4

Recent developments in improving the numerical accuracy of MPM

Elizaveta Wobbes, Roel Tielen, Matthias Möller, Cornelis Vuik

Delft University of Technology, Delft, the Netherlands

Vahid Galavi

Deltares, Delft, the Netherlands

4.1 Introduction

The MP Method has shown to be successful in simulating problems that involve large deformations. However, the standard algorithm suffers from many numerical shortcomings.While some of these shortcomings are inherited from the Finite Element Method (FEM), other drawbacks are MPM specific.

The FEM-type inaccuracies include interpolation, time integration, and mass lumping errors [219]. On the other hand, MPM suffers from the grid crossing errors [20] (see Chapter 2). The method typically projects the MP data to the background grid, also called mesh, and vice versa using piecewise-linear basis functions. The discontinuous gradients of these basis functions lead to unphysical oscillations in the solution when MPs cross element boundaries. In addition, MPM reconstructs scattered material point (MP) data using a low-order function-reconstruction technique that causes severe inaccuracies when large deformations are involved [222]. The most recent developments outlined in this chapter largely enhance the mathematical accuracy of the method, while keeping the physical qualities and computational efficiency close to those of original MPM.

B-spline Material Point Method (BSMPM) provides a fundamental approach to smoothing the gradients of the basis functions. In essence, it replaces the piecewise-linear basis functions by higher-order B-spline basis functions that guarantee at least C_0-continuity of the gradients. BSMPM was originally introduced by Steffen et al. [218], but Tielen et al. [236] proposed a more general and straightforward implementation. The new approach uses the Isogeometric Analysis (IgA) formulation of B-splines based on the Cox-de-Boor formula [65].

Although BSMPM reduces the grid crossing, interpolation, and time stepping errors within MPM [219, 236], the accuracy of the solution can be further improved by replacing the direct mapping of the MP data to the background grid by more advanced techniques. Several function-reconstruction methods have been proposed for the projection of the scattered MP (MP) data [222, 236, 261].

In contrast with many standard techniques, which do not conserve physical quantities like mass and momentum, the Taylor Least Squares (TLS) function reconstruction [261] not only decreases the numerical errors but also ensures the conservation of the total mass and linear momentum.
The TLS reconstruction is based on the least squares [128] approximation constructed from a set of Taylor basis functions [142]. It locally approximates quantities of interest, such as stress and density, allowing discontinuities across element boundaries. When used in combination with a suitable quadrature rule, the TLS technique preserves the mass and momentum after transferring the MP information to the background grid.
Furthermore, the mapping of MP information involves the solution of a linear system that includes a mass matrix at every time step. While a consistent mass matrix can significantly increase the accuracy of BSMPM combined with a reconstruction technique [236], it can also lead to stability issues. Therefore, further research of this topic is needed.

Meanwhile, p-multigrid can be applied to solve linear systems resulting from IgA discretisations to ensure an efficient computation with a consistent mass matrix. Compared to the standard techniques, such as the Conjugate Gradient method, p-multigrid requires a considerably lower number of iterations. In addition, p-multigrid can be used to solve the momentum balance equation within BSMPM. Based on the research of Love and Sulsky [137], it is expected that a consistent mass matrix will conserve the energy and angular momentum.

4.2 Most relevant MPM concepts

This section explains the mass lumping procedure and presents the computational steps of the widely used Modified-Update-Stress-Last (MUSL) scheme [223]. For simplicity, only one-dimensional deformations of a 1-phase continuum are considered.

4.2.1 Mass lumping

MPM solves the momentum balance equation in its weak form (see Chapter 1). After the spatial discretisation, the system solved on the background grid

follows Eq. 4.1.

$$\mathbf{M}^{\mathrm{C}}\vec{a} = \vec{f}_{\mathrm{ext}} - \vec{f}_{\mathrm{int}} \tag{4.1}$$

where $\mathbf{M}^{\mathrm{C}}$ is the consistent mass matrix, $\vec{a}$ the acceleration vector, $\vec{f}_{\mathrm{ext}}$ is the external force vector, and $\vec{f}_{\mathrm{int}}$ is the internal force vector.
The consistent mass matrix is given by Eq. 4.2.

$$\mathbf{M}^{\mathrm{C}} = \int_{\Omega} \rho\vec{\phi}\vec{\phi}^{\,\mathrm{T}}\,\mathrm{d}\Omega \tag{4.2}$$

where ρ is the density, $\vec{\phi}$ is the basis function vector, and Ω is the considered domain.
If the domain discretisation generates N_{n} degrees of freedom (DOFs), the basis function vector is denoted by $\vec{\phi}(x) = [\phi_1(x)\ \ \phi_2(x)\ \ldots\ \phi_{\mathrm{N_n}}(x)]^{\mathrm{T}}$, where ϕ_{i} represents a basis function associated with the ith DOF.

Similar to FEM, this non-diagonal matrix can be transformed into a diagonal matrix by the *lumping procedure* when piecewise-linear or B-spline basis functions are used. Denoting element (i, j) of the consistent and lumped mass matrices by $\mathbf{M}^{\mathrm{C}}_{(\mathrm{i,j})}$ and $\mathbf{M}_{(\mathrm{i,j})}$, respectively. The lumping procedure can be described by Eq. 4.3.

$$\mathbf{M}_{(\mathrm{i,j})} = \delta_{\mathrm{i,j}} \sum_{j} \mathbf{M}^{\mathrm{C}}_{(\mathrm{i,j})} \tag{4.3}$$

where $\delta_{\mathrm{i,j}}$ is the Dirac delta function.

Since piecewise-linear and B-spline basis functions satisfy the partition of unity property (i.e. $\sum_{\mathrm{i}=1}^{\mathrm{N_n}} \phi_{\mathrm{i}}(x) = 1\ \forall\ x \in \Omega$), mass lumping can also be achieved variationally (Eq. 4.4).

$$\mathbf{M} = \begin{bmatrix} \ddots & & \\ & \vec{m} & \\ & & \ddots \end{bmatrix} \quad \text{with} \quad \vec{m} = \int_{\Omega} \rho\vec{\phi}\,\mathrm{d}\Omega \tag{4.4}$$

In the version of MPM considered in this chapter, Eq. 4.1 is typically used with the lumped mass matrix $\mathbf{M}$. Lumping of the mass matrix has a number of advantages. For example, it reduces the computational costs and improves convergence characteristics of the method [137]. However, generally a lumped mass matrix limits the spatial convergence to $\mathcal{O}(h^2)$ [219] and hinders the conservation of energy and angular momentum [137].

4.2.2 Modified-Update-Stress-Last algorithm

Throughout this section, the superscript t denotes the time level, and Δt is the time-step length. The N_{mp} MPs are initialised at $t = 0$ s. Each MP carries a certain volume V_{mp}, density ρ_{mp}, position x_{mp}, displacement u_{mp}, velocity v_{mp}, and stress σ_{mp}. These values are time dependent, but the MP mass m_{mp} remains constant throughout the simulation. Assuming that all MP properties are known at time t, the computation for time $t + \Delta t$ proceeds as follows.

