Durham University

Durham E-Theses

B-spline based Boundary Method for the Material Point Method

YUN BING

How to cite:

BING, YUN (2017) B-spline based Boundary Method for the Material Point Method. Masters thesis, Durham University.

Use policy

The full-text may be used and/or reproduced, and given to third parties in any format or medium, without prior permission or charge, for personal research or study, educational, or not-for-profit purposes provided that:

- a full bibliographic reference is made to the original source
- a https://etheses.durham.ac.uk/id/eprint/12241/ is made to the metadata record in Durham E-Theses
- the full-text is not changed in any way

The full-text must not be sold in any format or medium without the formal permission of the copyright holders.

Please consult the full Durham E-Theses policy for further details.

Academic Support Office, The Palatine Centre, Durham University, Stockton Road, Durham, DH1 3LE
e-mail: e-theses.admin@durham.ac.uk Tel: +44 0191 334 6107
https://etheses.durham.ac.uk

B-spline based Boundary Method for the Material Point Method

Yun Bing

Abstract

Unlike the conventional finite element method, in which the mesh conforms to the material boundary, the material point method (MPM) does not provide a clear interpretation of the boundary. Consequently, difficulties arise when it comes to solving boundary-value problems during MPM simulations, in particular, applying traction (Neumann) and prescribed displacement (inhomogeneous Dirichlet) boundary conditions. However, little attention has been paid to this issue; no literature to date has presented an effective way to model and track boundaries in the MPM. Hence, developing new ways of boundary representation and boundary conditions application in the MPM is the focus of this research.

Formulation of the MPM is firstly presented followed by a review on current approaches to this boundary issue, where B-spline interpolation techniques and an implicit boundary method are identified as the methods to be taken forward. Essential knowledge on B-splines is then discussed. After comparing different B-spline interpolation techniques, a local cubic scheme is selected for boundary representation due to its ability to handle sharp corners and its relatively high computational stability. Next, enforcements of the boundary conditions are discussed. Tractions are applied through direct integration over the B-spline boundary and displacements are prescribed via a B-spline based implicit boundary method. Finally, this boundary method is verified through numerical examples, several of which were not possible with previous MPMs. The novelty of this thesis lies in providing a complete methodology on modelling and tracking the boundaries as well as accurately imposing both Neumann and Dirichlet boundary conditions in the MPM.

Durham University

B-spline based Boundary Method for the Material Point Method

Thesis by

Yun Bing

Submitted as partial consideration towards
the degree of Master of Science

Mechanics Research Group
School of Engineering & Computing Sciences
Durham University
United Kingdom

July 2017

Contents

List of Figures

List of Tables

Declaration

The work in this thesis is based on research carried out in the Mechanics Research Group, School of Engineering and Computing Sciences, Durham University. No part of this report has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Copyright © 2017 by Yun Bing.

"The copyright of this thesis rests with the author. No quotations from it should be published without the author's prior written consent and information derived from it should be acknowledged."

Chapter 1

Introduction

For decades, the well-developed finite element method (FEM) has dominated the realm of computational analysis of structures and solid mechanics. However, the FEM is not without disadvantages. The Lagrangian FEM is unable to model large deformation and fracture problems without going through the expensive processes of re-meshing and mapping of state variables. Whereas the Eulerian FEM, mainly used in fluid dynamics, has difficulties on dealing history-dependent materials. Although the arbitrary Lagrangian-Eulerian method minimises the drawbacks in the pure Lagrangian and the pure Eulerian FEMs, it cannot completely avoid re-meshing and remapping of the state variables [6]. The appearance of meshless methods (MMs, also known as meshfree methods) provides an alternative to the FEM. Early developments within the family of such methods include the smoothed particle hydrodynamics method [7], the Element-Free Galerkin method [8] and the reproducing kernel particle method [9]. Instead of dividing a continuum into finite elements (FEs), MMs use a set of particles to represent the problem domain and each particle is associated with a domain of influence which acts as the means of inter-particle communication. Although the elimination of a Lagrangian mesh allows MMs be able to model problems that are challenging for the FEM, these methods are not without disadvantages. Because the governing equations are solved on the particles, searching of the neighbour particles is essential and high order integration is required to accurately integrate the rational shape functions [10]. An alternative to all the choices mentioned above is the material point method (MPM).

The MPM was developed by Sulsky *et al.* [11, 12] in 1994 as a solid mechanics extension to the fluid implicit particle method [13] which was an advancement of the particle-in-cell method [14]. Interestingly, the name "material point method" was not used until the axisymmetric form of the method was introduced in 1996 [15]. Like meshless methods, the MPM discretises a problem domain by a finite number of particles, or rather material points (MPs), on which the material properties and history dependent variables are prescribed and carried throughout simulations. Further, each MP is assigned with a weight representing the volume under its influence

instead of a physical domain. In addition to these Lagrangian particles, an Eulerian background mesh is employed to solve the governing equations, which fully eliminates the need for neighbourhood searching. Mapping between the MPs and mesh nodes is usually achieved through linear interpolation functions, and the positions of the MPs are used as the integration points within each element. Although there are no restrictions placed on the form of the background mesh, a structured grid with regular elements is often chosen due to its high computational efficiency [16].

The MPM was firstly introduced for dynamic problems in its explicit form with stresses being updated at the beginning of the load steps (update stress first, USF). Although the update stresses last (USL) approach produces less accurate results comparing to the USF approach in the examples shown in [17, 18], the former has been found to provide more realistic results. This is due to the USL approach introducing numerical dissipation into the dynamic solution that damps out unresolved modes [17]. However, only the USF approach has been shown to be energy conservative [17]. More recently, the MPM has also been formulated for both dynamic and quasi-static problems using an implicit time (or pseudo-time) integration scheme, which has proven to be more cost effective and gives higher accuracy for certain problems compared to an explicit formulation [19, 20, 21]. The generalized interpolation material point (GIMP) method [22] was developed to overcome the inherent instability in the MPM which causes stress oscillation when MPs move from one element to another. The GIMP method improves the accuracy of the MPM by assigning a physical domain to each MP and uses interpolation functions that have a higher continuity. These modifications allow the material points to partially maintain the influences on the element that they were originally in. However, the issue of material separation is not eliminated by GIMP which does not consider the change in shapes of the volumes linked to the MPs during deformation. In 2006, Ma attempted to resolve this problem by tracking the velocities and displacements of the corners of each MP domain [23]. Later in 2009, Wallstedt and Guilkey proposed a weighted least square approach together with a marching cube technique to avoid this separation [24]. The most recent variations of the MPM (or GIMP) are the convected particle domain interpolation method (CPDI) [25] and the second-order convected particle domain interpolation method (CPDI2) [4]. Both methods allow deformation of the domains of the MPs, which reduce the instabilities appearing in the MPM and the GIMP method. However, the CPDI2 technique is effectively a dual mesh method with the mesh representing the material point domains being convected through a background mesh. This increases the memory requirement of the method and also introduces additional approximations through the calculation of the basis functions between the overlapping meshes.

As it originated from fluid mechanics, the MPM and its variants are popular choices for fluid-structure analyses. York *et al.* used the MPM for simulations of fluid-membrane interactions [26, 27] and the same problem was studied by Lian *et al.*

using a coupled membrane element and MPM [28]. Additionally, Hamad *et al.* developed a coupled FEM-MPM approach to model the interactions between fluid, solid and geomembranes [29, 1, 30]. The MPM has also been employed to tackle multiphase interaction problems [31, 32]. Another area where the MPM is widely used is the simulations of impact, penetration and explosion problems [33, 34, 35, 36, 37]. In order to minimise the error caused by localised extreme deformations in these problems, Lian *et al.* developed two methods for local refinement within the MPM framework [38, 39]. Other examples of the use of the MPM are for the modelling of granular materials [40, 19, 41], landslides [42, 43, 44, 45] and crack propagations [46, 47].

Despite the developments made to improve the accuracy of the MPM and its broad applications, one severe problem of the method is rarely discussed in the literature, that is the lack of a clear interpretation of the boundary, which leads to the difficulty in solving boundary-value problems. Unlike the traditional FEM, boundaries of a problem in the MPM do not necessarily align with the background mesh. This makes the application of boundary conditions (BCs) rather inaccurate, especially for tractions and prescribed displacements. However, many have managed to avoid this issue by aligning the problem boundaries with the background mesh or modelling problems that only involve body forces. Another workaround is shown in [25]. To ensure a boundary of a rotated bar is fully fixed, an identical bar is added to the other side of the boundary which is subjected to forces with the same magnitude but in the opposite direction. Tricks like this are clearly not a practical solution for enforcing BCs in complicated problems. A few papers in the MPM literature provide ways to improve the accuracy of imposing BCs using weighting parameters [48, 49, 1], but the accuracy of these methods still depends on the element size and mesh refinement around the boundaries is required to maintain sufficient accuracy. Alternatively, the moving mesh concept [2] allows the BCs to be imposed directly in the way as standard FEM, but only if the essential boundary does not change shape [5] and moves in one direction [1]. Another potential solution is the dual-grid extension approach [3]. However, simple 1D tests have shown that this method is sensitive to the location and element size of the boundaries [50] and an improvement to the method has yet to be reported. Although a promising solution to the problem of enforcing Dirichlet BCs in the MPM has been developed in [5] where an implicit boundary method (IBM) [51, 52, 53] is employed, the imposition of Neumann BCs has yet to be fully addressed in the literature.

However, in order to accurately enforce the BCs, tractions (Neumann BCs) and prescribed displacements (inhomogeneous Dirichlet BCs (IDBCs)), boundaries of the problem domain need to be defined. Instead of manually defining a curve as the initial boundary that cannot be tracked once the domain is deformed, the boundary should be reconstructible so that the BCs can be applied through a number of load steps. One approach could be to define the problem domain using the outer layer

of the MPs and fit a curve through these points to achieve a continuous boundary representation. Curve fitting techniques, such as polynomial interpolation and least squares, can often be found in data analysis. For computational geometric representation, splines are employed as the standard means for curves and surfaces fitting in computer aided design (CAD). A simple example of the splines are the (non-rational) B-splines [54] which are piecewise polynomials joined by knots and defined by a set of control points. NURBS (non-uniform rational B-splines), as generalisations of B-splines, are the most popular choice in CAD geometry representation. The reasons behind their popularity include their ability to use a unified mathematical formulation to precisely represent both analytic geometries, especially conics, and free-form curves and surfaces [55]. In addition, they are able to perform not only h-refinement (knot insertion) and p-refinement (order elevation) but also k-refinement (elevating the order of the original curve before inserting a new knot) [56]. However, NURBS also have some shortcomings, one of which is the difficulty in handling surface intersections [55]. This led to the development of T-splines (non-uniform B-spline surfaces with T-junctions) [57] which allow control points to be inserted or removed without disturbing other control points. Nonetheless, the simplest form, B-splines, are considered to be sufficient for the purpose of boundary representation in the MPM; further justification of this choice can be found in Chapter 3.

The research presented in this thesis provides a methodology on the boundary representation and boundary condition imposition in the MPM. Firstly, the essential equations required for the MPM are detailed in Chapter 2. Then, a review of the current approaches on applying BCs in the MPM and MMs and some unique boundary tracking techniques is presented in Chapter 3. In the same chapter, B-splines are identified as the means for boundary representation. Following this the background knowledge of B-splines are discussed in Chapter 4 and methods of imposing BCs are developed in Chapter 5. In Chapter 6, a number of static equilibrium problems are used to validate the suggested methods. Finally, conclusions are drawn in Chapter 7.

Chapter 2

Material Point Method

The philosophy behind the MPM is that the problem domain is discretised by a set of MPs that carry all the material properties and a "stationary" background mesh is generated to solve the equilibrium equations. At the end of each load step, the positions of the MPs are updated by the deformed background mesh which is reset for the next load step leaving the MPs at their new positions. An illustration of this process is shown in Figure 2.1.

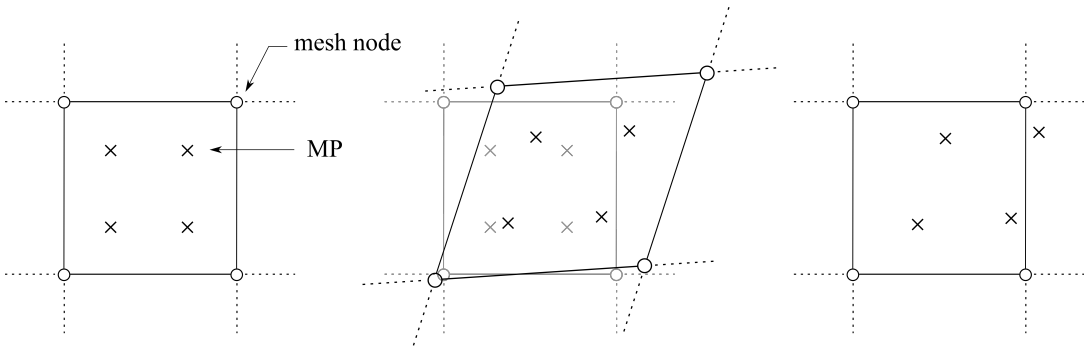


Figure 2.1: An illustration of the MPM simulation procedure: initialisation (left), deformation (middle) and reset mesh with updated MPs position (right).

As the MPM has the same underlying structure as the FEM, a derivation of the standard FEM is firstly presented in Section 2.1. The differences between the MPM and the standard FEM are then highlighted in Section 2.2 along with the implementation of the MPM.

2.1 Derivation of the standard FEM

In static solid mechanics, we have the strong form statement of equilibrium

$$\nabla \cdot [\sigma] + \{f^b\} = 0, \tag{2.1}$$

where $\nabla\cdot$ is the divergence, $[\sigma]$[i] is the Cauchy stress tensor and $\{f^b\}$[ii] is the body forces, which can also be expressed in component form as

$$\frac{\partial\sigma_{xx}}{\partial x}+\frac{\partial\sigma_{yx}}{\partial y}+\frac{\partial\sigma_{zx}}{\partial z}+f_x^b=0, \tag{2.2}$$

$$\frac{\partial\sigma_{xy}}{\partial x}+\frac{\partial\sigma_{yy}}{\partial y}+\frac{\partial\sigma_{zy}}{\partial z}+f_y^b=0 \quad \text{and} \tag{2.3}$$

$$\frac{\partial\sigma_{xz}}{\partial x}+\frac{\partial\sigma_{yz}}{\partial y}+\frac{\partial\sigma_{zz}}{\partial z}+f_z^b=0. \tag{2.4}$$

Since the Cauchy stress tensor is symmetric, (2.2–2.4) can be expressed in Voigt notation as

$$\begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial z} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{xy} \\ \sigma_{yz} \\ \sigma_{zx} \end{Bmatrix} + \begin{Bmatrix} f_x^b \\ f_y^b \\ f_z^b \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix}, \tag{2.5}$$

or more compactly as

$$[L]^T\{\sigma\}+\{f^b\}=0, \tag{2.6}$$

where $[L]$ is a differential operator whose definition is self evident.

For a linear-elastic body, the stresses $\{\sigma\}$ and the strains $\{\varepsilon\}$ (the six-component engineering strains) have the following relationship

$$\{\sigma\}=[D^e]\{\varepsilon\}, \tag{2.7}$$

where $[D^e]$ is the linear elastic stiffness matrix

$$[D^e]=\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} (1-\nu) & \nu & \nu & 0 & 0 & 0 \\ \nu & (1-\nu) & \nu & 0 & 0 & 0 \\ \nu & \nu & (1-\nu) & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{(1-2\nu)}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{(1-2\nu)}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{(1-2\nu)}{2} \end{bmatrix}, \tag{2.8}$$

with E and ν as the Young's modulus and Poisson's ratio respectively. For materials that includes non-linearity, (2.7) is no longer valid and other measures are needed to relate strains and stresses. However, only linear materials are considered here as the

[i] The square brackets are used to indicate matrices throughout this thesis.
[ii] The curly brackets are used to indicate column vectors throughout this thesis.

focus of this thesis is boundary representation and boundary condition imposition. With infinitesimal strains and small deformations, it is possible to express $\{\varepsilon\}$ as the gradient of displacement $\{u\}$, that is

$$\{\varepsilon\} = [L]\{u\}. \tag{2.9}$$

Displacements within a finite element (FE) can be calculated from the nodal displacements, $\{d^e\}$, of the element together with the standard FE shape functions, $[M]$, as

$$\{u\} = [M]\{d^e\}. \tag{2.10}$$

Therefore, strains within an element can be computed in terms of the nodal displacements as

$$\{\varepsilon\} = [L][M]\{d^e\} = [B]\{d^e\}, \tag{2.11}$$

where

$$[B] = \begin{bmatrix} \frac{\partial M_1}{\partial x} & 0 & 0 & \cdots & \frac{\partial M_{n_{en}}}{\partial x} & 0 & 0 \\ 0 & \frac{\partial M_1}{\partial y} & 0 & \cdots & 0 & \frac{\partial M_{n_{en}}}{\partial y} & 0 \\ 0 & 0 & \frac{\partial M_1}{\partial z} & \cdots & 0 & 0 & \frac{\partial M_{n_{en}}}{\partial z} \\ \frac{\partial M_1}{\partial y} & \frac{\partial M_1}{\partial x} & 0 & \cdots & \frac{\partial M_{n_{en}}}{\partial y} & \frac{\partial M_{n_{en}}}{\partial x} & 0 \\ 0 & \frac{\partial M_1}{\partial z} & \frac{\partial M_1}{\partial y} & \cdots & 0 & \frac{\partial M_{n_{en}}}{\partial z} & \frac{\partial M_{n_{en}}}{\partial y} \\ \frac{\partial M_1}{\partial z} & 0 & \frac{\partial M_1}{\partial x} & \cdots & \frac{\partial M_{n_{en}}}{\partial z} & 0 & \frac{\partial M_{n_{en}}}{\partial x} \end{bmatrix} \tag{2.12}$$

is the 6 by $3n_{en}$ strain-displacement matrix for 3-dimensional problems and n_{en} is the number of nodes per element.

There are three ways to formulate the weak form statement of equilibrium: virtual work, the variational method, and the weighted residual method; here we employ the weighted residual approach. Consider the initial strong form stated in (2.1), by pre-multiplying a weighting function $\{c\}^T$ and integrating over the volume of an element Ω, we have

$$\int_\Omega \{c\}^T \nabla \cdot [\sigma] \, \mathrm{d}\Omega + \int_\Omega \{c\}^T \{f^b\} \, \mathrm{d}\Omega = 0. \tag{2.13}$$

For the first term of (2.13), consider the following expansion with ∇ as the gradient and : being the double contraction,

$$\nabla \cdot (\{c\}^T[\sigma]) = (\nabla\{c\}^T) : [\sigma] + \{c\}^T(\nabla \cdot [\sigma]), \tag{2.14}$$

and rearrange to provide

$$\{c\}^T(\nabla \cdot [\sigma]) = \nabla \cdot (\{c\}^T[\sigma]) - (\nabla\{c\}^T) : [\sigma]. \tag{2.15}$$

Substituting (2.15) to (2.13) gives

$$\int_\Omega \nabla \cdot (\{c\}^T[\sigma])\,\mathrm{d}\Omega - \int_\Omega (\nabla\{c\}^T) : [\sigma]\,\mathrm{d}\Omega + \int_\Omega \{c\}^T\{f^b\}\,\mathrm{d}\Omega = 0. \tag{2.16}$$

Apply Gauss's theorem to the first term of (2.16) and substitute ∇ with the differential operator $[L]$, we arrive at the weak form statement of equilibrium, i.e.

$$\int_{\partial\Omega} (\{c\}^T[\sigma])\{\hat{n}\}\,\mathrm{d}S - \int_\Omega ([L]\{c\})^T\{\sigma\}\,\mathrm{d}\Omega + \int_\Omega \{c\}^T\{f^b\}\,\mathrm{d}\Omega = 0, \tag{2.17}$$

where $\{\hat{n}\}$ is the outward unit normal and $\partial\Omega$ represents the domain boundary.

The Galerkin approach of the weighted residual method constructs the weighting function by using the same FE shape functions that were used for mapping the nodal values, i.e. $\{c\} = [M]\{a\}$, where $\{a\}$ is some arbitrary nodal values assumed to be constant. Substituting for $\{c\}$ and $\{\sigma\}$ in (2.17), eliminating the arbitrary vector $\{a\}$ and rearranging gives

$$\int_{\partial\Omega} [M]^T[\sigma]\{\hat{n}\}\,\mathrm{d}S - \int_\Omega ([L][M])^T[D^e][L][M]\{d^e\}\,\mathrm{d}\Omega + \int_\Omega [M]^T\{f^b\}\,\mathrm{d}\Omega = \{0\}. \tag{2.18}$$

As the traction $\{t\}$ defined over $\partial\Omega$ can be expressed as $[\sigma]\{\hat{n}\}$ and the strain-displacement matrix $[B] = [L][M]$, (2.18) is reduced to

$$\int_\Omega [B]^T[D^e][B]\,\mathrm{d}\Omega\,\{d^e\} = \int_{\partial\Omega} [M]^T\{t\}\,\mathrm{d}S + \int_\Omega [M]^T\{f^b\}\,\mathrm{d}\Omega. \tag{2.19}$$

This can be written more concisely as

$$[k^e]\{d^e\} = \{f^{ext}\}, \tag{2.20}$$

where $[k^e]$ is the element stiffness matrix and $\{f^{ext}\}$ contains the external traction and body forces allocated at the element nodes. To solve the displacements for the whole structure under analysis, the element stiffness matrix and the element nodal force vector are later assembled into the structure stiffness matrix $[K]$ and the structure force vector $\{f\}$ respectively.

2.2 Implementation of the MPM

An implicit FEM implementation is adopted for the MPM formulation in this thesis. Although the majority of MPM implementations use an explicit approach to solve the governing equations, the implicit approach has the benefit that it allows larger load steps to be used in analyses, which reduces the number of load steps required. When choosing the load steps appropriately to analyses static problems

(which are the focuses of this thesis), an implicit approach can significantly reduce the computational cost [58]. Additionally, as mentioned in the introduction, another advantage of the implicit MPM is that it improves the accuracy of the solutions for certain problems [19, 20, 21].

Problem initialisation

Firstly, a background mesh as used in the FEM is generated at the start of the simulation. However, this mesh covers not only the problem domain but also extends to where the material may deform into. There are no restrictions on the choice of mesh shapes, but uniform quadrilateral or hexahedral elements are often chosen to ease the computation. The problem domain is then discretised by MPs. These MPs store and carry the state variables and material properties during the simulation. Additionally, each MP is assigned a weight according to its initial position inside the element and the value of the weight stays the same throughout the simulation in the standard MPM. This weight is equivalent to the local volume of the domain associated with the MP. It is convenient to use Gauss point positions according to the background mesh as the initial positions of the MPs because the weights of the MPs, in this case, are the same as the weights of the corresponding Gauss points. Finally, essential boundary conditions are enforced directly on the appropriate nodes in the background mesh, provided that the edges of the background mesh are coincident with the desired location of the boundaries.

Identifying the positions of the material points

At the start of each load step, it is necessary to identify in which element each MP is located. This is achieved by comparing the MPs' global coordinates $\{x_{mp}\}$ with the grid nodal coordinates $\{x\}$ in every dimension. This process is straightforward and relatively inexpensive when the mesh is aligned with the Cartesian coordinate system. Then, the MPs' local coordinates (ξ_{mp}, η_{mp}, ζ_{mp}) within each element are computed. Initially the local coordinates for each MP are set to be zeros, and the corresponding shape functions are calculated. The estimated global coordinates of the MP (found by multiplying the element's nodal coordinates by the values of the shape functions at the MP) are compared to its actual global coordinates. The correct local coordinates of the MP are then solved using a Newton-Raphson (NR) iteration. Note that for linear elements this iterative process converges in a single step.

Stiffness matrix and external force vector

The calculation of the stiffness matrix is similar to that used in the standard FEM where Gauss-Legendre quadrature is employed, except that MPs, regardless of their

positions inside the element of interest, along with their weights are used directly to perform the numerical integration.

At the beginning of the element loop, MPs resided in the current element are identified. Recall that from (2.19)

$$[k^e] = \int_\Omega [B]^T [D^e][B] \, \mathrm{d}\Omega, \tag{2.21}$$

which, under the framework of numerical quadrature, is determined using

$$[k^e] \cong \sum_{i=1}^{n_{mp}} [B]_i^T [D^e][B]_i w_i \det([J]_i), \tag{2.22}$$

where n_{mp} is the number of MPs inside the element i, w_i is the weight associated with MP i, and $\det([J]_i)$ is the determinant of the Jacobian for MP i. The calculation presented in (2.22) uses the local volumes of the domain associated with the MPs as the weights. Alternatively, the global volumes can also be used as the weights. In this case, the determinant of the Jacobian, $\det([J])$, can be removed from the stiffness calculation as it is constant throughout a regular mesh. As for when local volumes are used, $[J]$ can be calculated once. To obtain $[J]$, the local coordinates of the MPs and its shape functions are computed followed by calculating the derivatives of the shape functions with respect to the local coordinates. The Jacobian in 3D problems takes the following form

$$[J] = \begin{bmatrix} \frac{\partial x_{mp}}{\partial \xi_{mp}} & \frac{\partial y_{mp}}{\partial \xi_{mp}} & \frac{\partial z_{mp}}{\partial \xi_{mp}} \\ \frac{\partial x_{mp}}{\partial \eta_{mp}} & \frac{\partial y_{mp}}{\partial \eta_{mp}} & \frac{\partial z_{mp}}{\partial \eta_{mp}} \\ \frac{\partial x_{mp}}{\partial \zeta_{mp}} & \frac{\partial y_{mp}}{\partial \zeta_{mp}} & \frac{\partial z_{mp}}{\partial \zeta_{mp}} \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial M_1}{\partial \xi_{mp}} & \frac{\partial M_2}{\partial \xi_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial \xi_{mp}} \\ \frac{\partial M_1}{\partial \eta_{mp}} & \frac{\partial M_2}{\partial \eta_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial \eta_{mp}} \\ \frac{\partial M_1}{\partial \zeta_{mp}} & \frac{\partial M_2}{\partial \zeta_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial \zeta_{mp}} \end{bmatrix}}_{[M,\, \xi_{mp}]} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_{n_{en}} & y_{n_{en}} & z_{n_{en}} \end{bmatrix}. \tag{2.23}$$

The inverse of $[J]$ is pre-multiplied to $[M,\, _{\xi_{mp}}]$ to give the derivatives of the shape functions with respect to the global nodal coordinates ($[M,\, _{x_{mp}}]$), that is

$$\underbrace{\begin{bmatrix} \frac{\partial M_1}{\partial x_{mp}} & \frac{\partial M_2}{\partial x_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial x_{mp}} \\ \frac{\partial M_1}{\partial y_{mp}} & \frac{\partial M_2}{\partial y_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial y_{mp}} \\ \frac{\partial M_1}{\partial z_{mp}} & \frac{\partial M_2}{\partial z_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial z_{mp}} \end{bmatrix}}_{[M,\, x_{mp}]} = [J]^{-1} \begin{bmatrix} \frac{\partial M_1}{\partial \xi_{mp}} & \frac{\partial M_2}{\partial \xi_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial \xi_{mp}} \\ \frac{\partial M_1}{\partial \eta_{mp}} & \frac{\partial M_2}{\partial \eta_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial \eta_{mp}} \\ \frac{\partial M_1}{\partial \zeta_{mp}} & \frac{\partial M_2}{\partial \zeta_{mp}} & \cdots & \frac{\partial M_{n_{en}}}{\partial \zeta_{mp}} \end{bmatrix}. \tag{2.24}$$

$[M,\ _{x_{mp}}]$ is then used to form $[B]$. Finally, the structure stiffness matrix $[K]$ is assembled from the element stiffness matrices according to the global node indexing of each entry in $[k^e]$.

Although the external forces $\{f^{ext}\}$ can be applied directly onto the points, these forces need to be mapped to the grid nodes before the calculation is performed since the governing equations are solved on the background mesh. Note that $\{f^{ext}\}$ derived from the strong form includes both tractions and body forces; however, as discussed in Chapter 3, there is no generalised effective way to apply tractions in the MPM. Therefore, only body forces are considered at this stage. The mapping is achieved through the following procedure

$$\{f^{ext}\} = \sum_{i=1}^{n_{mp}} [M_i]^T \{f^b_{mp_i}\}, \tag{2.25}$$

where $\{f^b_{mp_i}\}$ is the body force assigned to MP i, which is calculated according to with the assumption that z is vertical and that gravity is the only body force

$$\{f^b_{mp_i}\} = V_{mp_i}\rho \begin{Bmatrix} 0 \\ 0 \\ -g \end{Bmatrix} = w_i \det([J]_i)\rho \begin{Bmatrix} 0 \\ 0 \\ -g \end{Bmatrix} \tag{2.26}$$

with V_{mp_i} being the global volume associated with the MP of interest, ρ being the material density and g being the gravitational constant; $[M_i]$ contains the standard FE shape functions of MP i evaluated according to its local coordinates. The size of $\{f^b_{mp_i}\}$ and $\{f^{ext}\}$ are 3 by 1 and $3n_{en}$ by 1 respectively. Similar to the assembling of $[K]$, the structure external force vector $\{f\}$ is formed according to the global degree of freedom of each entry of $\{f^{ext}\}$.

