Sebastian GEGENLEITHNER, BSc

Development of a Finite Element Program for 2-D Gravity Dam Analyses

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Civil Engineering Sciences, Geotechnics and Hydraulics

submitted to

Graz University of Technology

Supervisor

Univ.-Prof. Dipl.-Ing. Dr.techn. Gerald ZENZ

Graz, Mai 2017

AFFIDATIV

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis.

Date	Signature

Acknowledgements

First of all I want to express my gratitude to my colleague and friend Shervin Shariari, who provided me with this interesting topic and always supported me during the process of this thesis. Additionally I have to thank all the people, who piqued my interest in the topic of hydraulic engineering, especially my supervisor Prof. Gerald Zenz.

Furthermore I want to thank my sister, Lena, my parents, Renate and Christoph, my grandparents, all my relatives and friends for supporting me during my studies.

But my biggest thanks belongs to my girlfriend Eva, who was understanding and supportive throughout the process of this thesis.

Abstract

Within the framework of this thesis a software for the analysis of concrete gravity dams is developed. The main goals are simplicity, dependency on just open-source software and possibility of future implementations. Another important aim is the use of numerical methods, more specifically the Finite Element Method (FEM). The software is named 2DGDA, which stands for two dimensional gravity dam analyses. The program is designed to deal with a certain type of dam structures, which allow the following simplifications

- Pseudo-static analyses
- Westergaard's added mass approach
- Plane strain/stress
- Linear elastic material behaviour
- Rigid foundation

The software is validated using ANSYS® Academic Research, Release 17.2, more specifically two element types available in ANSYS® Workbench. The first element type is the plane 13 element, which is a linear quadrilateral element with basic FEM formulations. The second one is a higher order quadrilateral, using advanced FEM techniques, which is called plane 183. The validation shows that for the plane 13 element 2DGDA reproduces the exact same values as ANSYS®. Even for the plane 183 element the error is negligible for fine meshes.

In conclusion this program provides a good alternative to other available numerical packages for the computation of two dimensional gravity dam cases. Keeping in mind the program's limitations allows the user to perform Finite Element computations with an easy to use software.

Kurzfassung

Im Rahmen dieser Arbeit wird eine Software zur Bemessung von Gewichtsstaumauern entwickelt. Der Fokus liegt hierbei auf Einfachheit, ausschließlicher Abhängigkeit von Open-source Software und der Möglichkeit für zukünftige Weiterentwicklungen. Eine weitere Grundlage ist die Verwendung von Numerischen Methoden, respektive der Finite Elemente Methode (FEM) mit Hilfe isoparametrischer Formulierung. Die Software trägt den Namen 2DGDA (**T**wo **D**imensional **G**ravity **D**am **A**nalyses). Das entwickelte Programm ist für Dämme, die folgende Vereinfachungen zulassen, ausgelegt

- Pseudo-statische Analysen
- Vereinfachungen des Hydrodynamischen Lastfalles unter Verwendung von zusätzlichen Massen nach Westergaard
- Ebene Dehnungen oder Spannungen
- Elastisches Materialverhalten
- Starres Fundament

Die Validierung der Software wird mit ANSYS® Academic Research, Release 17.2 durchgeführt. Hierzu werden die Ergebnisse von zwei verschiedenen, in ANSYS® zur Verfügung stehenden Elementtypen mit den Ergebnissen von 2DGDA verglichen. Beim ersten Typ, dem „plane 13“ Element, handelt es sich um ein lineares Viereck-Element mit einfacher FEM Formulierung. Der zweite Typ, das „plane 183“ Element, ist ein Viereck-Element höherer Ordnung. Die Vergleiche zeigen, dass das „plane 13“ Element und 2DGDA exakt dieselben Werte liefern. Für feinere Berechnungsnetze wird der Fehler, verglichen mit dem „plane 183“ Element, vernachlässigbar klein.

Zusammengefasst bietet das Programm eine gute Alternative zu vorhandener Software für die Berechnung von 2-D Gewichtsstaumauern. Unter Berücksichtigung der Einschränkungen des Programmes können Finite Elemente Simulationen zur Ermittlung der Bauwerksbeanspruchung durchgeführt werden.

Contents

Nomenclature

$\boldsymbol{\sigma}$	Stress vector
$\boldsymbol{\varepsilon}$	Strain vector
$\boldsymbol{B}$	Strain-displacement matrix
$\boldsymbol{C}$	Stiffness tensor
$\boldsymbol{D}$	Operator matrix
$\boldsymbol{f}'$	Force vector in a local, with the element aligned, coordinate system
$\boldsymbol{F}$	Global force vector
$\boldsymbol{f}$	Local force vector
$\boldsymbol{f}_B$	Body force vector
$\boldsymbol{J}$	Jacobian
$\boldsymbol{K}$	Global stiffness matrix
$\boldsymbol{k}$	Local stiffness matrix
$\boldsymbol{m}$	Local mass matrix
$\boldsymbol{u}$	Displacement vector
$\ddot{\boldsymbol{u}}$	Acceleration vector
γ_{am}	Added mass per unit area
γ_{xy}	Shear strain in xy plane
γ_{xz}	Shear strain in xz plane
γ_{yz}	Shear strain in yz plane
ν	Poisson's ratio
ν^*	Adapted Poisson's ratio
Ω^e	Element domain
ρ_c	Specific weight of concrete
ρ_w	Specific weight of water

σ_x	Stresses in x direction
σ_y	Stresses in y direction
σ_z	Stresses in z direction
τ_{xy}	Shear stresses in the xy plane
τ_{xz}	Shear stresses in the xz plane
τ_{yz}	Shear stresses in the yz plane
ε_x	Strain in x direction
ε_y	Strain in y direction
ε_z	Strain in z direction
A	Area of the element
a_x	Earthquake acceleration in x direction
a_y	Earthquake acceleration in y direction
E	Elastic modulus
E^*	Adapted elastic modulus
f_x	Body forces in x direction
f_y	Body forces in y direction
h	Distance from the bottom of the reservoir to the water surface
L	Length of the element edge
m	Point mass
n_{GP}	Number of Gauss points
$p\,(x)$	Loading function in a global coordinate system
$p\,(x_n)'$	Loading function in local coordinates (aligned with the element)
t	Thickness of the element
u_x	Discplacement in x direction
u_y	Discplacement in y direction

1 Introduction

Constructing dams is one of the oldest and most important disciplines in civil engineering. The main purpose of these type of structures is to provide safe storage and retention of water. More specifically, depending on the technical and environmental requirements, they can be used for flood retention, energy generation, water supply, etc. Dependent on the construction material used, dams can be categorized into two types. The first type are the embankment dams, which are constructed from rockfill and/or earthfill material. The second type are the concrete dams, which include gravity dams, arch dams and buttress dams.

Often failure of such a dam can lead to large number of casualties and property damages. Therefore its safety must be ensured by careful design and the use of state of the art calculation techniques. Compared to conventional computation methods, like rigid block motion, nowadays numerical methods gain importance in dam engineering. Mostly because of the rapid increase of computational power and improvement of numerical methods. Examples for numerical methods are the Finite Element Method (FEM), the Finite Difference Method (FDM), Meshfree methods, etc. In civil engineering the FEM is by far the most often used method to solve for unknown displacements, strains and stresses in the structure.

Commercial codes, frequently used in dam engineering, come with a large range of applications. They are capable of considering all kind of influences, like cracking, foundation interaction or material non-linearities, just to name a few. Although this may be necessary in some cases, like complicated arch dams, for preliminary design of gravity dams simplifications can be made. Those simplifications are:

- Pseudo-static analyses
- Added mass approaches
- Plane strain/stress
- Linear elastic material behaviour
- Rigid foundation

For such specific cases designing the structure with commercial software is possible but it comes with major drawbacks. Usually these codes are associated with high licence costs and a steep learning curve. Furthermore they are not open-source, which is not desirable for academic use, since further implementations are not possible. During the author's research it was found that in fact very few open-source options for designing such simple cases are available. This was the motivation to create such a software within the framework of this thesis. The main aim is to provide an open-source code for the computation of simple two dimensional (2-D) gravity dams, based on the above stated simplifications, with the possibility of future implementations. Furthermore it should be easy to use and only be dependent on other open-source software. The software was named 2DGDA, which stands for Two Dimensional Gravity Dam Analyses. Additionally to the above stated simplifications, this program uses 2-D linear quadrilateral elements, with basic FEM oriented formulations.

This thesis is divided into six chapters, wherein chapter 1 is the introduction and chapter 2 gives a general overview of the FEM techniques used in this thesis. The implementation is explained in chapter 3. Furthermore a user manual is given in chapter 4, where the features of 2DGDA as well as ParaView are explained. ParaView is an open-source graphic processing tool and is used for post-processing the data in this thesis. The validation of the program is carried out in chapter 5, wherein 2DGDA's results are compared to values computed with ANSYS®. Two famous validation cases have been used, namely the Koyna Dam and the Pine Flat Dam.

Prior to using this application, the user has to confirm that the case allows all mentioned simplifications and does not exceed the software's limitations stated in chapter 4.

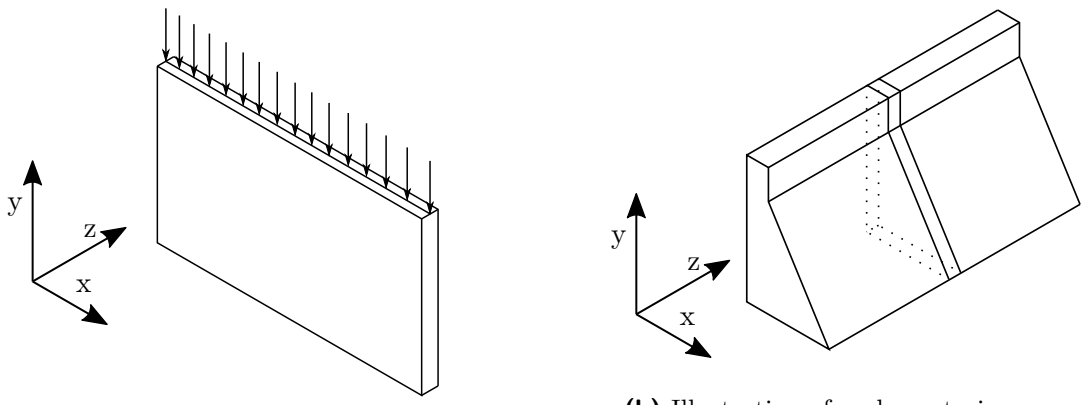
2 Introduction to the Finite Element Method

The starting point of any FEM simulation is a given Boundary Value Problem (BVP), which is defined by its domain, its boundary conditions and the governing equations presented in section 2.1. More specifically, in structural mechanics the domain is the structure, the boundary conditions are loadings and supports, and the governing equations describe equilibrium, material behaviour and kinematic relations. The governing equations are a set of partial differential equations (PDEs), which can be solved analytically for simple cases, such as cantilever beams. For more complicated cases, obtaining an analytical solution is not possible anymore, thus numerical methods have to be used.

To obtain a numerical solution the strong form has to be translated to a weak form. The most commonly used procedure is the Bubnov-Galerkin method. The system matrices are calculated using isoparametric formulations. The FEM theory in this chapter is a compendium of the following literature: Gould (1994), Anandarajah (2011), Rüberger & Zechner (2008), Bathe (2006), Bower (2009), Ibrahimbegovic (2009), Kattan (2010) and Center of Aerospace Structures (2016).

2.1 Governing equations for 2-D plane strain and plane stress

In some cases a three dimensional (3-D) elasticity problem can be simplified to 2-D. Depending on the problem a plane strain or a plane stress approach can be used. Generally plane stress is applicable for problems in which one side of the body is much smaller than the other two (Fig. 2.1a). Compared to that, the plane strain approach is suited for models with small cross sections, but large expansions in one direction, such as tunnels or dams (Fig. 2.1b).



(a) Illustration of a plane stress case

(b) Illustration of a plane strain case

Fig. 2.1: Illustration plane stress/strain

2.1.1 Governing equations for plane stress

Assuming an arbitrary geometry, simplyfied by a plane stress assumption, a differential part can be cut out and the Governing Equations can be set up. Such a part is illustrated in Fig. 2.2.

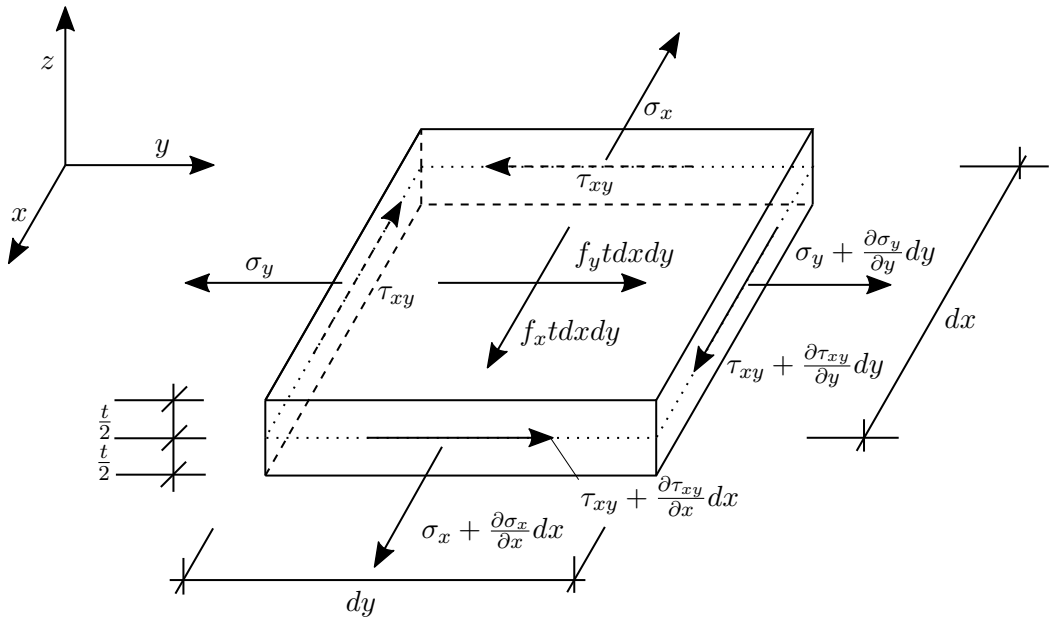


Fig. 2.2: Differential part of a geometry under plane stress conditions

For plane stress conditions the stresses in one direction are assumed to be zero. Furthermore the loading has to be parallel to the structure. Setting the stresses in z direction to zero yields the following expression

$$\sigma_z = \tau_{xz} = \tau_{yz} = 0 \tag{2.1}$$

Since the stresses in the partial element are not constant in x and y direction, a variation of the stress terms has to be considered. This is done by using the Taylor expansion series. Since dx is small, just the first two terms are taken into account (higher order terms are negligible). Also body forces f_x and f_y, acting on the element, have to be considered. Summing up all the forces on the partial element leads to

$$\begin{aligned}
\left(\sigma_x + \frac{\partial \sigma_x}{\partial x} dx\right) dy\, t - \sigma_x dy\, t + \left(\tau_{xy} + \frac{\partial \tau_{xy}}{\partial y} dy\right) dy\, t - \tau_{xy} dx\, t + f_x dx dy\, t = 0 \\
\left(\sigma_y + \frac{\partial \sigma_y}{\partial y} dy\right) dx\, t - \sigma_y dx\, t + \left(\tau_{xy} + \frac{\partial \tau_{xy}}{\partial x} dx\right) dx\, t - \tau_{xy} dy\, t + f_y dx dy\, t = 0
\end{aligned} \tag{2.2}$$

Simplifying Eqs. 2.2 yield the equilibrium equations as

$$\begin{aligned}
\frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + f_x = 0 \\
\frac{\partial \sigma_y}{\partial y} + \frac{\partial \tau_{xy}}{\partial x} + f_y = 0
\end{aligned} \tag{2.3}$$

For small strains the relation between displacements and strains is given by the kinematic equations as

$$\begin{aligned} \varepsilon_x &= \frac{\partial u_x}{\partial x} \\ \varepsilon_y &= \frac{\partial u_y}{\partial y} \\ \gamma_{xy} &= \frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x} \end{aligned} \tag{2.4}$$

in which ε_x and ε_y are the strains in x and y direction and γ_{xy} is the shear strain. Displacements are designated as u_x and u_y.

The last set of equations are the material-law equations, also known as constitutive relations. For elastic material, stresses are directly related to strains. If the relation between stresses and strains is linear, Hooke's law is valid. For a linear elastic, isotropic material the strain-displacement relations can be written as

$$\begin{aligned} \sigma_x &= \frac{E}{1-\nu^2}\left(\varepsilon_x + \nu\varepsilon_y\right) \\ \sigma_y &= \frac{E}{1-\nu^2}\left(\varepsilon_y + \nu\varepsilon_x\right) \\ \tau_{xy} &= \frac{E}{2\left(1+\nu\right)}\gamma_{xy} \end{aligned} \tag{2.5}$$

where E is the elastic modulus and ν stands for Poisson's ratio. For further proceedings it makes sense to express the governing equations in vector notation. The equilibrium equations, Eqs. 2.3, then become

$$\underbrace{\begin{pmatrix} \partial_x & 0 & \partial_y \\ 0 & \partial_y & \partial_x \end{pmatrix}}_{\boldsymbol{D}^T} \underbrace{\begin{pmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{pmatrix}}_{\boldsymbol{\sigma}} = \underbrace{\begin{pmatrix} f_x \\ f_y \end{pmatrix}}_{\boldsymbol{f}_B} \tag{2.6}$$

where $\boldsymbol{D}$ is an operator matrix, $\boldsymbol{\sigma}$ is the stress vector and $\boldsymbol{f}_B$ is the body force vector. The kinematic set of equations, Eqs. 2.4, can be rewritten as

$$\underbrace{\begin{pmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{pmatrix}}_{\boldsymbol{\varepsilon}} = \underbrace{\begin{pmatrix} \partial_x & 0 \\ 0 & \partial_y \\ \partial_y & \partial_x \end{pmatrix}}_{\boldsymbol{D}} \underbrace{\begin{pmatrix} u_x \\ u_y \end{pmatrix}}_{\boldsymbol{u}} \tag{2.7}$$

in which $\boldsymbol{\varepsilon}$ is the strain vector and $\boldsymbol{u}$ is the displacement vector. Furthermore the Constitutive equations, Eqs. 2.5, expressed by the stiffness tensor $\boldsymbol{C}$, are given as

$$\underbrace{\begin{pmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{pmatrix}}_{\boldsymbol{\sigma}} = \underbrace{\frac{E}{1-\nu^2}\begin{pmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{pmatrix}}_{\boldsymbol{C}} \underbrace{\begin{pmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{pmatrix}}_{\boldsymbol{\varepsilon}} \tag{2.8}$$

2.1.2 Governing equations for plane strain

For the case of plane strain, normal and shear strains in z direction are assumed to be zero, which is given by the following expression

$$\varepsilon_z = \gamma_{xz} = \gamma_{yz} = 0 \tag{2.9}$$

For linear, isotropic material the governing equations for plane strain are very similar to those of plane stress. In fact only the constitutive equations change. For plane strain conditions an adapted elastic modulus, E^*, and Poisson's ratio, ν^*,have to be considered. Similar to Eq. 2.8 the adapted constitutive equation looks like

$$\underbrace{\begin{pmatrix}\sigma_x\\ \sigma_y\\ \tau_{xy}\end{pmatrix}}_{\boldsymbol{\sigma}} = \underbrace{\frac{E^*}{1-\nu^{*2}}\begin{pmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu^*}{2}\end{pmatrix}}_{\boldsymbol{C}}\underbrace{\begin{pmatrix}\varepsilon_x\\ \varepsilon_y\\ \gamma_{xy}\end{pmatrix}}_{\boldsymbol{\varepsilon}} \tag{2.10}$$

where

$$E^* = \frac{E}{1-\nu} \tag{2.11}$$

and

$$\nu^* = \frac{\nu}{1-\nu} \tag{2.12}$$

Substituting Eq. 2.11 and Eq. 2.12 into Eq. 2.10 leads to the following constitutive relation for plane strain conditions

$$\underbrace{\begin{pmatrix}\sigma_x\\ \sigma_y\\ \tau_{xy}\end{pmatrix}}_{\boldsymbol{\sigma}} = \underbrace{\frac{E}{(1+\nu)(1-2\nu)}\begin{pmatrix}1-\nu & \nu & 0\\ \nu & 1-\nu & 0\\ 0 & 0 & \frac{1}{2}(1-2\nu)\end{pmatrix}}_{\boldsymbol{C}}\underbrace{\begin{pmatrix}\varepsilon_x\\ \varepsilon_y\\ \gamma_{xy}\end{pmatrix}}_{\boldsymbol{\varepsilon}} \tag{2.13}$$

Eq. 2.6, 2.7 and 2.13 are known as the governing equations for plane strain.

