature points beforehand and introduce distortion related error criteria inside the code in such a way that some sort of validation is performed before the numerical integration process is started.

Strain displacement interpolator for the infinitesimal strain tensor

The Q-th nodal contribution to the infinitesimal strain-displacement interpolator can be obtained in explicit form as follows. Let L_x^Q and L_y^Q be the spatial differential operators in x and y respectively. We have after expanding eq. (2.35)

$$L_x^Q = J_{xP}^{-1}\frac{\partial N^Q}{\partial r_P} \equiv J_{xr}^{-1}\frac{\partial N^Q}{\partial r} + J_{xs}^{-1}\frac{\partial N^Q}{\partial s}$$
$$L_y^Q = J_{yP}^{-1}\frac{\partial N^Q}{\partial r_P} \equiv J_{yr}^{-1}\frac{\partial N^Q}{\partial r} + J_{ys}^{-1}\frac{\partial N^Q}{\partial s}$$

or in matrix form

$$\begin{Bmatrix} L_x^Q \\ L_y^Q \end{Bmatrix} = \begin{bmatrix} J_{xP}^{-1} & J_{xs}^{-1} \\ J_{yr}^{-1} & J_{ys}^{-1} \end{bmatrix} \begin{Bmatrix} \frac{\partial N^Q}{\partial r} \\ \frac{\partial N^Q}{\partial s} \end{Bmatrix} \tag{2.38}$$

The Q-th nodal contribution is then assembled as follows;

$$\begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{Bmatrix} = \begin{bmatrix} & L_x^Q & 0 & \\ \cdots & 0 & L_y^Q & \cdots \\ & L_y^Q & L_{xy}^Q & \end{bmatrix} \begin{Bmatrix} \vdots \\ u^Q \\ v^Q \\ \vdots \end{Bmatrix} \tag{2.39}$$

Algorithm 2: Strain-displacement interpolator

Data: Nodal coordinates x^Q
Result: Strain-displacement interpolator B_{ij}^Q
Compute Jacobian $J_{iJ} = \frac{\partial N_i^Q}{\partial r_J} \hat{x}^Q$
Invert Jacobian $J_{iJ} \rightarrow J_{iJ}^{-1}$
Compute fundamental interpolator $L_{ij}^Q = J_{jP}^{-1} \frac{\partial N_i^Q}{\partial r_P}$
Assemble $B_{ij}^Q = \frac{1}{2} \left(L_{ij}^Q + L_{ji}^Q \right)$

Chapter 3

Wave propagation problems in the frequency domain

3.1 Problem formulation

3.2 Discretization in complex algebra

3.3 Discretization in real algebra

3.4 Scattering by a micropolar solid

3.5 Coupling with the BEM

Chapter 4

Wave propagation problems in the time domain

4.1 Problem formulation

4.2 Explicit time integration algorithm

4.3 Statement of the Problem

Consider the discrete dynamic equilibrium equations at time t

$$M\,{}^{t}A + C\,{}^{t}V + K\,{}^{t}U = {}^{t}F \tag{4.1}$$

where M, C, K are the assembled mass, damping and stiffness matrix respectively and similarly ${}^{t}A$, ${}^{t}V$, ${}^{t}U$, ${}^{t}F$ are the nodal accelerations, velocities, displacements and external loads vectors at time t. In terms of forces, eq. (4.1) can be written like;

$${}^{t}F^{I} + {}^{t}F^{D} + {}^{t}F^{s} = {}^{t}F \tag{4.2}$$

where ${}^{t}F^{I}$, ${}^{t}F^{D}$ and ${}^{t}F^{s}$ are inertial, damping and elastic nodal forces respectively.

Expanding the acceleration and velocity terms at time t in a consistent finite central differences scheme we have;

$$
\begin{aligned}
{}^{t}A &= \frac{1}{\Delta t^2}\left({}^{t-\Delta t}U - 2{}^{t}U + {}^{t+\Delta t}U\right) \\
{}^{t}V &= \frac{1}{2\Delta t}\left(-{}^{t-\Delta t}U + {}^{t+\Delta t}U\right)
\end{aligned}
\tag{4.3}
$$

It is convenient to write eq. (4.3) like;

$$
\begin{aligned}
{}^{t}A &= {}^{t}\hat{A} + \frac{1}{\Delta t^2}{}^{t+\Delta t}U \\
{}^{t}V &= {}^{t}\hat{V} + \frac{1}{2\Delta t}{}^{t+\Delta t}U
\end{aligned}
\tag{4.4}
$$

where:

$$
\begin{aligned}
{}^{t}\hat{A} &= \frac{1}{\Delta t^2}\left({}^{t-\Delta t}U - 2{}^{t}U\right) \\
{}^{t}\hat{V} &= -\frac{1}{2\Delta t}{}^{t-\Delta t}U
\end{aligned}
\tag{4.5}
$$

are termed predictors, while $\frac{1}{\Delta t^2}{}^{t+\Delta t}U$ and $\frac{1}{2\Delta t}{}^{t+\Delta t}U$ are termed correctors.

Using eq. (4.3) in eq. (4.1) yields;

$$
\left(\frac{1}{\Delta t^2}M + \frac{1}{2\Delta t}C\right){}^{t+\Delta t}U = {}^{t}F - M{}^{t}\hat{A} - C{}^{t}\hat{V} - K{}^{t}U \tag{4.6}
$$

or after letting;

$$
\hat{K} = \frac{1}{\Delta t^2}M + \frac{1}{2\Delta t}C
$$

and

$$
{}^{t}\hat{F} = {}^{t}F - M{}^{t}\hat{A} - C{}^{t}\hat{V} - K{}^{t}U
$$

we have;

$$
\hat{K}{}^{t+\Delta t}U = {}^{t}\hat{P}
$$

which can be solved for ${}^{t+\Delta t}U$.

Algorithm 3: Explicit algorithm

Data: Values at $t = 0$, Geometry, Material Paramters
Result: Displacements, Velocity and Acceleration at any time $t = t$
Compute predictors ${}^{t}\hat{A}$, ${}^{t}\hat{V}$, ${}^{t}\hat{U}$
Assemble $\hat{K}$, ${}^{t}\hat{P}$
Solve $\hat{K}^{t+\Delta t}U = {}^{t}\hat{P}$
Perform Corrections ${}^{t}A \leftarrow {}^{t}\hat{A} + \frac{1}{2\Delta t^2}{}^{t+\Delta t}U$ and ${}^{t}V \leftarrow {}^{t}\hat{V} + \frac{1}{2\Delta t}{}^{t+\Delta t}U$

Redefine forces as follows

$$\begin{aligned} {}^{j}F^{I} &= \frac{1}{\Delta t^2}M^{j}U \\ {}^{j}F^{D} &= \frac{1}{2\Delta t}C^{j}U \\ {}^{j}F^{S} &= K^{j}U \end{aligned} \tag{4.7}$$

and write (4.6) as;

$${}^{t+\Delta t}F^{I} + {}^{t+\Delta t}F^{D} = {}^{t}F - {}^{t}F^{s} + 2{}^{t}F^{I} - {}^{t-\Delta t}F^{I} + {}^{t-\Delta t}F^{D} \tag{4.8}$$

- (4.8) is an equilibrium equation at time $t = t$ allowing to predict the displacements at time $t = t + \Delta t$ in terms of previously known values at times t and $t = t - \Delta t$.
- The equation is exact within the error introduced by the expansion used in (4.3).

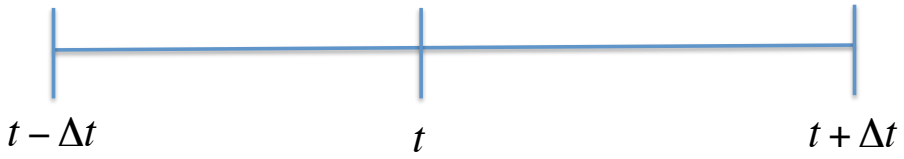


Figure 4.1. Definition of the general iteration

- The first predicted solution is at $t = \Delta t$ and we require data at $t = -\Delta t$ and at $t = 0$.

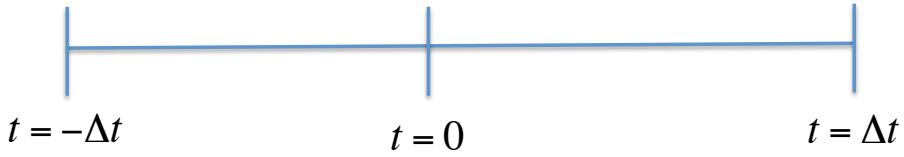


Figure 4.2. Definition of the general iteration

4.3.1 Damping Assumptions

1 Use Rayleigh Damping and retain the velocity expansion used in (4.3). That is;

$$C = \alpha M + \beta K \tag{4.9}$$

then we have (in terms of forces);

$$(1 + \beta \Delta t^2)^{t+\Delta t} F^I + \frac{\alpha}{2\Delta t}{}^{t+\Delta t} F^S = {}^t \hat{F} \tag{4.10}$$

where;

$${}^t \hat{F} = {}^t R - {}^t F^S + 2^t F^I - {}^{t-\Delta t} F^I + {}^{t-\Delta t} F^D$$

Solution in equation (4.10) requires the full assembly and factorization of an effective stiffness matrix.