1. The data from the MPs is mapped to the DOFs of the background grid. For instance, the diagonal of the lumped mass matrix (Eq. 4.5) and the internal forces $\vec{f}_{\text{int}}$ (Eq. 4.6) are computed.

$$\vec{m}^{\,\text{t}} = \sum_{mp=1}^{N_{\text{mp}}} m_{\text{mp}} \vec{\phi}\left(x_{\text{mp}}^{\text{t}}\right) \tag{4.5}$$

$$\left(\vec{f}_{\text{int}}\right)^{\text{t}} = \sum_{mp=1}^{N_{\text{mp}}} \sigma_{\text{mp}}^{\text{t}} \vec{\phi}'\left(x_{\text{mp}}^{\text{t}}\right) V_{\text{mp}}^{\text{t}} \tag{4.6}$$

   In other words, Eq. 4.5 implies that for the i^{th} DOF Eq. 4.7 holds.

$$m_{\text{i}}^{\text{t}} = \sum_{mp=1}^{N_{\text{mp}}} m_{\text{mp}} \phi_{\text{i}}\left(x_{\text{mp}}^{\text{t}}\right) \tag{4.7}$$

   This direct mapping technique is typical for standard MPM. An improved approach is provided in Section 4.4.

2. The accelerations at the DOFs are obtained after combining the internal forces with any external forces.

$$\vec{a}^{\,\text{t}} = \left(\mathbf{M}^{\text{t}}\right)^{-1} \left[\left(\vec{f}_{\text{ext}}\right)^{\text{t}} - \left(\vec{f}_{\text{int}}\right)^{\text{t}}\right] \tag{4.8}$$

3. The velocity of each MP at time $t + \Delta t$ is determined.

$$v_{\text{mp}}^{\text{t}+\Delta\text{t}} = v_{\text{mp}}^{\text{t}} + \Delta t\, \vec{\phi}^{\,\text{T}}\left(x_{\text{mp}}^{\text{t}}\right)\, \vec{a}^{\,\text{t}} \quad \forall\, \text{mp} = \{1, 2, \ldots, N_{\text{mp}}\} \tag{4.9}$$

4. The velocities at the DOFs are subsequently obtained.

$$\vec{v}^{\,\text{t}+\Delta\text{t}} = \left(\mathbf{M}^{\text{t}}\right)^{-1} \sum_{mp=1}^{N_{\text{mp}}} m_{\text{mp}} \vec{\phi}\left(x_{\text{mp}}^{\text{t}}\right) v_{\text{mp}}^{\text{t}+\Delta\text{t}} \tag{4.10}$$

   where $\vec{v}$ is the velocity vector consisting of the velocities at the DOFs v_{i}.

5. The incremental displacement vector $\Delta\vec{u}$ is computed.

$$\Delta\vec{u}^{\,\text{t}+\Delta\text{t}} = \Delta t\, \vec{v}^{\,\text{t}+\Delta\text{t}} \tag{4.11}$$

6. After these steps, the remaining part of the MP properties is updated.

$$u_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} = u_{\mathrm{mp}}^{\mathrm{t}} + \vec{\phi}^{\,\mathrm{T}}\left(x_{\mathrm{mp}}^{\mathrm{t}}\right)\ \Delta\vec{u}^{\,\mathrm{t}+\Delta\mathrm{t}} \tag{4.12}$$

$$x_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} = x_{\mathrm{mp}}^{\mathrm{t}} + \vec{\phi}^{\,\mathrm{T}}\left(x_{\mathrm{mp}}^{\mathrm{t}}\right)\ \Delta\vec{u}^{\,\mathrm{t}+\Delta\mathrm{t}} \tag{4.13}$$

$$\Delta\varepsilon_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} = \nabla\boldsymbol{\phi}^{\,\mathrm{T}}\left(x_{\mathrm{mp}}^{\mathrm{t}}\right)\ \Delta\vec{u}^{\,\mathrm{t}+\Delta\mathrm{t}} \tag{4.14}$$

where $\Delta\varepsilon_{\mathrm{mp}}$ is the MP incremental strain.

The MP stress at time $t + \Delta t$ is computed from σ^{t} and $\Delta\varepsilon_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}}$ using a constitutive model.

Considering only one-dimensional elastic deformations, it follows from [103, 148] that for small and large deformations, the stress is given by Eqs. 4.15 and 4.16, respectively.

$$\sigma_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} = \sigma_{\mathrm{mp}}^{\mathrm{t}} + E\ \Delta\varepsilon_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} \qquad \forall\, mp = \{1, 2, \ldots, N_{\mathrm{mp}}\} \tag{4.15}$$

$$\sigma_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} = \sigma_{\mathrm{mp}}^{\mathrm{t}} + \left(E - \sigma_{\mathrm{mp}}^{\mathrm{t}}\right)\ \Delta\varepsilon_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} \qquad \forall\, mp = \{1, 2, \ldots, N_{\mathrm{mp}}\} \tag{4.16}$$

where E is the Young's modulus.

7. The volume and density of each MP are obtained from the volumetric strain increment $\varepsilon_{\mathrm{vol}}$ (Eqs. 4.17-4.18).

$$V_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} = \left(1 + \Delta\varepsilon_{\mathrm{vol,mp}}^{\mathrm{t}+\Delta\mathrm{t}}\right) V_{\mathrm{mp}}^{\mathrm{t}} \tag{4.17}$$

$$\rho_{\mathrm{mp}}^{\mathrm{t}+\Delta\mathrm{t}} = \frac{\rho_{\mathrm{mp}}^{\mathrm{t}}}{\left(1 + \Delta\varepsilon_{\mathrm{vol,mp}}^{\mathrm{t}+\Delta\mathrm{t}}\right)} \tag{4.18}$$

4.3 B-spline Material Point Method

BSMPM [218,236] replaces the piecewise-linear basis functions by higher-order B-splines. The univariate B-spline basis functions can be introduced using a *knot vector*, a sequence of ordered non-decreasing points in $\mathbb{R}$ that are called *knots*. A knot vector is denoted by $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{\mathrm{N_n+p+1}}\}$ with N_{n} and p being the number of basis functions and the polynomial order, respectively. The first and last knot are repeated $p+1$ times to make the resulting B-spline interpolatory at both end points. In contrast to linear basis functions, $\phi_{\mathrm{i,p}}$ is not interpolatory, that is, $\phi_{\mathrm{i,p}}(x_{\mathrm{j}}) \neq \delta_{\mathrm{i,j}}$.

The Cox-de Boor formula [65] defines B-spline basis functions recursively. For $p = 0$, the basis functions are provided in Eq. 4.19.

$$\phi_{\mathrm{i},0}(\xi) = \begin{cases} 1 & \text{if } \xi_{\mathrm{i}} \leq \xi < \xi_{\mathrm{i+1}} \\ 0 & \text{otherwise} \end{cases} \tag{4.19}$$

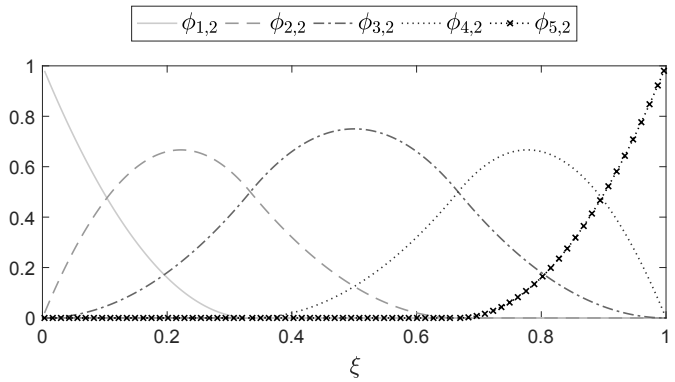

FIGURE 4.1: Example of quadratic B-spline basis functions.

