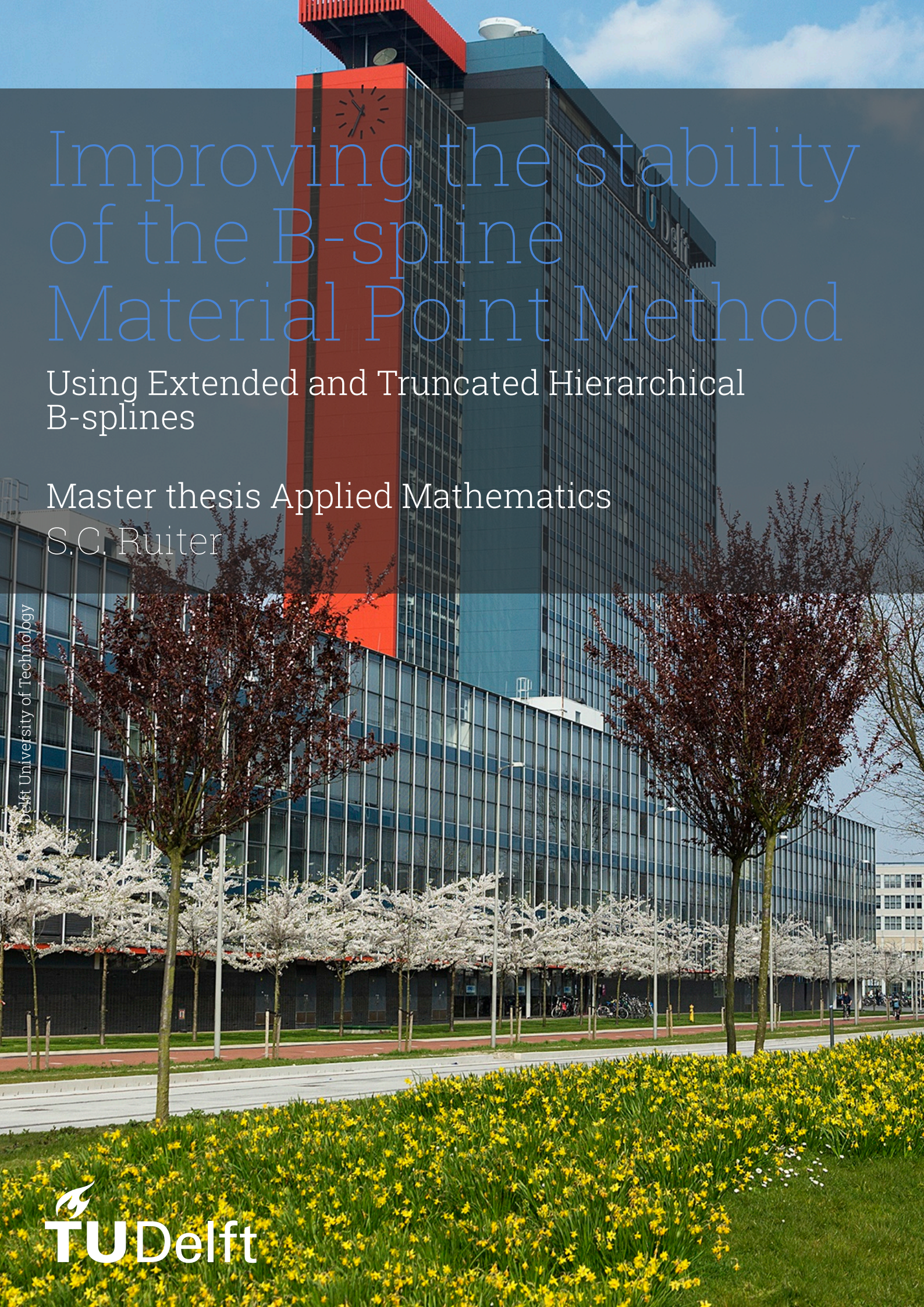
Improving the stability of the B-spline Material Point Method

Using Extended and Truncated Hierarchical B-splines

Master thesis Applied Mathematics

S.C. Ruiter



Delft University of Technology

TU Delft

Improving the stability of the B-spline Material Point Method

Using Extended and Truncated Hierarchical B-splines

by

S.C. Ruiter

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday December 22, 2022 at 10:30 AM.

Student Name	Student Number
Stijn Ruiter	4311582

Supervisors:	Dr. M. Möller
	Dr. D. Toshniwal
Thesis committee:	Dr. M. Möller
	Dr. D. Toshniwal
	Prof. Dr. H.M. Schuttelaars

An electronic version of this thesis is available at `https://repository.tudelft.nl/`.

TU Delft

Abstract

The Material Point Method (MPM) is a numerical method primarily used in the simulation of large deforming or multi-phase materials. An example of such a problem is a landslide or snow simulation. The MPM uses Lagrangian particles (material points) to store the interested physical quantities. These particles can move freely in the spatial domain. Each time step, these physical quantities are projected on a background grid, which is necessary to evaluate these quantities at a future time step. Once all necessary properties are projected, the acceleration on this grid can be updated using the conservation of linear momentum. At last, this updated acceleration can be projected to the particles to compute their updated velocity, position and other quantities.

The main problem addressed in this thesis arises in the projection of the particles to the background grid. The use of linear basis functions in the classical MPM causes instabilities, which can be resolved by using higher order B-splines. However, another problem arises when particles move between the background cells. Since these particles move freely through the spatial domain, it is possible for some cells to by *nearly-empty*. In this thesis, the use of Extended B-splines is explored to increase the stability in these areas by deactivating unstable B-splines and *extending* stable ones.

Furthermore, higher dimensional B-splines are constructed by a tensor product of one dimensional B-splines. The scalability can be a problem, as these B-splines cannot be refined locally. Therefore, Truncated Hierarchical B-splines are introduced to locally refine a geometry. The effectiveness of this technique in combination with the EB-splines is investigated.

The thesis shows that the use of EB-splines in the context of MPM increases the stability in the case of nearly-empty cells, and can also improve the quality of the solution near the boundary. Furthermore, in the neighborhood of high stress concentrations, the THB-splines have a similar accuracy as the regular B-splines, while drastically reducing the computational costs. In combination with EB-splines, THB-splines can be used to accurately refine a geometry in the presence of nearly-empty cells.

Before these techniques can widely be used in applications, the possibly of negative elements have to be investigated when using EB-splines, as this is not guaranteed in the extension. Furthermore, the use of THB-splines in the context of MPM has only been studied for a predefined refinement. In real applications, the presence of stress concentrations is not known and a more adaptive local refinement technique in the context of MPM has to be explored.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
FEM	Finite Element Method
MPM	Material Point Method
BSMPM	B-Spline Material Point Method
EB-spline	Extended B-spline
WEB-spline	Weighted Extended B-spline
HB-spline	Hierarchical B-spline
THB-spline	Truncted Hierarchical B-spline
ETHB-spline	Extended THB-spline

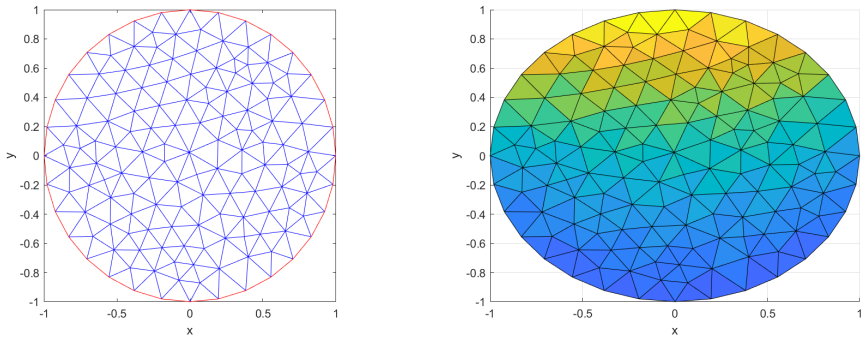
Symbols

Symbol	Definition	Unit
E	Young's Modulus	[Pa]
n_i	Degrees of freedom	[1]
n_p	Number of particles	[1]
P	Nominal stress	[Pa]
t	Time	[s]
u	Displacement	[m]
v	Velocity	[m/s]
x_p	Particle position	[m]
ϵ	Strain	[1]
λ, μ	Lame's parameters	[Pa]
ν	Poisson ratio	[1]
Ξ	Knot vector	
ρ	Density	[kg/m^3]
σ	Cauchy stress	[Pa]
ϕ	Basis function	[1]
Ω	MPM background grid	

1 Introduction

In solid mechanics, it is often required to determine the characteristics of materials, such as the deformation of metal beam under an external load. However, it is not always feasible or even possible to investigate the properties by performing experiments. In situations where a simple material is studied without complex geometries, it is possible to express these characteristics by solving partial differential equations analytically.

In many applications, it is impossible to solve the describing equations exactly and simulations are necessary to approximate a solution. In these simulations, numerical techniques are used which try to describe the characteristics of interest as accurate as possible. However, numerical simulations can be incredibly challenging, and many techniques are developed to overcome these challenges. Therefore, different methods may be suitable for different challenges.



(a) Typical example of a FEM mesh.

(b) Example of an arbitrary scalar solution computed using Finite Element Analysis.

Figure 1.1: Example of a typical FEM mesh and a scalar solution resulting from Finite Element Analysis.

A well-known numerical technique is the Finite Element Method, which uses a mesh to represent the material, see figure 1.1a. By using this finite mesh, a solution for the problem at hand can be approximated, see figure 1.1b. In the presence of an external load or other sources on the material, the representing mesh will move and deform to describe the resulting changes. In the case of large and complex 3D-geometries, these meshes can be difficult and time-consuming to generate. Furthermore, in the case of large deformation, this mesh can become extremely distorted which leads to excessive approximation errors and non-physical solutions. Ultimately, a new mesh has to be generated to continue the computation. An example of a such a problem is a landslide or avalanche, where a composition of various materials has large deformations which are difficult to simulate using methods such as the FEM.

To avoid the time-consuming re-meshing, other methods can be used which avoid these challenges. One approach to avoid the mesh distortion is to use a fixed mesh. Over time, the mesh does not move and instead flow of the material through a cell is tracked. When the mesh moves with the material,

solving and updated the material properties is straight-forward. However, if the mesh stays static, this is not the case due to nonlinear convective terms.

In these *mesh-based* methods, the materials are sometimes hard to accurately describe, which also prompted the development of *mesh-free* methods. In these types of methods, the material is not represented on a mesh, but is discretized by a cloud of particles. These particles hold the material properties and can move freely in the spatial domain, avoiding the convective terms and naturally handling the large deformations. However, the interaction between particles is not clearly defined and since the particles are used to approximate the continuous fields, the gradient of these fields is not trivial. This complicates solving the continuum equations.

To overcome the problems of both *mesh-based* and *mesh-free* methods, the Material Point Method (MPM) was introduced by Sulsky et al. [31] as an extension of the Particle in Cell (PIC) method [15]. This method uses aspects of both methods. As this name suggest, the PIC method uses particles to represent the material. These particles carry only a mass and position, but they are not used to solve the problems equations. Instead, a fixed background mesh is used to discretize the continuous fields and the material properties of the particles are projected on the mesh. Therefore, it uses the advantages of both methods described, but does have its drawbacks. The PIC method suffers from excessive computational dissipation as a result from transferring momentum between the grid and particles [8]. An improved version of this method is the Fluid Implicit Particle (FLIP) method, introduced in [8]. This method is further improved and adapted for solid mechanics by Sulsky et al., and eventually dubbed it the Material Point Method. A discretization example for MPM of the same problem as figure 1.1 is displayed in figure 1.2.

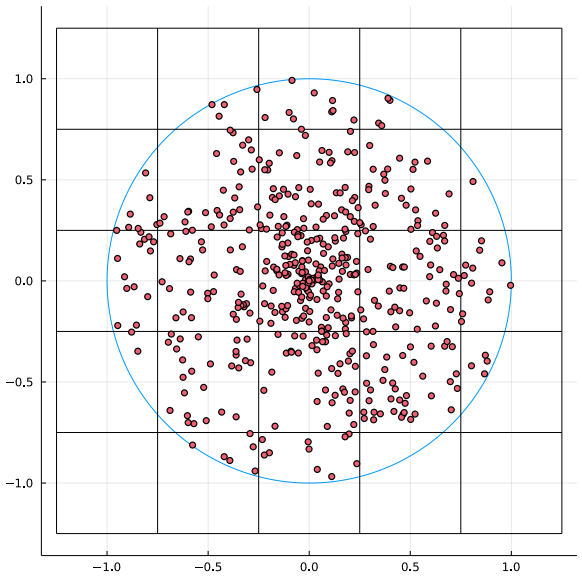


Figure 1.2: Example of the same domain as in figure 1.1 in the MPM context. The material points (particles) represent a material defined on a unit circle. A background grid is generated for the MPM computations. Each time step, the particle information is projected to the background grid.

The Material Point method itself does also have some challenges to overcome, primarily in the projection of the particles to the background grid. The traditional MPM uses linear Lagrange basis functions to interpolate the material properties between the particles and background mesh. These basis functions have a discontinuous derivative. These derivatives are necessary in the computation of the material stresses and when particles move between grid cells, these discontinuities cause unnatural jumps in the stress field. This phenomenon is called the *grid-crossing error*.

To overcome this problem, various approaches have been suggested, and the approach used in this thesis is the use of higher polynomial order B-splines instead of the linear basis functions. The higher order B-splines are smoother, they have continuous derivatives which resolves the unnatural jumps in the stress field. This does resolve the grid-crossing error, but still has a problem when cells are nearly-empty. When a grid cell is almost empty, the mass of that cell will be small, while the force on the

cell is still present. The resulting acceleration is therefore unnaturally large. These accelerations are interpolated to the material points, giving them unnatural velocities. This causes instabilities in MPM simulations and is a well-known issue which is dubbed the *nearly-empty cell* problem.

This problem becomes even more prominent when smaller grid sizes are used. A common approach to reduce the approximation error in a numerical method, is to reduce the grid size, but this increases the likelihood of cell to be nearly-empty as well. Therefore, it can be challenging to refine a grid, especially near the edges of a material where the problem mostly occurs.

The goal of this thesis is to improve the stability of B-spline MPM simulations by reducing the nearly-empty cell problem using Extended B-spline functions (EB-splines)[17, 16]. EB-splines are an extension to the regular B-splines, where B-splines with a small support (nearly-empty) are deactivated and nearby stable B-splines are extended to incorporate the deactivate B-splines support. This increases the stability of the active B-splines.

Furthermore, Truncated Hierarchical B-splines (THB-splines)[14] are introduced as a local refinement technique for B-splines to refine a background mesh of MPM. It does this by splitting B-splines into multiple B-splines of a smaller support, which can increase the accuracy of the spline approximation. In combination with EB-splines, this thesis aims to refine the MPM solution near the edge of a material, while avoiding the instabilities of the nearly-empty cell problem.

The outline of this thesis is as follows. In chapter 3, the Material Point Method is described in more detail. After the MPM is outlined, the problems described above can be displayed in chapter 3.6. In chapter 3.7, a brief overview is given of the various methods developed to overcome these problems. In chapter 4, isogeometric analysis is introduced, which is a computational approach where the B-splines and its extensions originate from. In chapter 5, the benchmark used in this thesis are explained and the results of the MPM simulations are displayed and discussed. In chapter 6, the final conclusions and recommendations are presented.

2

Physical model

2.1. Eulerian and Lagrangian specification

There are generally two specifications in continuum mechanics to describe the motion of a material. One is using the *Eulerian* description and the other is the *Lagrangian* description. Both specification should provide identical results, but the complexity of the equation varies depending on the chosen specification.

In the *Eulerian* description, a control volume is fixed in the spatial domain and the material is described as it moves through this control volume. In the *Lagrangian* description, a part of the material (e.g., a particles) is chosen to be the control volume and this control volume is tracked as it moves through the domain. One of the differences in these frameworks arises when the *material derivative* or the total derivative of a field f inside a control volume,

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla f, \tag{2.1}$$

where $\mathbf{v}$ is the velocity vector and ∇f the gradient of f. The change of a quantity f is described by both the partial derivative with respect to time and an advection term $\mathbf{v} \cdot \nabla f$. In the Eulerian specification, there is a change of the quantity f in the control volume as material is *advected* in and out of this volume. Therefore, this term does not vanish in the Eulerian specification. In the Lagrangian specification, the control volume moves with the material and therefore there is no material advection, thus the material derivatives become

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t}. \tag{2.2}$$

2.2. Motion

In the Material Point Method, the Lagrangian specification is used to describe the material in motion. In the following section, the equations of motions for these Lagrangian particles are defined.

Let t_0 be the initial time and let the domain Ω_t be the considered configuration of the domain at a given time t. The reference domain at t_0 is indicated by Ω_0. A particle has an initial position indicated by $\mathbf{X} \in \Omega_0$. As the material deforms, the particle moves in the spatial domain. The deformed (Eulerian) position $\mathbf{x} \in \Omega_t$ is then mapped by $\mathbf{x} = \phi(\mathbf{X}, t)$. The displacement of a particle can then be expressed as

$$\mathbf{u}(\mathbf{X}, t) = \mathbf{x}(\mathbf{X}, t) - \mathbf{x}(\mathbf{X}, t_0) = \mathbf{x}(\mathbf{X}, t) - \mathbf{X}. \tag{2.3}$$

The particle displacement is the joint result of the deformation and rigid body forces. Using the formulation of this displacement, the particle velocity can be derived as

$$\mathbf{v}(\mathbf{X}, t) = \frac{\partial \mathbf{u}}{\partial t} = \frac{\partial \mathbf{x}(\mathbf{X}, t)}{\partial t}, \tag{2.4}$$

which is the *Lagrangian* specification of the velocity. Similarly, the expression for the material acceleration is

$$\mathbf{a}(\mathbf{X}, t) = \frac{\partial \mathbf{v}}{\partial t} = \frac{\partial^2 \mathbf{x}(\mathbf{X}, t)}{\partial t^2}. \tag{2.5}$$

The term deformation is already used in the section above. While the material is not advected in the control volume, the volume does deform. The deformation of a control volume is the transformation from the reference volume to the current volume and is defined as

$$\mathbf{F}(\mathbf{X}, t) = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} = 1 + \frac{\partial \mathbf{u}}{\partial \mathbf{X}} = \mathbf{I} + \nabla_0 \mathbf{u}, \tag{2.6}$$

where ∇_0 indicates the gradient with respect to the reference framework. This expression for the deformation is an important part in the Lagrangian specification and is required in numerical computation, such as the Material Point Method.

2.3. Governing equations

In continuum mechanics, the motions of a continuum are described by the balance laws[29, 26], mainly the *conservation of mass*, *conservation of linear momentum*, *conservation of angular momentum* and *conservation of energy*. These balance equations are used in the Material Point Method to describe and compute the evolution of the particle motion.

Firstly, the *continuity equation* or conservation of mass can be written in the Eulerian specification as

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = \rho \nabla \cdot \mathbf{v}, \tag{2.7}$$

which as stated before is guaranteed by the material point method, as each particle has a constant mass.

Secondly, the conservation of linear momentum is expressed as

$$\rho \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{b}, \tag{2.8}$$

where $\boldsymbol{\sigma}$ is the Cauchy-stress tensor and $\mathbf{b}$ the (external) body forces applied to the material. This equation is primarily used in the basic MPM and is described in more detail in the next sections.

The conservation of angular momentum requires the stress tensor to be symmetric, which is a requirement for the constitutive relation, see section A.

For completeness, there is also the conservation of energy. However, as the default MPM is used for isothermal problems, the conservation of energy is not solved[3].

2.4. Constitutive equation

The previous section are general descriptions of the balance laws and are valid for all types of materials. When solving the conservation of linear momentum for a specific type of model, an additional set of equations is needed to describe the behavior of a material in more detail. Such sets of equations are called constitutive relations. Sevaral types of equations can be required depending on the specific problem or research field at hand, such as electro- or thermodynamic behavior. However, in this thesis, we are mainly concerned with the deformation of solid. For this, an expression for the Cauchy stress $\boldsymbol{\sigma}$ in the conservation of linear momentum is needed.

Depending on the type of material, different equations for this stress can be found in literature. In this thesis, we limit ourselves only to *hyperelastic* material, which are a specific type of material where the current state of the stress can be expressed by the deformation with respect to the reference state. A linear extension of these materials is called Neo-Hookean materials and for an isotropic Neo-Hookean material with large deformations, the constitutive relation can be expressed[37, 28] as

$$\boldsymbol{\sigma} = \lambda J^{-1} \log J \mathbf{I} + J^{-1} \mu (\mathbf{F}\mathbf{F}^T - \mathbf{I}), \tag{2.9}$$

where $J = \det \mathbf{F}$ and λ, μ are the Lamé constants. These constants are defined by

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \mu = \frac{E}{2(1+\nu)}, \tag{2.10}$$

with Young's modulus E and Poison's ration ν, which are material properties. λ and μ are also referred to as the first and second Lamé constants, respectively.

The above constitutive relation can be simplified in the case of small deformations as

$$\boldsymbol{\sigma} = \lambda \mathrm{tr}\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right)\mathbf{I} + 2\mu\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right). \tag{2.11}$$

A more detail explanation and derivation of these constitutive models can be found in appendix A.

In the material point method, the constitutive model as a function of the deformation is used. However, these models are often expressed in terms of the strain. The strain tensor $\boldsymbol{\epsilon}$ is defined as

$$\boldsymbol{\epsilon} = \frac{1}{2}\left(\nabla_0 \mathbf{u} + (\nabla_0 \mathbf{u})^T\right) = \mathrm{sym}\left(\nabla_0 \mathbf{u}\right). \tag{2.12}$$

Using this definition and equation 2.6, the linear model for small deformations can be expressed as

$$\boldsymbol{\sigma} = \lambda \mathrm{tr}\left(\boldsymbol{\epsilon}\right) + 2\mu\boldsymbol{\epsilon}, \tag{2.13}$$

which is the well-known *Hooke's law* constitutive model.

2.5. Boundary and initial conditions

Besides the constitutive model, the initial configuration of the material and the behavior at the edge are required. Therefore, we must specify the initial and boundary conditions.

The initial conditions are imposed as

$$\mathbf{u}(\mathbf{X}, t_0) = \mathbf{u}_0(\mathbf{X}), \tag{2.14}$$
$$\mathbf{v}(\mathbf{X}, t_0) = \mathbf{v}_0(\mathbf{X}), \tag{2.15}$$
$$\boldsymbol{\sigma}(\mathbf{X}, t_0) = \boldsymbol{\sigma}_0(\mathbf{X}), \tag{2.16}$$

where the right-hand side are problem specific functions describing the initial configuration of the material.

The boundary of the domain Ω is indicated by $\Gamma = \partial\Omega$. Both essential and natural boundary conditions are possible to be present in a given problem. Therefore, let $\Gamma = \Gamma_u \cap \Gamma_t$ where Γ_u is the part of the boundary containing essential boundary conditions and Γ_t the traction boundary conditions.

The essential boundary conditions are then defined by

$$\mathbf{u}(\mathbf{X}, t) = \mathbf{u}_b(t), \mathbf{X} \in \Gamma_u \tag{2.17}$$

and the traction boundary conditions are

$$\boldsymbol{\sigma}(\mathbf{X}, t) \cdot \mathbf{n} = \boldsymbol{\tau}(t), \mathbf{X} \in \Gamma_t \tag{2.18}$$

where $\mathbf{n}$ is the outward unit normal vector.

2.6. Weak formulation

The conservation of linear momentum in equation 2.8 is a strong formulation. In many numerical methods, such as the Finite Elements Method, a weak formulation of the partial differential equation is used. The resulting *weak solution* only holds with respect to specific test functions.

Let Ω be the domain of the problem and let $\mathscr{W}$ be the test function space of sufficiently smooth functions which are zero on the boundary where essential boundary conditions are imposed. The weak formulation of equation 2.8 can be derived by multiplying by a test function $\phi \in \mathscr{W}$ and integrating over the domain Ω, which results in

$$\begin{aligned}
\int_\Omega \phi\rho\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}\mathrm{d}\Omega &= \int_\Omega \phi\left(\nabla\cdot\boldsymbol{\sigma} + \rho\mathbf{b}\right)\mathrm{d}\Omega \\
&= \int_\Omega \phi\nabla\cdot\boldsymbol{\sigma}\mathrm{d}\Omega + \int_\Omega \phi\rho\mathbf{b}\mathrm{d}\Omega \\
&= \int_\Omega \nabla\cdot\left(\phi\boldsymbol{\sigma}\right)\mathrm{d}\Omega - \int_\Omega \nabla\phi\cdot\boldsymbol{\sigma}\mathrm{d}\Omega + \int_\Omega \phi\rho\mathbf{b}\mathrm{d}\Omega \\
&= \int_\Gamma \phi\mathbf{n}\cdot\boldsymbol{\sigma}\mathrm{d}\Gamma - \int_\Omega \nabla\phi\cdot\boldsymbol{\sigma}\mathrm{d}\Omega + \int_\Omega \phi\rho\mathbf{b}\mathrm{d}\Omega
\end{aligned}$$

where the last step is acquired by using the *divergence theorem*[35] and where $\Gamma = \partial\Omega$ indicates the boundary of Ω. Let $a = \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}$ be the acceleration and let $\mathscr{V}$ be the trial space of sufficiently smooth function which respect the boundary conditions. Then the weak formulation of the problem is

$$\text{Find } \mathbf{a} \in \mathscr{V} : \int_{\Omega} \phi \rho \mathbf{a} \mathrm{d}\Omega = \int_{\Gamma} \phi \mathbf{n} \cdot \boldsymbol{\sigma} \mathrm{d}\Gamma - \int_{\Omega} \nabla \phi \cdot \boldsymbol{\sigma} \mathrm{d}\Omega + \int_{\Omega} \phi \rho \mathbf{b} \mathrm{d}\Omega, \forall \phi \in \mathscr{W} \tag{2.19}$$

3

Material Point Method

In the previous section, the physical model is outlined with its necessary equations. These equations will be used in this section to describe the Material Point Method (MPM) in detail.

The Material Point Method is a numerical method introduced by Sulsky [31] and is an extension of the Particle in Cell (PIC) method [15].

The PIC method is primarily used in fluid mechanics. It uses Lagrangian particles, which carry only mass and position. To discretize the continuous fields, an Eulerian background mesh is used. Therefore, it can use the advantages of both a Lagrangian and Eulerian based method. However, this method suffers from excessive computational dissipation as a result from transferring momentum between the grid and particles. [8]

An improved version of this method is the Fluid Implicit Particle (FLIP) method, introduced in [8]. This method is further improved and adapted for solid mechanics by Sulsky, and eventually named it the Material Point Method.

The MPM uses material points (or particles) which hold all physical properties of the problems material, such as mass, velocity and stress. These particles can move freely through the domain, but they are not directly used to solve the conservation of linear momentum.

Instead, a background grid is used to solve this equation. At the start of a time step, all physical quantities stored in the particles is mapped to the nodes of the background grid, figure 3.1a. Using the mapped physical quantities at the nodes, the weak formulation of equation 2.19 is solved similar to the Finite Element Method (FEM), figure 3.1b.

After the *updated* physical quantities are computed at the nodes, the particles are updated using the inverse mapping and the background grid is reset to its original state, figure 3.1c.

This is a distinction with FEM, where the nodes represent the material, which results in inaccuracies if large deformations of the cells are considered. Since the MPM has freely moving particles and a static background grid, this problem never occurs.