Displacement calculation and stress recovery

After computing the structure stiffness matrix and the structure external force vector, we have the following linear system

$$[K]\{d\} = \{f\}, \tag{2.27}$$

to solve for the structure displacements $\{d\}$. (2.27) is firstly rearranged according to whether the component is linked to prescribed (with subscript $_r$) or free/unknown (with subscript $_f$) displacement degrees of freedom,

$$\begin{Bmatrix} \{f_f\} \\ \{f_r\} \end{Bmatrix} = \begin{bmatrix} [K_{ff}] & [K_{fr}] \\ [K_{rf}] & [K_{rr}] \end{bmatrix} \begin{Bmatrix} \{d_f\} \\ \{d_r\} \end{Bmatrix}. \tag{2.28}$$

As the reactions $\{f_r\}$ are also unknowns in the system, the first row in (2.28) is considered to obtain a general solution of $\{d_f\}$.

$$\{f_f\} = [K_{ff}]\{d_f\} + [K_{fr}]\{d_r\}, \tag{2.29}$$

where $\{f_f\}$ and $\{d_r\}$ are known, and the structure stiffness matrix can be calculated as described above. If all degrees of freedom in $\{d_r\}$ are constrained to be zero, entries associated with $\{d_r\}$ can then be eliminated to reduce the size of calculation.

After evaluating all the nodal displacements, the strain increments at MPs $\{\Delta\varepsilon\}$ resulting from the current load step can be computed according to (2.11) followed by the stress increments at MPs $\{\Delta\sigma\}$, recovered through (2.7).

Variable updating and mesh resetting

By knowing the displacements at every element's nodes for the current load step, it is possible to update the positions of the MPs. For particle i within element e, its displacements are found by mapping the nodal displacement $\{d^e\}$ to the particle through

$$\{\Delta u_{mp}\} = [M_i]\{d^e\}. \tag{2.30}$$

Global coordinates of these particles are then updated by $\{\Delta u_{mp}\}$

$$\{x_{mp}\}_{n+1} = \{x_{mp}\}_n + \{\Delta u_{mp}\}. \tag{2.31}$$

The strains and the stresses are updated in the same fashion at the end of the load step by using the increments calculated at the end of last section, that is

$$\{\varepsilon_{n+1}\} = \{\varepsilon_n\} + \{\Delta\varepsilon\} \quad \text{and} \quad \{\sigma_{n+1}\} = \{\sigma_n\} + \{\Delta\sigma\}. \tag{2.32}$$

In order to have a stationary background mesh throughout the simulation, the positions of the mesh nodes are reset at the end of each load step. In other words, there is no need to update the mesh nodal positions. This leaves the deformed problem domain being represented by the new positions of the MPs and the original background mesh. A code structure of the material point algorithm is presented in Figure 2.2.

2.3 Chapter review

In this chapter, the standard FEM was derived from the strong form of equilibrium to the weak form of equilibrium as the standard MPM has the same underlying structure as the FEM. Implementation procedures of the MPM were discussed; an implicit implementation of the standard MPM is adopted in this thesis. The next

chapter will review the current approaches on imposing BCs in the MPM and MMs and some unique boundary tracking techniques.

1. Input: $\{f^b_{mp}\}$, mesh information (nodal positions $\{x\}$ and connectivity), $\{x_{mp}\}$
2. Constant: $[D^e]$, w_i

   FOR loop over load step

   - Identify which element each material point is in

     FOR loop over elements

     - Identify MPs that are contained in the current element

       FOR loop over material points

       - Compute the local coordinates of the current MP, $(\xi_{mp_i}, \eta_{mp_i}, \zeta_{mp_i})$
       - Compute the MP's shape functions $[M]$ and the local derivatives $[M,\, _{\xi_{mp_i}}]$
       - Compute $[J]$ (2.23)
       - Compute shape function global derivatives $[M,\, _{x_{mp}}]$ (2.24) and form $[B]$ (2.12)
       - Compute $[k^e]$ (2.22)
       - Compute $\{f^b_{mp_i}\}$ (2.26) and form $\{f^{ext}\}$ (2.25)

       END

     - Store $[k^e]$ in compressed column storage or compressed row storage

     END

   - Assemble $[K]$
   - Solve for $\{d\}$ (2.29)

     FOR loop over material points

     - Compute displacements of each material point $\{\Delta u_{mp}\}$ (2.31)
     - Compute $\{\Delta\, \varepsilon\}$ (2.11) and $\{\Delta\, \sigma\}$ (2.7)

     END

   - Update positions of MPs $\{x_{mp}\}_{n+1}$ (2.31), strains $\{\varepsilon_{n+1}\}$ $(2.32)_1$, and stress $\{\sigma_{n+1}\}$ $(2.32)_2$

   END
3. Output: $\{x_{mp}\}$, $\{\varepsilon\}$, and $\{\sigma\}$

Figure 2.2: Algorithm structure of the MPM.

This is an intended blank page.

Chapter 3

Boundaries in Numerical Analysis

The lack of boundary representation in the standard MPM has limited the application of boundary conditions and problems that the MPM can be applied to. That is, the vast majority of MPM simulations presented in the available literature involve body forces and domain boundaries coincident with the background grid. The standard treatment of enforcing essential BCs is the same as that in the traditional FEM due to the fact that the governing equations of the MPM are calculated on the background mesh. However, the boundaries of the problem domain do not necessarily align with the background mesh. Although one can always adjust the position of the problem domain to achieve a boundary-mesh alignment, this is not convenient or even possible if a regular background mesh is used especially when non-zero essential BCs are applied through multiple load steps. While zero natural boundary conditions are automatically satisfied by the solution of the governing equations, non-zero ones, tractions in particular, are hard to be imposed accurately. The most common way to introduce tractions into the system is via boundary particles which normally are the outer layer of the material points. Prescribed tractions are carried by the boundary particles, and then mapped onto the nodes of elements that contain these particles through shape functions. The downside of this approach is that tractions are applied not on a surface but across a boundary band, which increases the thickness of the actual boundary that could increase numerical errors.

Similar issues also exist in the MMs and FEMs which use a non-conforming mesh. One way to apply the essential BCs is by modifying the Galerkin weak form to indirectly constrain the problem domain, i.e. by the use of a Lagrange multiplier, penalty or Nitsche method. However, the introduction of the Lagrange multiplier makes the stiffness matrix no longer positive definite; whereas the accuracy of the penalty method depends on the value of the penalty parameter which is required to be large enough to enforce the BC [59]. However, the larger the parameter is, the more ill-conditioned the stiffness matrix becomes. Although Nitsche's method stabilises the system by adding additional terms, the derivation of the weak form is unique to each problem. Detailed reviews and numerical comparisons of these

approaches can be found in [59]. Other methods to impose essential BCs include coupling the MMs with FEs and weighted interpolation methods, which will be explained later in Section 3.3.

In this chapter, existing solutions to the problem of enforcing boundary conditions in the MPM are reviewed. Additionally, BC imposition developed for MMs and non-conforming FEMs which are potentially applicable under the MPM framework are also presented. Finally, boundary representation techniques are investigated.

3.1 Boundary layer

In 2002, Chen presented the concept of a boundary layer to prescribe non-zero traction boundary conditions in a 2D explicit MPM by specifying a mass density normalised traction vector $\{t^s\}$ [48, 49]. The boundary layer is a collection of elements that contain the boundary particles. The discrete specific traction at an element node i is given by

$$\{t_{node_i}\} = h^{-1} \sum_{mp=1}^{n_{mp}} m_{mp} \{t^s_{mp}\} M_i \left(\{x_{mp}\}\right), \tag{3.1}$$

where n_{mp} is the number of material points inside the boundary element, m_{mp} is the mass of the MP of interest, h is the thickness of the boundary layer, $M_i\left(\{x_{mp}\}\right)$ is the standard nodal basis functions with $\{x_{mp}\}$ being the current position of the MP of interest, and $\{t^s_{mp}\}$ is equal to the value of $\{t^s\}$ on the MP at the current time. If an element contains both the boundary and interior particles, the interior particles will temporarily become boundary particles and the element itself will serve as the boundary layer. Each particle's contribution to the surrounding nodes is weighted by the reciprocal of the boundary layer thickness, and this seems to have a positive influence on the accuracy. Even so, the author still suggests using refinement methods that allow small elements to contain boundary particles only to reduce numerical errors.

However, there has been little validation about how efficient the boundary layer approach is, despite that it has been used as a default treatment in some studies. For example, Ma and Zhang have used the approach for impact and explosion problems [60]. Moreover, Chen claims in [49] that this boundary treatment allows the tracking of moving boundaries or material interfaces to be realised easily. However, apart from using a boundary element's thickness as a weighting parameter and identifying all material points inside a boundary element as the boundary particles, this method is the same as the standard approach using the outer layer of MPs to carry tractions. This realisation of boundary tracking therefore remains unclear. Unfortunately, the authors do not provide any detailed explanations.

3.2 Surface discretisation and prescribed particles

In 2014, Hamad presented a new approach of representing boundaries and implementing boundary conditions in the explicit MPM [1]. Two types of MPs were added under the MPM framework in this work, one of which are boundary particles acquired from surface discretisation, and the other are "prescribed particles" where non-zero kinematic boundary conditions are applied.

3.2.1 Surface discretisation

An irregular triangular mesh was used in [1] to form a surface discretisation of the problem domain and boundary particles are generated within each element (see Figure 3.1), because tracking the deformed surface with a mesh is convenient to calculate the unit normal for applying traction BCs. Since this surface mesh is separated from the interior MPM discretisation, this mesh can be much finer. It should be noted that the boundary particles can participate in the momentum equation with very little mass, or they can be specified to follow the deformation of the regular particles.

In pre-processing stage, the traction is assumed to be fully integrated and discretised on the boundary nodes. The mapping, shown in Figure 3.1, of the force from the nodes to the boundary particles then follows as

$$\{f_b^{trc}\} = \sum_{i=1}^{n_t} M_i \{\tilde{t}_i\} \frac{\Gamma_b}{\Gamma_i}, \tag{3.2}$$

where $\{f_b^{trc}\}$ is the surface traction of particle b, M_i is the value of shape function i at particle b, n_t is the number of nodes per face and $\{\tilde{t}_i\}$ is the nodal traction force. Moreover, this mapping is weighted by the area associated with the corresponding boundary particle (Γ_b) and the nodal corresponding area (Γ_i). The latter is calculated as

$$\Gamma_i = \sum_{j=1}^{n_{tri}} \frac{A_{tri}^j}{3}, \tag{3.3}$$

where Γ_i is the area of node i, n_{tri} is the number of triangles around the node and A_{tri}^j is the area of triangle j. In [1], the boundary particles were placed at the same locations as the nodes of the surface mesh, which in that case means that Γ_b is equal to Γ_i.

As the boundary particles lie on the surface of the mesh, the shape functions are reduced to two-dimensional interpolation functions. The surface traction ($\{f^{trc}\}$) can then be calculated directly on the material points according to

$$\{f_b^{trc}\} = \{t_b\}\Gamma_b, \tag{3.4}$$

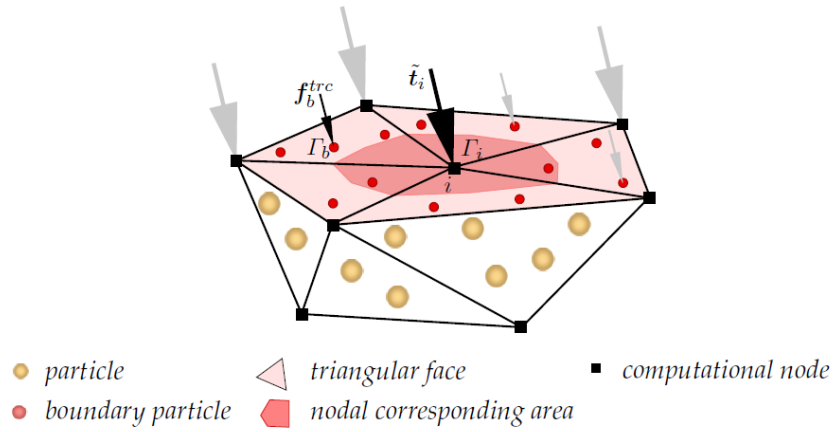


Figure 3.1: Mapping forces from nodes to boundary particles (taken from Hamad [1]).

where $\{t_b\}$ is the continuous traction that is approximated in a stepwise fashion over the domain of the boundary particle b. Then the nodal traction ($\{t_{node}\}$) is obtained through

$$\{t_{node}\} = \sum_{b=1}^{n_b} [M]^T \{f_b^{trc}\}, \tag{3.5}$$

where n_b is the number of particles where tractions are prescribed.

This surface discretisation approach is a rather intriguing concept for boundary tracking and the enforcement of traction BCs. However, this approach eliminates one of the advantages of the MPM which is that the material deformation is decoupled from the computational mesh during simulations. The independent surface mesh not only makes the analysis more complicated, but its ability to handle extreme deformations is also in doubt. Mesh distortion of the surface mesh can occur as it would not be reset after each load step, instead, the surface area over each boundary particle requires constant updates.

3.2.2 Prescribed particles

Prescribed particles (solid grey dots in Figure 3.2) are introduced as an additional set of particles to carry the prescribed displacement or velocity. The velocity, $\bar{v}_{pp}$, of the prescribed particle pp is assigned at the beginning of the time step, and then the velocity is mapped to the computational nodes of the element that contains these particles by a weighted function

$$\bar{v}_i = \frac{\sum_{pp} M_i w_{pp} \bar{v}_{pp}}{\sum_{pp} M_i w_{pp}}, \tag{3.6}$$

where $\bar{v}_i$ is the prescribed velocity at node i, and w_{pp} is a mass or volume dependent property.

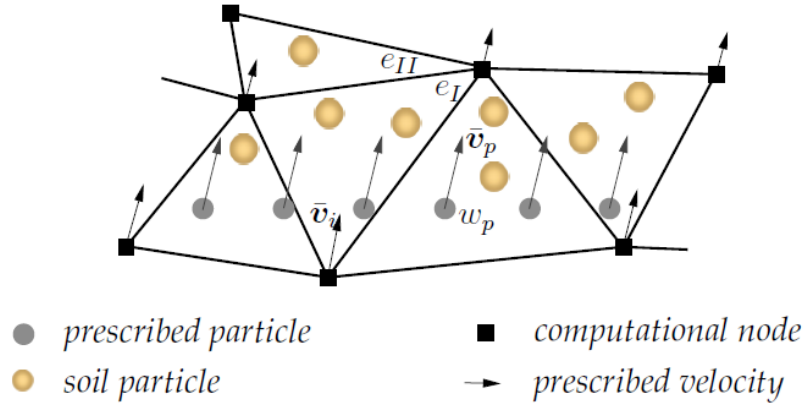


Figure 3.2: Prescribed particles and the prescribed velocity (taken from Hamad [1]).

Through the information given above, one can see that the prescribed particles are analogous to the boundary particles in the standard treatment. Thus, this method also exhibits numerical errors and its accuracy is constrained by the size of an element [1]. Stress oscillations along the layer that contains prescribed particles are reported in [1] during simulations. Unsurprisingly, the author suggests refining meshes for greater accuracy.

3.3 Weighted interpolation methods

In contrast to the weak imposition of Dirichlet BCs (DBCs) discussed at the beginning of this chapter, weighted interpolation methods involve direct modification of the problem solution, i.e. that underlying partial differential equations. There are three approaches in the literature in this category, which are the weighted extended B-spline method (Web-method) [61, 62, 63] the implicit boundary method (IBM) [51, 52, 53, 5] and the weighted finite cell method (weighted FCM) [64, 65]. To highlight the differences between these methods, the following FEM trial solution for an elastostatic problem will be modified

$$\{u\} = [M]\{d^e\}, \tag{3.7}$$

where $\{u\}$ is the displacement within a FE, $[M]$ contains the standard FE shape functions and $\{d^e\}$ are nodal displacements associated with the element.

Under the FE framework, the Web-method defines a new FE subspace on a tensor product grid with the basis functions being weighted extended B-splines B_{web} [63], that is

$$B_{web_i} = w \sum_k e_{i,k} b_k, \tag{3.8}$$

where i indicates a grid node, w is a positive weight function, b_k is a few neighbouring B-splines around i and $k \in \mathbb{Z}$, and $e_{i,k}$ is the extension coefficient for grid node i

that exists as a stabilisation factor [63]. Derivations of the extension coefficient $e_{i,k}$ can be found in [61]. Homogeneous Dirichlet BCs (HDBCs) are imposed by wise choices of w that make B_{web_i} vanish on the boundary of interest. According to [61], R-functions[i] [66] are the most suitable functions for constructing w. Therefore, the Web-method transforms (3.7) into

$$\{u\} = [B_{web}]\{d^e\}. \tag{3.9}$$

By using the Web-splines as the shape functions of the solution structure in the formation of the Galerkin weak form, HDBCs can be satisfied without mesh generation. However, this method is limited to the imposition of HDBCs.

While the Web-method applies HDBCs by modifying the FE shape functions, the IBM does not require this change. Implementation of fixed DBCs with problem boundaries parallel to one of the coordinates has been demonstrated in [51, 52, 53]. This approach has been extended to impose inhomogeneous Dirichlet BCs (IDBCs) on inclined boundaries and has been adopted to the MPM by Cortis *et al.* [5]. The IBM uses implicit equations to modify the boundary value problem, based on a method originally proposed by Kantorovich and Krylov [67]. The so-called Dirichlet functions (D-functions), implicit equations approximating step functions, serve as weighting functions in the trial solution. In the IBM (3.7) is expressed as

$$\{u\} = [D][M]\{d^e\} + \{u^a\}, \tag{3.10}$$

where $[D]$ contains the D-functions and $\{u^a\} = [M]\{u^e\}$ with $\{u^e\}$ being the prescribed nodal displacements. The D-functions are constructed such that they vanish at the degrees of freedoms where non-zero displacements are prescribed, and rise to unity when reaching the problem domain after crossing a thin band δ. The accuracy of the IBM depends on the choice of D-functions and, more importantly, the value of δ which typically needs to be smaller than 10^{-3} relative to the problem dimensions [52].

The weighted FCM shares the same solution structure as the IBM; however, instead of using step functions as the weighting and boundary value functions, the weighted FCM uses implicit functions directly. The trial solution in the weighted FCM follows as

$$\{u\} = W(\{x\})[M]\{d^e\} + G(\{x\}), \tag{3.11}$$

where W is a single function that combines all of the implicitly defined boundary conditions λ_i ($i = 1, \ldots, m$ with m being the total number of Dirichlet boundaries), G is the boundary value function and $\{x\}$ are the coordinate components. The construction of W is based on the level-set function (LSF). Examples of such LSF include using the product of all λ_i and employing the idea of R-functions [65]. The

[i] R-functions are real-value functions of real variables that are completely determined by the signs of their arguments and are independent of the arguments' magnitude.

boundary value function is given by

$$G(\{x\}) = \sum_{i=1}^{m} w_i(\{x\}) g_i, \tag{3.12}$$

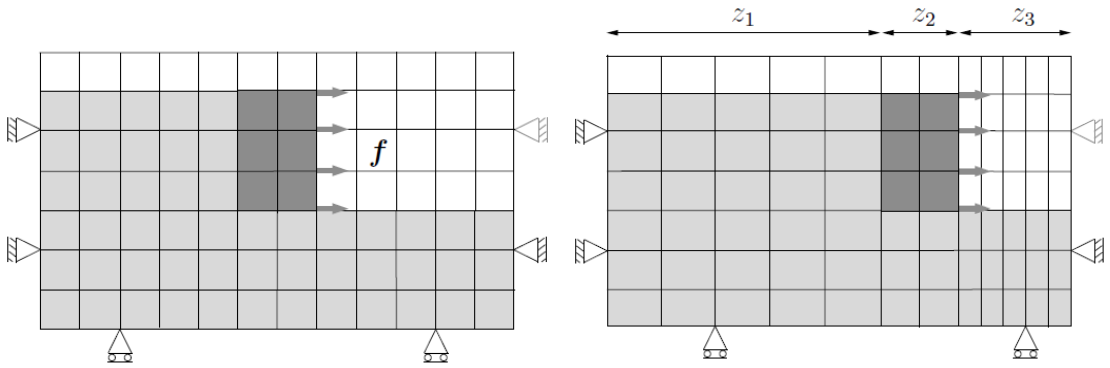
where w_i is a weighting coefficient and g_i is the prescribed value of Dirichlet boundary i. The computation of w_i is inspired by the idea of transfinite interpolation [68], and the general form and the general high-order form have also been proposed [65]. All three forms can be applied to formulate w_i, but only the transfinite interpolation form shows the partition of unity property [65]. Although it has been demonstrated that the weighted FCM can apply IDBCs exactly without mesh generation, as the trial solutions can be directly incorporated into the physical field, the accuracy of the overall solution largely depends on the level of the quadtree refinement that is performed around the problem boundaries and the order of Gauss quadrature used to perform the stiffness integration. This refinement procedure undermines one of the characteristic properties of the MPM which is that the same mesh can be used throughout the simulations without modifications. Additionally, there are no unique expressions of the weight function and the boundary value function because the implicit functions vary with individual BC. The non-unique expressions of the implicit functions also makes the weighted FCM a less suitable method to incorporate with the MPM for developing a generalised method on BC applications.

3.4 Moving mesh concept

Although it is a typical choice to use a stationary background mesh in the MPM, there is no such restriction that the mesh has to be fixed. In the review presented by Steffen *et al.*, implementing moving grid nodes was suggested as a method to track boundary forces [16]. Later in 2013, a moving mesh concept was proposed by Kafaji to model the penetration of piles into a soil body [2], and in the work of Phuong *et al.* for the same physical problem [69]. As the name states, part of the background mesh moves with the problem boundary where, normally, a displacement or velocity is prescribed to ensure that the mesh is always aligned with the boundary. Therefore, the boundary conditions are applied in the same way as in the FEM; no mapping between particles and the mesh nodes is required.

Figure 3.3 shows the initial and the deformed configuration of a rigid block being displaced under surface tractions. The moving mesh (z_2) is fixed to the material and moves with it during the deformation. Consequently, elements behind the moving mesh are stretched and those in front are compressed. Note that this method differs from the FEM in that the stretched and compressed mesh are reset after each load step. Additionally, the mesh deformations do not contribute to the stresses in the MPs; they are not physically deformed but are used as a way to track the moving

boundaries. In the case that mesh distortion under extreme deformation happens within a load step, meshing a wider region for the stretched and compressed zones or re-meshing after each load step can be performed.



(a) Initial configuration.

(b) Deformed configuration with stretched z_1, compressed z_3 and moving z_2 zone.

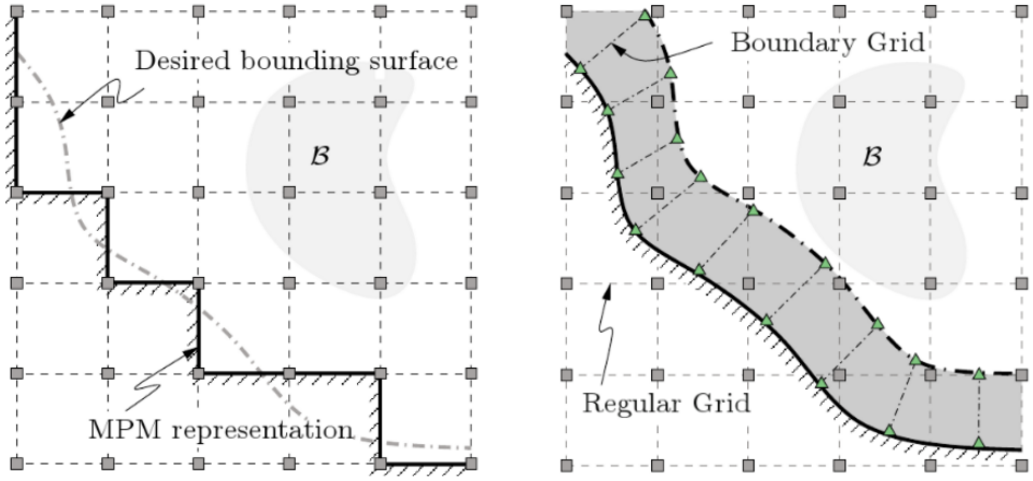
Figure 3.3: Initial and deformed configurations of the moving mesh concept (taken from Kafaji [2]).

Principally, the moving mesh procedure matches the computational mesh to the material velocity. Although this allows fine meshes to stay around the rigid body for boundary tracking, it is applicable only if the boundary does not change shape [5] and moves only in one direction [1] (for example, a rigid pile penetrating into a soil body).

3.5 Dual-grid extension

The dual-grid technique has been employed in the FEM for analysing sliding blocks and slopes [70], and solving nearly incompressible elasticity [71] and Stokes problems [72]. In the realm of MMs, Belytshko *et al.* introduced a method of enforcing essential BCs by implementing a string of FEs along the essential boundary [73]. The coupling between the MMs and the FEs is achieved at an interpolation level; the shape functions of both methods are combined to form a smooth transition from the meshless interior to the FE edge where BCs are imposed in the standard FEM way. This method was later generalised to a continuous blending for interpolation of all orders [74]. In another work, a bridging scale method was developed to allow the shape functions of the MMs to coexist with the FE shape functions at the essential boundaries [75].

Potentially inspired by the blending technique, Mast *et al.* proposed a similar idea for representing boundaries in the MPM in [3]. As illustrated in Figure 3.4a, standard MPM with a regular background mesh models a general boundary by horizontal and vertical lines; whereas, the dual-grid approach introduces a separate boundary grid (the grey area in Figure 3.4b) that conforms to the desired boundary.



(a) Conventional method. (b) Dual-grid approach.

Figure 3.4: Boundary representations in the MPM (taken from Mast *et al.* [3]).

The dynamics of this independent grid and the regular grid are linked through a blending approach as introduced in [50]. Both the boundary grid and the regular grid have their own velocity and acceleration fields with piecewise descriptions, and these two grids are simply connected by assigning all points that are on the grid interface (dotted dash line in Figure 3.4b) with the same velocity and acceleration. After solving the nodal accelerations and velocities for both grids, the particles are updated according to which grid they are located in. Alternatively, an enhanced velocity field approach was proposed in [50]. In this method, the total velocity and acceleration fields of the problem domain are results of the superposition of the velocity and acceleration fields from both grids. All particles are updated by the nodal values of the regular grid and an additional update is performed for the particles that are in the boundary grid.

Both approaches proposed in [50] were tested by modelling the impact between a one dimensional uniaxial steel bar and a rigid boundary. Results [50] indicate that both approaches are problematic. The blending method exhibits better accuracy providing that the boundary grid size is similar to the regular grid size. Although the early test results were not ideal, this dual-grid extension has the potential to accomplish the tasks of boundary tracking and boundary condition imposition in the MPM. However, no further development of this method has been seen in the published literature.

3.6 Second-order convected particle domain interpolation method

The second-order convected particle domain interpolation method (CPDI2) [4] is an extension to the convected particle domain interpolation method [25] which is, like the generalized interpolation material point method (GIMP) [22], an improvement of the standard MPM.

As mentioned in the introduction, prior to CPDI2, Ma [23] tracked the velocity and displacement at each corner of a MP domain to resolve separation between sets of MPs in GIMP method. However, he did not take it further to track the actual deformation of the particle domain. Instead, the shape of the particle domain remained rectangular with edges parallel to the coordinate axes.

In CPDI2, the problem domain is discretised into quadrilateral cells. The material points are located at the centroid of the elements. As the material deforms, instead of tracking the movement of the material points themselves, the corners of the elements are tracked and new quadrilaterals are formed. The material points are then repositioned at the centroid of the new quadrilaterals. From Figure 3.5, one can see that there are no gaps or overlaps of material point domains after the deformation and the material boundaries are clearly defined.

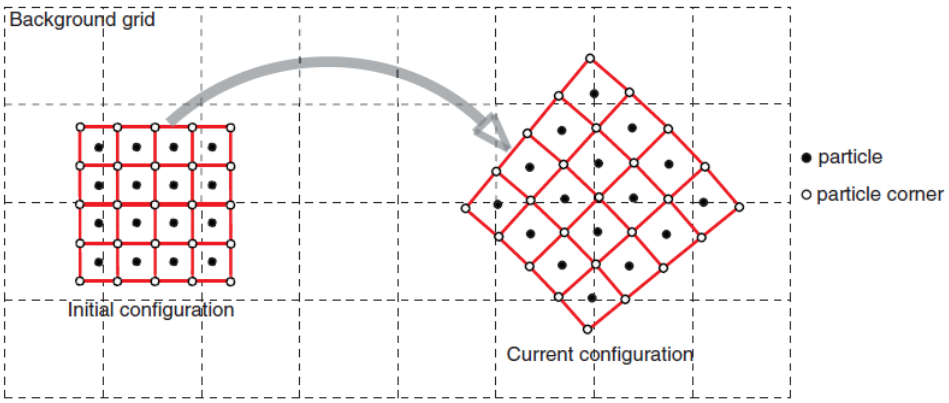


Figure 3.5: Deformation scheme of CPDI2 (taken from Sadeghirad [4]).

As for numerical calculations, material properties, momentum and forces are mapped from particles to corners and then from corners to the grid nodes at the beginning of each time step. The use of alternative basis functions that interpolate the corners of the particle domain with the background grid, however, introduces additional errors to the stiffness calculation in the CPDI methods because the piecewise linear nature of the background grid shape functions are lost. Moreover, unlike the GIMP method the integration is not performed exactly, and the CPDI methods assume that the variation of the background grid shape functions is linear over a particle domain which is only valid if all corners of the particle domain are located in the

same element. Overall, the CPDI methods provide a way to represent the material boundaries and they do make imposing traction boundary conditions simpler as tractions can be applied to the edges of the particle domains. However, applying IDBCs has not been simplified by the CPDI methods. Additionally, as mentioned in Chapter 1, the CPDI methods require additional memory to store both the mesh representing the material point domains and the background mesh.

3.7 Spline-based meshfree method (SBMFM)

In 2004, Natkar *et al.* [76] presented a new analysis methodology called constructive solid analysis where NURBS (non-uniform rational B-splines) basis functions were used to represent the boundaries of a problem domain and the field variables. Since the introduction of the term "isogeometric" analysis by Hughes *et al.* in 2005 [77], the use of NURBS has become even more popular in computational analysis for structural problems. Sevilla et al. [78] improved the traditional FEM by using NURBS defined boundaries and leaving the interior elements as standard FEs. They termed this method the NURBS-enhanced finite element method. Additionally, Steffen *et al.* [79] replaced the piecewise polynomial basis functions by quadratic B-spline basis in MPM to reduce the quadrature errors.