For $p > 0$, the basis functions are given by Eq. 4.20

$$\phi_{\mathrm{i,p}}(\xi) = \frac{\xi - \xi_{\mathrm{i}}}{\xi_{\mathrm{i+p}} - \xi_{\mathrm{i}}}\phi_{\mathrm{i,p-1}}(\xi) + \frac{\xi_{\mathrm{i+p+1}} - \xi}{\xi_{\mathrm{i+p+1}} - \xi_{\mathrm{i+1}}}\phi_{\mathrm{i+1,p-1}}(\xi) \quad \xi \in \hat{\Omega} \tag{4.20}$$

Fig. 4.1 shows quadratic basis functions with $\Xi = \{0, 0, 0, 1/3, 2/3, 1, 1, 1\}$. B-spline basis functions satisfy the following properties.

1. They form a partition of unity:

$$\sum_{i=1}^{N_{\mathrm{n}}} \phi_{\mathrm{i,p}}(\xi) = 1 \quad \forall\, \xi \in \hat{\Omega} \tag{4.21}$$

2. Each $\phi_{\mathrm{i,p}}$ has compact support $[\xi_{\mathrm{i}}, \xi_{\mathrm{i+p+1}}]$.

3. They are non-negative in their support:

$$\phi_{\mathrm{i,p}}(\xi) \geq 0 \quad \forall\, \xi \in \hat{\Omega} \tag{4.22}$$

The gradients of the B-spline basis functions are defined as given in Eq. 4.23 [65].

$$\frac{\mathrm{d}\phi_{\mathrm{i,p}}(\xi)}{\mathrm{d}\xi} = \frac{p}{\xi_{\mathrm{i+p}} - \xi_{\mathrm{i}}}\phi_{\mathrm{i,p-1}}(\xi) - \frac{p}{\xi_{\mathrm{i+p+1}} - \xi_{\mathrm{i+1}}}\phi_{\mathrm{i+1,p-1}}(\xi) \tag{4.23}$$

The gradients corresponding to the basis functions from Fig. 4.1 are provided in Fig. 4.2.

B-spline basis functions bring many advantages over the typically used linear ones. First of all, they guarantee at least C_0-continuity of the gradients and hence, significantly reduce the grid crossing error. Similarly to piecewise-linear basis functions, B-splines enable lumping of the mass matrix due to their non-negativity and partition of unity properties, which is an essential for many engineering studies. In addition, they decrease the interpolation and time-stepping errors [219].

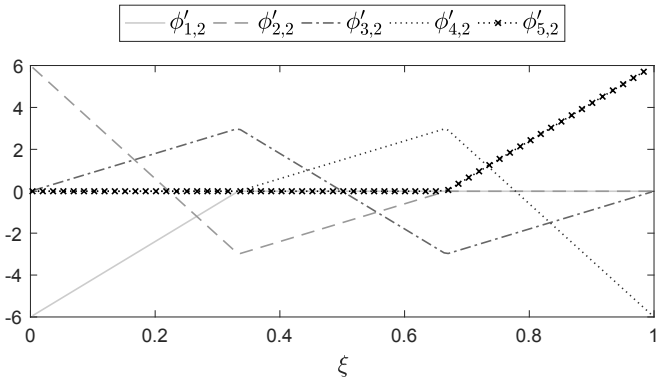

FIGURE 4.2: Example of the gradient of a quadratic B-spline basis function.

4.4 Mapping of material point data to background grid

In the MUSL algorithm, the computation of certain quantities involves a direct mapping of the MP data to the background grid. For example, the velocities at the DOFs are calculated from MP velocities and masses as shown in Eq. 4.10. This MPM mapping technique ensures the conservation of the mass $\mathcal{M}$ and linear momentum $\mathcal{P}$ of the system as follows.

$$\begin{aligned}\mathcal{M} &= \sum_{i=1}^{N_n} m_{\mathrm{i}} = \sum_{i=1}^{N_n} \left(\sum_{mp=1}^{N_{\mathrm{mp}}} m_{\mathrm{mp}} \phi_{\mathrm{i}}(x_{\mathrm{mp}}) \right) = \sum_{mp=1}^{N_{\mathrm{mp}}} m_{\mathrm{mp}} \sum_{i=1}^{N_n} \phi_{\mathrm{i}}(x_{\mathrm{mp}}) \\ &= \sum_{mp=1}^{N_{\mathrm{mp}}} m_{\mathrm{mp}} \end{aligned} \tag{4.24}$$

$$\begin{aligned}\mathcal{P} &= \sum_{i=1}^{N_n} m_{\mathrm{i}} v_{\mathrm{i}} = \sum_{i=1}^{N_n} m_{\mathrm{i}} \left(\frac{1}{m_{\mathrm{i}}} \sum_{mp=1}^{N_{\mathrm{mp}}} m_{\mathrm{mp}} \phi_{\mathrm{i}}(x_{\mathrm{mp}}) v_{\mathrm{mp}} \right) \\ &= \sum_{mp=1}^{N_{\mathrm{mp}}} m_{\mathrm{mp}} v_{\mathrm{mp}} \sum_{i=1}^{N_n} \phi_{\mathrm{i}}(x_{\mathrm{mp}}) = \sum_{mp=1}^{N_{\mathrm{mp}}} m_{\mathrm{mp}} v_{\mathrm{mp}} \end{aligned} \tag{4.25}$$

The above expressions are obtained from Eqs. 4.5 and 4.10, and by exploiting the partition of unity property of the piecewise-linear basis functions. Superscripts t and $\mathrm{t} + \Delta\mathrm{t}$ have been dropped to improve readability.

The MPM mapping can lead to significant numerical errors, especially when large deformations are considered [222]. For this reason, the TLS technique is discussed in this chapter. For each element, the TLS technique reconstructs quantities of interest, such as stress and density, from the MP data and

evaluates them at the integration points. After that, a numerical quadrature rule is applied to determine the internal forces and velocities at the DOFs. If the integration is exact, the proposed mapping approach preserves the total mass and linear momentum. The ideas are introduced for 1D, but can be extended to multiple dimensions in a straightforward manner. The proof of the conservation properties of the TLS technique can be found in Wobbes et al. [261].

4.4.1 Taylor Least Squares reconstruction

For the Least Squares approximation, a set of N_{mp} distinct data points $\{x_{\mathrm{mp}}\}_{mp=1}^{N_{\mathrm{mp}}}$ is considered. The generic data values of these points are denoted by $\{f(x_{\mathrm{mp}})\}_{mp=1}^{N_{\mathrm{mp}}}$. It is assumed that $f \in F$, where F is a normed function space on $\mathbb{R}$, and $P = \mathrm{span}\{\psi_{\mathrm{i}}\}_{i=1}^{n_{\mathrm{b}}} \subset F$ is a set of n_{b} basis functions. The Least Squares approximation at a point $x \in \mathbb{R}$ can be written as in Eq. 4.26.

$$\tilde{f}(x) = \sum_{i=1}^{n_{\mathrm{b}}} \alpha_{\mathrm{i}} \psi_{\mathrm{i}}(x) = \vec{\psi}^{\mathrm{T}}(x)\vec{\alpha} \quad \text{with} \quad \vec{\alpha} = \mathbf{D}^{-1}\mathbf{B}\vec{F} \tag{4.26}$$

where $\vec{\alpha}$ is the vector of coefficients for the basis functions obtained using Eqs. 4.27, 4.28 and 4.29.

$$\mathbf{D} = \sum_{mp=1}^{N_{\mathrm{mp}}} \vec{\psi}(x_{\mathrm{mp}})\vec{\psi}^{\mathrm{T}}(x_{\mathrm{mp}}) \tag{4.27}$$

$$\mathbf{B} = [\vec{\psi}(x_1) \quad \vec{\psi}(x_2) \quad \dots \quad \vec{\psi}(x_{N_{\mathrm{mp}}})] \tag{4.28}$$

$$\vec{F} = [f(x_1) \quad f(x_2) \quad \dots \quad f(x_{\mathrm{N_{mp}}})]^{\mathrm{T}} \tag{4.29}$$