2.2 From strong to weak form

The governing equations from section 2.1 describe the strain, stress and displacement fields at any arbitrary point in the domain Ω, thus they are called strong form. For solving a complicated BVP, the strong form has to be replaced with a weak form (integral formulation). This can be done by using the Bubnov-Galerkin Finite Element Method, which approximates the displacement field by a combination of linear functions, the so-called ansatz or shape functions. Under consideration of a discretized domain and boundary conditions, the following linear system of equations can be obtained

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{F} \tag{2.14}$$

in which $\boldsymbol{K}$ is the global stiffness matrix, $\boldsymbol{u}$ are the nodal displacements (unknown) and $\boldsymbol{F}$ is the global force vector. $\boldsymbol{K}$ and $\boldsymbol{F}$ are computed using the isoparametric formulation. For the bilinear quadrilateral element this is explained in section 2.4.

2.3 Discretization of the domain - meshing

Discretizing the domain into Finite Elements, called mesh, is the first step of every FEM simulation. The quality of the computed values depend highly on the generated mesh. A high quality mesh should be fine enough (depending on the order of the element), without exceeding the computational limits. Furthermore it is important to additionally refine the mesh at regions with a high or low gradient of stresses or strains. Such areas are for example corners, changes in material properties, etc.

In the present work the isoparametric formulation is used, thus all the elements in the domain are mapped from a certain reference element. Fig. 2.3 gives an overview of the most important 2-D reference elements. As for the quadrilateral elements just the Lagrange class is illustrated. The Serendipity class looks similar but with the midside nodes dropped. In this thesis linear, quadrilateral elements are used.

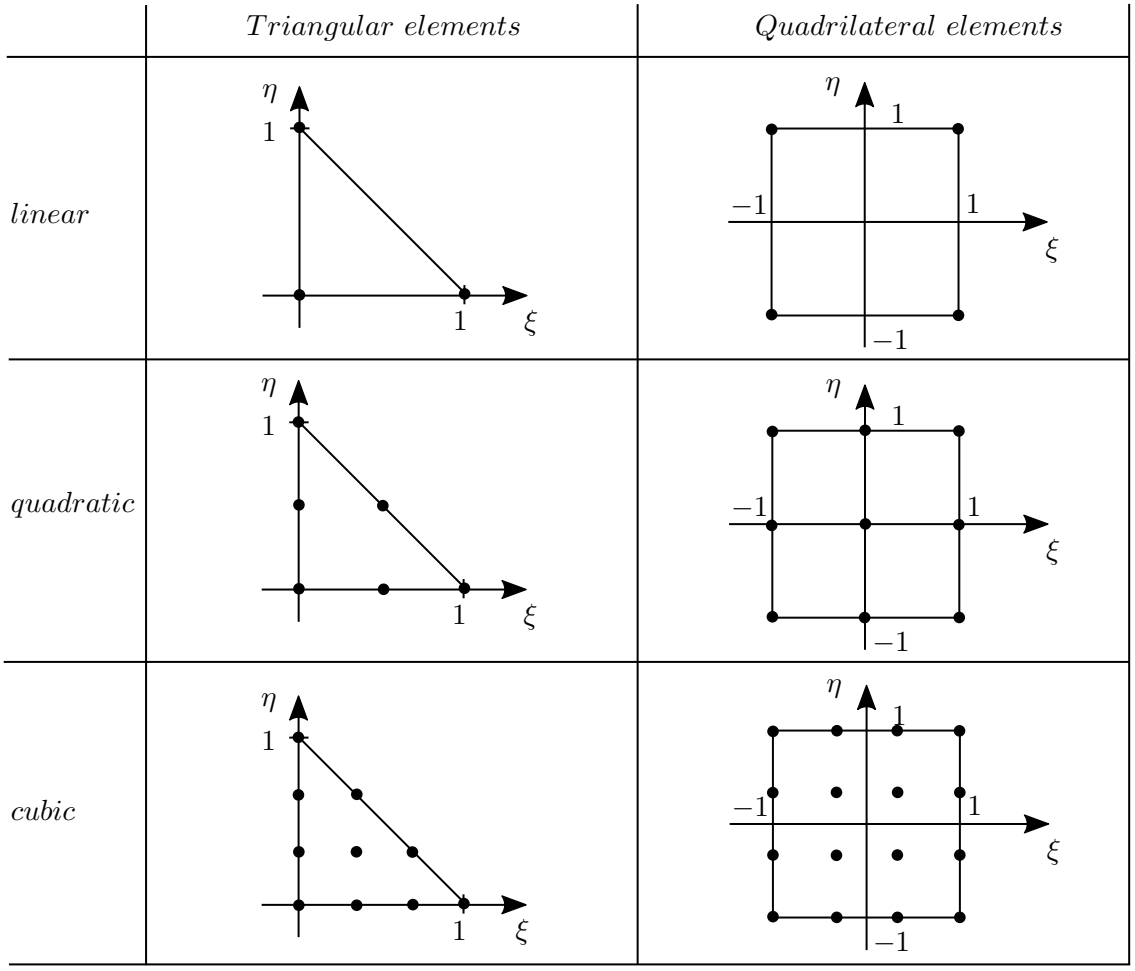


Fig. 2.3: Important 2-D reference elements

2.4 Isoparametric formulation for the 2-D bilinear quadrilateral

Compared to the parametric formulation, which uses two sets of different shape functions to interpolate the geometry and the unknowns (in structural mechanics displacements), the isoparametric formulation uses the same shape functions. This has major advantages, thus the isoparametric formulation is widely used. In this section the isoparametric formulation for the 2-D bilinear quadrilateral element is explained.

The integrations are carried out numerically, using a 2x2 Gauss rule. For the linear quadrilateral reference element (illustrated in Fig. 2.3) the locations of the Gauss points are at $(-\sqrt{\frac{1}{3}}, -\sqrt{\frac{1}{3}})$, $(\sqrt{\frac{1}{3}}, -\sqrt{\frac{1}{3}})$, $(-\sqrt{\frac{1}{3}}, \sqrt{\frac{1}{3}})$ and $(\sqrt{\frac{1}{3}}, \sqrt{\frac{1}{3}})$ distance to the coordinate zero. The integration weights are 1.

2.4.1 Elemental stiffness matrix

As mentioned the isoparametric formulation uses the same shape functions for geometric and displacement interpolation. The geometric interpolation is given as

$$\begin{aligned} x &= \sum_{i=1}^{n} N_i x_i \\ y &= \sum_{i=1}^{n} N_i y_i \end{aligned} \tag{2.15}$$

in which x and y are the coordinates at any arbitrary point at the element and x_i and y_i are the coordinates at the nodes n. Similar to Eq. 2.15 the displacement interpolation is given as

$$\begin{aligned} u_x &= \sum_{i=1}^{n} N_i u_{xi} \\ u_y &= \sum_{i=1}^{n} N_i u_{yi} \end{aligned} \tag{2.16}$$

for the bilinear quadrilateral element, used in this thesis, the shape functions are

$$\begin{aligned} N_1 &= \frac{1}{4}(1-\xi)(1-\eta) \\ N_2 &= \frac{1}{4}(1+\xi)(1-\eta) \\ N_3 &= \frac{1}{4}(1+\xi)(1+\eta) \\ N_4 &= \frac{1}{4}(1-\xi)(1+\eta) \end{aligned} \tag{2.17}$$

First the strain-displacement matrix, $\boldsymbol{B}$, has to be computed. This matrix describes the relation between strains and displacements and is given as

$$\boldsymbol{\varepsilon} = \underbrace{\begin{pmatrix} \frac{\partial N_1^e}{\partial x} & 0 & \frac{\partial N_2^e}{\partial x} & 0 & \frac{\partial N_3^e}{\partial x} & 0 & \frac{\partial N_4^e}{\partial x} & 0 \\ 0 & \frac{\partial N_1^e}{\partial y} & 0 & \frac{\partial N_2^e}{\partial y} & 0 & \frac{\partial N_3^e}{\partial y} & 0 & \frac{\partial N_4^e}{\partial y} \\ \frac{\partial N_1^e}{\partial y} & \frac{\partial N_1^e}{\partial x} & \frac{\partial N_2^e}{\partial y} & \frac{\partial N_2^e}{\partial x} & \frac{\partial N_3^e}{\partial y} & \frac{\partial N_3^e}{\partial x} & \frac{\partial N_4^e}{\partial y} & \frac{\partial N_4^e}{\partial x} \end{pmatrix}}_{\boldsymbol{B}} \mathbf{u} \tag{2.18}$$

in which $\boldsymbol{\varepsilon}$ is the strain vector and $\boldsymbol{u}$ contains the elemental nodal displacements. To be able to compute $\boldsymbol{B}$, the shape function derivatives, with respect to the real coordinates, are required. Those derivatives can be calculated using the chain rule

$$\begin{aligned} \frac{\partial N_i^e}{\partial x} &= \frac{\partial N_i^e}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial N_i^e}{\partial \eta}\frac{\partial \eta}{\partial x} \\ \frac{\partial N_i^e}{\partial y} &= \frac{\partial N_i^e}{\partial \xi}\frac{\partial \xi}{\partial y} + \frac{\partial N_i^e}{\partial \eta}\frac{\partial \eta}{\partial y} \end{aligned} \tag{2.19}$$

Writing Eqs. 2.19 in matrix form leads to

$$\begin{pmatrix} \frac{\partial N_i^e}{\partial \xi} \\ \\ \frac{\partial N_i^e}{\partial \eta} \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{pmatrix}}_{\boldsymbol{J}} \begin{pmatrix} \frac{\partial N_i^e}{\partial x} \\ \\ \frac{\partial N_i^e}{\partial y} \end{pmatrix} \tag{2.20}$$

in which $\boldsymbol{J}$ is the Jacobi-matrix or Jacobian. This operator relates the real coordinate derivatives to the reference coordinate derivatives. The mapping can be carried out both ways by either using $\boldsymbol{J}$ or $\boldsymbol{J}^{-1}$. Now the stiffness matrix on element level can be formulated as

$$\boldsymbol{k} = \int_{\Omega^e} \boldsymbol{B}^T \boldsymbol{C} \boldsymbol{B} \, d\Omega^e \tag{2.21}$$

in which $\boldsymbol{C}$ describes the constitutive relations and is given by Eq. 2.8 for plane stress and by Eq. 2.13 for plane strain. The integration is done in the reference coordinate system, therefore $d\,\Omega^e$ writes as

$$d\Omega^e = t \, det(\boldsymbol{J}) \, d\xi \, d\eta \tag{2.22}$$

in which t is the elemental thickness. Now Eq. 2.21 can be rewritten as

$$\boldsymbol{k} = t \int_{-1}^{1} \int_{-1}^{1} \boldsymbol{B}^T \boldsymbol{C} \boldsymbol{B} \, det(\boldsymbol{J}) \, d\xi \, d\eta \tag{2.23}$$

The integration is carried out with a 2x2 Gauss rule.

2.4.2 Computing the local force vector

Like the stiffness matrix, computed using Eq. 2.23, the force vector is also calculated on element level first. The applied forces can be divided into acceleration induced loading, called body forces and distributed loads.

Body forces

Body forces, introduced by acceleration, are obtained by

$$\boldsymbol{f} = \boldsymbol{m}\,\ddot{\boldsymbol{u}} \tag{2.24}$$

where $\boldsymbol{f}$ is the local force vector, $\boldsymbol{m}$ is the mass matrix on element level and $\ddot{\boldsymbol{u}}$ is the acceleration vector. For computing the mass matrix two different methods are explained in more detail.

The first one is the direct mass method, in which the total mass of an element is computed and distributed evenly to the element nodes. The outcome is a diagonally lumped mass matrix (DLMM). Fig 2.4 shows a solid element with area A, thickness t and density ρ. For this element the total mass can be computed as

$$m = \rho\, A\, t \tag{2.25}$$

Dividing the resulting mass by the number of nodes yields lumped masses, connected with a wire frame.

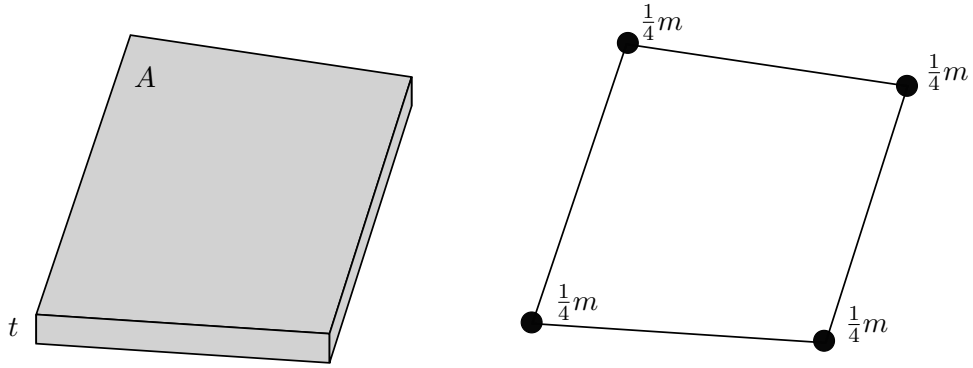


Fig. 2.4: Mass contribution of a 2-D element

The second method is the variational mass lumping. It is based on a variational principle. The mass matrix on element level is given as

$$\boldsymbol{m} = \rho \int_{\Omega^e} \boldsymbol{N}^T \boldsymbol{N}\, d\Omega \tag{2.26}$$

If the same shape functions as in section 2.4.1 are used, this mass matrix is called consistent mass matrix (CMM).

Distributed loads

Distributed loads act on the element edge, thus a 1-D isoparametric representation can be applied. From the virtual work, done by the line load, the force vector on element level yields

$$\boldsymbol{f} = \int_{-1}^{1} \boldsymbol{N}^T p\,(x)\,\boldsymbol{J}\,d\xi \tag{2.27}$$

in which $p\,(x)$ is the loading function on the element edge in force-per-unit-length, $\mathbf{J}$ is the Jacobian which is given as

$$\boldsymbol{J} = \frac{dx}{d\xi} = \frac{L}{2} \tag{2.28}$$

where L is equal to the total length of the element (edge) and $\boldsymbol{N}$ are the 1-D shape functions of the form

$$\begin{aligned} N_1 &= \frac{1-\xi}{2} \\ N_2 &= \frac{1+\xi}{2} \end{aligned} \tag{2.29}$$

Both distributed loads used in this thesis, hydrostatic and hydrodynamic pressure, have a most efficient way to set up their loading function (explained in section 3.8.1 and 3.8.2). Therefore it makes sense to compute the force vector in a local coordinate system first (fitted for the respective loading case). Considering this, Eq. 2.27, for a force vector in local x'-y' coordinates, can be rewritten as

$$\boldsymbol{f}\,' = \int_{-1}^{1} \boldsymbol{N}^T p\,(x)'\,\boldsymbol{J}\,d\xi \tag{2.30}$$

in which $p\,(x)'$ is the loading function in local coordinates. After computing $\boldsymbol{f}'$ from Eq. 2.30, the global force vector on element level, $\boldsymbol{f}$, can be obtained by performing a coordinate transformation. Fig. 2.5 illustrates the mapping from the reference coordinate to the local coordinate system and then the coordinate transformation to global x-y coordinates.

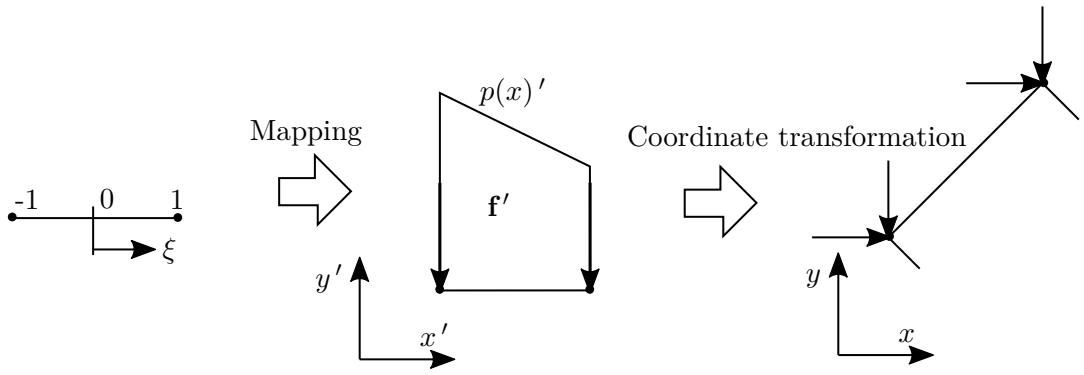


Fig. 2.5: Loading on the element edge

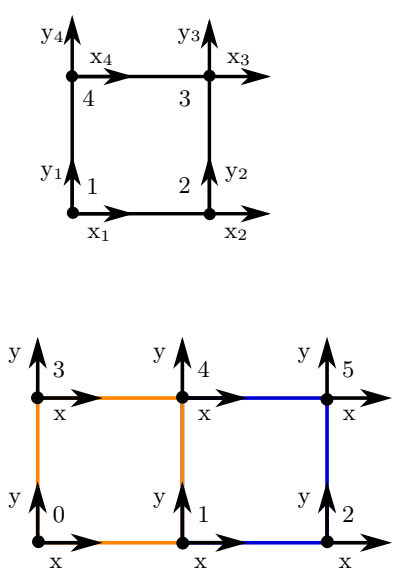
2.4.3 Global matrices and the implementation of Dirichlet boundary conditions

The next step is to distribute the local contributions from $\boldsymbol{k}$, $\boldsymbol{m}$ and $\boldsymbol{f}$ to their corresponding global matrices. After that the BCs can be applied to those global matrices.

The assembly process is explained for the stiffness matrix, $\boldsymbol{k}$, but the other matrices are obtained similarly. Recall the shape of the local stiffness matrix for a linear quadrilateral element

$$\boldsymbol{k} = \begin{pmatrix} k_{1,1} & k_{1,2} & k_{1,3} & k_{1,4} & k_{1,5} & k_{1,6} & k_{1,7} & k_{1,8} \\ k_{2,1} & k_{2,2} & k_{2,3} & k_{2,4} & k_{2,5} & k_{2,6} & k_{2,7} & k_{2,8} \\ k_{3,1} & k_{3,2} & k_{3,3} & k_{3,4} & k_{3,5} & k_{3,6} & k_{3,7} & k_{3,8} \\ k_{4,1} & k_{4,2} & k_{4,3} & k_{4,4} & k_{4,5} & k_{4,6} & k_{4,7} & k_{4,8} \\ k_{5,1} & k_{5,2} & k_{5,3} & k_{5,4} & k_{5,5} & k_{5,6} & k_{5,7} & k_{5,8} \\ k_{6,1} & k_{6,2} & k_{6,3} & k_{6,4} & k_{6,5} & k_{6,6} & k_{6,7} & k_{6,8} \\ k_{7,1} & k_{7,2} & k_{7,3} & k_{7,4} & k_{7,5} & k_{7,6} & k_{7,7} & k_{7,8} \\ k_{8,1} & k_{8,2} & k_{8,3} & k_{8,4} & k_{8,5} & k_{8,6} & k_{8,7} & k_{8,8} \end{pmatrix}$$

This shape is determined by multiplying the number of nodes with the degrees of freedom per node. The size of the global matrix is obtained in the same way. Under consideration of the connectivity matrix (explained in section 3.4) the contributions of each element can be added to the global matrix. This process is best explained with an example. Considering a domain with two quadrilateral elements, each element matrix has the shape illustrated above and degrees of freedom shown in Fig. 2.6. Furthermore the node numbering follows that of 2DGDA (Fig. 3.2b). The resulting global stiffness matrix for that case is illustrated in Fig. 2.6.