2 Neglect damping (This is however inconvenient for finite domains)

$${}^{t+\Delta t} F^I = {}^t F - {}^t F^S + 2^t F^I - {}^{t-\Delta t} F^I \tag{4.11}$$

3 Use Rayleigh damping but modify the velocity expansion introduced in (4.3). Using

$${}^t V = \frac{1}{\Delta t}({}^t U - {}^{t-\Delta t} U) \tag{4.12}$$

yielding;

$${}^{t+\Delta t} F^I = {}^t F - {}^t F^S + 2^t F^I - {}^t F^D + {}^{t-\Delta t} F^D - {}^{t-\Delta t} F^I \tag{4.13}$$

Defining a set of forces associated to the initial conditions like;

$${}^{t-\Delta t} F^{IC} = {}^{t-\Delta t} F^I - {}^{t-\Delta t} F^D$$

we have

$${}^{t+\Delta t} F^I = {}^t F - {}^t F^S + 2^t F^I - {}^t F^D - {}^{t-\Delta t} F^{IC} \tag{4.14}$$

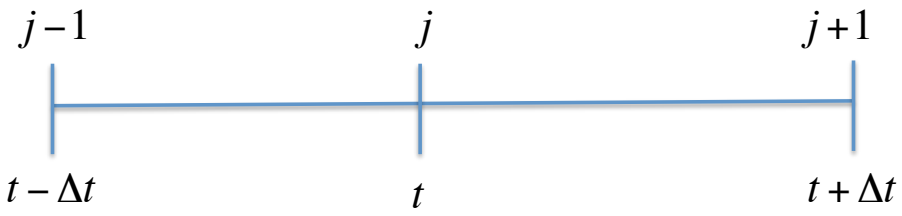


Figure 4.3. Definition of the general iteration

4.3.2 Algorithm corresponding to the damping assumption 3

Let us write (4.14) like

$$ {}^{j+1}F^I = {}^{j}F - {}^{j}F^S + 2^{j}F^I - {}^{j}F^D - {}^{j-1}F^{IC} \tag{4.15} $$

where the initialization process corresponds to;

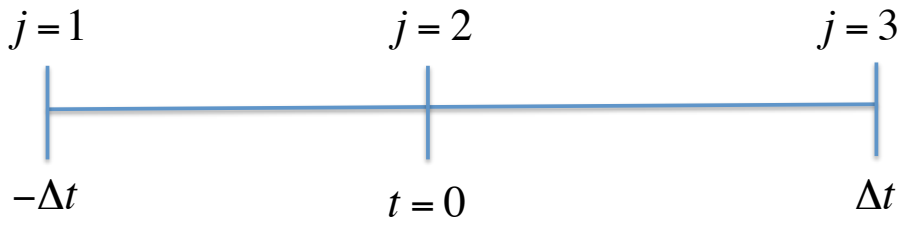


Figure 4.4. Definition of the initial iteration

Applying (4.15) for $t = 0$ we have;

$$ {}^{0}F^I = {}^{0}F - {}^{0}F^S + 2^{0}F^I - {}^{0}F^D - {}^{-\Delta t}F^{IC} $$

from which it is clear that we require ${}^{-\Delta t}U$. Applying the central difference expansion at $t = 0$ and solving for ${}^{-\Delta t}U$ yields;

$$ {}^{-\Delta t}U = {}^{0}F - \Delta t\,{}^{0}V + \frac{\Delta t^2}{2}\,{}^{0}A \tag{4.16} $$

Using (4.16) in (4.15) allows us to start up the algorithm.

Particulars

In what follows we concentrate on this last algorithm and in order to study some details we return to its standard displacements form. Writing (4.15) in terms of displacements and re-arranging yields;

$$\frac{1}{\Delta t^2}M^{t+\Delta t}U =^t F-(1+\frac{\beta}{\Delta t})K^tU+(\frac{2}{\Delta t^2}-\frac{\alpha}{\Delta t})M^tU-(\frac{1}{\Delta t^2}-\frac{\alpha}{\Delta t})M^{t-\Delta t}U+(\frac{\beta}{\Delta t})K^{t-\Delta t}U \quad (4.17)$$

Let;

$$\begin{aligned} a_1 &= 1+\frac{\beta}{\Delta t} \\ a_2 &= \frac{2}{\Delta t^2}-\frac{\alpha}{\Delta t} \\ a_3 &= \frac{1}{\Delta t^2}-\frac{\alpha}{\Delta t} \\ a_4 &= \frac{\beta}{\Delta t} \end{aligned}$$

$${}^{t+\Delta t}F^I =^t F - a_1K^tU + a_2M^tU - a_3M^{t-\Delta t}U + a_4K^{t-\Delta t}U \quad (4.18)$$

4.3.3 Decoupling

Consider the equation for the i-th d.o.f;

$$\frac{1}{\Delta t^2}M_{ij}^{t+\Delta t}U_j =^t F_i - a_1K_{ij}^tU_j + a_2M_{ij}^tU_j - a_3M_{ij}^{t-\Delta t}U_j + a_4K_{ij}^{t-\Delta t}U_j \quad (4.19)$$

where we keep i fixed in (4.19). For a lumped mass matrix we can write;

$$M_{ij} = m_I\delta_{ij}$$

then (4.19) becomes;

$$\frac{1}{\Delta t^2}m_I^{t+\Delta t}U_i =^t F_I - a_1K_{ij}^tU_j + a_2m_I^tU_i - a_3m_I^{t-\Delta t}U_i + a_4K_{ij}^{t-\Delta t}U_j \quad (4.20)$$

Let;

$$\begin{aligned} {}^{t+\Delta t}F_i^I &= \frac{1}{\Delta t^2} m_I^{t+\Delta t} U_i \\ {}^{t}\hat{F}_i^S &= a_1 K_{ij}^t U_j \\ {}^{t}\hat{F}_i^I &= a_2 m_I^t U_i \\ {}^{t-\Delta t}\hat{F}_i^I &= a_3 m_I^{t-\Delta t} U_i \\ {}^{t-\Delta t}\hat{F}_i^S &= a_4 K_{ij}^{t-\Delta t} U_j \end{aligned} \tag{4.21}$$

so the recursive equation takes the form;

$${}^{t+\Delta t}F_i^I = {}^{t}\ F_i - {}^{t}\ \hat{F}_i^S + {}^{t}\ \hat{F}_i^I - {}^{t-\Delta t}\ \hat{F}_i^I + {}^{t-\Delta t}\ \hat{F}_i^S \tag{4.22}$$

and the algorithm then reduces to;

Algorithm 4: Summarized Algorithm

Data: Time span, Geometry, Material Parameters
Result: Displacements, Velocity and Acceleration time histories
Compute ${}^{t+\Delta t}F_i^I$
Solve for ${}^{t+\Delta t}U_i = \left(\frac{\Delta t^2}{m_I}\right)^{t+\Delta t} F_i^I$
Update ${}^{t}V_i$, ${}^{t}A_i$

To initialize the algorithm we apply the FD's equations at $t = 0$

$$\frac{1}{\Delta t^2} m_I^{\Delta t} U_i = {}^{0}\ F_i - a_1 K_{ij}^0 U_j + a_2 m_I^0 U_i - a_3 m_I^{-\Delta t} U_i + a_4 K_{ij}^{-\Delta t} U_j$$

where ${}^{-\Delta t}U_i$ is obtained from (4.16)

$${}^{-\Delta t}U_i = {}^{0}\ U_i - \Delta t^0 V_i + \frac{\Delta t^{2\,0}}{2} A_i \tag{4.23}$$

The initial acceleration is obtained after assuming homogeneous IC's;

$$m_I^0 A_i + C_{ij}^0 V_j + K_{ij}^0 U_j = {}^{0}\ F_i$$

therefore

$$m_I^0 A_i = \frac{^0F_i}{m_I}$$
$$^{-\Delta t}U_i = \frac{\Delta t^2}{2m_I}\,^0F_i$$

Moreover, neglecting the damping effects on the prediction of $^{\Delta t}U_i$ yields;

$$^{\Delta t}U_i = \frac{\Delta t^2}{2m_I}\,^0F_i$$

Algorithm 5: Full Algorithm

Data: Time span, Geometry, Material Parameters
Result: Displacements, Velocity and Acceleration time histories
Initialize solution vectors ($j = 1$);
$^0U_i \longrightarrow ^1U_i = 0$, $^0V_i = 0$, $^1A_i = \frac{^1R_i}{m_I}$;
Select Δt and integration constants a_1,a_2, a_3, a_4;
Fix 1-st predicted value (let $j = 2$);

$$^{\Delta t}U_i \longleftarrow \frac{\Delta t^2}{2m_I}\,^0F_i \longleftrightarrow \left[^2U_i \longleftarrow \frac{\Delta t^2}{2m_I}\,^1F_i\right]$$

Time Integration Phase;
while $j \leq N$ **do**

$$^{j+1}F_i^I \longleftarrow ^jF_i - a_1K_{ij}^jU_j + a_2m_I^jU_j - a_3m_I^{j-1}U_i + a_4K_{ij}^{j-1}U_j$$
$$^{j+1}U_i \longleftarrow \frac{\Delta t^2}{2m_I}\,^0F_i^I$$
$$^jA_i \longleftarrow \frac{1}{\Delta t^2}\left(^{j-1}U_i - 2^jU_i + ^{j+1}U_i\right)$$
$$^jV_i \longleftarrow \frac{1}{2\Delta t}\left(-^{j-1}U_i + ^{j+1}U_i\right)$$
$$j \longleftarrow j+1$$

end

Nodal Assembler

In the uncoupled explicit finite element formulation the equation solving process proceeds one degree of freedom at a time. This implies a different assembly process to the one used in an implicit algorithm where a formal coefficient matrix is assembled and inverted. Now the mass,

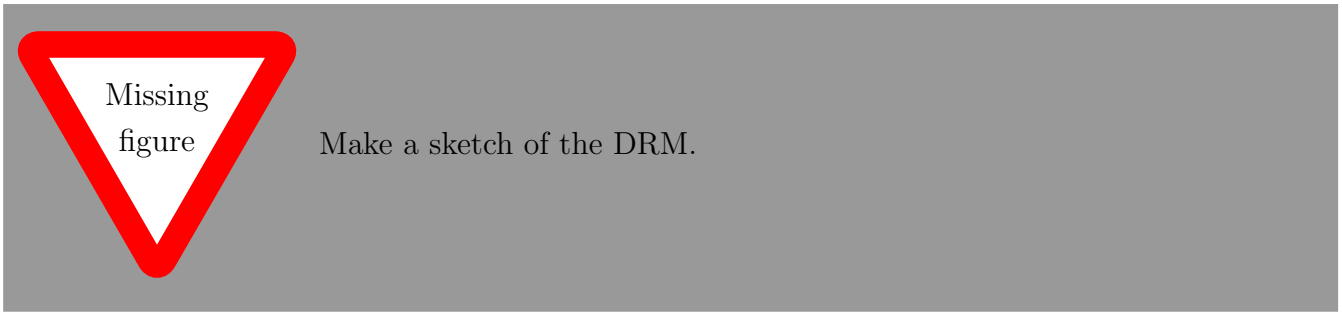
damping and stiffness elemental matrices are used to obtain effective nodal forces at each degree of freedom. In summary the mesh is not covered in an element by element basis, but in a node by node basis. In the following algorithm we discuss this nodal assembly process where in order to solve the displacement at a given degree of freedom prior knowledge of the element contributing to the given node is necessary. In the nodal assembler algorithm the following arrays are needed.