The basis for P is formed by the local Taylor basis functions, which are defined using the concept of the volume average of a function u over Ω_e shown in Eq. 4.30.

$$\overline{f} = \frac{1}{|\Omega_{\mathrm{e}}|} \int_{\Omega_e} f \, \mathrm{d}\Omega_{\mathrm{e}} \tag{4.30}$$

where $|\Omega_{\mathrm{e}}|$ is the volume of cell e.
For example, if the element is one dimensional (i.e, $\Omega_{\mathrm{e}} = [x_{\min}, x_{\max}]$ with $x_{\max} > x_{\min}$), then $|\Omega_{\mathrm{e}}| = x_{\max} - x_{\min}$.
The Taylor basis functions are then given by Eq. 4.31.

$$\psi_1 = 1 \quad \psi_2 = \frac{x - x_{\mathrm{c}}}{\Delta x} \quad \psi_3 = \frac{(x - x_{\mathrm{c}})^2}{2\Delta x^2} - \frac{\overline{(x - x_{\mathrm{c}})^2}}{2\Delta x^2} \tag{4.31}$$

$$\text{where} \quad x_c = \frac{x_{\max} + x_{\min}}{2} \quad \text{and} \quad \Delta x = \frac{x_{\max} - x_{\min}}{2}$$

An important quality of the Taylor basis that ensures the conservation property of the reconstruction technique is shown in Eq. 4.32 [142].

$$\int_{\Omega_\mathrm{e}} \psi_\mathrm{i} \, \mathrm{d}\Omega_\mathrm{e} = \begin{cases} |\Omega_\mathrm{e}| & \text{if } i = 1, \\ 0 & \text{if } i \neq 1. \end{cases} \tag{4.32}$$

Suppose that a function f has to be reconstructed in such way that its integral over Ω_e, $\int_{\Omega_\mathrm{e}} f(x) \, \mathrm{d}\Omega_\mathrm{e} = c$ ($c \in \mathbb{R}$), is preserved. The TLS approximation of f is equal to a linear combination of Taylor basis functions (Eq. 4.33).

$$f(x) \approx \tilde{f}(x) = \sum_{i=1}^{n_\mathrm{b}} a_\mathrm{i} \psi_\mathrm{i}(x) \tag{4.33}$$

Using Eq. 4.32, the integral of $\tilde{f}$ can then be written as Eq. 4.34.

$$\int_{\Omega_\mathrm{e}} \tilde{f}(x) \, d\Omega_\mathrm{e} = \int_{\Omega_\mathrm{e}} \sum_{i=1}^{n_\mathrm{b}} \alpha_\mathrm{i} \psi_\mathrm{i}(x) \, d\Omega_\mathrm{e} = \sum_{i=1}^{n_\mathrm{b}} \alpha_\mathrm{i} \int_{\Omega_\mathrm{e}} \psi_\mathrm{i}(x) \, d\Omega_\mathrm{e} = \alpha_1 |\Omega_\mathrm{e}| \tag{4.34}$$

Therefore, explicitly enforcing the condition shown in Eq. 4.35 conserves the integral.

$$\alpha_1 = \frac{c}{|\Omega_\mathrm{e}|} \tag{4.35}$$

Consequently, the number of basis functions in the Least Squares approximation is reduced by one.

4.4.2 Example of Taylor Least Squares reconstruction

The outstanding properties of the TLS technique can be illustrated by reconstructing $f(x) = \sin(x) + 2$ on $[0, 4\pi]$. In this case, the integral that should be preserved is equal to 8π. The domain is discretised using four elements of size π and contains 11 data points. Two data points are located at the boundaries of the first element, (i.e., 0 and π). In $[2\pi, 3\pi]$, the data points are distributed uniformly in the interior of the domain. The remaining data points have random positions creating different types of data distribution within each element.

The TLS approximation is obtained using three Taylor basis functions. Fig. 4.3 visualises the data-point distribution for 10 integration points per element. The overall performance of the TLS technique is quantified by the Root-Mean-Square (RMS) error for function f and the relative error for its integral. Both errors are computed using 10 Gauss points within each element. The RMS error is equal to $3.8139 \cdot 10^{-2}$, while the relative error for the integral is equal to $7.0679 \cdot 10^{-16}$. It should be noted that the relative error for the integral computed with only two Gauss points per element is equal to $2.7903 \cdot 10^{-15}$. Thus, the TLS approach preserves the integral up to machine precision for this example.

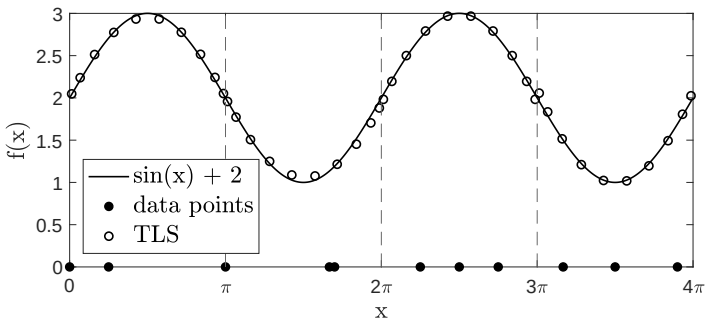

FIGURE 4.3: TLS reconstructions of $f(x) = \sin(x) + 2$ on $[0, 4\pi]$ for different types of data point distribution within an element.

4.4.3 Integration of Taylor Least Squares reconstruction into MPM algorithm

When the TLS reconstruction is considered as part of the MPM algorithm, MPs serve as data points. In order to ensure an accurate and conservative mapping of the information from MPs to the grid, the technique is combined with a Gauss quadrature rule with a suitable number of Gauss points. However, Gauss quadrature can be replaced by any numerical integration that provides an exact result.

The TLS technique is applied to replace the MPM integration in Eq. 4.6. Since in this case, the conservation is not required, all unknown coefficients of the Taylor basis functions are obtained from the least-square approximation (i.e. coefficient a_1 is not enforced by Eq. 4.35. Thus, the internal forces at the DOFs are computed as follows.

1. Apply TLS approximation to reconstruct the stress field from the MP data within each active element without specifying the coefficient of the first Taylor basis function (Eq. 4.36).

$$\tilde{\sigma}_{\mathrm{e}} = \sum_{i=1}^{n_{\mathrm{b}}} s_{\mathrm{i}} \psi_{\mathrm{i}} \tag{4.36}$$

   where s_{i} is the coefficient corresponding to the ith Taylor basis function. Outside of Ω_{e}, $\tilde{\sigma}_{\mathrm{e}}$ is zero.

   The global approximation of the stress function, $\tilde{\sigma}$, is then given by Eq. 4.37.

$$\tilde{\sigma} = \sum_{e=1}^{N_e} \tilde{\sigma}_{\mathrm{e}} \tag{4.37}$$

2. Integrate the stress approximation using a Gauss quadrature (Eq. 4.38).

$$\vec{f}_{\text{int}} \approx \int_{\Omega} \tilde{\sigma}(x,t)\vec{\phi}'\, d\Omega = \sum_{g=1}^{N_{\text{g}}} \tilde{\sigma}\left(x_g\right) \vec{\phi}'\left(x_{\text{g}}\right) \omega_{\text{g}} \tag{4.38}$$

   where N_{g} is the total number of Gauss points, x_{g} the global position of a Gauss point, and ω_{g} the weight of a Gauss point.