In order to explore the benefits of the isogeometric concept when applied to a MM, Kim and Youn [80] introduced a new class of MM called the spline-based meshfree method (SBMFM). In the SBMFM, the NURBS surface is not used to form the physical geometry of the material, instead it acts as the background mesh and trimming techniques are employed to describe the analysis domain. This means that the domain boundaries are clearly defined by smooth spline curves that have their own control points[ii].

When using SBMFM to analyse a geometrically nonlinear problem, trimming surface analysis [81, 82] is firstly performed, which includes the identification of the trimmed elements, numerical integration and the construction of stiffness matrix. Then the NURBS surface where the problem domain lies is deformed, followed by trimming curve evolution which updates the deformed problem domain. Finally the deformed NURBS surface is reset, and this procedure is repeated. NURBS curve fitting algorithm is adopted to update the deformed geometry. New spline curves are constructed by interpolation of sampling points on the deformed geometry. Control points ($\{P\}$) of these spline curves are assumed to be the only unknowns; the weights are prescribed to be 1. Let $\{Q_k\}$, $k = 0, 1, \ldots, n$, be the sampling points which has the same number as the spline control points. For a pth-degree NURBS curve, a $(n+1) \times (n+1)$ system can be written to solve the position of control

[ii]The control points mentioned here are referred to the control points in the sense of NURBS, and more details regarding the NURBS can be found in [54].

points

$$\{Q_k\} = \{C(\bar{\xi}_k)\} = \sum_{i=0}^{n} N_{i,p}(\bar{\xi}_k)\{P_i\}, \tag{3.13}$$

where $\bar{\xi}_k$ is a parameter (chosen by the centripetal method [54]) assigned to each sampling point, and $N_{i,p}$ is the ith NURBS basis function with degree p. This allows the boundary of the problem domain to be tracked.

Although the SBMFM still needs to be extended to three-dimensional problems, it has already demonstrated its ability to track a deformed boundary with great accuracy in two-dimensional space through the examples provided in [80]. Regardless of the fact that the SBMFM is rather different from the MPM, it offers an idea of boundary representation and reconstructing the deformed boundary with NURBS through sampling points.

3.8 Chapter review

Almost all methods regarding the application of BCs in the MPM literature are variations of the standard treatment, i.e. by the use of boundary particles, with different weighting parameters. This means that the accuracy of these methods depends on the element size and mesh refinement around the boundaries is required to maintain sufficient accuracy. However, this undermines one of the key benefits that the MPM provides, which is that the same structured mesh can be used throughout the simulation without any modifications. Alternatively, the moving mesh concept allows the BCs to be imposed in the FEM way, but only if the essential boundary does not change shape and moves in one direction. Another potential solution to the problem of BCs imposition is the dual-grid extension approach which blends a string of FE elements that locates the essential boundary with respect to the background mesh of the MPM. Unfortunately, simple tests have shown that this method is problematic and an improvement to the method has yet to be reported.

Additionally, methods of enforcing DBCs in the MMs and FEM that use a non-conforming mesh have also been investigated. The Web-method imposes HDBCs by using Web-splines as the shape functions. The IBM employs step functions as weight functions to approximate IDBCs; whereas, the weighted FCM incorporates the IDBCs into the physical field by using implicit equations directly as the weight function and the boundary value function. In this thesis, the IBM is chosen to be taken forward in the current research to impose IDBCs because the Web-method only enforces HDBCs and requires the use of non-standard shape functions. Although the weighted FCM is capable of applying IDBCs exactly, refining the mesh around the boundaries is necessary in order to achieve a high accuracy of the overall solution. Moreover, the IBM has already been adopted to the MPM framework and extended to include the imposition of “roller” BCs on inclined boundaries [5].

After reviewing some unique boundary representation methods, employing splines (such as in the SBMFM) seems to be the most appropriate choice. The outer layer of MPs of a problem domain can be used as sampling points from which a NURBS interpolated boundary is formed. In the brief introduction of this approach above, the weights in the NURBS curve are chosen to be 1, which makes the NURBS curve essentially a B-spline curve. Therefore, this thesis employs B-splines to define the problem boundaries in the MPM. Advantages of using this boundary description include the high continuity of B-splines and their ability to be manipulated independently from the background mesh. This chosen boundary representation method, i.e. B-splines, as well as a selection of B-spline fitting techniques will be introduced in the next chapter.

This is an intended blank page.

Chapter 4

B-spline Boundary Approximation

This chapter provides the essential background knowledge of B-splines boundary representation. The construction of B-spline curves and surfaces is firstly introduced in Section 4.1. Then detailed discussions on how to fit a B-spline to a set of data are presented. There are two basic types of fitting: interpolation and approximation [54]. A curve or surface constructed by the former method satisfies the given data precisely, i.e. the curve or the surface passes through the data points, whereas the latter only gives an approximation. In order to have a greater control on the B-spline represented boundaries and to ensure that the spline passes through key boundary points, the interpolation method is carried forward. The majority of the fitting algorithms can be categorised as either global or local [54]. Both methods will be explained and numerical comparisons will be made to select the most suitable boundary representation technique.

4.1 B-spline curves and surfaces

A B-spline curve consists of two elements: basis functions that are formed by a knot vector and a set of control points. A knot vector ($\{\Xi\}$) is defined as $\{\Xi\} = \{\xi_0, \xi_1, \ldots, \xi_r\}$, where $\xi_0, \xi_1, \ldots, \xi_r$ are called the knots and form a sequence of nondecreasing real numbers, i.e. $\xi_i \in \mathbb{R}$ and $\xi_i \leq \xi_{i+1}$ with $i = 0, 1, \ldots, r-1$. An open (clamped) knot vector is the most popular choice due to its interpolative nature with the first and the last control points and its wide application in CAD [56]. An open knot vector has $(p+1)$ repeated knots at the beginning and the end (normally taking the value of 0 and 1 respectively) of the knot vector with p being the spline's polynomial degree.

There are many ways to define B-spline basis functions. However, to ease the computation process, the recurrence formula [83] is adopted. So, the ith B-spline basis

function with degree p can be defined as, for $p = 0$

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

and for $p > 0$

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

Equation (4.1) shows that $N_{i,0}(\xi)$ is a step function, equal to zero everywhere apart from the half-open knot span with $\xi \in [\xi_i, \xi_{i+1})$. However, there is a special case. If the arbitrary ξ takes the value of ξ_{i+1}, where $i = r - p$ in a open knot vector, $N_{i,0}(\xi) = 1$ unlike (4.1) states. Equation (4.2) indicates that $N_{i,p}(\xi)$ is a linear combination of two $(p-1)$-degree basis functions for all $p > 0$, and with $(r+1)$ knots, there are $(r-p)$ pth-degree basis functions $N_{i,p}(\xi)$. Note that in (4.2), a quotient of $\frac{0}{0}$ is defined as zero.

A pth-degree B-spline curve is defined as

$$\{C(\xi)\} = \sum_{i=0}^{n} N_{i,p}(\xi)\{P_i\}, \tag{4.3}$$

where $\{P_i\}$ are the control points, and $N_{i,p}(\xi)$ represents the pth-degree B-spline basis functions. The linear interpolation of the control points forms the control polygon. An important point to be noted here is the relationship between the number of knots $(r+1)$, the number of control points $(n+1)$ and the degree (p) of the B-spline curve, that is

$$r = n + p + 1. \tag{4.4}$$

A B-spline surface is a tensor product of a control net and two basis functions formed from two knot vectors respectively

$$\{S(\xi, \eta)\} = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(\xi) N_{j,q}(\eta)\{P_{i,j}\} \tag{4.5}$$

with

$$\{\Xi\} = \{\xi_0, \xi_1, \ldots, \xi_r\} \quad \text{and} \quad \{H\} = \{\eta_0, \eta_1, \ldots, \eta_s\},$$

where n is the number of control points in the ξ direction, m is the number of control points in the η direction, $\{P_{i,j}\}$ are the $(n+1) \times (m+1)$ control points, $N_{i,p}(\xi)$ is the pth-degree B-spline basis functions associated with knot vector $\{\Xi\}$, and $N_{j,q}(\eta)$ is the qth-degree B-spline basis functions associated with knot vector $\{H\}$. Similar to B-spline curves, the number of knots $(s+1)$, the number of control points $(m+1)$

and the degree (q) of the B-splines have the following additional relationship

$$s = m + q + 1. \tag{4.6}$$

4.2 Global interpolation

The global interpolation method inverts the calculation procedure of (4.3), where $\{C(\xi)\}$ consists of a set of sampling points. If the degree and knots are prescribed, the resulting system would be a set of easy-to-solve linear equations with the control points being the only unknowns. Because the fitted curve is determined as a whole, any perturbation in the sampling data would have an influence on the shape of the entire B-spline curve or surface, however this influence on the curve's or surface's shape reduces when moving away from the perturbation point [54].

4.2.1 Curve fitting

Given a set of $(n+1)$ sampling points $\{Q_k\}$, $k = 0, \ldots, n$, that we want to interpolate with a pth-degree B-spline curve, a parameter (knot) $\bar{\xi}_k$ is assigned to each sampling point. Then a suitable knot vector, $\{\Xi\} = \{\xi_0, \xi_1, \ldots, \xi_r\}$ is chosen to construct the basis functions. What we end up with is a $(n+1)\times(n+1)$ linear system of equations with the form

$$\{Q_k\} = \{C(\bar{\xi}_k)\} = \sum_{i=0}^{n} N_{i,p}(\bar{\xi}_k)\{P_i\}, \tag{4.7}$$

where $\{P_i\}$, the control point positions, are the only unknowns.

There are three common methods of obtaining $\bar{\xi}_k$, which are the equally spaced, chord length and centripetal methods. Because the centripetal method is based on the curvature of the curves [84], it has the best performance comparing to the other methods when the sampling data take sharp turns. As sharp turns are essential features in many geometrical simulations, the centripetal method is adopted in this thesis. With the assumption $\bar{\xi} \in [0, 1]$, it produces $\bar{\xi}_k$ as

$$\begin{aligned} &\bar{\xi}_0 = 0, \quad \bar{\xi}_n = 1 \quad \text{and} \\ &\bar{\xi}_k = \bar{\xi}_{k-1} + \frac{\sqrt{|\{Q_k\} - \{Q_{k-1}\}|}}{\sum_{l=1}^{n} \sqrt{|\{Q_l\} - \{Q_{l-1}\}|}}, \quad k = 1, \ldots, n-1, \end{aligned} \tag{4.8}$$

where $\bar{\xi}_0$ and $\bar{\xi}_n$ are the knots associated with the first and the last sampling points respectively.

One of the methods of obtaining the open knot vector $\{\Xi\}$ is to simply equally space the knots; however, $\{\Xi\}$ found though this method could lead (4.7) to becoming a singular system. Instead, knots in $\{\Xi\}$ can be calculated based on the value of $\bar{\xi}_k$

as follows,

$$\begin{aligned}
&\xi_0 = \ldots = \xi_p = 0, \\
&\xi_{m-p} = \ldots = \xi_r = 1 \quad \text{and} \\
&\xi_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} \bar{\xi}_i, \quad j = 1, \ldots, n-p,
\end{aligned} \tag{4.9}$$

where $\xi_0 = \ldots = \xi_p = 0$ and $\xi_{m-p} = \ldots = \xi_r = 1$ are the first and last $(p+1)$ knots in the knot vector $\{\Xi\}$. With this method, the knot vector reflects the distribution of $\bar{\xi}_k$, and (4.7) results in a totally positive matrix (which is a square matrix where the determinate of each square submatrix is positive) with a semi-bandwidth less than p [54]. This means that the linear system could be solved by Gaussian elimination without pivoting. After solving for the control points, the fitted B-spline curve can be constructed by using (4.3).

4.2.2 Surface fitting

To fit a surface, $(n+1) \times (m+1)$ data points $\{Q_{k,l}\}$ ($k = 0, \ldots, n$ and $l = 0, \ldots, m$) are interpolated by a (p, q)th-degree B-spline as

$$\{Q_{k,l}\} = \{S(\bar{\xi}_k, \bar{\eta}_l)\} = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(\bar{\xi}_k) N_{j,q}(\bar{\eta}_l) \{P_{i,j}\}. \tag{4.10}$$

Similar to curve fitting, the control point positions, $\{P_{i,j}\}$, are the only unknowns, and $\bar{\xi}_k$ and $\bar{\eta}_l$ are the knot values assigned to each sampling points in the direction of ξ and η respectively.

For a fixed value of l, $\bar{\xi}_k^l$ are calculated using the centripetal method (4.8), and then $\bar{\xi}_k$ is obtained by averaging across all $\bar{\xi}_k^l$ with $l = 0, \ldots, m$, that is

$$\bar{\xi}_k = \frac{1}{m+1} \sum_{l=0}^{m} \bar{\xi}_k^l, \quad k = 0, \ldots, n, \tag{4.11}$$

and $\bar{\eta}_l$ is calculated in the same fashion as $\bar{\xi}_k$. The open knot vectors $\{\Xi\}$ and $\{H\}$ can then constructed by (4.9).

Because $\{S(\bar{\xi}_k, \bar{\eta}_l)\}$ is obtained from a tensor product, (4.10) can be written as

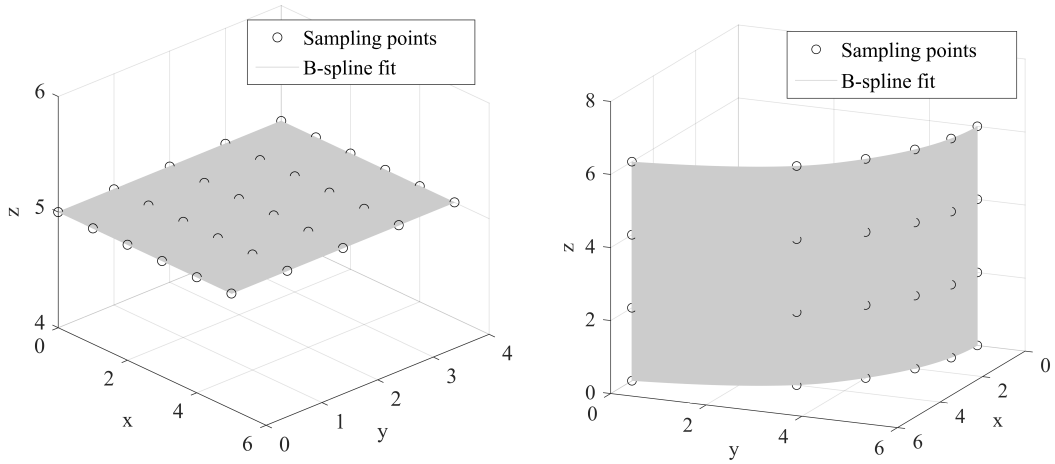
$$\{Q_{k,l}\} = \{S(\bar{\xi}_k, \bar{\eta}_l)\} = \sum_{i=0}^{n} N_{i,p}(\bar{\xi}_k) \{U_{i,l}\}, \tag{4.12}$$

where

$$\{U_{i,l}\} = \sum_{j=0}^{m} N_{j,q}(\bar{\eta}_l) \{P\}_{i,j}. \tag{4.13}$$

To find the control points $\{P_{i,j}\}$, l is firstly fixed. For each value of l, $\{U_{i,l}\}$ is obtained by $(m+1)$ curve interpolations through $\{Q_{0,l}\}, \ldots, \{Q_{n,l}\}$ with $l = 0, \ldots, m$ (4.12). Then for each value of i, $\{P_{i,j}\}$ is calculated by $(n+1)$ curve interpolations through $\{U_{i,0}\}, \ldots, \{U_{i,m}\}$ with $i = 0, \ldots, n$ (4.13).

To verify the implementation, two surface fittings were simulated. The first, illustrated in Figure 4.1a is a biquadratic B-spline fit of a plane with constant $z = 5$. Twenty equally spaced data points were sampled (circles in Figure 4.1a) and the calculated control points are also located at these locations; the B-spline fit (grey surface) interpolates the sampling points exactly.



(a) A plane

(b) A quarter of a cylinder

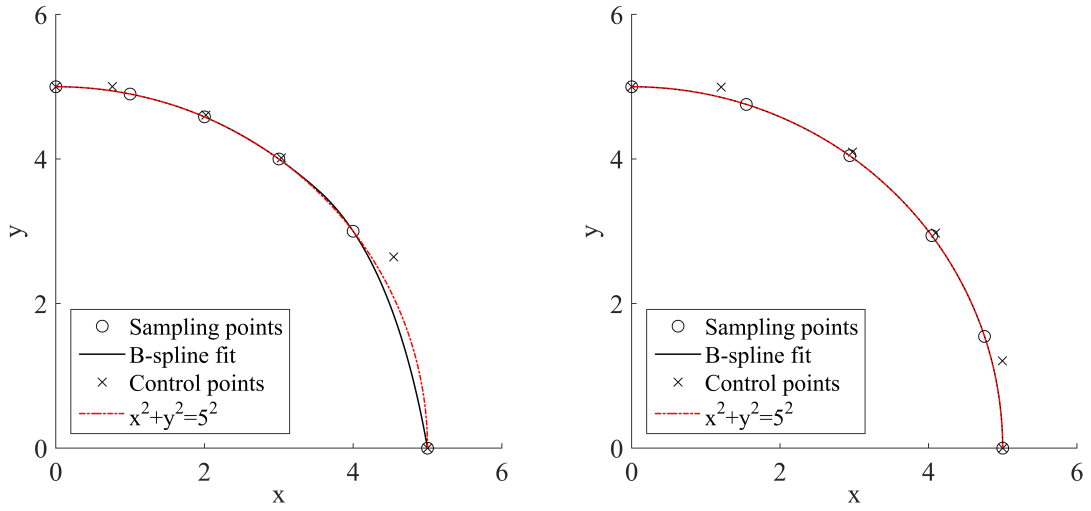
Figure 4.1: Examples of biquadratic B-spline surface fit.

The other example given here is the fitting of the surface of a quarter of a cylinder. Figure 4.1b shows the surface fitted with a biquadratic B-spline. Twenty-four data points were sampled on the surface with equally spaced x values. Values of y were varied according to

$$y = \sqrt{5^2 - x^2}, \tag{4.14}$$

and z was varied independently from 0 to 6. Figure 4.2a is a top view of the B-spline fit. The black solid line represents the fitted curve whereas the red dotted dash line is the analytical quarter circle. Additionally, control points governing this curve are shown as crosses. It can be seen that in the region $x \in [0, 3]$ where the curve is more constrained, B-spline approximation shows a reasonable fit to the desired geometry. This means with more sampling points being placed at $x \in (3, 5]$, a better fit could be achieved. Alternatively, equally spacing the same number of sampling points in terms of their angular positions from the origin has significantly improved the fitting accuracy (see Figure 4.2b). To quantify the comparison, relative errors between the fitted curve and the analytical curve of the quarter circle were calculated in 2D using (4.46) for both distributions of sampling points. Distribution shown in Figure 4.2a has an error of 0.03 whereas the other distribution gives a relative error of

2.49×10^{-4}.



(a) Sampling points with equal x spacing

(b) Sampling points with equal angular spacing

Figure 4.2: Compression between the analytical cylinder base and the B-spline fit.

4.3 Curve fitting by local interpolation

The local interpolation method constructs curves or surfaces in a piecewise fashion. Only local data are used at each step, therefore it is usually computationally less expensive than the global method. Additionally, a fluctuation in data would only affect the curve or surface locally. Local interpolation also has the ability to deal with sharp corners, however, this comes at a price of lower continuity and multiple interior knots [54].

In general, the local interpolation of curve fitting performs as follows: local polynomial or spline curves are constructed between every two sampling points by calculating the control points, and then a knot vector is determinated according to the requirement of continuity for the overall curve. Finally, a B-spline curve (or surface) is constructed from the knot vector and the control points.

Given $\{Q_k\}$, $k = 0, \ldots, n$, the $(n+1)$ sampling points we want to interpolate. Let $\{C_i(\xi)\}$, $i = 0, \ldots, n-1$ be the n curve segments with end points $\{Q_i\}$ and $\{Q_{i+1}\}$. To achieve at least G^1 continuity[i], the tangent at the end of a curve segment should be in the same direction of the tangent at the start of the following curve segment. In other words, if $\bar{\xi}_{i+1}$ is the end parameter of $\{C_i(\xi)\}$ and the start parameter of $\{C_{i+1}(\xi)\}$, then $\{C_i'(\bar{\xi}_{i+1})\}$ should be in the same direction of $\{C_{i+1}'(\bar{\xi}_{i+1})\}$ regardless of their magnitudes.

[i]The geometric continuity (G continuity) is the continuity of implicit functions; a curve that is G^0 continuous consists curve segments that are touched at the join point. A G^1 continuous curve requires its segments to share common tangents at the join points.

One way to represent the curve segments $\{C_i(\xi)\}$ is via Bézier curves [85]. Bézier curve is a parametric polynomial curve, where polynomials are used as the coordinate functions [54]. The main difference between the Bézier curves and the B-spline curves is that a nth degree Bézier curve represents one nth degree polynomial whereas a B-spline curve patches several Bézier curves together by the knot vector. To obtain the Bézier segments, inner control points, in addition to the sampling points, need to be calculated. One point is required for quadratic curves and two for cubic curves. These control points lie on the tangents of $\{Q_k\}$; therefore, calculations of the tangent (either D_k or V_k) at each $\{Q_k\}$ are required. Although there are many ways to compute the tangents [86], all methods can be categorised into two forms: one with assigned knot values at the sampling points and the other is independent from the knot values. An illustration of the relationships between different quantities that are required in the tangent calculation is shown in Figure 4.3.

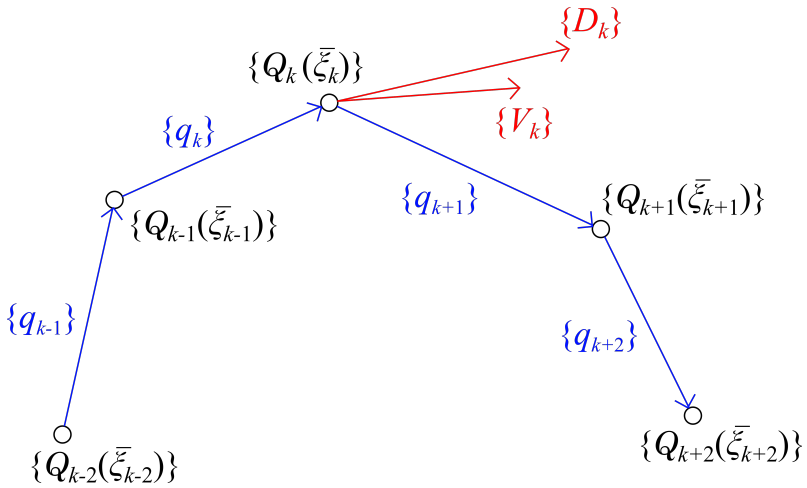


Figure 4.3: Calculation of the tangent vectors D_k and V_k for local interpolation.

With $k = 1, \ldots, n$, we define

$$\{q_k\} = \{Q_k\} - \{Q_{k-1}\}, \quad \Delta\bar{\xi}_k = \bar{\xi}_k - \bar{\xi}_{k-1} \quad \text{and} \quad \{d_k\} = \frac{\{q_k\}}{\Delta\bar{\xi}_k}. \tag{4.15}$$

The tangent, $\{D_k\}$, $k = 1, \ldots, n-1$, calculated from the former category is a linear interpolation of the cord lengths' gradients $\{d_k\}$

$$\{D_k\} = (1 - \alpha_k)\{d_k\} + \alpha_k\{d_{k+1}\}. \tag{4.16}$$

One way of determining α_k is via the Bessel method [87]

$$\alpha_k = \frac{\Delta\bar{\xi}_k}{\Delta\bar{\xi}_k + \Delta\bar{\xi}_{k+1}}, \tag{4.17}$$

which is a three-point method using sampling points $\{Q_{k-1}\}$, $\{Q_k\}$, and $\{Q_{k+1}\}$. A

special treatment at the ends is required, where we set

$$\{D_0\} = 2\{d_1\} - \{D_1\} \quad \text{and} \quad \{D_n\} = 2\{d_n\} - \{D_{n-1}\}. \tag{4.18}$$

The method that is independent from the knot values produces unit length tangents $\{T_k\}$, $k = 0, \ldots, n$

$$\{T_k\} = \frac{\{V_k\}}{|\{V_k\}|} \quad \text{and} \quad \{V_k\} = (1 - \beta_k)\{q_k\} + \beta_k\{q_{k+1}\}. \tag{4.19}$$

β_k can be found through a five-point method [88]

$$\beta_k = \frac{|\{q_{k-1}\} \times \{q_k\}|}{|\{q_{k-1}\} \times \{q_k\}| + |\{q_{k+1}\} \times \{q_{k+2}\}|}, \tag{4.20}$$

and $\{q_k\}$ is calculated by $(4.15)_1$ with ends treatments

$$\{q_0\} = 2\{q_1\} - \{q_2\}, \qquad \{q_{-1}\} = 2\{q_0\} - \{q_1\}, \tag{4.21}$$

$$\{q_{n+1}\} = 2\{q_n\} - \{q_{n-1}\} \qquad \text{and} \qquad \{q_{n+2}\} = 2\{q_{n+1}\} - \{q_n\}. \tag{4.22}$$

This five-point method has the advantage that a straight line through three points can be reproduced by setting $\beta_k = 1$; however, β_k can become undefined if $\{q_{k-1}\}$ is parallel to $\{q_k\}$ and $\{q_{k+1}\}$ is parallel to $\{q_{k+2}\}$. The undefined β_k indicates either a straight line from $\{Q_{k-1}\}$ to $\{Q_{k+2}\}$, or a corner at $\{Q_k\}$. A flag system is used to indicate at which $\{Q_k\}$ occurs a corner, and one can choose to either smooth out the corner by setting $\beta_k = \frac{1}{2}$ [54] or preserve it. In the case of preserving the corner, two tangents are required at $\{Q_k\}$ to calculate the additional control points. One in the direction of $\{q_k\}$, i.e. $\beta_k = 0$; the other in the direction of $\{q_{k+1}\}$, i.e. $\beta_k = 1$. The latter method is employed for the computation of control points, because it does not require pre-assignment of the knot value at each sampling point and it has the ability to reproduce straight lines.

4.3.1 Quadratic scheme

Assume there is an intersection of tangents $\{T_{k-1}\}$ and $\{T_k\}$. The inner control point, $\{R_k\}$, $k = 1, \ldots, n$, of a curve segment is located at this intersection point if $\{R_k\}$ is in the positive direction of $\{T_{k-1}\}$ and the negative direction of $\{T_k\}$ (see Figure 4.4a). Then, $\{R_k\}$ is calculated as follows,

$$\{R_k\} = \{Q_{k-1}\} + \gamma_{k-1}\{T_{k-1}\}, \qquad \gamma_{k-1} > 0, \tag{4.23}$$

$$\{R_k\} = \{Q_k\} + \gamma_k\{T_k\} \qquad \text{and} \qquad \gamma_k < 0. \tag{4.24}$$

However, if the conditions $\gamma_{k-1} > 0$ and $\gamma_k < 0$ are not satisfied simultaneously, two parabolic curves between the sampling points are constructed instead of one (see Figure 4.4b). This means that three inner points are required [54]

$$\{R'_k\} = \{Q'_{k-1}\} + \gamma_k\{T_{k-1}\}, \tag{4.25}$$

$$\{R'_{k+1}\} = \{Q'_k\} - \gamma_{k+1}\{T_k\} \qquad \text{and} \tag{4.26}$$

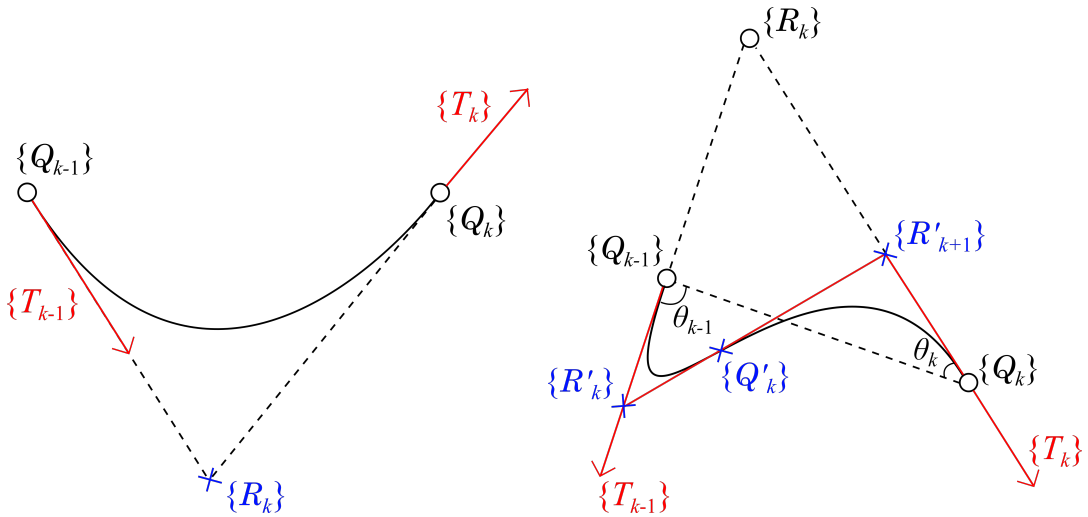
$$\{Q'_k\} = \frac{\gamma_k\{R'_{k+1}\} + \gamma_{k+1}\{R'_k\}}{\gamma_k + \gamma_{k+1}}. \tag{4.27}$$

To clarify, the apostrophe of $\{R'_k\}$, $\{R'_{k+1}\}$ and $\{Q'_k\}$ is just an indication of additional points and it dose not imply differentiation. For this case, parameters γ_k and γ_{k+1} are set to be

$$\gamma_k = \frac{1}{4}\frac{|\{Q_{k-1}\}\{Q_k\}|}{a\cos\theta_k + (1-a)\cos\theta_{k-1}} \qquad \text{and} \tag{4.28}$$

$$\gamma_{k+1} = \frac{1}{4}\frac{|\{Q_{k-1}\}\{Q_k\}|}{a\cos\theta_{k-1} + (1-a)\cos\theta_k}, \tag{4.29}$$

where θ_{k-1} is the included angle of $\{Q_{k-1}\}\{Q_k\}$ and $\{T_{k-1}\}$ and $\theta_{k-1} \geq 0$; θ_k is the included angle of $\{Q_{k-1}\}\{Q_k\}$ and $\{T_k\}$ and $\theta_k \leq 90°$; $0 \leq a \leq 1$, through observations from [54], a is chosen to be $\frac{2}{3}$.