ILIST(): Stores the elements connected to the current node.
LPLIST():Stores the local position of the current node in each one of the elements of ILIST().
NIEL():Number of elements at the current node. This array is used to access ILIST().

Algorithm 6: Nodal Assembler

Data: Number of nodal points, number of elements, Model
Result: Displacements, Velocity and Acceleration time histories
while $i \leq NUMNP$ **do**
| $K \leftarrow NIEL(i)$
end

4.4 The DRM approach



Make a sketch of the DRM.

Chapter 5

Periodic media problems

A periodic material is defined as the repetition of a given motif in one, two or three dimensions. This motif refers to heterogeneities at micro-structural level and it can contains several materials and diverse geometries. These materials can be solids or fluids.

A periodic material is completely described by a lattice and a elementary unit, that is termed *elementary cell*. The lattice is defined by a vector base, and the dimension of thi base is the number of direction for the periodicity. This set is called lattice *base vectors*, and they allow to build the whole material applying successive translation operations [6]. For periodic material in 3D, we will denote the base vectors as $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$ (see Figure 5.1). In the case of 2D periodicity, we can have a material that is infinite or finite in the third dimension (see Figure 5.2), we denote the vectors as $\mathbf{a}_1$ and $\mathbf{a}_2$. There is also possible to have periodic materials in one direction, and this one can be finite or infinite in the other two directions.

The mesh generator `gmsh` provides an option to link elements in opposite sides. [7]. Finite Element Method Magnetic (FEMMM) provides the option of *periodic* boundary conditions, where Multipoint constraints are imposed, this are used for approximation of infinite boundary conditions [8].

Describe periodic boundary conditions and Bloch theorem as boundary conditions.

Add Periodic media info.

Periodic boundary conditions are common in molecular dynamics and micromechanics/homogenization.

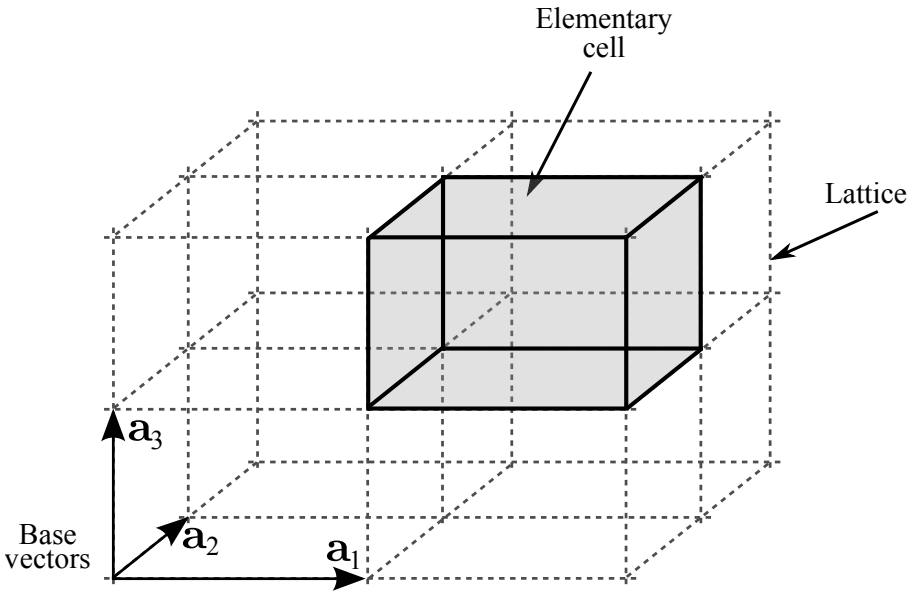


Figure 5.1. Depiction of a periodic material in 3D.

5.1 Bloch theorem

Let us start from an equation of the form

$$Lu(\mathbf{x}) = \omega^2 u(\mathbf{x}) \ , \tag{5.1}$$

where L is a differential operator with algebraic properties that are similar to the wave equation [9]. The Bloch theorem establishes that the solution for (5.1) is given by

$$u(\mathbf{x}) = w(\mathbf{x}) e^{i\mathbf{k}\cdot\mathbf{x}} \ , \tag{5.2}$$

where $w(\mathbf{x})$ is function with the same periodicity of the material and is termed Bloch function. The theorem according to [10] states

> *The eigenfunctions of the wave equation for a periodic potential are the product of a plane wave* $\exp(i\mathbf{k}\cdot\mathbf{r})$ *times a function* $u_{\mathbf{k}}(\mathbf{r})$ *with the periodicity of the crystal lattice.*

If we write the solution (5.2) for the point $\mathbf{x}+\mathbf{a}$, we obtain

$$u(\mathbf{x}+\mathbf{a}) = w(\mathbf{x}+\mathbf{a}) e^{i\mathbf{k}\cdot(\mathbf{x}+\mathbf{a})} \ ,$$

being $\mathbf{a} = n_i \mathbf{a}_i$ a vector with the periodicity of the lattice, and $n_i \in \mathbb{Z}$. Since$w(\mathbf{x})$ should present the same periodicity of the lattice

$$u(\mathbf{x}+\mathbf{a}) = w(\mathbf{x}) e^{i\mathbf{k}\cdot(\mathbf{x}+\mathbf{a})} \ ,$$

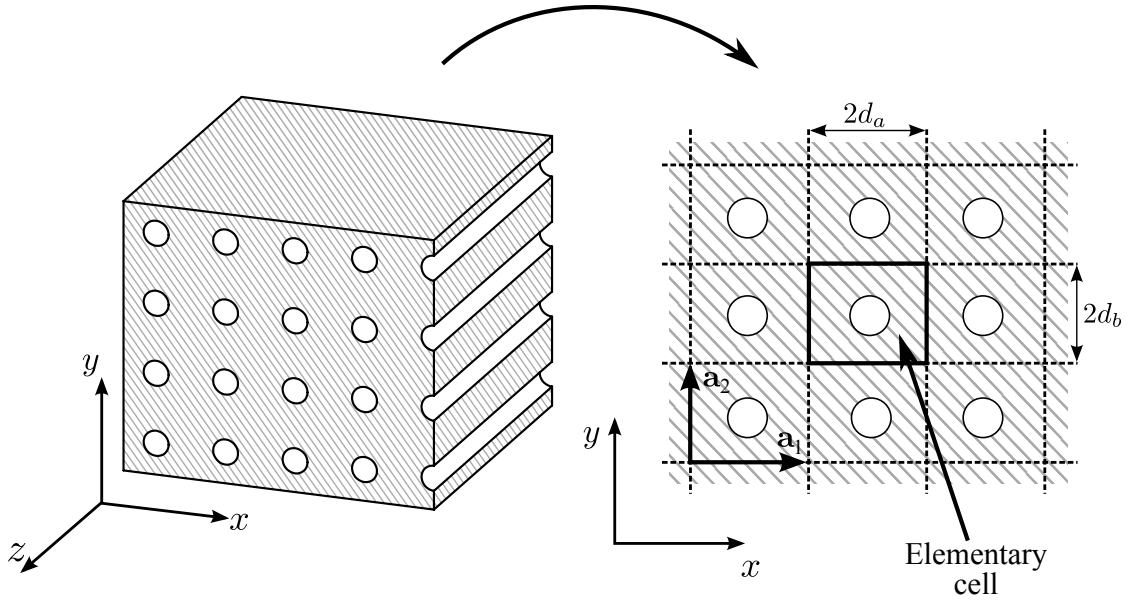


Figure 5.2. Depiction of a periodic material in 2D.

implying

$$w(\mathbf{x}) = u(\mathbf{x}+\mathbf{a})e^{-i\mathbf{k}\cdot(\mathbf{x}+\mathbf{a})} \ ,$$

and substituting this in (5.2), we get

$$u(\mathbf{x}+\mathbf{a}) = u(\mathbf{x})e^{i\mathbf{k}\cdot\mathbf{a}} \ . \tag{5.3}$$

Equation (5.3) is termed Bloch periodicity or Bloch-periodic condition. Besides the form in equation (5.1), we can also have an equation of the form

$$Lu(\mathbf{x}) = \omega^2 u(\mathbf{x}) + \mathbf{f}(\omega) \ , \tag{5.4}$$

where $\mathbf{f}(\omega)$ is a harmonic body force with circular frequency ω that presents the same periodicity that the lattice, see [11].