   For exact integration of the approximated function, each active element should contain $N_{\text{g}}/N_{\text{e}}$ Gauss points, where N_{e} is the total number of elements (knot spans) and N_{g} satisfies $n_{\text{b}} \leq 2N_{\text{g}}/N_{\text{e}}$. This implies that for a quadratic TLS approach the numerical integration requires at least two Gauss points per element.

The TLS technique is also used to map the MP velocities to the DOFs (i.e. it replaces Eq. 4.10). However, the coefficient of the first basis function is specified according to Eq. 4.35. The remaining coefficients are calculated from Eq. 4.26 without ψ_1 to avoid changing of the integral value.

1. Apply TLS approximation to reconstruct the density field and momentum, which is given by the product of density and velocity from the MP data within each active element, while preserving the mass and momentum of the element (Eqs. 4.39 and 4.40).

$$\tilde{\rho}_{\text{e}} = \sum_{i=1}^{n_{\text{b}}} r_{\text{i}}\psi_{\text{i}} \;\text{ with }\; r_1 = \frac{1}{|\Omega_{\text{e}}|} \sum_{\{p|x_{\text{mp}}\in\Omega_{\text{e}}\}} m_{\text{mp}} \tag{4.39}$$

$$(\widetilde{\rho v})_{\text{e}} = \sum_{i=1}^{n_{\text{b}}} \gamma_{\text{i}}\psi_{\text{i}} \;\text{ with }\; \gamma_1 = \frac{1}{|\Omega_{\text{e}}|} \sum_{\{p|x_{\text{mp}}\in\Omega_{\text{e}}\}} m_{\text{mp}} v_{\text{mp}} \tag{4.40}$$

   where r_{i} and γ_{i} are the coefficients corresponding to the i-th Taylor basis function. Outside of Ω_{e}, $\tilde{\rho}_{\text{e}}$ and $(\widetilde{\rho v})_{\text{e}}$ are zero.

   The global approximations are given by Eq. 4.41.

$$\tilde{\rho} = \sum_{e=1}^{N_{\text{e}}} \tilde{\rho}_{\text{e}} \;\text{ and }\; (\widetilde{\rho v}) = \sum_{e=1}^{N_{\text{e}}} (\widetilde{\rho v})_{\text{e}} \tag{4.41}$$

2. Integrate the approximations using a Gauss quadrature to obtain the momentum vector $\vec{p}$ and the consistent mass matrix $\mathbf{M}^{\text{C}}$ (Eqs. 4.42-4.43).

$$\vec{p} = \sum_{g=1}^{N_{\text{g}}} (\widetilde{\rho v})\left(x_{\text{g}}\right) \omega_{\text{g}} \vec{\phi}\left(x_{\text{g}}\right) \tag{4.42}$$

$$\mathbf{M}^{\text{C}} = \sum_{g=1}^{N_{\text{g}}} \tilde{\rho}\left(x_{\text{g}}\right) \omega_{\text{g}} \vec{\phi}\left(x_{\text{g}}\right) \left(\vec{\phi}\left(x_{\text{g}}\right)\right)^{\text{T}} \tag{4.43}$$

As previously mentioned, the number of Gauss points per element should be specified so that exact integration is ensured.

3. Compute the velocity vector (Eq. 4.44).

$$\vec{v} = (\mathbf{M}^{\mathrm{C}})^{-1}\vec{p} \tag{4.44}$$

$\mathbf{M}^{\mathrm{C}}$ can be replaced in Eq. 4.44 by a lumped mass matrix without losing the conservation properties of the algorithm. A consistent mass matrix typically provides more accurate results, but may lead to stability issues [261].

4.5 Application to vibrating bar problem

This section compares the performance of the standard MPM that follows the MUSL algorithm with its more advanced versions, such as BSMPM and BSMPM with the TLS reconstruction (TLS-BSMPM). The comparison is done based on an example that describes the vibration of a 1-phase bar with fixed ends. A system of partial differential equations for the velocity and stress captures the motion (Eqs. 4.45 and 4.45).

$$\rho\frac{\partial v}{\partial t} = \frac{\partial \sigma}{\partial x} \tag{4.45}$$

$$\frac{\partial \sigma}{\partial t} = E\frac{\partial v}{\partial x} \tag{4.46}$$

where v is the displacement. v is displacement or velocity? conflict with following equation. Most chapters have v for velocity and u for displacement.

The system is extended by a relation between the velocity v and displacement u given by Eq. 4.47.

$$v = \frac{\partial u}{\partial t} \tag{4.47}$$

The vibration is triggered by an initial velocity that varies along the bar leading to the following initial and boundary conditions (Eqs. 4.48 and 4.49).

$$u(x,0) = 0, \quad v(x,0) = v_0 \sin\left(\frac{\pi x}{h}\right), \quad \sigma(x,0) = 0; \tag{4.48}$$

$$u(0,t) = 0, \quad u(h,t) = 0 \tag{4.49}$$

where H is the length of the bar and v_0 is the maximum initial velocity.

For small deformations, the analytical solution in terms of displacement,

velocity, and stress is given by Eqs. 4.50 to 4.52.

$$u(x,t) = \frac{v_0 h}{\pi\sqrt{E/\rho}} \sin\left(\frac{\pi\sqrt{E/\rho}\,t}{h}\right) \sin\left(\frac{\pi x}{h}\right) \tag{4.50}$$

$$v(x,t) = v_0 \cos\left(\frac{\pi\sqrt{E/\rho}\,t}{h}\right) \sin\left(\frac{\pi x}{h}\right) \tag{4.51}$$

$$\sigma(x,t) = v_0\sqrt{E\rho} \sin\left(\frac{\pi\sqrt{E/\rho}\,t}{h}\right) \cos\left(\frac{\pi x}{h}\right) \tag{4.52}$$

Table 4.1 gives the parameter values for the vibrating bar benchmark under small deformations. The contribution of the temporal errors to the overall error generated during the computation is minimised by selecting a small time-step size and short simulation time. To be more precise, the time-step size and total simulation time are set to $1 \cdot 10^{-7}$ s and $1.9 \cdot 10^{-6}$ s, respectively. Furthermore, the number of elements (knot spans) is varied from 5 to 40, while the number of MPs per cell or knot span (PPC) is fixed to 12. Grid crossing does not occur, and the maximal observed strain is equal to $5.3 \cdot 10^{-7}$ m.

The obtained results are illustrated in Fig. 4.4. As expected, MPM shows second-order convergence in the displacement and velocity. Since the stress is computed as a derivative of the displacement, its convergence rate is one. The use of BSMPM with the standard MPM mapping leads to a lower RMS error and third-order convergence for the velocity, but hinders the performance of the method for the displacement and stress. The poor performance of the method in terms of displacement and stress is caused by the large values of the error at the boundaries of the domain. This is illustrated in Fig. 4.5 for the stress distribution.

The TLS reconstruction in conjunction with a lumped mass matrix has little influence on the convergence behaviour of BSMPM. For this reason, the corresponding results are not shown in Fig. 4.4. However, the TLS technique eliminates the boundary issues due to the B-spline basis functions when a consistent mass matrix is used for the mapping of MP information. As a result, it significantly decreases the RMS error in the displacement and stress and leads to a higher convergence order for both quantities. The performance

TABLE 4.1
Parameters for small and large deformation vibrating bar.