3.1. Space discretization

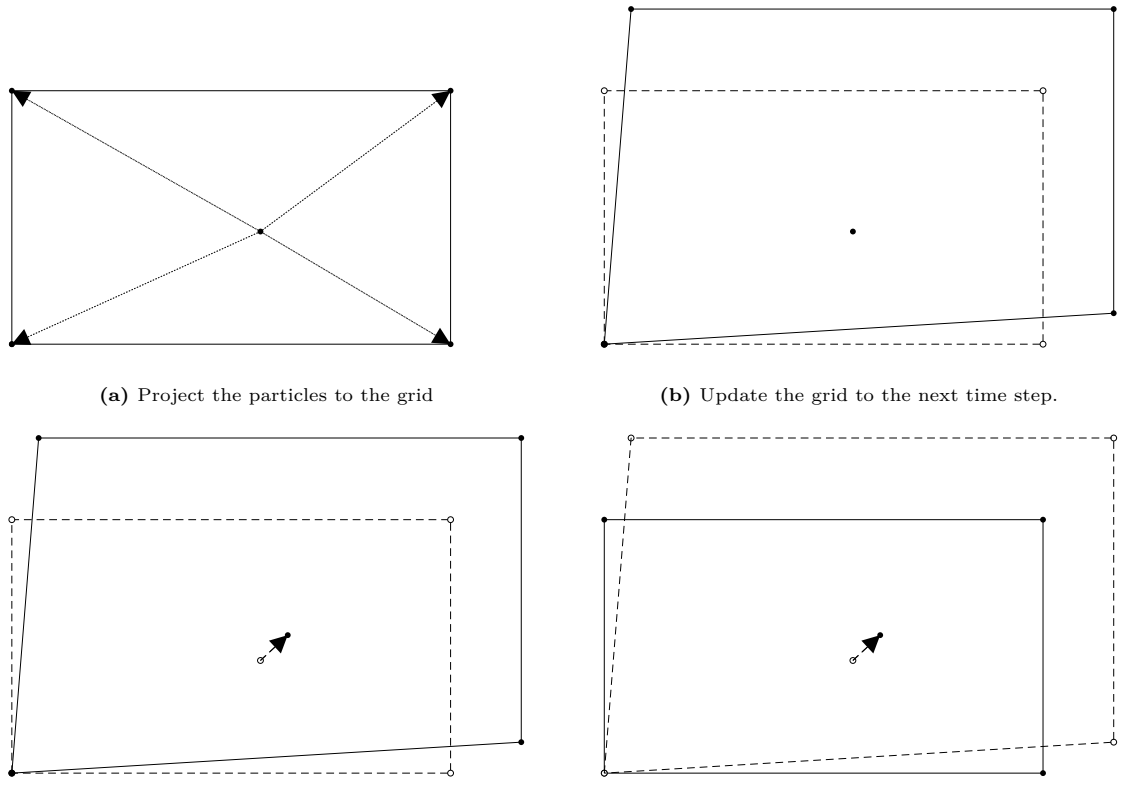
Similar as in FEM, the weak formulation of equation 2.19 is solved on the background grid of MPM. For simplicity, this grid usually is a structured (regular) grid, but this is not required, e.g., [22].

On this background grid, let $\phi_i \in \mathscr{W}$ be a set of n_i linear independent basis function which span a subspace $\mathscr{W}^h \subset \mathscr{W}$. A similar trial subspace $\mathscr{V}^h \subset \mathscr{V}$ can be defined. This reduces the problem to a n_i-dimensional problem and the weak formulation becomes

$$\text{Find } \mathbf{a}^h \in \mathscr{V}^h : \int_\Omega \phi_i \rho \mathbf{a}^h \mathrm{d}\Omega = \int_\Gamma \phi_i \mathbf{n} \cdot \boldsymbol{\sigma} \mathrm{d}\Gamma - \int_\Omega \nabla \phi_i \cdot \boldsymbol{\sigma} \mathrm{d}\Omega + \int_\Omega \phi_i \rho \mathbf{b} \mathrm{d}\Omega, \forall \phi_i \in \mathscr{W}^h \tag{3.1}$$

We can approximate the acceleration by a linear combination of these basis functions, namely

$$\mathbf{a}(\mathbf{X}, t) \approx \mathbf{a}^h(\mathbf{X}, t) = \sum_{i=1}^{n_i} \phi_i(\mathbf{X}) \mathbf{a}_i(t), \tag{3.2}$$



(a) Project the particles to the grid

(b) Update the grid to the next time step.

(c) Project the updated quantities back to the particles

(d) Reset the background grid to its original state.

Figure 3.1: Illustration of the MPM procedure during a single time step.

where $\mathbf{a}^h$ denotes the approximation of $\mathbf{a}$.

Substituting this approximation into the weak formulation equation, this results in

$$\sum_{j=1}^{n_i} \mathbf{a}_j \int_\Omega \phi_i \rho \phi_j \mathrm{d}\Omega = \int_\Gamma \phi_i \mathbf{n} \cdot \boldsymbol{\sigma} \mathrm{d}\Gamma - \int_\Omega \nabla \phi_i \cdot \boldsymbol{\sigma} \mathrm{d}\Omega + \int_\Omega \phi_i \rho \mathbf{b} \mathrm{d}\Omega. \tag{3.3}$$

To solve this equation for all $\mathbf{a}_j$, we define the matrix elements for $\mathbf{M}$ as

$$M_{i,j} = \int_\Omega \phi_i \rho \phi_j \mathrm{d}\Omega \tag{3.4}$$

and the vector elements $\mathbf{F}^{\text{tract}}, \mathbf{F}^{\text{int}}, \mathbf{F}^{\text{ext}}$ as

$$\mathbf{F}_i^{\text{tract}} = \oint_\Gamma \phi_i \mathbf{n} \cdot \boldsymbol{\sigma} \mathrm{d}\Gamma \tag{3.5}$$

$$\mathbf{F}_i^{\text{int}} = -\int_\Omega \nabla \phi_i \cdot \boldsymbol{\sigma} \mathrm{d}\Omega \tag{3.6}$$

$$\mathbf{F}_i^{\text{ext}} = \int_\Omega \phi_i \rho \mathbf{b} \mathrm{d}\Omega, \tag{3.7}$$

Note that for a d-dimensional problem, equation 3.3 is a set of d equation, one for each dimension. This also mean that each vector element $\mathbf{F}_i$ consists of d elements for each dimension and so does $\mathbf{a}_j$.

In matrix form, equation 3.3 is written as

$$\mathbf{M}\mathbf{a} = \mathbf{F}^{\text{tract}} + \mathbf{F}^{\text{int}} + \mathbf{F}^{\text{ext}}, \tag{3.8}$$

where $\mathbf{a}$ is the acceleration vector containing the coefficients $\mathbf{a}_j$. Once the matrices and vectors are assembled, the acceleration vector can be computed using linear algebra.

In many application, the mass matrix is lumped to reduce the computational cost of the MPM simulations. A lumped mass matrix is a diagonal matrix where the diagonal elements are the row sums of the mass matrix of equation 3.4. Since the matrix is diagonal, the acceleration in equation 3.8 can be calculated directly without solving the system of equations.

However, this lumping procedure induces additional inaccuracies as is observed in [22]. To avoid these inaccuracies and to better investigate the effects of the stabilizing methods introduced in this thesis, mass lumping will not be used.

3.2. Integration points

In the regular FEM approach, the integrals of the matrix and vector elements of equation 3.3 can be assembled directly at each time step, as the values are known at the nodes. MPM uses a mapping of the particles in the domain to the nodes of the background grid, and an additional step is necessary.

First, the material points (particles) need to be defined. Let $\{P_p\}$ be a set of n_p particles, each with a position $\mathbf{X}_p \in \Omega$, a mass m_p and a volume V_p. In the classical MPM, each particle does have a volume V_p, but particles do not have a domain $\Omega_p \subset \Omega$. In extended MPM versions such as GIMP or CPDI, these domains are introduced, see section 3.7, but they will not be used in this thesis. The deformation of the particle volume is tracked by the matrix $\mathbf{F}$ and is initially set to the identity matrix $\mathbf{I}$.

By defining in the material points this way, we will treat the material points as integration points, where the volume V_p is used as weight, thus

$$\int_\Omega f(x)\mathrm{d}\Omega = \sum_{p=1}^{n_p} V_p f_p. \tag{3.9}$$

Substituting this equation into equation 3.3, the weak formulation becomes the following summation

$$\sum_{j=1}^{n_i}\left[\sum_{p=1}^{n_p} V_p\rho_p\phi_i(\mathbf{X}_p)\phi_j(\mathbf{X}_p)\right]\mathbf{a}_j^h = \int_\Gamma \phi\mathbf{n}\cdot\boldsymbol{\sigma}\mathrm{d}\Gamma - \sum_{p=1}^{n_p} V_p\nabla\phi_i\cdot\boldsymbol{\sigma}_p\Big|_{\mathbf{X}_p} + \sum_{p=1}^{n_p}\phi_i(\mathbf{X}_p)V_p\rho_p\mathbf{b}(\mathbf{X}_p). \tag{3.10}$$

Using $m_p = \rho_p V_p$, each matrix element from the previous section can be written as a particle sum;

$$M_{i,j} = \sum_{p=1}^{n_p}\phi_i(\mathbf{X}_p)\phi_j(\mathbf{X}_p)m_p \tag{3.11}$$

$$\mathbf{F}_i^{\text{tract}} = \oint_\Gamma \phi\mathbf{n}\cdot\boldsymbol{\sigma}\mathrm{d}\Gamma \tag{3.12}$$

$$\mathbf{F}_i^{\text{int}} = -\sum_{p=1}^{n_p} V_p\nabla\phi_i(\mathbf{X}_p)\cdot\boldsymbol{\sigma}(\mathbf{X}_p), \tag{3.13}$$

$$\mathbf{F}_i^{\text{ext}} = \sum_{p=1}^{n_p}\phi_i(\mathbf{X}_p)m_p\mathbf{b}(\mathbf{X}_p) \tag{3.14}$$

3.3. Imposing boundary conditions

In this thesis, two types of boundary conditions are used, *traction* or *natural* boundary conditions and *Dirichlet* or *essential* boundary condition. If a problem imposes any of these boundary conditions, they must be imposed. However, this is not always obvious as to how they are treated.

3.3.1. Natural boundary condition

The traction boundary conditions or *natural* boundary conditions are imposed implicitly by the weak formulation in equation 3.12,

$$\mathbf{F}_i^{\text{tract}} = \oint_\Gamma \phi\mathbf{n}\cdot\boldsymbol{\sigma}\mathrm{d}\Gamma. \tag{3.15}$$

In the case of homogeneous traction, i.e., $\boldsymbol{\tau} = \mathbf{n}\cdot\boldsymbol{\sigma} = \mathbf{0}$, this term vanishes. However, this is not always the case and if non-homogeneous boundary conditions $\tau(t)$, this term is not always clear.

For a 1D-problem with $\Omega = [x_0, x_1]$, this problem reduces to

$$F_i^{\text{tract}} = \phi_i \tau|_{x_0}^{x_1} = \phi_i(x_1)\tau(x_1, t) - \phi_i(x_0)\tau(x_0, t), \tag{3.16}$$

where the boundary conditions $\tau(x_0, t)$ and $\tau(x_1, t)$ are known.

For a 2D-problem, this integral becomes more complex. In this thesis, a 2D background grid is generated by a tensor product, and the test function is written as

$$\phi_{i,j}(x, y) = \phi_i(x)\phi_j(y). \tag{3.17}$$

For a simple boundary condition, where a rectangular domain is given by $\Omega = [x_0, x_1] \times [y_0, y_1]$, the traction term becomes

$$\begin{aligned} F_i^{\text{tract}} &= \oint_\Gamma \phi_{i,j} \mathbf{n} \cdot \boldsymbol{\sigma}_t \mathrm{d}\Gamma \\ &= \int_{x_0}^{x_1} \phi_i(x)\phi_j(y_0)\boldsymbol{\tau}(x, y_0, t)\mathrm{d}x \\ &\quad + \int_{y_0}^{y_1} \phi_i(x_1)\phi_j(y_0)\boldsymbol{\tau}(x_1, y, t)\mathrm{d}y \\ &\quad + \int_{x_1}^{x_0} \phi_i(x)\phi_j(y_1)\boldsymbol{\tau}(x, y_1, t)\mathrm{d}x \\ &\quad + \int_{y_1}^{y_0} \phi_i(x_0)\phi_j(y)\boldsymbol{\tau}(x_0, y, t)\mathrm{d}y. \end{aligned}$$

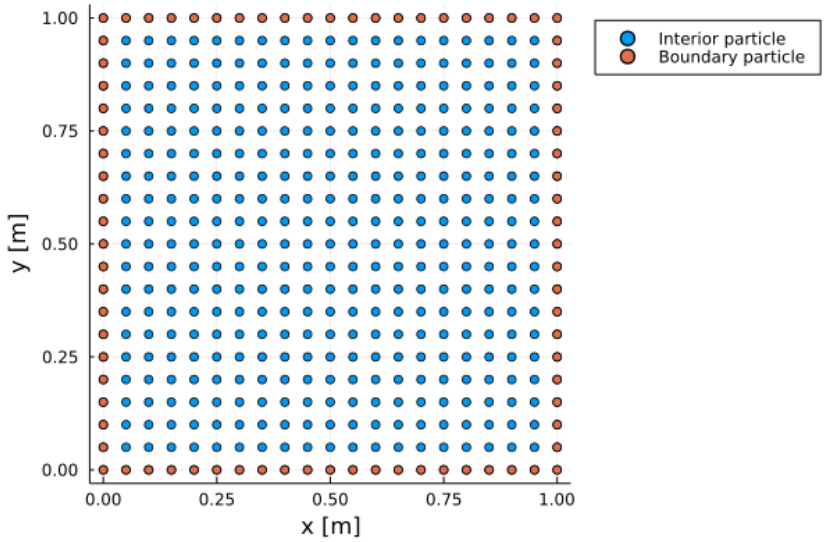


Figure 3.2: Example of boundary and interior particles on a unit square.

In general, the boundary is does not coincide with any background grid. Instead, a set of boundary particles is generated together with the interior material points. These particles are mass-less and only carry the necessary boundary information to correctly represent the boundary $\Gamma = \partial\Omega$. The traction boundary term can then be expressed as

$$\begin{aligned} \mathbf{F}_i^{\text{tract}} &= \int_\Gamma \phi_i(\mathbf{X})\boldsymbol{\tau}(\mathbf{X}, t)\mathrm{d}\Gamma \\ &= \sum_{p=1}^{n_{bp}} \phi_i(\mathbf{X}_p)\boldsymbol{\tau}(\mathbf{X}_p, t)L_p, \end{aligned} \tag{3.18}$$

where L_p is the length of the line segment of boundary particle p and n_{bp} the number of boundary particles. In general, the number of boundary particles per line segment should be larger than the number of particles per cell[9].

In this thesis, the analytic solution of each problem is known and therefore the exact location of these boundary particles and the traction is known.

3.3.2. Essential boundary condition

For the Dirichlet or *essential* boundary conditions, it is not obvious how they can be imposed on the background grid, especially since the material points in MPM can move freely in the domain. Various methods can be used to impose them, and we make a distinction between grid-conforming boundary conditions, where the boundary coincides with the background grid and non-conforming boundary conditions where the boundary is located arbitrarily with respect to the background grid.

In the case of conforming boundary conditions, a similar approach as in FEM can be used. Suppose the numbering of degrees of freedom (nodes) of the background grid are rearranged such that first we have the interior not-prescribed nodes and the prescribed nodes afterwards. In this case, the matrix representation of equation 3.8 can be rewritten as[35]

$$\begin{bmatrix} \mathbf{M}_{ii} & \mathbf{M}_{ib} \\ \mathbf{M}_{bi} & \mathbf{M}_{bb} \end{bmatrix} \begin{bmatrix} \mathbf{a}_i \\ \mathbf{a}_b \end{bmatrix} = \begin{bmatrix} \mathbf{F}_i \\ \mathbf{F}_b \end{bmatrix} \tag{3.19}$$

where the subscript i consists of all the nodes with non-prescribed unknowns, and b the prescribed boundary nodes. Since the Dirichlet boundary conditions are known, we know $\mathbf{a}_b$ and the equation can be reduced to

$$\mathbf{M}_{ii}\mathbf{a}_i = \mathbf{F}_i - \mathbf{M}_{ib}\mathbf{a}_b. \tag{3.20}$$

This approach is relatively simple and many MPM application choose the background grid in such a way that this method can be utilized.

However, this method cannot be used for arbitrary background grids or moving boundaries where the conformity is not guaranteed. In this thesis, the penalty method is employed to impose the Dirichlet condition weakly on the boundary. If $u = g$ on Γ be the boundary condition, a penalty condition is generated to replace this boundary condition. Let $\epsilon > 0$, then the penalty condition is given[1, 4] by

$$\frac{\partial u}{\partial n} = \epsilon^{-1}(u - g) \text{ on } \Gamma. \tag{3.21}$$

This penalty is then used in the weak formulation directly and the solution satisfies the boundary condition as $\epsilon \to 0$. However, choosing the right penalty constant ϵ is not obvious, as a small value of ϵ leads to an ill-conditions problem and is thus depended on the grid size[25, 21].

3.4. Basis functions

In the classic Material Point method, linear Lagrange basis function are used similar as in the finite element method. For each cell of the background grid, the Lagrange basis functions are defined by

$$l_j(x) = \prod_{0 \le m \le k m \ne j} \frac{x - x_m}{x_j - x_m} \tag{3.22}$$

where ξ_i are the nodes associated with the cell. For a 1-dimensional grid, with nodes $\{0, 1, 2, 3, 4, 5\}$, two basis function ϕ_1, ϕ_2 are displayed in figure 3.3.

These functions are also used for the mapping of the particle information to the nodes. To map the values to the nodal points of the background grid,

$$f_i(t) = \sum_{p=1}^{n_p} \phi_i(\mathbf{X}) f_p(t), \tag{3.23}$$

and similarly, to map the information back to the particles

$$f_p(t) = \sum_{i=1}^{n_i} \phi_i(\mathbf{X}) f_i(t). \tag{3.24}$$

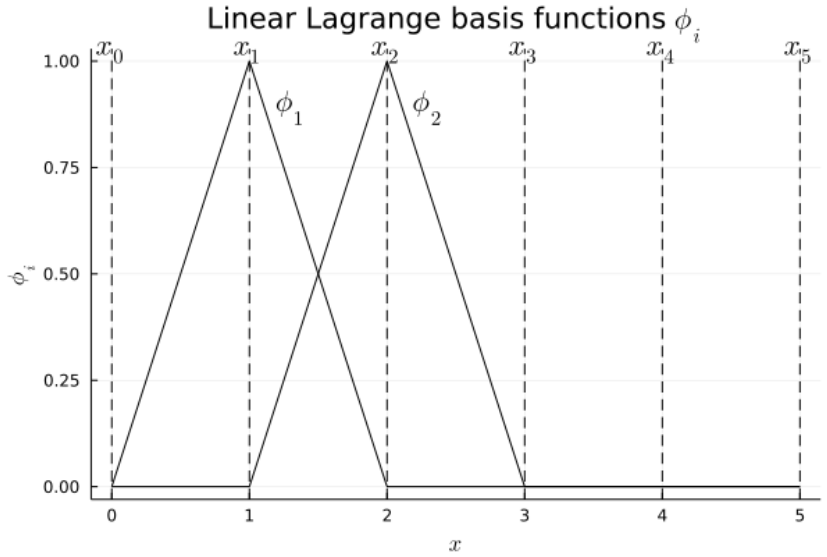


Figure 3.3: Example of two linear Lagrange basis function ϕ_1, ϕ_2 for a node set $\{0, 1, 2, 3, 4, 5\}$.

Note that the Lagrange basis function are differentiable, and the derivative at the background nodes can be computed using

$$\nabla f_i(t) = \sum_{p=1}^{n_p} \nabla \phi_i(\mathbf{X}) f_p(t), \tag{3.25}$$

3.5. The MPM Algorithm

There are a few variations of the traditional MPM of Sulsky et al.[31, 33], which all satisfy the conservation of mass and momentum. These variations are all different in the order of calculation the stress, thus the constitutive relation. Traditionally, the algorithm updates the stress after the nodal velocities are calculated, which Bardenhagen[3] referred to as the Update Stress Last. Other variations are the Update Stress First (USF) and the Modified Update Stress Last (MUSL). First, the USL algorithm is explained.

3.5.1. Preliminaries

Before the MPM algorithm is explained, a few more expressions must be determined. First, the particle volume V_p changes as the particle deforms. The relation between the particle volume and the deformation gradient F_p of the particle at a time t is given by

$$V_p^t = \det F_p^t V_p^0 \tag{3.26}$$

where V_p^0 is the initial volume. Since the conservation of mass is satisfied in the material point method, the relations

$$m_p^0 = \rho_p^0 V_p^0 = m_p^t = \rho_p^t V_p^t = V_p^0 \det \mathbf{F}_p^t = \frac{m_p}{V_p^t} \tag{3.27}$$

thus the density is given by

$$\rho_p^t = \frac{\rho_p^0}{\det \mathbf{F}_p^t}. \tag{3.28}$$

For the deformation tensor, we know that

$$\frac{\partial \mathbf{F}}{\partial t} = \frac{\partial}{\partial t}\left[\frac{\partial \mathbf{x}}{\partial \mathbf{X}}\right] = \frac{\partial \mathbf{v}}{\partial \mathbf{X}} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial \mathbf{X}} = \mathbf{L}\mathbf{F}$$

where the velocity gradient is defined as $L = \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$. Using the time integration) this equation becomes

$$\frac{\mathbf{F}_p^{t+\Delta t} - \mathbf{F}_p^t}{\Delta t} = L_p^{t+\Delta t}\mathbf{F}_p^t \Rightarrow \mathbf{F}_p^{t+\Delta t} = (\mathbf{I} + \Delta t \mathbf{L}_p^{t+\Delta t})\mathbf{F}_p^t \tag{3.29}$$

and the velocity gradient can be computed by projected the velocity of the nodes at the updated time $t + \Delta t$ as

$$\mathbf{L}_p^{t+\Delta t} = \nabla \mathbf{v}_p^{t+\Delta t} = \sum_{i=1}^{n_i} \nabla \phi_i(\mathbf{X}_p) \mathbf{v}_i^{t+\Delta t}. \tag{3.30}$$

Here, it is assumed that the stress is updated last, thus the velocity gradient at time $t+\Delta t$ is already computed beforehand.

3.5.2. Update Stress Last

At the start of a simulation, n_p particles are initialized, each with a position $\mathbf{x}_p$, velocity $\mathbf{v}_p$, stress $\boldsymbol{\sigma}_p$, volume V_p and density ρ_p. The background grid with n_i nodes is initialized with its shape function ϕ_i, with each particle inside the background grid. Originally, MPM uses *linear Lagrange polynomials* for ϕ_i, but this is not a requirement for MPM as we will see in section 4.

Algorithm 1: MPM scheme: Update Stress Last (USL)[31]

Result: MPM result at time t_{end}

1 Initialize particles and background grid.
2 Set $t = 0$;
3 **while** $t < t_{end}$ **do**
4 **Particles to Mesh**
5 Calculate the mass matrix elements m_{ij}^t for each node i
6 Calculate the force elements $\mathbf{f}_i^t$ for each node i
7 Calculate the linear momentum $\mathbf{p}_i^t = \sum_{p=1}^{n_p} \phi_I(\mathbf{x}_p^t)(m\mathbf{v})_p^t$ at each note i
8 Calculate the nodal velocity: $\mathbf{v}^t = (M^t)^{-1}\mathbf{p}^t$
9 Calculate the nodal acceleration: $\mathbf{a}^t = (M^t)^{-1}\mathbf{f}^t$
10 **Update** New nodal velocity: $\mathbf{v}_i^{t+\Delta t} = \mathbf{v}_i^t + \mathbf{a}_i^t \Delta t$
11 **Mesh to Particles**
12 Calculate the new particle velocity gradient: $\mathbf{L}_p^{t+\Delta t} = \sum_{I=1}^{n_n} \nabla \phi_i(\mathbf{X}_p^t) \mathbf{v}_i^{t+\Delta t}$
13 And the particle deformation: $\mathbf{F}_p^{t+\Delta t} = \left(\mathbf{I} + \Delta t \mathbf{L}_p^{t+\Delta t}\right) \mathbf{F}_p^t$
14 New particle velocity: $\mathbf{v}_p^{t+\Delta t} = \mathbf{v}_p^t + \sum_{I=1}^{n_n} \phi_I(\mathbf{x}_p^t)\mathbf{a}_I^t$
15 New particle position: $\mathbf{x}_p^{t+\Delta t} = \mathbf{x}_p^t + \sum_{I=1}^{n_n} \phi_I(\mathbf{x}_p^t)\mathbf{v}_I^{t+\Delta t}$
16 Update particle stress $\boldsymbol{\sigma}_p^{t+\Delta t}$ using constitutive equation 2.9 or 2.11
17 Update particle volume: $V_p^{t+\Delta t} = \det\left(\mathbf{F}_p^{t+\Delta t}\right) V_p^0$ and density $\rho_p^{t+\Delta t} = \left(\det \mathbf{F}_p^{t+\Delta t}\right)^{-1} \rho_p^0$
18 **Advance time** $t \longleftarrow t + \Delta t$
19 **end**

3.5.3. Modified Update Stress Last

Shortly after the introduction of the USL MPM method, a modified version was introduced[33], which was later dubbed the Modified Update Stress Last (MUSL) method. The major difference between the two methods is the way the incremental strain $\Delta\epsilon_p^{t+\Delta t}$ is calculated from the nodal velocities $\mathbf{v}_I^{t+\Delta t}$, line 12 of Algorithm 1

In the USL method, the next time step velocity was calculated directly from the acceleration. However, in the modified version, there is a remapping of the velocity. First, the new particle velocities are calculated, and using the new particle velocity, the nodal velocity is recalculated as

$$(m\mathbf{v})_i^{t+\Delta t} = \sum_{p=1}^{n_p} \phi_i(\mathbf{x}_p^t)(m\mathbf{v})_p^{t+\Delta t} \tag{3.31}$$

3.5.4. Update Stress First

There is a possibility to change to order of calculating in Algorithm 1, which is called the Update Stress First (USF) method[3]. As the name implies, in this variation the stresses are updated at the start of

each time step, instead of the end. The force at the nodes is then calculated using $\boldsymbol{\sigma}_p^{t+\Delta t}$ instead. Note that this procedure is mathematically speaking fairly similar to the MUSL, as the MUSL updates the stress at the end of a time step, while the USF updates the stress at the beginning of the next time step. The only difference is the use of the shape functions ϕ_i, where MUSL uses the particle positions of the current time step ($\phi_i(\mathbf{x}_p^t)$) and USF the particle positions of the next time step, thus $\phi_i(\mathbf{x}_p^{t+\Delta t})$

As both methods use the same formulation, they both conserve mass en momentum. However, the difference is present in the conservation of energy[3].

3.6. Problems in MPM

Although the material point method uses the power of both mesh-based and mesh-free methods, the MPM has its own shortcomings.

3.6.1. Grid-crossing error

In the standard MPM algorithm, linear Lagrange basis function ϕ_I or tent functions are used, e.g. figure 3.3. For the internal force calculation (equation 3.13), $\nabla\phi_I$ is used which is discontinuous at the cell boundaries as displayed in figure 3.4. If particles in a cell cross these boundaries to another cell, there is a gradient jump resulting in unexpected behavior of the internal forces and stresses, which will be illustrated in the example below. This phenomenon is referred to as *grid-crossing* or the cell-crossing error.