5.1.1 Bloch theorem in elastodynamics

Let us consider the wave propagation in an elastic, linear isotropic material. This is described using the Navier-Cauchy equation in the frequency domain (neglecting body forces)

$$(\lambda+\mu)\nabla(\nabla\cdot\mathbf{u}) + \mu\nabla\times(\nabla\times\mathbf{u}) = -\omega^2\rho\mathbf{u} \ .$$

The displacement vector can be expressed as $\mathbf{u} = \nabla\varphi + \nabla\times\boldsymbol{\psi}$ due to Helmholtz decomposition theorem [12, 13], and the potentials $\varphi, \boldsymbol{\psi}$ are the scalar and vector potential, respectively. These

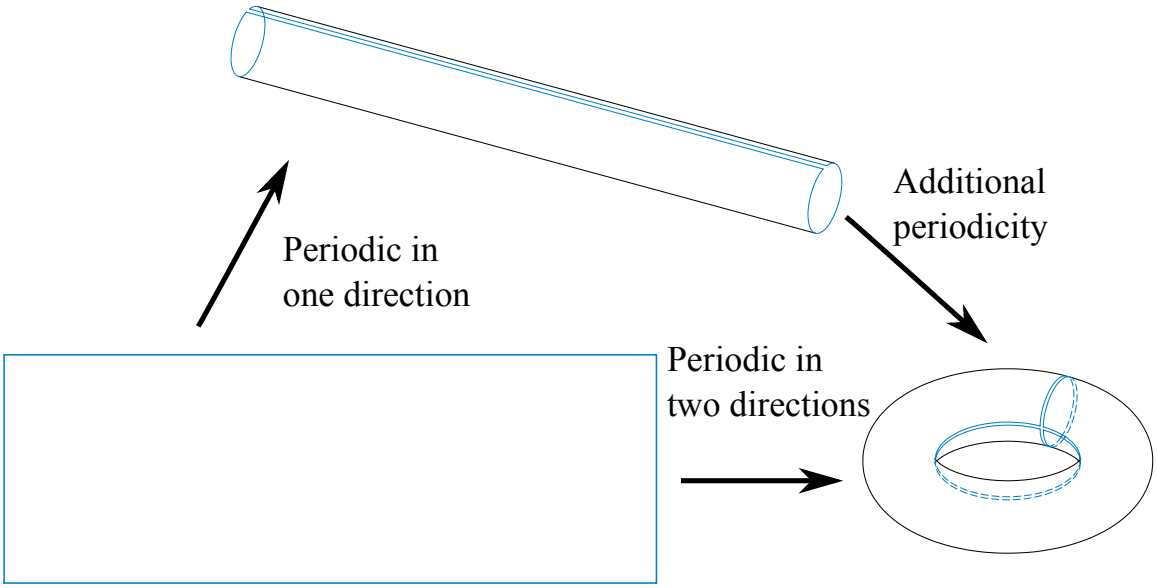


Figure 5.3. Topology for a rectangular region with periodic boundary conditions.

potential verify

$$\nabla^2\varphi = -\frac{\omega^2}{\alpha^2}\varphi \ ,$$
$$\nabla^2\boldsymbol{\psi} = -\frac{\omega^2}{\beta^2}\boldsymbol{\psi} \ ,$$

where α, β correspond with the speed for the longitudinal and transverse waves.

Since the equations for the potentials $\varphi, \boldsymbol{\psi}$ are wave equations, they satisfy Bloch theorem and then

$$\varphi(\mathbf{x}) = \varphi(\mathbf{x}+\mathbf{a})e^{i\mathbf{k}\cdot\mathbf{a}},$$
$$\boldsymbol{\psi}(\mathbf{x}) = \boldsymbol{\psi}(\mathbf{x}+\mathbf{a})e^{i\mathbf{k}\cdot\mathbf{a}} \ ,$$

and

$$\mathbf{u}(\mathbf{x}) = \nabla\varphi(\mathbf{x}) + \nabla\times\boldsymbol{\psi}(\mathbf{x}) = e^{i\mathbf{k}\cdot\mathbf{a}}\nabla\varphi(\mathbf{x}) + e^{i\mathbf{k}\cdot\mathbf{a}}\nabla\times\boldsymbol{\psi}(\mathbf{x}),$$
$$= e^{i\mathbf{k}\cdot\mathbf{a}}\left[\nabla\varphi(\mathbf{x}) + \nabla\times\boldsymbol{\psi}(\mathbf{x})\right] = e^{i\mathbf{k}\cdot\mathbf{a}}\mathbf{u}(\mathbf{x}+\mathbf{a}) \ .$$

Thus, the Bloch theorem is satisfied for the displacement field, i.e.

$$\mathbf{u}(\mathbf{x}) = e^{i\mathbf{k}\cdot\mathbf{a}}\mathbf{u}(\mathbf{x}+\mathbf{a}) \ .$$

5.2 Bloch theorem as boundary conditions

If we start from the reduced wave equation [14] the boundary value problem would be given by

$$\begin{aligned}
&(\lambda+2\mu)\nabla(\nabla\cdot\mathbf{u})+\mu\nabla\times(\nabla\times\mathbf{u})=-\omega^2\rho\mathbf{u} && \text{in } \Omega,\\
&\mathbf{u}(\mathbf{x}+\mathbf{a})=\mathbf{u}(\mathbf{x})e^{i\mathbf{k}\cdot\mathbf{a}} && \text{in } \Gamma_u,\\
&\mathbf{t}(\mathbf{x}+\mathbf{a})=-\mathbf{t}(\mathbf{x})e^{i\mathbf{k}\cdot\mathbf{a}} && \text{in } \Gamma_t\ .
\end{aligned}$$

We will call this type of boundary conditions *Bloch-periodicity*[1]. The operator of this partial differential equation is self-adjoint and positive definite for Bloch-periodic conditions [15, 16], thus the eigenvalues are real and positive. After discretization (using FEM), the resulting matrices are Hermitian and positive definite. What makes sense from a physical point of view, the energy of the system is always bigger than zero and the eigenvalues (squared frequencies) must be positive.

The problem can be equivalently formulated for the Bloch function $\mathbf{w}(\mathbf{x})$, where the Bloch-periodic conditions are replace by (plane) periodic conditions [17]. The equivalent boundary value problem is

$$\begin{aligned}
&B\mathbf{w}=-\omega^2\rho\mathbf{w} && \text{in } \Omega,\\
&\mathbf{w}(\mathbf{x}+\mathbf{a})=\mathbf{w}(\mathbf{x}) && \text{in } \Gamma_u,\\
&\frac{\partial\mathbf{w}}{\partial\hat{\mathbf{n}}}(\mathbf{x}+\mathbf{a})=\frac{\partial\mathbf{w}}{\partial\hat{\mathbf{n}}}(\mathbf{x}) && \text{in } \Gamma_t\ ,
\end{aligned}$$

where B is the operator

$$B=(\lambda+\mu)\left[\nabla\nabla\cdot+i\{\mathbf{k}\nabla\cdot+\mathbf{k}\cdot\nabla+\mathbf{k}\times\nabla\times\}\right]()-\mu\left[\nabla^2+2i\mathbf{k}\cdot\nabla-\|\mathbf{k}\|^2\right]()\ .$$

5.3 Variational form for elastodynamics with Bloch-periodicity

Let us start from the momentum equation in the frequency domain

$$\sigma_{rs,s}+f_r(\omega)=-\rho\omega^2u_r,\ \text{en } \Omega,$$

where f_r is a harmonic force in the direction r, $\sigma_{rs}=C_{rskl}\epsilon_{kl}$ and $\epsilon_{kl}=1/2(u_{k,l}+u_{l,k})$. The boundary conditions are

$$u_r(x_s+a_s)=u_r(x_s)e^{ik_sa_s}\ \text{in } \Gamma_u, \tag{5.5a}$$
$$t_r(x_s+a_s)=-t_r(x_s)e^{ik_sa_s}\ \text{in } \Gamma_t, \tag{5.5b}$$

being t_r the traction on the surface of the cell.

[1] Periodic boundary conditions are a particular case, where the factor $e^{i\mathbf{k}\cdot\mathbf{a}}=1$.

Now, let us multiply the momentum equation by the complex conjugate of an arbitrary function v_r and integrate over the domain[2]

$$\int_\Omega v_r^*[\sigma_{rs,s} + f_r(\omega)]d\Omega = -\omega^2 \int_\Omega \rho v_r^* u_r d\Omega \ ,$$

the complex conjugate is denoted by $*$. Then, using the constitutive relation

$$\int_\Omega v_r^* C_{rskl}\epsilon_{kl,s} d\Omega + \int_\Omega v_r^* f_r d\Omega = -\omega^2 \int_\Omega \rho v_r^* u_r d\Omega \ ,$$

and using divergence theorem

$$-\int_\Omega v_r, s^* C_{rskl}\epsilon_{kl} d\Omega + \int_\Omega v_r^* t_r(n_s) d\Gamma + \int_\Omega v_r^* f_r d\Omega = -\omega^2 \int_\Omega \rho v_r^* u_r d\Omega \ ,$$

with $t_r^{n_s} = C_{rskl}\epsilon_{kl} n_s$, if we take $v_r = u_r$ and consider that the tensor ϵ_{rs} is symmetric, we obtain

$$-\int_\Omega \epsilon_{rs} C_{rskl}\epsilon_{kl} d\Omega + \int_\Gamma u_r^* t_r(n_s) d\Gamma \int_\Omega u_r^* f_r(\omega) d\Omega = -\omega^2 \int_\Omega \rho u_r^* u_r d\Omega \ .$$

And neglecting the body forces, the Energy Functional for this problem is written as

$$\Pi(\omega) = \int_\Omega \epsilon_{rs}^*(\mathbf{x}) C_{rskl}(\mathbf{x})\epsilon_{kl}(\mathbf{x}) d\Omega - \omega^2 \int_\Omega \rho(\mathbf{x}) u_r^*(\mathbf{x}) u_r(\mathbf{x}) d\Omega - \int_\Gamma u_r^*(\mathbf{x}) t_r(\mathbf{x}) d\Gamma \ . \tag{5.6}$$

The first term in the equation (5.6) is the deformation energy inside the domain, the second term corresponds with the kinetic energy for harmonic motion with angular frequency ω, the last term is the deformation energy on the boundary of the domain. The boundary term is related with the imposition of natural boundary conditions (Neumann BCs) in the formulation of the FEM. In this case the Bloch-periodic conditions for the forces are implicitly satisfied in the variational form, as we will see in detail.

Since the domain is a single cell of a lattice, it should form a tessellation [3]. Hence, the boundaries can be grouped by pairs, i.e., each side (*reference side*) should have an opposite side (*image side*), so the last term in (5.6)can be rewritten as

$$\int_\Gamma u_r^*(\mathbf{x}) t_r(\mathbf{x}) d\Gamma = \sum_l \int_{\Gamma_l} [u_r^*(\mathbf{x}) t_r(\mathbf{x}) + u_r^*(\mathbf{x} + \mathbf{a_l}) t_r(\mathbf{x} + \mathbf{a}_l)] d\Gamma \ ,$$

[2] In the deduction of weak forms for FEM it is customary to multiply by a function v_r that is real. In this case the function is complex; in order to satisfy the properties for an inner product, we need to multiply by the complex conjugate (see for example [12, 18]).