		0		1		2		3		4		5	
		x	y	x	y	x	y	x	y	x	y	x	y
0	x	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$			$k_{1,7}$	$k_{1,8}$	$k_{1,5}$	$k_{1,6}$		
	y	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$	$k_{2,4}$			$k_{2,7}$	$k_{2,8}$	$k_{2,5}$	$k_{2,6}$		
1	x	$k_{3,1}$	$k_{3,2}$	$k_{1,1}$ $k_{3,3}$	$k_{1,2}$ $k_{3,4}$	$k_{1,3}$	$k_{1,4}$	$k_{3,7}$	$k_{3,8}$	$k_{1,7}$ $k_{3,5}$	$k_{1,8}$ $k_{3,6}$	$k_{1,5}$	$k_{1,6}$
	y	$k_{4,1}$	$k_{4,2}$	$k_{2,1}$ $k_{4,3}$	$k_{2,2}$ $k_{4,4}$	$k_{2,3}$	$k_{2,4}$	$k_{4,7}$	$k_{4,8}$	$k_{2,7}$ $k_{4,5}$	$k_{2,8}$ $k_{4,6}$	$k_{2,5}$	$k_{2,6}$
2	x			$k_{3,1}$	$k_{3,2}$	$k_{3,3}$	$k_{3,4}$			$k_{3,7}$	$k_{3,8}$	$k_{3,5}$	$k_{3,6}$
	y			$k_{4,1}$	$k_{4,2}$	$k_{4,3}$	$k_{4,4}$			$k_{4,7}$	$k_{4,8}$	$k_{4,5}$	$k_{4,6}$
3	x	$k_{7,1}$	$k_{7,2}$	$k_{7,3}$	$k_{7,4}$			$k_{7,7}$	$k_{7,8}$	$k_{7,5}$	$k_{7,6}$		
	y	$k_{8,1}$	$k_{8,2}$	$k_{8,3}$	$k_{8,4}$			$k_{8,7}$	$k_{8,8}$	$k_{8,5}$	$k_{8,6}$		
4	x	$k_{5,1}$	$k_{5,2}$	$k_{7,1}$ $k_{5,3}$	$k_{7,2}$ $k_{5,4}$	$k_{7,3}$	$k_{7,4}$	$k_{5,7}$	$k_{5,8}$	$k_{7,7}$ $k_{5,5}$	$k_{7,8}$ $k_{5,6}$	$k_{7,5}$	$k_{7,6}$
	y	$k_{6,1}$	$k_{6,2}$	$k_{8,1}$ $k_{6,3}$	$k_{8,2}$ $k_{6,4}$	$k_{8,3}$	$k_{8,4}$	$k_{6,7}$	$k_{6,8}$	$k_{8,7}$ $k_{6,5}$	$k_{8,8}$ $k_{6,6}$	$k_{8,5}$	$k_{8,6}$
5	x			$k_{5,1}$	$k_{5,2}$	$k_{5,3}$	$k_{5,4}$			$k_{5,7}$	$k_{5,8}$	$k_{5,5}$	$k_{5,6}$
	y			$k_{6,1}$	$k_{6,2}$	$k_{6,3}$	$k_{6,4}$			$k_{6,7}$	$k_{6,8}$	$k_{6,5}$	$k_{6,6}$

Fig. 2.6: Example: Assembly of a global stiffness matrix

The only possible Dirichlet BCs in this thesis are $u_x = u_y = 0$. For those fixed nodes the corresponding columns and rows in the linear system of equations (Eq. 2.14) have to be deleted. Assuming that in the example above node 0 and 3 are fixed, the rows and columns with the index 0 and 3 (x and y) have to be deleted. The same rows also have to be erased in the global force and displacement vector.

2.4.4 Stress recovery

Calculating stresses is done after computing displacements as part of the post-processing. For elastic material, stresses are directly related to strains and strains to displacements. The elemental strains can be computed using Eq. 2.18. Furthermore the stresses on element level are given as

$$\boldsymbol{\sigma} = \boldsymbol{CBu} \tag{2.31}$$

The computed normal and shear stresses are non-continuous over the element edges, therefore averaging is required to smooth the function. To be able to carry out the averaging procedure, the stresses are required at the element nodes. There are two options available to get the stresses at the nodes

- Compute $\boldsymbol{\sigma}$ directly at the nodes
- Compute $\boldsymbol{\sigma}$ at the Gauss points and then extrapolate the values to the element nodes

The averaging can be done by adding up all the nodal contributions and dividing it by the number of contributing values. Another possibility would be to make a weighted averaging. In general the non-weighted averaging is sufficiently accurate.

3 Code implementation

The implementation of the theoretical part to the software as well as a detailed explanation of the code is given in this chapter. The chosen programming language is Python 2.7, which is a high-level interpreted language. Python was picked because of its growing importance and wide variety of applications. Furthermore it is very intuitive and therefore easy to understand. An object oriented structure was focused, which facilitates further development.

Because of simplicity no graphical user interface (GUI) was designed. Instead the software input is done via a command window (CMD) in combination with a simple plain text input file.

3.1 CMD user interface

For creating the command window, Python's CMD package was used. A class, defining the form of the CMD window, is initialized by a loop and displayed in Source Code 3.1. The loop keeps the CMD window up until the *quit* command (line 10) is called. Each function, like the *quit* function from line 10, has to start with a *do_* statement. Line 6 creates the core *run* function, thus by calling *run*, all the finite element computations are carried out. Additionally the CMD package provides a *help* function. Simply typing *help* displays all the functions available in the program. Typing *help* and the function name prints a short description, which has to be defined just after creating the operation (e.g. line 7).

Source Code 3.1: CMD input

```
1  from cmd import Cmd
2  from run_FEM import run_FEM
3
4  class MyPrompt(Cmd):
5
6      def do_run(self, args):
7          """runs the core, taking the input from input.txt and writing the results to output.vtu."""
8          run_FEM()
9
10     def do_quit(self, args):
11         """Quits the program."""
12         print "Quitting."
13         raise SystemExit
14
15 if __name__ == '__main__':
16     prompt = MyPrompt()
17     prompt.prompt = '> '
18     prompt.cmdloop('Welcome to EFA!')
```

3.2 Program input and variable handling

The variables are fed to the program using an input file in .txt format. Like in Python, # is used for writing comments. The following code is a segment of the input file, defining the parameters for the mesher.

Source Code 3.2: Input file meshing

```
# Meshing

nelem_vertical_above_kink = 1 # if no kink set to 0
nelem_vertical_below_kink = 4
nelem_horizontal = 2
```

A function which reads the input file and stores all the introduced variables was created. For easy access the variables are saved to an object initialized from a container class. The object can be called in every file, thus all the variables can be accessed easily. The following example code shows how to get the variable *nelem_horizontal* in an arbitrary file.

Source Code 3.3: Variable handling

```
# importing the function that reads the .txt file
import import_from_txt as imp
# importing the container class, called Box
from var_container import Box

# creating an object using Box as a blueprint
variables = Box()
# adding variables from input.txt to the container
variables = imp.import_variables(variables)

# call e.g. the geometry variable like
nelem_horizontal = variables.nelem_horizontal
```

3.3 Finite Element core

The purpose of this function is to put together all the pieces needed for the Finite Element computation. All relevant functions are imported and executed step by step. As explained in section 3.1, this function can be called in the program with the *run* command. Fig. 3.1 illustrates the FEM procedure once the *run* command was called.

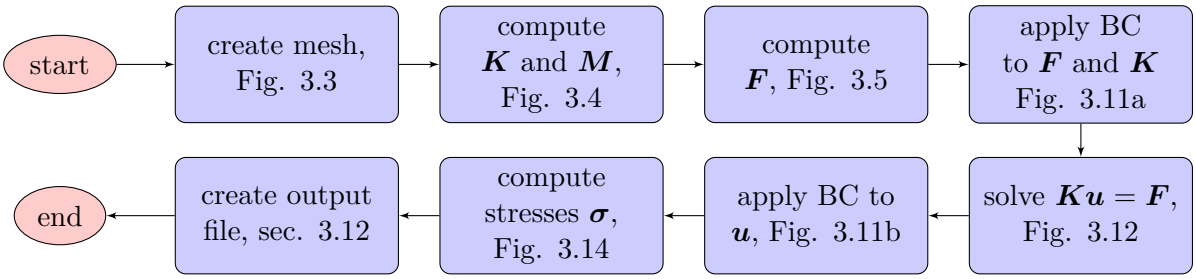


Fig. 3.1: FEM procedure

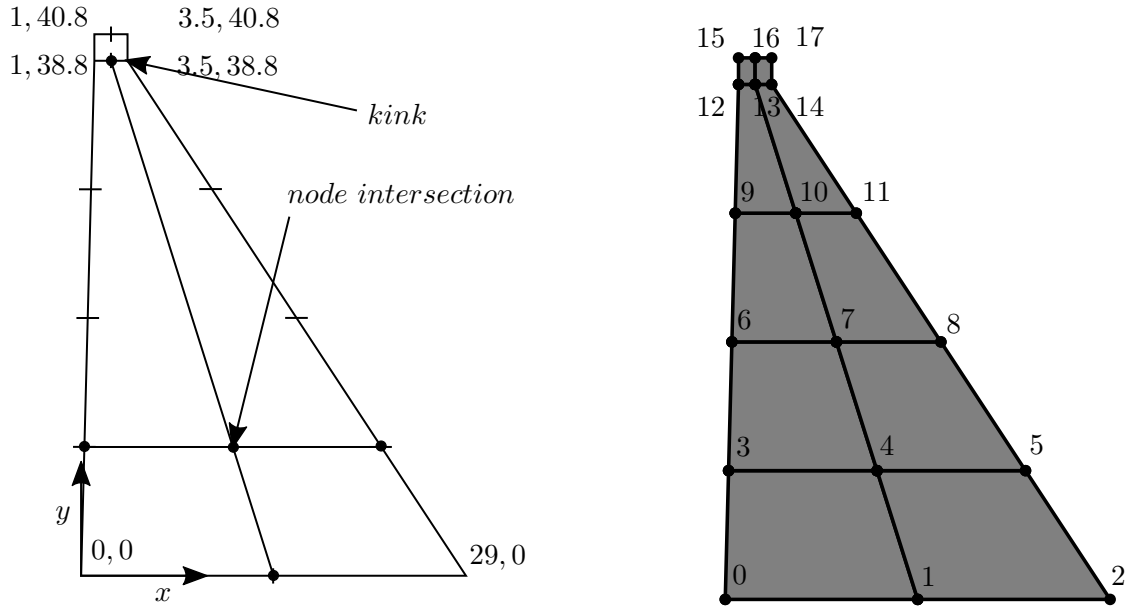
3.4 Meshing procedure

The program's mesher discretizes the domain using linear quadrilateral elements. It is well suited for simple geometries of gravity dams.

In general a 2-D mesh can be created with respect to the following restrictions

- The lowest point of the domain has to be the farthest left
- Up to one kink on the upstream and downstream surface can be implemented, with the condition that the kinks have to be at the exact same height.
- No sloped foundations and crowns are allowed

The input for the mesher are geometry points, which have to be defined counter-clockwise starting at the zero point of the coordinate system. From this input a wire frame is created. In the next step this wire frame is divided equally, depending on the given edge divisions. Connecting the computed nodes on the edges leads to a quadrilateral mesh. Furthermore the intersection points can be determined and stored, along with the outer nodes, in an array. This procedure for an arbitrary case is illustrated in Fig. 3.2a. The resulting mesh, taking the input from Source Code 3.2, is shown in Fig. 3.2b.



(a) Illustration of the meshing procedure **(b)** Example mesh

Fig. 3.2: From geometry to mesh

From the resulting unsorted array, connectivity and nodal matrix can be created. Those two matrices explicitly describe the generated mesh and are used in the Finite Element core. The nodal matrix is a list of nodes and their x-y coordinates. The connectivity matrix basically describes the elements by expressing the node connectivities. For the case illustrated in Fig. 3.2b, nodal and connectivity matrices are shown in Tab. 3.1

Tab. 3.1: Node and connectivity matrix

nodes	
0.0	0.0
14.5	0.0
29.0	0.0
0.25	9.7
11.44	9.7
22.63	9.7
0.5	19.4
8.38	19.4
16.25	19.4
0.75	29.1
5.31	29.1
9.88	29.1
1.0	38.8
2.25	38.8
3.5	38.8
1.0	40.8
2.25	40.8
3.5	40.8

connectivity			
0	1	3	4
1	2	5	4
3	4	7	6
4	5	8	7
6	7	10	9
7	8	11	10
9	10	13	12
10	11	14	13
12	13	16	15
13	14	17	16

The code created for the meshing procedure is illustrated as a flow chart in Fig. 3.3.

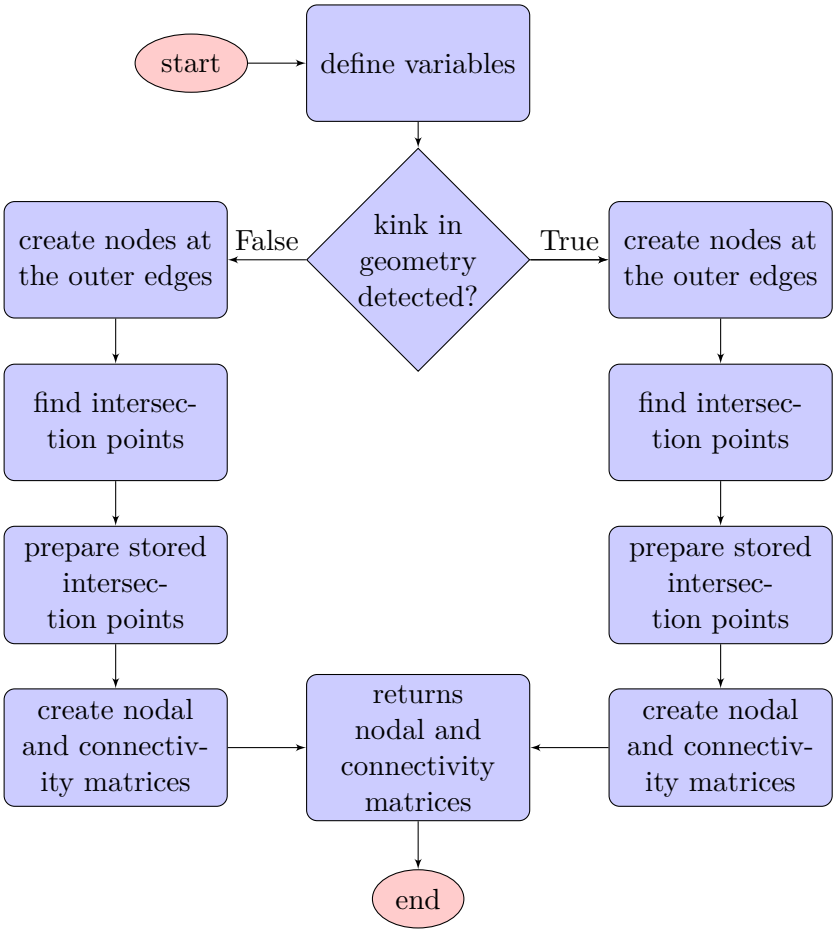


Fig. 3.3: Implementation of the mesher

3.5 Computing global stiffness and mass matrix

The computation of the stiffness and the mass matrix is done in the same function. This comes in handy because several steps have to be carried out just once, which saves computational time.

First all necessary variables from the input file have to be initialized. The required variables for this function are:

- Poisson's ratio, ν
- Elastic modulus, E
- Specific weight of concrete, ρ_c
- Keyword for plane strain or stress
- Keyword for the used mass method: lumped or consistent mass method

The next step is to initialize zero matrices for $\boldsymbol{K}$ and $\boldsymbol{M}$ and define Gauss points for a 2x2 integration rule. Since $\boldsymbol{C}$ is constant, it can be computed on the outermost level, using Eq. 2.8 for plane stress and Eq. 2.13 for plane strain. Now the loop over all elements in the connectivity matrix starts. Here the element stiffness matrix $\boldsymbol{k}$ and the element mass matrix $\boldsymbol{m}$ are initialized. To be able to give values to this initialized matrices, a loop over all the Gauss points (numeric integration) is necessary. For every Gauss point the shape function derivatives in the reference coordinate system are computed, mapped using the Jacobian (Eq. 2.20) and added to the strain-displacement matrix $\boldsymbol{B}$ from Eq. 2.18. Computing the stiffness matrices at each Gauss point and simply adding them up leads to the elemental stiffness matrix $\boldsymbol{k}$. Dependent on the user input, the elemental mass matrix $\boldsymbol{m}$ is either computed using a lumped (DLMM) or consistent (CMM) mass approach.

After computing $\boldsymbol{k}$ and $\boldsymbol{m}$ for the element, its contributions are assembled to previously initialized corresponding global matrices $\boldsymbol{K}$ and $\boldsymbol{M}$. After finishing the elemental loop, $\boldsymbol{K}$ and $\boldsymbol{M}$ are returned.

For better understanding, this function is illustrated in Fig. 3.4.

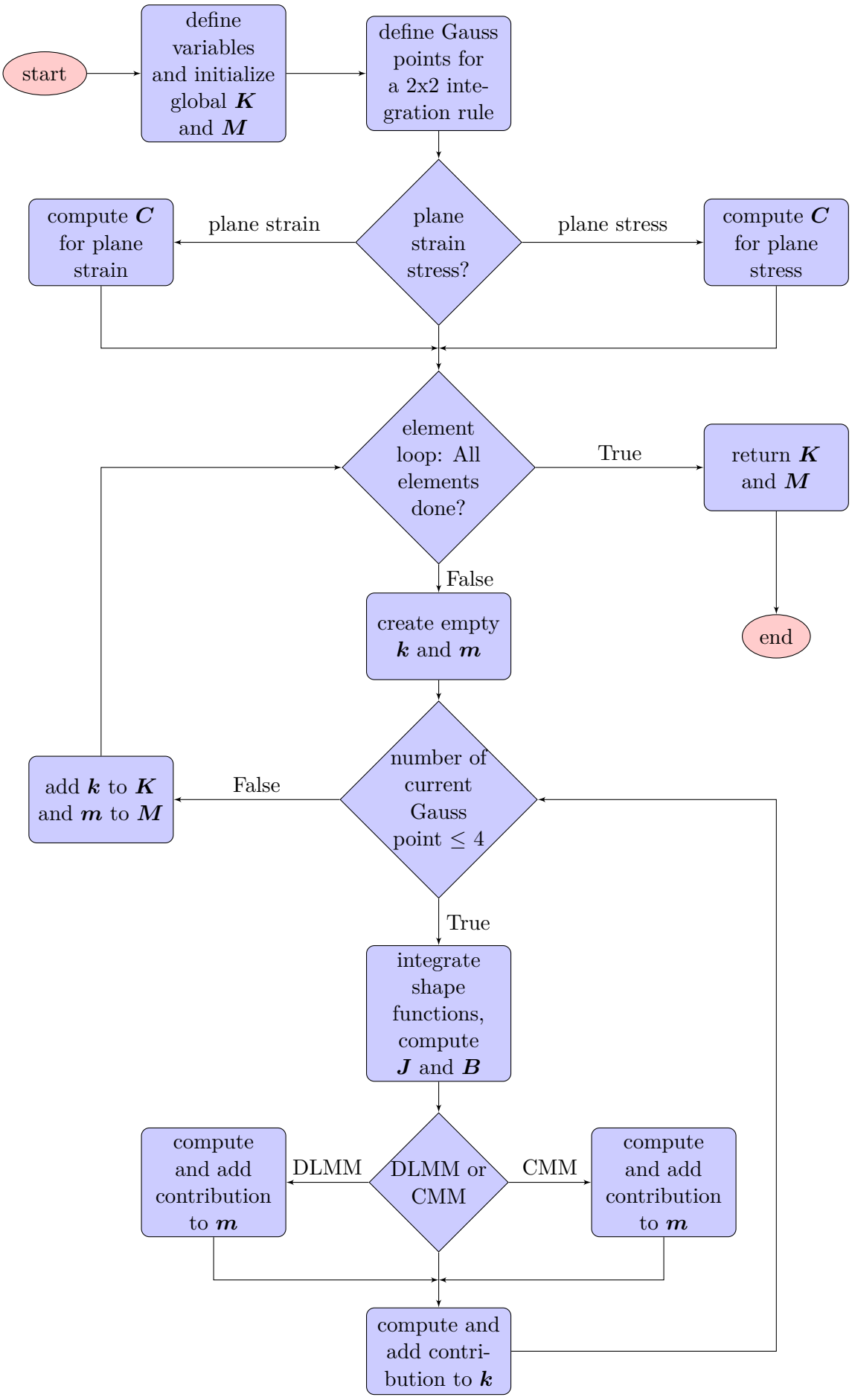


Fig. 3.4: Implementation of the FEM core

3.6 Computing the global force vector

The global force vector, $\boldsymbol{F}$, can be calculated by summarizing all global force vectors from the relevant loading cases, which are:

- Dead load, $\boldsymbol{F}_{dead}$
- Hydrostatic loading, $\boldsymbol{F}_{hydrostatic}$
- Hydrodynamic loading due to Westergaard, $\boldsymbol{F}_{hydrodynamic}$
- Dam acceleration, $\boldsymbol{F}_{dam,a}$

As mentioned in section 2.4.2 those cases can be divided into acceleration induced forces, like dead load or forces due to dam acceleration, and distributed loads, which are hydrostatic and hydrodynamic forces. More specifically, Eq. 2.24 is used to compute acceleration induced forces and Eq. 2.27 for distributed loads.