Parameter	**Symbol**	**Unit**	**Value**
Height	h	m	1.00
Density	ρ	kg/m^3	$2.00 \cdot 10^3$
Young's modulus	E	kPa	$7.00 \cdot 10^3$
Max. initial velocity (small def.)	v_0	m/s^2	0.28
Max. initial velocity (large def.)	v_0	m/s^2	0.80

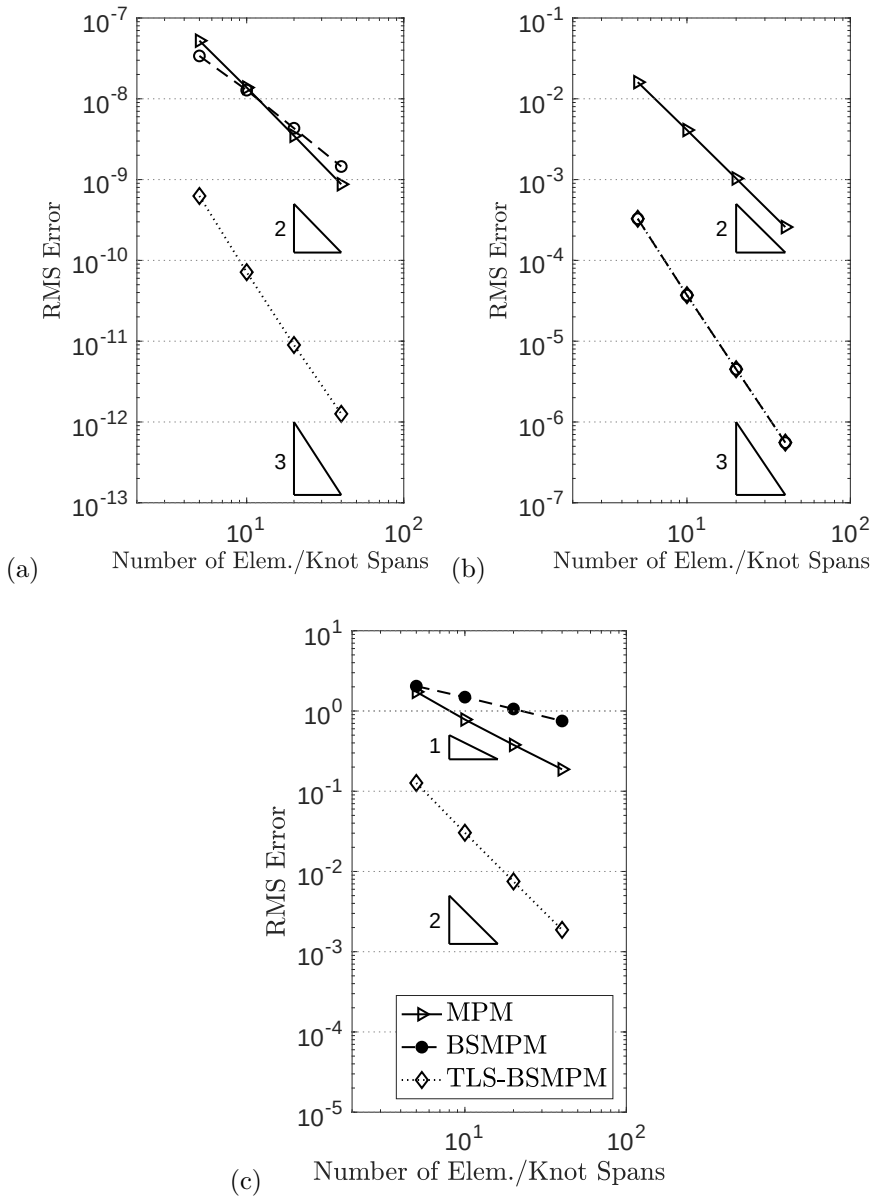

FIGURE 4.4: Spatial convergence of MP methods for the vibrating bar problem without grid crossing: (a) displacement, (b) velocity, and (c) stress.

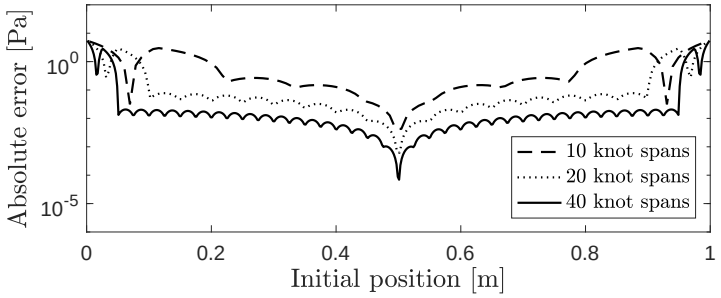

FIGURE 4.5: Absolute error obtained with BSMPM for stress distribution in the vibrating bar problem without grid crossing.

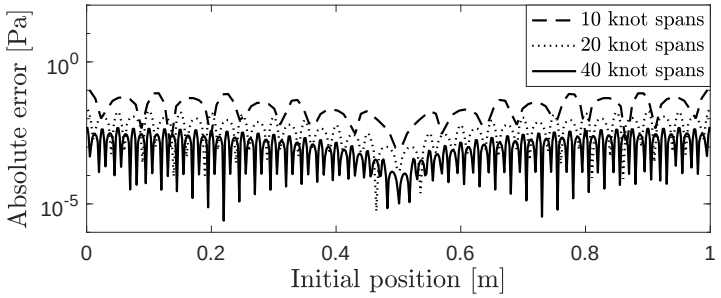

FIGURE 4.6: Absolute error obtained with TLS-BSMPM for stress distribution in the vibrating bar problem without grid crossing.

of the TLS function reconstruction technique in terms of the absolute error for the stress is depicted in Fig. 4.6. Moreover, the mapping with the TLS reconstruction preserves the relative error in the total mass and momentum under $7.5033 \cdot 10^{-15}$ and $2.1007 \cdot 10^{-16}$, respectively.

For large deformations, the parameters from Table 4.1 are used and the simulation time is increased to 0.1 s. The computations are performed with the time-step size of $1 \cdot 10^{-5}$ s, 20 elements (knot spans) and initially 8 MPs in each cell. Since the analytical solution is not available when large strains are considered, the results are compared to the solution generated with Updated Lagrangian Finite Element Method (ULFEM) [22] using 120 DOFs.

Under large deformations, the MPs cross the element boundaries more than 450 times when the standard MPM is used. This results in unphysical oscillations in the solution for the stress as shown in Fig. 4.7. The use of the B-spline basis functions reduces the oscillatory behaviour, but still significantly deviates from the ULFEM solution. When the TLS reconstruction (with a consistent mass matrix) is applied as well, the solutions improves significantly leading to the reduction of the MPM error by a factor of 9.8. The method limits

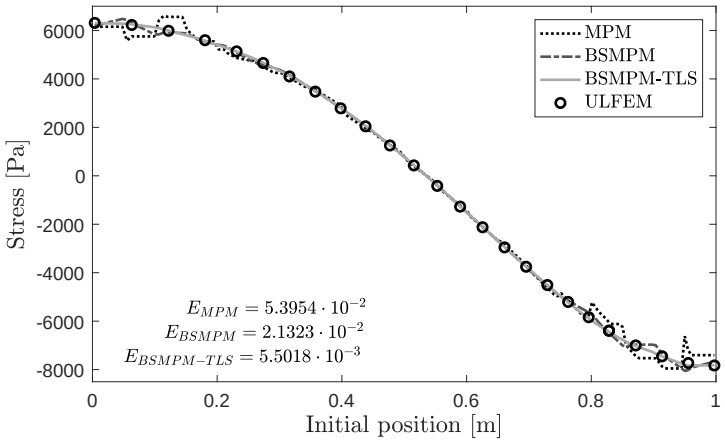

FIGURE 4.7: Stress distribution and corresponding relative errors in the L_2-norm in the vibrating bar with grid crossing.

the error in the mass and linear momentum of the system to $2.9104 \cdot 10^{-14}$ and $5.7205 \cdot 10^{-15}$ during the simulation.