[3] A tessellation is a regular pattern that covers all the space with no overlaps and no gaps.

where the index l refers to each one of the pairs of opposite sides in the boundary. Thus

$$\int_\Gamma u_r^*(\mathbf{x})t_r(\mathbf{x})d\Gamma = \sum_l \int_{\Gamma_l} [u_r^*(\mathbf{x})t_r(\mathbf{x}) + e^{-i\mathbf{k}\cdot\mathbf{a}_l}u_r^*(\mathbf{x})t_r(\mathbf{x}+\mathbf{a}_l)]d\Gamma \ ,$$
$$\int_\Gamma u_r^*(\mathbf{x})t_r(\mathbf{x})d\Gamma = \sum_l \int_{\Gamma_l} u_r^*(\mathbf{x})[t_r(\mathbf{x}) + e^{-i\mathbf{k}\cdot\mathbf{a}_l}t_r(\mathbf{x}+\mathbf{a}_l)]d\Gamma \ ,$$

and the expression in square brackets is the condition (5.5b), that is equal to zero. This variational form can be computed for other operators that have similar algebraic properties to wave equations and are positive definite [9, 15, 16].

5.4 Implementation

5.4.1 Bloch-periodicity in FEM

The Bloch-periodic conditions in a Finite Element problem or, more generally, in a discrete problem can be formulated as a set of relations between the degrees of freedom of opposite sides of the unit cell. Let us start from the original problem

$$[K - \omega^2 M]\{u\} = \{f\} \ , \tag{5.7}$$

where

$$u = [u_l \ u_r \ u_b \ u_t \ u_{lb} \ u_{rb} \ u_{lt} \ u_{rt} \ u_i]^T = [u_1 \ u_2 \ u_3 \ u_4 \ u_5 \ u_6 \ u_7 \ u_8 \ u_9]^T \ ,$$
$$f = [f_l \ f_r \ f_b \ f_t \ f_{lb} \ f_{rb} \ f_{lt} \ f_{rt} \ f_i]^T = [f_1 \ f_2 \ f_3 \ f_4 \ f_5 \ f_6 \ f_7 \ f_8 \ f_9]^T \ ,$$

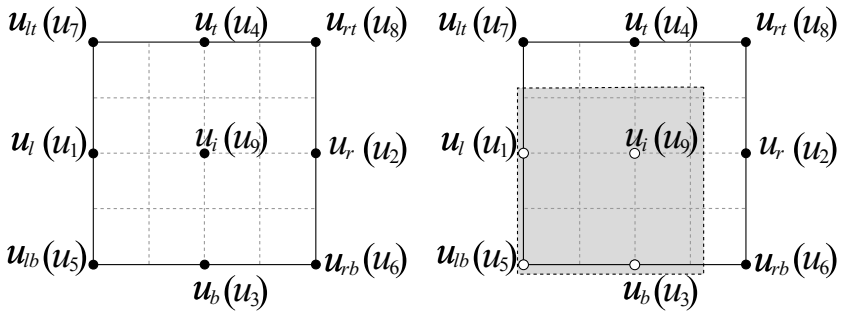
the subindices are describe in Figure 5.4.

From Bloch theorem we know

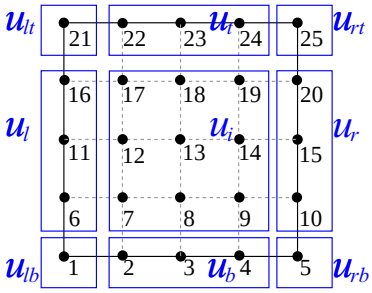
$$u_2 = e^{i\psi_x}u_1, \quad u_4 = e^{i\psi_y}u_3, \quad u_6 = e^{i\psi_x}u_5,$$
$$u_7 = e^{i\psi_y}u_5, \quad u_8 = e^{i(\psi_x+\psi_y)}u_5,$$

with $\psi_x = 2k_x d_a$ and $\psi_y = 2k_y d_b$ phase shifts in x and y, and $(k_x, k_y) = \mathbf{k}$ wave vector. If we think about this relations as a linear transformation we can write

$$\underbrace{\begin{Bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{Bmatrix}}_{u} = \underbrace{\begin{bmatrix} I_{nl} & 0 & 0 & 0 \\ I_{nl}e^{i\psi_x} & 0 & 0 & 0 \\ 0 & I_{nb} & 0 & 0 \\ 0 & I_{nb}e^{i\psi_y} & 0 & 0 \\ 0 & 0 & I_2 & 0 \\ 0 & 0 & I_2 e^{i\psi_x} & 0 \\ 0 & 0 & I_2 e^{i\psi_y} & 0 \\ 0 & 0 & I_2 e^{i(\psi_x+\psi_y)} & 0 \\ 0 & 0 & 0 & I_{ni} \end{bmatrix}}_{T} \underbrace{\begin{Bmatrix} u_1 \\ u_3 \\ u_5 \\ u_9 \end{Bmatrix}}_{u_R} , \tag{5.8}$$



(a) Original cell. **(b)** *Reduced* cell.



(c) The black circles now correspond to single degrees of freedom. Notice that the nodal numbering does not match the original numbering for groups of degrees of freedom.

Figure 5.4. Relevant sets of DOF for an schematic unitary cell discretized with several finite elements. Each circle represents a family of degrees of freedom for a typical mesh and not a single degree of freedom. In part a) we show all the degrees of freedom for the unitary cell before imposing the relevant Bloch-periodic boundary conditions. Part b) shows the reduced cell where the white circles enclosed by the dark square represent the *reference* nodes containing the information from the *image* nodes which will be eventually deleted from the system. Part c) shows an example of node grouping for a mesh of 4×4 elements.

being I_n identity matrices of size n (equal to the number of degrees of freedom associated to each group). We can write the equation (5.8) in compact form as $u = Tu_R$, and substituting (5.8) in (5.7) we obtain

$$[KT]u_R = \omega^2[MT]u_R \ . \tag{5.9}$$

Also, from the Bloch theorem we have the following equilibrium conditions

$$f_2 + e^{i\psi_x} f_1 = 0, \quad f_4 + e^{i\psi_y} f_3 = 0,$$
$$f_8 + e^{i\psi_x} f_6 + e^{i\psi_y} f_7 + e^{i(\psi_x+\psi_y)} f_5 = 0 \ .$$

And, again, thinking this relations as another linear transformation

$$\underbrace{\begin{Bmatrix} 0 \\ 0 \\ 0 \\ f_9 \end{Bmatrix}}_{f_R} = \underbrace{\begin{bmatrix} I_{nl} & I_{nl}e^{-i\psi_x} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{nb} & I_{nb}e^{-i\psi_y} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_2 & I_2e^{-i\psi_x} & I_2e^{-i\psi_y} & I_2e^{-i(\psi_x+\psi_y)} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_{ni} \end{bmatrix}}_{T^H} \underbrace{\begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{Bmatrix}}_{f} . \tag{5.10}$$

Now, we multiply (5.9) by T^H, where T^H is the Hermitian transpose of T,

$$\left[\underbrace{T^H K T}_{K_R} - \omega^2 \underbrace{T^H K T}_{M_R}\right] \{u_R\} = \{f_R\} \ . \tag{5.11}$$

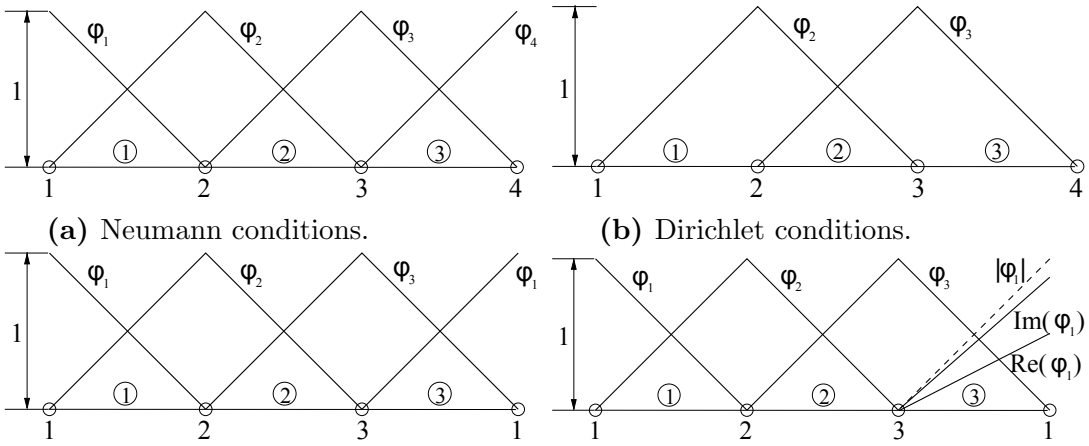
The new system of equations is of size $n \times n$, being n the number of independent variables in u_R, and the matrices K_R, M_R are Hermitian. This conditions is derived from the fact that the original matrices K, M are real and symmetric. Neglecting the body forces f_R we obtain the generalized eigenvalue problem for the reduced system

$$[K_R]\{u_R\} = \omega^2 [M_R]\{u_R\} \ . \tag{5.12}$$

The implementation of the periodic and Bloch-periodic boundary conditions can be achieved in two different ways:

- changing the connectivity, and the respective interpolation functions (see Figure 5.5); or
- assembling the mass and stiffness matrices assuming they do not have boundary conditions and then, using elementary row operations and elementary column operations, obtain the matrices with the desired conditions.

The modification over the mesh is well illustrated in Figure 5.5, where the continuity is warranted in the periodic case and a phase shift appears for the Bloch-periodic one.



(a) Neumann conditions. **(b)** Dirichlet conditions.

(c) Periodicity conditions. **(d)** Bloch-periodicity conditions.