Since all possible forces can be divided into one of the mentioned categories, implementation into the software gets easier. Two functions with slight exceptions, depending on the loading case, were programmed and can be used for every case.

First the present function imports the needed variables and the list of key options from the input file. Key option 1 stands for dead load, 2 for hydrostatic loading and 3 for earthquake loading, which includes hydrodynamic forces and forces due to dam acceleration. After importing all the required variables, a zero force vector is initialized. Now for each loading case the corresponding function is called and the force, that was previously set to zero, overwritten with the computed values. At the end of the function all values are summed up and returned. The explained code is illustrated in Fig. 3.5.

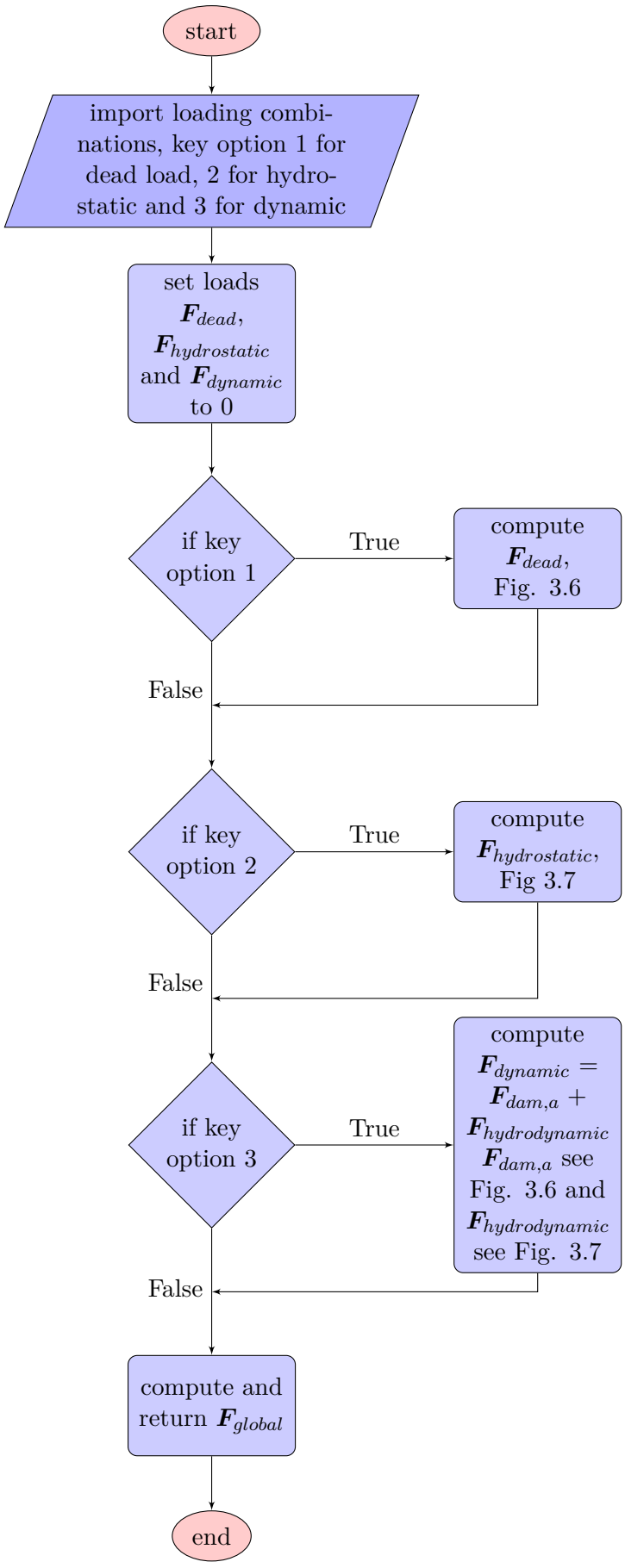


Fig. 3.5: Implementation of global forces

3.7 Acceleration induced forces

The implementation of the acceleration induced forces was simply done by taking the global mass matrix, $\boldsymbol{M}$, and multiplying it with the global acceleration vector $\boldsymbol{g}$ or $\ddot{\boldsymbol{u}}_{x,y}$. This procedure is illustrated in Fig. 3.6.

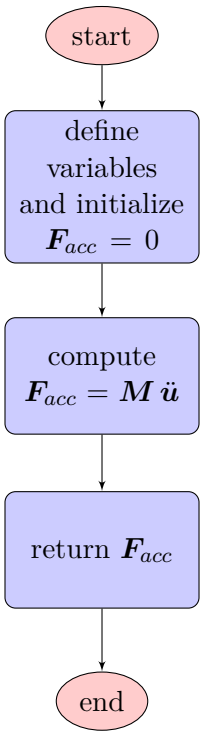


Fig. 3.6: Implementation of acceleration induced forces

3.8 Implementation of distributed loads

In general the computation of the distributed loads follows the principle explained in section 2.4.2 (Eq. 2.30). For a 2-point Gauss rule Eq. 2.30 becomes

$$\boldsymbol{f}' = \sum_{n=0}^{2} \boldsymbol{N}^T p\,(x_n)'\,\boldsymbol{J} \tag{3.1}$$

wherein $\boldsymbol{f}'$ is the elemental force vector in local coordinates and $p\,(x_n)'$ is the chosen loading function in which x_n is the distance from the coordinate zero to the Gauss point. $\boldsymbol{N}^T$ are the transposed shape functions and $\boldsymbol{J}$ is the Jacobian.

First the Gauss points on the 1-D reference element are defined in the function. Besides defining the Gauss points, the local force vector $\boldsymbol{f}'$ is initialized and set to zero. For each Gauss point the shape function derivatives are computed and mapped to the local coordinate system, using the Jacobian. Also the distance from the Gauss point to the zero coordinate of the function is computed. The position of the zero coordinate is discussed in more detail in section 3.8.1 and 3.8.2. Looping over all the Gauss points of an element results in $\boldsymbol{f}'$. The next step is to perform a coordinate transformation to transfer the local force vector to the global coordinate system. This vector, $\boldsymbol{f}$, is then added to the global force vector $\boldsymbol{F}$. The described procedure is illustrated in Fig. 3.7.

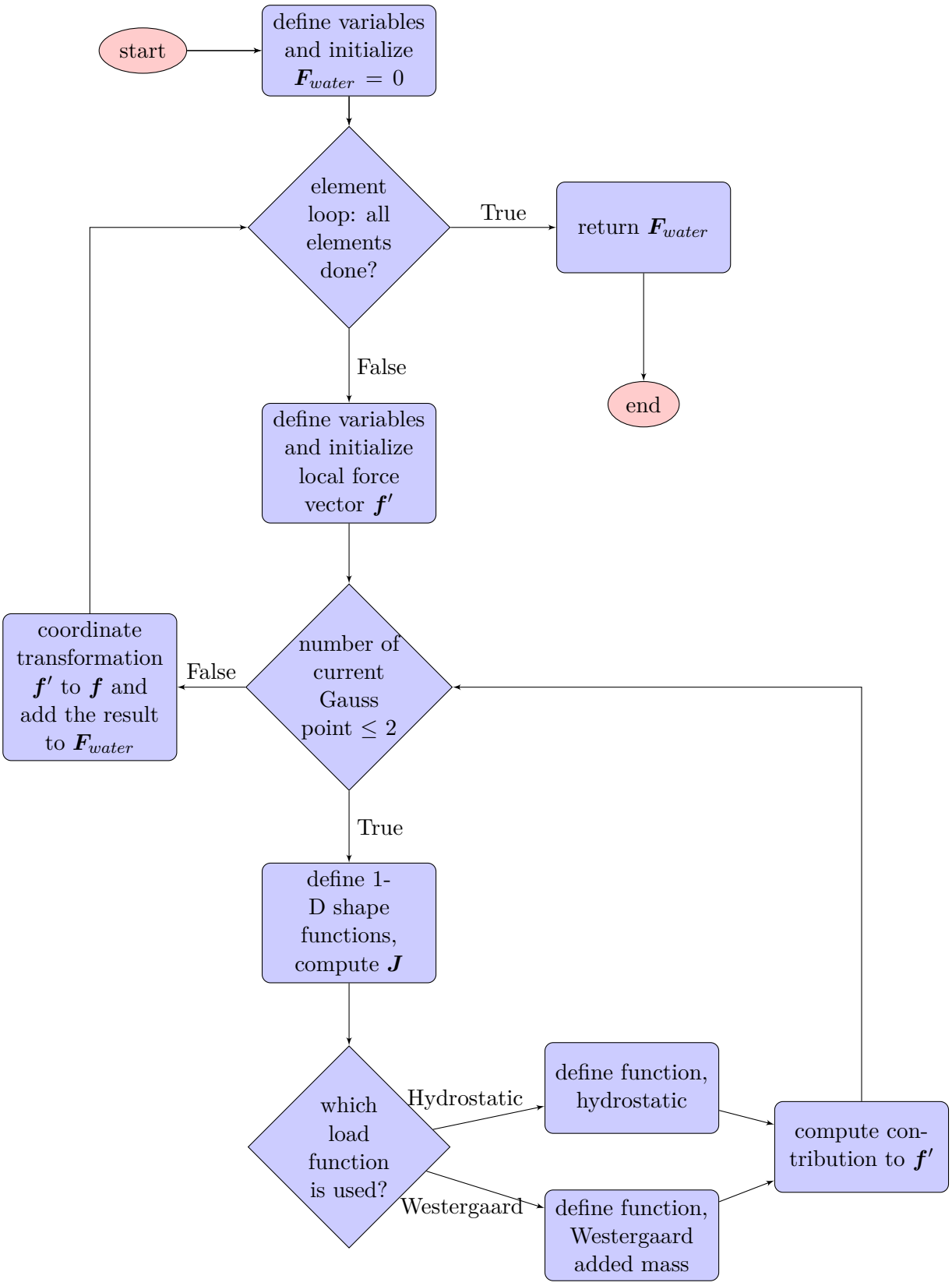


Fig. 3.7: Implementation of distributed loads

3.8.1 Implementation of hydrostatic water pressure

The hydrostatic water pressure is a distributed load that acts perpendicularly to the upstream surface. The pressure increases linearly with the water depth. Fig. 3.8 shows a dam with a submerged kink and the resulting distributed water pressure.

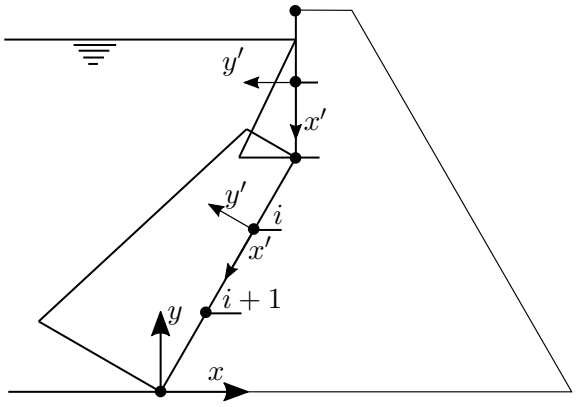


Fig. 3.8: Hydrostatic pressure distribution

For computing the hydrostatic forces the local coordinate system (x',y') is aligned with the element edges. The loading function, $p(x_n)'$, for solving Eq. 3.1, is in fact a trapezoid and yields

$$p(x_n)' = \frac{p_{i+1} - p_i}{L}\, x' + p_i \tag{3.2}$$

in which L is the length of the 1-D element, x' is the distance to the Gauss points (for numeric integration) and p is the pressure at the element node. Here i donates the first node in the local coordinate system and $i+1$ the second one. The pressure is dependent on the vertical distance of the node to the water surface and is given as

$$p_i = y_i\, g\, \rho_w \tag{3.3}$$

where h_i is the vertical distance from the node to the water surface, g is the gravity acceleration and ρ_w the specific weight of water.

3.8.2 Implementation of the Westergaard added mass

During the event of an earthquake, additional hydrodynamic pressure acts on the upstream surface of a gravity dam. Westergaard (1933) derived a pressure function under the following simplifications, stated by Kuo (1982):

- The dam is simplified as a two dimensional rigid body. Additionally a vertical upstream surface is assumed
- The expansion of the reservoir in upstream direction is infinite
- Displacements of the fluid particles are negligible
- Surface waves are neglected

- Only horizontal acceleration is considered

Westergaard (1933) stated that the function of the additional pressure has a parabolic shape. Furthermore he found that under horizontal ground acceleration a certain amount of water moves back and forth with the dam. This body of water is called the 'added mass'. Kuo (1982) later made additional assumptions, so that Westergaard's approach is applicable to any geometry.

The simplified added mass per unit area stated in Saouma (2006) yields

$$\gamma_{am} = \frac{7}{8}\rho_w\sqrt{h\,y} \tag{3.4}$$

and is illustrated in Fig. 3.9.

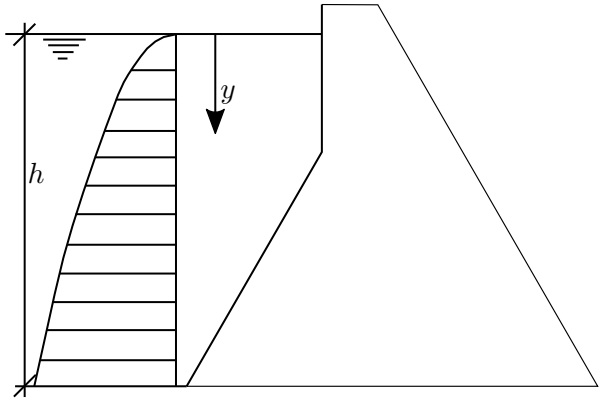


Fig. 3.9: Hydrodynamic pressure distribution

To distribute this mass per unit area to nodal masses, numeric integration in a local coordinate system is used. In fact the same formula as for the hydrostatic pressure (Eq. 3.1) can be applied. The only difference is that the outcome is not forces but lumped masses. Therefore the lumped masses at the element nodes, in the local coordinate system (x', y'), can be computed using

$$\boldsymbol{m}' = \sum_{n=0}^{2} \boldsymbol{N}^T p\,(x_n)'\,\boldsymbol{J} \tag{3.5}$$

in which $p(x_n)'$ is

$$p(x_n)' = \frac{7}{8}\rho_w\sqrt{h\,x'} \tag{3.6}$$

and where x' is the distance from the water level to the elemental Gauss points. After computing the lumped masses, coordinate transformation can be applied to obtain the directional masses. By multiplying those masses with the corresponding earthquake accelerations the global force vector can be computed. Fig. 3.10 illustrates this procedure as well as the required coordinate systems.

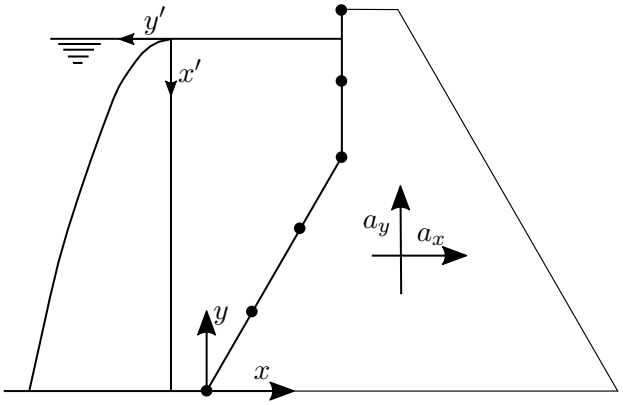


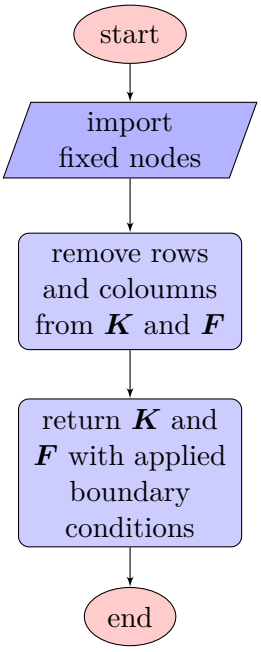
Fig. 3.10: Local and global coordinates for added mass computation

3.9 Support boundary conditions

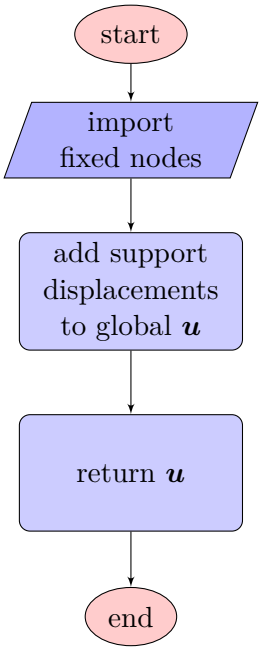
Since the foundation is assumed to be rigid, the displacements of the contact nodes have to be set to zero. More specifically, for the bottom nodes of the dam $u_x = u_y = 0$ is valid.

These BCs are applied by deleting the corresponding rows and columns of the fixed nodes, explained in section 2.4.3. Regarding the mass matrix, $\boldsymbol{M}$, the boundary conditions are indirectly applied at the global force vector. Fig. 3.11a illustrates the procedure.

After solving the linear system of equations, explained in section 3.10, the displacements of the previously deleted entries have to be added again. Those displacements are zero but necessary for computing stresses. The routine is similar to those explained before and is illustrated in 3.11b.



(a) Boundary conditions for $\boldsymbol{K}$ and $\boldsymbol{F}$



(b) Boundary conditions for $\boldsymbol{u}$

Fig. 3.11: Implementation of boundary conditions

3.10 Solving

After applying the boundary conditions, Eq. 2.14 can be solved to obtain nodal displacements. The linear system of equations is solved, using *linalg.solve*, which is part of Python's numpy package. This function uses the Linear Algebra Package (LAPACK), originally coded in Fortran. LAPACK's *gesv* routine is used, which is based on Gaussian elimination. More specifically, lower upper (LU) decomposition is used. The investigated solver is a direct one, which is commonly used for applications similar to this one. Fig. 3.12 illustrates the function.