The vibrating bar example demonstrates that BSMPM with the standard MPM mapping considerably reduces the unphysical oscillations originating from grid crossing. However, its performance can be hindered by the issues at the boundaries of the domain. The application of the TLS reconstruction ensures an accurate solution at the boundaries leading to higher-order convergence. As a result, TLS-BSMPM significantly improves the solution of the standard MPM for small and large deformations. The obtained results also show that TLS-BSMPM conserves the mass and linear momentum of the system up to machine precision. Thus, the method preserves the physical properties of the standard MPM.

For this example, the use of a consistent mass matrix for the velocity computation in TLS-BSMPM is vital for the optimal performance of the method. Although not considered here, the consistent mass matrix in Eq. 4.1 can further improve the accuracy and physical properties of the algorithm. Therefore, it is important to ensure an efficient solution of the resulting linear systems.

4.6 Iterative solvers

When solving a linear system of equations, two general solution strategies can be distinguished. Direct solvers, such as Gaussian Elimination and Cholesky Decomposition, determine the solution of a linear system of equations di-

rectly. However, since direct solvers require a high amount of computational resources, they are less preferable in the solution of large linear systems. Direct solvers can be used though in a different setting, for example as a preconditioner [197].

Alternatively, iterative solvers can be adopted, in which an initial guess is updated successively until a converged end-of-step solution has been reached. Hence, when considering Eq. 4.1, a sequence of approximations $\vec{a}^{(0)}, \vec{a}^{(1)}, \vec{a}^{(2)}, \ldots$ is constructed based on an initial guess $\vec{a}^{(0)}$. For each solution $\vec{a}^{(\mathrm{n})}$, the corresponding residual vector $\vec{r}^{(\mathrm{n})} = \vec{f} - \mathbf{M}^{\mathrm{C}}\vec{a}^{(\mathrm{n})}$ is determined. Once the residual is smaller then a predefined tolerance, the method is said to have converged and the solution at the corresponding iteration is used. Different types of iterative methods can be distinguished. Basic iterative methods like the (damped) Jacobi or Gauss-Seidel method are easy to implement but require a relatively large number of iterations. The more advanced Krylov subspace methods like the Conjugate Gradient and GMRES method define acronym show better convergence rates, which results in a smaller number of iterations needed to converge.

Next to the choice of the iteration scheme, one or more stopping criteria have to be chosen, such as Eq. 4.53.

$$\frac{||\vec{r}^{(\mathrm{k})}||_2}{||\vec{r}^{(0)}||_2} < \epsilon \tag{4.53}$$

Apart from the adopted stopping criterion and iterative method, the properties of the corresponding matrix play an import role in the performance of the iterative solver. In particular, the condition number influences the rate of convergence of the chosen iterative method. The condition number $\kappa(\mathbf{M})$ of a matrix $\mathbf{M} \in \mathbb{R}^{\mathrm{N}\cdot\mathrm{N}}$ in 2-norm is defined as given in Eqs. 4.54 and 4.55

$$\kappa(\mathbf{M}) := ||\mathbf{M}||_2 \cdot ||\mathbf{M}^{-1}||_2 \tag{4.54}$$

$$||\mathbf{M}||_2 := \sqrt{\lambda_{\max}(\mathbf{M}^\top \mathbf{M})} \tag{4.55}$$

where $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue of the corresponding matrix. In case the matrix is symmetric and positive definite (SPD), the condition number is given by Eq. 4.56 [197].

$$\kappa(\mathbf{M}) := \frac{\lambda_{\max}(\mathbf{M})}{\lambda_{\min}(\mathbf{M})} \tag{4.56}$$

To illustrate the influence of the condition number on the convergence of iterative methods, the Conjugate Gradient method is considered. The approximated solution after n iterations, $\vec{a}^{(\mathrm{n})}$, obtained with the Conjugate Gradient method and the exact solution $\vec{a}$ satisfy inequality given in Eq. 4.57 [94].

$$||\vec{a} - \vec{a}^{(\mathrm{n})}|| \leq 2 \left(\frac{\sqrt{\kappa(\mathbf{M})} - 1}{\sqrt{\kappa(\mathbf{M})} + 1} \right)^{\mathrm{n}} ||\vec{a} - \vec{a}^{(0)}|| \tag{4.57}$$

TABLE 4.2
$\kappa(\mathbf{M})$ for different mesh widths and orders of B-spline basis functions.

$\mathbf{p} \setminus \mathbf{h^{-1}}$	**8**	**16**	**32**	**64**
1	$7.38 \cdot 10^0$	$8.55 \cdot 10^0$	$8.89 \cdot 10^0$	$8.97 \cdot 10^0$
2	$5.21 \cdot 10^1$	$5.51 \cdot 10^1$	$5.60 \cdot 10^1$	$5.62 \cdot 10^1$
3	$4.40 \cdot 10^2$	$4.24 \cdot 10^2$	$4.20 \cdot 10^2$	$4.22 \cdot 10^2$
4	$3.77 \cdot 10^3$	$3.36 \cdot 10^3$	$3.29 \cdot 10^3$	$3.30 \cdot 10^3$
5	$3.30 \cdot 10^4$	$2.61 \cdot 10^4$	$2.52 \cdot 10^4$	$2.52 \cdot 10^4$

Hence, the factor at which the initial difference between the approximation and the exact solution $\vec{a}$ decreases every iteration, depends heavily on the condition number of the matrix. High values of $\kappa(\mathbf{M})$ imply a slow convergence, while low values of $\kappa(\mathbf{M})$ result in fast convergence.

Within MPM, the condition number of the mass matrix determines the rate of convergence when an iterative method is used to solve Eqs. 4.1 and 4.10. Therefore, the condition number of the mass matrix for different orders of the B-spline basis functions p and mesh widths h is presented in Table 4.2.

Although the numerical estimates are obtained on relatively coarse meshes, the strong dependence of the condition number on the approximation order p can be observed. Hence, a spatial discretisation with high-order B-spline basis functions leads to an ill-conditioned linear system of equations.

To illustrate the effect of the condition number on the performance of iterative solvers, Eq. 4.1 is considered and has to be solved in every time step with the mass matrix is defined by Eq. 4.58.

$$\mathbf{M}^{\mathrm{C}} = \int_{\Omega} \rho \vec{\phi} \vec{\phi}^{\mathrm{T}} \, \mathrm{d}\Omega \tag{4.58}$$

where the B-spline basis functions ϕ can be chosen of arbitrary order p.
The force vector $\vec{f}$ is chosen to be constant, simulating a gravitational force of 9.81 in the negative x- and y-direction. The resulting linear system is then solved with the Conjugate Gradient method for different approximation orders p. A tolerance of $\epsilon = 1 \cdot 10^{-8}$ is chosen combined with a zero initial guess. The number of iterations needed with the CG method, which can be adopted since $\mathbf{M}^{\mathrm{C}}$ is SPD, are presented in Table 4.3.

The effect of the condition number on the number of iterations needed before the CG method converges is clearly visible: a high condition number leads to a high number of CG-iterations. The use of high-order B-spline basis function therefore leads to a new challenge: The efficient solution of linear systems arising in IgA discretisations.

TABLE 4.3
Number of iterations needed with the CG method for different values of p and h.