Figure 5.5. Basis functions for linear finite elements. (a) Neumann conditions; (b) Dirichlet conditions; (c) Periodic conditions; and (d) Bloch-periodic conditions. The real and imaginary part of φ_1 are illustrated in the basis with Bloch conditions in (d) for a phase shifts of $ka = \pi/3(\omega = [0, a])$. The modulus of φ_1 is the same at $x = 0$ and at $x = a$.

When obtaining the matrices without boundary conditions, the *reduced* matrices can be obtained in two different ways. One of them is applying elementary operations. We call the nodes of one side *reference nodes* and the corresponding opposite nodes *image nodes*. The procedure to obtain the reduced matrices is [18, 19]

- Multiply the elements of A_{kl} associated with a boundary node by $f_k^* f_l$.
- For each row i of A associated with a reference node, add all the rows k associated with the corresponding image nodes. For each column j associated with a reference node, add all the columns l associated with the corresponding image nodes.
- Eliminate all the rows k and columns l associates with image nodes.

The factor $f_j = e^{i\mathbf{k}\cdot\mathbf{x}_j}$, where $\mathbf{x}_j$ are the coordinates of the jth node and f^* is the complex conjugate of f. A complete description is presented in algorithm 7.

Another way to impose the Bloch-periodic conditions is to *assemble* the transformation matrix T and compute the matrix multiplications shown in (5.11). After applying this procedure the *image nodes* are removed from the system of equations and they are condensed in the correspondid reference nodes [11, 20], and the equation identifiers (equations enumeratino) have changed. The procedure to assemble the matrix T is:

- Find the new identifier for the reference nodes after eliminating the image nodes.
- For reference and interior nodes correspond a 1 in the matrix. The row is the current identifier and the column is the future one.

- For image nodes correspond the complex number $e^{ik_x(x_{\text{img}}-x_{\text{ref}})}e^{ik_y(y_{\text{img}}-y_{\text{ref}})}$ in the matrix; being $x_{\text{ref}}, x_{\text{img}}, y_{\text{ref}}, y_{img}$ the coordinates x, y for the reference an image nodes. The row is the identifier for that image node and the column is the future identifier for the corresponding reference node.
- The remaining entries of the matrix are zeros.

This procedure is equivalent to arrange the elementary row/column operations for the procedure described above in a matrix [21]. This procedure is described with further insight in Algorithm 8.

5.4.2 Real algebra implementation

Following [22], we can formulate the problem using just real algebra. This allow to implement the Bloch-periodicity in *standard finite element programs* like Abaqus/Calculix and FEAP [23–25], that do not allow to impose conditions with complex valued quantities. The idea is to duplicate the degrees of freedom, i.e., to have two meshes: one for real values and the other one for imaginary ones (see Figure 5.6)

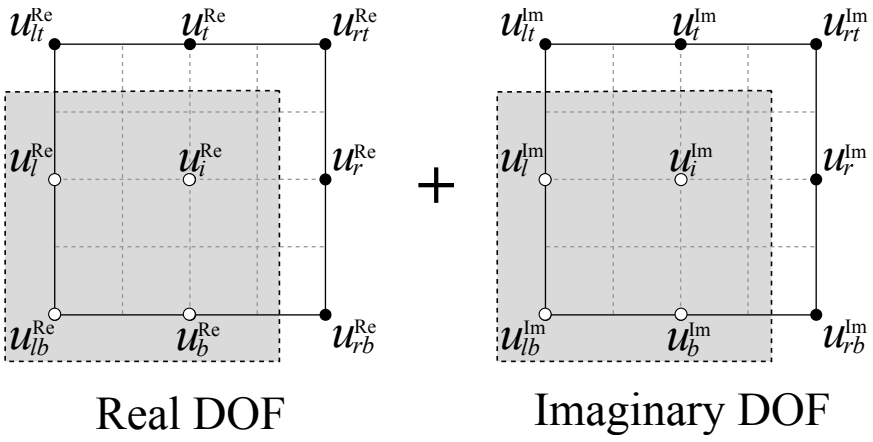


Figure 5.6. Mesh for the real algebra implementation.

We can express the values in boundary as

$$\begin{aligned} u_r(x_s) &= u_r^{\text{Re}}(x_s) + iu_r^{\text{Im}}(x_s) \\ t_r(x_s) &= t_r^{\text{Re}}(x_s) + it_r^{\text{Im}}(x_s) \quad , \end{aligned}$$

and replacing these in (5.5), we obtain

$$u_r^{\text{Re}}(x_s) = u_r^{\text{Re}}(x_s + a_s)\cos(k_s a_s) + u_r^{\text{Im}}(x_s + a_s)\sin(k_s a_s) \tag{5.13a}$$

$$u_r^{\text{Im}}(x_s) = u_r^{\text{Im}}(x_s + a_s)\cos(k_s a_s) - u_r^{\text{Re}}(x_s + a_s)\sin(k_s a_s) \quad , \tag{5.13b}$$

and

$$t_r^{\mathrm{Re}}(x_s + a_s) = -t_r^{\mathrm{Re}}(x_s)\cos(k_s a_s) + t_r^{\mathrm{Im}}(x_s)\sin(k_s a_s) \tag{5.14a}$$
$$t_r^{\mathrm{Im}}(x_s + a_s) = -t_r^{\mathrm{Im}}(x_s)\cos(k_s a_s) - t_r^{\mathrm{Re}}(x_s)\sin(k_s a_s) \ . \tag{5.14b}$$

In this case, the displacements and forces are arranged in block vectors as

$$\mathbf{U} = \left[\mathbf{U}^{\mathrm{Re}}\ \mathbf{U}^{\mathrm{Im}}\right]^T$$
$$\mathbf{F} = \left[\mathbf{F}^{\mathrm{Re}}\ \mathbf{F}^{\mathrm{Im}}\right]^T \ ,$$

with

$$\mathbf{U}^{\mathrm{Re}} = \left[\mathbf{u}_l^{\mathrm{Re}}\ \mathbf{u}_r^{\mathrm{Re}}\ \mathbf{u}_b^{\mathrm{Re}}\ \mathbf{u}_t^{\mathrm{Re}}\ \mathbf{u}_{lb}^{\mathrm{Re}}\ \mathbf{u}_{rb}^{\mathrm{Re}}\ \mathbf{u}_{lt}^{\mathrm{Re}}\ \mathbf{u}_{rt}^{\mathrm{Re}}\ \mathbf{u}_i^{\mathrm{Re}}\right]^T$$
$$\mathbf{U}^{\mathrm{Im}} = \left[\mathbf{u}_l^{\mathrm{Im}}\ \mathbf{u}_r^{\mathrm{Im}}\ \mathbf{u}_b^{\mathrm{Im}}\ \mathbf{u}_t^{\mathrm{Im}}\ \mathbf{u}_{lb}^{\mathrm{Im}}\ \mathbf{u}_{rb}^{\mathrm{Im}}\ \mathbf{u}_{lt}^{\mathrm{Im}}\ \mathbf{u}_{rt}^{\mathrm{Im}}\ \mathbf{u}_i^{\mathrm{Im}}\right]^T$$
$$\mathbf{F}^{\mathrm{Re}} = \left[\mathbf{f}_l^{\mathrm{Re}}\ \mathbf{f}_r^{\mathrm{Re}}\ \mathbf{f}_b^{\mathrm{Re}}\ \mathbf{f}_t^{\mathrm{Re}}\ \mathbf{f}_{lb}^{\mathrm{Re}}\ \mathbf{f}_{rb}^{\mathrm{Re}}\ \mathbf{f}_{lt}^{\mathrm{Re}}\ \mathbf{f}_{rt}^{\mathrm{Re}}\ \mathbf{f}_i^{\mathrm{Re}}\right]^T$$
$$\mathbf{F}^{\mathrm{Im}} = \left[\mathbf{f}_l^{\mathrm{Im}}\ \mathbf{f}_r^{\mathrm{Im}}\ \mathbf{f}_b^{\mathrm{Im}}\ \mathbf{f}_t^{\mathrm{Im}}\ \mathbf{f}_{lb}^{\mathrm{Im}}\ \mathbf{f}_{rb}^{\mathrm{Im}}\ \mathbf{f}_{lt}^{\mathrm{Im}}\ \mathbf{f}_{rt}^{\mathrm{Im}}\ \mathbf{f}_i^{\mathrm{Im}}\right]^T \ .$$

For the two dimensional example presented above the equations read

$$\begin{aligned}
\mathbf{u}_r^{\mathrm{Re}} &= \mathbf{u}_l^{\mathrm{Re}}\cos(\psi_x) - \mathbf{u}_l^{\mathrm{Im}}\sin(\psi_x)\\
\mathbf{u}_r^{\mathrm{Im}} &= \mathbf{u}_l^{\mathrm{Im}}\cos(\psi_x) + \mathbf{u}_l^{\mathrm{Re}}\sin(\psi_x)\\
\mathbf{u}_t^{\mathrm{Re}} &= \mathbf{u}_b^{\mathrm{Re}}\cos(\psi_y) - \mathbf{u}_b^{\mathrm{Im}}\sin(\psi_y)\\
\mathbf{u}_t^{\mathrm{Im}} &= \mathbf{u}_b^{\mathrm{Im}}\cos(\psi_y) + \mathbf{u}_b^{\mathrm{Re}}\sin(\psi_y)\\
\mathbf{u}_{rb}^{\mathrm{Re}} &= \mathbf{u}_{lb}^{\mathrm{Re}}\cos(\psi_x) - \mathbf{u}_{lb}^{\mathrm{Im}}\sin(\psi_x)\\
\mathbf{u}_{rb}^{\mathrm{Im}} &= \mathbf{u}_{lb}^{\mathrm{Im}}\cos(\psi_x) + \mathbf{u}_{lb}^{\mathrm{Re}}\sin(\psi_x)\\
\mathbf{u}_{lt}^{\mathrm{Re}} &= \mathbf{u}_{lb}^{\mathrm{Re}}\cos(\psi_y) - \mathbf{u}_{lb}^{\mathrm{Im}}\sin(\psi_y)\\
\mathbf{u}_{lt}^{\mathrm{Im}} &= \mathbf{u}_{lb}^{\mathrm{Im}}\cos(\psi_y) + \mathbf{u}_{lb}^{\mathrm{Re}}\sin(\psi_y)\\
\mathbf{u}_{rt}^{\mathrm{Re}} &= \mathbf{u}_{lb}^{\mathrm{Re}}\cos(\psi_x+\psi_y) - \mathbf{u}_{lb}^{\mathrm{Im}}\sin(\psi_x+\psi_y)\\
\mathbf{u}_{rt}^{\mathrm{Im}} &= \mathbf{u}_{lb}^{\mathrm{Im}}\cos(\psi_x+\psi_y) + \mathbf{u}_{lb}^{\mathrm{Re}}\sin(\psi_x+\psi_y) \ ,
\end{aligned}$$

where ψ_x and ψ_y are the phase shifts in x and y. Similar expressions hold for the forces. Although the expressions are written explicitly here, they are all contained compactly in (5.13) and (5.14). The phase shift is $\psi = k_j d_j$, with k_j the components of the wave vector and d_j the distance between the image and reference node in j direction.