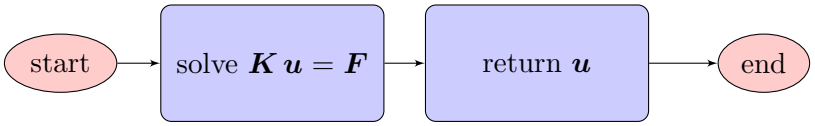


Fig. 3.12: Implementation of the solving procedure

3.11 Stress recovery

After solving for displacements, stresses are computed as part of the post-processing, using Eq. 2.31. To be able to compute $\boldsymbol{\sigma}$, the strain-displacement matrix, $\boldsymbol{B}$, is needed. Calculating $\boldsymbol{B}$ follows the procedure already explained in section 3.5. Based on that, strains and stresses can be computed at the Gauss points (Eq. 2.31). Those stresses are then extrapolated to the element nodes, using the classical shape functions.

Recalling that the stresses across the element boundaries are non-continuous, nodal averaging is required. In this thesis non-weighted averaging is used, which means that the stresses at each node, i, can be computed as

$$\boldsymbol{\sigma}_i = \frac{\sum_{j=1}^{n_{GP}} \boldsymbol{\sigma}_j}{n_{GP}} \tag{3.7}$$

where n_{GP} is the number of contributing Gauss points and $\boldsymbol{\sigma}_j$ is the stress vector, extrapolated from a single Gauss point. Fig. 3.13 illustrates the averaging of an arbitrary node influenced by four Gauss points. A flow chart of the function is shown in Fig. 3.14.

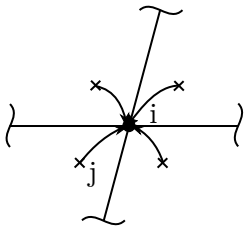


Fig. 3.13: Nodal averaging

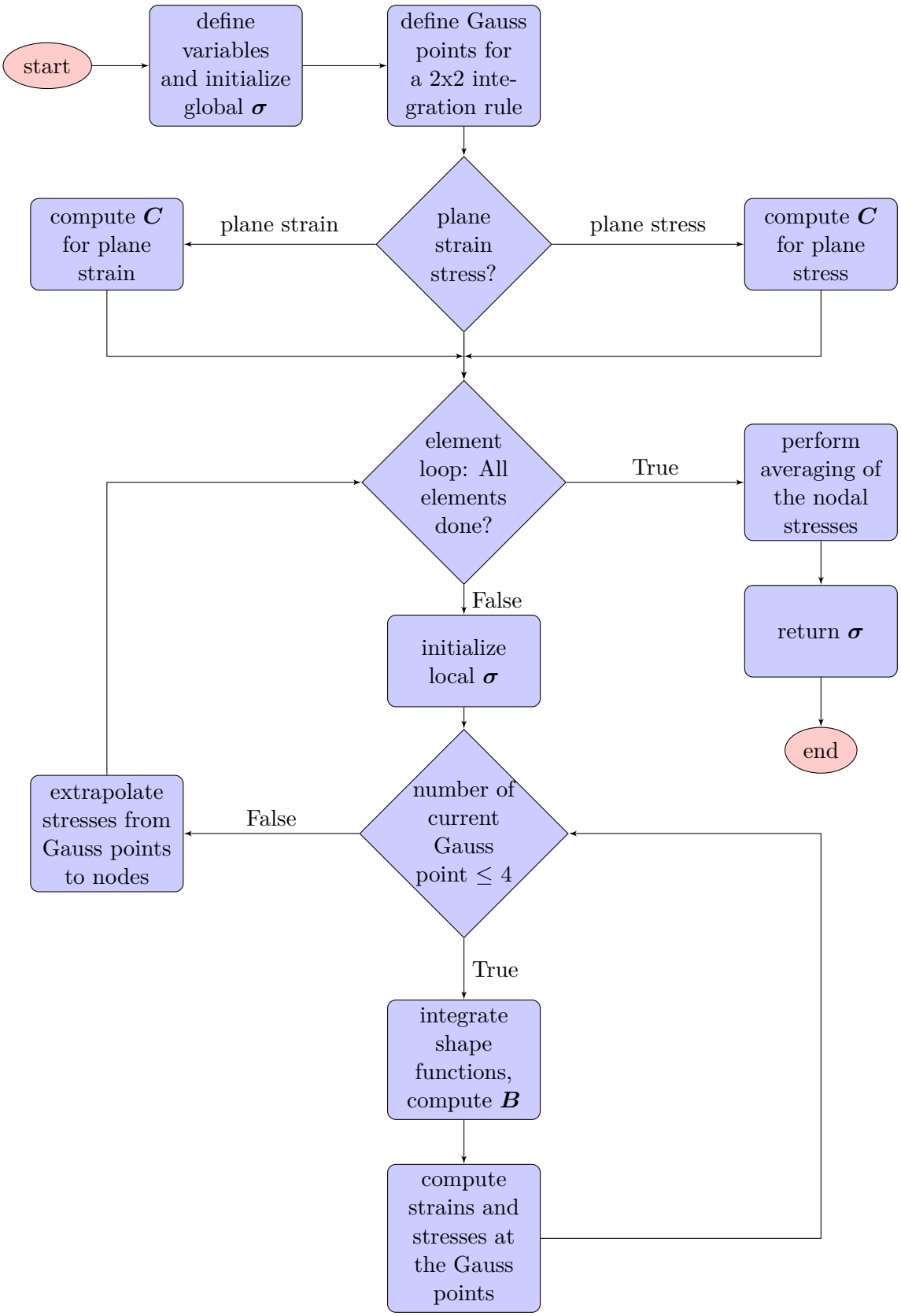


Fig. 3.14: Implementation of stress recovery

3.12 Creating the output file

There are several options to import the generated data to ParaView. The most suited option for the needs of this thesis is the Visualization Toolkit (VTK). VTK is a very large and powerful tool for visualization and graphic processing. Furthermore it provides the possibility to use XML syntax, which enables a wide variety of features.

VTK xml file types have certain extensions, depending on the type of the data. For serial unstructured grids, which are commonly used for finite element meshes, the vtu file format is recommended. Source Code 3.4 shows the general structure of an unstructured grid dataset.

Source Code 3.4: Output file

```
1  <?xml version="1.0"?>
2  <VTKFile type="UnstructuredGrid" version="0.1" byte_order="LittleEndian">
3    <UnstructuredGrid>
4      <Piece NumberOfPoints= "n_points" NumberOfCells="n_cells">
5        <Points>
6          <DataArray type="Float32" NumberOfComponents="3" format="ascii"/>
7          x   y   z
8        </Points>
9        <Cells>
10         <DataArray type="Int32" Name="connectivity" format="ascii"/>
11         i   j   k   l
12         <DataArray type="Int32" Name="offsets" format="ascii"/>
13         4   8   12   16 ...
14         <DataArray type="Int32" Name="types" format="ascii"/>
15         9   9   9   9 ...
16       </Cells>
17       <PointData Vectors="displacements">
18         <DataArray type="Float32" NumberOfComponents="3" Name="displacements" format="ascii"/>
19         ux   uy   uz ...
20         <DataArray type="Float32" Name="sigma_x" format="ascii"/>
21         sigma_x ...
22         <DataArray type="Float32" Name="sigma_y" format="ascii"/>
23         sigma_y ...
24         <DataArray type="Float32" Name="tau_xy" format="ascii"/>
25         tau_xy ...
26       </PointData>
27     </Piece>
28   </UnstructuredGrid>
29 </VTKFile>
```

Besides general settings, like the XML version or the file type, the dataset contains information about nodes (<Points>), elements (<Cells>) and computed data (<PointData>). The nodal information consists of coordinates, whereas the elements are specified by their connectivity, offset and element type. Summing up the nodes of each element yields the offset. Furthermore the element type for unstructured grids has to be set to 9 for each element (VTK convention). In the data section, reaching from line 17 to 26, all the computed datasets have to be embedded. A simple function was created to write the output, as just described.

4 2DGDA user manual

To be able to carry out simulations with the created software a user manual is required. No external manual is available, but this chapter fulfils this function. Since the post-processing is a large part of every FEM simulation, the most important functions of the post-processing tool (ParaView) are explained as well. For more advanced applications ParaView's user manual is recommended, Ayachit (2015).

First setting up EFA as well as ParaView is explained for a windows based operating system. As a second step the program limitations are introduced, which have to be adhered to to achieve satisfying results. Furthermore the input file is discussed in detail. Usually wrong or incomplete input files are the main source of errors. At the end of this chapter the author gives some recommendations for the application of the software.

4.1 Getting started

4.1.1 Installation of necessary applications

2DGDA comes with a stand-alone Windows Executable, a licence file and pre-built test input files. The package is available at the homepage of the Institute of Hydraulic Engineering and Water Resources Management (TUGraz). The current version is usable on windows 32 and 64 bit systems. The newest version of ParaView can be acquired from `http://www.paraview.org/`. Instructions for installing ParaView should be taken from the mentioned URL.

4.1.2 Program limitations

The created software comes with a few limitations. Those are defined by the mesher (section 3.4) and summarized below

- Up to one kink on the upstream and downstream surface can be implemented, with the condition that the kinks have to be at the exact same height.

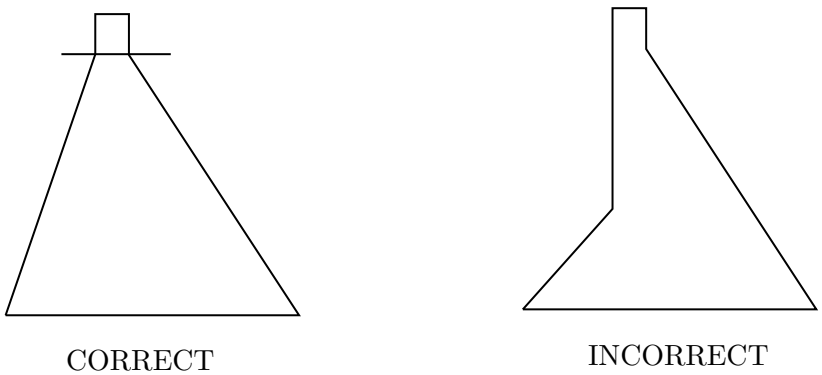


Fig. 4.1: Limitation: kink

- No sloped foundations and crowns are allowed

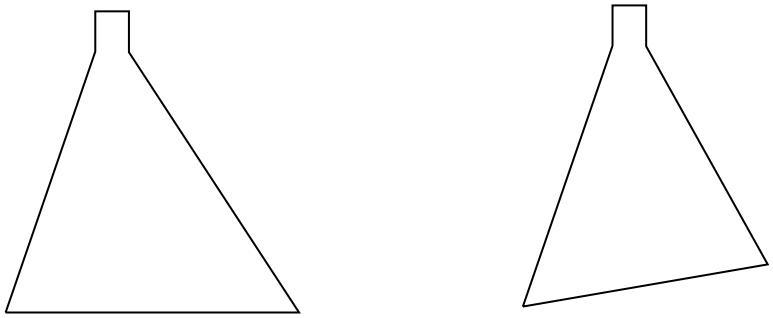


Fig. 4.2: Limitation: Crown and foundation inclination

- The lowest point in the domain has to be the furthest left. Usually this limitation does not occur for normal dam geometries, but it is mentioned for the sake of completeness.

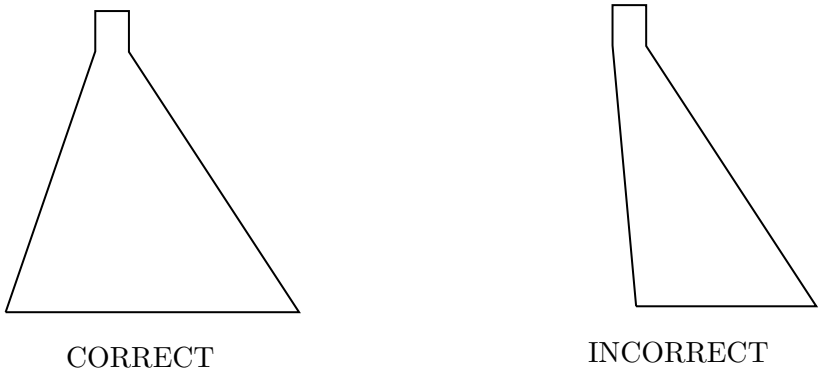


Fig. 4.3: Limitation: lowest point

4.2 The input file

The input file is a plain text document. 2DGDA reads the variables from this file and feeds them into the functions. It is very important to give the variables the exact same name as described below, otherwise an error will arise. Empty lines and lines starting with a hash are skipped by the software.

It is very important to mention that all inputs are in SI-units and derivatives of those. Furthermore all the values are given in respect to the coordinate zero, which is located at the bottom left of the dam. Herein the x axis is aligned with the dam foundation and is positive to the right. The y axis is perpendicular thereto and is defined positively in direction of the dams crest.

4.2.1 Geometry and meshing

Any dam that fulfils the requirements stated above can be defined by the x-y-coordinates of its outer nodes, which are specified inside an array in counter-clockwise direction. After defining the outer shape of the domain, the meshing parameters (edge divisions) for horizontal and vertical (below and above kink) edges have to be set. Source Code 4.1 shows the initialization of geometry and meshing parameters. For geometries with no kink, line 4 takes the value zero.

Source Code 4.1: Input mesher

```
# --------------------  Geometry and Meshing ----------------------------

Geometry = [[0,0],[29,0],[3.5,38.8],[3.5,40.8],[1,40.8],[1,38.8]]
nelem_vertical_above_kink = 4 # if no kink set to 0
nelem_vertical_below_kink = 8
nelem_horizontal = 4
```

4.2.2 Material properties

In this part of the input file material properties and gravitational acceleration are defined. The needed values are

- Poisson's ratio, ν
- Elastic modulus, E
- Specific weight of concrete, ρ_c
- Specific weight of water, ρ_w
- Earth acceleration, g

Source Code 4.2 shows this specific part of the input file for an arbitrary case.

Source Code 4.2: Input material properties

```
# --------------------  Material properties ----------------------------

NU = 0.2 # Poisson's ratio
EE = 25000000000 # elastic modulus
rho_concrete = 2400 # rho concrete
rho_water = 1000 # rho water
gravity = 9.81 # gravity
```

4.2.3 Loading

The sample input, illustrated in Source Code 4.3, is used to describe the loading part of the input file. At the beginning all desired loading combinations have to be specified inside an array (line 7). This can be done by using keywords. More specifically 1 for dead load, 2 for hydrostatic pressure and 3 for forces introduced by earthquake acceleration. Furthermore specific loading variables have to be defined. Line 10 defines the water level relative to the zero coordinate. The horizontal and vertical acceleration, needed for the added mass technique, are given in line 10 and 11.

Source Code 4.3: Input loading

```
# -------------------------  loading ------------------------------------
# ----------------------  combinations ---------------------------------
# dead load = 1, hydrostatic = 2, earthquake = 3
loading = [1,2,3]

# -------------------  waterload settings ------------------------------
water_level = 38.8

# -------------------  earthquake settings ------------------------------
ax = 5 # horizontal acceleration
ay = 0 # vertical acceleration
```

4.2.4 Support boundary conditions

Since the foundation is assumed to be rigid, the nodal displacements at the contact area are zero. Therefore the boundary conditions for the foundation nodes are $u_x = u_y = 0$. Two options are available to restrict the displacements of the desired nodes. The user can either choose the nodes manually by setting the keyword for the BC method to 1 and define all nodes in the BC array, or use keyword 2, which fixes all bottom nodes of the domain. When choosing the nodes manually, the user has to consider the pattern of the node numbering illustrated in Fig. 4.4.

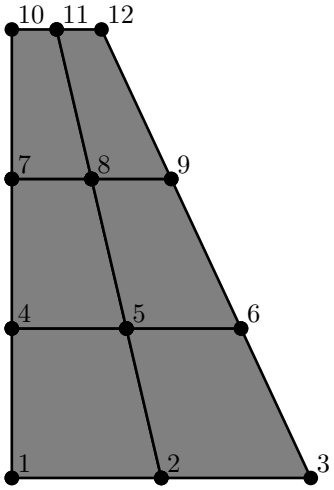


Fig. 4.4: Node numbering 2DGDA

For the case shown above, the input is displayed in Source Code 4.4. Choosing key option 2 for *BC_method* (line 4) and removing line 5 would lead to the same result.

Source Code 4.4: Input boundary conditions

```
1  # ----------------------  Boundary conditions ---------------------------
2  # BC method: give specific nodes = 1, fix all bottom nodes = 2
3
4  BC_method = 1
5  BC = [1,2,3] # fixed nodes  (just for method 1)
```

4.2.5 FEM settings

In this part of the input file the user can define FEM specific settings, such as plane strain/stress simplifications or the technique used for computing the mass matrix. Source Code 4.5 shows the possible settings. The key option set in line 3 specifies the used mass approach, 1 for DLMM or 2 for CMM. Furthermore line 4 defines the 2-D simplification approach, 1 for plane strain and 0 for plane stress.

Source Code 4.5: Input FEM settings

```
1  # ------------------------  FEM settings -------------------------------
2
3  mass_method = 2 # diagonally lumped mass matrix (DLMM = 1), consistant mass matrix (CMM) = 2
4  PS = 1          # plane strain = 1, plane stress = 0
```

4.3 Using 2DGDA

After defining all the parameters in the input file, the program can be started by running the windows executable. Starting up may take several seconds. The program can be used when 'Welcome to 2DGDA!' is displayed. Fig. 4.5 shows the program's interface.

Fig. 4.5: 2DGDA interface

Since most of the input can be controlled in the input file, the program itself needs very few commands. The available functions are listed below:

help: Executing *help* shows a list of all available commands of the software. By calling *help* and the desired function a short description about the command can be displayed.

quit: By calling *quit* the program shuts down. The program can be quit at any time since all the data is stored in the input file, thus no data can get lost.

run: By using the *run* function, the finite element procedure is carried out and an output file in vtu format is created.

Every time modifications in the input file have been made the run function has to be called again to update the output file. If an error occurs, the input file has to be checked for completeness and for spelling errors of the variables.

4.4 Using ParaView

After starting up ParaView, the vtu file can be imported under 'File > Open' (Fig. 4.6).

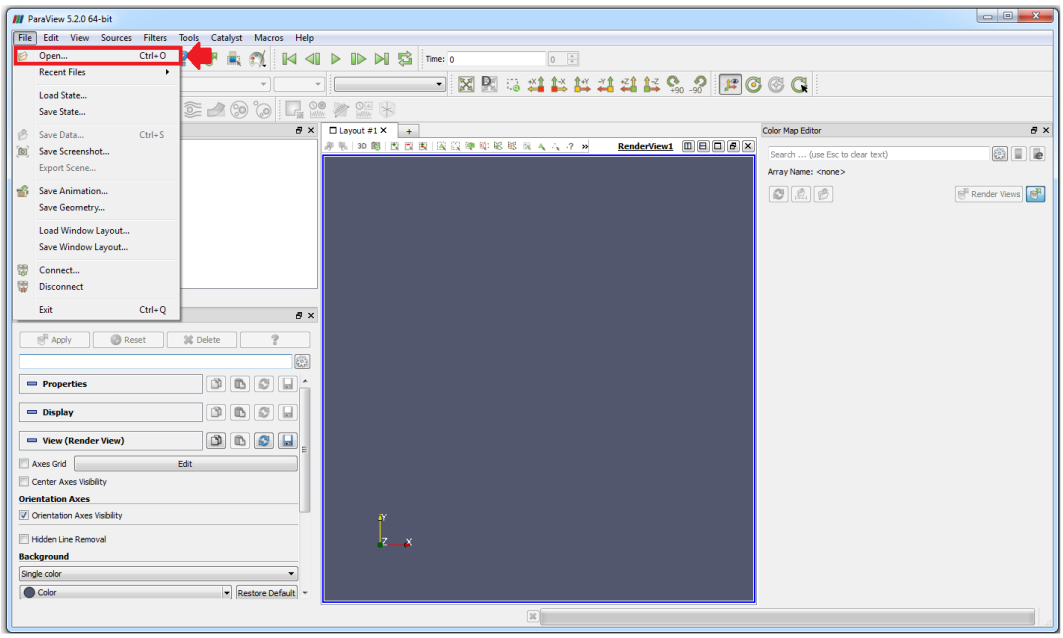


Fig. 4.6: ParaView start interface

The next step is to confirmed the data by pressing *Apply*. Now the geometry of the dam is already displayed. By splitting the screen horizontally and selecting *SpreadSheetView* all the data, like node numbering, nodal coordinates, displacements, normal stresses and shear stresses are displayed in table form. By clicking a single or multiple points in the spread sheet the node also gets highlighted in the Layout and vice versa. Fig. 4.7 shows the imported data.