(a) Conditions $\gamma_{k-1} > 0$ and $\gamma_k < 0$ are satisfied. (b) A turing point is implied by $\{T_{k-1}\}$ and $\{T_k\}$.

Figure 4.4: Computation of the control points.

We now drop the assumption that $\{T_{k-1}\}$ and $\{T_k\}$ intersect. If both $\{T_{k-1}\}$ and $\{T_k\}$ are parallel to the cord $\{Q_{k-1}\}\{Q_k\}$, the additional control point is simply calculated as the midpoint of the chord

$$\{R_k\} = \frac{1}{2}\left(\{Q_{k-1}\} + \{Q_k\}\right). \tag{4.30}$$

If tangents $\{T_{k-1}\}$ and $\{T_k\}$ are parallel but not to the cord $\{Q_{k-1}\}\{Q_k\}$, e.g. a curve exhibits a 180° turn, (4.25), (4.26) and (4.27) are used with values of γ_k and γ_{k+1} set to

$$\gamma_k = \gamma_{k+1} = \frac{1}{2} \left| \{Q_{k-1}\}\{Q_k\} \right|. \tag{4.31}$$

To obtain a C^1 continuous curve, all the sampling points are omitted in the formation of the control polygon apart from the ends. The control points for the fitted B-spline are formed as follows

$$\{Q_0\}, \{R_1\}, \{R_2\}, \ldots, \{R_n\}, \{Q_n\}.$$

If any curve segment is constructed by two parabolic curves $\{Q'_k\}$, then, forms part of the sampling points as the curve interpolates this point. Consequently, $\{R'_k\}$ becomes the inner control point of arc $\{Q_k\}\{Q'_k\}$, and $\{R'_{k+1}\}$ becomes the inner control point of arc $\{Q'_k\}\{Q_{k+1}\}$.

The interior knots are calculated by equating the start and end derivatives at $\{Q_{k-1}\}$ [54],

$$\begin{aligned} &\bar{\xi}_0 = 0, \quad \bar{\xi}_n = 1 \quad \text{and} \\ &\bar{\xi}_k = \bar{\xi}_{k-1} + \left(\bar{\xi}_{k-1} - \bar{\xi}_{k-2}\right) \frac{|\{R_k\} - \{Q_{k-1}\}|}{|\{Q_{k-1}\} - \{R_{k-1}\}|}, \quad k = 2, \ldots, n', \end{aligned} \tag{4.32}$$

where $\bar{\xi}_0$ and $\bar{\xi}_n$ are the knots associated with the first and the last sampling points respectively. There is only one occurrence of each interior knot in the open knot vector

$$\{\Xi\} = \{0, 0, 0, \frac{\bar{\xi}_1}{\bar{\xi}_n}, \frac{\bar{\xi}_2}{\bar{\xi}_n}, \ldots, \frac{\bar{\xi}_{n-2}}{\bar{\xi}_n}, \frac{\bar{\xi}_{n-1}}{\bar{\xi}_n}, 1, 1, 1\}. \tag{4.33}$$

In the case that there are sharp corners being preserved, G^0 continuity is introduced at the corners. All the sampling points now form part of the control point polygon

$$\{Q_0\}, \{R_1\}, \{Q_1\}, \{R_2\}, \{Q_2\}, \ldots, \{R_n\}, \{Q_n\}.$$

The choice of interior knots does not have an impact on the shape of the curve, only parametrisation does. Therefore, they are chosen to be equal spaced between 0 and 1 and each knot has a multiplicity two apart from the end knots,

$$\{\Xi\} = \{0, 0, 0, \xi_1, \xi_1, \ldots, \xi_{n-1}, \xi_{n-1}, 1, 1, 1\}.$$

Finally, the overall curve is fitted by computing the basis functions through (4.1) and (4.2), and then applying (4.3).

4.3.2 Cubic scheme

Two additional control points are required for this scheme for each Bézier segment, which are computed by four quantities: two end points, $\{Q_k\}$ and $\{Q_{k+1}\}$, and their tangents, $\{T_k\}$ and $\{T_{k+1}\}$. Additionally, to achieve a true uniform parametrisation, constant speed (that is, a constant first derivative of the overall fitted curve) is required through out the curve.

Let δ_v denote the speed. For any Bézier curve segment $\{C_{Bz_i}(\xi)\}$, $\xi \in [0,1]$, we have end control points $\{P_0\} = \{Q_i\}$ with tangent $\{T_0\} = \{T_i\}$, $\{P_3\} = \{Q_{i+1}\}$ with tangent $\{T_3\} = \{T_{i+1}\}$ and the additional control points $\{P_1\}$ and $\{P_2\}$ calculated as

$$\{P_1\} = \{P_0\} + \frac{1}{3}\delta\{T_0\} \quad \text{and} \quad \{P_2\} = \{P_3\} - \frac{1}{3}\delta\{T_3\}, \tag{4.34}$$

respectively. The speed, δ_v, is set to be equal at the start-, the mid- and the end-points of the curve, that is

$$\delta_v = \left|\{C'_{Bz_i}(0)\}\right| = \left|\{C'_{Bz_i}\}\left(\frac{1}{2}\right)\right| = \left|\{C'_{Bz_i}(1)\}\right|, \tag{4.35}$$

where the derivative of $\{C_{Bz_i}(\xi)\}$ is

$$\{C'_{Bz_i}(\xi)\} = p\sum_{j=0}^{p-1}\{B_{j,p-1}(\xi)\}\left(\{P_{j+1}\} - \{P_j\}\right) \tag{4.36}$$

and $\{B(\xi)\}$ is the Bézier basis function

$$\{B_{j,p}(\xi)\} = \frac{p!}{j!(p-j)!}\xi^j(1-\xi)^{p-j}. \tag{4.37}$$

Consider $\xi = \dfrac{1}{2}$ on a cubic Bézier curve, we have

$$\begin{aligned}\{C'_{Bz_i}\}\left(\frac{1}{2}\right) &= 3\sum_{j=0}^{2}\{B_{j,2}\}\left(\frac{1}{2}\right)\left(\{P_{j+1}\} - \{P_j\}\right)\\ &= \frac{3}{4}\left(\{P_3\} + \{P_2\} - \{P_1\} - \{P_0\}\right).\end{aligned} \tag{4.38}$$

Substituting (4.38) into (4.35), and then using relationships (4.34), we have

$$\begin{aligned}\frac{4}{3}\delta_v &= \left|\left(\{P_3\} + \{P_2\} - \{P_1\} - \{P_0\}\right)\right|\\ &= \left|\left(\{P_3\} + \left(\{P_0\} + \frac{1}{3}\delta\{T_0\}\right) - \left(\{P_3\} - \frac{1}{3}\delta\{T_3\}\right) - \{P_0\}\right)\right|.\end{aligned} \tag{4.39}$$

Rearranging (4.39) arrives at a quadratic equation by which δ_v can be solved,

$$\begin{aligned} 0 = &\left(16 - |\{T_3\} + \{T_0\}|^2\right) \delta_v^2 \\ &+ 12\left(\{P_3\} - \{P_0\}\right) \cdot \left(\{T_3\} + \{T_0\}\right) \delta_v - 36\,|\{P_3\} - \{P_0\}|^2 . \end{aligned} \tag{4.40}$$

Finally, the positive value of δ is substituted into $(4.34)_1$ and $(4.34)_2$ to obtain $\{P_1\}$ and $\{P_2\}$ respectively.

With all the control points calculated, the next step is to determine the B-spline knot vector. A fit with C^1 continuity is achievable by including the calculated control points in the control polygon, $\{P_{k,j}\}$ with $k = 0, \ldots, n-1$, and $j = 0, 1$ and the end sampling points, that is

$$\{Q_0\}, \{P_{0,0}\}, \{P_{0,1}\}, \{P_{1,0}\}, \{P_{1,1}\}, \ldots, \{P_{n-1,0}\}, \{P_{n-1,1}\}, \{Q_n\} \tag{4.41}$$

Then the interior knots and the knot vector are calculated as

$$\bar{\xi}_0 = 0, \quad \bar{\xi}_{k+1} = \bar{\xi}_k + 3\,|\{P_{k,1}\} - \{P_{k,0}\}|\,, \tag{4.42}$$

and

$$\{\Xi\} = \{0, 0, 0, 0, \frac{\bar{\xi}_1}{\bar{\xi}_n}, \frac{\bar{\xi}_1}{\bar{\xi}_n}, \frac{\bar{\xi}_2}{\bar{\xi}_n}, \frac{\bar{\xi}_2}{\bar{\xi}_n}, \ldots, \frac{\bar{\xi}_{n-1}}{\bar{\xi}_n}, \frac{\bar{\xi}_{n-1}}{\bar{\xi}_n}, 1, 1, 1, 1\}. \tag{4.43}$$

Similar to the quadratic scheme, when there are sharp corners being preserved, all the sampling points become part of the control polygon,

$$\begin{aligned} \{Q_0\}, \{P_{0,0}\}, \{P_{0,1}\}, &\{Q_1\}, \{P_{1,0}\}, \{P_{1,1}\}, \ldots \\ &\ldots, \{Q_{n-1}\}, \{P_{n-1,0}\}, \{P_{n-1,1}\}, \{Q_n\}. \end{aligned} \tag{4.44}$$

The interior knots are chosen to be equal spaced between 0 and 1 and each has multiplicity three apart from the end knots,

$$\{\Xi\} = \{0, 0, 0, 0, \xi_1, \xi_1, \xi_1, \ldots, \xi_{n-1}, \xi_{n-1}, \xi_{n-1}, 1, 1, 1, 1\}. \tag{4.45}$$

Finally, the overall curve is fitted in the same fashion as the quadratic scheme.

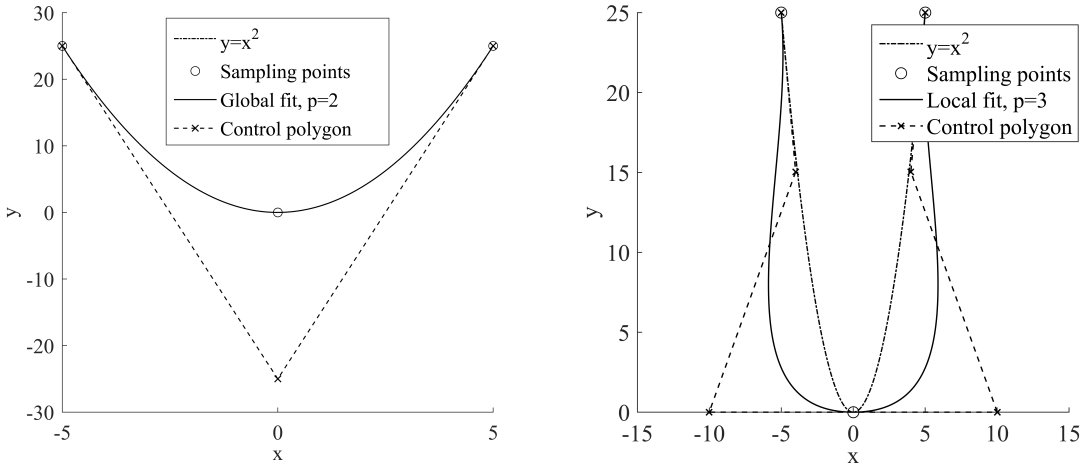
4.4 Numerical comparisons

Different curves, involving $y = x^2$, $y = x^3$, and sharp corners, were simulated using global interpolation with different B-spline degrees and both quadratic and cubic local interpolation. Differences between the analytical and interpolated curves were quantified. In order to calculate the errors, a large number of points were sampled from the fitted curve $(\{x^h\}, \{y^h\})$ and $\{x^h\}$ was fed to the analytical equations to

acquire the analytical values of y ($\{y^a\}$). Then, the relative error R_c between the analytical and the approximated curves was calculated as

$$R_c = \sqrt{\frac{\{\{y^h\} - \{y^a\}\}^T \{\{y^h\} - \{y^a\}\}}{\{y^a\}^T \{y^a\}}}. \tag{4.46}$$

Starting with curve $y = x^2$, $x \in [-5, 5]$, Figure 4.5a shows the fitting by using a 2nd-degree B-spline with 3 sampling points under the global scheme. The x values of the sampling points were chosen to be equally spaced between -5 and 5 (i.e. $x = -5, 0, 5$) and y were calculated accordingly. In Figure 4.5a, sampling points are shown as circles and the dashed line is the control polygon with the crosses being the control points. Curves in dotted dash and solid lines are plots of the analytical and B-spline fit respectively. The reason that the dotted dash line cannot be seen is that an exact fit is achieved. It is the same story using the local quadratic scheme; however, 3 sampling points are not enough for the local cubic method to give an exact fit (see Figure 4.5b). This is because the quadratic curve segment between two sampling points was approximated by cubic polynomials.



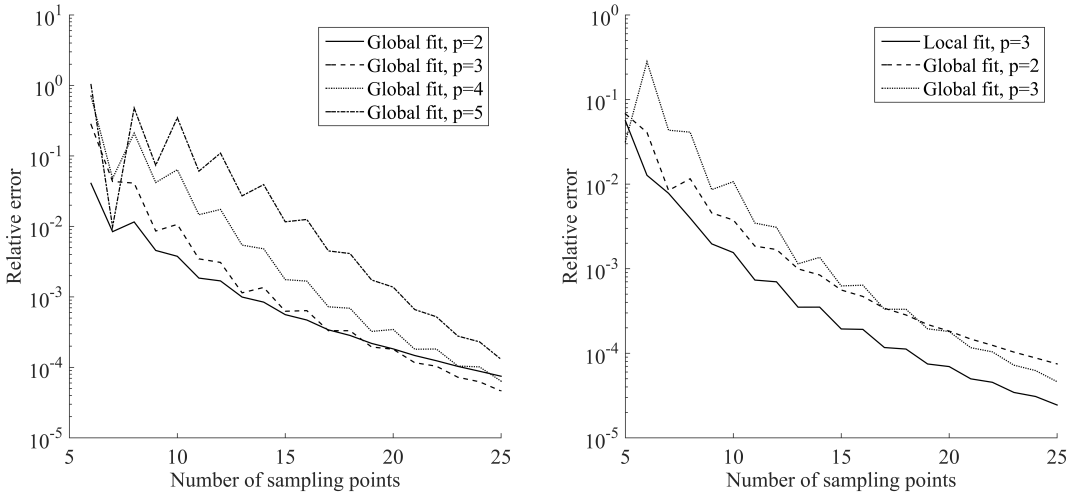
(a) Global interpolation by 2nd-degree B-spline (b) Local interpolation by 3rd-degree B-spline

Figure 4.5: Interpolation of $y = x^2$ through 3 sampling points.

In Figure 4.6a, the relative error between the analytical curve and the fitted curve for $y = x^2$ is plotted against the number of sampling points with different degrees of B-spline. By increasing the number of points sampled along the curve, all degrees show the general trend of decreasing error; however, increasing the B-spline's degree at a given number of sampling points generates the opposite trend. When the number of sampling points is less than 19, the 2nd-degree B-spline gives the best approximation compared to higher degree B-splines. Nonetheless, the difference between the errors of different B-spline degrees reduces when more sampling points are used. All degrees show some regular fluctuations. The reason could be that the

sampling points are placed at less desirable locations when even number of sampling points are used which causes higher errors comparing to the approximations that used odd number of sampling points. However, the level of fluctuation reduces when more sampling points are introduced. In Figure 4.5a, the simulation is exact by a 2nd-degree B-spline curve with only 3 sampling points.

Figure 4.6b illustrates the relative error of local cubic fitting, global quadratic and global cubic fitting for $y = x^2$. Additionally, a plot of the relative error of the local quadratic scheme not included in Figure 4.6b to make the trends of the other plots more visible as the relative error of local quadratic scheme is around 10^{-16} for all numbers of sampling points used. This, unsurprisingly, makes the quadratic scheme the most accurate method for simulating $y = x^2$.



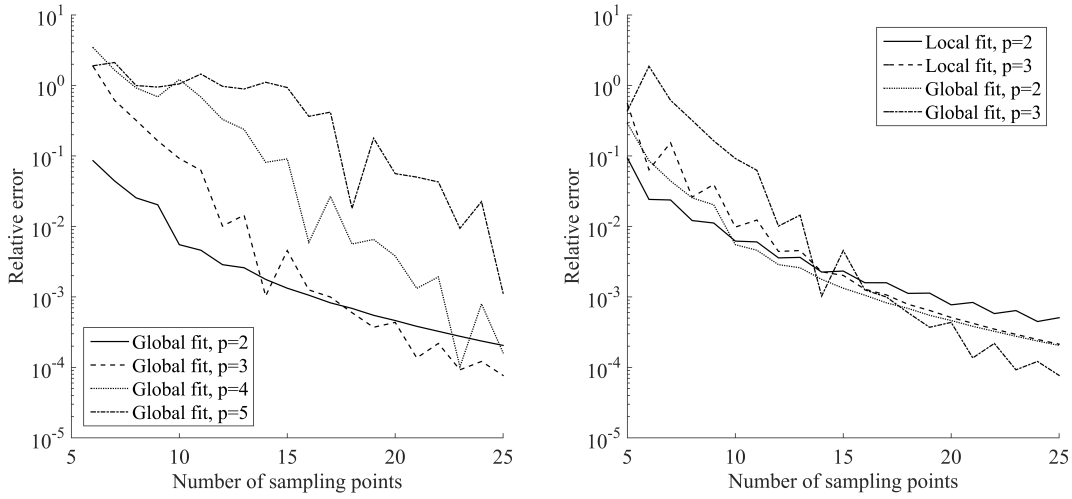
(a) Global interpolation

(b) Global and local interpolation

Figure 4.6: Relative errors regarding the fitting of $y = x^2$.

Similar to the previous example, a cubic interpolation method should give an exact fit for $y = x^3$ with 4 sampling points located at the correct positions. The relative errors of global interpolation of $y = x^3$ by different degrees are also plotted against the number of sampling points (see Figure 4.7a). Sampling points were placed in the same way as in the simulation of $y = x^2$. Again, relative errors decrease for all degrees when the number of sampling points is increased. When more sampling points were introduced, the 3rd-degree B-spline fit gives better accuracy than the 2nd-degree one.

To compare the performance of the global and local methods, relative errors of fittings by 2nd-degree and 3rd-degree global and local methods are plotted together (see Figure 4.7b). When the number of sampling points is increased, the 3rd-degree B-spline with global scheme has the best performance on approximating the curve $y = x^3$.



(a) Global interpolation (b) Global and local interpolation

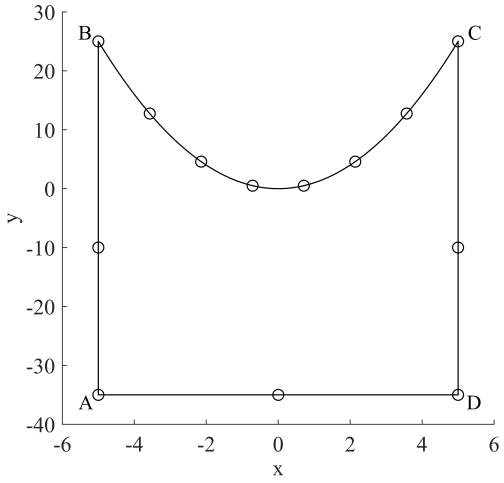
Figure 4.7: Relative errors regarding the fitting of $y = x^3$.

To further test these methods' ability of handling various shapes, sharp corners and sudden turns were introduced into the geometries and approximation again tested. Figure 4.8a is an illustration of the testing shape. Sharp corners occur at point A and D, and curve BC is described by $y = x^2$. Double sampling points were placed at A to start and end the interpolation.

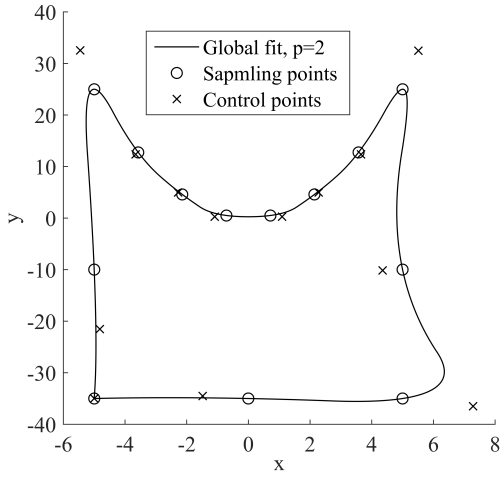
Figure 4.8b and 4.8c show the attempts to approximate the geometry. Circles indicate the sampling points, crosses are the control points and the solid lines are the fitted shape. Although the inflection points at B and C have been handled reasonably well, both 2nd- and 3rd-degree global schemes show their inability to deal with sharp corners and straight lines. The reason behind the asymmetric approximations of the global schemes is that discontinuity is allowed at point A; whereas, the B-spline curve passing through point D is C^1 continuous but C^0 continuity is required to achieve the sharp corners.

Promising results, however, were obtained through local interpolation method (Figure 4.8d and 4.8e). Both the quadratic and the cubic schemes have proved their potential in recovering the sharp corners and straight lines in the original geometry. The quadratic B-spline interpolation appears to handled the sudden turn without a problem and give a slightly better fit; however, when one zooms in around the first sampling point to the right of point B, the result is inaccurate. This indicates that the interpolation achieved by the 2nd-degree B-spline may be affected by the sudden turn feature.

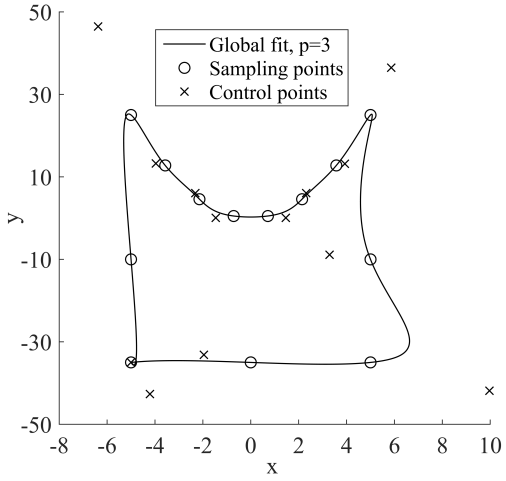
Due to the inability of any global interpolation to simulate sharp corners and straight lines, the method taken forward for boundary representation will be based on a local method. Either quadratic or cubic B-spline representations will be chosen. Apart from the tests on shape recovery, robustness of the two local algorithms has also



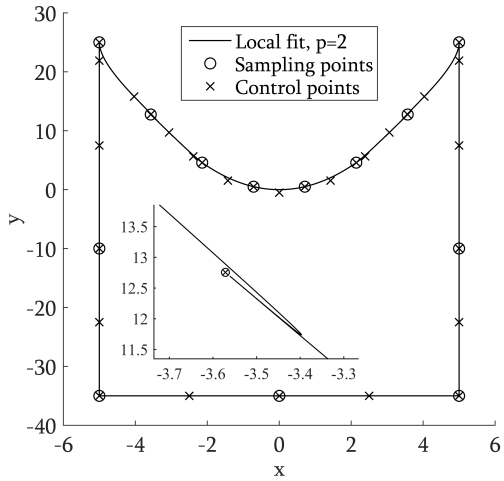
(a) A geometry involving sharp corners and sudden turns



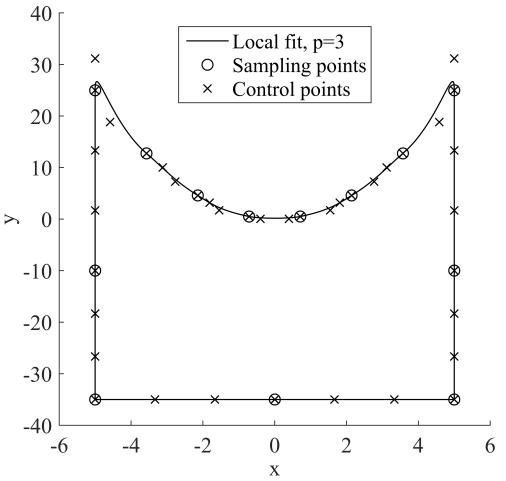
(b) 2nd-degree B-spline global interpolation



(c) 3rd-degree B-spline global interpolation



(d) Quadratic local interpolation



(e) Cubic local interpolation

Figure 4.8: An illustration of the attempted geometry and interpolations of sharp corners.

been tested. As the procedure of the cubic interpolation is more straightforward and no special treatments are required at inflection points, it is also faster than the quadratic interpolation when a small number of sampling points were used. For example, with 10 sampling points used in the example of approximating $y = x^3$, it took the cubic scheme 0.033 seconds which is about a tenth the time consumed by the quadratic scheme. Nevertheless, increasing the number of sampling points does not have a large impact on the time used by the quadratic method, but does for the cubic case. For the same problem, when 1000 sampling points were used, the time consumed by the cubic method increased to 0.65 seconds which turned out to be longer than that by the quadratic approach (0.55 seconds). The reason behind this could be that for every curve segment simulated by the cubic scheme, two control points were calculated; whereas, for most of the curve segments, only one control point was computed by the quadratic method.

4.5 Chapter review

Despite the fact that the quadratic local interpolation has achieved higher accuracy in most of the cases and it is more robust when a large amount of sampling points are used, it has been shown not to handle complicated shapes involving sharp corners well. Additionally, as presented in Section 4.3.1, the local quadratic scheme has many special cases to implement and many of them involve angle calculations (see (4.28) and (4.29)). On the other hand, the cubic scheme has achieved comparable accuracy with a simpler algorithm structure. More importantly, it gives a desirable result when sharp corners and sudden turns are involved in the geometry. Therefore, the method chosen to be used for boundary representation is the cubic local interpolation, and a pseudo-code of the scheme is presented in Figure 4.9.

Now that an appropriate boundary representation method has been selected, using cubic B-splines by choosing appropriate material points as the sampling points. If there are no MPs on the domain boundaries, separate sampling points that do not participate in the stiffness calculation can be inserted along the boundaries. The next chapter will cover the implementation of both Dirichlet and Neumann boundary conditions on boundaries represented by B-splines.

1. Input: n, $\{Q\}$
2. Constant: p
   (a) Calculate $\{q_k\}$ (4.21), $(4.15)_1$, (4.22).
   (b) Identify sharp corners and set flags.
   (c) Calculate tangents $\{T_k\}$.
   - Calculate β_k (4.20).
   - Calculate $\{V_k\}$ $(4.19)_2$.
   - Calculate $\{T_k\}$ $(4.19)_1$.

   (d) Determine the additional control points (4.34).
   (e) Compute the overall control points, $\{P\}$ and knot vector, $\{\Xi\}$
   - IF sharp corners exist
     – Add all sampling points (4.44).
     – Calculate equally spaced ξ_i and form the knot vector (4.45).
   - ELSE
     – Add the ends sampling points to the ends of the additional control points (4.41).
     – Calculate ξ_i (4.42) and form the knot vector (4.43).
   - END IF
3. Output: $\{\Xi\}$, $\{P\}$

Figure 4.9: A pseudo-code of local cubic B-spline interpolation.

Chapter 5

Imposition of Boundary Conditions

Having described a means of boundary representation in Chapter 4, the issue of imposition of BCs can now be addressed. Tractions are applied directly and displacements are prescribed via the IBM [51, 52, 53, 5] which has been briefly discussed in Chapter 3. In order to achieve a high accuracy from the numerical method, integration is performed over the boundaries element by element. Therefore, a search algorithm is required to determine the intersection points of the boundary and the mesh. This chapter will firstly discuss some preliminaries including the search algorithm in Section 5.1 followed by traction boundary condition application and the B-spline based IBM in Sections 5.2 and 5.3.