Algorithm 7: Imposition of Bloch's conditions using elemental row operations.

Input : A: Original matrix
k_x, k_y: wave-vector components
nodes: array with nodal coordinates
image, reference: array with corresponding image and reference nodes
reference2: array with reference nodes without repetition

Output: Matrix A with imposed Bloch's conditions

for *i=1* **to** *n_img* **do** `// Phase shift for image nodes`

img_index ← image[i]
x_img ← nodes[img_index,1]
y_img ← nodes[img_index,2]
fac ← $e^{ik_x \mathrm{x\_img}} e^{ik_y \mathrm{y\_img}}$
A[img_index, :] ← A[img_index, :]fac
A[:, img_index] ← A[:, img_index]fac*

end

for *i=1* **to** *n_ref* **do** `// Phase shift for reference nodes`

ref_index ← reference2[i]
x_ref ← nodes[ref_index,1]
y_ref ← nodes[ref_index,2]
fac ← $e^{ik_x \mathrm{x\_ref}} e^{ik_y \mathrm{y\_ref}}$
A[ref_index, :] ← A[ref_index, :]fac
A[:, ref_index] ← A[:, ref_index]fac*

end

for *i=1* **to** *n_cond* **do** `// Addition of image row/columns to reference row/colums`

img_index ← image[i]
ref_index ← reference[i]
A[ref_index, :] ← A[ref_index, :] + A[img_index, :]
A[:, ref_index] ← A[:, ref_index] + A[:,img_index, :]

end

for *i=1* **to** *n_imag* **do** `// Elimination of image row/columns`

img_index ← image[i]
A[img_index, :] ← erase row img_index from A
A[img_index, :] ← erase column img_index from A

end

Algorithm 8: Impose Bloch-periodic conditions using tranformation matrices.

Input : A: Original matrix
k_x, k_y: wave-vector components
nodes: array with nodal coordinates
image, reference: array with corresponding image and reference nodes
reference2: array with reference nodes without repetition

Output: Matrix A with imposed Bloch's conditions

index_vec $\leftarrow \vec{0}$ `// Array with new equation identifiers`
for *i=1* **to** *ncond* **do**
  index_vec[image[i]] $\leftarrow$ - reference[i] `// Negative of the current identifier for the reference node`
end
for *i=1* **to** *ndof* **do** `// Assign new identifiers`

  cont $\leftarrow$ 0
  for *j=1* **to** *ncond* **do** `// How many image nodes before the current one`

    if *index_vec*[*i*] < 0 **then**
      if *image*[*j*] $<$ |*index_vec*[*i*]| **then**
        cont $\leftarrow$ cont + 1
      end
    else if *image*[*j*] $< i$ **then**
      cont $\leftarrow$ cont + 1
    end
  end
  if *index_vec*[*i*] < 0 **then** `// Assign new identifiers`

    index_vec[i] $\leftarrow$ index_vec[i] + cont
  else
    index_vec[i] $\leftarrow$ i - cont
  end
end
T $\leftarrow$ 0; index_ref $\leftarrow$ - index_vec
for *i=1* **to** *ndof* **do**
  if *index_vec*[*i*] > 0 **then**
    j $\leftarrow$ index_vec[i]
    T[i,j] $\leftarrow$ 1
  else
    j $\leftarrow$ |index_vec[i]|
    i_ref $\leftarrow$ index_ref[i]; i_img $\leftarrow$ i
    x_img $\leftarrow$ coords[i_img,1]; y_img $\leftarrow$ coords[i_img,2]
    x_ref $\leftarrow$ coords[i_ref,1]; y_ref $\leftarrow$ coords[i_ref,2]
    fac $\leftarrow$ $e^{i\psi_x(\text{x\_img - x\_ref})} e^{i\psi_y(\text{y\_img - y\_ref})}$
    T $\leftarrow$ fac
  end
end
A $\leftarrow$ T^HAT

Algorithm 9: Generate the list of Multipoint constraints for the Bloch-periodic imposition using real algebra.

Input : k_x, k_y, k_z: wave-vector components
nodes: array with nodal coordinates
image, reference: array with corresponding image and reference nodes
Output: List of Multipoint constraints

ncond ← Size of image
for *i=1* **to** *ncond* **do**
  x_img, y_img, z_img ← coordinates of image[i]
  x_ref, y_ref, z_ref ← coordinates of reference[i]
  $\psi \leftarrow k_x$(x_img - x_ref) + k_y(y_img - y_ref) + k_z(z_img - z_ref)
  $u^{\mathrm{Re}}_{\mathrm{img}} \leftarrow u^{\mathrm{Re}}_{\mathrm{ref}} \cos\psi + u^{\mathrm{Im}}_{\mathrm{ref}} \sin\psi$
  $u^{\mathrm{Im}}_{\mathrm{img}} \leftarrow u^{\mathrm{Im}}_{\mathrm{ref}} \cos\psi - u^{\mathrm{Re}}_{\mathrm{ref}} \sin\psi$
end

Chapter 6

Computational aspects

6.1 Python implementations

6.2 Commercial codes and user subroutines

Chapter 7

Nonlinear Equilibrium Equations in a Finite Element Discretization

7.1 General Equilibrium Equations

In this section we develop the incremental equations needed in a general non-linear problem. First, geometric non-linearities valid for any stress-strain pair are considered. Second, focus is shifted towards the particular case of the Second Piola-Kirchoff-Green Lagrange Strain pair.

Let represent the current (deformed) configuration of a deformable medium with equilibrium equations and boundary conditions

$$\sigma_{ij,j} + f_i = 0 \tag{7.1}$$

$$t_i = \sigma_{ij}\hat{n}_j \tag{7.2}$$

where represents the Cauchy definition of stress (or force per unit of deformed surface) and the corresponding tractions vector at a surface with outward normal . In the equilibrium equations stated in (1) the domain is unknown which makes the problem inherently non-linear. On the other hand this equilibrium statement is mathematically indeterminate since 9 unknown stress components must be solved out of 6 field equations. The indeterminacy is destroyed after the problem is kinematically described and connected to the stress field via constitutive modeling which can involve additional sources of non-linearity. To summarize the problem is non-linear since the domain is unknown, this is what we typically call a Geometric Non-linearity and will be reflected in the mathematical description of changes in configuration.

7.1.1 Lagrangian description of the equilibrium equations

The conceptual definition of Cauchy stress is the only one useful for the engineer since it describes forces per unit deformed surface. This validity is clearly identified in the fact that the equilibrium equations stated in (1) have been formulated in the stressed deformed domain where the body is in fact in equilibrium. The difficulty associated with the unknown domain may be dealt with in alternative ways. In the case of a solid body it is convenient to refer everything to the undeformed or reference configuration (with known domain) and proceed from there using linearization which corresponds to a Total Lagrangian (TL) approach. It is then useful to understand the transformation of the problem to via pullback operations. This process implies the introduction or consideration of mathematically defined stress definitions and newly developed kinematic descriptions. In this section we consider such transformations first from a dynamic point of view and later using thermodynamic principles we address the problem of identifying the proper strain measures.

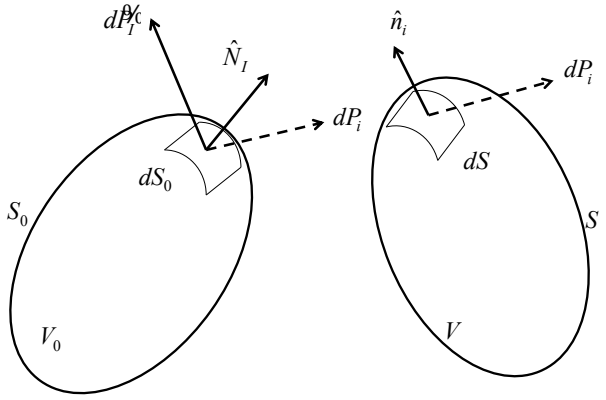


Figure 7.1. Definition of the natural domain

Nanson's formula

For the treatment that follows it will result useful to consider the relation between oriented differential surface elements in the reference and deformed configurations in the so-called Nanson's formula;

$$\hat{n}_i dS = \frac{\rho}{\rho_0} \hat{N}_I f_{Ii} dS_0 \tag{7.3}$$

Lagrangian stress definition

Let be the differential force associated to the Cauchy physical stress such

$$dP_i = t_i dS \equiv \sigma_{ji} n_j dS \tag{7.4}$$

Assume that can also be obtained in the undeformed configuration associated with a new traction definition;

$$dP_i = t_i^0 dS_0 \equiv T_{Ii}^0 N_I dS_0 \equiv \sigma_{ji} n_j dS \tag{7.5}$$

Using Nanson's formula it is possible to write;

$$T_{Ii}^0 N_I dS_0 \equiv \sigma_{ji} \frac{\rho_0}{\rho} N_I f_{Ij} dS_0 \tag{7.6}$$

Which simplifies into

$$T_{Ii}^0 = \frac{\rho_0}{\rho} f_{Ij} \sigma_{ji} \tag{7.7}$$

and this is the asymmetric First Piola-Kirchoff stress tensor.