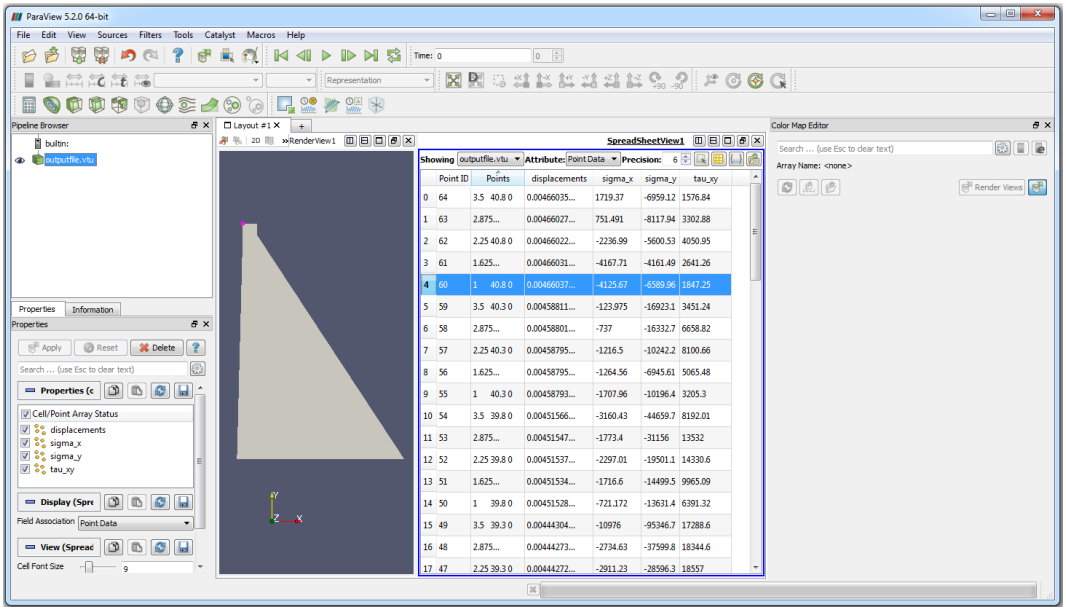


Fig. 4.7: ParaView data

After importing the generated data, the case can be post-processed. Therefore the author explains the most important functions of ParaView. As mentioned above, user requirements exceeding the described functions can be found in ParaView's manual. The most frequently needed functions are illustrated in Fig. 4.8 and described below.

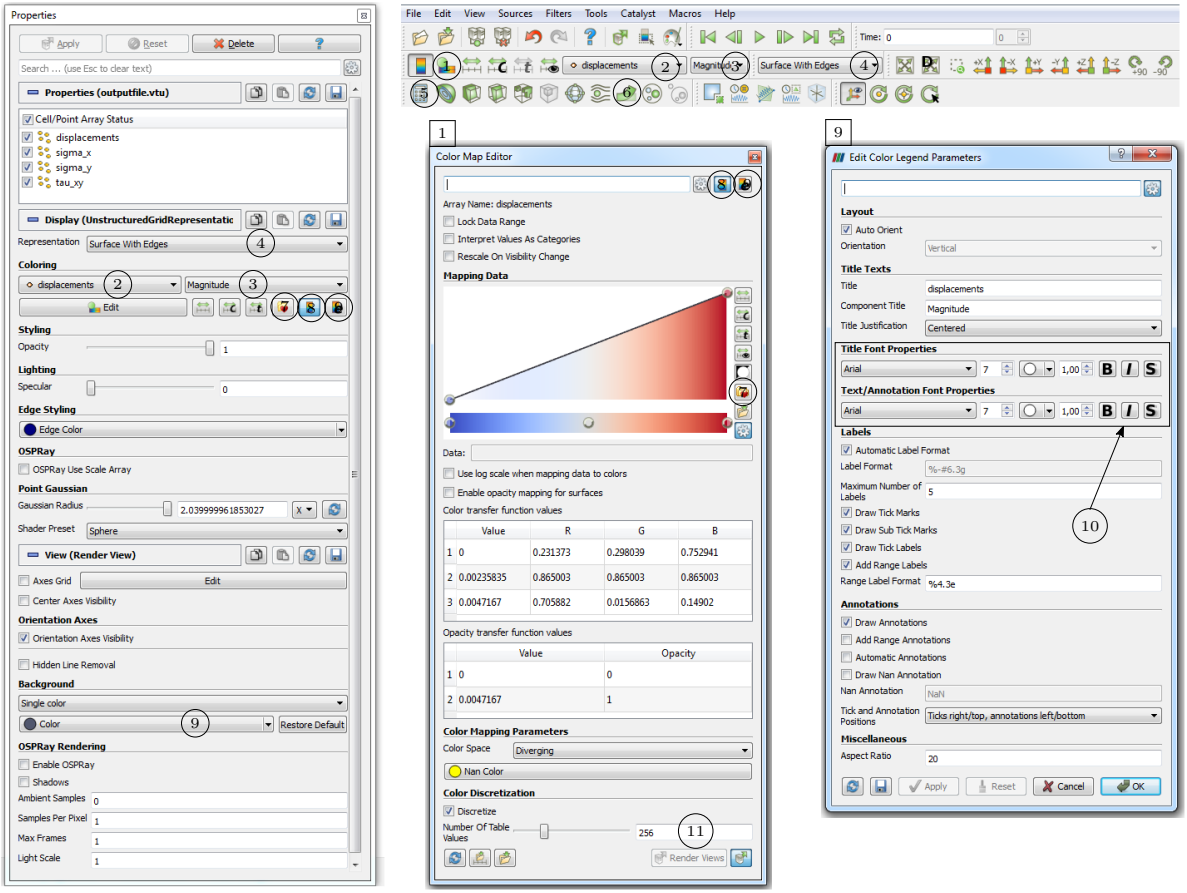


Fig. 4.8: Most important ParaView functions

(1) *Edit Color Map*: Opens the Color Map Editor for advanced display settings.

(2) *Data drop down menu*: Displays the chosen value. The user can choose between displacements, normal stresses $\boldsymbol{\sigma}_x$, $\boldsymbol{\sigma}_y$ and shear stresses $\boldsymbol{\tau}_{xy}$.

(3) *Directional drop down (for vectors)*: Displacements are implemented as a vector with three components at the nodes. That means the user can select directional displacements in x,y or z direction, wherein the z values are zero for 2-D simplifications.

(4) *Contour plot display drop down*: Allows settings for displaying the contour of the dam. Besides others, possible options are the display of nodes, mesh, or the contour including the wire frame of the mesh.

(5) *Calculator*: The calculator tool allows the user to compute additional values from the imported dataset. A good example would be the stresses in z direction for plane strain conditions.

(6) *Warp by Vector*: This function is used to show the displaced dam body. The user can manually set the magnification factor.

(7) *Choose pesent*: This option lets the user choose the colour mapping. A good option is the rainbow pesent.

(8) *Show/hide color legend*: Toggles the legend.

(9) *Edit color legend properties*: Opens the 'Edit Color Legend Parameters' window, which allows advanced settings like modifying the legend's title.

(10) *Title Font properties*: In this window settings regarding the legend, especially its font size, can be set.

(11) *Number of table values*: Lets the user set the number of displayed table values.

After setting all parameters the contour plot can be exported in .png format under 'File > Save Screenshot'.

4.5 Recommendations

The purpose of this section is to give general user recommendations.

Since dams are structures with small face area but large expansion in one direction, a plane strain approach is recommended. Furthermore the author recommends to use the consistent mass method (CMM). This approach usually leads to more accurate results compared to the lumped technique. Even though the CMM is less favourable regarding computation time, it makes no significant difference for static analyses.

Recalling that 2DGDA is based on a basic Finite Element formulation, enhances the possibility of numerical errors. One of those is the locking effect, which occurs in bending dominated domains. Such numeric errors may lead to convergence issues. That means 2DGDA needs a higher number of elements to reach convergence than for example ANSYS®, which allows the use of advanced FEM techniques. Therefore the user is encouraged to perform a mesh sensitivity analysis when using 2DGDA.

In general the user's responsibility is to make sure that all the necessary simplifications can be applied to his/her case. If not, more sophisticated computation methods have to be considered.

5 Validation

For the validation two famous gravity dam cases are compared. The first one is an Indian dam, namely the Koyna Dam. The second one is the Pine Flat Dam, located in the United States. Geometries as wells as material properties are taken from the referenced papers. Regarding ground acceleration due to earthquake, reasonable assumptions are made.

The software used to validate the results is ANSYS® Academic Research, Release 17.2. More specific the comparison is carried out between 2DGDA and two different 2-D element types, available in the ANSYS® Workbench The elements are plane 13 and plane 183, which are explained in section 5.1. Compared are the directional displacements on the upstream surface of the dam (in x-direction), the stresses in the dam body, and the convergence at a certain node. Since the added mass is not that straightforward to implement in ANSYS®, this load case is only considered for the directional displacements computed with the plane 13 element. For the remaining investigations only dead load and hydrostatic pressure are considered.

5.1 Setting up ANSYS®

This section explains how to set up the test cases in ANSYS®. After starting ANSYS® Workbench the *Static Structural* analysis type is initialized by simply dragging it from the toolbox and dropping it to the *Project Schematic* window. The *Analysis Type* in the Geometry part of the *Static Structural* window has to be set to 2-D. The necessary steps that have to be carried out at the Workbench are shown in Fig. 5.1.

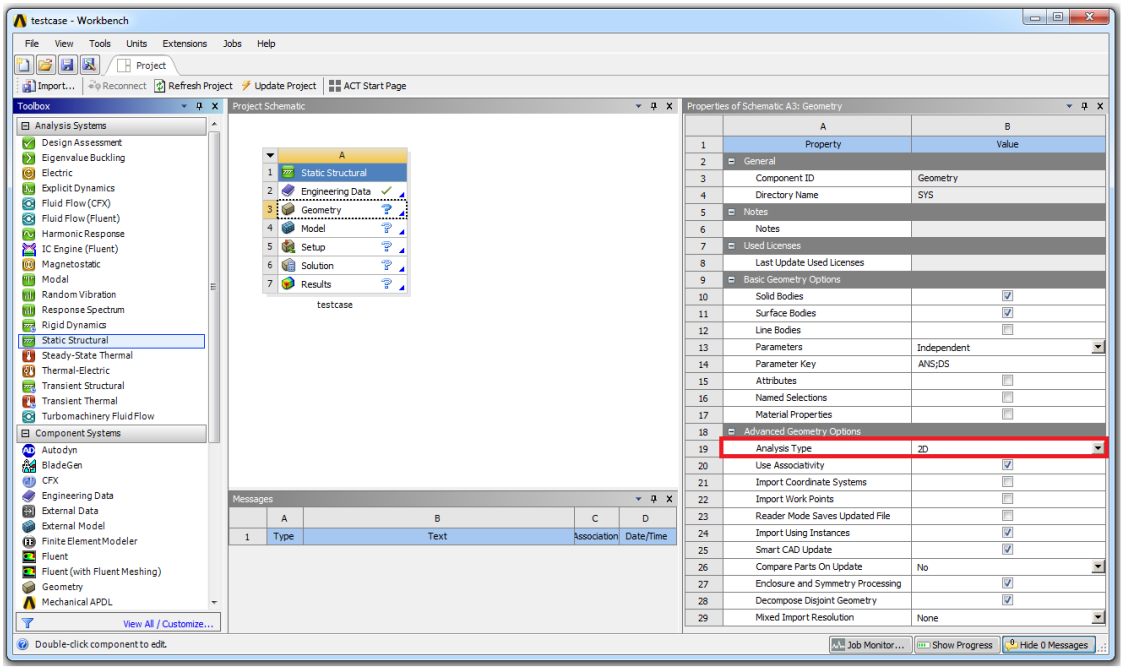


Fig. 5.1: ANSYS® interface

By double-clicking on *Geometry* ANSYS® DesignModeler can be started. This tool allows the user to create the geometry. There are multiple ways to do so, whereby in this work a point file is used. The content of this file is shown in Fig. 5.2. After importing these points to the DesignModeler, lines and surfaces can be created from those. For further work suppressing the line body is recommended. The necessary steps are illustrated in Fig. 5.2.

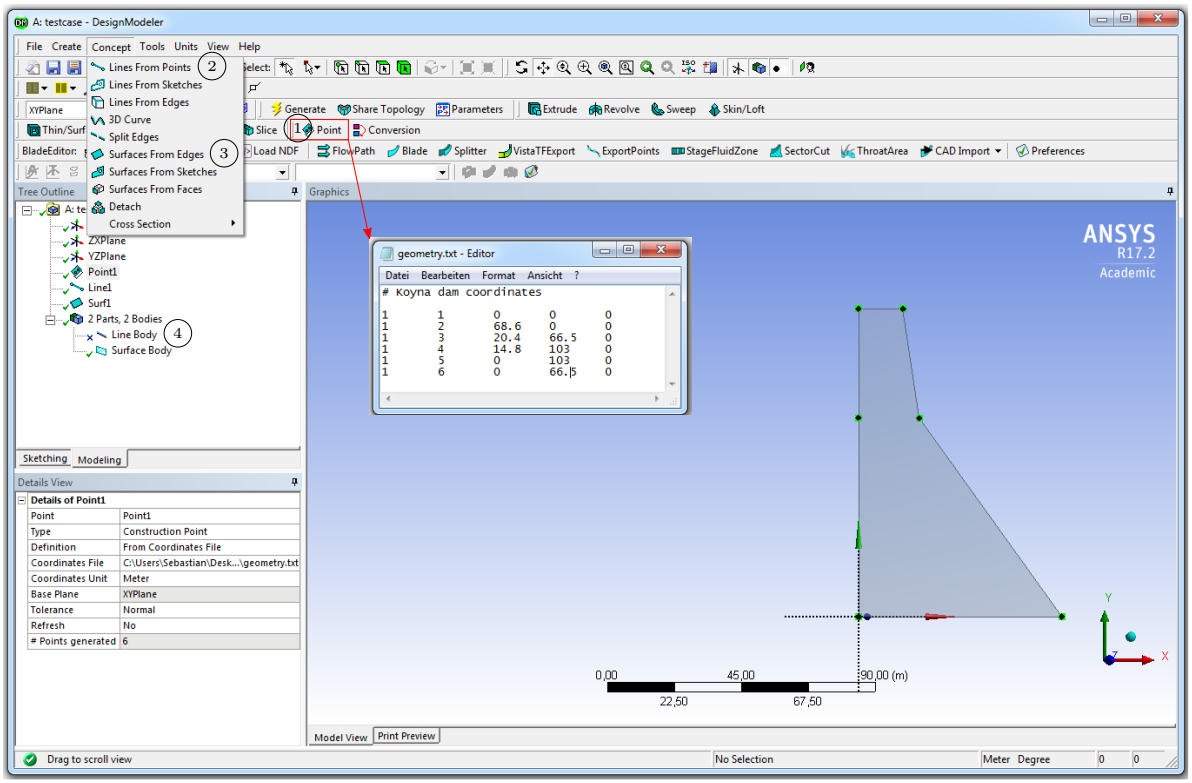


Fig. 5.2: Creating the geometry

After closing the DesignModeler, the *Model* can be opened in the Workbench interface (Fig. 5.1). The major components that are modified are *Geometry*, *Mesh*, *Named Selection* and *Static Structural*, which can be found in the *Project Outline* tree on the left side.

Geometry

First the element type and the material properties have to be set. Two element types are used to validate the results in this thesis.

- Plane 13 (ANSYS®, ANSYS Mechanical APDL Element Reference): Plane 13 is a 2-D solid element, suited for structural analyses. This is a very basic element without any advanced technology. Turning off the extra shape functions for this element allows the computation based on pure FEM. Since 2DGDA is also based on basic finite element techniques, this element type is well suited for validating the developed software.
- Plane 183 (ANSYS®, ANSYS Mechanical APDL Element Reference): Plane 183 is a 2-D, higher order element, which supports structural analyses. This element uses state of the art FEM techniques, thus would most likely be used for the present test cases.

To specify the element type and define the material properties, a command file in the *Surface Body* branch has to be created. The content of this file, for the case of the Koyna Dam, is illustrated in Source Code 5.1. Line 1 shows the initialization of the plane 13 element type with u_x and u_y degrees of freedom, plane strain conditions and no additional shape functions. The usage of the

plane 183 element, initialized as a 8-node quadrilateral with plane strain assumptions and pure displacement formulation, is shown in line 3. Lines 5 to 9 in Source Code 5.1 are used to set the material properties, namely Elastic Modulus (line 6), Poisson's ratio (line 7) and unit weight of the concrete (line 8).

Source Code 5.1: Geometry command file

```
1  et,matid,13,3,1,0
2
3  !et,matid,183,0,,2
4
5  mpdele,all,matid
6  mp,ex,matid,31000000000
7  mp,prxy,matid,0.2
8  mp,dens,matid,2640
9  etcontrol,off,off
```

Additionally to geometry and element type, Westergaard's added mass has to be applied in the *Geometry* section. To achieve that, the mass per unit area is computed at each node, using Eq. 3.4. Multiplying this mass with the area of influence of the specific node gives a nodal lumped mass. These masses can be manually added to the nodes by using the *point mass* tool.

Mesh

The element type is already defined in the command file, shown in Source Code 5.1. To create the mesh, edge sizing is applied. Furthermore to get a mapped mesh, the *Face Meshing* function has to be added to the *Mesh* branch.

Named Selection

Creating *Named Selections* is a crucial part for applying forces. The submerged upstream nodes are named 'usdam'. Furthermore all elements in the domain are named 'dam'. The purpose of this labelling is explained in the *Static structural* section below.

Static Structural

In this part of the analysis, applying the boundary conditions (loadings and supports) is explained. Support BCs are restrictions of the nodal u_x and u_y degrees of freedom at the bottom of the dam, which can be introduced by using the *Fixed Support* function for the bottom edge. For earthquake loading the desired acceleration values can be added. Those act on the masses of the dam as well as on the added mass. Applying the gravitational acceleration in the same manner would lead to wrong results, because the added mass is not to be accelerated with this value. Therefore implementing gravitational loading as well as hydrostatic pressure is done in a command file, which is shown in Source Code 5.2.

Source Code 5.2: Static Structural command file

```
1  /prep7
2  sfgrad,pres,0,y,0,-9810
3  cmsel,s,usdam,node
4  esln,s
5  sf,all,pres,966285
6  allsel,all
7
8  cmacel,dam,,9.81
9  /solu
```

Line 2 to 6 add the hydrostatic pressure distribution and line 8 the gravitational loading. More specifically, line 2 creates a linearly increasing function with a slope of -9810 ($= \rho_{water}\, g$). At line 3 the nodes on which this function is acting are defined, namely the previously created 'usdam'

nodes. The maximum pressure, which implicitly defines the water level, is given in line 5 for the Koyna Dam test case. The implementation of forces introduced by gravity on the *Named Selection*, 'dam', is shown in line 8.

5.2 Koyna Dam

5.2.1 Introduction

The Koyna Dam is located in the soutwestern part of India. It is a 850 m wide and 103 m high gravity dam. The relevant parameters for the validation are taken from Chopra & Chakrabarti (1973). Furthermore the dam and its dimensions are illustrated in Fig. 5.3.