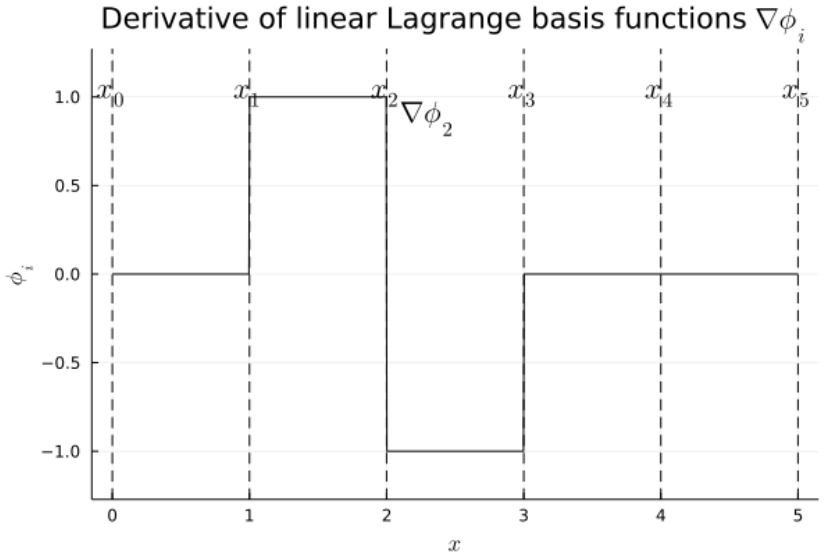


Figure 3.4: Example of the derivative of a linear Lagrange basis function ϕ_2 for a node set $\{0, 1, 2, 3, 4, 5\}$.

3.6.2. Example: 1D vibrating bar

To illustrate the grid-crossing error, a 1-dimensional vibrating bar of length L is considered[3, 34, 22]. This bar is fixed at both ends and a velocity v_0 is prescribed at $t = 0$.

Ignoring any friction or damping forces, the partial differential equations for the 1-dimensional vibrating bar is

$$\frac{\partial^2 u}{\partial t^2} = \frac{E}{\rho}\frac{\partial^2 u}{\partial x^2}, \tag{3.32}$$

which is subject to the following initial and boundary conditions

$$\begin{aligned} u(0,t) &= u(L,t) = 0 \\ u(x,0) &= 0 \\ \frac{\partial u}{\partial t}(x,0) &= v_0 \sin\left(\frac{\pi x}{L}\right). \end{aligned}$$

This equation can be solved analytically using separation of variables and combined with the boundary conditions and initial condition, the solution becomes

$$u(x,t) = \frac{v_0}{\omega}\sin(\omega t)\sin\left(\frac{\pi x}{L}\right) \text{ with } \omega = \frac{\pi\sqrt{\frac{E}{\rho}}}{L}. \tag{3.33}$$

Since there is an analytic solution for this equation, the numerical results from the MPM calculation will be compared to this solution.

Using these equations, the problem can be solved using MPM with B-splines, see section 4 and the linear constitutive model for small deformations is used, equation 2.11. For this, the following parameters are used

$$\begin{aligned}\rho &= 1\text{kg}\cdot\text{m}^{-3}\\ E &= 50\text{Pa}\\ L &= 1\text{m}\\ \Delta t &= 10^{-5}s\\ v_0 &= 0.1m\cdot s^{-1}\end{aligned}$$

Note that this indeed holds the CFL conditions

$$\Delta t \leq \sqrt{\frac{\rho}{E}}\Delta x. \tag{3.34}$$

To illustrate the effect of the grid crossing error, specific values are chosen for the space discretization. The domain $[0, L]$ is divided into 25 and 50 equally sized cells. The same domain $[0, L]$ is populated with uniformly distributed particles such that each cell initially has 4 particles inside it.

The result of the Cauchy stress σ at $t = 0.5s$ using 25 cells is displayed in figure 3.5 using B-splines of degree 1 and 2. Note that linear B-splines and linear Lagrange basis function are identical, see chapter 4 for more detail.

Although both figures calculate the stress accurately, the stress using linear basis functions have distinct steps. This is a direct result of the use of shape functions ϕ_i. Since linear basis function are used in figure 3.5a, its derivative $\nabla\phi_i$ used in the stress calculations is constant but discontinuous at the nodes. This effect disappears when smoother (C^1 or higher) basis function. This thesis primarily uses quadratic B-splines which has continuous derivatives, as is displayed in figure 3.5b.

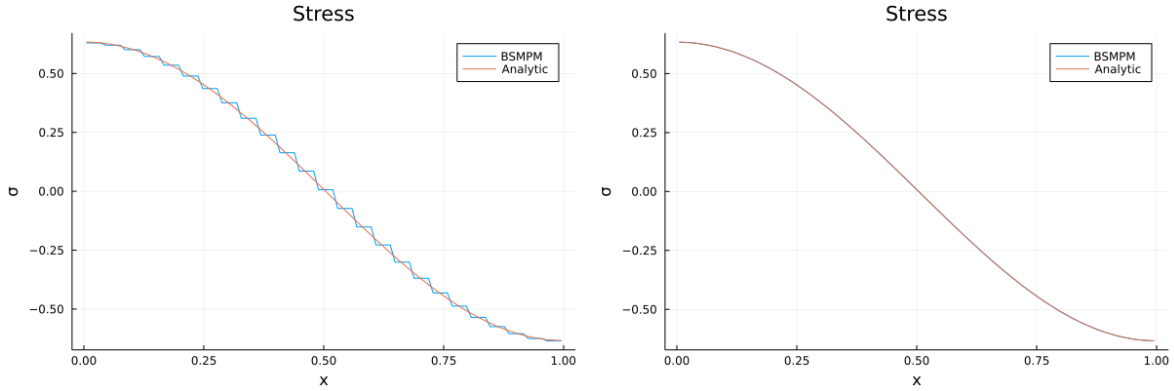
The discontinuous derivatives of the B-splines compute the stress accurately if the particles do not move to other cells. Since the derivatives are constant, a particle moving from one cell to its neighbor causes the stress to jump to another value. This is displayed in figure 3.6 where the cells are smaller and the vibrating particles move between cells. This is mitigated by using quadratic B-splines as displayed in figure 3.6b where the stress is still calculated accurately.

3.6.3. Nearly-empty cells

Another problem with the MPM is the so called nearly-empty cell problem. Since material points can move freely through space and are independent of the background grid, it is possible that cells (generally near the boundary) are empty.

If a cell is empty, there will be a node $\mathbf{x}_i$ without mass. If node $\mathbf{x}_i$ is included in the calculations, the mass matrix becomes singular, and acceleration cannot be computed in equation 3.8. However, since there are no material points inside the cell, the cell does not contribute to the solution and the node can be ignored.

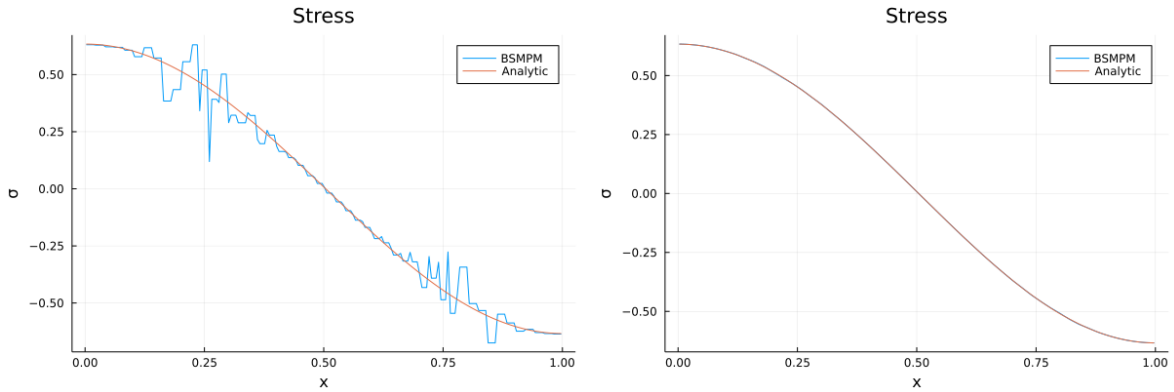
Since particles can move freely through the spatial domain and the background grid can be chosen arbitrarily, it is possible for a cell to be (or become) *nearly-empty*, meaning that cell has significantly less particles inside than the other cells. This causes the mass at specific nodes to be significantly smaller and the matrix M to be ill-conditioned. An example of such a matrix where a cell has only one



(a) 25 cells, initially with 4 particles per cell, $p = 1$

(b) 25 cells, initially with 4 particles per cell, $p = 2$

Figure 3.5: Stress at $t = 0.5s$ using $\Delta t = 10^{-5}$ using B-splines of degree 1 and 2. The bar is split into 50 equal sized cells.



(a) 50 cells, initially with 4 particles per cell, $p = 1$

(b) 50 cells, initially with 4 particles per cell, $p = 2$

Figure 3.6: Stress along the vibrating bar at $t = 0.5s$ using $\Delta t = 10^{-5}$ using B-splines of degree 1 and 2. The bar is split into 50 equal sized cells.

particle on the edge is

$$M = \begin{bmatrix} \ddots & \ddots & \ddots & & \emptyset \\ \ddots & 0.35 & 0.14 & 0.01 & \\ \ddots & 0.14 & 0.33 & 0.09 & 6.5 \cdot 10^{-12} \\ & 0.01 & 0.09 & 0.05 & 6.5 \cdot 10^{-12} \\ \emptyset & & 6.5 \cdot 10^{-12} & 6.5 \cdot 10^{-12} & 1.1 \cdot 10^{-21} \end{bmatrix}, \tag{3.35}$$

which causes significant errors in the acceleration, eq. 3.8, and stresses.

A commonly used method to resolve this issue is the use of a cut-off technique[10], where nodes are ignored if the mass entry is smaller than a chosen tolerance, i.e.

$$\mathbf{v}_I^{t+\Delta t} = \begin{cases} \frac{(m\mathbf{v})_I^{t+\Delta t}}{m_I^t} & \text{if } m_I^t > tol \\ 0 & \text{else} \end{cases} \tag{3.36}$$

However, the tolerance is experimentally chosen and is highly dependent on the problem, grid and particles.

3.6.4. Example: Dynamic traction at the boundary

To illustrate the effect of nearly-empty cells, another benchmark is used. In this benchmark, a horizontal bar of length L is fixed at one end, and a forcing function is applied on the right side of the bar[2, 30, 37]. A sine forcing function is chosen as

$$q(x,t) = \delta(x-L)H(t)\tau \sin\left(\frac{x}{t}\right). \tag{3.37}$$

This results in the following stress profile[30]

$$\sigma(x,t) = \begin{cases} 0 & \text{if } x \in [0, L-x) \\ -\tau \sin(\omega(t+x)) & \text{if } t \in [L-x, L+x) \\ -\tau(\sin(\omega(t+x)) + \sin(\omega(t-x))) & \text{if } t \in [L+x, 3L-x) \\ -\tau \sin(\omega(t-x)) & \text{if } t \in [3L-x, 3L+x) \\ 0 & \text{if } t \in [3L+x, 4L) \end{cases} \tag{3.38}$$

where $\omega = \frac{\pi}{L}$. The traction boundary condition is then expressed as $\sigma_t(x,t) = \delta(x-L)\sigma(x,t)$. The displacement is given[30] by

$$u(x,t) = \begin{cases} 0 & \text{if } x \in [0, L-x) \\ \alpha(1+\cos(\omega(t+x)) & \text{if } t \in [L-x, L+x) \\ \alpha(\cos(\omega(t+x)) - \sin(\omega(t-x))) & \text{if } t \in [L+x, 3L-x) \\ \alpha(-1-\cos(\omega(t-x))) & \text{if } t \in [3L-x, 3L+x) \\ 0 & \text{if } t \in [3L+x, 4L) \end{cases} \tag{3.39}$$

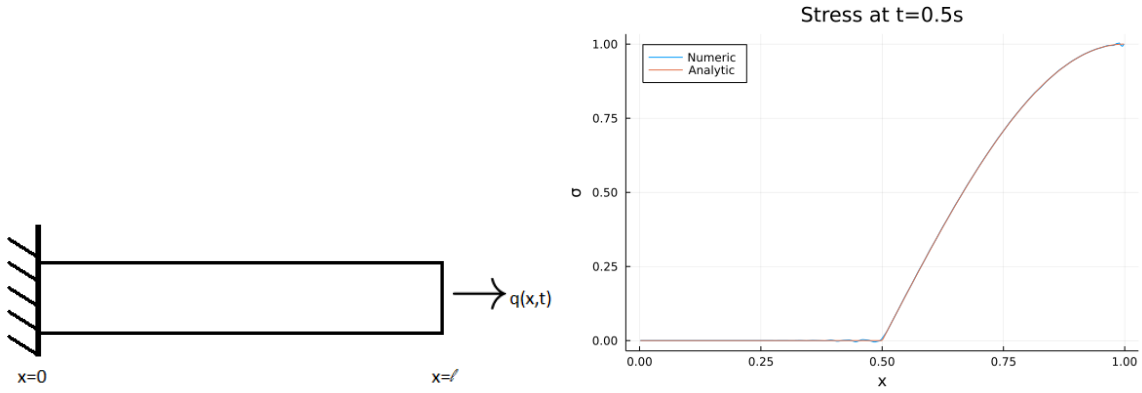
A graphical interpretation of this problem is displayed in figure 3.7.

Since the force $q(x,t)$ applied on the right-hand side of the bar causes the bar to expand (or contract) and the background grid remains fixed, it is not directly known where the traction boundary condition is applied. However, from the analytic solution, we know that the length of the bar at a given time is expressed as

$$L(t) = L + u(L,t), \tag{3.40}$$

and the traction boundary condition can use equation 3.16, resulting in

$$f_i^{\text{tract}}(t) = \int_\Gamma \phi_i \mathbf{n} \cdot \boldsymbol{\sigma} \mathrm{d}\Gamma = \phi_i(L + u(L,t))\sigma(L,t), \tag{3.41}$$



(a) Horizontal bar of length L with fixed left end and a forcing function $q(x,t)$ applied on the right-hand side.

(b) Solution of the stress at $t = 0.5s$, using quadratic B-splines, 80 cells, and 4 particles per cell.

Figure 3.7: Illustration of the dynamic traction boundary condition (left) and the stress of this benchmark at $t = 0.5s$.

which implies that the initial background grid must be chosen sufficiently large. Therefore, we calculate that the maximum extension of the bar occurs at $t = 1s$, and the background grid domain is chosen to be $[0, L + u(L, 1)]$. However, at the start of the simulation, particles only populate the domain $[0, L]$.

Using the equation for the external force and construction of the domain, the solution can be computing, which is displayed in figure 3.7b for quadratic B-splines. The domain is divided in $n_i = 80$ cells, with approximately 4 particles per cell, meaning that there are $n_p = 4 \cdot n_i$ particles, distributed uniformly withing $[0, L]$.

For this configuration, the most right cell still has particles inside it, and the impact of the nearly-empty cell is less noticeable. However, if $n_i = 160$ is chosen, the right cell is empty at the start of the simulation and will eventually be filled when the bar expands. For the quadratic B-splines, this happens between time $t = 0.2356s$ and $t = 0.2358s$. The Cauchy stress σ and displacement u at these time steps are displayed in figure 3.8. This problem is run using both the USL and MUSL algorithm, with comparable results.

The proposed method to reduce this effect is the use of EB-splines, see section 4.5. To illustrate the effect EB-splines have on the solution, the exact same simulation is done using B-splines. However, at time $t = 0.2356s$, the calculations are executed using EB-splines. The result after this time step is displayed in figure 3.9, which indeed resolves the empty cell problem.

Comparable results can be obtained if the problematic right particle is removed or moved slightly left, making the right cell empty again. In that case, regular B-splines compute the solution accurately, illustrating the effect of nearly-empty cells.

3.7. MPM extensions

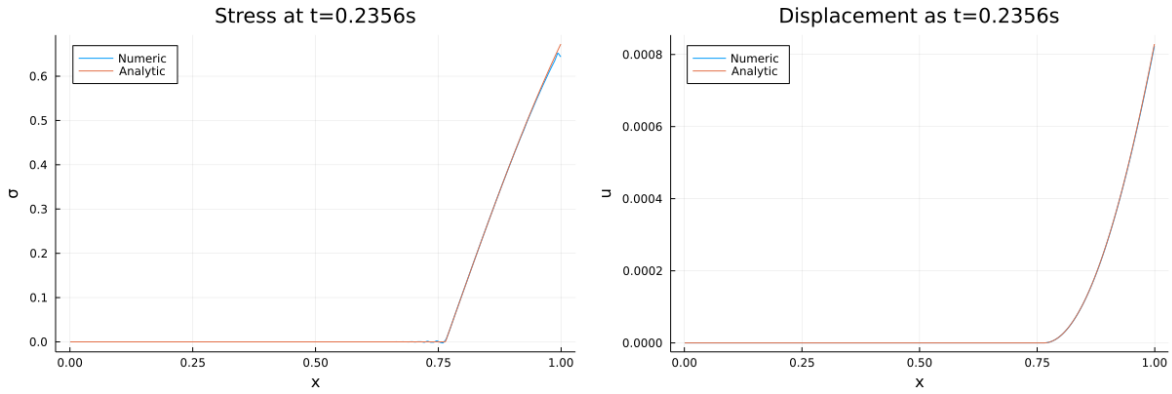
Various methods have been developed to overcome these problems. A couple of the most used extensions and improvements are listed below.

3.7.1. GIMP and CPDI

The generalized interpolation material point (GIMP)[2] and the Convected Particle Domain Interpolation (CDPI)[28] are methods introduced to resolve the grid crossing error. Instead of using particles only, each particle has a particle domain which is tracked throughout the simulation. In GIMP, there are still gaps between particle domains resulting in inaccuracies, which CDPI resolves by connecting the particle domains. However, this removes part of the power of MPM being that it must calculate the mesh.

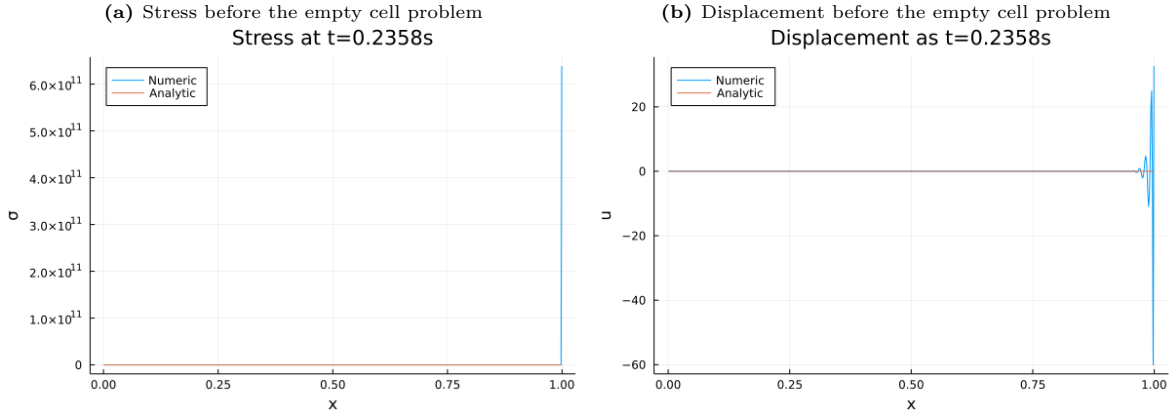
3.7.2. Dual-Domain MPM

This method, introduced in [39], introduces a modified gradient technique to resolve the grid crossing error as well as creating a MPM variation for unstructured grids. In this method, the particle properties are first mapped to the nodal points and afterwards interpolated creating a continuous stress field.



(a) Stress before the empty cell problem

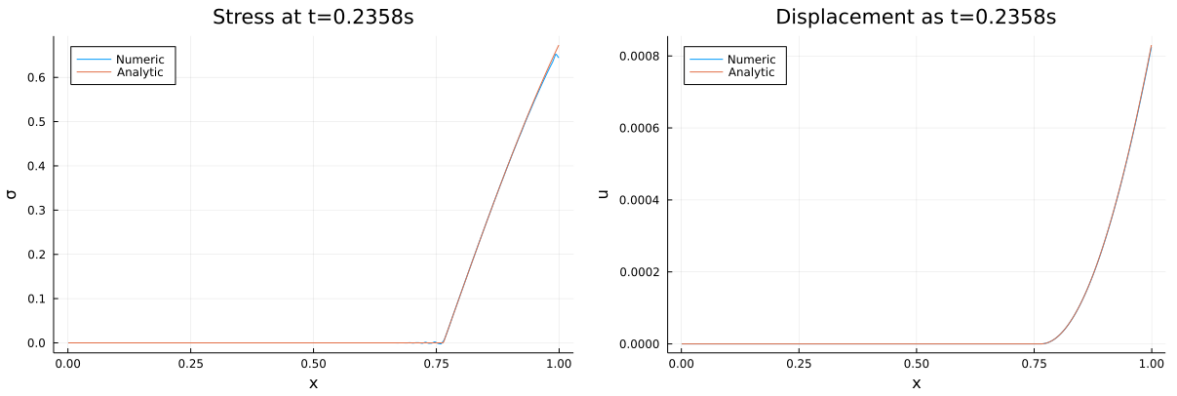
(b) Displacement before the empty cell problem



(c) Stress after the empty cell problem

(d) Displacement after the empty cell problem

Figure 3.8: Example of the impact of the nearly empty cell problem on both the stress and the displacement. Quadratic B-splines are uses, with $n_i = 160$.



(a) Stress after using EB-splines

(b) Displacement after using EB-splines

Figure 3.9: Until $t = 0.2356s$, the exact same B-splines were used with similar results as in figure 3.8a and 3.8c. However, since the next time step suffers from the empty cell problem, only this step is computed using the EB-splines instead of B-splines, resolving the problem.

3.7.3. Quadrature points

Particles are used for the quadrature points during the integration step. Particles can however position arbitrarily, and are therefore not necessarily position at optimal locations, making convergence difficult and calculations inaccurate.

The improved MPM method[32] resolves this problem by using moving least squares and a similar method using Taylor is also introduces in [38].

3.7.4. BSMPM

Another method is the B-spline MPM, which uses B-splines as shape functions. Where the first order B-splines have similar shapes as the linear Lagrange basis function, including the same grid crossing error, higher order B-splines resolves this problem. The use of B-splines in MPM will be explained in more detail.

4

Isogeometric Analysis

The use of B-splines or basis splines are not a new concept, but is widely used in approximations in computational mathematics, especially in the context of Computer Aided Design (CAD).

In many mathematical benchmarks, the geometries of these domains are simple, such as a circle or a rectangle in 2 dimensions, or a sphere or cube in 3 dimensions. In engineering applications however, these geometries become more complex and are often created using Computer Aided Design (CAD). The CAD model describes the geometry of a given problem by various methods. One of the tools used to describe these geometries is by a linear combination of B-splines and a set of control points.

It is often required to solve a partial differential equation on this geometry. While this is sometimes possible to solve analytically, this is generally not possible, and a numerical method is required. Finite Element Method is a powerful numerical method for solving these Partial Differential Equations[35] on a specified domain.

Usually, a CAD geometry is not readily available to use in FEM, but a mesh must be approximated from this geometry. This is both time consuming and generates new challenges. These mesh approximations can lead to undesirable inaccuracies.

Isogeometric Analysis (IGA)[19] tries to overcome this mesh approximation, by adapting to techniques used in CAD directly into the numerical method. One of these tools are the Non-Uniform Rational B-Splines (NURBS), which are built from B-Splines. In the next section, a similar approach is considered, where B-splines are used as test function in the Material Point Method.

4.1. B-splines

B-splines or basis splines are piecewise polynomials which are defined recursively by a knot vector Ξ. Any spline of order p on a given set of *knot* can be expressed as a linear combination of B-splines of order p. The B-splines are then the basis of the spline function space V, hence the name. Before defining the B-splines, a few terms such as knot vectors, must be explained.

4.1.1. Terminology

Knot vector

Let $\Omega \subseteq \mathbb{R}$ be the considered 1 dimensional domain. Let Ξ be a set of $n + p + 1$ non-decreasing number, $\xi_0 \leq \xi_1 \leq \cdots \leq \xi_{n+p+1}, \xi_i \in \Omega$, where n is the number of B-splines and p the polynomial order. The numbers ξ_i are called the *knots* of the *knot vector* $\Xi = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$ and divide the parametric domain $[\xi_0, \xi_{n+p+1}]$ in sub-domains $[\xi_i, \xi_{i+1})$ called *knot spans*, which do not necessarily exists if ξ_i appear multiple times.

Knot multiplicity

Unlike the Lagrange polynomials, a knot ξ can appear multiple times in the same knot vector. If a knot appears k times, i.e., $\xi_i = \xi_{i+1} = \cdots = \xi_{i+k-1}$, then ξ_i has a multiplicity of k. In fact, the shape of B-splines can be modified by changing the multiplicity of knots, see knot insertion.

Open knot vector

A knot vector is called *open* if the first and last knot appear at least $p+1$ times. In this thesis, only open knot vectors are used.

Uniform knot vector

A knot vector is called uniform if $\xi_{i+1} - \xi_i = \text{const}, \forall \xi \in \Xi$. An open uniform knot vector is an open knot vector, where the interior knots are equidistant, i.e.,

$$\begin{cases} \xi_i = \xi_{i+1} & \text{if } 0 \leq i \leq p-1 \\ \xi_{i+1} - \xi_i = \text{const} & \text{if } p \leq i \leq n \\ \xi_i = \xi_{i+1} & \text{if } n+1 \leq i \leq n+p \end{cases}$$

4.1.2. Definition

A B-spline $B_{i,p}(\xi)$ is defined recursively using the *Cox-de Boor* recursion formula[7]. Here, the indices indicate the ith B-spline of polynomial degree p and let Ξ be a knot vector of size $n+p+1$

For $p = 0$, we use

$$B_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{other} \end{cases}. \tag{4.1}$$

and using this formulation, we get the basis functions for $p > 0$ using the Cox-de Boor recursion formula

$$B_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} B_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} B_{i+1,p-1}(\xi). \tag{4.2}$$

Using this recursive formula, the parametric derivative can be derived

$$\frac{\mathrm{d}B_{i,p}}{\mathrm{d}\xi}(\xi) = \frac{p}{\xi_{i+p} - \xi_i} B_{i,p-1}(\xi) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} B_{i+1,p-1}(\xi). \tag{4.3}$$

Higher order derivatives can be derived in a similar fashion, but this does not mean that $B_{i,p}(\xi)$ is $p-1$ *continuously differentiable*, but only $p-k$ times, where k is the multiplicity of the knot (except when $p = 0$).

Non-negative partition of unity

The basis functions have some important properties. Regardless of the polynomial order and the knot vector Ξ, we know that B-splines satisfy the *partition of unity*:

$$\sum_{i=1}^{n} B_{i,p}(\xi) = 1, \forall \xi \tag{4.4}$$

Moreover, the basis functions $B_{i,p}$ are non-negative. When B-splines are used as the shape functions ϕ_I in MPM, this guaranties that mass element M_{IJ} in eq. 3.11 using the basis function will not be negative and satisfy the conservation of mass.

Support

The support of a function $f : X \to Y$ is defined as sub-domain where the function is non-negative, thus

$$\text{supp}(f) = \{x \in X : f(x) \neq 0\} \subseteq X. \tag{4.5}$$

For each B-spline $B_{i,p}(\xi)$, this support can be expressed in terms of the knots spans $[\xi_i, \xi_{i+1})$, namely

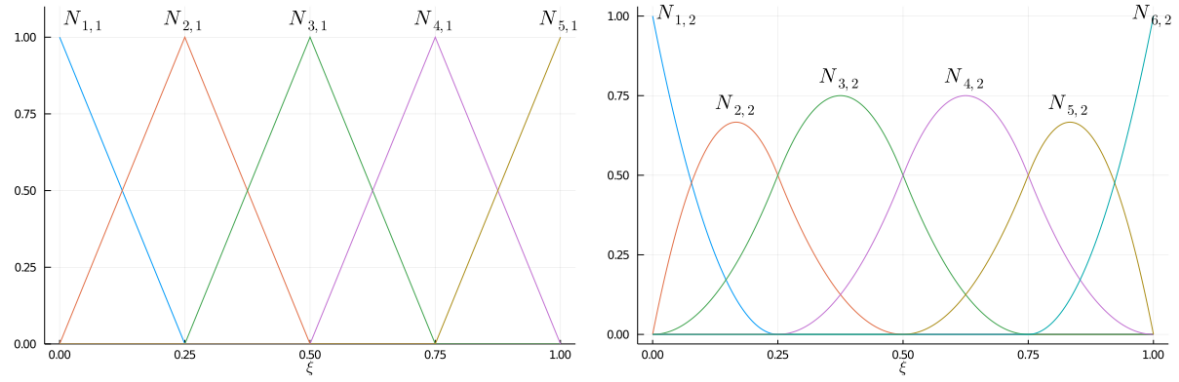
$$\text{supp}(B_{i,p}(\xi)) = [\xi_i, \xi_{i+p+1}) \subseteq \Omega. \tag{4.6}$$

Example

An example of these B-spline basis functions is given in figure 4.1. Note that the linear B-spline basis function ($p = 1$) coincide with the linear Lagrange basis function, typically used in FEM.