5.1 Preliminaries

5.1.1 Derivatives of B-spline curves

Recall the definition of a pth-degree B-spline curve, constructed by knot vector $\{\Xi\} = \{0, \ldots, 0, \xi_{p+1}, \ldots, \xi_{r-p-1}, 1, \ldots, 1\}$ and control points $\{P\}$

$$\{C(\xi)\} = \sum_{i=0}^{n} N_{i,p}(\xi)\{P_i\}. \tag{5.1}$$

The derivative of which is

$$\{C'(\xi)\} = \sum_{i=0}^{n} N'_{i,p}(\xi)\{P_i\}, \tag{5.2}$$

where the derivatives of the basis functions $N'_{i,p}(\xi)$ are given by [54]

$$N'_{i,p}(\xi) = \frac{p}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{p}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi). \tag{5.3}$$

Expanding (5.2) by substituting in (5.3), and after rearranging we have

$$\{C'(\xi)\} = p\frac{N_{0,p-1}(\xi)\{P_0\}}{\xi_p - \xi_0} + p\sum_{i=0}^{n-1} N_{i+1,p-1}(\xi)\frac{\{P_{i+1}\} - \{P_i\}}{\xi_{i+p+1} - \xi_{i+1}} + p\frac{N_{n+1,p-1}(\xi)\{P_n\}}{\xi_{n+p+1} - \xi_{n+1}}, \tag{5.4}$$

where the first and the last terms on the right hand side have values of $\frac{0}{0}$ which by definition is 0. So we can further reduce (5.4) to

$$\{C'(\xi)\} = \sum_{i=0}^{n-1} N_{i+1,p-1}(\xi)\{Q_i\}, \quad \text{with} \quad Q_i = p\frac{\{P_{i+1}\} - \{P_i\}}{\xi_{i+p+1} - \xi_{i+1}}, \tag{5.5}$$

which is similar to (5.1). Now, instead of computing the basis functions based on the initial knot vector $\{\Xi\}$, we drop the first and the last knots of $\{\Xi\}$ to form a new knot vector $\{\Xi'\}$,

$$\{\Xi'\} = \{\underbrace{0, \ldots, 0}_{p}, \xi_{p+1}, \ldots, \xi_{r-p-1}, \underbrace{1, \ldots, 1}_{p}\}. \tag{5.6}$$

By using $\{\Xi'\}$, we can easily compute the required basis function as $N_{i,p-1}$, and calculating the B-spline derivative is equivalent to the computation of a $(p-1)$th-degree B-spline curve with modified control points $\{Q\}$, i.e.

$$\{C'(\xi)\} = \sum_{i=0}^{n-1} N_{i,p-1}(\xi)\{Q_i\}. \tag{5.7}$$

5.1.2 B-spline integration

As Gauss-Lagrange quadrature is employed to perform numerical integration, local coordinates ranging from -1 to 1 are required. However, the local coordinate of the B-spline interpolated boundary, ξ, can only have positive values. Therefore, a separate space, called the parent domain [56], is introduced. Figure 5.1 is an illustration of the three spaces: the physical space, the parametric space, and the parent domain. In the physical space, the boundary geometry is described by the global coordinates. The parametric space contains the knots (local coordinates) which run along the curve from one end to the other. The parent domain is a bi-unit element on which numerical integration is performed.

When integrating over a pth-degree B-spline segment, the same rule is applied as used for integrating the same degree polynomials, i.e. using $(p-1)$th order Gauss

quadrature [56]. To perform the integration, the desired curve segment is pulled back from the physical space to the parametric space. In other words, the local coordinates (ξ_j and ξ_{j+1}) of the start and the end point of the segment are identified by using their global coordinates.

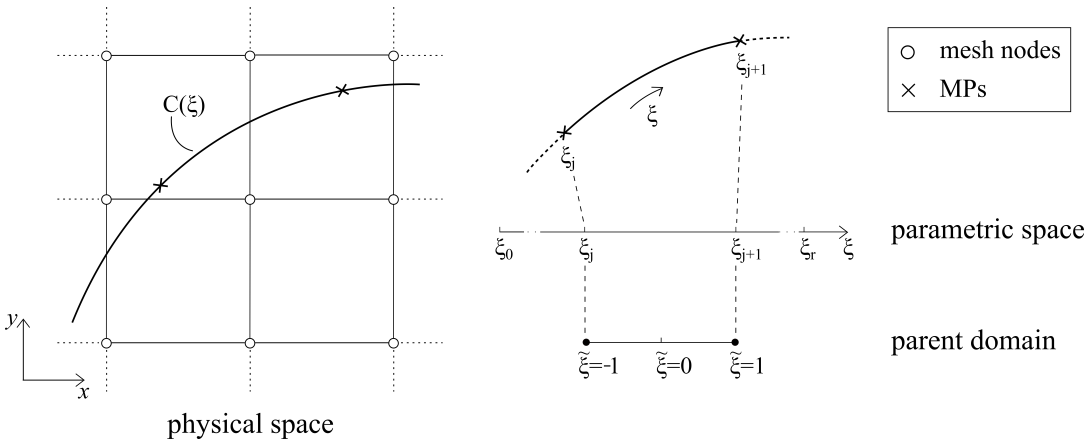


Figure 5.1: An illustration of different spaces.

A linear transformation between the parent element ($[-1, 1]$) and the parametric space ($[\xi_j, \xi_{j+1}]$) is then developed to map the locations of Gauss points. By inspection, we have

$$\xi = \xi_j + \frac{(\tilde{\xi} + 1)(\xi_{j+1} - \xi_j)}{2}. \tag{5.8}$$

So the resulting Jacobian contains two steps of mapping,

$$[J_B] = \left[\frac{d\{C\}}{d\tilde{\xi}}\right] = \left[\frac{d\{C\}}{d\xi}\right]\left[\frac{d\xi}{d\tilde{\xi}}\right], \tag{5.9}$$

where

$$\frac{d\xi}{d\tilde{\xi}} = \frac{\xi_{j+1} - \xi_j}{2}. \tag{5.10}$$

5.1.3 A search algorithm for finding ξ at intersection points

In order to find all the values of ξ at the points where the interpolated boundary intersects with the background mesh, a search algorithm based on the NR iteration has been developed. Figure 5.2 shows a spline that intersects the background mesh at B, C, D, and E. To find the values of B-spline local coordinate ξ at these intersection points, we start from A where $\xi = 0$ and work along the spline towards F where $\xi = 1$.

The element that contains A is firstly located (element 1) and then possible intersections with each edge of the element is identified iteratively. Before performing the iterations the *element* starting point may be changed to a point that is closer to the

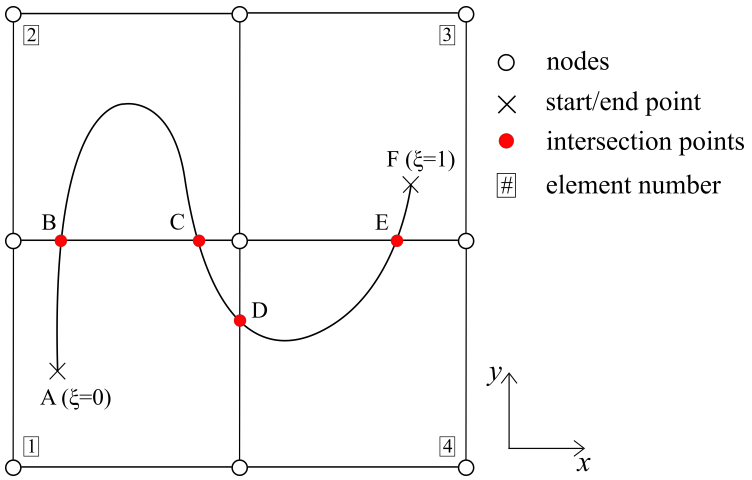


Figure 5.2: A spline passing through a background mesh.

grid to make sure the NR method works properly. This is achieved by dividing the remaining range of ξ into relatively small intervals which is inversely proportional to the boundary length. Adding one interval at a time to the previous ξ until the corresponding global coordinate x (when comparing with vertical edges or y with horizontal edges) is no longer smaller/larger than the upper/lower x (or y) value of the element edge. The resulting point is then used as the *iteration* starting point. If no such point is found when ξ reaches 1, the edge will not be considered for further iteration.

If more than one intersection points other than the element starting point are found, the one with the smallest ξ will be chosen. In element 1, both B and D will be identified in the first round of searching, and B is determined as the first intersection point because the B-spline local coordinate (ξ) of point B has a smaller value. The edge that B rests on also belongs to element 2, so element 2 is the next element in which we search for intersection points. When searching in this element, B is used as the element starting point. As shown in Figure 5.2, the next intersection point rests on the same edge as B. In situations like this, C is easily overlooked from the searching procedure as B will be identified as the only intersection point of that edge. To make sure that C will be found, a small increment is added to the initial ξ and the steps discussed above are followed to double check whether there is another intersection point on that edge. Again, the increment is inversely proportional to the boundary length.

Moving back to element 1, only C and D will be identified in this round of searching and the latter is the point of interest. Followed by searching in element 4, the above procedure is executed and E will be found. The search is terminated in element 3 as the end of the boundary is reached, i.e. there is no intersection point other than the element starting point or, in some cases, the other intersection point has a local value of $\xi = 1$. Figure 5.3 is a flow chart illustrating the procedures of the search algorithm.

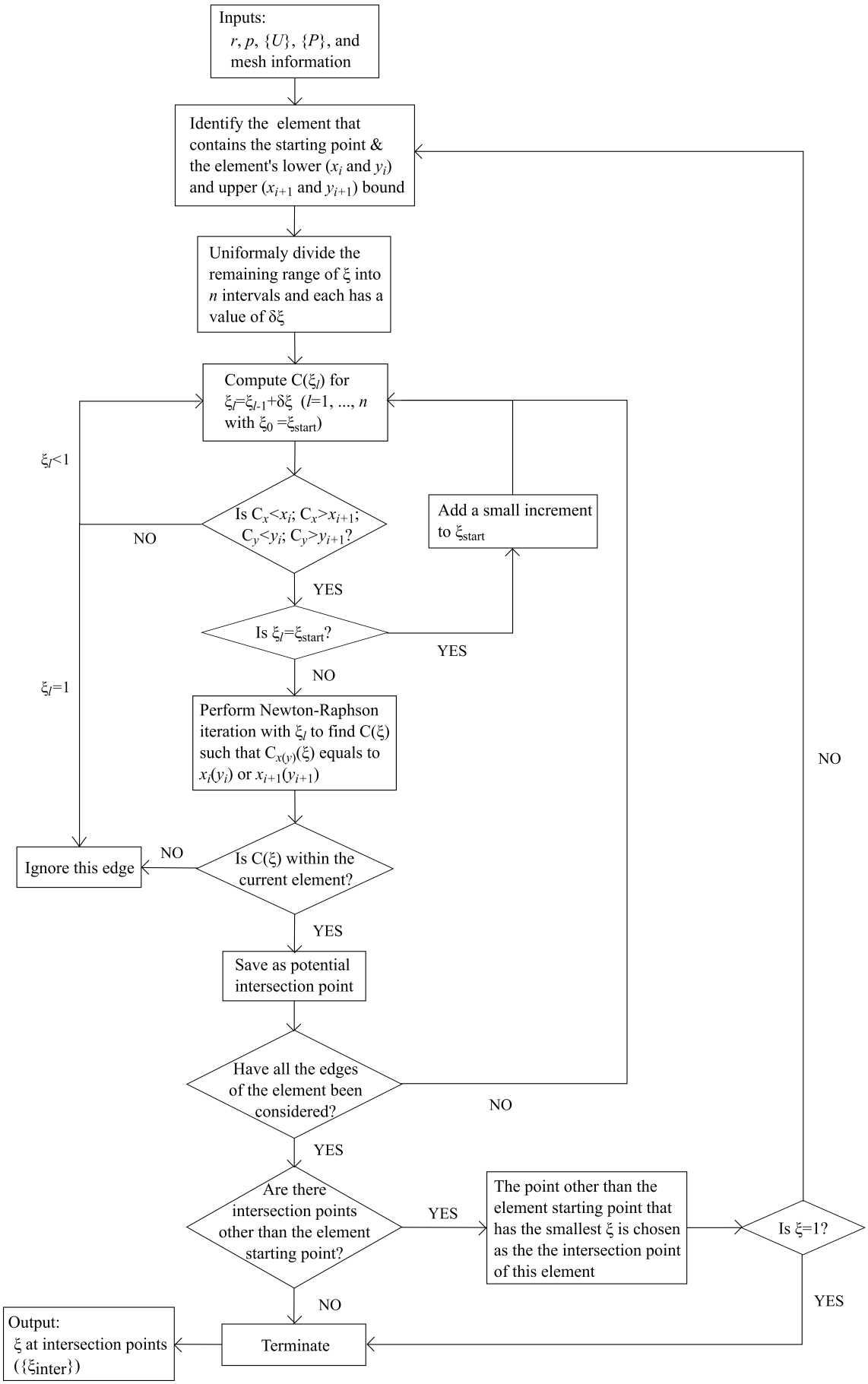


Figure 5.3: B-spline boundaries and the background mesh intersections search algorithm.

5.2 B-spline based Neumann boundary conditions

The application of traction to a boundary is rather straightforward. Recall that

$$\{f^t\} = \int_{\partial\Omega} [M]^T \{t\} \, \mathrm{d}S, \tag{5.11}$$

where $\{f^t\}$ is the resultant nodal force due to traction, $[M]$ contains the standard finite element shape functions and $\{t\}$ is the prescribed traction. Applying Gauss quadrature to (5.11), we have

$$\{f^t\} = \sum_{i=1}^{n_{gp}} [M_i]^T \{t\} \det([J_B]_i) w_i, \tag{5.12}$$

where n_{gp} is the number of Gauss points and w_i is the weight associated with Gauss point i. Because the boundary where the traction is prescribed is described by B-splines, the transformation between the integration domain and the global coordinates is achieved by using $[J_B]$ which is calculated according to (5.9).

Figure 5.4 presents the algorithm for imposing traction boundary conditions, which will be executed just after the start of the load step loop in the MPM algorithm (see

1. Input: $\{\xi_{inter}\}$, $\{t\}$, $\{\Xi\}$, $\{P\}$, mesh information (nodal positions $\{x\}$ and connectivity), $\{\tilde{\xi}\}$, $\{w\}$
2. Constant: p, r, order of Gauss quadrature

   FOR loop over elements ($\{\xi_{inter}\}$ intervals)

   - Identify the values of ξ_j and ξ_{j+1}

     FOR loop over Gauss points (GP)

     - Compute the parametric space coordinate (ξ) of the GP (5.8)
     - Compute the basis functions $[N]$ and the global coordinates $\{C(\xi)\}$ of the GP
     - Identify which element the GP is in and compute the nodal degrees of freedom
     - Compute the finite element shape functions $[M]$ for the GP
     - Compute $\{C'(\xi)\}$ (5.7)
     - Compute $[J_B]$ (5.9) and its determinate
     - Compute $\{f^t\}$ (5.11)

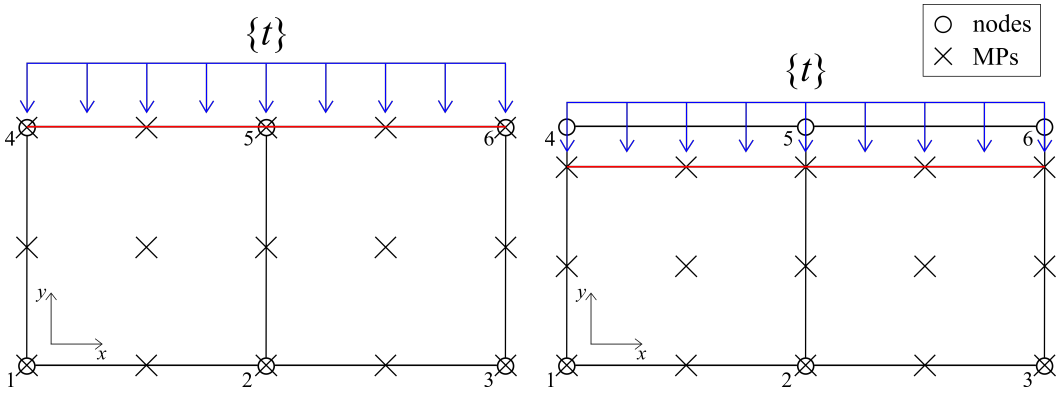
     END

   END
3. Output: $\{f^t\}$

Figure 5.4: A pseudo-code of Neumann boundary conditions.

Figure 2.2) provided that the boundary of interest is approximated by B-splines. Entries in $\{f^t\}$ contributes to the external force vector $\{f\}$ making the reactions $\{f_r\}$ in (2.28) no longer unknowns. This means that the structure displacements $\{d\}$ can now be solved by directly applying LU decomposition or an iterative method to (2.27).

The implementation described above has been tested on a number of benchmark problems which are presented below. The first problem, as shown in Figure 5.5a, comprises a uniform distribution of 15 material points in two four-noded quadrilateral elements to represent a 2m by 1m rectangular domain, the boundaries of which are coincident with the mesh.



(a) The top boundary is coincident with the mesh.

(b) The top boundary is located at 4/5 of the mesh height.

Figure 5.5: Verification of traction boundary condition implementation.

The top boundary (indicated as a red line in Figure 5.5a) is approximated by B-spline and a constant traction of 40MPa is applied in the negative y direction. Using two Gauss points per element, the analytical (f^a) and the numerical (f^h) solutions of the vertical nodal force at each node are shown in Table 5.1. Absolute errors between the two solutions ($|f^h - f^a|$) are calculated to have a more direct comparison. It can be seen from Table 5.1 that the numerical results are very accurate as the errors are in the order of 10^{-14} which is close to the machine precision.

Node No.	1	2	3	4	5	6
f^a (MN)	0	0	0	-10	-20	-10
f^h (MN)	-2.56×10^{-15}	-5.13×10^{-15}	-2.56×10^{-15}	-10.00	-20.00	-10.00
$\lvert f^h - f^a \rvert$ ($\times 10^{-14}$MN)	0.256	0.513	0.256	0.711	2.842	0.711

Table 5.1: Results of the problem shown in Figure 5.5a.

In the second problem (see Figure 5.5b), the problem domain is reduced to 2m by 0.8m. The same mesh was used but the top boundary of the problem was no longer

coincident with the mesh. Having B-spline representation on the top boundary, a constant traction of -40MPa was applied vertically. Using the same number of MPs as the previous problem, the results are shown below in Table 5.2. Absolute errors resulting from the calculation are, again, all within 10^{-14}MN which are considered to be negligible. Therefore, the algorithm of traction application through B-spline and its implementation are valid.

Node No.	1	2	3	4	5	6
f^a (MN)	-4	-8	-4	-16	-32	-16
f^h (MN)	-4.00	-8.00	-4.00	-16.00	-32.00	-16.00
$\lvert f^h - f^a \rvert$ ($\times 10^{-14}$MN)	0.355	0.444	0.400	0.355	2.132	0.355

Table 5.2: Results of the problem shown in Figure 5.5b.

5.3 B-spline based implicit boundary method

The methodology behind the IBM [51, 52, 53] is that essential BCs are enforced by introducing extra stiffness to the system. Dirichlet functions (D-functions) are involved in the solution structure to impose the prescribed displacements directly and integration is performed over a small bandwidth along the boundaries to provide the extra stiffness.

Recall the approximation for displacements from Chapter 2,

$$\{u\} = [M]\{d^e\}, \tag{5.13}$$

where $\{d^e\}$ is the element nodal displacement. Instead of using this simple approximation, the following trial function is employed:

$$\{u'\} = [D]\{u\} + \{u^a\}, \tag{5.14}$$

where $[D]$ is a diagonal matrix that contains the D-functions, i.e. $[D] = diag(D_i)$ $(i = 1, \ldots, n_d)$ and n_d is the dimensionality of the physical problem (in 2D, n_d=2), $\{u\}$ takes the form of (5.13), and $\{u^a\}$ is the essential boundary conditions. For every $\{u_i^a\}$ that is non-zero, the corresponding D-function, $[D_i]$, will have a value of zero to allow the prescribed displacement be enforced.

In the derivation of the MPM presented before, the weighting function, $\{c\}$, was chosen to be

$$\{c\} = [M]\{a\}. \tag{5.15}$$

Instead of assuming $\{a\}$ as some constant arbitrary nodal values, a test function is

introduced,

$$\{a\} = \{\delta u\} = [D]\{\delta d^e\}, \tag{5.16}$$

where $[D]$ includes the same D-functions as the ones used in the trial functions.

Continuing the MPM derivation by substituting the trial and test functions, we arrive at the same weak form expression as (2.19),

$$\int_\Omega [B]^T[D^e][B]\,\mathrm{d}\Omega\,\{d^e\} = \int_{\partial\Omega} [M]^T\{t\}\,\mathrm{d}S + \int_\Omega [M]^T\{f^b\}\,\mathrm{d}\Omega, \tag{5.17}$$

but with D-functions being included in the strain-displacement matrix $[B]$. For 2D problems, $[B]$ is a $3 \times 2n_{en}$ matrix that takes the following form, with $k = 1, \ldots, n_{en}$ and n_{en} is the number of nodes per element,

$$[B_k] = \begin{bmatrix} D_1\dfrac{\partial M_k}{\partial x} + M_k\dfrac{\partial D_1}{\partial x} & 0 \\ 0 & D_2\dfrac{\partial M_k}{\partial y} + M_k\dfrac{\partial D_2}{\partial y} \\ D_1\dfrac{\partial M_k}{\partial y} + M_k\dfrac{\partial D_1}{\partial y} & D_2\dfrac{\partial M_k}{\partial x} + M_k\dfrac{\partial D_2}{\partial x} \end{bmatrix}. \tag{5.18}$$

These D-functions approximate a step function in terms of a distance function ϕ which has a value of zero on the boundary of interest, and $\phi < 0$ indicates the exterior whereas $\phi > 0$ implies the interior of the problem domain.

The essential boundary conditions are imposed by integrating along the boundary across a small bandwidth, δ. When $\phi < 0$, $D(\phi) = 0$ which ensures the exterior would not interfere the analysis; when $\phi > \delta$, $D(\phi) = 1$ which allows $[B]$ returns to its regular form, i.e. contains only the derivatives of the shape functions. Then a smooth transition across the band is required. In [51], Burla *et al.* suggested the following as one of the suitable functions,

$$D(\phi) = \begin{cases} 0 & \phi < 0 \\ 1 - \left(1 - \dfrac{\phi}{\delta}\right)^2 & 0 \le \phi \le \delta \\ 1 & \phi > \delta \end{cases}. \tag{5.19}$$

Additionally, the band of the boundary can also be extended to the outside of the boundary. In this case, $D(\phi)$ takes a value of zero when $\phi < -\delta$ and the transition across a band with a width of 2δ is suggested in [52] as

$$D(\phi) = \frac{1}{2} + \frac{\phi}{2\delta}. \tag{5.20}$$

This gives us another set of step functions

$$D(\phi) = \begin{cases} 0 & \phi < -\delta \\ \frac{1}{2} + \frac{\phi}{2\delta} & -\delta \leq \phi \leq \delta \\ 1 & \phi > \delta \end{cases}. \tag{5.21}$$

However, using these functions to fill the diagonal matrix $[D]$ only enforces fully fixed essential boundary conditions. To achieve "roller" boundary conditions, $D(\phi)$ is assigned to have a value of 1 in the direction that is free to move [5].

Moving on to the stiffness matrix calculation, the strain-displacement matrix is firstly decomposed into two matrices: one contains the gradient of the shape functions and the other contains the gradient of the D-functions. We write

$$[B] = [B_1] + [B_2], \tag{5.22}$$

where

$$[B_{1k}] = \begin{bmatrix} D_1 \frac{\partial M_k}{\partial x} & 0 \\ 0 & D_2 \frac{\partial M_k}{\partial y} \\ D_1 \frac{\partial M_k}{\partial y} & D_2 \frac{\partial M_k}{\partial x} \end{bmatrix} = \begin{bmatrix} D_1 & 0 & 0 & 0 \\ 0 & D_2 & 0 & 0 \\ 0 & 0 & D_1 & D_2 \end{bmatrix} \begin{bmatrix} \frac{\partial M_k}{\partial x} & 0 \\ 0 & \frac{\partial M_k}{\partial y} \\ \frac{\partial M_k}{\partial y} & 0 \\ 0 & \frac{\partial M_k}{\partial x} \end{bmatrix} = [\bar{D}_1][\bar{B}_{1k}], \tag{5.23}$$

and

$$[B_{2k}] = \begin{bmatrix} M_k \frac{\partial D_1}{\partial x} & 0 \\ 0 & M_k \frac{\partial D_2}{\partial y} \\ M_k \frac{\partial D_1}{\partial y} & M_k \frac{\partial D_2}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{\partial D_1}{\partial x} & 0 \\ 0 & \frac{\partial D_1}{\partial y} \\ \frac{\partial D_1}{\partial y} & \frac{\partial D_1}{\partial x} \end{bmatrix} \begin{bmatrix} M_k & 0 \\ 0 & M_k \end{bmatrix} = [\bar{D}_2][\bar{B}_{2k}]. \tag{5.24}$$

This allows the expression of the stiffness matrix be expanded in 2D as

$$\begin{aligned} [K^e] = &\int_S [B_1]^T [D^e][B_1] \, \mathrm{d}S \\ &+ \int_S [B_1]^T [D^e][B_2] \, \mathrm{d}S + \int_S [B_2]^T [D^e][B_1] \, \mathrm{d}S + \int_S [B_2]^T [D^e][B_2] \, \mathrm{d}S, \end{aligned} \tag{5.25}$$

or more concisely as

$$[K^e] = [K_1^e] + \left([K_2^e] + [K_2^e]^T\right) + [K_3^e] = [K_1^e] + [K_{bc}^e], \tag{5.26}$$

where $[K_1^e]$ can be viewed as the regular stiffness matrix when $\delta \to 0$; $[K_2^e]$ and $[K_3^e]$ contribute as the additional stiffness that only exist if the element contains an essential boundary, and $[K_{bc}^e]$ $(= \left([K_2^e] + [K_2^e]^T\right) + [K_3^e])$ is used to indicate the stiffness contributed from the essential boundaries.

As the D-functions and their derivatives are constant along the boundary and the latter is zero outside the band, $[K_2^e]$ and $[K_3^e]$ are only non-zero over this thin area. Hence, we can convert the surface integral into a line integral using a local coordinate system based on the tangent and the normal of the boundary, i.e. (t, n). Providing the essential boundary is parallel to the global coordinates as shown in Figure 5.6a, the penalty stiffness matrices can be evaluated as

$$[K_2^e] = \int_S [B_1]^T [D^e][B_2]\,\mathrm{d}S = \int_t [\bar{B}_1]^T \left(\int_0^\delta [\bar{D}_1][D^e][\bar{D}_2]\,\mathrm{d}n \right) [\bar{B}_2]\,\mathrm{d}t, \tag{5.27}$$

$$[K_3^e] = \int_S [B_2]^T [D^e][B_2]\,\mathrm{d}S = \int_t [\bar{B}_2]^T \left(\int_0^\delta [\bar{D}_2][D^e][\bar{D}_2]\,\mathrm{d}n \right) [\bar{B}_2]\,\mathrm{d}t. \tag{5.28}$$

However, when the essential boundary is inclined with the global coordinates at an angle as shown in Figure 5.6b, transformation between the coordinates is required especially when roller boundary conditions are applied. The transformation matrix has been shown in [5] to be

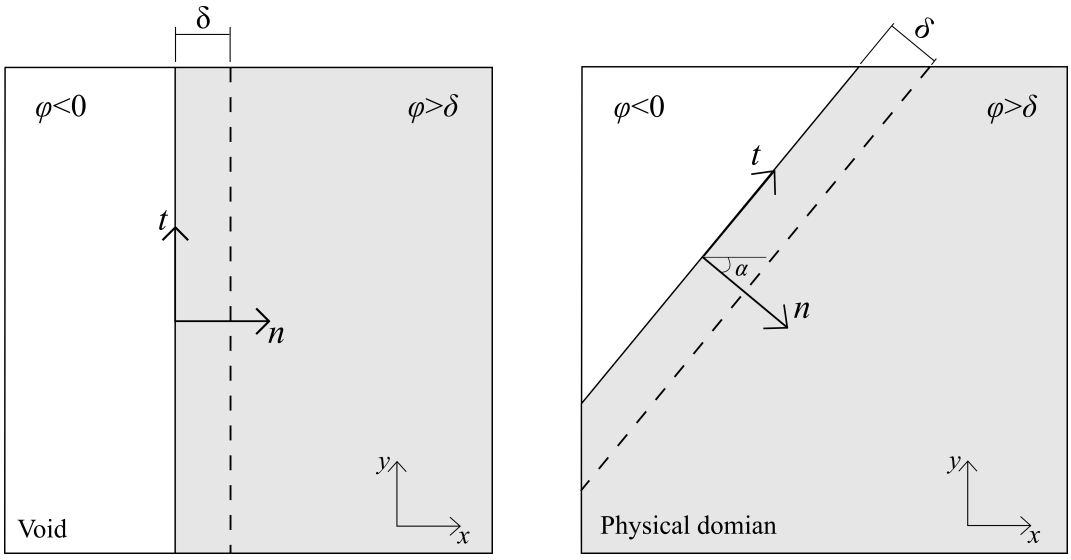
$$[J_T] = \begin{bmatrix} \dfrac{\partial n}{\partial x} & \dfrac{\partial n}{\partial y} \\ \dfrac{\partial t}{\partial x} & \dfrac{\partial t}{\partial y} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \tag{5.29}$$

where θ is the angle required to align direction n with x and t with y. One thing to pay attention to is that the angle θ can take any value between -2π to 2π depending on one's preferred calculation method. In the case shown in Figure 5.6b, the rotation angle θ can either be $-|\alpha|$ or $(2\pi - |\alpha|)$ as long as the local coordinates are aligned with the global one after (5.29) is performed.

Because $[\bar{D}_1]$ only contains the D-functions which are invariant with respect to the coordinate system while $[\bar{D}_2]$ is populated by the derivatives of the D-functions with respect to (x, y), and the latter has a significantly larger value than the former; $[J_T]$ is only applied to transform $[\bar{D}_2]$. So in general, $[K_2^e]$ and $[K_3^e]$ can be determined as

$$[K_2^e] = \int_S [B_1]^T [D^e][B_2]\,\mathrm{d}S = \int_t [\bar{B}_1]^T \left(\int_0^\delta [\bar{D}_1][D^e][\bar{D}_2][J_T]\,\mathrm{d}n \right) [\bar{B}_2]\,\mathrm{d}t, \tag{5.30}$$

$$[K_3^e] = \int_S [B_2]^T [D^e][B_2]\,\mathrm{d}S = \int_t [\bar{B}_2]^T \left(\int_0^\delta [J_T]^T [\bar{D}_2][D^e][\bar{D}_2][J_T]\,\mathrm{d}n \right) [\bar{B}_2]\,\mathrm{d}t. \tag{5.31}$$



(a) Parallel to the global coordinates. (b) Inclinde with the global coordinates.

Figure 5.6: Implicit boundary coordinate system.

Roller boundary conditions permitting the material to have free movement in the tangential direction requires D_2 to have a constant value of 1 throughout the band, while D_1 depends on the distance function and vice versa.