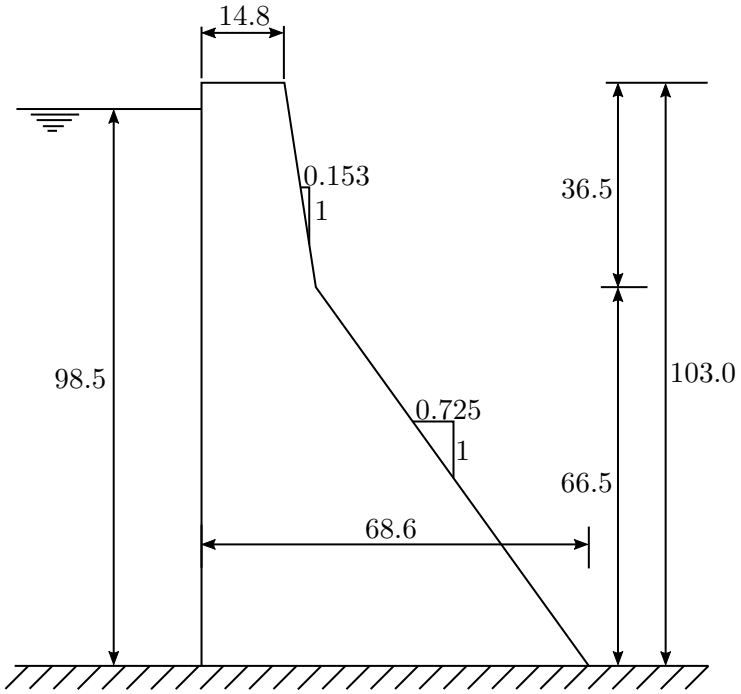


Fig. 5.3: Geometry Koyna Dam

The illustrated dam allows a simplification to 2-D, assuming plane strain conditions and a thickness of 1 m. The foundation is assumed to be rigid. For constructing the Koyna Dam four different concrete mixes were used. Nevertheless Chopra & Chakrabarti (1973) stated that a homogeneous material can be assumed for the purpose of a FEM simulation. Thus the modulus of elasticity results in 31000 MPa, the unit weight of concrete takes the value 2640 $\frac{kg}{m^3}$ and the Poisson ratio is 0.2. As for the unit weight of the reservoir water, 1000 $\frac{kg}{m^3}$ is assumed. As usual the gravitational acceleration is 9.81 $\frac{m}{s^2}$. Furthermore the earthquake accelerations, needed for Westergaard's added mass method, are $a_x = 5.0\,\frac{m}{s^2}$ (horizontal) and $a_y = 0.0\,\frac{m}{s^2}$ (vertical).

5.2.2 Setting up

After having explained the geometry and the material properties in section 5.2.1, additional parameters needed for the simulation like meshing, setting BCs, general FEM settings and loading combinations have to be declared. A fairly rough mesh is chosen and all explained computations

are carried out on this. Later on, for the convergence study, this grid is further refined. The initial mesh has 6 horizontal edge divisions, 8 vertical above the kink and 10 vertical below the kink (Fig. 5.6). As explained at the beginning of the chapter, most computations use the loading cases dead load and hydrostatic pressure. Regarding general FEM settings, a plane strain approach in combination with a consistent mass matrix is used. As for the support conditions, the bottom nodes are fixed in all coordinate directions. The input for 2DGDA is shown in Source Code 5.3.

Source Code 5.3: Koyna Dam input 2DGDA

```
# --------------------  Geometry and Meshing ----------------------------
Geometry = [[0,0],[68.6,0],[20.4,66.5],[14.8,103],[0,103]]
nelem_vertical_above_kink = 8 # if no kink set to 0
nelem_vertical_below_kink = 10
nelem_horizontal = 6

# --------------------  Material properties ----------------------------
NU = 0.2 # Poisson ratio
EE = 31000000000 # elastic modulus
rho_concrete = 2640 # rho concrete
rho_water = 1000 # rho water
gravity = 9.81 # gravity

# -------------------------  loading -----------------------------------
# -------------------------  combinations ------------------------------
# dead load = 1 , hydrostatic = 2, earthquake = 3
loading = [1,2,3]

# -------------------  waterload settings ------------------------------
water_level = 98.5

# -------------------  earthquake settings ------------------------------
ax = 5.0 # horizontal acceleration
ay = 0.0 # vertical acceleration

# ----------------------  Boundary conditions --------------------------
# give specific points = 1, fix all bottom nodes = 2
BC_method = 2
BC = [1,2,3,4,5] # fixed nodes, just for method 1

# -------------------------  FEM settings ------------------------------
mass_method = 2 # Diagonally lumped mass matrix (DLMM = 1), consistant mass matrix (CMM) = 2
PS = 1            # plane strain = 1, plane stress = 0
```

5.2.3 Evaluation

First the directional displacements in x direction are compared between 2DGDA and ANSYS® plane 13 element. The initial mesh is used and all 3 possible loading cases are considered. The resulting values are shown in Fig. 5.4. The error adds up and reaches a maximum of 0.337 mm, which are about 0.8 %, at the dams crest. This error most likely is introduced at the dynamic loading case, where 2DGDA uses numeric integration to get the lumped masses.

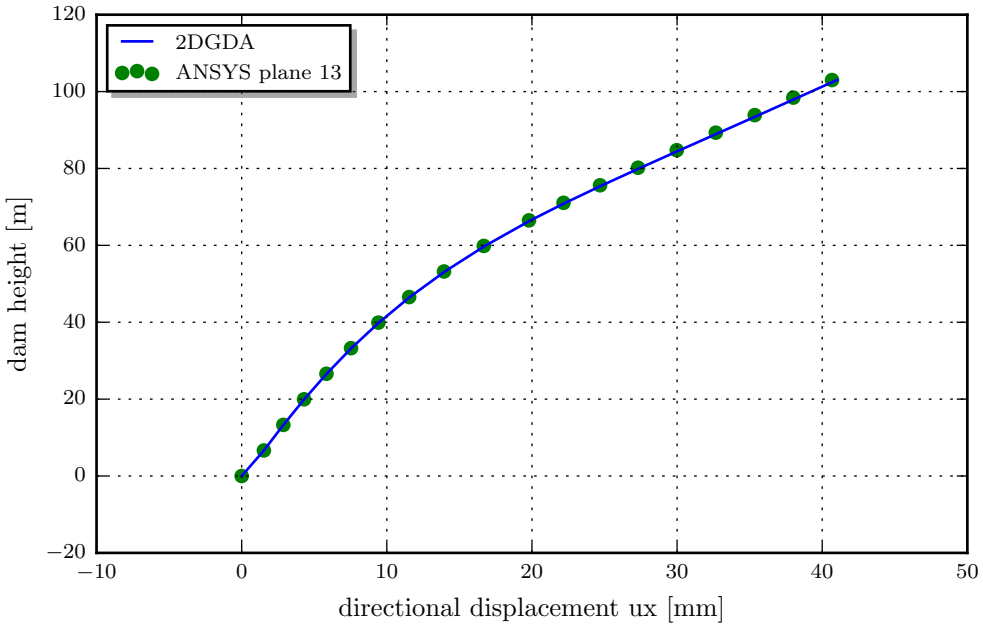


Fig. 5.4: Comparison of directional displacements for all loading cases, Koyna Dam

Now the directional displacements in x direction are compared between ANSYS® plane 13 and plane 183 element types, and 2DGDA. The initial mesh is used. Furthermore only dead load and hydrostatic loading is considered. Since 2DGDA and ANSYS® plane 13 use the same techniques for the explained assumptions, the computed values are exactly the same. Compared to the more sophisticated plane 183 element, the maximum error at the top left node yields 2.6 %, which are 0.188 mm. The comparison is shown in Fig. 5.5

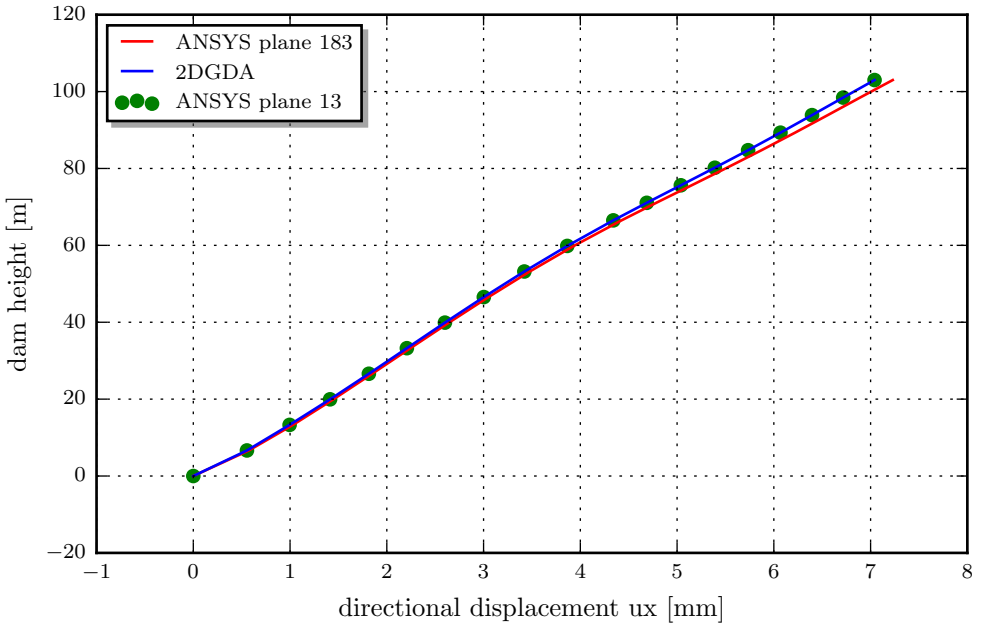
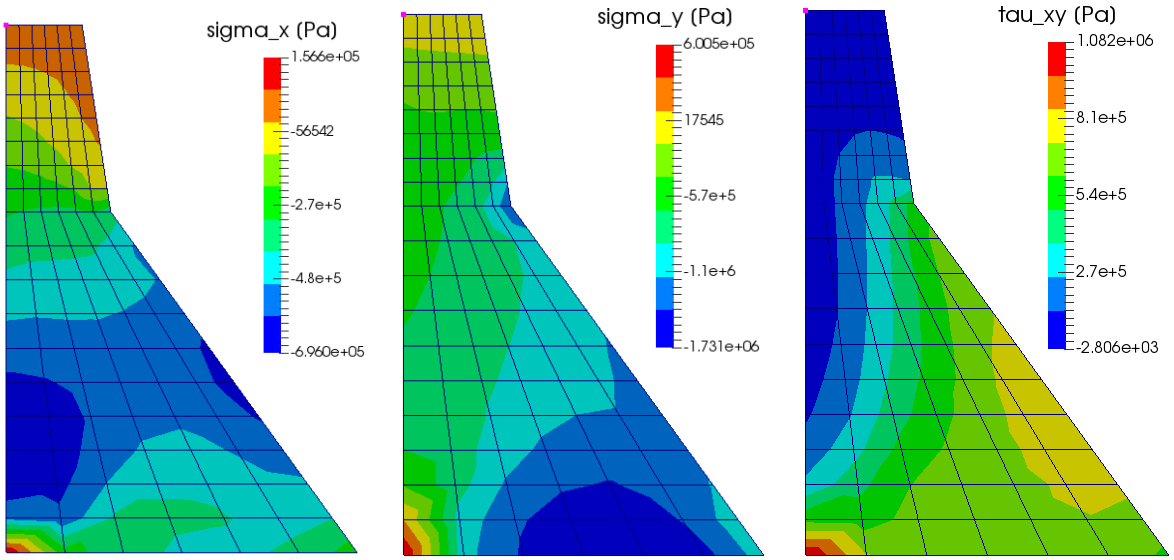
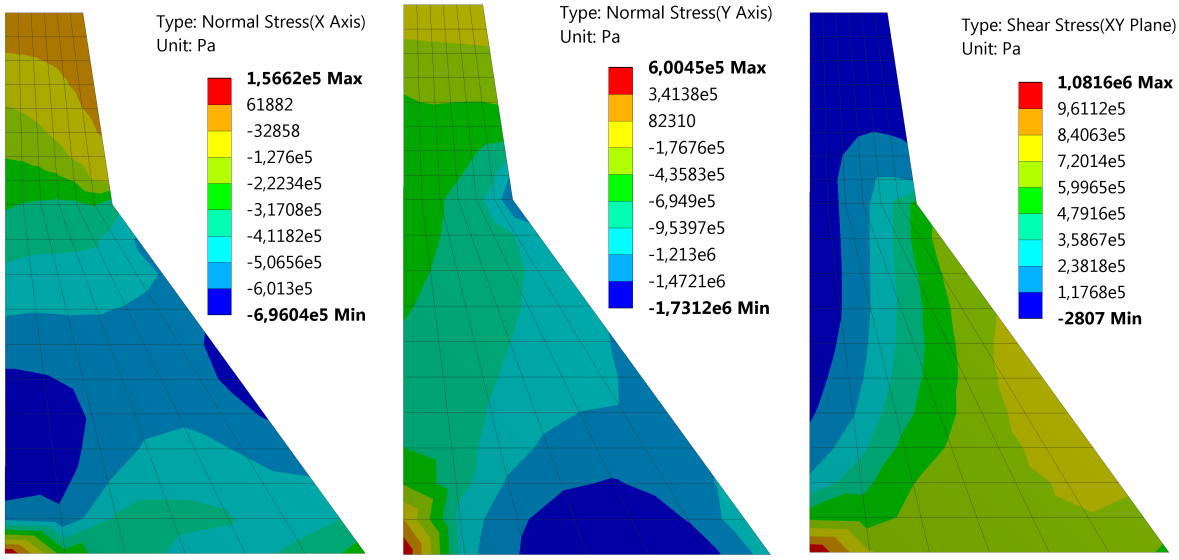


Fig. 5.5: Comparison of directional displacements, hydrostatic and dead load only, Koyna Dam

The next comparison concerns the in-plane stresses $\boldsymbol{\sigma}_x$, $\boldsymbol{\sigma}_y$ and $\boldsymbol{\tau}_{xy}$. Those are computed for the initial mesh as well as hydrostatic loading and dead load. Fig. 5.6a, Fig. 5.6b and Fig. 5.6c show the contour plots computed by 2DGDA and visualized in ParaView, whereas the contours exported from ANSYS® are illustrated in Fig. 5.6d, Fig. 5.6e and Fig. 5.6f. Both use the same scaling and colour mapping, thus a visual comparison of all stresses in the domain is possible. At first sight it can be seen that the computed values from 2DGDA and ANSYS® are very alike. The maximum directional stresses are at the dam's heel, whereas the location of the minimum directional stresses varies.



(a) $\boldsymbol{\sigma}_x$ Koyna Dam, 2DGDA **(b)** $\boldsymbol{\sigma}_y$ Koyna Dam, 2DGDA **(c)** $\boldsymbol{\tau}_{xy}$ Koyna Dam, 2DGDA



(d) $\boldsymbol{\sigma}_x$ Koyna Dam, plane 13 **(e)** $\boldsymbol{\sigma}_y$ Koyna Dam, plane 13 **(f)** $\boldsymbol{\tau}_{xy}$ Koyna Dam, plane 13

Fig. 5.6: Contour plots normal stresses, Koyna Dam

To be able to estimate the errors, the converged values have to be found. The convergence study is carried out in ANSYS® (plane 183 element) for x displacements at the top left node. To get the converged values, the mesh is refined (doubling up the edge divisions) until the error between the compared values is smaller than 0.1 %. That leads to a converged x displacement value at the top left node of 7.248 mm. Fig. 5.7 shows the directional displacements in x direction of ANSYS®

plane 13 and plane 183, as well as 2DGDA's results for an increasing number of elements. The graph shows that the results converge with refining the mesh. Furthermore ANSYS® plane 183 element shows the fastest convergence, since the higher order element reduces numerical errors. This element type already returns sufficiently accurate values, even for the initial mesh. Thus by refining the mesh the results improve just slightly. As for 2DGDA's results and the values computed by the plane 13 element, the error resulting from computations at the initial mesh is 2.9%. By doubling up the edge divisions the error to the converged value reduces to 0.9%, which can be considered as sufficiently accurate. By increasing the number of divisions again, the error reduces to 0.3%.

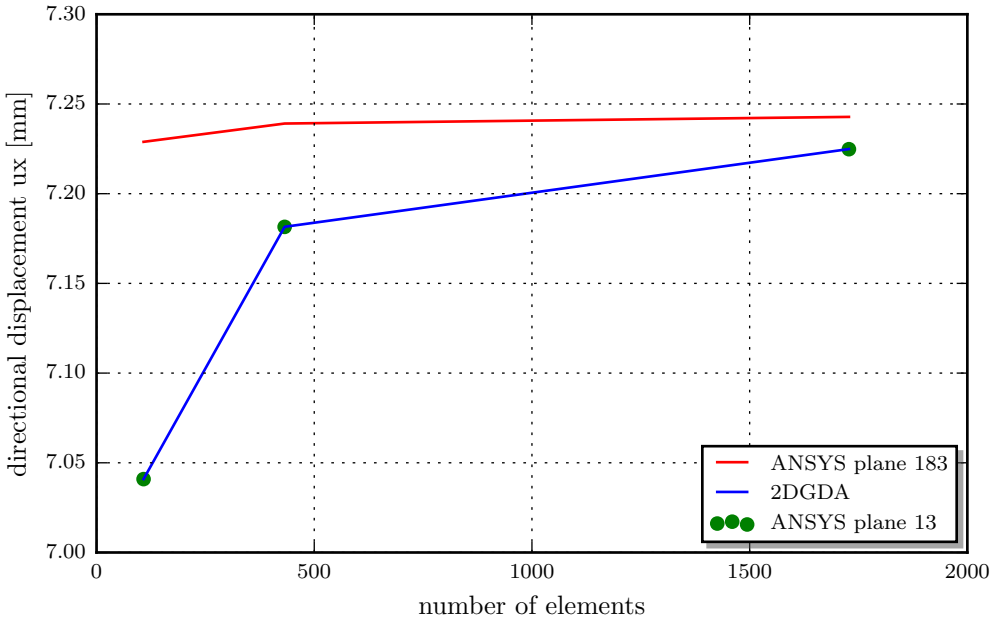


Fig. 5.7: Convergence Koyna Dam

5.3 Pine Flat Dam

5.3.1 Introduction

The 122 m high and 560 m wide Pine Flat Dam is located on the Kings River in California. The relevant parameters for this gravity dam are taken from Løkke & Chopra (2014). The geometry, illustrated in Fig. 5.8, is a simplified version of the real geometry, fulfilling the needs of 2DGDA's mesher.

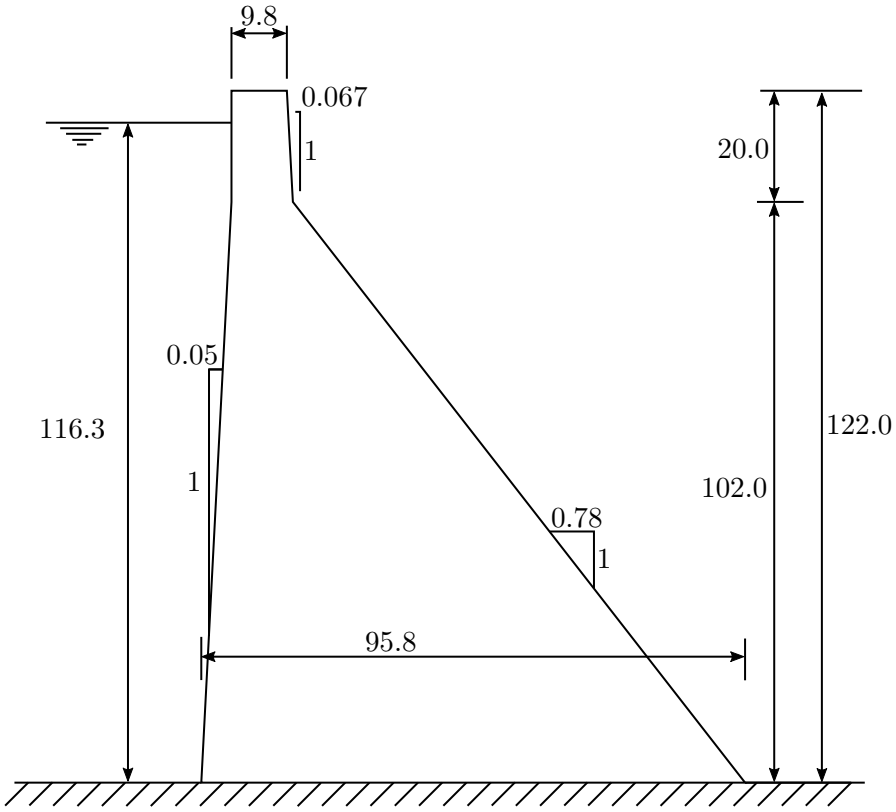


Fig. 5.8: Geometry Pine Flat Dam

The 3-D gravity dam can be simplified to 2-D, assuming plane strain conditions. The thickness of the slice is 1 m and the foundation is assumed to be rigid. Furthermore the elastic modulus is 22400 MPa, the unit weight of the used concrete is 2480 $\frac{kg}{m^3}$ and the Poisson ratio is 0.2. As for the unit weight of water and the accelerations the same values as in section 5.2 are used.