In combination with a set of control points, these basis functions generate B-spline curves, defined as

$$C(\xi) = \sum_{i=1}^{n} B_{i,p}(\xi) C_i, \tag{4.7}$$



(a) $\Xi = [0, 0, 0.25, 0.5, 0.75, 1, 1], p = 1$ **(b)** $\Xi = [0, 0, 0.25, 0.5, 0.75, 1, 1], p = 2$

Figure 4.1: B-spline basis functions with a polynomial order $p = 1$ and $p = 2$, generated by eq. 4.2

where $B_{i,p}(\xi)$ are the B-spline basis functions defined on the parametric domain ξ and C_i the control points. Generating a B-spline surface is possible by tensor products e.g.

$$S(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} B_{i,p}(\xi) M_{j,q}(\nu) C_{i,j}. \tag{4.8}$$

4.2. Refinements

Refinements of the B-splines are not performed in the traditional mesh refinements sense, but by refining the B-spline functions. Generally, there are two techniques, in the form of h-refinements (knot insertions) and p-refinements (order elevation).

The latter is displayed in figure 4.1, where splines of order (polynomial degree) 1 and 2 are displayed. In general, the B-splines have $p - 1$ continuous derivatives in the absence of repeated knots, which is the reason the grid-crossing error is reduced in figure 3.5b,3.6b.

Note that this also increases the *support* of the B-splines function, meaning a B-spline becomes less "local". Furthermore, this also increases the number of non-negative B-splines in the ith-knot span by $B_{i-p,p}, \ldots, B_{i,p}$.

Knot insertion manipulates the set of B-splines by inserting additional elements inside the knot vector Ξ (at the correct index such that the vector remains non-decreasing). The procedure of generating B-splines is identical, except that there are now $n + 1$ B-splines. However, if these B-splines are used for interpolation, the control points $\{C_1, C_2, \ldots, C_n\}$ must be updated. If $\bar{\xi} \in [\xi_k, \xi_{k+1})$ is inserted, the new set of control points is created by[19]

$$\bar{C}_i = (1 - \alpha) C_{i-1} + \alpha_i C_i \tag{4.9}$$

$$\alpha_i = \begin{cases} 1 & i \leq k - p \\ \frac{\bar{\xi} - \xi_i}{\xi_{i+p} - \xi_i} & k - p + 1 \leq i \leq k \\ 0 & k + 1 \leq i \leq n + p + 2 \end{cases}. \tag{4.10}$$

or in matrix notation[23]

$$\bar{C}_i = \sum_{j=0}^{n} S_{i,j} C_j \tag{4.11}$$

$$S_{i,i-1} = 1 - \alpha_i \tag{4.12}$$

$$S_{i,i} = \alpha_i, \tag{4.13}$$

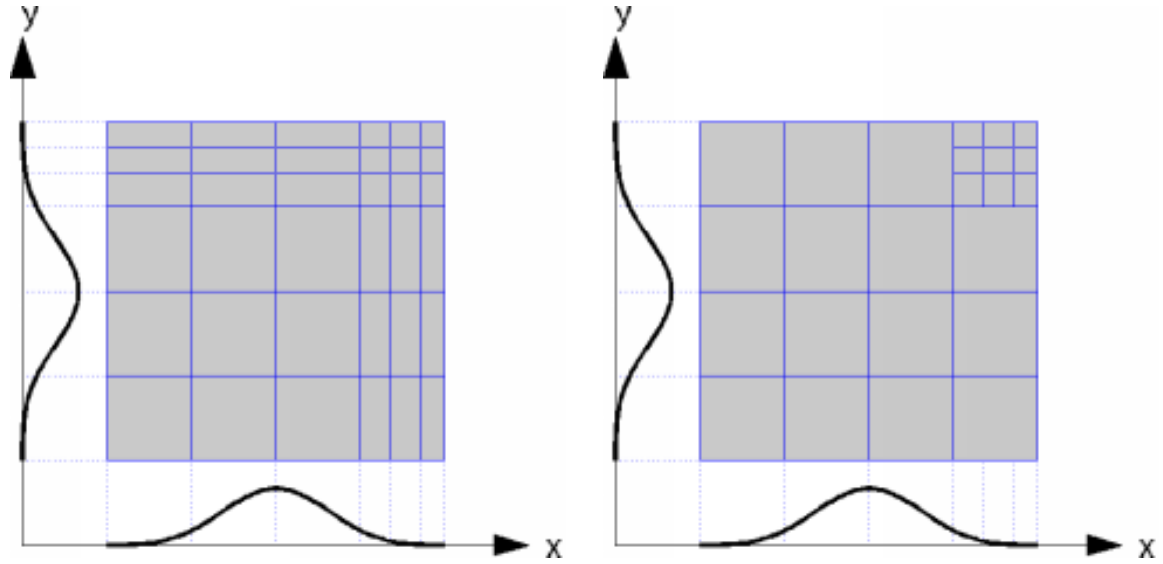
where $S_{i,j}$ are the matrix elements of the so-called *subdivision matrix* S. If multiple knots $\{\xi_1, \xi_2, \ldots, \xi_n\}$ are inserted, the combined matrix S can be calculated by multiplying the subdivision matrix of each knot;

$$S = S^n S^{n-1} \ldots S^2 S^1. \tag{4.14}$$

These subdivision matrices are not directly required for the generation of the B-splines, but they are used in the construction of truncated hierarchical B-splines (THB-splines).

4.3. Hierarchical B-splines

While the above describe knot insertion technique can be considered a ”local” refinement technique for univariate B-splines (1 dimensional), this is not the case when using the tensor products to describe surfaces or volumes, as illustrated in figure 4.2a for 2 dimensions. In the case of large domains with various refinements, this can become computationally intensive. Therefore, a local refinement technique is desirable, as illustrated in figure 4.2b.



(a) Knot insertion B-spline (not local) **(b)** Locally refined B-spline

Figure 4.2: Illustrating of bivariate knot-insertion (non-local) and the motivation for a desirable local refinement technique.

Since local refinements are desirable, various techniques are developed in the context of Isogeometric Analysis, such as T-splines[11], LR B-spline[20] and hierarchical B-splines[36]. In this thesis, we will focus on the use of hierarchical B-splines (HB-splines) and its extensions, the truncated hierarchical B-splines (THB-splines).

The concept of hierarchical B-splines was first introduced by Forsey[12, 13], where finer local patches of the parent surface are added to create the refinements. This principle is used to create an adaptive local refinement technique in IGA[36].

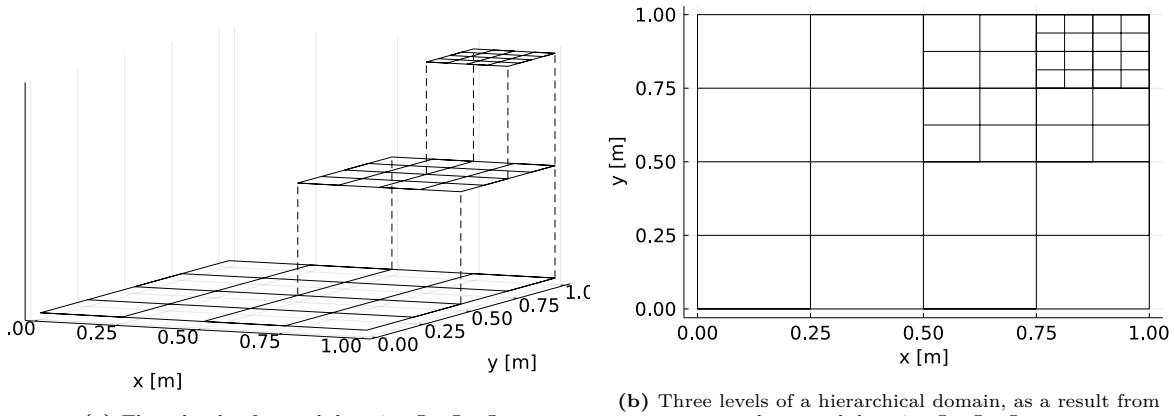
The HB-splines use a *nested* set of domains where Ω^0 is the original domain or ground level and $l \in \mathbb{N}$ the number of nested domains. Let there be l n-variate B-spline spaces $\{\mathcal{N}^i\}_{i=0,1,\dots,l-1}$ which are nested, thus

$$\mathcal{N}^0 \subset \mathcal{N}^1 \subset \cdots \subset \mathcal{N}^{l-1}, \tag{4.15}$$

and a sequence of l n-dimensional bounded sets $\{\Omega^i\}_{i=0,1,\dots,l-1}$ which are nested too,

$$\Omega^{l-1} \subseteq \Omega^{l-2} \subseteq \cdots \subseteq \Omega^0 \tag{4.16}$$

and let $\Omega^l = \emptyset$. The boundary $\partial\Omega^i$ lies on the knot line of $\mathcal{N}^i$ or $\mathcal{N}^{i+1}$. An example of this nested domain structure in 2 dimensions is displayed in figure 4.3a



(a) Three levels of nested domains $\Omega_0, \Omega_1, \Omega_2$

(b) Three levels of a hierarchical domain, as a result from the nested domains $\Omega_0, \Omega_1, \Omega_2$

Figure 4.3: Nested and hierarchical domains example

Note that the spline spaces are nested, $\mathcal{N}^i \subset \mathcal{N}^{i+1}$, thus the polynomial degree p_k of the B-spline in each dimension k must hold $p_k^i \leq p_k^{i+1}$.

The definition for the support, eq. 4.5, of a B-spline B_i^l at level l is slightly altered to

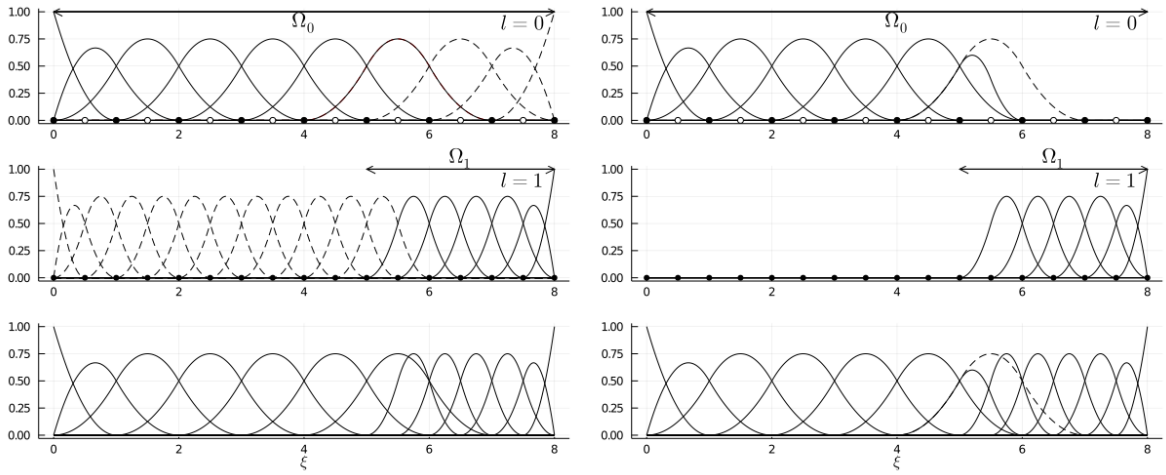
$$\text{supp}(B_i^l) = \{x \in \Omega^0 : f(x) \neq 0\}. \tag{4.17}$$

The hierarchical B-spline basis $\mathcal{H} = \mathcal{H}^{N-1}$ is defined recursively[36, 14] by

$$\begin{aligned}
\mathcal{H}^0 &= \{f \in \mathcal{N}^0 : \text{supp}(f) \neq \emptyset\} \\
\mathcal{H}^l &= \mathcal{H}_A^l \cup \mathcal{H}_B^l \text{ for } l = 1, \ldots, N-1 \\
\mathcal{H}_A^l &= \{f \in \mathcal{H}^{l-1} : \text{supp}(f) \not\subseteq \Omega^l\} \\
\mathcal{H}_B^l &= \{f \in \mathcal{N}^l : \text{supp} \subseteq \Omega^l\}
\end{aligned}$$

This recursive process *activates* B-splines of level l inside a domain Ω_l ($\mathcal{H}_B^l$) and *deactivate* B-splines of the previous level $l-1$ which support lies inside Ω_l ($\mathcal{H}_A^l$).

An example of this process for an univariable HB-spline is shown in figure 4.4a, where $\Omega_0 = [0, 8]$, $\Omega_1 = [5, 8]$ and $\Xi = \{0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 8, 8\}$ with $p = 2$.



(a) B-splines at level $l = 0$ (above) and $l = 1$ (middle), where the black dots indicate the knots. The solid and dashed splines indicate the active and deactivated B-splines. The resulting hierarchical B-splines are shown in the bottom figure.

(b) Activate truncated B-splines at level $l = 0$ (above) and $l = 1$ (middle), where the black dots indicate the knots. The dashed line indicates the spline before truncation. The resulting hierarchical splines are shown in the bottom figure.

Figure 4.4: Construction of hierarchical B-splines (a) and truncated hierarchical B-splines (b) using $\Omega_0 = [0, 8]$, $\Omega_1 = [5, 8]$ and $\Xi = \{0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 8, 8\}$

4.4. THB-splines

One property of the B-splines is the positive partition of unity;

$$\begin{cases} B_{i,p}(\xi) \geq 0 \\ \sum_{i=0}^{n} B_{i,p}(\xi) = 1, \forall \xi \in [\xi_1, \xi_{n+p+1}] \end{cases}.$$

However, this property is lost in the case of hierarchical B-splines. Since B-splines of various levels are combined, the sum over them is not necessarily one. This is also illustrated in the bottom plot of figure 4.4a, where $\sum_{i=0}^{n} B_{i,p}(\xi) > 1$ when $\xi \in (6, 7)$.

Although, it is stated in [36] that the partition of unity can be acquired by using weights (scaling), a truncated basis of the HB-splines is introduced[14] which guaranteed this property. Furthermore, the linear independence of (scaled) hierarchical B-splines is not guaranteed but can be acquired by identifying and removing overlapping linear dependent HB-splines[23]. The truncated hierarchical B-splines (THB-splines) are linear independent. Another improvement of the THB-splines is the decrease of the overlapping B-splines of distinct levels, increasing the stability properties.

4.4.1. THB-splines construction

The construction of the truncated bases for the hierarchical B-splines makes use of the fact that lower-level B-splines are linear combinations of the higher levels, thus

$$\tau = \sum_{f \in \mathcal{N}^{l+1}} c_f^{l+1}(\tau) f. \tag{4.18}$$

The truncation of τ with respect to $\mathcal{N}^{l+1}$ and Ω^{l+1} is then defined as[14]

$$\text{trunc}^{l+1}\tau = \sum_{f \in \mathcal{N}^{l+1}, \text{supp} f \not\subseteq \Omega^{l+1}} c_f^{l+1}(\tau) f \tag{4.19}$$

and $\mathcal{H}_A^l$ in the hierarchical B-spline procedure is updated by only adding the truncation $\text{trunc}^l f, f \in \mathcal{H}$ instead of f. The truncated version of the recursive HB-spline is then defined as

$$\begin{aligned} \mathcal{T}^0 &= \{f \in \mathcal{N}^0 : \text{supp}(f) \neq \emptyset\} \\ \mathcal{T}^l &= \mathcal{T}_A^l \cup \mathcal{T}_B^l \text{ for } l = 1, \ldots, N-1 \\ \mathcal{T}_A^l &= \{\text{trunc}^l \tau : \tau \in \mathcal{T}^{l-1} \wedge \text{supp}(\tau) \not\subseteq \Omega^l\} \\ \mathcal{T}_B^l &= \{f \in \mathcal{N}^l : \text{supp}(f) \subseteq \Omega^l\} \end{aligned}$$

An example of this process for an univariable THB-spline is shown in figure 4.4a, where $\Omega_0 = [0, 8]$, $\Omega_1 = [5, 8]$ and $\Xi = \{0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 8, 8\}$ with $p = 2$.

4.4.2. Computing the truncation coefficients

Since the B-spline function spaces are nested, it holds that for the knot vectors of level i hold $\Xi^i \subset \Xi^{i+1}$. The knot vector Ξ^{i+1} of level $i+1$ can be created by inserting multiple knots into knot vector Ξ^i. The elements resulting subdivision matrix S can then be used in eq.4.18. Let η_D be the set of deactivated B-splines of level $l+1$, then the truncation of a B-spline B_i can be written as

$$\text{trunc}^{l+1} B_i^l = \sum_{B_j^{l+1} \in \eta_D} S_{ij} B_j^{l+1}. \tag{4.20}$$

4.5. Univariate Extended B-splines

Up until this point, only B-splines with a *stable* basis have been considered. However, if a (part of the) domain is *trimmed*, some of the B-spline supports can lie outside the trimmed domain $A^v \subset \Omega$. This in turn can mean the support inside the trimmed domain can be small, significantly impacting the

stability (as is the case in the nearly-empty cell problem). Hollig et al. [17, 16] specified the formulation of a n-dimensional EB-spline procedure, but in this section only a univariate (1-dimensional) EB-spline is described. In the next section, is procedure is extended to a bivariate EB-spline.

To classify the splines, the *Greville abscissae* of a B-spline i is used, defined by the associated knot vector as

$$\bar{\xi}_i = \frac{1}{p}\sum_{j=i}^{i+p} \xi_j, \tag{4.21}$$

and using this definition, three types of B-splines can exist;

- Stable: $\bar{\xi}_i \in A^v$
- Degenerated or boundary: $\bar{\xi}_i \notin A^v$ and $\text{supp}(B_i) \cap A_v \neq \emptyset$
- Exterior: $\text{supp}(B_i) \not\subseteq A^v$

The objective of extending B-splines is to establish stability inside a trimmed basis, by adding polynomial segments of degenerate ones to stable B-splines and deactivating the degenerate B-splines. This effectively removing the problematic unstable splines by adding the stable part of the degenerate spline to neighboring stable splines. This approach was first introduced for uniform knots[17] and later generalized to non-uniform knots[16].

First, we take a trimmed domain $A^v \subseteq \Omega \in \mathbb{R}$ and refer to the trimmed knot span t as $[\xi_t, \xi_{t+1})$ (which has degenerate B-splines). Next, the closest stable knot span s is identified, where all non-zero B-splines are stable. We write the extended polynomial segment $\mathcal{B}_i^s$ of this stable knot span s for spline i in terms of the trimmed knot span t as

$$\mathcal{B}_i^s(\xi) = \sum_{j=t-p}^{t} B_{j,p}(\xi) e_{i,j}, \qquad \xi \in [\xi_t, \xi_{t+1}], \tag{4.22}$$

and use the *extension* coefficients $e_{i,j}$ to extend the stable B-splines $B_{i,p}$ with degenerated B-splines $B_{j,p}$;

$$B_{i,p}^e = B_{i,p} + \sum_{j \in \mathbb{J}_i} e_{i,j} B_{j,p} \tag{4.23}$$

where $\mathbb{J}_i$ contains all indices of degenerated B-splines related to $B_{i,p}^e$.

In some application, in particular in finite element applications with homogeneous boundary conditions, the test functions have to vanish at the boundary to satisfy the boundary conditions. In the context of EB-splines, an additional *weight* function w is introduced[17]. Let w be a positive function which describes the domain and let ξ_i be the center of an inner grid cell of $\text{supp}(B_i)$. Then, the Weighted Extended B-splines (WEB-splines) are defined as

$$B_{i,p}^{we} = \frac{w}{w(\xi_i)} \left(B_{i,p} + \sum_{j \in \mathbb{J}_i} e_{i,j} B_{j,p} \right). \tag{4.24}$$

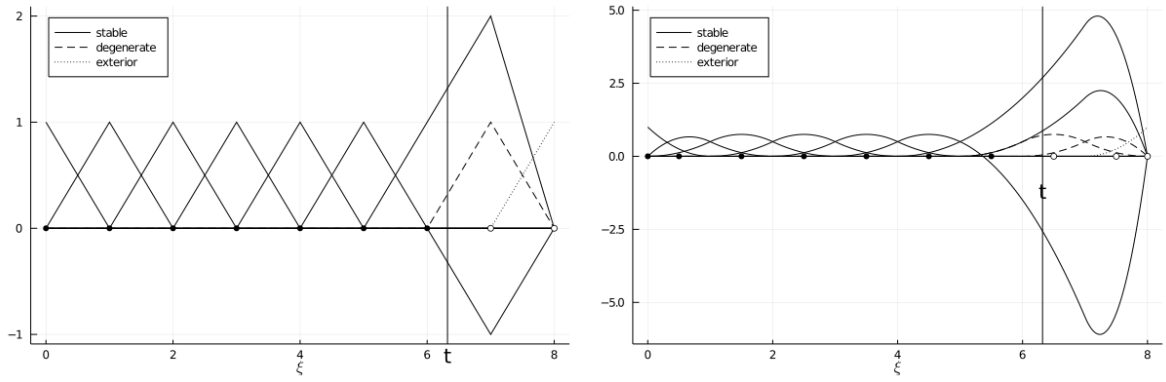
In this thesis, this weight function is not required and henceforth use $w = 1$. Therefore, we will only use the EB-splines.

To compute the extension coefficients $e_{i,j}$, spline interpolation cannot be used, since the chosen anchors or abscissae are not guaranteed the be inside the trimmed knot span t. Therefore, quasi-interpolation is used, in this case using the *de Boor-Fix* functional[6],

$$e_{i,j} = \lambda_{j,p}(\mathcal{B}_i^s) = \frac{1}{p!}\sum_{k=0}^{p} (-1)^k \psi_{j,p}^{(p-k)}(\mu_j) \mathcal{B}_i^{s(k)}(\mu_j), \qquad \mu_j \in [\xi_j, \xi_{j+p+1}], \tag{4.25}$$

where $\psi_{j,p}$ are the *Newton polynomials* defined as

$$\psi_{j,p}(\xi) = \prod_{m=1}^{p} (\xi - \xi_{j+m}) = \sum_{k=0}^{p} \beta_k \xi^k \tag{4.26}$$



(a) $\Xi = [0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 8], p = 1$ **(b)** $\Xi = [0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 8, 8], p = 2$

Figure 4.5: EB-spline basis functions with a polynomial order $p = 1$ and $p = 2$, where the domain in trimmed as $t = 6.3$, generated by eq. 4.23. The interior and exterior anchors are displayed by black and white dots.

As explained in [23, 24], the polynomial segment $\mathcal{B}_i^s$ can be expressed by a Taylor expansion as

$$\mathcal{B}_i^s(\xi) = \sum_{k=0}^{p} \frac{B_{i,p}^{(k)}}{k!}(\xi - \tilde{\xi})^k = \sum_{k=0}^{p} \alpha_k(\xi - \tilde{\xi})^k \tag{4.27}$$

$$= \sum_{k=0}^{p} \tilde{\alpha}_k \xi^k \text{ with } \tilde{\alpha}_k = \sum_{m=k}^{p} \binom{m}{k} \frac{B_{i,p}^{(k)}}{k!}(-\tilde{\xi})^{m-k}. \tag{4.28}$$

Using these coefficients $\tilde{\alpha}_k$ and the coefficients β_k from the *Newton polynomials* $\psi_{j,p}$, the extension coefficients can be written as

$$e_{i,j} = \frac{1}{p!} \sum_{k=0}^{p} (-1)^k (p-k)! \beta_{p-k} k! \tilde{\alpha}_k \tag{4.29}$$

and a one variable example is given in figure 4.5.

4.6. Bivariate EB-splines

In the univariate case, determining which splines are stable, degenerate or exterior is straight forward, and the corresponding extended splines is computed afterwards using the indices from equation 4.22. Since there are at-most 2 positions where trimmings arise, the position for the closest stable knot span is left or right of the trimming.

In a n-variate settings, the closest stable knot span is less clear. To produce a similar extension for n-variate B-splines, the classification of B-splines is slightly altered. The next section refers to bivariate tensor B-splines, where *grid cells* are *rectangles*, but the procedure is identical for 3-dimensional B-splines (*cubes*) or higher dimensional B-splines.

Let $B_i(\xi, \nu) = \phi_{v_i}(\xi)\psi_{w_i}(\mu)$ be the bivariate B-spline, which is tensor product of the univariate B-splines $\phi_{v_i}(\xi)$ and $\psi_{w_i}(\mu)$. The associated knot vectors are indicated by Ξ_ξ and Ξ_μ respectively. The classification of the splines, we use similar terminology as in [16].

The knot vector Ξ_ξ, Ξ_μ can have repeating knots. However, a *grid cell* is defined by two consecutive knots in each parametric direction which are not equal, thus the set C of grid cells is defined as

$$\mathcal{C}^{\xi\times\mu} = \left\{ [\xi_i, \xi_{i+1}] \times [\mu_j, \mu_{j+1}] : \begin{matrix} \xi_i < \xi_{i+1} \text{ where } \xi_i, \xi_{i+1} \in \Xi_\xi, \\ \mu_j < \mu_{j+1} \text{ where } \mu_j, \mu_{j+1} \in \Xi_\mu \end{matrix} \right\} \tag{4.30}$$

We can approximate a function on a bounded domain $D \subset \Omega^0$ by a linear combination of *relevant* B-splines $B_k, k \in K$. However, this set of relevant B-splines can contain *degenerate* B-splines, which have a small support inside D. The classification of as relevant B-spline B_k is then similar as the univariate case, but now using the grid cells.

The grid cells $c_i \in \mathcal{C}^{\xi\times\mu}$ is classified as

- Interior grid cell: $c_i \subset D$

- Boundary grid cell: $c_i \cap D \neq \emptyset$ and $c_i \cap (\Omega \backslash D) \neq \emptyset$
- Exterior grid cell: $c_i \cap D = \emptyset$

and using this classification of the grid cells, a B-spline B_i is then classified by

- Stable B_i: supp(B_i) contains at least one interior grid cell
- Degenerated B_i: supp(B_i) contains no interior grid cells, but does contain boundary cells
- Exterior B_i: supp(B_i) contains only exterior grid cells

and we let J be the index set of degenerate B-splines. The indexset of interior B-splines is then $I = K \backslash J$.

In the process of computing the extended B-splines using equation 4.23, a set $\mathbb{J}_i$ of all degenerate B-splines associated with B_i must be determined.

For a degenerate B-spline $B_j, j \in J$, the closest interior grid cell Q_j to the supp(B_j) must be determined. In Hollig[16] this is done by the Housdorff-metric. Note that in the case of rectangular grid cells and supports, this is identical to the Euclidean metric of its centers.

The related inner splines to B_j are then

$$\mathbb{I}_j = \{i \in I : Q_j \subset \text{supp}(B_i)\} \tag{4.31}$$

and the "reversed" set $\mathbb{J}_i$ is then defined similar to equation 4.22 as

$$\mathbb{J}_i = \{j \in J : i \in \mathbb{I}_j\} \tag{4.32}$$

An example of this process is displayed in figure 4.6.