Incorporating the IBM with the B-spline represented boundaries only requires a separate calculation of $[K_2^e]$ and $[K_3^e]$ in addition to $[K_1^e]$ which is evaluated via the procedures discussed in Chapter 2. In most of the examples shown later, both additional stiffness matrices are evaluated by using a 2-point Gauss quadrature scheme as illustrated in Figure 5.7 for one element. The boundary Gauss points (indicated in red) are responsible for the outer integral of (5.30) and (5.31). Then, for every boundary Gauss point, two more Gauss points (indicated in blue) are placed across the band along the corresponding normal direction to perform the inner integral. For more complicated boundary geometries, such as inclined boundaries and curved boundaries, it is necessary to increase the number of Gauss points, especially across the bandwidth as the function under approximation has a very sharp gradient.

Upon applying Gauss quadrature, $[K_2^e]$ and $[K_3^e]$ can be approximated as

$$[K_2^e] = \sum_{i=1}^{n_{gp1}} [\bar{B}_1]^T \left(\sum_{j=1}^{n_{gp2}} [\bar{D}_1]^T [D^e] [\bar{D}_2] [J_T] \det([J_D]_j) w_j \right) [\bar{B}_2] \det([J_B]_i) w_i, \quad (5.32)$$

and

$$[K_3^e] = \sum_{i=1}^{n_{gp1}} [\bar{B}_2]^T \left(\sum_{j=1}^{n_{gp2}} [J_T]^T [\bar{D}_2]^T [D^e] [\bar{D}_2] [J_T] \det([J_D]_j) w_j \right) [\bar{B}_2] \det([J_B]_i) w_i, \quad (5.33)$$

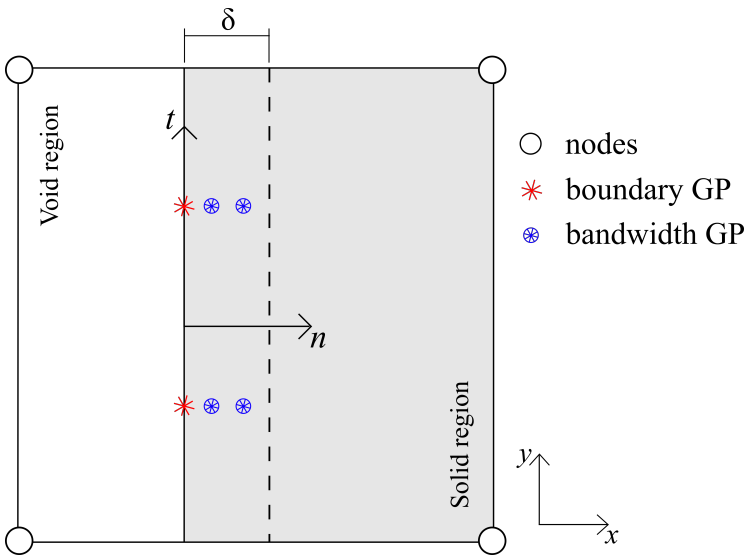


Figure 5.7: Integration scheme along a B-spline represented implicit boundary.

respectively. $[J_T]$ is the transformation matrix formulated by (5.29); $[J_D]$ provides the mapping between the implicit boundary coordinate n and the Gauss quadrature coordinate; $[J_B]$ is the link between the global coordinates and the boundary integration domain and is calculated by (5.9); w indicates the weight associated with the Gauss points, and n_{gp1} and n_{gp2} are the number of Gauss points along the boundary and across the band respectively.

Inhomogeneous Dirichlet BCs are imposed simply by computing the element reaction forces, $\{f^d\}$, due to the prescribed nodal displacements $\{u^e\}$ as

$$\{f^d\} = [K^e_{bc}]\{u^e\}. \tag{5.34}$$

These reaction forces then contribute to the structure external force vector $\{f\}$. The introduction of $\{f^d\}$ allows the structure displacements be solved in the same fashion as that in the enforcement of Neumann BCs.

One thing to note here is that in order to compute the angle θ within $[J_T]$, the inward normal, $\{n\}$, of the B-spline boundary at the Gauss point positions is required. $\{n\}$ of a point with local coordinate ξ and global coordinates $\{C(\xi)\} = [C_x, C_y]$ on a B-spline curve takes the following form [89],

$$\{n\} = \begin{Bmatrix} n_x \\ n_y \end{Bmatrix} \quad \text{with} \quad n_x = \frac{1}{J(\xi)}\frac{\mathrm{d}C_y}{\mathrm{d}\xi}, \quad n_y = \frac{-1}{J(\xi)}\frac{\mathrm{d}C_x}{\mathrm{d}\xi}. \tag{5.35}$$

The derivatives in $(5.35)_2$ and $(5.35)_3$ are calculated via the procedures discussed in Section 5.1.1, and $[J(\xi)]$ is defined as

$$J(\xi) = \sqrt{\left(\frac{\mathrm{d}C_x}{\mathrm{d}\xi}\right)^2 + \left(\frac{\mathrm{d}C_y}{\mathrm{d}\xi}\right)^2}. \tag{5.36}$$

Figure 5.8 is a pseudo-code computing the additional stiffness matrices and $\{f^d\}$ of an element, in which $[K^e_{2n}]$ and $[K^e_{3n}]$ are the approximation of the inner integral of (5.30) and (5.31).

1. Input: $\{\xi_{inter}\}$, $\{\Xi\}$, $\{P\}$, mesh information (nodal positions $\{x\}$ and connectivity), $\{\tilde{\xi}\}$, $\{w\}$
2. Constant: p, r, order of Gauss quadrature

   FOR loop over elements ($\{\xi_{inter}\}$ intervals)

   - Initialise $[K^e_2]$ and $[K^e_3]$
   - Identify the values of ξ_j and ξ_{j+1}

     FOR loop over boundary Gauss points

     - Compute the parametric space coordinate (ξ) of the GP (5.8)
     - Compute the basis functions $[N]$ and the global coordinates $\{C(\xi)\}$ of the GP
     - Identify which element the GP is in and compute the nodal degrees of freedom
     - Compute the finite element shape functions $[M]$ for the GP
     - Form $[\bar{B}_1]$ (5.23) and $[\bar{B}_2]$ (5.24)
     - Compute $\{C'(\xi)\}$ (5.7)
     - Compute $\{n\}$ at the GP $(5.35)_1$ and θ
     - Form $[J_T]$ (5.29)
     - Initialise $[K^e_{2n}]$ and $[K^e_{3n}]$

       FOR loop over Gauss points across the band

       - Map the GP's coordinate to boundary local coordinate (n)
       - Compute D-functions and their derivatives with respect to the boundary local coordinates
       - Form $[\bar{D}_1]$ (5.23) and $[\bar{D}_2]$ (5.24)
       - Compute $[K^e_{2n}]$ (5.32) and $[K^e_{2n}]$ (5.33)

       END

     - Compute $[J_B]$ (5.9) and its determinate
     - Calculate $[K^e_2]$ (5.32) and $[K^e_3]$ (5.33)

     END

   - Compute the non-zero entries in $[K^e_{bc}] = [K^e_2] + [K^e_2]^T + [K^e_3]$
   - Compute the element reaction force $\{f^d\}$ (5.34)
   - Store $[K^e_{bc}]$ in compressed column storage or compressed row storage

   END
3. Output: $[K_{bc}]$, $\{f^d\}$

Figure 5.8: A pseudo-code of computing $[K_{bc}]$ and $\{f^d\}$ for an element by B-spline based IBM.

This procedure is also executed at the beginning of the load step loop just after the B-spline approximation of the corresponding boundary is obtained. Instead of testing the implementations of this B-spline based IBM through a set of benchmark problems, it is validated with examples shown in the next chapter.

5.4 Chapter review

Details of the application of BCs in the MPM have been fully addressed in this chapter. Both Neumann and Derichlet boundary conditions can now be easily applied to any B-spline represented boundaries; the external forces evaluated in the MPM are no longer limited to body forces and the application of displacement BCs does not require mesh conformed boundaries. The enforcement of tractions involves computing the nodal forces through integrating over the boundaries, and displacements are prescribed by the B-spline based IBM. Validations of the implementations of these BCs are presented by a number of problems in the next chapter.

This is an intended blank page.

Chapter 6

Numerical Examples

This chapter will present a series of numerical examples to verify the method of imposing BCs described in the previous chapter. All examples are in 2D under plane strain and use an infinitesimal strain assumption with linear elastic materials which have a Young's modulus of 1MPa and Poisson's ratio of 0.25 (unless otherwise stated). Additionally, body forces are ignored.

The performance of the methods is measured by errors between the numerical results and the analytical solutions. Relative stress error R_{σ_i} at a MP i is computed as

$$R_{\sigma_i} = \frac{|(\sigma_i^h - \sigma_i^a)|}{|\sigma_i^a|}, \tag{6.1}$$

where σ^h and σ^a are the numerical and analytical stresses, and $|(\cdot)|$ is the absolute value of $(\cdot)$. Similarly, error of displacements at a MP can be calculated as

$$R_{d_i} = \frac{|(d_i^h - d_i^a)|}{|d_i^a|}, \tag{6.2}$$

where d^h and d^a are the numerical and analytical displacements. Note that u and v are used later to indicate horizontal and vertical displacements respectively. Absolute errors are used when the analytical results contain zero values in the domain, those are

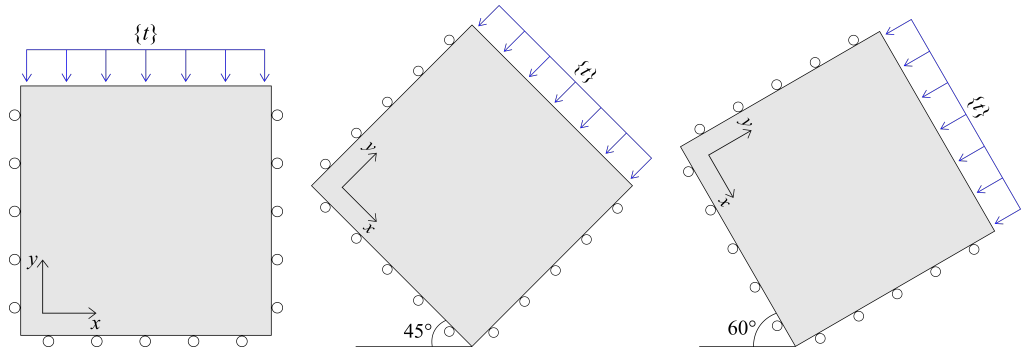
$$A_{\sigma_i} = |(\sigma_i^h - \sigma_i^a)| \quad \text{and} \quad A_{d_i} = |(d_i^h - d_i^a)|. \tag{6.3}$$

To provide an average error over the problem domain, measures at each MP are summed together then divided by the total number of MPs.

6.1 Compression of a 2D square domain

Examples in this section investigate the performance of the boundary conditions applied via B-spline approximated boundaries in a simple 1D compression problem.

The configurations used of this problem are shown in Figure 6.1 and the size of the problem domain will be specified at the beginning of each simulation. Three sides of the square are subjected to roller BC and a constant traction of 8000Pa is imposed perpendicular to the fourth edge over a single load step. The roller BCs were firstly enforced to the 0°model on the mesh directly by conventional means and the traction was applied as described in Chapter 5. For all three models, the B-spline based IBM for imposing HDBCs is introduced when the boundaries of the problem domain no longer coincide with the background mesh. An initial value of 10^{-6} (with the same units as the problem under analysis) was assigned to the bandwidth δ and the relationship between the accuracy of the result and δ was examined. Additionally, the ability of the B-spline based IBM on enforcing IDBCs was tested with the 45°and 60°models.



(a) 0° model. (b) 45° model. (c) 60° model.

Figure 6.1: Uniaxial compression of a square.

6.1.1 Conventional Dirichlet and B-spline Neumann

Figure 6.2 shows the initial discretisation of the problem shown in Figure 6.1a with a domain size of 1m by 1m. A 2 by 2 background mesh made of quadrilateral elements with 4 MPs per element was used to model the domain. These MPs were positioned such that they have the same distances a between the adjacent MPs and half a between the mesh and MPs that are located closest to the mesh. All boundaries of the square are coincident with the mesh boundaries. Imposition of the roller BCs is embedded in the standard MPM algorithm; whereas, 5 points that do not participate in the stiffness integrations were inserted along the boundary (marked by the red line in Figure 6.2) to form a B-spline description of the boundary, on which the traction was applied by following the algorithm shown in Figure 5.4.

The average errors of this initial discretisation are shown in Table 6.1. With only 16 MPs, these small errors indicate that the traction boundary conditions have been implemented correctly. Small variations are observed in the errors due to machine precision while the number of MPs increases, which is reasonable given that the problem under analysis is simple and the background mesh is coarse.

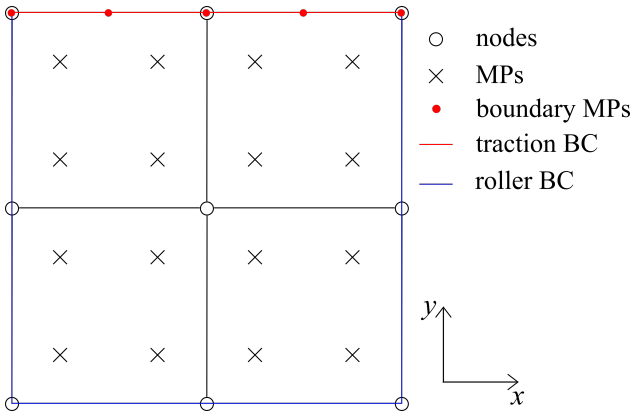


Figure 6.2: Initial discretisation of the problem domain.

Error	$\bar{R}_{\sigma_{xx}}$	$\bar{R}_{\sigma_{yy}}$	$\bar{A}_v$
Value	1.25×10^{-6}	3.70×10^{-16}	1.67×10^{-5}m

Table 6.1: Conventional Dirichlet and B-spline Neumann: average errors of the initial discretisation.

6.1.2 Implicit HDBCs and B-spline Neumann

As the previous problem is a constant stress problem with linear deformation, the size of the problem domain does not have an impact on the comparison of results. To test the B-spline based IBM, the same mesh and initial discretisation as illustrated in Figure 6.2 were used; however, the problem boundaries now confirms with the outer layer of the MPs as shown in Figure 6.3. Because of this modification, weights of the MPs on the boundaries were adjusted accordingly (MPs at corners had weights equal to a quarter of their original weights, and other boundary MPs had half of their original weights).

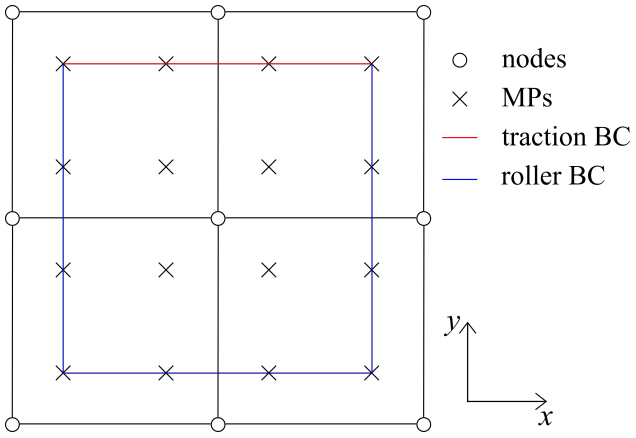
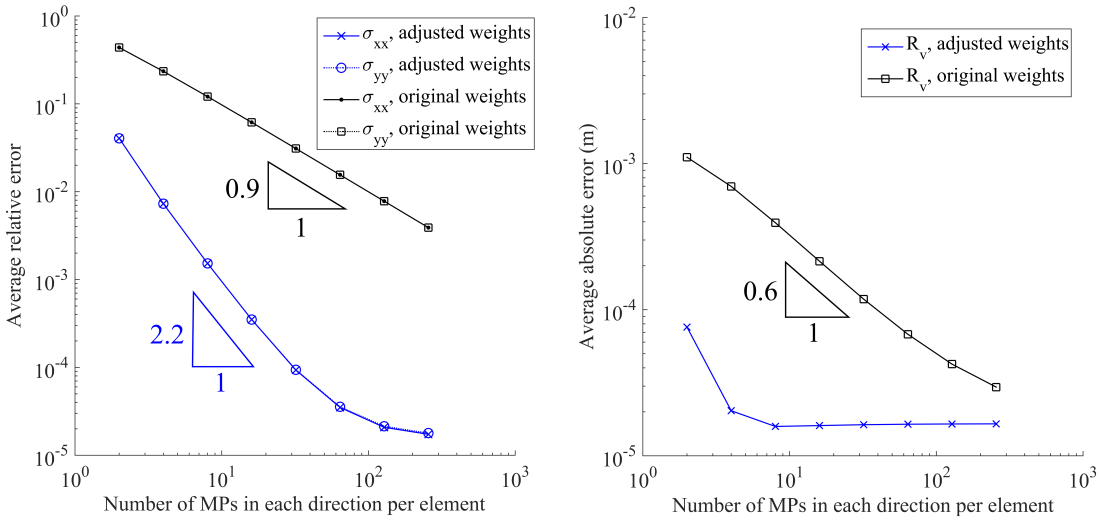


Figure 6.3: Initial discretisation of the 0° model with boundaries coincident with the outer layer of the MPs.

Convergence plots of average stress and displacement errors with respect to the

number of MPs in each direction per element are shown in Figure 6.4. Both stresses and displacement show convergence which means the implicit HDBCs are imposed correctly; however, there is no specific meaning behind the convergence rate as there is no physical link between the convergence rate and the number of MPs. Specifically, increasing the number of MPs reduces integration errors associated with non-optimum integration point positioning. The simulations with adjusted boundary particles' weights exhibit smaller errors and a faster convergence rate than the ones with original weights. This is because that the areas associated with the boundary MPs are being represented more accurately with adjusted weights, which means a higher precision of the stiffness integration is achieved. In both Figure 6.4a and 6.4b, errors represented by the blue lines reach constant values close to 10^{-5}. This is because the problem domain expands when number of MPs increases, and the maximum size of the problem domain is limited by the background mesh due to the way by which the MPs were generated. This means that the errors converge towards the results shown in Table 6.1.

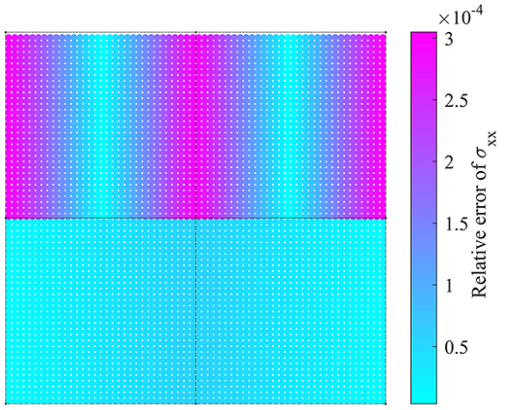


(a) $\bar{R}_{\sigma_{xx}}$ and $\bar{R}_{\sigma_{yy}}$. (b) $\bar{A}_v$.

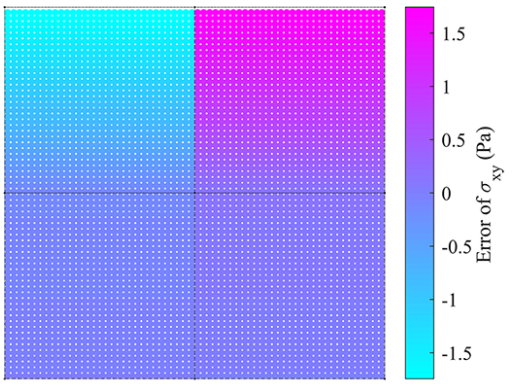
Figure 6.4: Implicit HDBCs with B-spline Neumann (0° model): convergence plots of errors vs. number of MPs in each direction per element.

A close examination of the errors obtained with 32^2 MPs per element is presented in Figure 6.5a–6.5c. The raw error, $(\sigma_i^h - \sigma_i^a)$, plot of σ_{xy} provides insight into the distribution of $R_{\sigma_{xx}}$. As shown in Figure 6.5b, shear stress in the domain has both negative and positive values on the upper left and right elements respectively. In other words, the existing shear force tries to split the square vertically from the middle, which explains why σ_{xx} and σ_{yy} have larger errors toward the middle of the square. As for why the analysis gives non-zero shear stress from a vertically applied traction is most likely because that the mesh was not coincident with the geometry. As a result, the sum of the forces at the top corner nodes has a value that is slightly less than a half of the total force applied, which leads to an uneven deformation of

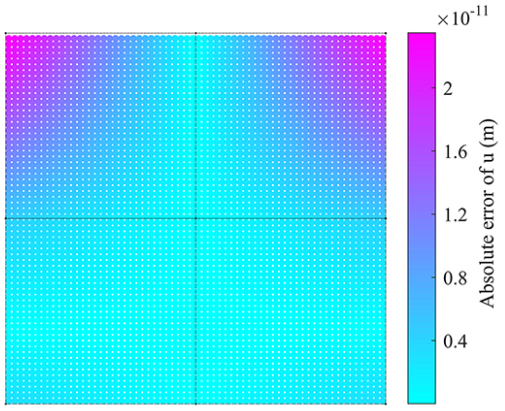
the background mesh. This causes errors in the displacement field, especially at the top corners of the square (see Figure 6.5c), and eventually leads to non-zero shear values. To further show that this is the case, an additional plot of $R_{\sigma_{xx}}$ with the same number of MPs but with the HDBC and traction BC imposed coinciding with the mesh was produced. As shown in Figure 6.5d, a rather even error distribution throughout the material is obtained.



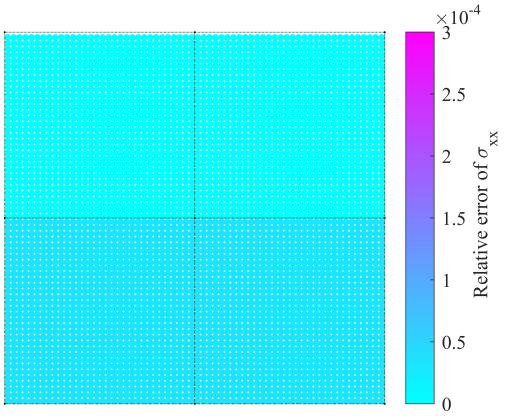
(a) $R_{\sigma_{xx}}$.



(b) Raw error of σ_{xy}.



(c) A_u.



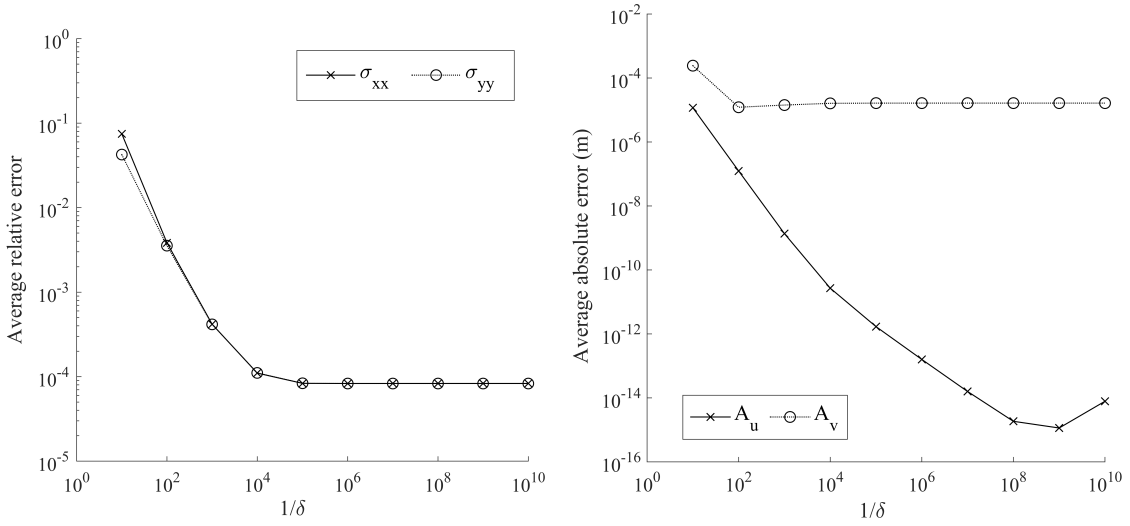
(d) $R_{\sigma_{xx}}$ with BCs imposed on mesh.

Figure 6.5: Implicit HDBCs with B-spline Neumann (0° model): error distribution plots with 32^2 MPs per element.

Figure 6.6a and 6.6b illustrate the convergence of stress and displacement errors for the 0° model with 32^2 MPs per element with respect to δ. Errors of σ_{xx} and σ_{yy} converge to a value of 8.30×10^{-6}, and $\delta < 10^{-6}$m does not reduce the stresses errors. $\bar{A}_u$ and $\bar{A}_v$ reach their minimum with a value of 1.22×10^{-5} when $\delta = 10^{-2}$m and 1.15×10^{-15} when $\delta = 10^{-9}$m respectively. Vertical displacement error settles to 1.64×10^{-5} for $\delta < 10^{-5}$; whereas, horizontal displacement error starts increasing after the minimum because of ill conditioning of the structure stiffness matrix.

The same analysis was performed using various number of MPs. It was found that

to achieve the minimum errors for the stresses, the optimum bandwidth needs to be reduced when the number of MP increases. However, as shown in Figure 6.4a, the stress errors reach a constant value when 64^2 MPs per element were used. Hence, a small difference in the bandwidth used does not significantly influence the result. As for displacement errors, there is little variation in the optimum bandwidth when the number of MPs was changed. Therefore, a δ value between 10^{-6} and 10^{-9} with the same units as the problem is recommended.



(a) $\bar{R}_{\sigma_{xx}}$ and $\bar{R}_{\sigma_{yy}}$. (b) $\bar{A}_u$ and $\bar{A}_v$.

Figure 6.6: Implicit HDBCs with B-spline Neumann (0° model): convergence plots of errors obtained with 32^2 MPs per element vs. δ.

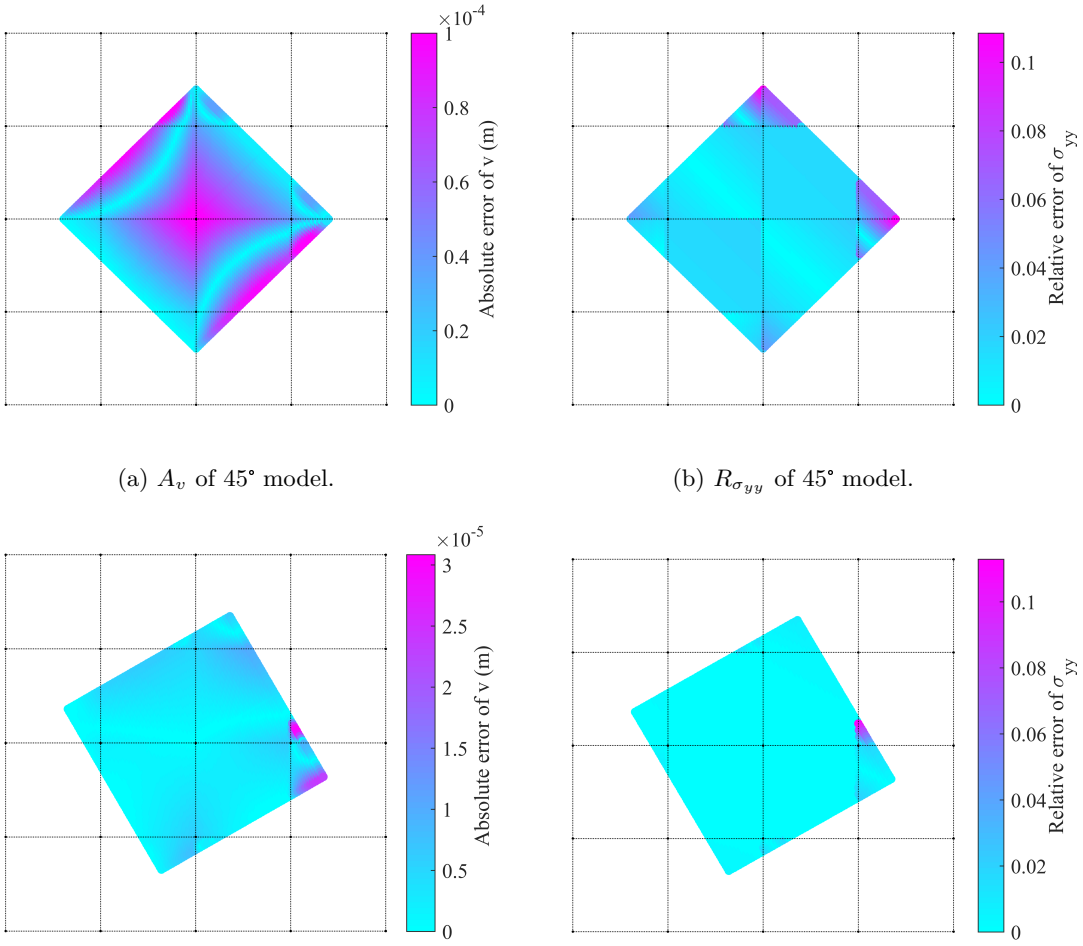
To test the ability of the B-spline based IBM on modelling inclined boundaries, analyses of the problems shown in Figure 6.1b and 6.1c were undertaken. A 4 by 4 background mesh (with 1m by 1m elements) was used to analyse the rotated geometries. The size of the problem domain was limited to 2m by 2m. Traction with the same magnitude was applied perpendicular to the top right boundary and roller BCs were applied to the other boundaries.

The average absolute errors of displacement v and stress σ_{yy} obtained by using 64^2 MPs for the two inclined models are shown in Table 6.2. Although values of $\bar{A}_v$ are reasonable, values of $\bar{R}_{\sigma_{yy}}$ are significantly larger than seen previously. To further examine this issue, error distributions of v and σ_{yy} are plotted in Figure 6.7. For both models, high errors are located around the edges, especially near the edge where traction was applied, and corners of the problem domain where MPs are concentrated at a small portion of the background element. Even though a large number of MPs were used in the discretisation, the poor positioning of the MPs within each element leads to poor accuracy of the stiffness integration of the background mesh element.