5.3.2 Setting up

Setting up this test case is done similarly to the previous one, explained in section 5.2.2. The initial grid is shown in Fig. 5.12. For the convergence study this grid is further refined. The initial mesh has 6 horizontal edge divisions, 7 vertical above the kink and 15 vertical below the kink. The considered loading cases are defined individually for each of the investigations, but mostly dead load and hydrostatic pressure are used. The computations are done with a general plane strain approach and a consistent mass matrix. Similar to the Koyna Dam case, the degrees of freedom of the bottom nodes are restricted. The input for 2DGDA is shown in Source Code 5.4.

Source Code 5.4: Pine Flat Dam input 2DGDA

```
# --------------------  Geometry and Meshing ---------------------------
Geometry = [[0,0],[95.8,0],[16.24,102],[14.9,122],[5.1,122],[5.1,102]]
nelem_vertical_above_kink = 7 # if no kink set to 0
nelem_vertical_below_kink = 15
nelem_horizontal = 6

# --------------------  Material properties ----------------------------
NU = 0.2 # Poisson ratio
EE = 22400000000 # elastic modulus
rho_concrete = 2480 # rho concrete
rho_water = 1000 # rho water
gravity = 9.81 # gravity

# -------------------------  loading -----------------------------------
# -------------------------  combinations ------------------------------
# dead load = 1 , hydrostatic = 2, earthquake = 3
loading = [1,2,3]

# -------------------  waterload settings ------------------------------
water_level = 98.5

# -------------------  earthquake settings -----------------------------
ax = 5.0 # horizontal acceleration
ay = 0.0 # vertical acceleration

# ----------------------  Boundary conditions --------------------------
# give specific points = 1, fix all bottom nodes = 2
BC_method = 2
BC = [1,2,3,4,5] # fixed nodes, just for method 1

# -------------------------  FEM settings ------------------------------
mass_method = 2 # Diagonally lumped mass matrix (DLMM = 1), consistant mass matrix (CMM) = 2
PS = 1            # plane strain = 1, plane stress = 0
```

5.3.3 Evaluation

First the directional displacements of the upstream nodes in x direction are evaluated. The results are compared between ANSYS® plane 13 element and 2DGDA. All loading cases are considered. The resulting values are shown in Fig. 5.9. The error increases with the dam height and reaches a maximum of 1.06 mm, which are 2.18 %, at the dam's crest. As before, the error is introduced at the dynamic loading case, where 2DGDA uses numeric integration to get the lumped masses.

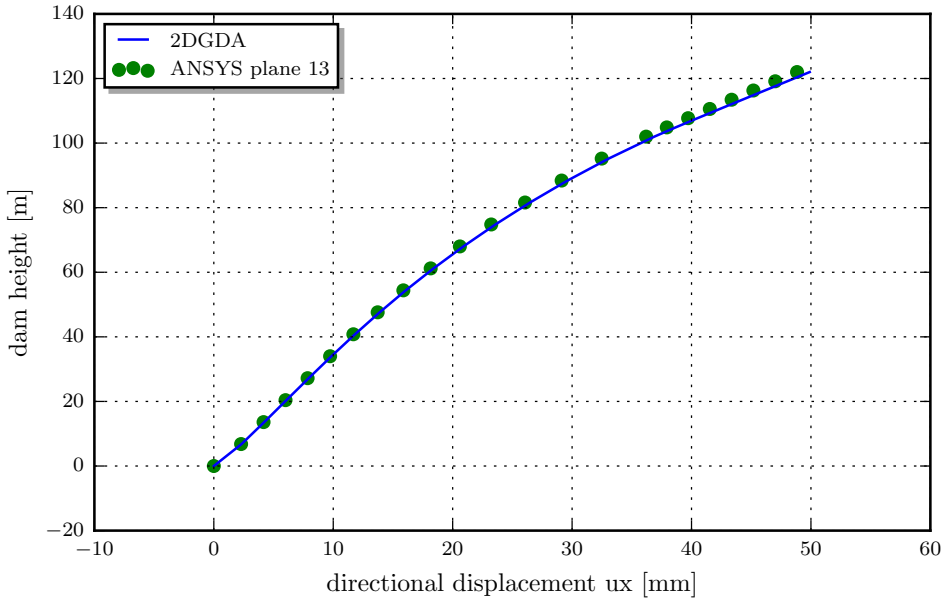


Fig. 5.9: Comparison of directional displacements for all loading cases, Pine Flat Dam

Second the directional displacements in x direction are compared between ANSYS® plane 13 and plane 183 element types, and 2DGDA. The initial mesh is used. Furthermore only dead load and hydrostatic loading is considered. As mentioned, 2DGDA and ANSYS® plane 13 use the same FEM techniques, thus the computed values are exactly the same. Compared to the more sophisticated plane 183 element the error increases with the height of the dam and reaches a maximum of 0.16 mm, which are 1.8 %, at the dam's crest. The comparison is shown in Fig. 5.10.

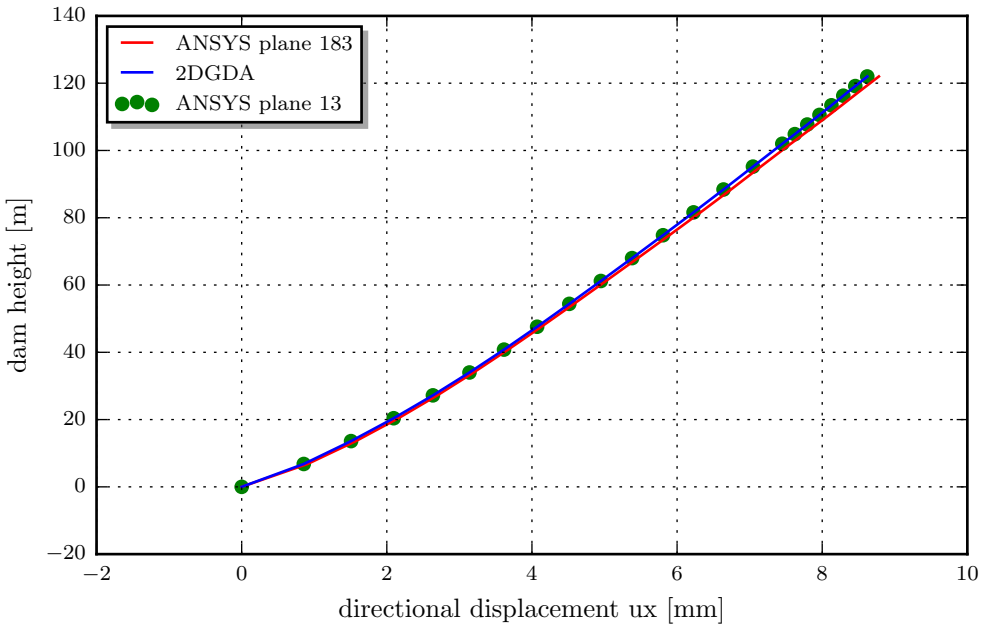


Fig. 5.10: Comparison of directional displacements, hydrostatic and dead load only, Pine Flat Dam

The converged x displacement at the top left node of the dam is 8.793 mm. Like before this value is found by refining the mesh until the error to the further refined mesh is less than 0.1 %. Fig. 5.11 shows the directional displacements, computed with dead load and hydrostatic loading, for an increasing number of elements. The computations are carried out with ANSYS® plane 13 and plane 183 element types, as well as 2DGDA. Like for the Koyna Dam test case the values computed with the plane 183 element are sufficiently accurate, even for the initial mesh. The error of the plane 13 element and 2DGDA, compared to the converged value, is 2.0 %. The next refinement step reduces this error to 0.6 %. Increasing the number of division again, leads to an error of 0.2 %.

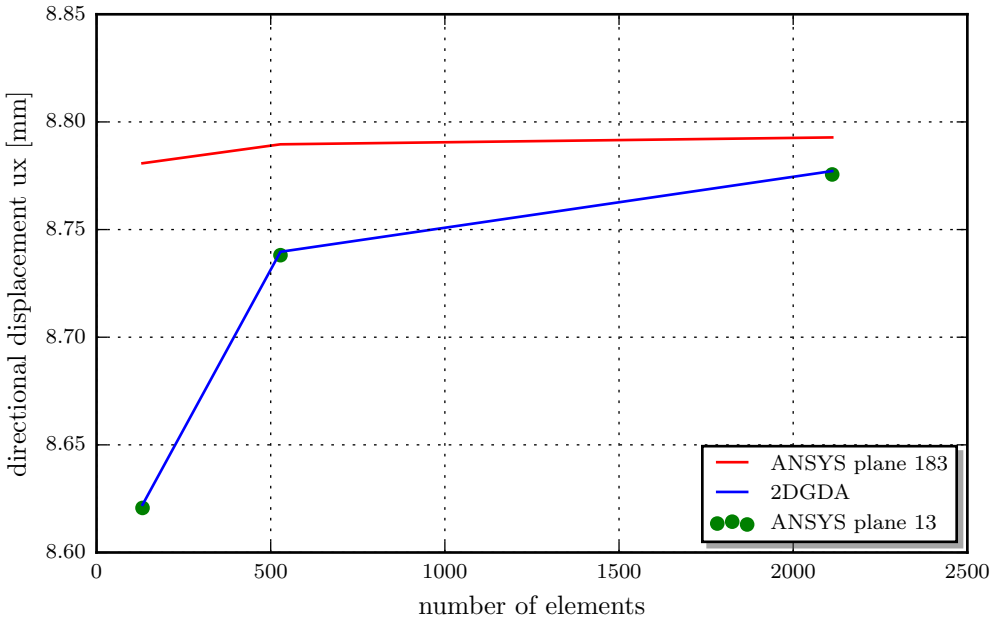
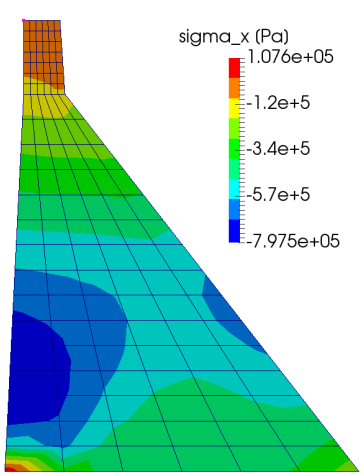
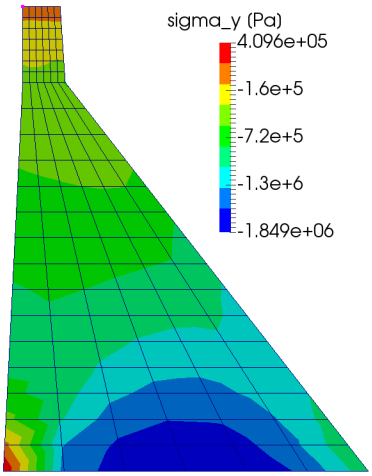


Fig. 5.11: Convergence Pine Flat Dam

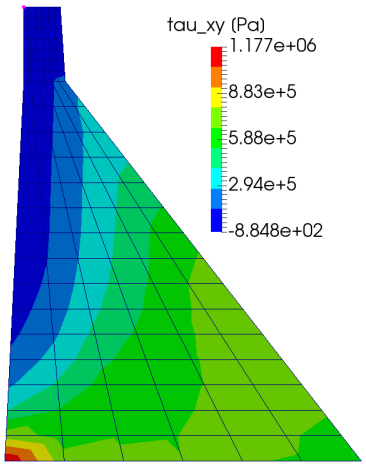
The comparison between the directional in-plane stresses of the Pine Flat Dam is shown in Fig. 5.12. Those stresses are computed for the initial mesh and for hydrostatic and dead load loading cases. Fig. 5.12a, Fig. 5.12b and Fig. 5.12c show the contour plots computed by 2DGDA and visualized in ParaView. The contours exported from ANSYS® (plane 13 element) are illustrated in Fig. 5.12d, Fig. 5.12e and Fig. 5.12f. Both use the same scaling and colour mapping, thus a visual comparison of all stresses in the domain is possible. At first sight it can be seen that the computed values from 2DGDA and ANSYS® are very alike. The maximum directional stresses are at the dam's heel, whereas the location of the minimum directional stresses varies.



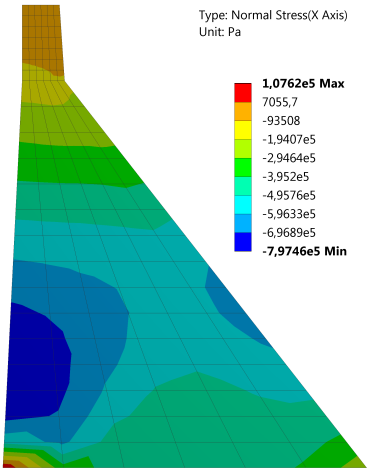
(a) σ_x Pine Flat Dam, 2DGDA



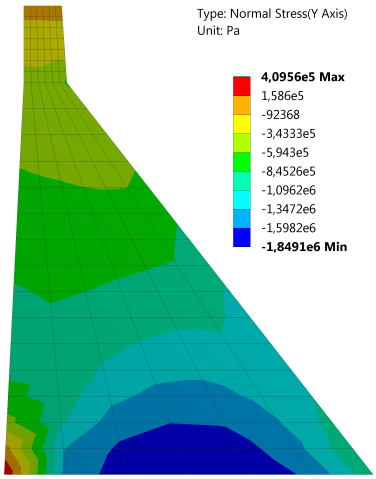
(b) σ_y Pine Flat Dam, 2DGDA



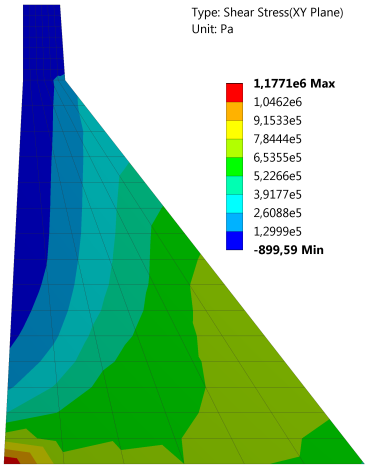
(c) τ_{xy} Pine Flat Dam, 2DGDA



(d) σ_x Pine Flat Dam, plane 13



(e) σ_y Pine Flat Dam, plane 13



(f) τ_{xy} Pine Flat Dam, plane 13

Fig. 5.12: Contour plots normal stresses, Pine Flat Dam

6 Conclusion

6.1 Summary

Within the framework of this thesis a software for the analysis of concrete gravity dams is developed. The main goals are simplicity, dependency on open-source software only and the possibility of future implementations. Another important aim is to apply the FEM for stability analyses. The software is named 2DGDA, which stands for Two Dimensional Gravity Dam Analyses. The FEM computations in this thesis can be summarized as follows

- Pre-Processing: As part of the pre-processing the needed variables, like material properties, loading cases, FEM settings, etc., are imported from the input file. For creating the mesh a function was developed that returns nodal coordinates and their connectivity by using the outer points of the geometry as well as edge divisions as input.
- Processing: The processing contains building up and solving the equations. The code uses basic FEM formulations, with no advanced methods and an isoparametric approach. Solving the resulting linear system of equations leads to nodal displacements.
- Post-Processing: This step contains deriving values from the computed displacements (e.g. stresses). Furthermore the results can be visualized and therefore interpreted in a correct way. For this purpose a third-party software, ParaView, is used. ParaView is open-source and therefore fulfils the previously stated requirement of depending on open-source products only.

The implementation of the steps listed above to the software is explained in great detail within this thesis. The programming language chosen is Python 2.7. To facilitate future implementations an object oriented structure is focused. Furthermore a detailed user manual for the developed software is included. The validation of the software is carried out for two famous gravity dam test cases, namely Koyna Dam and Pine Flat Dam, are discussed. The comparison is done between 2DGDA and two different element types available in ANSYS® Workbench. The first element is a linear quadrilateral element with basic FEM formulations, namely plane 13. The second one is a higher order quadrilateral, using advanced FEM techniques, which is called plane 183. The validation shows that for the plane 13 element 2DGDA reproduces the exact same values as ANSYS®. Even for the plane 183 element the error becomes negligible for fine meshes.

In conclusion this program provides a good alternative to other available numerical packages for the computation of two dimensional gravity dam cases. Keeping in mind the program's limitations allows the user to perform Finite Element computations with an easy to handle software. It is worth mentioning that the program returns stresses and displacements without assessing the dam stability. The user has to choose the correct guideline and assess the dam safety as part of the post-processing.

6.2 Outlook

As stated above, one of the most important goals of this thesis is to facilitate future implementations. Therefore the present code aims to serve as a basis for further research. A wide variety of future research topics are available. Some of those are listed below

- Development of a more sophisticated mesher with less limitations
- Implementation of higher order FEM
- Investigation joint opening at the foundation and cracking at critical parts inside the dam geometry
- Consideration of material non-linearity
- Implementation of dam foundation interaction

List of Figures

List of Tables

List of Listings

References

Anandarajah, A. (2011). *Computational methods in elasticity and plasticity: solids and porous media*. Springer Science & Business Media.

ANSYS® Academic Research, Release 17.2, Help System. (2016). Ansys mechanical apdl element reference [Computer software manual].

Avila, L. e. a. (2010). *The vtk user's guide, 11th edition*. Kitware New York.

Ayachit, U. (2015). The paraview guide [Computer software manual]. USA. Retrieved 2016-09, from `http://www.paraview.org/paraview-guide/`

Bathe, K. J. (2006). *Finite element procedures*. Klaus-Jurgen Bathe.

Bower, A. F. (2009). *Applied mechanics of solids*. CRC press.

Center of Aerospace Structures. (2016). *Introduction to finite element methods*. University of Colorado Boulder. Retrieved 2016-03, from `http://www.colorado.edu/engineering/CAS/courses.d/IFEM.d`

Chopra, A. K., & Chakrabarti, P. (1973). The koyna earthquake and the damage to koyna dam. *Bulletin of the Seismological Society of America*, *63*(2), 381–397.

Gould, P. L. (1994). *Introduction to linear elasticity*. Springer.

Ibrahimbegovic, A. (2009). *Nonlinear solid mechanics: theoretical formulations and finite element solution methods* (Vol. 160). Springer Science & Business Media.

Kattan, P. I. (2010). *Matlab guide to finite elements: an interactive approach*. Springer Science & Business Media.

Kuo, J. S.-H. (1982). *Fluid-structure interactions: added mass computations for incompressible fluid*. University of California, College of Engineering, Earthquake Engineering Research Center.

Løkke, A., & Chopra, A. K. (2014). Response spectrum analysis of concrete gravity dams including dam-water-foundation interaction. *Journal of Structural Engineering*, *141*(8), 04014202.

Rüberger, T., & Zechner, J. (2008). *Introduction to Finite Element Method*. Graz University of Technology, Institute of Structural Analysis.

Saouma, V. (2006). Merlin theory manual. *University of Colorado, Boulder, Boulder*.

Westergaard, H. M. (1933). Water pressures on dams during earthquakes. *Trans. ASCE*, *98*, 418–432.