At last, the bivariate extension coefficients need to be determined. Similar as the bivariate B-splines, the extension coefficients are a tensor product of their univariate coefficients.

Let the tensor product B-spline be written as $B_i(\xi, \mu) = \phi_{v_i}(\xi)\psi_{w_i}(\mu)$ and $B_j(\xi, \mu) = \phi_{v_j}(\xi)\phi_{w_j}(\mu)$, then the extension coefficients can be written as

$$e_{i,j} = e_{v_i,v_j} e_{w_i,w_j} \tag{4.33}$$

4.7. Identifying boundary cells

In the previous section, a definition for the interior and boundary grid cells is given. However, in MPM simulations, the domain D is not clearly defined. Instead, the same boundary particles are used as they were introduced in section 3.3 and the weight function w is set to $w = 1$.

Let $\{p_i^{int}\}$ be the set of interior particles and $\{p_i^{bd}\}$ the set of boundary particles and $\{p_i\} = \{p_i^{int}\} \cap \{p_i^{bd}\}$ with $n_p = |\{p_i\}|$ The grid cells $c_i \in \mathcal{C}^{\xi \times \mu}$ is classified as

- Interior grid cell: $c_i \cap \{p_i^{int}\} \neq \emptyset$ and $c_i \cap \{p_i^{bd}\} = \emptyset$
- Boundary grid cell: $c_i \cap \{p_i^{bd}\} \neq \emptyset$
- Exterior grid cell: $c_i \cap \{p_i\} = \emptyset$

Using the same example as the previous section, the boundary grid cells are identified using boundary particles in figure 4.7. A large enough number of boundary particles must be chosen to get an accurate representation of the domain D.

4.8. Extending THB-splines

Although EB-splines is a powerful method to create a stable basis of a trimmed domain, it is possible that the closest stable knot span to the trimmed knot span is far away. This is especially the case if a coarse knot vector and/or B-splines with a high polynomial order are chosen. This distance between the center of the stable knot span and the trimmed edge is indicated by the extrapolation length d_e, which is displayed in figure 4.8a and 4.8c. If a higher polynomial order or a coarser knot vector is chosen, the extrapolation length d_e can become larger, depending on the trimming.

On the other hand, if a (truncated) hierarchical refinement technique is used near the edge of a trimmed surface, more stability issues can arise as the support is smaller. However, these hierarchical B-spline can reduce the extrapolation length, as displayed in figure 4.8b and 4.8d.

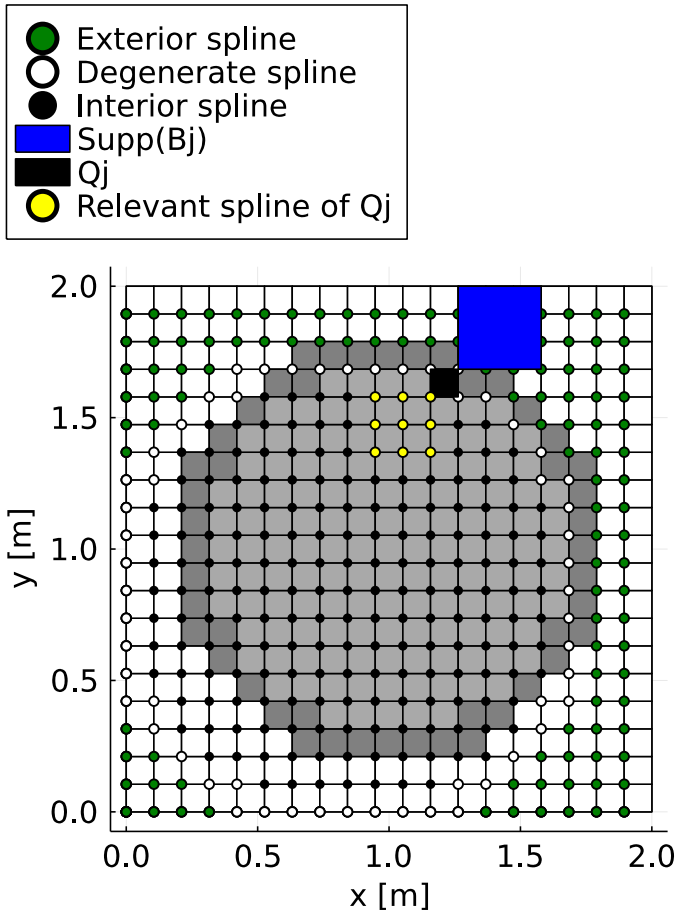


Figure 4.6: Example of the classification process for $\Omega = [0, 2] \times [0, 2]$ with D a circle with center $(1, 1)$ and $r = 0.5$. First, the interior (green), boundary (blue) and exterior (red) grid cells are identified. Next, the stable (black dot), degenerate (white dot) and exterior (green dot) are identified. An example of a $\text{sup}(B_j)$ is indicated by the gray square. For this spline B_j, the closest interior grid cell Q_j is indicated by the black square. At last, the index set of related inner splines $\mathbb{I}_j$ of Q_j is then determined using equation 4.31 and is displayed by the yellow dots.

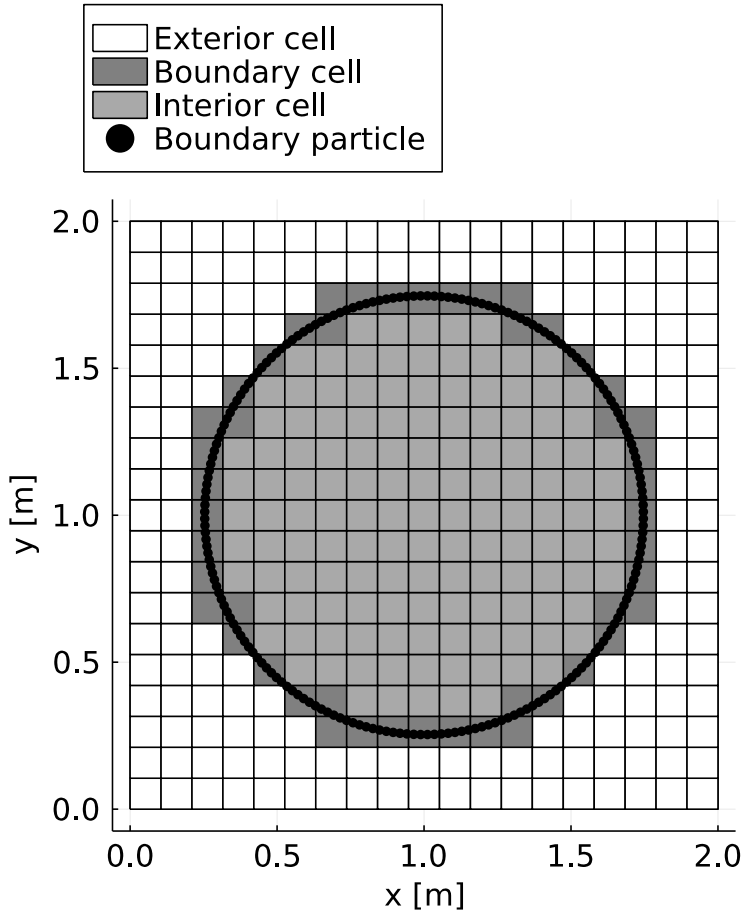
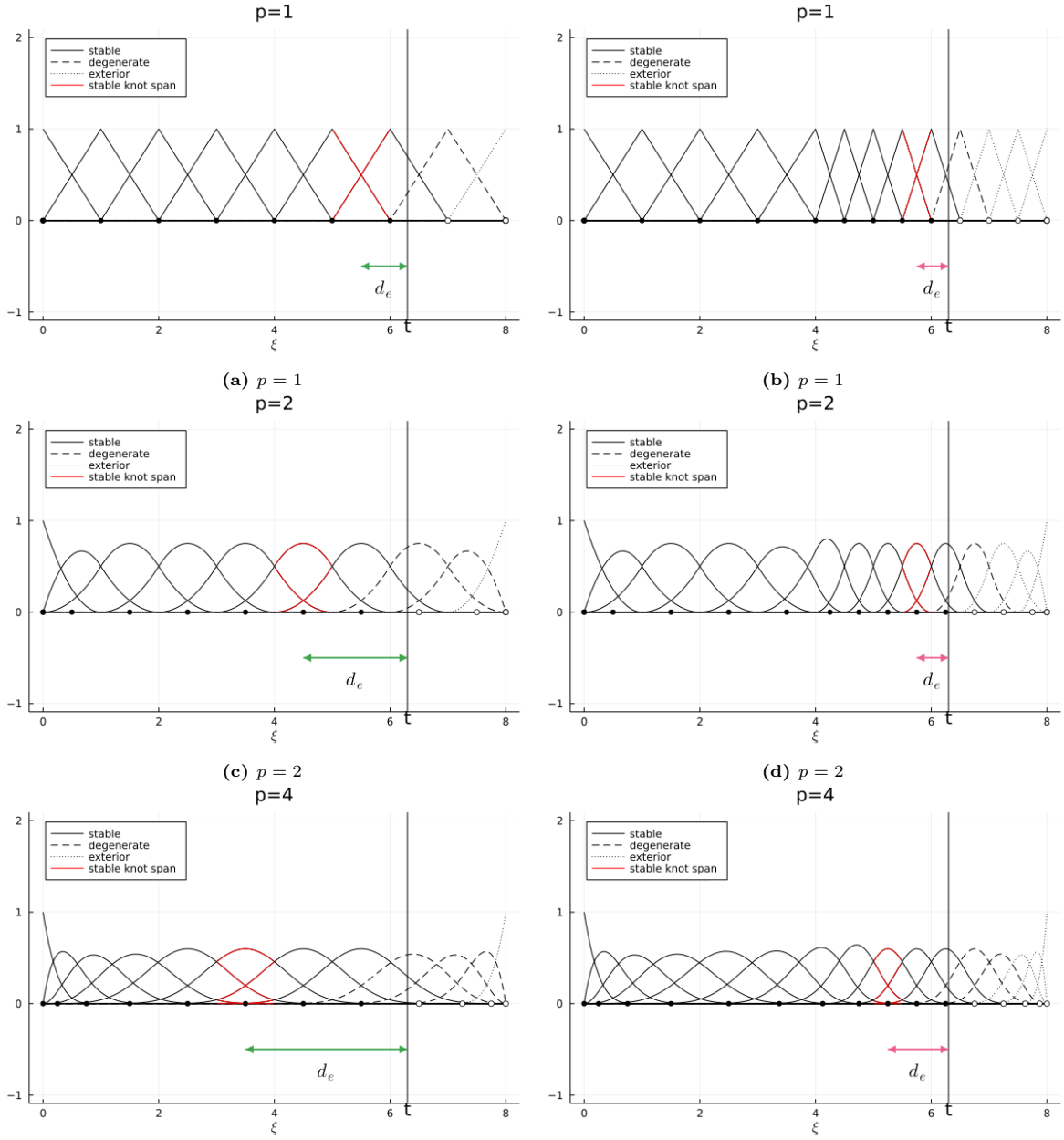


Figure 4.7: Example of the grid cell identification process for $\Omega = [0, 2] \times [0, 2]$ with D a circle with center $(1, 1)$ and $r = 0.5$ using boundary particles. First, the interior (green), boundary (blue) and exterior (red) grid cells are identified. A grid cell is identified as boundary cell if it contains a boundary particle. If a grid cell only contains non-boundary particles, then it's an interior cell. The remaining cells must be exterior grid cells.



(a) $p = 1$ **(b)** $p = 1$

(c) $p = 2$ **(d)** $p = 2$

(e) $p = 4$ **(f)** $p = 4$

Figure 4.8: Illustration of the extrapolation lengths using different polynomial order B-splines, where an open uniform knot vector is used. The trimmed domain of $A^v = [0, t]$ is used, and the extrapolation length d_e to the nearest stable knot span is indicated. The top figures only display unrefined B-splines, with a knot span of length 1. On the bottom, truncated hierarchical B-splines are displayed using two levels, with a knot span of length 1 and $\frac{1}{2}$. The black and white dots indicate the *Greville abscissae*.

A natural result is the combination of both (truncated) hierarchical B-splines and the EB-splines, which is proposed in [23]. Here, a criterion is used to combine both techniques, namely

$$d_e < c_e h_\xi, \tag{4.34}$$

where d_e is the extrapolation length and h_ξ the average knot span size of the non-refined grid. c_e is a user-defined parameter, which this paper proposed to be $c_e = \frac{p}{2}$.

During the process of finding the appropriate level of refinement such that criterion 4.34 is satisfied, the highest refinement level l_{max} is determined. The extended B-spline is only applied to this finest refinement level l_{max}, and the affected B-splines are activated directly. After this, lower-level B-splines are activated or deactivated according to the previously activated higher level B-splines.

In the case of truncated hierarchical B-splines, the partition of unity is already satisfied by using the subdivision matrices and the splines of the finest refinement level are unaffected, meaning that the EB-splines are applicable directly. However, in the case of scaled hierarchical B-spline, EB-splines must employed with more caution. If specific hierarchical B-splines are scaled, we have to make sure that these scaled B-splines are not used in the EB-procedure or that the scaling is incorporated in the extension.

Since both techniques retain the properties of B-splines, they can directly be used together[14]. In the case of THB-splines, the generation of extended B-splines must be altered slightly. Equation 4.23 can then be written as

$$\tau_{i,p}^{l,e} = \tau_{i,p}^{l} + \sum_{j \in \mathbb{J}_i} e_{i,j} \tau_{j,p}^{l} \tag{4.35}$$

and the truncation at a level $l+1$ can be obtained using the truncation equation 4.20 with the subdivision matrix $S^{l,l+1}$ from l to $l+1$

$$\text{trunc}^{l+1} \tau_{i,p}^{l,e} = \sum_{B_k^{l+1} \in \eta_D} S_{ik}^{l,l+1} B_k^{l+1} + \sum_{j \in \mathbb{J}_i} e_{i,j} \left[\sum_{B_k^{l+1} \in \eta_D} S_{jk}^{l,l+1} B_k^{l+1} \right] \tag{4.36}$$

5 Results

5.1. Error analysis

In this chapter, various benchmarks are used to verify the effectiveness of EB- and THB-spline in the context of MPM. To do this, an error estimate must be defined to compare exact values to the approximations. In MPM, the physical quantities such as stress and particle displacement are approximated using B-spline basis function, while the exact values of these quantities can be expressed explicitly in these benchmarks. To compare the error between them, the L^2-norm is used.

For two real-values functions f, g defined on Ω, the difference in the L^2-norm using the inner product is defined as

$$||f-g||_2 = \sqrt{\langle f-g, f-g\rangle} = \sqrt{\int_\Omega (f-g)^2 \mathrm{d}\Omega}. \tag{5.1}$$

In MPM, the particles are used as integration points and the physical quantities are only known at these positions, thus the expression becomes

$$||f_{\text{exact}} - f_{\text{mpm}}||_2 = \sqrt{\int_\Omega (f_{\text{exact}} - f_{\text{mpm}})^2 \mathrm{d}\Omega} = \sqrt{\sum_{p=1}^{n_p} V_p (f_{\text{exact}}(\mathbf{X}_p) - f_p)^2}. \tag{5.2}$$

Note that this error norm is similar to the RMS-error if all particles have identical volumes with a total volume of 1.

5.2. Quasi 2D dynamic traction at the boundary

In this section, the performance of extended B-splines and B-splines in MPM are compared. For this, the dynamic traction at the boundary is used, as described in section 3.6.4. However, since this thesis will focus primarily on domains in the 2-dimensional setting, this problem is extended.

Let $D = [0, L] \times [0, H]$ be the material domain, where the same traction force is applied at the right-hand side, thus at $x = L(t)$. This force is applied homogeneous along the edge. The traction force has to be modified slightly,

$$\begin{aligned}
\mathbf{F}_i^{\text{tract}} &= \oint_\Gamma \phi_i \mathbf{n} \cdot \boldsymbol{\sigma} \mathrm{d}\Gamma \\
&= \int_0^H \phi_{i,p}(L(t)) \phi_j(y) \begin{bmatrix} \sigma(L(t),t) \\ 0 \end{bmatrix} \mathrm{d}y \\
&= \begin{bmatrix} \phi_{i,p}(L(t)) \sigma(L(t),t) \\ 0 \end{bmatrix} \int_0^H \phi_{i,p}(y) \mathrm{d}y \\
&= \begin{bmatrix} \phi_{i,p}(L(t)) \sigma(L(t),t) \\ 0 \end{bmatrix} \frac{\xi_{i+p+1} - \xi_i}{p+1} \sum_{j=i}^{n} \phi_{j,p+1}(\xi).
\end{aligned}$$

In the last step, an expression for the integration of a B-spline is used, which is derived in appendix C.

To avoid the nearly-empty cell problem as much as possible, a background grid is chosen such that no empty cells exist. For this benchmark, the maximum extension will be at $t = 1s$, and $u(L,t) \approx 0.0064m$. Therefore, an appropriate background grid is chosen to be $\Omega = [0, 1.15] \times [0, 1]$. To get an accurate representation of the influence of EB-splines compared to B-splines, the difference between the two methods is minimized. For this benchmark, the extension is only employed in the x-direction, i.e.,

$$\text{2-dimensional B-splines: } \phi_{ij}(x,y) = \phi_i(x)\phi_j(y) \tag{5.3}$$

$$\text{Quasi 2-dimensional EB-splines: } \phi^e_{ij}(x,y) = \phi^e_i(x)\phi_j(y) \tag{5.4}$$

For the Dirichlet boundary condition on the left-hand side, $x = 0$, the penalty-based approach is used. While the boundary coincides with the knotline for regular B-splines, this is less clear for EB-splines where degenerate B-splines are deactivated while stable splines are extended. Since the performance of B-splines and EB-splines is compared, the same boundary imposing method is used.

To compare the difference in performance, the exact solutions for the stress (eq. 3.38) and displacement (eq. 3.39) are used. Therefore, they will be no interaction in the y-direction, thus let Poisson ration $\nu = 0$. The other quantities are

$$\begin{aligned} L &= 1m \\ H &= 0.25m \\ \nu &= 0 \\ \rho &= 100\text{kg} \cdot m^{-3} \\ E &= 100N \cdot m^{-2} \\ \Delta t &= 10^{-4}s \end{aligned}$$

5.2.1. Convergence

Although higher polynomial order B-splines results attempts to increase the order of the quadrature error, if the L^2-error is calculated using polynomial order $p = 1, 2, 3$, this is not the case, as displayed in figure 5.1. Initially the error reduces linearly and quadratically for $p = 1, 2$ respectively, and one might expect the error of $p = 3$ with a similar rate. This is not the case, as the error reduces quadratically as well.

This follows from the fact that the material points are used as integration points, resulting in type of midpoint rule[30]. Although the midpoint rule itself is third-order, the stress calculations are still second order, resulting in the same order of error for $p = 2$ and $p = 3$.

Note that the error for the linear B-splines eventually increases because of the grid-crossing error.

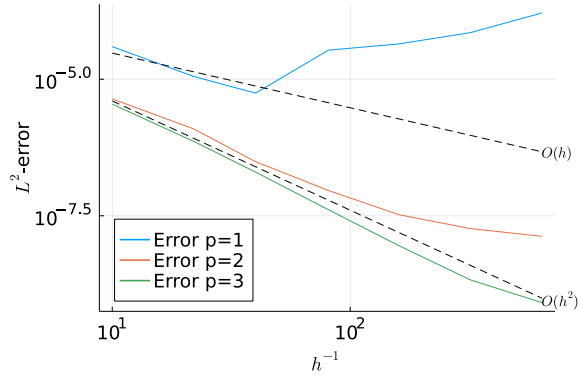
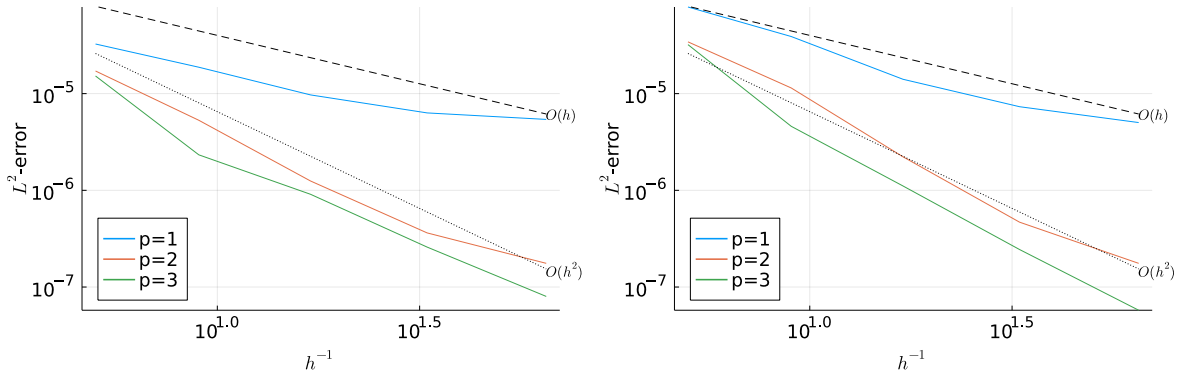


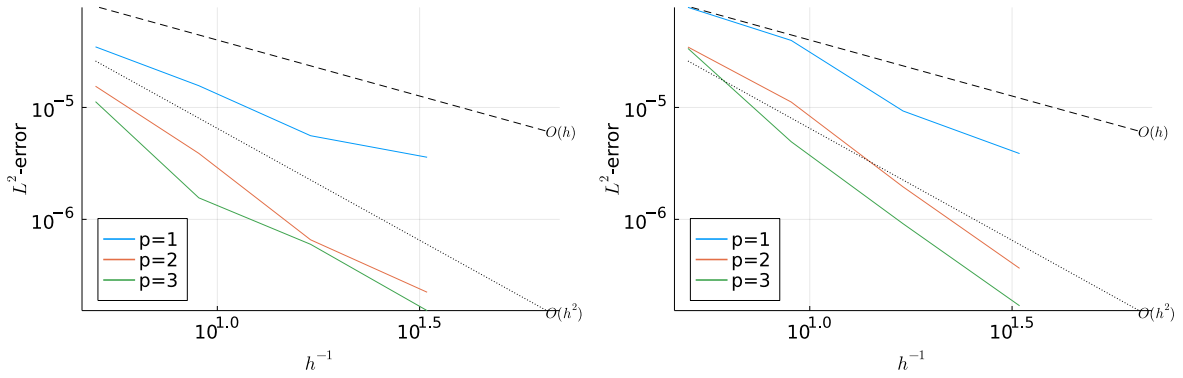
Figure 5.1: L^2-error of the 1D dynamic traction boundary condition at time $t = 0.5s$ for 1D.

In the 2D-benchmark, a similar convergence rate can be observed, both in the B-spline and EB-spline MPM,



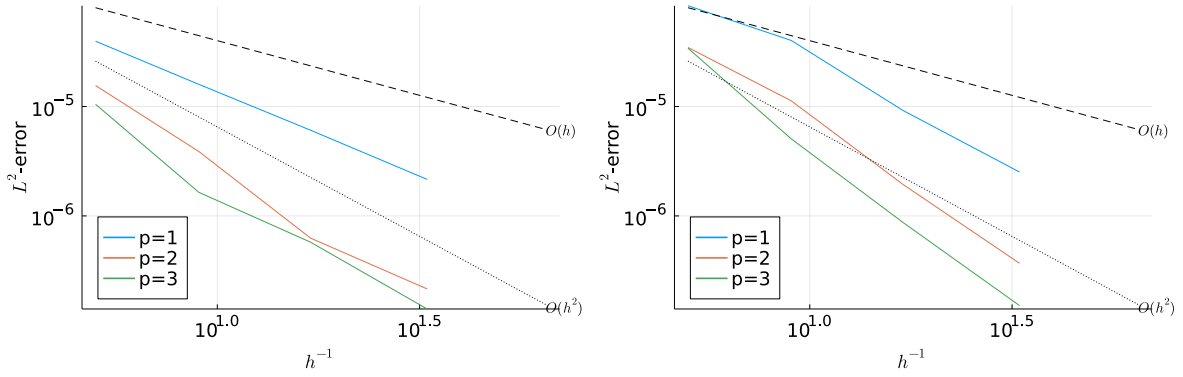
(a) B-splines **(b)** EB-splines

Figure 5.2: L^2-error of the 2-dimensional dynamic traction benchmark when using B-splines and EB-splines in MPM and 4^2 particles per cell.



(a) B-splines **(b)** EB-splines

Figure 5.3: L^2-error of the 2-dimensional dynamic traction benchmark when using B-splines and EB-splines in MPM and 8^2 particles per cell.



(a) B-splines **(b)** EB-splines

Figure 5.4: L^2-error of the 2-dimensional dynamic traction benchmark when using B-splines and EB-splines in MPM and 16^2 particles per cell.

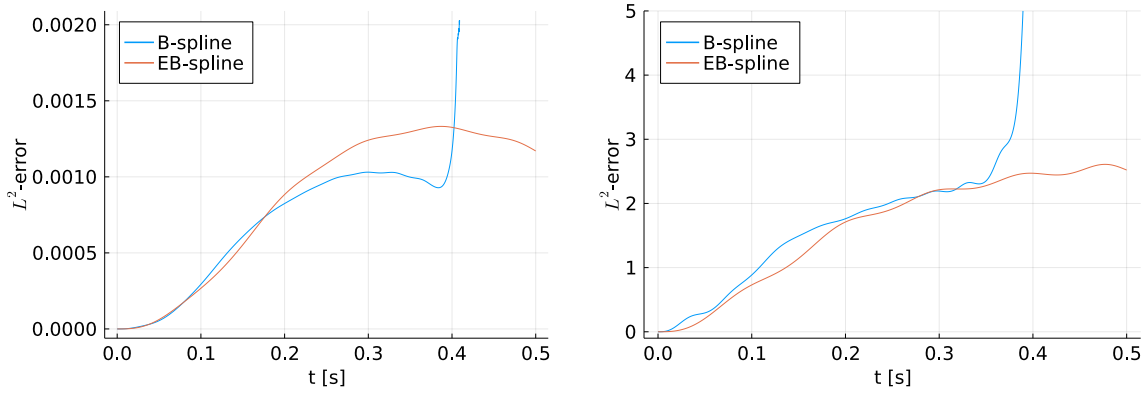
5.2.2. Nearly-empty cell problem

In the previous section, it was shown that the BS-MPM and the extension have a similar performance, where the background grid was chosen to be $[0, 1.15] \times [0, 1]$. One of the characteristics of MPM is that the background grid is independent of the material considered (if the material is contained in the background grid). Therefore, by choosing a different grid, a similar approximation for the displacement

and stress would be expected. Let the new background grid be

$$\Omega = [-0.25, 1.25] \times [-0.25, 1.25], h = 0.09375m \tag{5.5}$$

and keep all other parameters identical to the previous section. The L^2-error of this problem over time is displayed in figure 5.5



(a) L^2-error of the displacement

(b) L^2-error of σ_{xx}

Figure 5.5: L^2-error of the 2-dimensional dynamic traction benchmark when using B-splines and EB-splines in MPM and 4^2 particles per cell.

5.2.3. Discussion

The dynamic traction at the boundary benchmark is performed in both the 1- and 2-dimensional setting. The 1-dimensional benchmark is a well-known benchmark in MPM and the L^2-error has a similar convergence rate as the literature [30].

In the 2-dimensional setting, the same benchmark is performed with a Poisson number $\nu = 0$. Since there is no interaction between the strain in x and y direction, the displacement in the x-direction should be identical to the 1-dimensional case. The following L^2-error convergence is of the same order as in the 1-dimensional case, for both B-spline and EB-splines.

Furthermore, the magnitude is the error for B-splines and EB-splines is comparable in figures 5.2, 5.3 and 5.4. In the procedure of extending the B-splines, some splines are deactivated, resulting in less degrees of freedom. The resulting error is therefore slightly higher for a large mesh size (meaning a less degrees of freedom). For a smaller mesh size, this effect becomes less noticeable.