The B-spline based IBM and traction imposition were tested further by using the

Model	45°	60°
$\bar{A}_v$ (m)	3.88×10^{-5}	2.68×10^{-6}
$\bar{R}_{\sigma_{yy}}$	0.015	0.0014

Table 6.2: Implicit HDBCs with B-spline Neumann on inclined boundaries: $\bar{A}_v$ and $\bar{R}_{\sigma_{yy}}$ with 32^2 MPs per element.



(a) A_v of 45° model.

(b) $R_{\sigma_{yy}}$ of 45° model.

(c) A_v of 60° model.

(d) $R_{\sigma_{yy}}$ of 60° model.

Figure 6.7: Implicit HDBCs with B-spline Neumann on inclined boundaries: error distribution plots with 32^2 MPs per element.

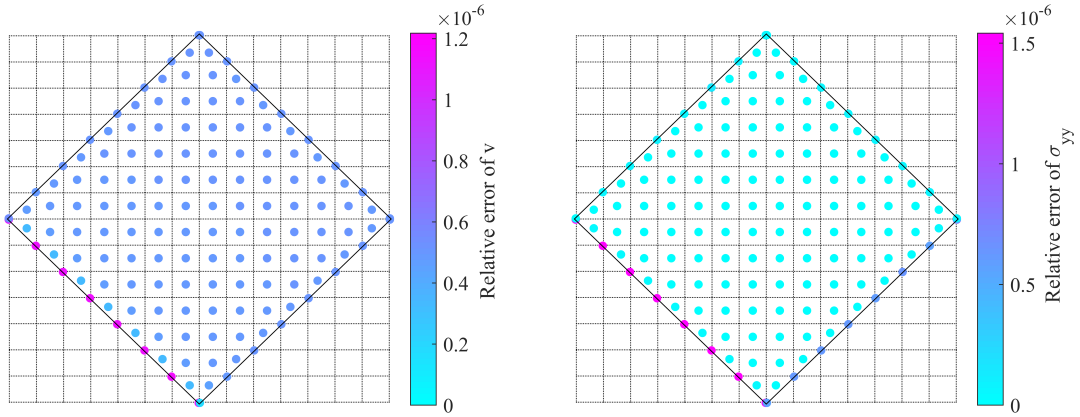
discretisation of [5] with a problem domain size of 100^2mm. This descretisation uses a background mesh of 14^2 10mm by 10mm elements and one MP per element was used to discretise the geometry. Additional points were introduced along the boundaries to form B-spline descriptions of the boundaries. Note that these points were assigned zero weights, which means that they did not take part in the material stiffness calculation. The same BCs as the previous example were enforced.

Average relative errors of displacement v and stress σ_{yy} for both models using the

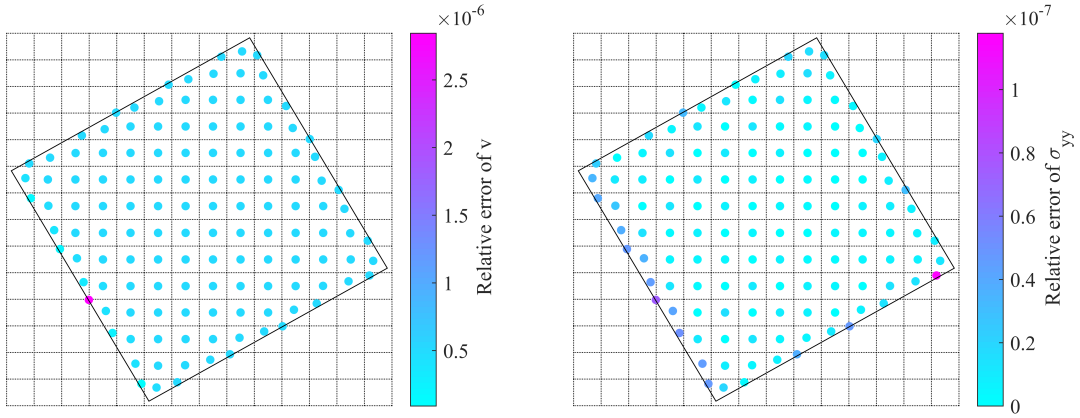
new discretisation are shown in Table 6.3. Values of $\bar{R}_v$ and $\bar{R}_{\sigma_{yy}}$ are reduced compared to the errors obtained by the previous discretisation for both models, especially $\bar{R}_{\sigma_{yy}}$. Error distributions illustrated in Figure 6.8 indicate that both roller and traction BCs have been applied successfully on the inclined boundaries. This means that, in addition to poor stiffness integration, errors in the previous discretisation are due to the relative size of the mesh to the domain and the number of MPs.

Model	45°	60°
$\bar{R}_v$	5.20×10^{-7}	4.98×10^{-7}
$\bar{R}_{\sigma_{yy}}$	1.35×10^{-7}	1.16×10^{-8}

Table 6.3: Implicit HDBCs with B-spline Neumann on inclined boundaries: $\bar{R}_v$ and $\bar{R}_{\sigma_{yy}}$ with descretisation of [5].



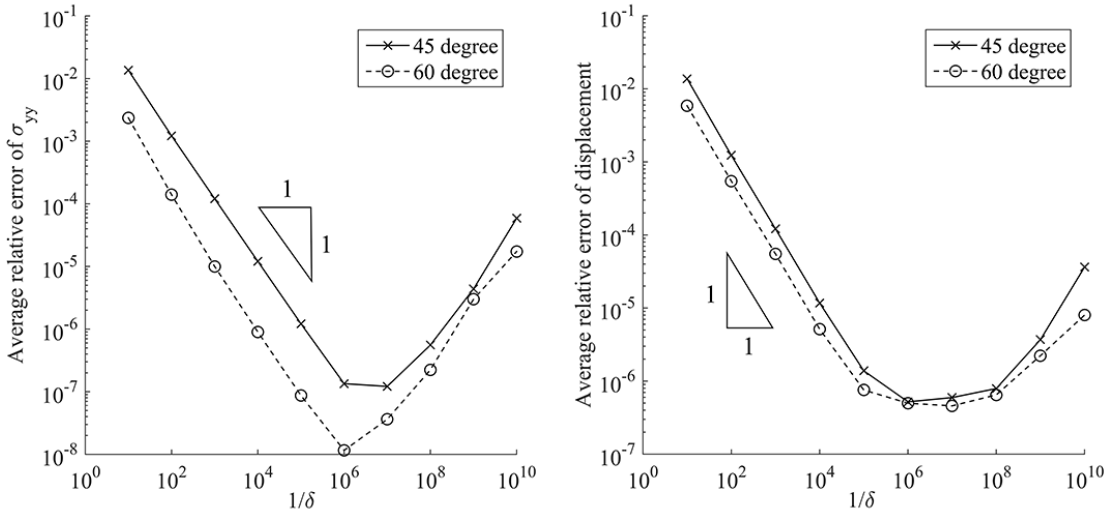
(a) R_v of 45° model. (b) $R_{\sigma_{yy}}$ of 45° model.



(c) R_v of 60° model. (d) $R_{\sigma_{yy}}$ of 60° model.

Figure 6.8: Implicit HDBCs with B-spline Neumann on inclined boundaries: error distribution plots with descretisation of [5].

Furthermore, convergence analyses of the displacement and stress errors with respect to the bandwidth δ for both rotated models were performed and the results are shown in Figure 6.9. Average stress error, $\bar{R}_{\sigma_{yy}}$, of the 45° model reaches its minimum when $\delta = 10^{-7}$mm; whereas, $\bar{R}_v$ reaches its minimum with $\delta = 10^{-6}$mm. For the 60° model, the optimum bandwidth for σ_{yy} is 10^{-6}mm and that for v is 10^{-7}mm. All errors converge at the same rate towards their minimum and then diverge after the optimum bandwidth. The divergence shown is due to the increasing condition number of the stiffness matrix as the bandwidth reduces. Therefore, a bandwidth of 10^{-6} or 10^{-7} with the same unit as the problem is recommended.



(a) $\bar{R}_{\sigma_{yy}}$. (b) $\bar{R}_v$.

Figure 6.9: Implicit HDBCs with B-spline Neumann on inclined boundaries: error convergence plots vs. bandwidth δ.

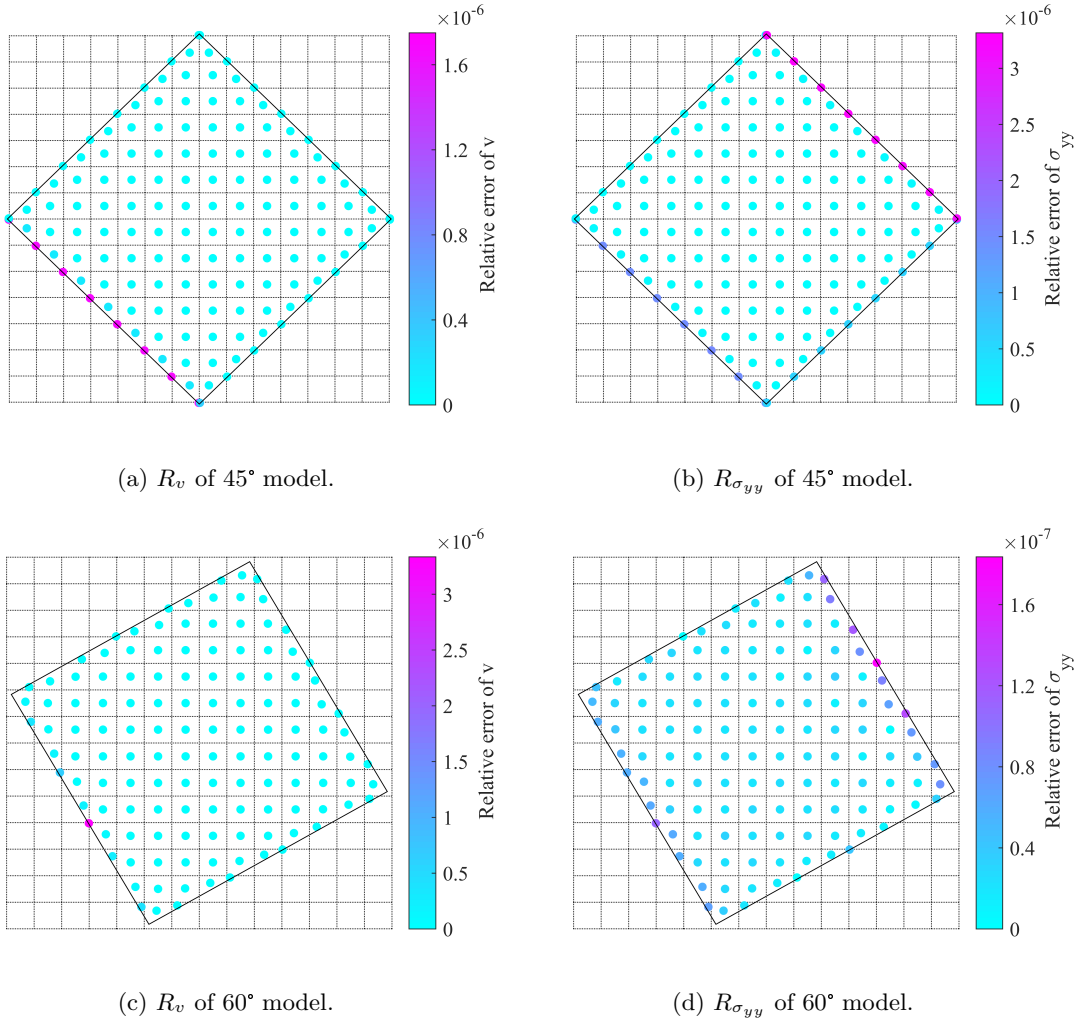
6.1.3 Implicit IDBCs

To demonstrate the ability of the B-spline based IBM to impose non-zero displacement boundary conditions, the traction applied to the top right boundary of the 45°and 60°models is now replaced by a constant displacement of 1mm. The discretisation of [5] was employed and a bandwidth of 10^{-6}mm was used. Average relative errors of displacement v and stress σ_{yy} are listed in Table 6.4.

Model	45°	60°
$\bar{R}_v$	1.13×10^{-7}	5.65×10^{-8}
$\bar{R}_{\sigma_{yy}}$	3.19×10^{-7}	3.15×10^{-8}

Table 6.4: Imposition of IDBCs: $\bar{R}_v$ and $\bar{R}_{\sigma_{yy}}$.

Values of $\bar{R}_v$ and $\bar{R}_{\sigma_{yy}}$ for both angles of inclination are analogous to the results obtained with tractions applied, which means that the IDBC is imposed correctly by the B-spline based IBM. To further demonstrate this, error distributions of v and σ_{yy} are presented in Figure 6.10. Higher displacement errors are concentrated at the bottom of the geometry because the analytical solution of those areas are close to zero. Additionally, σ_{yy} around the edges tends to have a larger error, which is possibly a consequence of integration errors caused by partially filled elements and reduced MP domains.



(a) R_v of 45° model.

(b) $R_{\sigma_{yy}}$ of 45° model.

(c) R_v of 60° model.

(d) $R_{\sigma_{yy}}$ of 60° model.

Figure 6.10: Imposition of IDBCs: error distribution plots.

6.2 Cantilever beam with applied traction

The second example uses a cantilever beam to demonstrate the post-processing aspect of the boundary representations. As shown in Figure 6.11, the modelled cantilever beam has a length of 10m and a depth of 2m. The left hand side boundary is fixed at the middle and with rollers above and below. A constant pressure of 1500Pa

was applied along the top boundary, which has been maintained perpendicular to the boundary through out the analysis.

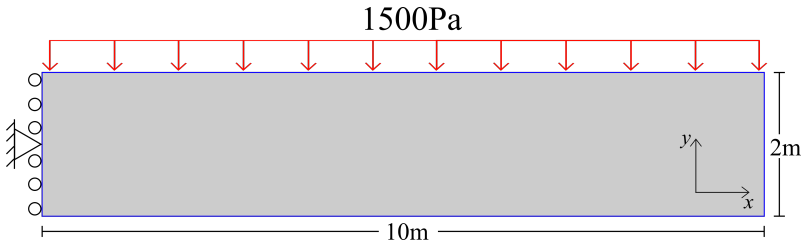
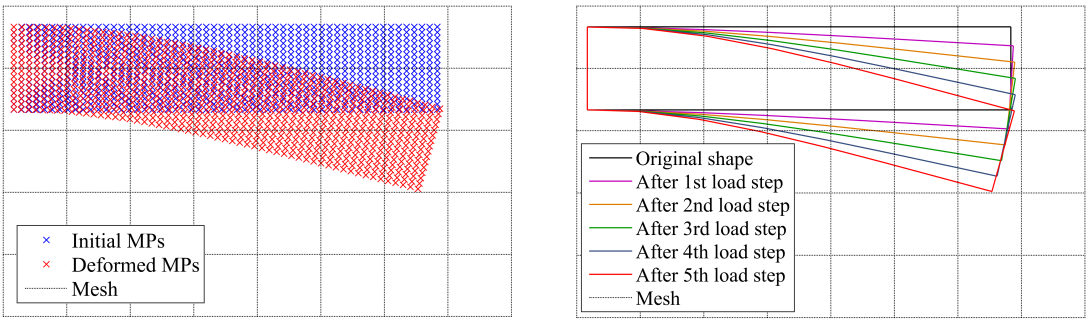


Figure 6.11: A cantilever bean under constant pressure.

A background mesh with 1.5m by 1.5m elements was used, and the problem domain was discretised by using 896 uniformly distributed MPs. The outer layer of the MPs were identified as the problem boundaries which were approximated by using B-splines. BCs on the left boundary ware applied by the B-spline based IBM and the pressure applied through 5 load steps. The initial discretisation and the final deformed cantilever beam are shown in Figure 6.12a; however, having boundaries represented by B-splines, the boundaries can be tracked after each load step without plotting out all the MPs (see Figure 6.12b) and the curvature of the deformed shape has been successfully captured by the B-spline approximation.



(a) Final deformation.

(b) Visualisation after ecah load step.

Figure 6.12: Illustrations of the deformed cantilever beam.

As shown in Figure 6.12a, the overall response of the cantilever beam is non-linear; however, the MPM presented in this thesis is incrementally linear. Therefore, there is no analytical solution for this problem as the adopted formulation of this lies between large deformation mechanics and small strain theory. Therefore, a convergence test on the displacement at the centre of the beam tip was carried out by performing h-refinement on the mesh and increasing the number of MPs. As shown in Figure 6.13, for all three mesh configurations, the mid-tip displacement converges when more MPs are introduced. Reduction of the change in displacement with mesh refinement indicates that the mid-tip displacement also converges when the mesh size is decreased.

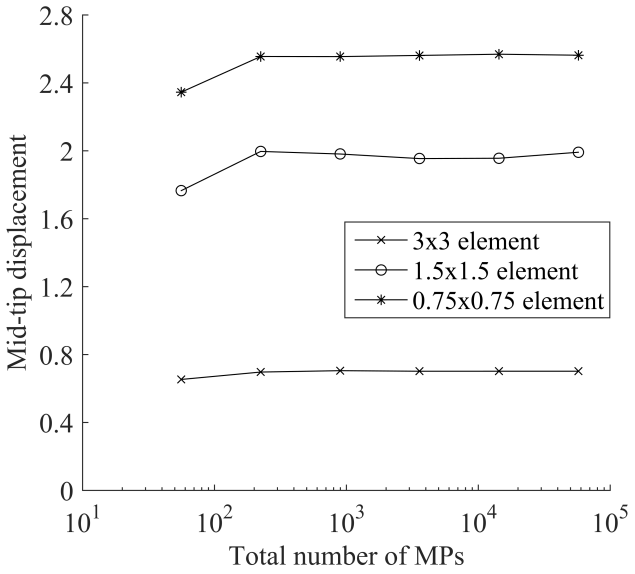
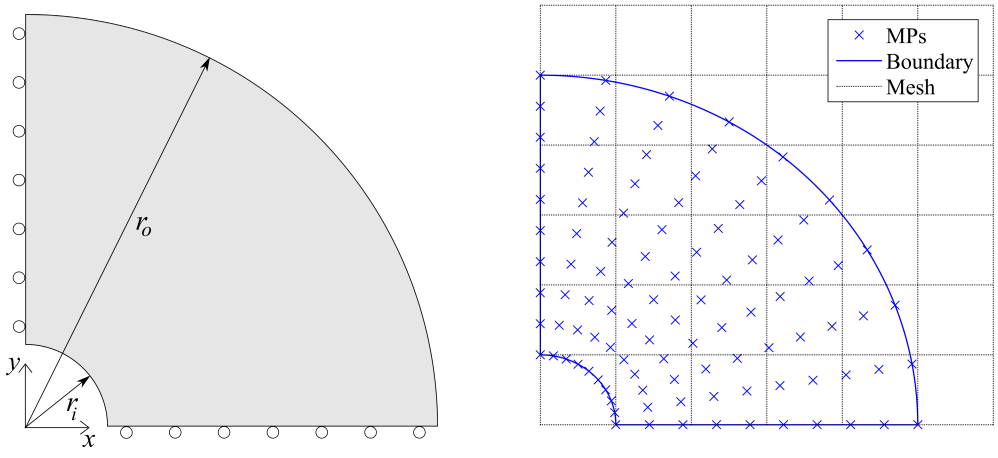


Figure 6.13: Convergence of mid-tip displacement.

6.3 Internally pressurised thick-walled cylinder

This example models an internally pressurised thick-walled cylinder. Only a quarter of the cross-section was analysed due to symmetry; the problem domain and boundary conditions are shown in Figure 6.14a where $r_i = 1$m and $r_o = 5$m. In order to double the inner radius, a required internal pressure of 0.756×10^6Pa was computed according to the analytical solution [90], and this pressure was then applied in the simulation.



(a) Illustration of the problem domain.

(b) Discretisation of the porblem domain.

Figure 6.14: A quarter of the cross-section of a thick-walled cylinder.

To discretise the cylinder, the same number of MPs were placed uniformly along the circumference and across the radii on a 6m by 6m background mesh with 1m by 1m elements. Figure 6.14b illustrates the discretisation with 10^2 MPs. Boundaries were

approximated by interpolating the outer layer of MPs using cubic B-splines.

Starting with 8^2 MPs, the average relative error of both the inner and outer radii converge when the number of MPs was increased (see Figure 6.15a). However, the inner radius shows a much higher error than seen at the outer radius, which is because the background mesh is too coarse to accurately capture the expansion. Evidences of this explanation are provided by Figure 6.15c and 6.15d where 128^2 MPs are used; it is clear that a more accurate simulation of the deformation at the inner radius was achieved with a finer mesh. To quantify the errors, convergence analyses of the inner radius error versus mesh size were performed. 512^2 MPs were used in this analysis as further increasing the number of MPs only reduced the error by less than 1% with same mesh. The results are shown in Figure 6.15b. Simulations with roller BCs enforced by the B-spline based IBM has a convergence rate of approximately 3 which is very similar to that using conventional means.

6.4 Infinite plate with a circular hole under far field stress

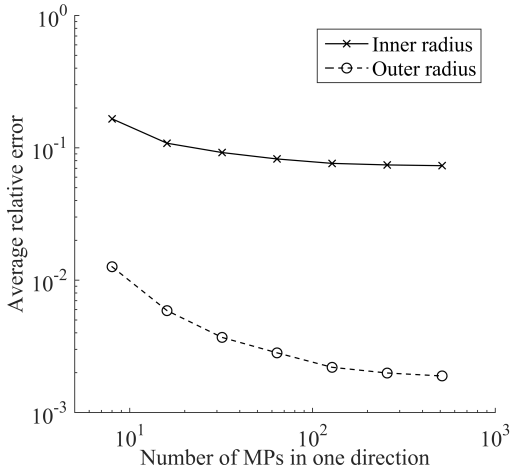
The last example analysed an infinite plate with a circular hole under a far field stress, which particularly challenges the newly developed traction application method on applying varying tractions along a boundary. A far field normal stress, $S = 10^6$Pa was applied horizontally at both sides of the plate. Because of symmetry, a quarter of the plate with a width of 4m was analysed. The circular hole at the centre of the plate had a diameter $2a$ = 1m, and the material used to model this problem had a Young's modulus of 10MPa and Poisson's ratio of 0.2. The problem geometry along with the boundary conditions are shown in Figure 6.16a. An illustration of the problem discretised by using a 3m by 3m background mesh containing 1m by 1m elements and 8^2 MPs per element is shown in Figure 6.16b; any MP with a distance from the origin less than the circle's radius, i.e. 0.5m, has been deleted. The B-splines boundaries are formulated by points that are placed along the domain boundaries and these points have zero weights.

Roller boundary conditions were applied through the B-spline based IBM. To accurately apply the tractions on the top and right boundaries, the analytical solution to this problem is required. The stress solution [90] described in the Cartesian coordinate system is given by:

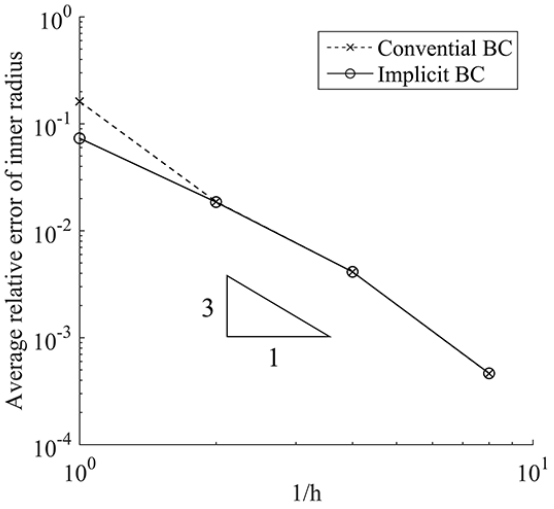
$$\sigma_{xx} = S - S\left(\frac{a^2}{r^2}\right)\left(\frac{3}{2}\cos(2\theta) + \cos(4\theta)\right) + S\left(\frac{3a^4}{2r^4}\right)\cos(4\theta), \tag{6.4}$$

$$\sigma_{yy} = -S\left(\frac{a^2}{r^2}\right)\left(\frac{1}{2}\cos(2\theta) + \cos(4\theta)\right) - S\left(\frac{3a^4}{2r^4}\right)\cos(4\theta) \quad \text{and} \tag{6.5}$$

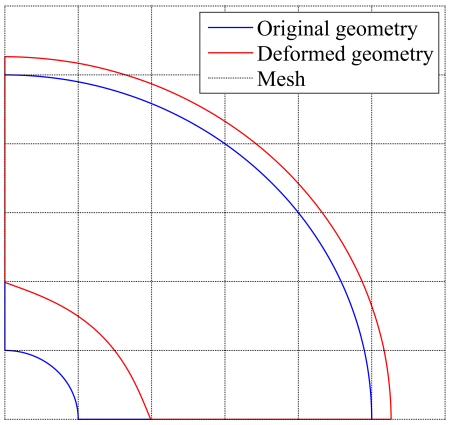
$$\sigma_{xy} = -S\left(\frac{a^2}{r^2}\right)\left(\frac{1}{2}\sin(2\theta) + \sin(4\theta)\right) + S\left(\frac{3a^4}{2r^4}\right)\sin(4\theta), \tag{6.6}$$



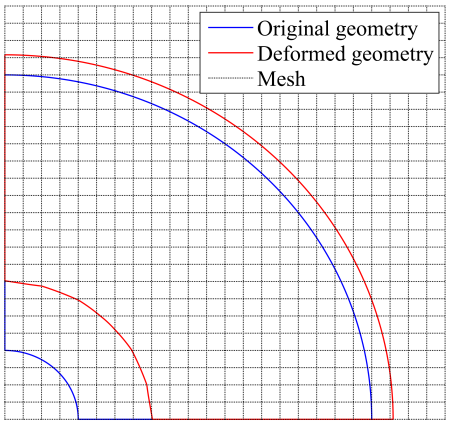
(a) Convergence plot with increase in number of MPs.



(b) h-refinement convergance plot.



(c) 1m by 1m elements.

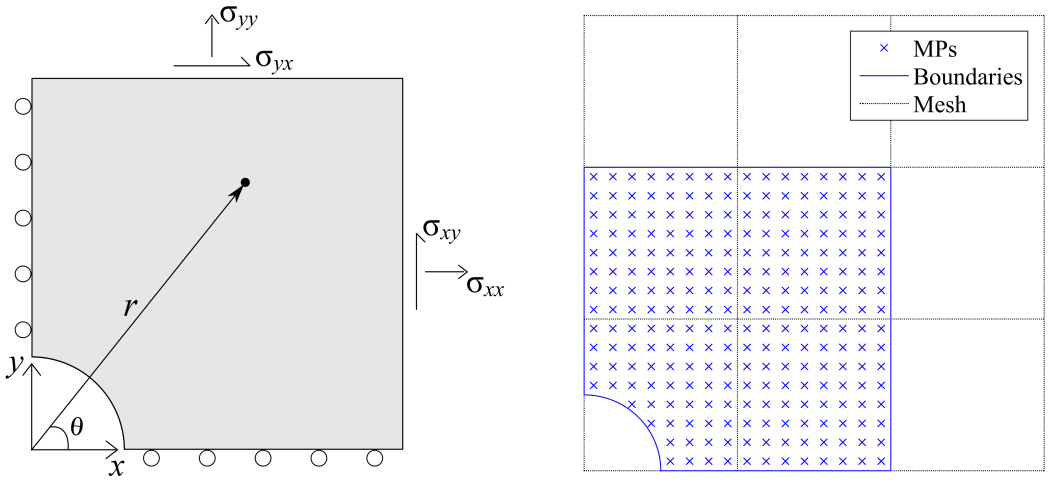


(d) 0.25m by 0.25m elements.

Figure 6.15: Error convergence plots and deformation illustrations of the cylinder.

where a is the radius of the circular hole, r is the distance from the centre of the circle to a point on the plate, θ is the angle between r and the positive x-direction, and S is the far field stress. The value of r for each Gauss point on the boundary was calculated by using the global coordinates of the point and then the traction at this point was computed by (6.5-6.6).

Because of the complexity of this problem, using a fine mesh is necessary. Therefore, elements with width $h = 0.0625$m and 8^2 MPs per element were used. Figure 6.17a, 6.17c and 6.17e show the relative error distributions of σ_{xx}, σ_{yy} and σ_{xy} respectively. To note that the upper limits shown in these plots has been adjusted to a smaller value to show the error distributions more clearly. Although the upper limits (in bright yellow) after adjustment are still quite large (0.1 for $R_{\sigma_{xx}}$ and 1 for both $R_{\sigma_{yy}}$ and $R_{\sigma_{xy}}$), it is found that the yellow areas are concentrated at certain places in the



(a) A quarter of the problem.

(b) Discretisation of the porblem domain.

Figure 6.16: A quarter of the cross-section of a thick-walled cylinder.

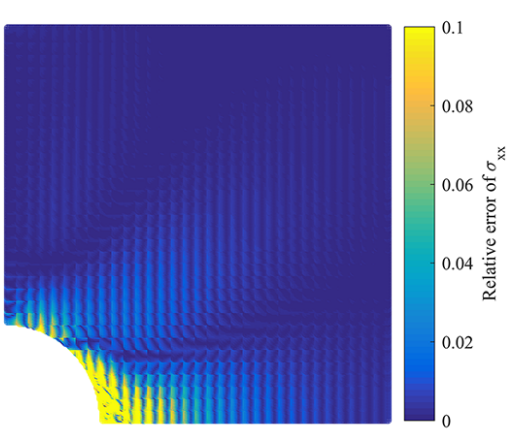
distributions plots and errors at other areas seems to have reasonable values. The high relative errors are due to near zero analytical solutions, which is evidenced by the absolute analytical solution plots in Figure 6.17b, 6.17d and 6.17f. Comparing the relative error plots with the analytical solution plots side by side, it is clear that the bright yellow areas in the former plots are located at the same places as the pale purple bands in the latter plots.