$\mathbf{p} \setminus \mathbf{h^{-1}}$	**16**	**32**	**64**	**128**
1	18	18	18	17
2	34	46	46	44
3	68	93	95	91
4	93	178	196	187
5	156	242	252	227

4.7 Efficient solvers for Isogeometric Analysis

Sangali and Tani [199] investigated the preconditioners based on the solution of the Sylvester equation and Collier et al. [58] the use of direct solvers. An alternative class of iterative solvers are multigrid methods. Multigrid methods aim to solve linear systems of equations by using a hierarchy of discretisations (Fig. 4.8). At each level of the hierarchy a basic iterative method (e.g. Jacobi, Gauss-Seidel) is applied (smoothing), whereas on the coarsest level a correction is determined (coarse grid correction). This correction is then transferred back to the finest level and used to update the solution. Information between different levels is transferred using prolongation and restriction operators.

Starting from the finest level, different strategies can be adopted to traverse the hierarchy, leading to different cycle types. Fig. 4.8 illustrates the most common cycle type, the V-cycle.

The hierarchy can be obtained in various ways. With h-multigrid, each level of the hierarchy corresponds to a discretisation with a certain mesh width h. Typically, one chooses h, $2h$, $4h$, ... to construct the hierarchy. h-Multigrid

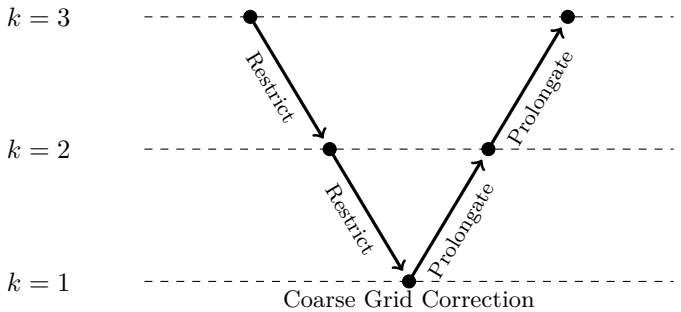

FIGURE 4.8: Description of a V-cycle

methods for IgA discretisations have been studied in [88, 104]. An alternative solution strategy is the use of p-multigrid based methods, in which a hierarchy is constructed based on discretisations of different approximation orders. the linear system is considered to illustrate the structure of p-multigrid methods. Should this Eq. be cross-referenced in this paragraph?

$$\mathbf{M}^{\mathrm{C}}_{\mathrm{h,p}} \vec{a}_{\mathrm{h,p}} = \vec{f}_{\mathrm{h,p}} \tag{4.59}$$

Eq. 4.59 results from a discretisation with B-spline basis functions of order p and mesh width h. Starting from an initial guess $\vec{a}^{0}_{\mathrm{h,p}}$, a single iteration of the (two-grid) p-multigrid method consists of the following steps [235].

1. Apply ν_1 pre-smoothing steps on Eq. 4.59 using Eq. 4.60.

$$\vec{a}^{(0,\mathrm{m}+1)}_{\mathrm{h,p}} = \vec{a}^{(0,\mathrm{m})}_{\mathrm{h,p}} + \mathcal{S}(\vec{f}_{\mathrm{h,p}} - \mathbf{M}^{\mathrm{C}}_{\mathrm{h,p}} \vec{a}^{(0,\mathrm{m})}_{\mathrm{h,p}}), \quad \mathrm{m} = 0, \ldots, \nu_1 - 1 \tag{4.60}$$

   where $\mathcal{S}$ is a smoother (i.e. a basic iterative method like Jacobi or Gauss-Seidel).

2. Project the residual from level p onto level $p-1$ using the restriction operator $I^{\mathrm{p}-1}_{\mathrm{p}}$ and solve the residual equation given by Eq. 4.61 at level $p-1$ to obtain the coarse grid correction.

$$\mathbf{M}^{\mathrm{C}}_{\mathrm{h,p}-1} \vec{e}_{\mathrm{h,p}-1} = \vec{r}_{\mathrm{h,p}-1} \tag{4.61}$$

   where $\vec{e}_{\mathrm{h,p}-1}$ denotes the error (or correction) at the coarse level.

3. Project the correction $\vec{e}_{\mathrm{h,p}-1}$ onto level p using the prolongation operator $I^{\mathrm{p}}_{\mathrm{p}-1}$ and update $\vec{a}^{(0,\nu_1)}_{\mathrm{h,p}}$ using Eq. 4.62.

$$\vec{a}^{(0,\nu_1)}_{\mathrm{h,p}} := \vec{a}^{(0,\nu_1)}_{\mathrm{h,p}} + I^{\mathrm{p}}_{\mathrm{p}-1}(\vec{e}_{\mathrm{h,p}-1}) \tag{4.62}$$

4. Apply ν_2 post-smoothing steps to Eq. 4.59 to obtain $\vec{a}^{(0,\nu_1+\nu_2)}_{\mathrm{h,p}} =: \vec{a}^{1}_{\mathrm{h,p}}$

The two-grid multigrid method can be applied recursively until level $p = 1$ has been reached, which results in a V-cycle. Alternatively, different schemes can be applied.

Eq. 4.61 is solved at level $p = 1$ using a Conjugate Gradient (CG) solver. Eq. 4.53 with $\epsilon = 10^{-4}$ is chosen as a stopping criterion for the CG method. A detailed description of the prolongation and restriction operator can be found in [235].

The mass matrices, which are needed at each level for the smoothing procedure, are obtained by rediscretisation. The solution $\vec{a}^{1}_{\mathrm{h,p}}$ is used as an initial guess for the next cycle.

The advantage of p-multigrid methods is the fact that on the 'coarsest' level a linear system is solved, where the mass matrix has a more favourable condition number. Furthermore, since they coincide with B-spline basis functions

TABLE 4.4
Number of V-cycles needed with p-multigrid for different values of p and h.

$\mathbf{p \setminus h^{-1}}$	**8**	**16**	**32**	**64**
1	1	1	1	1
2	4	4	4	3
3	15	12	10	9
4	22	17	13	11
5	58	38	27	20

for $p = 1$, established solution techniques for Lagrange Finite Elements can be used. The potential of p-multigrid is illustrated by considering, again, Eq. 4.1 resulting, again, from applying a force in the negative x- and y-direction. The number of pre- and post-smoothing steps is identical for all numerical experiments ($\nu_1 = \nu_2 = 8$).

The number of V-cycles needed with p-multigrid are presented in Table 4.4 and can be compared with the ones presented in Table 4.3. Note that the number of cycles needed with p-multigrid is significantly lower compared to the iterations needed with the CG method. Furthermore, p-multigrid, as standard h-multigrid methods, exhibits the h-independence property, implying that the number of cycles needed before the method has converged is independent of the mesh width h.

Numerical results indicate that p-multigrid methods can be used as an efficient solution technique to solve the equations arising in MPM.

4.8 Closure

The combined use of B-spline basis functions and a TLS reconstruction (TLS-BSMPM) significantly improves the solution of the standard MPM algorithm. For small strains, smaller errors and, in many cases, higher convergence rates can be obtained, while for large deformations, TLS-BSMPM reduces the oscillations originating from grid crossing. Furthermore, in contrast to many standard reconstruction techniques, the total mass and linear momentum are conserved provided a sufficiently accurate numerical quadrature method is adopted.

The use of the consistent mass matrix instead of the lumped one for the mapping of MP data to the background grid considerably improves the solution. However, the use of the consistent mass matrix within BSMPM requires efficient solution techniques for high order B-spline discretisations. Since the condition number grows exponentially with p in this case, (standard) iterative

methods are inefficient. p-Multigrid methods, in contrast, have been shown to be an efficient solution approach that requires a significantly smaller number of iterations as compared to CG in this case.