These results are obtained by choosing an ideal background grid for the problem at hand. This is not always the case, as the background grid is independent of the material in MPM. This can result in nearly-empty cells. An example of this phenomenon can be seen in figure 5.5 using B-splines where a less-ideal background grid is chosen. This results in a nearly-empty cell around $4s$, and the L^2-error for both stress and displacements increases drastically.

Since EB-splines extend stable B-splines to incorporate the degenerate (small support) B-splines, each spline in the computation has a large support. This results in no more nearly empty cells, and thus a more stable MPM method.

5.3. Semi-clamped plate benchmark

The dynamic traction at the boundary benchmark from the previous section did show that the B-spline and EB-spline based MPM perform similar when the background grid is chosen to avoid nearly-empty cells. However, since the benchmark is a quasi-2D benchmark, the extension to EB-splines is not fully explored.

Therefore, another benchmark is constructed using the Method of Manufacturing solutions (MMS). This is a well-known method in numerical mathematics where the exact solution is assumed, and the applied forces, initial conditions and boundary conditions are derived from the exact solution. This is because many "simple" 2D-problems are already too complex to easily obtain an explicit solution.

In this benchmark, a plane is considered with $\Omega_0 = [0, L] \times [0, H]$. The assumed displacement in this benchmark is

$$\mathbf{u}(x, y, t) = \begin{bmatrix} u_e(1 - e^{-\omega_x t}) \cdot x \cdot \left(\frac{y}{H}\right)^2 \\ u_e(1 - e^{-\omega_y t}) \cdot y \cdot \left(\frac{x}{L}\right)^2 \end{bmatrix}. \tag{5.6}$$

For simplicity, the problem is assumed to be symmetric along its diagonal and let $L = H$ and $\omega = \omega_x = \omega_y$, thus the displacement becomes

$$\mathbf{u}(x, y, t) = \frac{u_e}{L^2}(1 - e^{-\omega t}) \begin{bmatrix} x \cdot y^2 \\ x^2 \cdot y \end{bmatrix}. \tag{5.7}$$

Since this displacement solution must satisfy the conservation of linear momentum in equation 2.8, a body force is derived in appendix B as

$$\mathbf{b} = -\frac{u_e}{L^2}\omega^2 e^{-\omega t} \begin{bmatrix} xy^2 \\ x^2 y \end{bmatrix} + \frac{u_e}{L}\frac{2\lambda + 4\mu}{\rho_0}(1 - e^{-\omega t}) \begin{bmatrix} x \\ y \end{bmatrix}. \tag{5.8}$$

While all constitutive relation for the stress $\boldsymbol{\sigma}$ can be used in the MMS and the resulting body force should always match the displacement solution. However, for a somewhat physically representative benchmark, the approximation for small deformations is used for simplicity, eq. 2.11. The initial conditions for the displacement, velocity and stress can directly be obtained from the exact solutions at $t = 0$, see appendix equations B.6-B.8.

For the boundary conditions, the boundary is split into 4 parts consistent with the 4 edges of the initial rectangular domain. For the bottom ($y = 0$) and left ($x = 0$) boundary, a Dirichlet boundary condition is imposed, where the exact value can be obtained from the exact solution equation 5.6. To impose this condition in the MPM computation, the penalty method is used.

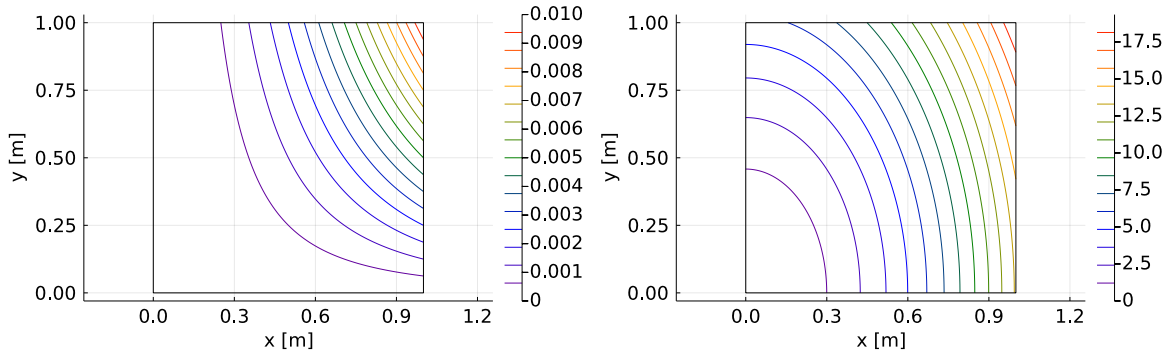
On the top ($y = H$) and right ($x = L$) boundary, a traction boundary condition is imposed. The exact traction can then be obtained from the exact stress and the normal vector, which are derived in appendix B. The resulting boundary force vector becomes

$$\mathbf{F}_i^{\text{boundary}} = \int_{F_{\text{left}} \cap F_{\text{bottom}}} \phi_i \epsilon^{-1}(\mathbf{u}^h - \mathbf{u}) \mathrm{d}\Gamma + \int_{F_{\text{top}} \cap F_{\text{right}}} \phi_i \mathbf{n} \cdot \boldsymbol{\sigma} \mathrm{d}\Gamma \tag{5.9}$$

The parameters for this benchmark are set to

$$\begin{aligned} E &= 1000N \cdot m^{-2} \\ \rho &= 1000kg \cdot m^{-3} \\ \nu &= 0.3 \\ u_e &= 0.01 \\ L &= H = 1m \\ \Delta t &= 10^{-3}s \\ \omega &= \frac{\sqrt{\frac{E}{\rho}}}{L} \end{aligned}$$

Using these parameters, the results of this benchmark are displayed in figure 5.6. Note that the magnitudes of the displacement and stress are symmetric along the diagonal, but the individual components are not, but are mirrored along the diagonal. For example, consider the exact displacement in equation 5.6 at a point $(\hat{x}, \hat{y})$. We find that the $u_x(\hat{x}, \hat{y}, t) = u_y(\hat{y}, \hat{x}, t)$. A similar observation can be made for the stress components, i.e., $\sigma_{xx}(\hat{x}, \hat{y}, t) = \sigma_{yy}(\hat{y}, \hat{x}, t)$.



(a) u_x **(b)** σ_{xx}

Figure 5.6: Visualization of the x-displacement and Cauchy stress component σ_{xx} in the plane when $t \to \infty$.

5.3.1. Error analysis

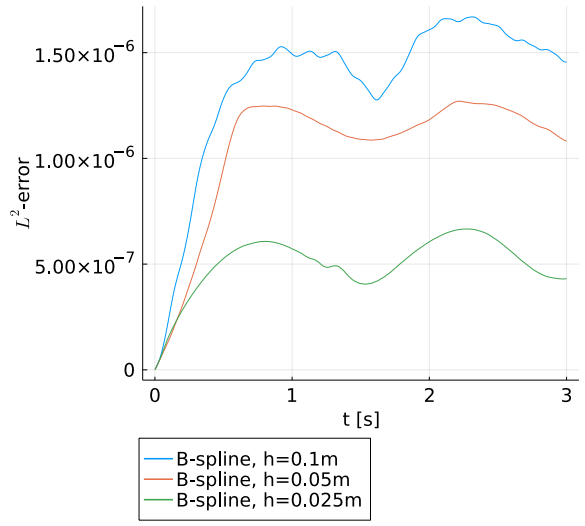
The above-described problem is solved using 4^2 particles per cell using 3 different background grids. The background grid consists of square cells, with a size $h = 0.1m, 0.05m, 0,025m$. The L^2-error of the solution over time is displayed in figure 5.7 using both B-splines and EB-spline based MPM. To obtain a better understanding of the behavior of the two methods, the σ_{xx} solution at the end of the simulation is displayed in figure 5.8.

5.3.2. Discussion

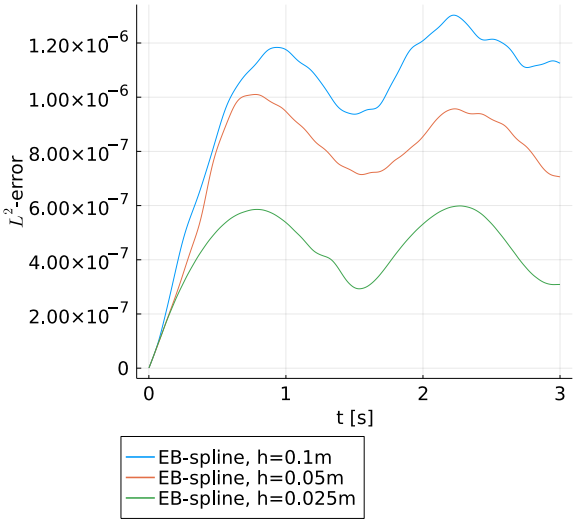
In this section, the semi-clamped plated benchmark is employed. The problem is computed using both B-spline and EB-spline MPM, which perform comparable. However, since the top and right edge of the material plane move outwards, the error near the edge becomes more prominent, especially when a low number of particles is considered, which can be seen in figure 5.7, where the error in the stress is lower when using EB-spline MPM compared to B-spline MPM. For a larger number of particles, this effect becomes less noticeable.

An explanation for this can be seen in figure 5.8 where the error stress oscillations near the edge are displayed. This effect causes non-physical waves to propagate into the material, causing additional errors. This effect is smaller for EB-splines compared to B-spline MPM, which explains the slightly lower error in the displacement and stress.

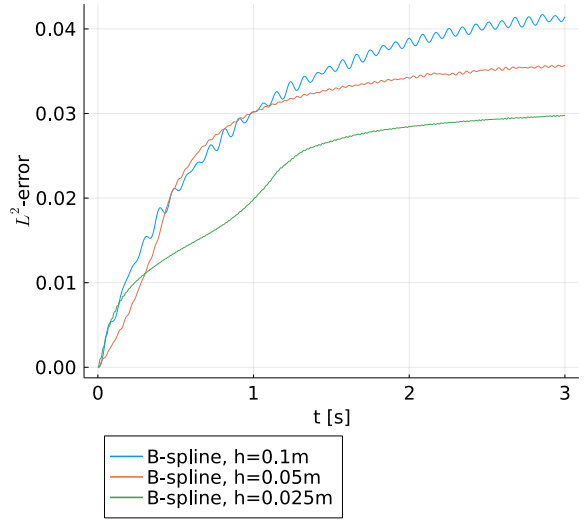
In this benchmark, the THB-spline based MPM can be used to solve the problem. However, local refinement techniques would not significantly impact the quality of the solution. As displayed in figure 5.6, the problem has no local stress concentrations. Refining the background grid near the top-right corner would reduce the error in that area. However, the other regions have an error of a similar order, which will dominate compared to the refined region. Therefore, the benchmark is slightly altered in the next section to investigate the performance of THB-splines.



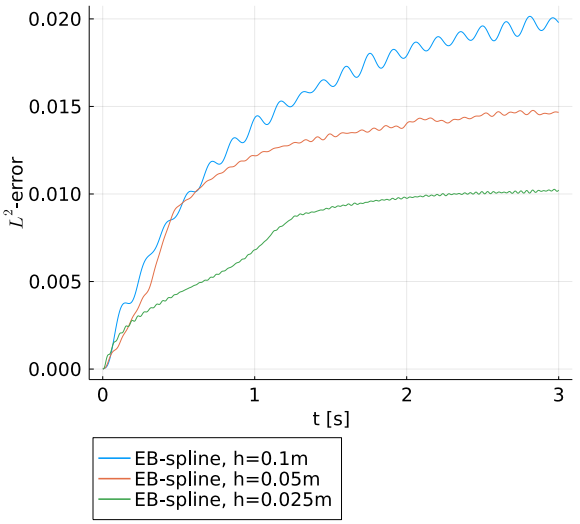
(a) L^2-error of the x-displacement u_x over time using B-splines



(b) L^2-error of the x-displacement over time using EB-splines



(c) L^2-error of the Cauchy stress σ_{xx} over time using B-splines



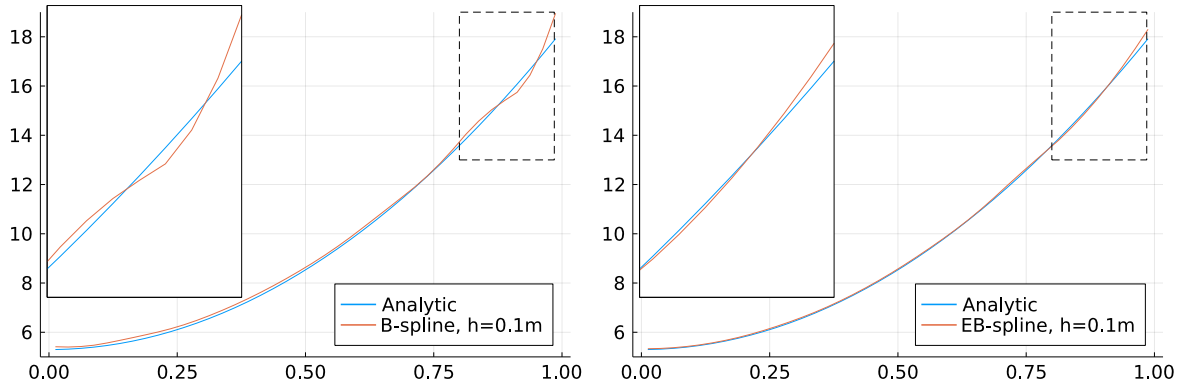
(d) L^2-error of the Cauchy stress σ_{xx} over time using EB-splines

Figure 5.7: L^2-error of the x-displacement u_x and the Cauchy stress σ_{xx} over time using 2^2particles per cell. Note, the vertical axis in the left and right figures use a different scale.

5.4. Semi-clamped plate benchmark with localized stress concentration

In the previous section, the semi-clamped plate benchmark was introduced, with a quadratic displacement and stress in the domain $[0, L] \times [0, H]$. Therefore, the stress concentrations are not particularly local and the use of a local refinement technique such as THB-splines is not effective. The benchmark is slightly altered, to create a more local stress concentration;

$$\mathbf{u}(x, y, t) = \frac{u_e}{L^2}(1 - e^{-\omega t}) \begin{bmatrix} x^5 \cdot y^{10} \\ x^{10} \cdot y^5 \end{bmatrix}. \tag{5.10}$$



(a) B-spline MPM **(b)** EB-spline MPM

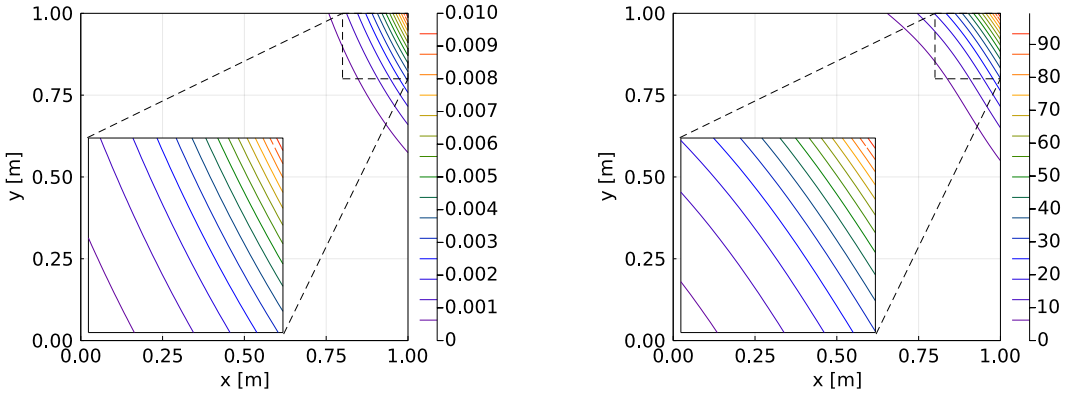
Figure 5.8: Solution for σ_{xx} near the edge of the material domain at time $t = 3.0s$ using $h = 0.1m$.

For a more realistic example, for the constitutive model, the Neo-Hookean model is used

$$\boldsymbol{\sigma} = \lambda J^{-1} \log J \mathbf{I} + J^{-1} \mu (\mathbf{F}\mathbf{F}^T - \mathbf{I}), \tag{5.11}$$

and the same MMS approach is used for the derivation of the body force and the initial and boundary conditions as in the previous benchmark, see appendix B. The long-time solution for this problem is computed and displayed in figure 5.9.

Here, we see again that while the magnitude of the solution is symmetric along the diagonal, the individual components are not. We find for a point $(\hat{x}, \hat{y})$ for the displacement $u_x(\hat{x}, \hat{y}, t) = u_y \hat{y}, \hat{x}, t$. A similar observation can be made for the stress components.



(a) u_x **(b)** σ_{xx}

Figure 5.9: Magnitude of the displacement and σ_{xx} at a time $t \to \infty$.

5.4.1. THB-splines

To quantify the behavior of THB-splines in combination with EB-splines, the semi-clamped plate benchmark is used. The background grid for the MPM computations is refined in the top-right corner using 2 refinement levels;

$$\Omega_0 = [0, L + u_e] \times [0, H + u_e] \tag{5.12}$$

$$\Omega_1 = \left[\frac{1}{2}L, L + u_e\right] \times \left[\frac{1}{2}H, H + u_e\right] \tag{5.13}$$

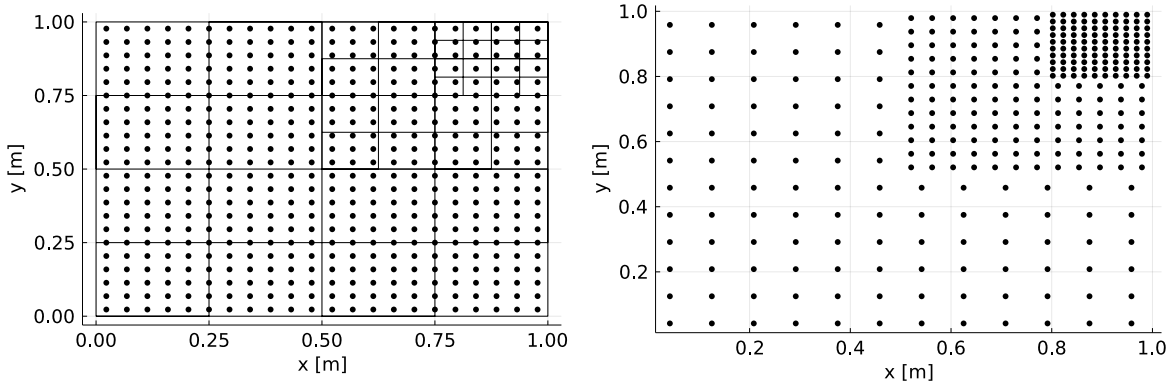
$$\Omega_1 = \left[\frac{4}{5}L, L + u_e\right] \times \left[\frac{4}{5}H, H + u_e\right], \tag{5.14}$$

which can be visualized in figure 4.3b.

	ndof B-spline	ndof EB-spline	np 2ppc	np 4ppc	np 8ppc
h=0.1m	144	100	400	1600	6400
h=0.05m	484	400	1600	6400	25600
h=0.025m	1764	1600	6400	25600	102400
THB, l=1	219	165	700	2800	12864
THB, l=2	267	205	892	3568	15280

Table 5.1: Active degrees of freedom (ndof) and number of particles (np) for each simulation, displayed in figures 5.11, 5.12 and 5.13. Degenerate B-splines are deactivated in the EB-spline process, resulting in less degrees of freedom. The first 3 rows display the simulations without truncated hierarchical B-splines, and the last to rows the use of (E)THB-splines with 1 and 2 refinement levels.

When refining the background grid, the number of particles must be considered. In the previous simulation, a distribution is used where the number of particles per grid cell is used as a representation. However, if this same strategy is used in THB-splines, the refinement level must be considered. If the ground level Ω_0 is used for the reference cell size, with for example 8 particles per grid cell, this would mean that for grid cells of Ω^2, there is only 1 particle per cell which can cause stability issues, see figure 5.10a. In this thesis, this distribution is referred to as the *uniform particle distribution* on Ω_0. Note that for a quadratic bivariate B-spline without repeating knots, the support consists of 3×3 grid cells.



(a) Uniform particle distribution in a hierarchical grid based on Ω_0

(b) Uniform particle distribution in a hierarchical grid based on each subdomain Ω_l

Figure 5.10: Uniform particle distribution in a hierarchical grid based on domains $\Omega_0 - \Omega_2$ of equation 5.12-5.14

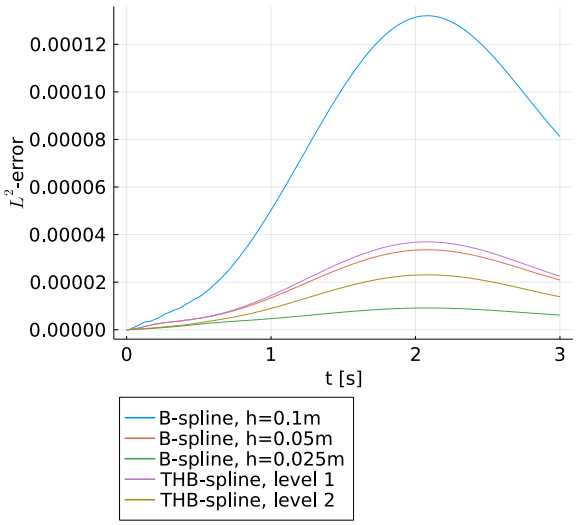
On the other hand, the particles can be distributed with a similar strategy of uniform particle distribution based on the highest-level domain at a given location. This means that each grid cell of a given domain Ω_l has the same number of particles, uniformly distributed, but the initial particles are not uniformly distributed in Ω_0, see figure 5.10b. In this thesis, this distribution will be referred to as the *nonuniform particle distribution* on Ω_0.

5.4.2. Error analysis

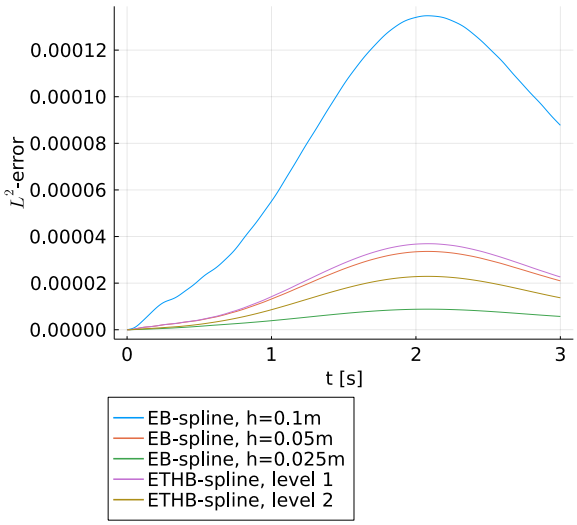
To compare the two approaches, the benchmark is computed for 3000 time steps using $\Delta t = 10^{-3}s$ using square grid cells, i.e., $h_x = h_y$ and the index is omitted. For the cell size of Ω_0, $h_0 = 0.1m$ is used. Therefore, the sizes on the refined levels are $h_1 = 0.05m$ and $h_2 = 0.025m$. In figure 5.7, the L^2-error of the displacement and σ_{xx} are displayed when using regular B-splines on the finest grid size h_2 and THB-splines using only one refinement level Ω_1 and using both refinement levels Ω_1 and Ω_2. The results for 4^2 and 8^2 particles per cells are displayed in figures 5.12 and 5.13.

5.4.3. Discussion

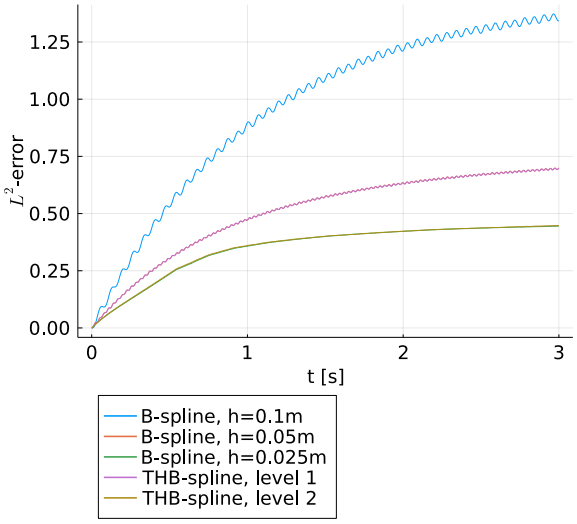
In this section, a 2D-benchmark with a more local stress concentration is employed. The process of solving the solution comparable to the benchmark in the previous section, but in this benchmark, the THB-spline MPM is included in the computations. The location of the stress concentration is known inside the domain (the top-right corner), and this is refined on 2 levels using the THB-splines. The



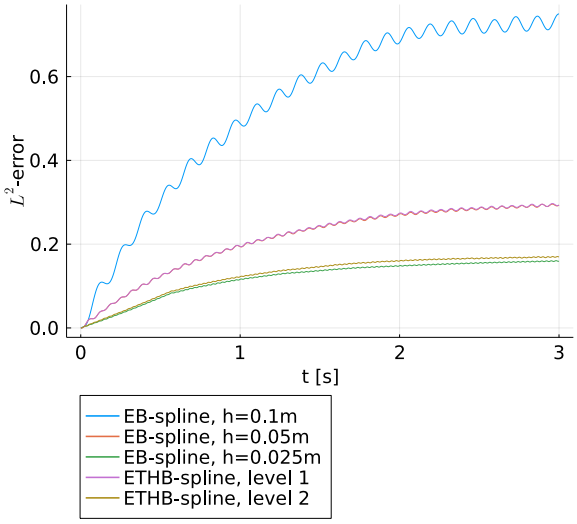
(a) L^2-error of the displacement over time using B-splines



(b) L^2-error of the displacement over time using EB-splines



(c) L^2-error of σ_{xx} over time using B-splines



(d) L^2-error of σ_{xx} over time using EB-splines

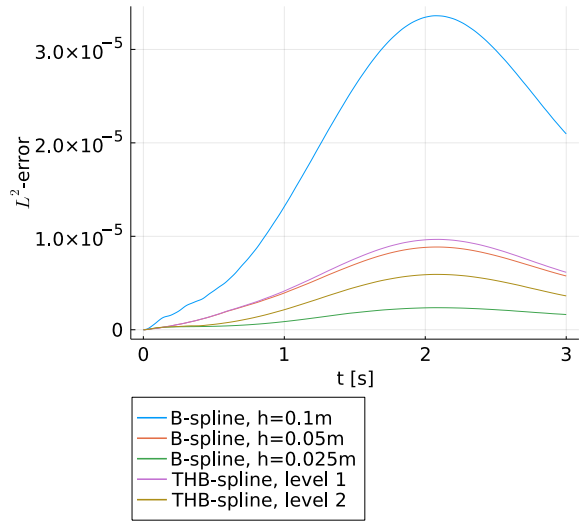
Figure 5.11: L^2-error of the displacement and σ_{xx} over time using 2^2particles per cell. Note, the vertical axis in the left and right figures use a different scale.

L^2-error in the displacement u_x and Cauchy stress σ_{xx} is displayed in figures 5.11,5.12 and 5.13. Note that the y-axis for B-spline and EB-spline are different.