Figure 6.18 shows the absolute error distributions of displacements in both horizontal and vertical directions. The distribution patterns are as expected with maximum errors exist around the circular hole.

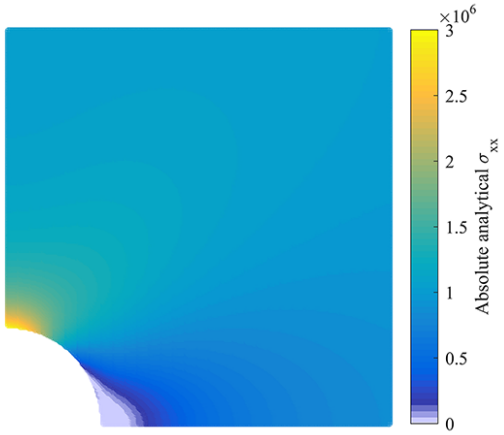
Finally convergence analysis was performed. The background mesh was refined while the number of MPs per elements was maintained the same (8^2 MPs per element). The results are shown in Figure 6.19, where all factors under examination show convergence when the mesh was refined. Both displacement errors have a convergence rate of approximately 2 which is consistent with a linear FE basis as a convergence rate of (element order+1) is expected; whereas all stress errors converge at a rate of approximately 0.8 which is slightly lower than the expected value (which is the order of the element).

6.5 Chapter review

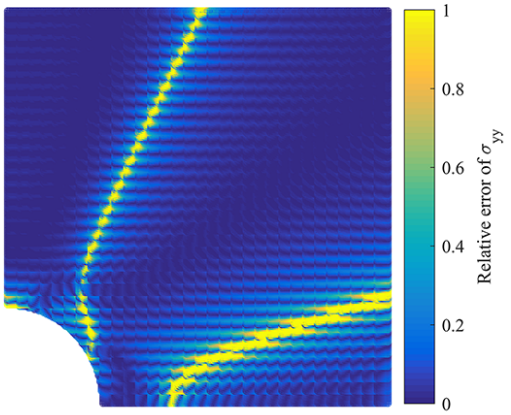
This chapter has validated the boundary representation and boundary condition imposition techniques, and demonstrated the ability of these methods using four problems: a simple 1D compression problem, a cantilever beam with applied traction, a internally pressurised thick-walled cylinder and an infinite plate with a circular hole. The first example thoroughly investigated the performance of applying



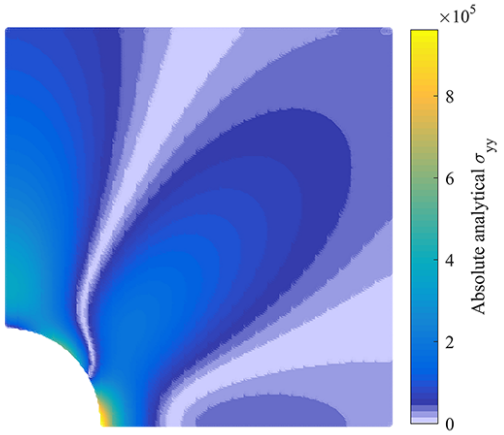
(a) $R_{\sigma_{xx}}$.



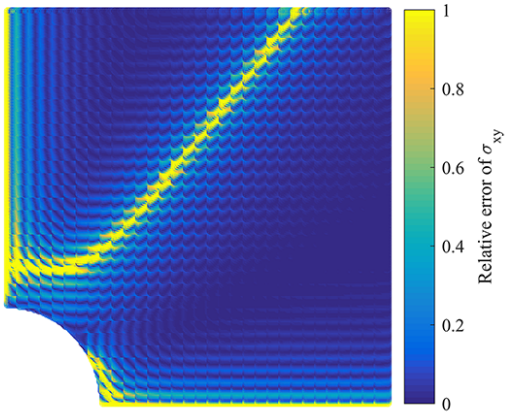
(b) $|\sigma_{xx}|$.



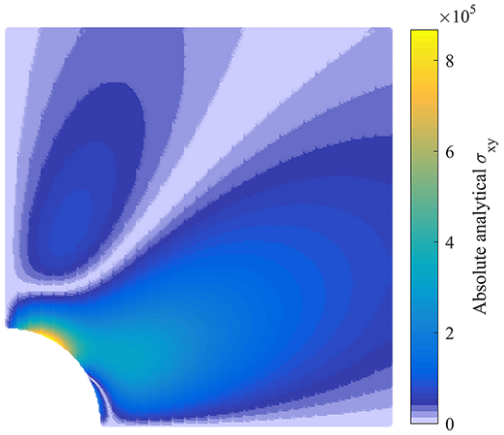
(c) $R_{\sigma_{yy}}$.



(d) $|\sigma_{yy}|$.

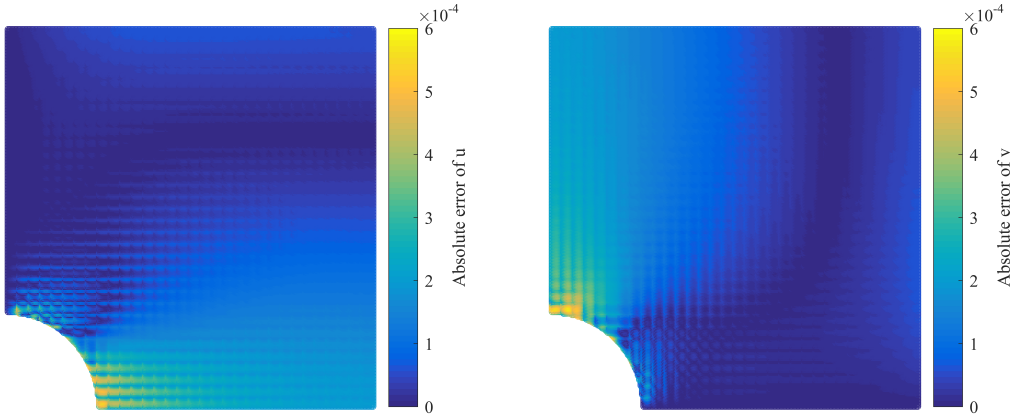


(e) $R_{\sigma_{xy}}$.



(f) $|\sigma_{xy}|$.

Figure 6.17: Distributions of relative stress errors and absolute analytical solutions.



(a) A_u. (b) A_v.

Figure 6.18: Absolute error distributions of displacements.

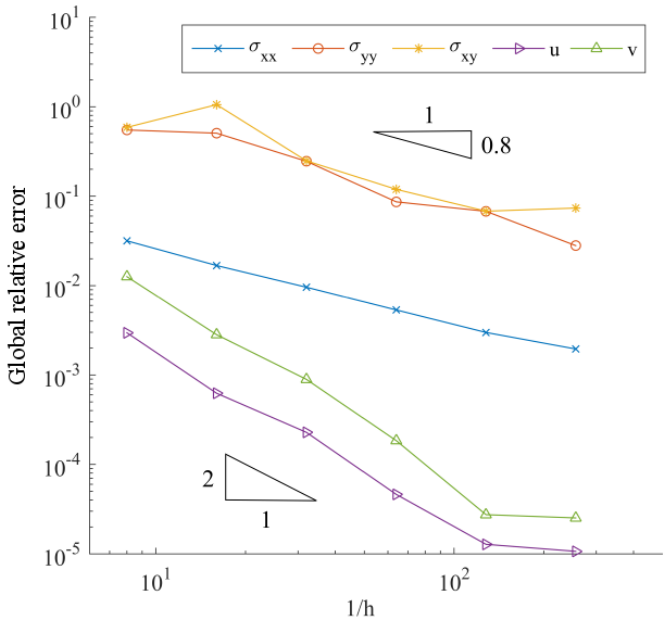


Figure 6.19: Convergence of relative errors.

B-spline based Drichlet and Neumann BCs. Tractions have been applied to the problem successfully and the B-spline based IBM has shown to be capable of applying IDBCs as well as both coincident and inclined roller BCs. Additionally, the value of the bandwidth δ has been studied in detail. In general, inclined or not, a bandwidth of 10^{-6} with the same unit as the problem gives the best accuracy. The benefits of using the B-spline described boundaries have been demonstrated in examples 2 and 3. Not only can the deformed problem domain be visualised without plotting all the MPs, but the B-spline has the ability to accurately capture the curvatures exist in the boundaries. Finally, the infinite plate problem shows the ability of the method to apply non-constant tractions. Overall, these numerical examples have

demonstrated the capability of the B-spline based boundary methods, which allows the MPM to be able to model problems that were previously not possible.

Chapter 7

Conclusions

The MPM, as an alternative to the FEM, is a growingly popular choice in solid mechanics especially for problems that involve large deformations. However, because the MPM discretises a continuum body with a set of MPs and that the background mesh does not necessarily conforms the problem boundaries, visualising the deformed domain and applying boundary conditions accurately are an issue. Surprisingly, few researchers have chosen to address this problem to date. To address this, a complete method of boundary representation and BC imposition has been developed and verified in this thesis.

A clearly defined boundary is formed by interpolating the outer layer of material points of the problem domain with B-splines which have been chosen due to their high continuity and the ability of being manipulated independently from the background mesh. In particular, a local cubic interpolation technique is employed in the current work because it is capable of handling sharp corners and sudden turns that often appear in problem geometries. They are also straightforward to implement as discussed in Chapter 4. Based on this boundary description, a methodology for enforcing both Neumann and Dirichlet boundary conditions is given in Chapter 5. Tractions are imposed by direct integration over the B-spline boundaries, and displacements prescribed through incorporating B-spines with the IBM.

In Chapter 6, the feasibility of this B-spline based boundary method has been proven by a number of numerical examples. The first problem considered one dimensional compression of an elastic square domain. It has been demonstrated that both natural and essential BCs are imposed successfully not only on the boundaries that are parallel to but also inclined with the coordinate axes. The key factor affecting the accuracy of the IBM has also been studied. It has been found that a bandwidth with a value of 10^{-6} in the same units as the problem is the optimum choice. Additionally, the importance of the relative size of the background mesh to the problem domain and the number of MPs as well as the positioning of the MPs within each element have also been discussed. The problems of the cantilever beam and the internally

pressured cylinder illustrated the advantages of using B-spline interpolated boundaries; visualisation of the deformed domain has been achieved without plotting out any material points and the curvatures of the boundaries have been accurately captured. Furthermore, the ability of this method on imposing non-constant tractions has been shown in the problem of an infinite plate with a circular hole.

In conclusion, the B-spline based boundary representation and BC imposition technique descried in this thesis has further improved the standard MPM which is now able to analysis a wider range of problems, and it is not hard to extend this approach into 3D. Recall the problem of a rotated bar [25] discussed in the introduction, which imposes zero essential boundary condition on an inclined boundary by adding a "ghost" bar on the other side of the boundary. This BC can now be easily enforced by the B-spline based IBM. However, this B-spline based boundary method still requires further testing and improvements. First of all, the performance of the B-spline based IBM on enforcing IDBCs to curved boundaries would be investigated. The problem described in Section 6.3 can be used again with the internal pressure replaced by a displacement of 1m. Stress distributions produced by applying the IDBC should be the same as the one obtained by applying traction. Secondly, the B-spline based boundary method should be tested with problems involving large deformation and fractures because these are the main focus of the applications of the MPM. For example, problems, such as collapse of a sand pile, can be used to test the method's performance on tracking largely deformed boundaries. Finally, in order to achieve a full automation of this method, optimising the current search algorithm for finding the intersections of the B-spline curve with the background mesh is suggested as instability has been observed when the problem boundary coincides with the mesh.

Bibliography

[1] F. M. Hamad, "Formulation of a dynamic material point method and applications to soil-water-geotextile systems," Ph.D. dissertation, University of Stuttgart, 2014.

[2] I. K. J. al Kafaji, "Formulation of a dynamic material point method (MPM) for geomechanical problems," Ph.D. dissertation, University of Stuttgart, Holzgartenstr. 16, 70174 Stuttgart, 2013.

[3] C. Mast, P. Mackenzie-Helnwein, P. Arduino, and G. Miller, "Landslide and debris flow-induced static and dynamic loads on protective structures," in *Multiscale and Multiphysics Processes in Geomechanics*, ser. Springer Series in Geomechanics and Geoengineering, R. Borja, Ed. Springer Berlin Heidelberg, 2011, pp. 169–172.

[4] A. Sadeghirad, R. Brannon, and J. Guilkey, "Second-order convected particle domain interpolation (CPDI2) with enrichment for weak discontinuities at material interfaces," *International Journal for Numerical Methods in Engineering*, vol. 95, no. 11, pp. 928–952, 2013.

[5] M. Cortis, W. M. Coombs, C. Augarde, M. Brown, A. Brennan, and S. Robinson, "Imposition of essential boundary conditions in the material point method," *International Journal for Numerical Methods in Engineering*, 2017.

[6] H. Stoker, "Developments of the arbitrary lagrangian-eulerian method in nonlinear solid mechanics applications to forming processes," Ph.D. dissertation, Universiteit Twente, 2 1999.

[7] L. B. Lucy, "A numerical approach to the testing of the fission hypothesis," *Astronomical Journal*, vol. 82, pp. 1013–1024, 1977.

[8] T. Belytschko, Y. Y. Lu, and L. Gu, "Element-free Galerkin methods," *International Journal for Numerical Methods in Engineering*, vol. 37, no. 2, pp. 229–256, 1994.

[9] W. K. Liu, S. Jun, and Y. F. Zhang, "Reproducing kernel particle methods," *International Journal for Numerical Methods in Fluids*, vol. 20, no. 8-9, pp. 1081–1106, 1995.

[10] V. P. Nguyen, T. Rabczuk, S. Bordas, and M. Duflot, "Meshless methods: A review and computer implementation aspects," *Mathematics and Computers in Simulation*, vol. 79, no. 3, pp. 763 – 813, 2008.

[11] D. Sulsky, Z. Chen, and H. Schreyer, "A particle method for history-dependent materials," *Computer Methods in Applied Mechanics and Engineering*, vol. 118, no. 1, pp. 179 – 196, 1994.

[12] D. Sulsky, S.-J. Zhou, and H. L. Schreyer, "Application of a particle-in-cell method to solid mechanics," *Computer Physics Communications*, vol. 87, no. 1, pp. 236 – 252, 1995.

[13] J. Brackbill and H. Ruppel, "Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions," *Journal of Computational Physics*, vol. 65, no. 2, pp. 314 – 343, 1986.

[14] F. Harlow, "The particleincell computing method for fluid dynamics," *Methods for Computational Physics*, vol. 3, pp. 319–343, 1964.

[15] D. Sulsky and H. L. Schreyer, "Axisymmetric form of the material point method with applications to upsetting and taylor impact problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 139, no. 1, pp. 409 – 429, 1996.

[16] M. Steffen, P. Wallstedt, J. Guilkey, R. Kirby, and M. Berzins, "Examination and analysis of implementation choices within the material point method (MPM)," *Computer Modeling in Engineering and Sciences*, vol. 31, no. 2, pp. 107–127, 2008.

[17] S. Bardenhagen, "Energy conservation error in the material point method for solid mechanics," *Journal of Computational Physics*, vol. 180, no. 1, pp. 383 – 403, 2002.

[18] O. Buzzi, D. M. Pedroso, and A. Giacomini, "Caveats on the implementation of the generalized material point method," *Computer Modeling in Engineering and Sciences*, 2008.

[19] S. Cummins and J. Brackbill, "An implicit particle-in-cell method for granular materials," *Journal of Computational Physics*, vol. 180, no. 2, pp. 506 – 548, 2002.

[20] J. E. Guilkey and J. A. Weiss, "Implicit time integration for the material point method: Quantitative and algorithmic comparisons with the finite element method," *International Journal for Numerical Methods in Engineering*, vol. 57, no. 9, pp. 1323–1338, 2003.

[21] B. Wang, P. J. Vardon, M. A. Hicks, and Z. Chen, "Development of an implicit material point method for geotechnical applications," *Computers and Geotechnics*, vol. 71, pp. 159 – 167, 2016.

[22] S. Bardenhagen and E. Kober, "The generalized interpolation material point method," *Computer Modeling in Engineering and Sciences*, vol. 5, no. 6, pp. 477–495, 2004.

[23] J. Ma, "Multiscale simulation using the generalized interpolation material point method, discrete dislocations and molecular dynamics," Ph.D. dissertation, Oklahoma State University, 2006.

[24] P. C. Wallstedt and J. E. Guilkey, "A weighted least squares particle-in-cell method for solid material mechanics," *International Journal for Numerical Methods in Engineering*, vol. 85, no. 13, pp. 1687–1704, 2011.

[25] 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," *International Journal for Numerical Methods in Engineering*, vol. 86, no. 12, pp. 1435–1456, 2011.

[26] A. R. York, D. Sulsky, and H. Schreyer, "The material point method for simulation of thin membranes," *International Journal for Numerical Methods in Engineering*, vol. 44, pp. 1429–1456, 1999.

[27] A. R. York, D. Sulsky, and H. L. Schreyer, "Fluid-membrane interaction based on the material point method," *International Journal for Numerical Methods in Engineering*, vol. 48, pp. 901–924, 2000.

[28] Y.-P. Lian, Y. Liu, and X. Zhang, "Coupling of membrane element with material point method for fluid–membrane interaction problems," *International Journal of Mechanics and Materials in Design*, vol. 10, no. 2, pp. 199–211, 2014.

[29] F. Hamad, C. Moormann, and P. Vermeer, "Development of a coupled fem-mpm approach to model a 3d membrane with an application of releasing geocontainer from barge," in *Installation Effects in Geotechnical Engineering*. CRC Press, 2013, pp. 176–183.

[30] F. Hamad, D. Stolle, and P. A. Vermeer, "Modelling of membranes in the material point method with applications," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 39, no. 8, pp. 833–853, 2015.

[31] D. Z. Zhang, Q. Zou, W. B. VanderHeyden, and X. Ma, "Material point method applied to multiphase flows," *Journal of Computational Physics*, vol. 227, no. 6, pp. 3159 – 3173, 2008.

[32] P. Mackenzie-Helnwein, P. Arduino, W. Shin, J. A. Moore, and G. R. Miller, "Modeling strategies for multiphase drag interactions using the material point method," *International Journal for Numerical Methods in Engineering*, vol. 83, no. 3, pp. 295–322, 2010.

[33] Y. Lian, X. Zhang, X. Zhou, S. Ma, and Y. Zhao, "Numerical simulation of explosively driven metal by material point method," *International Journal of Impact Engineering*, vol. 38, no. 4, pp. 238–246, 2011.

[34] Y. Wang, H. Beom, M. Sun, and S. Lin, "Numerical simulation of explosive welding using the material point method," *International Journal of Impact Engineering*, vol. 38, no. 1, pp. 51 – 60, 2011.

[35] S. Zhou, X. Zhang, and H. Ma, "Numerical simulation of human head impact using the material point method," *International Journal of Computational Methods*, vol. 10, no. 04, 2013.

[36] P. Liu, Y. Liu, X. Zhang, and Y. Guan, "Investigation on high-velocity impact of micron particles using material point method," *International Journal of Impact Engineering*, vol. 75, pp. 241 – 254, 2015.

[37] J. H. Lee and D. Huang, "Modeling and testing of snow penetration," *Journal of Terramechanics*, vol. 59, pp. 35 – 47, 2015.

[38] Y. Lian, X. Zhang, F. Zhang, and X. Cui, "Tied interface grid material point method for problems with localized extreme deformation," *International Journal of Impact Engineering*, vol. 70, pp. 50 – 61, 2014.

[39] Y. Lian, P. Yang, X. Zhang, F. Zhang, Y. Liu, and P. Huang, "A mesh-grading material point method and its parallelization for problems with localized extreme deformation," *Computer Methods in Applied Mechanics and Engineering*, vol. 289, pp. 291 – 315, 2015.

[40] S. Bardenhagen, J. Brackbill, and D. Sulsky, "The material-point method for granular materials," *Computer Methods in Applied Mechanics and Engineering*, vol. 187, no. 3, pp. 529 – 541, 2000.

[41] W. Solowski and S. W. Sloan, "Material point method modelling of granular flow in inclined channels," *Applied Mechanics and Materials*, vol. 553, pp. 501–506, 2014.

[42] L. Beuth, T. Benz, and P. A. Vermeer, "Large deformation analysis using a quasi-static material point method," *Journal of Theoretical and Applied Mechanics*, 2008.

[43] S. Andersen, "Material-point analysis of large-strain problems: modelling of landslides," Ph.D. dissertation, Aalborg University, Denmark, 2009.

[44] S. Andersen and L. Andersen, "Modelling of landslides with the material-point method," *Computational Geosciences*, vol. 14, no. 1, pp. 137–147, 2010.

[45] A. Yerro, N. M. Pinyol, and E. E. Alonso, "Internal progressive failure in deep-seated landslides," *Rock Mechanics and Rock Engineering*, vol. 49, no. 6, pp. 2317–2332, 2016.

[46] F. A. Gilabert, V. Cantavella, E. Sánchez, and G. Mallol, "Modelling fracture process in ceramic materials using the material point method," *EPL (Europhysics Letters)*, vol. 96, no. 2, p. 24002, 2011.

[47] F. Li, J. Pan, and A. Cocks, "A new numerical scheme for computer simulation of multiple cracking in ceramic films during constrained sintering," *Modelling and Simulation in Materials Science and Engineering*, vol. 20, no. 3, p. 035008, 2012.

[48] Z. Chen and R. Brannon, "An evaluation of the material point method," Sandia National Laboratories, Albuquerque, New Mexico 87185 and Livermore, California 94550, Tech. Rep., 2002.

[49] Z. Chen, W. Hu, L. Shen, X. Xin, and R. Brannon, "An evaluation of the mpm for simulating dynamic failure with damage diffusion," *Engineering Fracture Mechanics*, vol. 69, no. 17, pp. 1873 – 1890, 2002.

[50] C. Mast, "Representing arbitrary bounding surfaces in the material point method," 2010, 6^{th} MPM Workshop in Albuquerque, New Mexico.

[51] R. K. Burla and A. V. Kumar, "Implicit boundary method for analysis using uniform b-spline basis and structured grid," *International Journal for Numerical Methods in Engineering*, vol. 76, no. 13, pp. 1993–2028, 2008.

[52] A. V. Kumar, S. Padmanabhan, and R. Burla, "Implicit boundary method for finite element analysis using non-conforming mesh or grid," *International Journal for Numerical Methods in Engineering*, vol. 74, no. 9, pp. 1421–1447, 2008.

[53] A. V. Kumar, R. Buria, S. Padmanabhan, and L. Gu, "Finite element analysis using nonconforming mesh," *Journal of Computing and Information Science in Engineering*, vol. 8, 2008.

[54] L. Piegl and W. Tiller, *The NURBS Book*. Springer-Verlag Berlin Heidelberg, 1997.

[55] L. Piegl, "On NURBS: A survey," *IEEE Comput. Graph. Appl.*, vol. 11, no. 1, pp. 55–71, Jan. 1991.

[56] J. A. Cottrell, T. J. Hughes, and Y. Bazilevs, *Isogeometric analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, 2009.

[57] T. W. Sederberg, J. Zheng, A. Bakenov, and A. Nasri, "T-splines and T-NURCCs," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 477–484, Jul. 2003.

[58] D. Sulsky and A. Kaul, "Implicit dynamics in the material-point method," *Computer Methods in Applied Mechanics and Engineering*, vol. 193, no. 1214, pp. 1137 – 1170, 2004, meshfree Methods: Recent Advances and New Applications.

[59] S. Fernández-Méndez and A. Huerta, "Imposing essential boundary conditions in mesh-free methods," *Computer Methods in Applied Mechanics and Engineering*, vol. 193, pp. 1257 – 1275, 2004, meshfree Methods: Recent Advances and New Applications.

[60] S. Ma and X. Zhang, "Material point method for impact and explosion problems," in *Computational Mechanics*. Springer Berlin Heidelberg, 2009, pp. 156–166.

[61] K. Höllig, U. Reif, and J. Wipper, "Weighted extended b-spline approximation of dirichlet problems," *SIAM Journal on Numerical Analysis*, vol. 39, no. 2, pp. 442–462, 2001.

[62] K. Höllig and U. Reif, "Nonuniform web-splines," *Computer Aided Geometric Design*, vol. 20, no. 5, pp. 277 – 294, 2003.

[63] K. Höllig, C. Apprich, and A. Streit, "Introduction to the web-method and its applications," *Advances in Computational Mathematics*, vol. 23, no. 1, pp. 215–237, 2005.

[64] W. Zhang, L. Zhao, and S. Cai, "Shape optimization of dirichlet boundaries based on weighted b-spline finite cell method and level-set function," *Computer Methods in Applied Mechanics and Engineering*, vol. 294, pp. 359 – 383, 2015.

[65] W. Zhang and L. Zhao, "Exact imposition of inhomogeneous dirichlet boundary conditions based on weighted finite cell method and level-set function," *Computer Methods in Applied Mechanics and Engineering*, vol. 307, pp. 316 – 338, 2016.

[66] V. Rvachev and T. Sheiko, "R-functions in boundary value problems in mechanics," *Applied Mechanics Reviews*, vol. 48, no. 4, pp. 151–188, 1995.

[67] L. V. Kantorovich and V. I. Krylov, *Approximate Methods of Higher Analysis*. New York, Interscience Publishers, 1958.

[68] V. Rvachev, T. Sheiko, V. Shapiro, and I. Tsukanov, "Transfinite interpolation over implicitly defined sets," *Computer Aided Geometric Design*, vol. 18, no. 3, pp. 195 – 220, 2001.

[69] N. Phuong, A. van Tol, A. Elkadi, and A. Rohe, "Numerical investigation of pile installation effects in sand using material point method," *Computers and Geotechnics*, vol. 73, pp. 58 – 71, 2016.

[70] X. Zhang, Q. Yang, and J. Bao, "The dual grid method for stability analysis of sliding blocks and slopes," *International Society for Rock Mechanics*, 2010.

[71] B. P. Lamichhane, "Inf-sup stable finite-element pairs based on dual meshes and bases for nearly incompressible elasticity," *IMA Journal of Numerical Analysis*, vol. 29, pp. 404–420, 2009.

[72] B. P. Lamichhane, "A mixed finite element method for nearly incompressible elasticity and stokes equations using primal and dual meshes with quadrilateral and hexahedral grids," *Journal of Computational and Applied Mathematics*, vol. 260, pp. 356 – 363, 2014.

[73] T. Belytschko, D. Organ, and Y. Krongauz, "A coupled finite element-element-free Galerkin method," *Computational Mechanics*, vol. 17, no. 3, pp. 186–195, 1995.

[74] A. Huerta and S. Fernández-Méndez, "Enrichment and coupling of the finite element and meshless methods," *International Journal for Numerical Methods in Engineering*, vol. 48, no. 11, pp. 1615–1636, 8 2000.

[75] G. J. Wagner and W. K. Liu, "Hierarchical enrichment for bridging scales and mesh-free boundary conditions," *International Journal for Numerical Methods in Engineering*, vol. 50, no. 3, pp. 507–524, 1 2001.

[76] D. Natekar, X. Zhang, and G. Subbarayan, "Constructive solid analysis: a hierarchical, geometry-based meshless analysis procedure for integrated design and analysis," *Computer-Aided Design*, vol. 36, no. 5, pp. 473 – 486, 2004.

[77] T. Hughes, J. Cottrell, and Y. Bazilevs, "Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, pp. 4135 – 4195, 2005.

[78] R. Sevilla, S. Fernández-Méndez, and A. Huerta, "Nurbs-enhanced finite element method (nefem)," *International Journal for Numerical Methods in Engineering*, vol. 76, no. 1, pp. 56–83, 2008.

[79] M. Steffen, R. M. Kirby, and M. Berzins, "Analysis and reduction of quadrature errors in the material point method (MPM)," *International Journal for Numerical Methods in Engineering*, vol. 76, no. 6, pp. 922–948, 2008.

[80] H.-J. Kim and S.-K. Youn, "Spline-based meshfree method," *International Journal for Numerical Methods in Engineering*, vol. 92, no. 9, pp. 802–834, 2012.

[81] H.-J. Kim, Y.-D. Seo, and S.-K. Youn, "Isogeometric analysis for trimmed cad surfaces," *Computer Methods in Applied Mechanics and Engineering*, vol. 198, pp. 2982 – 2995, 2009.

[82] H.-J. Kim, Y.-D. Seo, and S.-K. Youn, "Isogeometric analysis with trimming technique for problems of arbitrary complex topology," *Computer Methods in Applied Mechanics and Engineering*, vol. 199, pp. 2796 – 2812, 2010.

[83] C. de Boor, "On calculating with b-splines," *Journal of Approximation Theory*, vol. 6, no. 1, pp. 50 – 62, 1972.

[84] E. Lee, "Choosing nodes in parametric curve interpolation," *Computer-Aided Design*, vol. 21, no. 6, pp. 363 – 370, 1989.

[85] P. E. Bézier, *Numerical Control: Mathematics and Applications*. John Wiley, 1972.

[86] W. Böhm, G. Farin, and J. Kahmann, "A survey of curve and surface methods in cagd," *Computer Aided Geometric Design*, vol. 1, no. 1, pp. 1–60, 1984.

[87] C. de Boor, *A Practical Guide to Splines*. Springer-Verlag New York, 1978.

[88] G. Renner, "A method of shape description for mechanical engineering practice," *Computers in Industry*, vol. 3, no. 1, pp. 137 – 142, 1982.

[89] A. Becker, *The Boundary Element Method in Engineering: A complete course*. McGraw-Hill, 1992.

[90] S. Timoshenko and J. N. Goodier, *Theory of Elasticity*, 2nd ed. McGraw-Hill, 1951.