Assume now that there is a pseudo-force in the undeformed configuration which results from a pullback operation on the physical force . Recalling the kinematic connection between the current and undeformed configurations

$$dX_I = f_{Ii} dx_i \tag{7.8}$$

where is the inverse deformation gradient, we can write for the force and pseudo-force vectors;

$$d\tilde{P}_I = f_{Ii} dP_i. \tag{7.9}$$

Now we can define a tractions vector in the undeformed configuration and associated with the pseudo-force like

$$d\tilde{P}_I = \tilde{t}_I dS_0 \equiv T_{JI} N_J dS_0 \tag{7.10}$$

where we have at the same time introduced the associated stress tensor. It then directly follows that;

$$T_{JI}N_J dS_0 = f_{Ii}dP_i \equiv f_{Ii}\sigma_{ji}n_j dS \tag{7.11}$$

once again using Nanson's formula we can write

$$T_{JI}N_J dS_0 = f_{Ii}\sigma_{ji}\frac{\rho_0}{\rho}N_J f_{Jj}dS_0 \tag{7.12}$$

and

$$T_{JI} = \frac{\rho_0}{\rho}f_{Jj}\sigma_{ji}f_{Ii} \tag{7.13}$$

which corresponds to the symmetric Second Piola-Kirchoff stress tensor.

For the derivations that follow and in the actual computational implementation it will be convenient to have the inverse relationships expressing the Cauchy stress tensor in terms of the First and Second Piola-Kirchoff stress definitions.

Once again recall that and use the definition found for the first PK stress tensor to write

7.2 Non-linear problems

In this section we will study the basic 1-single independent variable problem of numerically finding possible roots to the non-linear equation;

$$f(x) = 0 \tag{7.14}$$

and then we will proceed via generalization to the more realistic case of a system of non-linear equations of the form (29). This last general form will then be converted to the particular case of an equilibrated system of forces as encountered in the Finite Element Method or other numerical techniques appearing in the solution of continuum mechanics problems.

Since the problem is assumed non-linear the root finding process must use iterations. The general idea is that of establishing an initial trial solution and then proposed an efficient algorithm to improve the solution until some desired tolerance is satisfied.

In this section we will concentrate on Newton methods since most of the followed techniques in FEM analysis are either Newton or improved Newton Methods. A general and detailed survey of root finding and non-linear equation solving can be found in Press et al.

The basic idea of the Newton method is that of taking an initial guess ${}_iX$ (the trial solution), extending the tangent line at this point until it crosses zero and then taking as the next approximation ${}_{i+1}X$ the abscissa of that zero crossing, see fig. 7.2.

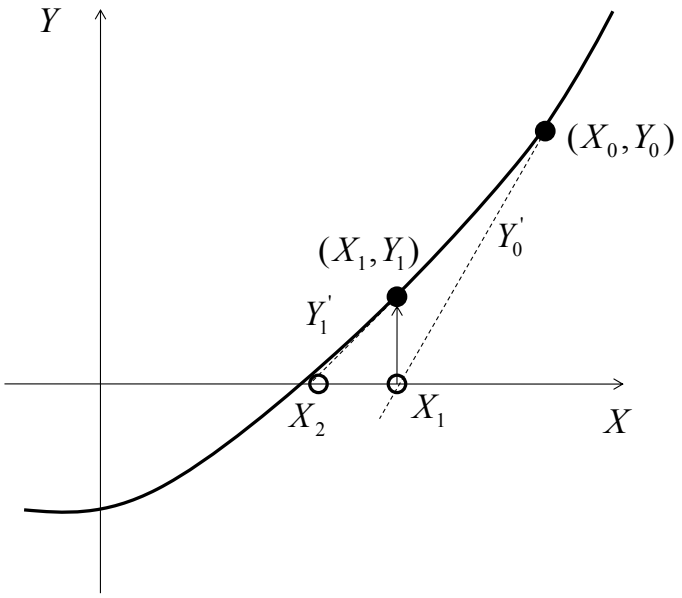


Figure 7.2. Fundamental Newton-Raphson iteration

The main disadvantage of the Newton method is that we formally need to obtain the function derivative $Y_i^{'}$ at every point.

Let δ be the correction to an approximation X to the actual root P such;

$$P = X + \delta \tag{7.15}$$

Using a Taylor series expansion for $f(P)$ in the neighborhood of X we have;

$$\begin{aligned} f(P) \equiv f(X+\delta) &= f(X) + f^{'}(X) + \frac{f^{''}(X)}{2}\delta^2 \\ f(P) \equiv f(X+\delta) &= f^{'}(X)\delta + \frac{f^{''}(X)}{2}\delta^2 \\ f(P) \equiv f(X+\delta) &= f(X) + f^{'}(X)(X-P) + \frac{f^{''}(X)}{2}(X-P)^2 \end{aligned} \tag{7.16}$$

Using $f(P) = 0$ and linearizing the Taylor series yields;

$$P \approx X - \frac{f(X)}{f'(X)} \tag{7.17}$$

From eq. (7.16) it is clear that the correction δ satisfies;

$$\delta = -\frac{f(X)}{f'(X)}$$

and we can write the general Newton-Raphson algorithm like;

$$\begin{aligned} {}_{i+1}X &= {}_{i}X + {}_{i}\delta \\ {}_{i}\delta &= -\frac{f({}_{i}X)}{f'({}_{i+1}X)} \end{aligned} \tag{7.18}$$

In an actual finite element algorithm we are to find an incrementally related history of discrete roots N_{roots} such

$${}^{I+1}X = {}^{I}X + {}^{I+1}\Delta X \tag{7.19}$$

and where

$$\begin{aligned} f({}^{I+1}X) &= 0 \\ f({}^{I}X) &= 0 \end{aligned} \tag{7.20}$$

Bibliography

[1] Bathe, Klaus Jurgen: *Finite Element Procedures*. Prentice Hall, 2nd edition, June 1995.

[2] Hibbitt, HD, BI Karlsson, and P Sorensen: *Abaqus theory manual, version 6.3*, 2006.

[3] Burden, R.L. and J.D. Faires: *Numerical Analysis*. Brooks/Cole, 9th edition, 2011.

[4] Press, William H: *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[5] Wikipedia: *Variational principle — wikipedia, the free encyclopedia*. `http://en.wikipedia.org/w/index.php?title=Variational_principle&oldid=646823082`, 2015. [Online; accessed 30-May-2015].

[6] Brillouin, Léon: *Wave propagation in periodic structures: electric filters and crystal lattices*. Courier Dover Publications, 2003.

[7] Geuzaine, C and J Remacle: *Gmsh Reference Manual. Gmsh 2.9: A Three-Dimensional Finite Element Mesh Generator With Built-in Pre-and Post-Processing Facilities*, April 2015. `http://geuz.org/gmsh/doc/texinfo/gmsh.pdf`.

[8] Meeker, David: *Finite Element Method Magnetics: Version 4.2 User's Manual*, 2010. `http://www.femm.info/Archives/doc/manual42.pdf`.

[9] Johnson, Steven G.: *Notes on the algebraic structure of wave equations*. Technical report, Massachusetts Institute of Technology, 2010.

[10] Kittel, Charles and Paul McEuen: *Introduction to solid state physics*, volume 8. Wiley New York, 1986.

[11] Langlet, Philippe: *Analyse de la Propagation des Ondes Acoustiques dans les Materiaux Periodiques a l'aide de la Methode des Elements Finis*. PhD thesis, L'Universite de Valenciennes et du Hainaut-Cambresis, 1993.

[12] Arfken, George B., Hans J. Weber, and Frank Harris: *Mathematical Methods for Physicists*. Academic Press, 6th edition, 2005.

[13] Sepúlveda, Alonso: *Física Matemática*. Editorial Universidad de Antioquia, 1st edition, 2009.

[14] Achenbach, J.D.: *Wave Propagation in Elastic Solids*. North Holland Publishing Company, 1973.

[15] Reddy, J.N.: *Applied Functional Analysis and Variational Methods in Engineering*. Krieger Publishing, 1st edition, 1991.

[16] Kreyzsig, Erwin: *Introductory Functional Analysis with Applications*. Wiley, 1st edition, 1989.

[17] Gockenbach, Mark S.: *Partial Differential Equations: Analytical and Numerical Methods*. SIAM, 2002.

[18] Sukumar, N. and J. E. Pask: *Classical and enriched Finite element formulations for Bloch-periodic boundary conditions*. International Journal of Numerical Methods in Engineering, 77(8):1121–1138, February 2009.

[19] Guarín-Zapata, Nicolás: *Simulación Numérica de Problemas de Propagación de Ondas: Dominios Infinitos y Semi-infinitos*. Master's thesis, Universidad EAFIT, 2012.

[20] Hladky-Hennion, Anne Christine and Jean Noël Decarpigny: *Analysis of the scattering of a plane acoustic wave by a doubly periodic structure using the finite element method: Application to Alberich anechoic coatings*. J. Acoust. Soc. Am, 90:3356–3367, 1991.

[21] Poole, David: *Álgebra lineal: una introducción moderna*. Cengage Learning Editores, 2nd edition, 2007. ISBN 9706865950.

[22] Aberg, M. and P. Gudmundson: *The usage of standard finite element codes for computation of dispersion relations in materials with periodic microstructure*. Journal of the Acoustical Society of America, 102:2007–2013, 1997.

[23] Taylor, RL: *FEAP–A Finite Element Analysis Program, Version 8.4 User Manual*, 2013.

[24] Hibbit, Karlsson & Sorensen: *ABAQUS User's Manual: Version 6.12*, 2014.

[25] Dhondt, Guido: *CalculiX CrunchiX User's Manual, version 2.7*, 2014.

[26] Geuzaine, Christophe and Jean François Remacle: *Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities*. International Journal for Numerical Methods in Engineering, 79(11):1309–1331, 2009. `http://geuz.org/gmsh/doc/preprints/gmsh_paper_preprint.pdf`.