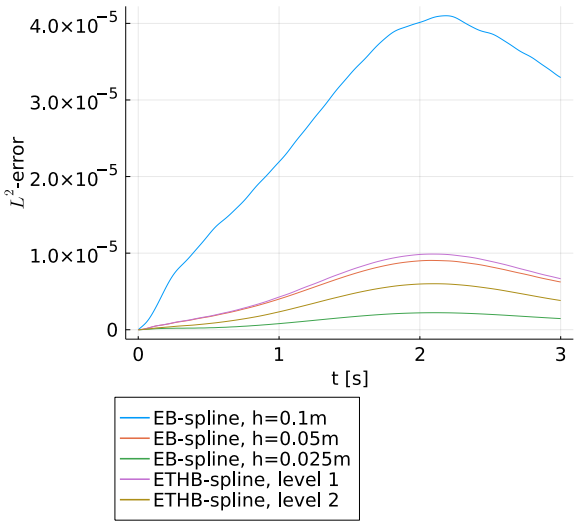
EB-splines

For 2^2, it can be observed that while the error in the displacement is of a comparable magnitude in both B-spline and EB-spline based MPM, this is not the case for the Cauchy stress. Here, the EB-spline based MPM has a factor 2 lower stress error. Using 4^2 particles per cell, this is not the case anymore and the L^2-error are similar for $h = 0.1m$. For smaller meshes, EB-spline perform better than B-splines. For the 8^2 particles per cell, it can be observed that error in the stress in B-spline MPM is lower than the EB-spline MPM in the case for $h = 0.1m$. For smaller meshes, both methods are of a similar magnitude.

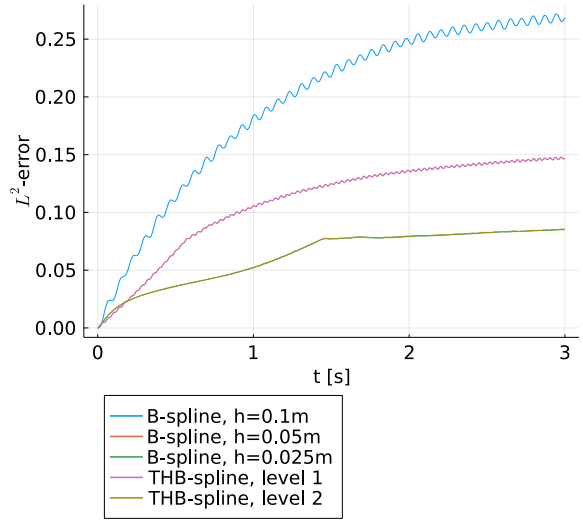
In MPM, the quality of the solution is dependent on both the number of particles and the degrees of freedom of the background grid. By reducing the mesh size h, the degrees of freedom increase



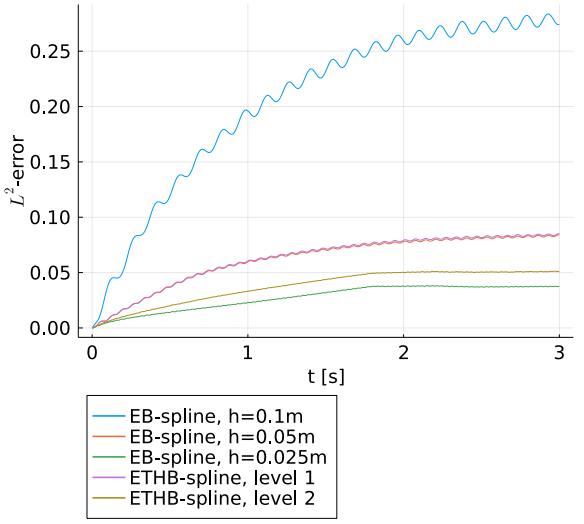
(a) L^2-error of the displacement over time using B-splines



(b) L^2-error of the displacement over time using EB-splines



(c) L^2-error of σ_{xx} over time using B-splines

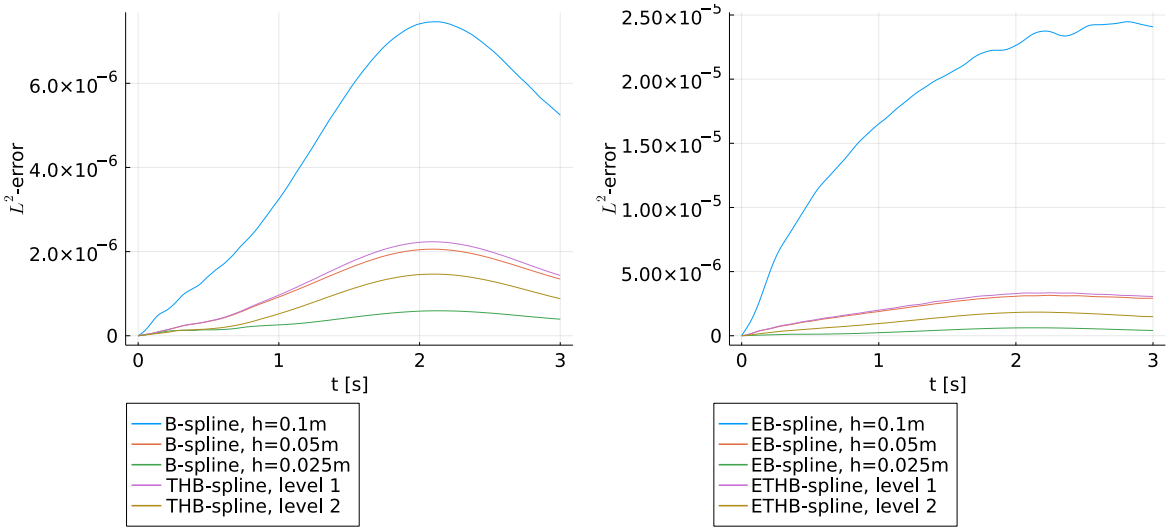


(d) L^2-error of σ_{xx} over time using EB-splines

Figure 5.12: L^2-error of the displacement and σ_{xx} over time using 4^2particles per cell.

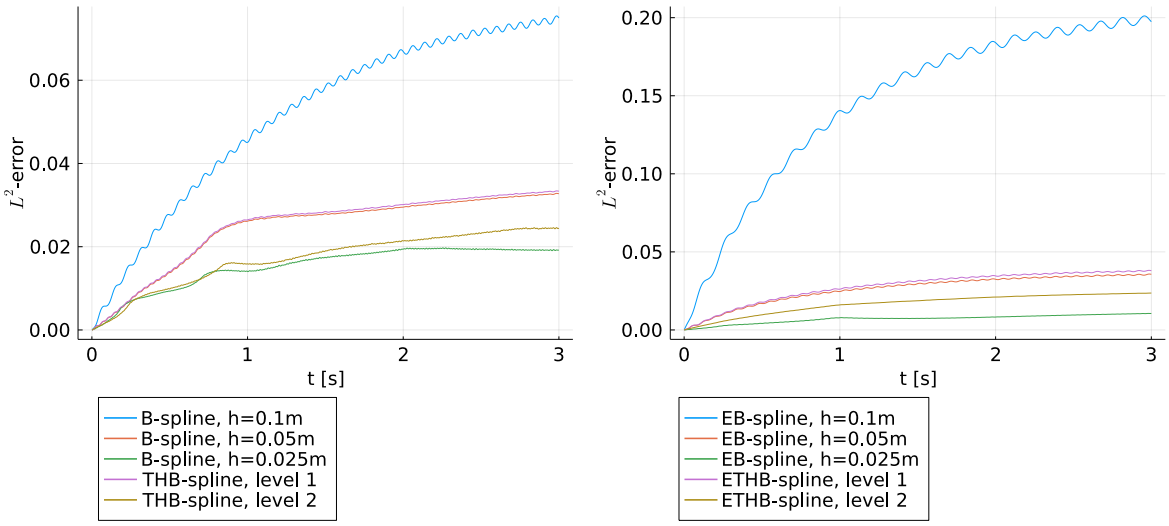
and the error is reduced. By increasing the number of particles, the projection of the scalar and vector quantities is more accurate resulting in a reduced error. In table 5.1, the degrees of freedom and number of particles are displayed.

In the process of stabilizing B-splines by deactivating degenerate splines, the degrees of freedom are reduced. This is relatively more the case for smaller mesh sizes. In figure 5.11, only 2^2 particles per cell are used in the computations. This small number of particles has an impact of on the accuracy and the effect of using EB-splines to reduce the error near the boundary is noticeable. This effect is observed in the previous benchmark as well, see figure 5.8. However, as the number of particles increases, the physical quantities are interpolated more accurate, and the effect of the low degrees of freedom dominates. The difference in degrees of freedom between B-spline and EB-splines for $h = 0.1m$ is significant (144 vs 100), and reduces for $h = 0.05m$ (444 vs 400) and $h = 0.025m$ (1764 vs 1600), resulting in a less noticeable effect for smaller grid sizes.



(a) L^2-error of the displacement over time using B-splines

(b) L^2-error of the displacement over time using EB-splines



(c) L^2-error of σ_{xx} over time using B-splines

(d) L^2-error of σ_{xx} over time using EB-splines

Figure 5.13: L^2-error of the displacement and σ_{xx} over time using 8^2particles per cell.

THB-splines

However, in real application, it is desired to reduce the number of particles as much as possible, since large amount of particles increases the memory and computational cost. This is also the case when a large number of degrees of freedoms are desired. Therefore, THB-splines are also used to refine the top-right corner of the problem.

It can be observed that in all figures, the L^2-error of THB-spline MPM with 1 refinement level is comparable to B-spline MPM with $h = 0.05m$. With only 1 refinement, the top-right corner of the THB-grid coincides with the B-spline grid of $h = 0.05m$. The same holds for the particles in this part of the domain. For THB-spline MPM with 2 refinement levels, this observation can be made when comparing to its B-spline counterpart with $h = 0.025m$.

This does indicate that the solution does benefit from local refinement in this part of the domain. Both THB-splines and B-splines perform similar when the same configuration in the top-right corner is used. THB-splines are more complex to implement and does increase the computational time at a degree of freedom. However, courser grid sizes can be used in other parts of the domain resulting in

much less degrees of freedom needed to obtain the solution. This can be observed in table 5.1.

5.4.4. Stress oscillations

In each result, there are stress oscillations present in the L^2-error. This is a nonphysical behavior that must be explained in the following section.

Unlike the oscillations from figure 3.6, these oscillations are not a result of the grid crossing error. The grid-crossing error is induced by the discontinuous jumps of linear basis functions and quadratic basis functions reduces this effect. Smaller grid-sizes result in more particles crossing the grid boundaries, which should increase the number of jumps. However, the opposite is observed as higher degrees of freedom increases the oscillations frequency and reducing the amplitude.

The stress oscillations are a result of inaccurate approximations of the boundary conditions, which affects the top-right corner the most. The effect of the inaccuracies at the boundary particles is displayed in figure 5.14, where the L^2-error in the stress is computed using 5.1 using subdomains of Ω_0.

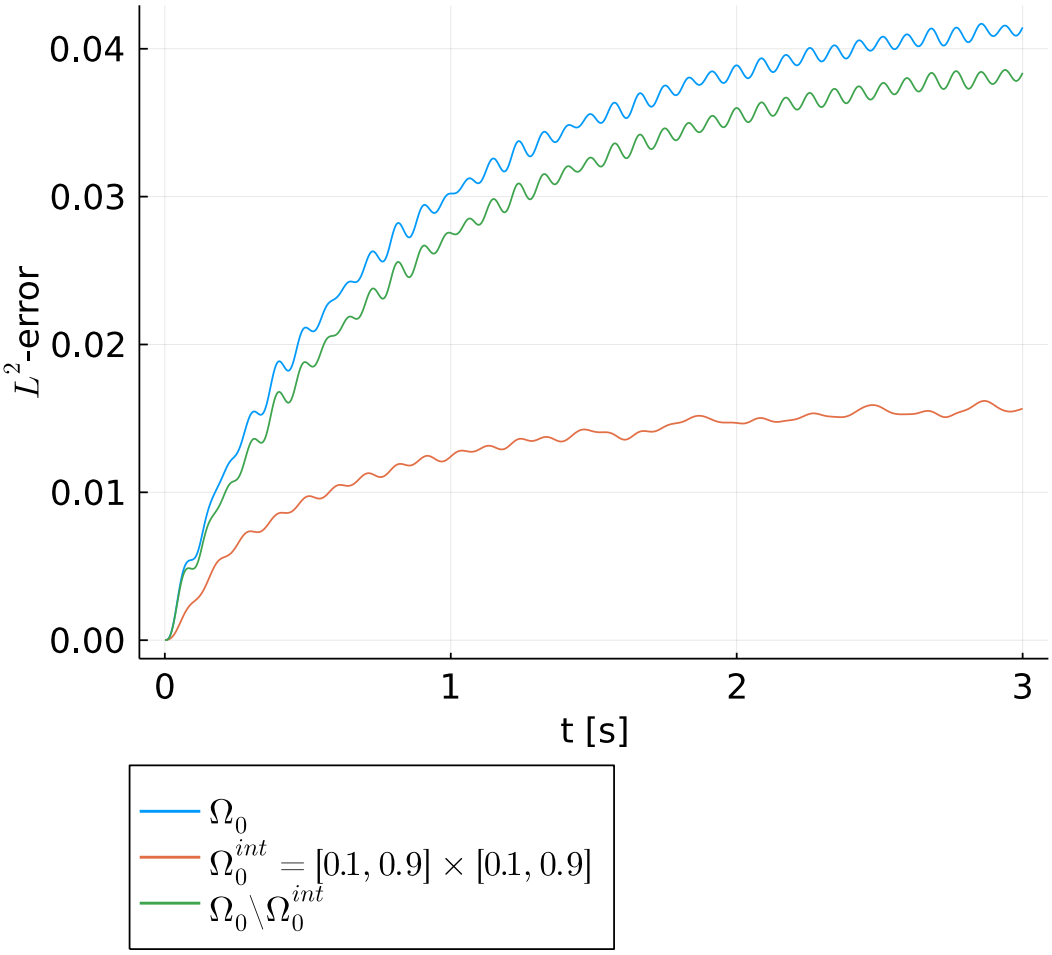
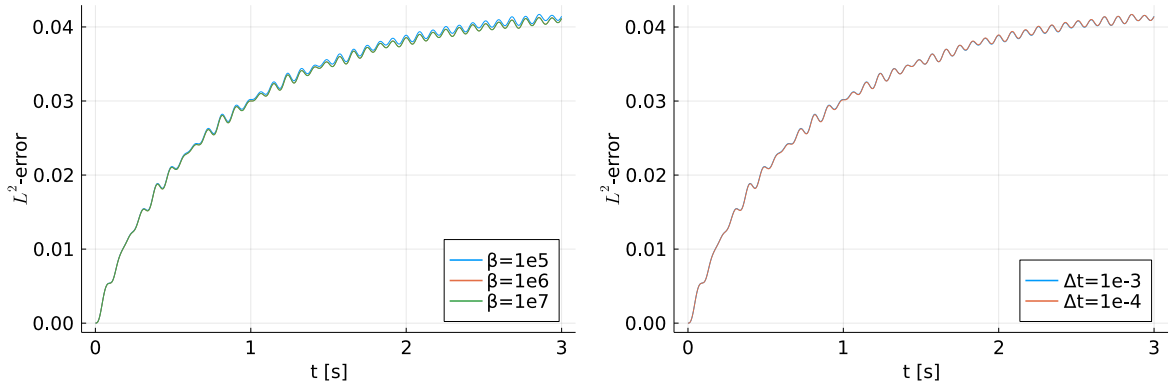


Figure 5.14: L^2-error over time in different subdomains of Ω_0, using equation 5.1 using B-splines with $h = 0.1$. It can be observed that stress inaccuracies are dominated by the particles near the boundary. Stress oscillations over time are mainly present in the boundary particles.

To increase the quality of the solution near the edge, one might suggest increasing the $\beta = \epsilon^{-1}$ parameter of the Dirichlet boundary condition. However, as is displayed in figure 5.15a, varying this value has no effect on the oscillations. A similar observation can be made by reducing the time step Δt in figure 5.15b.



(a) L^2-error over time using different values $\beta = \epsilon^{-1}$ to approximate the Dirichlet boundary conditions.

(b) L^2-error over time using different time steps Δt.

Figure 5.15: L^2-error over time using B-splines, 4 particles per cell and $h = 0.1m$. Changing the value of β and the time step Δt has no significant effect on the stress oscillations.

To increase the accuracy of the solution at the boundary, the number of particles can increase and the degrees of freedom. In figures 5.11, 5.12 and 5.13, it is observed that both the number of particles and degrees of freedom reduce the overall error. By increasing the number of particles, the magnitude of the error reduces, and therefore the amplitude of the oscillations. However, the relative amplitude compared to the error magnitude is unaffected.

On the other hand, increasing the degrees of freedom reduces the amplitude of the oscillations by increasing the frequency. Therefore, the oscillations are less observable for $h = 0.05m$ and $h = 0.025m$. Furthermore, this explains the difference between the oscillations in B-splines and EB-splines, as EB-splines has fewer degrees of freedom, see table 5.1.

In short, this effect is a result of inaccurate boundary conditions, and can be reduces by increasing the degrees of freedom. This behavior is not observed in other benchmarks, where other error contributions dominate. In the semi-clamped plate benchmark, this effect is amplified as the stress gradient is largest at the top-right corner boundary.

6 Conclusion

The Material Point Method is a numerical technique used to simulate solid, liquids and other continuum material. It uses material points or particles, which can move freely in space and are independent of the computational background grid. It is therefore a well-known method when simulating large deformations and multi-phase interactions.

This independence between the particles and background grid causes instabilities such as grid-crossing and the nearly-empty cells. The grid-crossing error is a direct result of the linear interpolation basis function and a C^1 or smoother basis function can eliminate this problem. A possible solution for this problem originates from Isogeometric analysis, where quadratic B-spline basis functions are used to overcome this problem. However, these basis functions do not resolve the nearly-empty cell problem, and the primary focus of this thesis is to increase the stability of B-spline MPM by eliminating this problem.

Apart from the fact that the background grid must be chosen sufficiently large to contain the material, it is independent of the material and particles. The particles can move freely during the simulation, making it possible for them to enter or leave grid cells and potentially leaving cell empty or nearly-empty. This leaves some B-splines with a very small support, causing the approximations to be unstable. This phenomenon also results in small mass entries at these degrees of freedom, causing the mass matrix to be ill-conditioned.

6.1. EB-splines

This thesis explores the use of Extended B-splines (EB-splines) to resolve the nearly-empty cell problem, by extending the support of stable B-splines to incorporate the small support of the unstable B-splines and deactivating them. This approach guarantees that only stable B-splines are active in MPM simulations, which eliminates the nearly-empty cell problem. A 1- and 2-dimensional benchmark with a dynamic traction boundary condition is used to verify this behavior.

Two conclusions can be drawn from this benchmark. At first, an ideal background grid is chosen such that no nearly-empty cell problem arises. The two-spline approximation perform similar in the context of MPM, with a similar magnitude and convergence rate in the L^2-error. There are slight variations in the error, which can be explained by the slight difference in active degrees of freedom and the particles being non-ideal integration point. This is the behavior that is desired of these methods when no nearly-empty cell problems are present.

Secondly, a background grid is chosen were nearly-empty cell problem are present. This causes significant errors in the stress en displacement computations when using B-splines. Because of these errors, the particles move with nonphysical speeds through the domain that, after a couple of time steps, the boundary particles leave the expected domain and causes the program to stop prematurely. This problem is not present when using EB-splines, where the behavior is similar to simulations where other background grids are used. This makes the EB-spline MPM method a more reliable and stable method.

However, there is still a drawback to the use of EB-splines opposed to B-splines. B-spline functions form a partition of unity, which guarantees a positive mass matrix. By using EB-splines, this is not the

case. The diagonal mass element will always be positive, but the off-diagonal entries can be negative when the mass is concentrated near the edge. While the mass matrix will not be singular, it is possible in theory that the lumped mass matrix often used in MPM applications becomes singular. Further research has to be done to determine if this can indeed be a problem is some cases and if it can be resolved.

6.2. THB-splines

The problem of nearly-empty cells can be eliminated by using extended B-splines and this opens the door to locally refine the background grid near the edge. Normally, this would increase the likeliness of nearly-empty cell, since the cells are smaller but not by using the extended B-splines. Therefore, this thesis explores the use of Truncated Hierarchical B-splines (THB-splines) to locally refine the MPM background grid.

The Method of Manufacturing Solutions is used to create a new artificial benchmark, the semi-clamped plate benchmark with localized stress concentrations, where local refinement can be used near the edge. Using the technique, the exact solution is explicitly known, and an accurate error study can be performed.

This benchmark is solved using both regular B-splines and THB-splines, with and without the extension. To accurately compare the various methods, the number of particles and the cell size h on the finest level (near the edge) are compared. Indeed, the L^2-error in these simulations are of the same order, with the ETHB-spline MPM performing slightly better compared to THB-splines MPM.

The main difference is in the degrees of freedom and the number of particles. While the L^2-error in these runs are similar, the degrees of freedom by using two refinement THB-splines significantly lower compared to the regular B-splines (267 and 1764 respectively). Within the THB-spline approach, the number of particles can be chosen to be uniformly distributed or altered slightly according to the refinement levels. This results in a similar approximation result while simultaneously using significantly less particles, see table 5.1.

Therefore, the use of THB-splines is a very powerful extension to the existing B-spline Material Point Method when stress concentrations are present in the problem. By refining the interested domain locally, the degrees of freedom are significantly lower than by refining the B-spline background grid globally. By distributing the particles efficiently over the refinement levels, the number of particles can be reduced as well. Both observations are desirable effect from a computational point since less particles and less degrees of freedom reduce the memory cost and the computational time.

In this thesis, the refinement levels used in the benchmark are chosen statically since the exact solution is known and therefore the locations of the stress concentrations. In real problem, this is not the case and refinement levels have to be chosen dynamically or adaptively when concentrations disappear. For an advanced and accurate approximation, this is desirable. Further research has to be done to investigate the performance of these techniques in a MPM setting.

References

[1] Ivo Babuška. "The finite element method with Lagrangian multipliers". In: *Numerische Mathematik* 20.3 (1973), pp. 179–192. ISSN: 0029599X. DOI: 10.1007/BF01436561.

[2] S G Bardenhagen and E M Kober. "The Generalized Interpolation Material Point Method". In: *Tech Science Press cmes* 5.6 (2004), pp. 477–495.

[3] S. G. Bardenhagen. "Energy conservation error in the material point method for solid mechanics". In: *Journal of Computational Physics* 180.1 (2002), pp. 383–403. ISSN: 00219991. DOI: 10.1006/jcph.2002.7103.

[4] John W. Barrett and Charles M. Elliott. "Finite element approximation of the Dirichlet problem using the boundary penalty method". In: *Numerische Mathematik* 49.4 (1986), pp. 343–366. ISSN: 0029599X. DOI: 10.1007/BF01389536.

[5] T. Belytschko, W.K. Liu, and B. Moran. *Nonlinear Finite Elements for Continua and Structures*. Wiley, 2000. ISBN: 0-471-98773-5.

[6] C. de Boor and G. J. Fix. "Spline approximation by quasiinterpolants". In: *Journal of Approximation Theory* 8.1 (May 1973), pp. 19–45. ISSN: 10960430. DOI: 10.1016/0021-9045(73)90029-4.

[7] Carl de Boor. *A Practical Guide to Spline*. Vol. Volume 27. Jan. 1978. DOI: 10.2307/2006241.

[8] J.U. Brackbill and H.M. Ruppel. "FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions". In: *Journal of Computational Physics* 65.2 (1986), pp. 314–343. ISSN: 0021-9991. DOI: https://doi.org/10.1016/0021-9991(86)90211-1. URL: https://www.sciencedirect.com/science/article/pii/0021999186902111.

[9] Bodhinanda Chandra et al. "Nonconforming Dirichlet boundary conditions in implicit material point method by means of penalty augmentation". In: *Acta Geotechnica* 16.8 (2021), pp. 2315–2335. ISSN: 18611133. DOI: 10.1007/s11440-020-01123-3.

[10] Alban De Vaucorbeil et al. "Material point method after 25 years: Theory, implementation, and applications". In: *Advances in Applied Mechanics*. Vol. 53. 2020, pp. 185–398. DOI: 10.1016/bs.aams.2019.11.001.

[11] Michael R. Dörfel, Bert Jüttler, and Bernd Simeon. "Adaptive isogeometric analysis by local h-refinement with T-splines". In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (Jan. 2010), pp. 264–275. ISSN: 00457825. DOI: 10.1016/j.cma.2008.07.012.

[12] David R. Forsey and Richard H. Bartels. "Hierarchical B-spline refinement". In: *Computer Graphics (ACM)* 22.4 (1988), pp. 205–212. ISSN: 00978930. DOI: 10.1145/378456.378512.

[13] David R. Forsey and Richard H. Bartels. "Surface Fitting with Hierarchical Splines". In: *ACM Transactions on Graphics (TOG)* 14.2 (Jan. 1995), pp. 134–161. ISSN: 15577368. DOI: 10.1145/221659.221665.

[14] Carlotta Giannelli, Bert Jüttler, and Hendrik Speleers. "THB-splines: The truncated basis for hierarchical splines". In: *Computer Aided Geometric Design*. Vol. 29. 7. North-Holland, Oct. 2012, pp. 485–498. DOI: 10.1016/j.cagd.2012.03.025.

[15] Francis H Harlow. "The particle-in-cell method for numerical solution of problems in fluid dynamics". In: (Mar. 1962). DOI: 10.2172/4769185. URL: https://www.osti.gov/biblio/4769185.

[16] Klaus Höllig and Ulrich Reif. "Nonuniform web-splines". In: *Computer Aided Geometric Design* 20.5 (2003), pp. 277–294. ISSN: 01678396. DOI: 10.1016/S0167-8396(03)00045-1.

[17] Klaus Höllig, Ulrich Reif, and Joachim Wipper. "Weighted extended B-spline approximation of Dirichlet problems". In: *SIAM Journal on Numerical Analysis* 39.2 (2002), pp. 442–462. ISSN: 00361429. DOI: 10.1137/S0036142900373208.

[18] Cornelius O. Horgan and Giuseppe Saccomandi. "Constitutive Models for Compressible Nonlinearly Elastic Materials with Limiting Chain Extensibility". In: *Journal of Elasticity* 77.2 (Nov. 2004), pp. 123–138. ISSN: 1573-2681. DOI: 10.1007/s10659-005-4408-x. URL: https://doi.org/10.1007/s10659-005-4408-x.

[19] T. J.R. Hughes, J. A. Cottrell, and Y. Bazilevs. "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement". In: *Computer Methods in Applied Mechanics and Engineering* 194.39-41 (2005), pp. 4135–4195. ISSN: 00457825. DOI: 10.1016/j.cma.2004.10.008.

[20] Kjetil André Johannessen, Trond Kvamsdal, and Tor Dokken. "Isogeometric analysis using LR B-splines". In: *Computer Methods in Applied Mechanics and Engineering* 269 (Feb. 2014), pp. 471–514. ISSN: 00457825. DOI: 10.1016/j.cma.2013.09.014.

[21] Mika Juntunen and Rolf Stenberg. "Nitsche's method for general boundary conditions". In: *Mathematics of Computation* 78.267 (2009), pp. 1353–1374. ISSN: 0025-5718. DOI: 10.1090/s0025-5718-08-02183-2.

[22] Pascal de Koster et al. "Extension of B-spline Material Point Method for unstructured triangular grids using Powell–Sabin splines". In: *Computational Particle Mechanics* (2020). ISSN: 21964386. DOI: 10.1007/s40571-020-00328-3. URL: https://doi.org/10.1007/s40571-020-00328-3.

[23] Benjamin Marussig, René Hiemstra, and Thomas J.R. Hughes. "Improved conditioning of isogeometric analysis matrices for trimmed geometries". In: *Computer Methods in Applied Mechanics and Engineering* 334 (2018), pp. 79–110. ISSN: 00457825. DOI: 10.1016/j.cma.2018.01.052. URL: https://doi.org/10.1016/j.cma.2018.01.052.

[24] Benjamin Marussig et al. "Stable isogeometric analysis of trimmed geometries". In: *Computer Methods in Applied Mechanics and Engineering* 316 (Apr. 2017), pp. 497–521. ISSN: 00457825. DOI: 10.1016/j.cma.2016.07.040. arXiv: 1603.09660.

[25] J. Nitsche. "Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind". In: *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg* 36.1 (1971), pp. 9–15. ISSN: 18658784. DOI: 10.1007/BF02995904.

[26] J. Tinsley (John Tinsley) Oden. "An introduction to mathematical modeling : a course in mechanics". In: (2011), p. 334.

[27] R.W. Ogden. *Non-linear Elastic Deformations*. Dover Publications. Dover Publications, 1997. ISBN: 0-486-69648-0.

[28] A. Sadeghirad, R. M. Brannon, and J. Burghardt. "A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations". In: *International Journal for Numerical Methods in Engineering* 86.12 (June 2011), pp. 1435–1456. ISSN: 1097-0207. DOI: 10.1002/NME.3110. URL: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/full/10.1002/nme.3110%20https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/abs/10.1002/nme.3110%20https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/10.1002/nme.3110.

[29] A.J.M. Spencer. *Continuum Mechanics*. Longman, 1980. ISBN: 0-582-44282-6.

[30] M. Steffen, R.M. Kirby, and M. Berzins. "Analysis and reduction of quadrature errors in the material point method (MPM)". In: *International Journal for Numerical Methods in Engineering* 76.6 (2008), pp. 922–948. DOI: 10.1002/nme.2360.

[31] D. Sulsky, Z. Chen, and H.L. Schreyer. "A particle method for history-dependent materials". In: *Computer Methods in Applied Mechanics and Engineering* 118.1 (1994), pp. 179–196. ISSN: 0045-7825. DOI: https://doi.org/10.1016/0045-7825(94)90112-0. URL: https://www.sciencedirect.com/science/article/pii/0045782594901120.

[32] Deborah Sulsky and Ming Gong. "Improving the Material-Point Method". In: *Lecture Notes in Applied and Computational Mechanics* 81 (2016), pp. 217–240. DOI: 10.1007/978-3-319-39022-2_10. URL: https://link-springer-com.tudelft.idm.oclc.org/chapter/10.1007/978-3-319-39022-2_10.

[33] Deborah Sulsky, Shi Jian Zhou, and Howard L. Schreyer. "Application of a particle-in-cell method to solid mechanics". In: *Computer Physics Communications* 87.1-2 (1995), pp. 236–252. ISSN: 00104655. DOI: 10.1016/0010-4655(94)00170-7.

[34] Roel Tielen et al. "A High Order Material Point Method". In: *Procedia Engineering* 175 (2017), pp. 265–272. ISSN: 18777058. DOI: 10.1016/j.proeng.2017.01.022.

[35] JJIM van Kan, A Segal, and FJ Vermolen. *Numerical methods in scientific computing*. Undefined/Unknown. VSSD, 2005. ISBN: 90-71301-50-8.

[36] A. V. Vuong et al. "A hierarchical approach to adaptive local refinement in isogeometric analysis". In: *Computer Methods in Applied Mechanics and Engineering* 200.49-52 (Dec. 2011), pp. 3554–3567. ISSN: 00457825. DOI: 10.1016/j.cma.2011.09.004.

[37] Elizaveta Wobbes et al. "Comparison and unification of material-point and optimal transportation meshfree methods". In: *Computational Particle Mechanics* (2020). ISSN: 21964386. DOI: 10.1007/s40571-020-00316-7. URL: https://doi.org/10.1007/s40571-020-00316-7.

[38] Elizaveta Wobbes et al. "Conservative Taylor least squares reconstruction with application to material point methods". In: *International Journal for Numerical Methods in Engineering* 117.3 (2019), pp. 271–290. ISSN: 10970207. DOI: 10.1002/nme.5956.

[39] Duan Z. Zhang, Xia Ma, and Paul T. Giguere. "Material point method enhanced by modified gradient of shape function". In: *Journal of Computational Physics* 230.16 (July 2011), pp. 6379–6398. ISSN: 0021-9991.

A

Hyperelastic material

To completely specify the behavior of a material, additional equations are required. These equations are called *constitutive equations*. Modeling a specific or specific type of material requires the use of a constitutive model [29]. Generally, these equations encompass all properties of the material, such as thermodynamics, but in this thesis, we are only interested in the stress response of a material.

For simplicity, only *simple elastic materials* or *Cauchy elastic materials* are considered. In these materials, the current state of the stress solely depends on its deformation with respect to an arbitrary reference configuration[27], and we can write

$$\boldsymbol{\sigma} = G(\mathbf{F}). \tag{A.1}$$

However, this does not necessarily mean that the work done by the stress is independent of the deformation path and therefore cannot express the work as a function of the deformation.

Hyperelastic or *Green elastic materials* are a special case of these materials where the work can be expressed by a scalar function dependent on the deformation. This function is called the *strain-energy density function* $W(\mathbf{F})$.

A.1. Strain-energy density function

A constitutive relation can be derived from the strain-energy density function. This relationship should describe the material and should not be dependent of the coordinate system and rigid body motions. Therefore, constructing the constitutive relation by de deformation $\mathbf{F}$ directly is not ideal.

Instead, the *right Cauchy-Green deformation tensor* was introduced as

$$\mathbf{C} = \mathbf{F}^T\mathbf{F}, \tag{A.2}$$

which makes $\mathbf{C}$ a rotation independent tensor. This is desirable as the pure rotation of a material is a rigid body motion and should not contribute to the material strain.

For *isotropic* hyperelastic materials, the strain-energy density function can then be expressed in terms of principle invariants of $\mathbf{C}$[27]. Since this is a rank 2 tensor, the invariants can be expressed as

$$\begin{aligned} I_1 &= \mathrm{tr}(\mathbf{C}) = \lambda_1 + \lambda_2 + \lambda_3 \\ I_2 &= \frac{1}{2}\left((\mathrm{tr}(\mathbf{C})^2 - \mathrm{tr}\left(\mathbf{C}^2\right)\right) = \lambda_1\lambda_2 + \lambda_1\lambda_3 + \lambda_2\lambda_3 \\ I_3 &= \mathrm{det}\mathbf{C} = \lambda_1\lambda_2\lambda_3 \end{aligned}$$

Using these invariants, we write the strain-energy density function[18] as

$$W(I_1, I_2, I_3) = \tilde{W}(I_1, I_2, I_3) + W_{vol}(J) \tag{A.3}$$

where $J = \mathrm{det}\mathbf{F} = \sqrt{I_3}$. The second term is a pure volume term and is a direct result of the pressure-volume response. For incompressible materials, we have $J = 1$ and $W_{vol}(J) = 0$.

For isotropic compressible hyperelastic materials, various models are proposed for the density function and verified to experimental data. In this thesis, we use the *Neo-Hookean material model*, which is an extension of the isotropic linear law for large deformations[5]. For this type of materials, the strain-energy density functions can be approximated by

$$\tilde{W}(I_1, I_2, I_3) = \frac{\mu}{2}(I_1 - 3 - 2\ln J) \tag{A.4}$$

$$W_{vol}(J) = \frac{\lambda}{2}(\ln J)^2 \tag{A.5}$$

A.2. Neo-Hookean constitutive equation

From this energy density function, the Cauchy stress $\boldsymbol{\sigma}$ can be computing using the relation[5]

$$\boldsymbol{\sigma} = J^{-1}\mathbf{F}\mathbf{S}\mathbf{F}^T, \tag{A.6}$$

where $\mathbf{S}$ is the *second Piola–Kirchhoff stress* (PK2). This stress can be derived from

$$\begin{aligned}\mathbf{S} &= 2\frac{\partial W(\mathbf{C})}{\partial \mathbf{C}} \\ &= 2\left(\frac{\partial W}{\partial I_1}\frac{\partial I_1}{\partial \mathbf{C}} + \frac{\partial W}{\partial I_2}\frac{\partial I_2}{\partial \mathbf{C}} + \frac{\partial W}{\partial I_3}\frac{\partial I_3}{\partial \mathbf{C}}\right)\end{aligned} \tag{A.7}$$

where the derivatives of the principal invariants can be expressed[27, 5] as

$$\frac{\partial I_1}{\partial \mathbf{C}} = \mathbf{I} \text{ and } \frac{\partial I_2}{\partial \mathbf{C}} = I_1\mathbf{I} - \mathbf{C}^T \text{ and } \frac{\partial I_3}{\partial \mathbf{C}} = I_3\mathbf{C}^{-T} \tag{A.8}$$

Using equation A.3, A.4 and A.5, the PK2 stress can be computed for an isotropic hyperelastic material as

$$\begin{aligned}\mathbf{S} &= 2\frac{\partial W}{\partial I_1}\mathbf{I} + 2\left(I_1\mathbf{I} - \mathbf{C}^T\right)\frac{\partial W}{\partial I_2} + 2I_3\frac{\partial W}{\partial I_3}\mathbf{C}^{-T} \\ &= 2\left(\frac{\mu}{2}\right)\mathbf{I} + 2I_3\left(-\frac{\mu}{2I_3} + \frac{\lambda\log J}{2I_3}\right)\mathbf{C}^{-T} \\ &= \mu\left(\mathbf{I} - \mathbf{C}^{-T}\right) + \lambda\log J\mathbf{C}^{-T}\end{aligned}$$

and since the right Cauchy-Green stress tensor is symmetric, the PK2 stress is becomes

$$\mathbf{S} = \mu(\mathbf{I} - \mathbf{C}^{-1}) + \lambda\log J\mathbf{C}^{-1}. \tag{A.9}$$

Using equation A.6, the Cauchy stress[37, 28] in terms of the deformation becomes

$$\begin{aligned}\boldsymbol{\sigma} &= J^{-1}\mathbf{F}\mathbf{S}\mathbf{F}^T \\ &= J^{-1}\mathbf{F}\left[\mu(\mathbf{I} - \mathbf{C}^{-1}) + \lambda\log J\mathbf{C}^{-1}\right]\mathbf{F}^T \\ &= J^{-1}\mathbf{F}\left[\mu(\mathbf{I} - \mathbf{F}^{-1}\mathbf{F}^{-T}) + \lambda\log J\mathbf{F}^{-1}\mathbf{F}^{-T}\right]\mathbf{F}^T \\ &= \lambda J^{-1}\log J\mathbf{I} + J^{-1}\mu(\mathbf{F}\mathbf{F}^T - \mathbf{I})\end{aligned} \tag{A.10}$$

And the nominal stress $\mathbf{P}$ (required in the total Lagrangian framework) is derived from the PK2 stress as

$$\mathbf{P} = \mathbf{S}\mathbf{F}^T \tag{A.11}$$

$$\begin{aligned}&= \left[\lambda\log J\mathbf{F}^{-1}\mathbf{F}^{-T} + \mu(\mathbf{I} - \mathbf{F}^{-1}\mathbf{F}^{-T})\right]\mathbf{F}^T \\ &= \lambda\log J\mathbf{F}^{-1} + \mu\mathbf{F}^{-1}\left(\mathbf{F}\mathbf{F}^T - \mathbf{I}\right)\end{aligned} \tag{A.12}$$

A.3. Small strain approximation

The nominal stress $\mathbf{P}$ and Cauchy stress $\boldsymbol{\sigma}$ derived in the previous section are nonlinear, which makes algebraic derivation complex. Therefore, the constitutive is linearized for small strains. The strain is defined as

$$\boldsymbol{\epsilon} = \operatorname{sym}(\nabla_0 \mathbf{u}) = \frac{1}{2}\left(\nabla_0 \mathbf{u} + (\nabla_0 \mathbf{u})^T\right) \tag{A.13}$$

Let $\boldsymbol{\delta} > 0$, and assume $\nabla_0 \mathbf{u} = \boldsymbol{\delta}$, then the deformation tensor becomes

$$\mathbf{F} = \mathbf{I} + \boldsymbol{\delta}. \tag{A.14}$$

Some expressions required in computing the nominal stress $\mathbf{P}$ follow

$$\begin{aligned}
\mathbf{F}\mathbf{F}^T &= (\mathbf{I} + \boldsymbol{\delta})(\mathbf{I} + \boldsymbol{\delta})^T \\
&= \mathbf{I} + \boldsymbol{\delta} + \boldsymbol{\delta}^T + \mathcal{O}(\boldsymbol{\delta}^2) \\
&\approx \mathbf{I} + 2\boldsymbol{\epsilon} \\
J = \det(\mathbf{F}) &= \det(\mathbf{I} + \boldsymbol{\delta}) = 1 + \operatorname{tr}(\boldsymbol{\delta}) + \mathcal{O}(\boldsymbol{\delta}^2) \\
&\approx 1 + \operatorname{tr}(\boldsymbol{\delta}) = 1 + \operatorname{tr}(\boldsymbol{\epsilon})
\end{aligned}$$

where the last expression comes from the fact that $\operatorname{tr}(A) = \operatorname{tr}(A^T)$.

Recall that the power series of the natural logarithm is

$$\log(1 - x) = -\sum_{1}^{\infty} \frac{x^n}{n!} = -x + \mathcal{O}(x^2)$$

thus

$$\log J \approx \log(1 + \operatorname{tr}(\boldsymbol{\epsilon})) \approx \operatorname{tr}(\boldsymbol{\epsilon}).$$

To approximate the inverse deformation tensor $\mathbf{F}^{-1}$, recall that for a matrix $\mathbf{A}$ holds

$$\mathbf{I} + \mathbf{A}^k = (\mathbf{I} + \mathbf{A})(I - \mathbf{A} - \mathbf{A}^2 - \cdots - \mathbf{A}^k),$$

thus

$$(\mathbf{I} + \mathbf{A})^{-1} = \mathbf{I} - \mathbf{A} + \mathcal{O}(\mathbf{A}^2). \tag{A.15}$$

The linearized constitutive model for a Neo-Hookean material with small strains $\boldsymbol{\epsilon}$ can be approximated as

$$\begin{aligned}
\mathbf{P} &= \lambda \log J \mathbf{F}^{-1} + \mu \mathbf{F}^{-1}\left(\mathbf{F}\mathbf{F}^T - \mathbf{I}\right) \\
&\approx \lambda \operatorname{tr}(\boldsymbol{\epsilon})\left(\mathbf{I} + \boldsymbol{\delta} + \mathcal{O}(\boldsymbol{\delta}^2)\right) + \mu\left(\mathbf{I} + \boldsymbol{\delta} + \mathcal{O}(\boldsymbol{\delta}^2)\right)(2\boldsymbol{\epsilon}) \\
&= \lambda \operatorname{tr}(\boldsymbol{\epsilon})\mathbf{I} + 2\mu\boldsymbol{\epsilon} + \mathcal{O}(\boldsymbol{\delta}^2) \\
&\approx \lambda \operatorname{tr}(\boldsymbol{\epsilon})\mathbf{I} + 2\mu\boldsymbol{\epsilon} && \text{(A.16)} \\
&= \lambda \operatorname{tr}\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right)\mathbf{I} + 2\mu\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right) && \text{(A.17)}
\end{aligned}$$

where equation A.16 is the well-known Hooke's Law for linear elastic materials.

The Cauchy-stress $\boldsymbol{\sigma}$ is then derived from $\mathbf{P}$ as

$$\begin{aligned}
\boldsymbol{\sigma} &= J^{-1}\mathbf{F}\mathbf{P} \\
&\approx \frac{\lambda}{J}\operatorname{tr}(\boldsymbol{\epsilon})\mathbf{F} + \frac{2\mu}{J}\mathbf{F}\boldsymbol{\epsilon} && \text{(A.18)} \\
&= \frac{\lambda}{J}\operatorname{tr}\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right)\mathbf{F} + \frac{2\mu}{J}\mathbf{F}\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right) && \text{(A.19)}
\end{aligned}$$

which is used in MPM application[37].

A.4. Total and updated Lagrangian framework

For simple 1-dimensional cases, it is possible to determine an analytic solution from a given boundary value problem. However, for higher dimensional problems, this task becomes increasingly complex or impossible. Therefore, the method of manufacturing solutions (MMS) is often used in code verification procedures. Before this method can be used, a distinction between the updated and total Lagrangian framework will be made.

In finite element analysis, both the updated and total Lagrangian framework is used. The total Lagrangian framework uses the reference frame or the initial material to compute the internal strains and stresses, while the updated Lagrangian framework uses the current reference frame. The conservation of linear momentum for these frameworks can then be written respectively as[5]

$$\frac{\partial \rho_0 \mathbf{v}}{\partial t} - \nabla_0 \cdot \mathbf{P} - \rho_0 \mathbf{b} = 0, \tag{A.20}$$

$$\frac{\partial \rho \mathbf{v}}{\partial t} - \nabla \cdot \boldsymbol{\sigma} - \rho \mathbf{b} = 0. \tag{A.21}$$

The 0-subscript indicates the initial framework. Since the framework is different, the stress computation varies as well. Here, the updated Lagrangian framework uses the Cauchy stress $\boldsymbol{\sigma}$ while the total Lagrangian framework uses the nominal stress $\mathbf{P}$ with respect to the initial reference frame. While both frameworks can be used in FEA, the Material Point method is only defined in the updated Lagrangian framework.

For the method of manufacturing solutions, it is useful to use the total Lagrangian framework, since only the initial material configuration is known.

B

Method of manufacturing solutions

For this problem, a Neo-Hookean material model is assumed which in the total Lagrangian framework is defined by equation A.17 in combination with the conservation of linear momentum equation A.20:

$$\rho_0 \frac{\partial^2 \mathbf{u}}{\partial t^2} - \nabla_0 \cdot \mathbf{P} - \rho_0 \mathbf{b} = 0,$$
$$\mathbf{P} = \lambda \mathrm{tr}\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right)\mathbf{I} + 2\mu\left(\frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}\right).$$

Note that while the small strain approximation is used here, the regular Neo-Hookean constitutive model of equation A.12 can be used analogous, e.g., in [28].

For this problem, a material plane of length L and height H is considered, with the bottom left corner of the plane at the origin. It is assumed that a force is present which "pulls" the top right corner of the plane along the diagonal. This results in a displacement vector of

$$\mathbf{u}(x, y, t) = \begin{bmatrix} u_e(1 - e^{-\omega_x t}) \cdot x \cdot \left(\frac{y}{H}\right)^2 \\ u_e(1 - e^{-\omega_y t}) \cdot y \cdot \left(\frac{x}{L}\right)^2 \end{bmatrix} \tag{B.1}$$

where x and y are the position on the initial plane. For simplicity, it is assumed that $L = H$ and $\omega_x = \omega_y = \omega$, simplifying the displacement as

$$\mathbf{u}(x, y, t) = \frac{u_e}{L^2}(1 - e^{-\omega t}) \begin{bmatrix} xy^2 \\ yx^2 \end{bmatrix} = T(t) \begin{bmatrix} xy^2 \\ yx^2 \end{bmatrix}, \tag{B.2}$$

where $T(t) = \frac{u_e}{L^2}(1 - e^{-\omega t})$. This results in a displacement gradient of

$$\nabla_0 \mathbf{u} = T(t) \begin{bmatrix} y^2 & 2xy \\ 2xy & x^2 \end{bmatrix} \tag{B.3}$$

and since the displacement gradient is symmetric, this results in

$$\boldsymbol{\epsilon} = \frac{1}{2}\left(\nabla_0 \mathbf{u} + (\nabla_0 \mathbf{u})^T\right) = \frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I} = \nabla_0 \mathbf{u}, \tag{B.4}$$

which results in a nominal stress

$$\mathbf{P} = T(t) \begin{bmatrix} \lambda x^2 + (\lambda + 2\mu)y^2 & 4\mu xy \\ 4\mu xy & (\lambda + 2\mu)x^2 + \lambda y^2 \end{bmatrix} \tag{B.5}$$

and

$$\nabla_0 \cdot \mathbf{P} = (2\lambda + 4\mu) T(t) \begin{bmatrix} x \\ y \end{bmatrix}.$$

The body force which is required by the conservation of linear momentum is then expressed as

$$\begin{aligned}\mathbf{b} &= \frac{\partial^2 \mathbf{u}}{\partial t^2} - \frac{1}{\rho_0}\nabla_0 \cdot \mathbf{P} \\ &= -\frac{u_e}{L^2}\omega^2 e^{-\omega t}\begin{bmatrix} xy^2 \\ x^2 y\end{bmatrix} + \frac{u_e}{L}\frac{2\lambda + 4\mu}{\rho_0}(1 - e^{-\omega t})\begin{bmatrix} x \\ y\end{bmatrix}.\end{aligned}$$

The initial conditions of this problem can then be expressed by

$$\mathbf{u_0}(x, y) = \mathbf{u}(x, y, t = 0) = \frac{u_e}{L^2}\begin{bmatrix} x \cdot y^2 \\ x^2 \cdot y\end{bmatrix} \tag{B.6}$$

$$\mathbf{v_0}(x, y) = \mathbf{v}(x, y, t = 0) = \left.\frac{\partial \mathbf{u}}{\partial t}\right|_{t=0} = \mathbf{0} \tag{B.7}$$

$$\boldsymbol{\sigma}(x, y) = \boldsymbol{\sigma}(x, y, t = 0) = J^{-1}\mathbf{P}\mathbf{F}|_{t=0} \tag{B.8}$$

For the boundary conditions, the boundary is split into 4 segments (the 4 edges of the domain). On the left boundary Γ_{left} $(x = 0)$ and the bottom boundary Γ_{bottom} $(y = 0)$, Dirichlet boundary conditions are applied using the penalty method, using equation 5.6 for the exact value at the boundary. For the top boundary Γ_{top} $(y = H)$ and the right boundary Γ_{right} $(x = L)$, a Neumann boundary conditions is imposed. The force vector imposing the boundary conditions can then be expressed as

$$\begin{aligned}\mathbf{F}_i^{\text{boundary}} &= \mathbf{F}_i^{\text{Dirichlet}} + \mathbf{F}_i^{\text{tract}} \\ &= \int_{F_{\text{left}} \cap F_{\text{bottom}}} \phi_i \epsilon^{-1}(\mathbf{u}^h - \mathbf{u})\mathrm{d}\Gamma + \int_{F_{\text{top}} \cap F_{\text{right}}} \phi_i \mathbf{n} \cdot \boldsymbol{\sigma}\mathrm{d}\Gamma\end{aligned} \tag{B.9}$$

Here, the exact solution for $\mathbf{u}$ and $\boldsymbol{\sigma}$ are known. Because of the nature of this problem, the boundary does move and the normal vector $\mathbf{n}$ is not clearly defined. However, since the exact displacement is known, the normal vector can be expressed. The position at a time t given its initial position (x, y) can be expressed as

$$\tilde{\mathbf{x}}(x, y, t) = \begin{bmatrix} x + u_e(1 - e^{-\omega_x t}) \cdot x \cdot \left(\frac{y}{H}\right)^2 \\ y + u_e(1 - e^{-\omega_y t}) \cdot y \cdot \left(\frac{x}{L}\right)^2\end{bmatrix} \tag{B.10}$$

Bottom boundary, $y = 0$

To get the parametric representation of the boundary at the bottom, let $y = 0$ and $x(p) = Lp, p \in [0, 1]$, thus

$$\tilde{\mathbf{x}}(p, t) = \tilde{\mathbf{x}}(Lp, 0, t) = \begin{bmatrix} Lp \\ 0\end{bmatrix}, \tag{B.11}$$

which result in a tangent function

$$\frac{\mathrm{d}\tilde{\mathbf{x}}}{\mathrm{d}p}(p, t) = \begin{bmatrix} L \\ 0\end{bmatrix} \tag{B.12}$$

and rotating this vector $\frac{\pi}{2}$ and dividing by its length, the outward unit normal vector is given by

$$\hat{\mathbf{n}} = \begin{bmatrix} 0 \\ -1\end{bmatrix} \tag{B.13}$$

Left: $x = 0$

Let $x = 0$ and $y(p) = Hp$, resulting in

$$\tilde{\mathbf{x}}(p, t) = \tilde{\mathbf{x}}(0, Hp, t) = \begin{bmatrix} 0 \\ Hp,\end{bmatrix} \tag{B.14}$$

resulting in a tangent and unit normal vector of

$$\frac{\mathrm{d}\tilde{\mathbf{x}}}{\mathrm{d}p}(p, t) = \begin{bmatrix} 0 \\ H\end{bmatrix} \text{ and } \hat{\mathbf{n}} = \begin{bmatrix} -1 \\ 0\end{bmatrix} \tag{B.15}$$

Right: $x = L$

The parametric representation of the boundary at the righthand side, let $x = L$ and $y(p) = Hp$, resulting in

$$\tilde{\mathbf{x}}(p,t) = \tilde{\mathbf{x}}(L, Hp, t) = \begin{bmatrix} L + u_e(1 - e^{-\omega_x t}) \cdot L \cdot p^2 \\ Hp + u_e(1 - e^{-\omega_y t}) \cdot Hp \end{bmatrix} \tag{B.16}$$

and the tangent vector

$$\frac{\mathrm{d}\tilde{\mathbf{x}}}{\mathrm{d}p}(p,t) = \begin{bmatrix} 2u_e L(1 - e^{-\omega_x t})p \\ H + u_e H(1 - e^{-\omega_y t}) \end{bmatrix} \tag{B.17}$$

and the normal vector

$$\mathbf{n}(p,t) = \begin{bmatrix} H + u_e H(1 - e^{-\omega_y t}) \\ -2u_e L(1 - e^{-\omega_x t})p \end{bmatrix} \tag{B.18}$$

and the unit normal vector

$$\hat{\mathbf{n}}(p,t) = \frac{\mathbf{n}(p,t)}{|\mathbf{n}(p,t)|} \tag{B.19}$$

Assuming $H = L$ and $\omega_x = \omega_y$ and let $T(t) = u_e(1 - e^{-\omega_x t})$, we get

$$\mathbf{n}(p,t) = L \begin{bmatrix} 1 + T(t) \\ -2T(t)p \end{bmatrix} \tag{B.20}$$

and

$$\hat{\mathbf{n}}(p,t) = \frac{1}{\sqrt{1 + 2T(t) + (4p^2 + 1)T^2(t)}} \begin{bmatrix} 1 + T(t) \\ -2T(t)p \end{bmatrix} \tag{B.21}$$

Top: $(y = H)$

The parametric representation of the boundary at the righthand side, let $y = H$ and $x(p) = Lp$, resulting in

$$\tilde{\mathbf{x}}(p,t) = \tilde{\mathbf{x}}(Lp, H, t) = \begin{bmatrix} Lp + u_e L(1 - e^{-\omega_x t})p \\ H + u_e H(1 - e^{-\omega_y t})p^2 \end{bmatrix} \tag{B.22}$$

and the resulting outward unit vector is obtained

$$\hat{\mathbf{n}}(p,t) = \frac{1}{\sqrt{1 + 2T(t) + (4p^2 + 1)T^2(t)}} \begin{bmatrix} -2T(t)p \\ 1 + T(t) \end{bmatrix}. \tag{B.23}$$

C

Spline integration

A spline can be written as a linear combination of n B-splines

$$S(\xi) = \sum_{i=1}^{n} \alpha_i B_{i,p}(\xi). \tag{C.1}$$

The integration of a spline can be obtained using the expression for the derivative, equation 4.3, and the derivation follows from De Boor[7] as

$$\int_{-\xi_0}^{\xi} \sum_{i=1}^{n} \alpha_i B_{i,p}(\tilde{\xi}) \mathrm{d}\tilde{\xi} = \sum_{i=1}^{n} \left[\sum_{j=1}^{i} \alpha_j \frac{\xi_{j+p+1} - \xi_j}{p+1} \right] B_{i,p+1}(\xi)$$

thus, the integration of a B-spline $B_{i,p}$ is obtained by setting the control points to $\alpha_i = 1$ and $\alpha_j = 0, j \neq i$, which results in the integral of a B-spline as

$$\int_{-\xi_0}^{\xi} B_{i,p}(\tilde{\xi}) \mathrm{d}\tilde{\xi} = \frac{\xi_{i+p+1} - \xi_i}{p+1} \sum_{j=i}^{n} B_{j,p+1}(\xi).$$