Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 00 (2016) 000–000

Procedia Engineering

www.elsevier.com/locate/procedia

1st International Conference on the Material Point Method, MPM 2017

An enhanced smoothing algorithm for MPM to stabilize hydrodynamic impact problems with embedded solids

Wen-Chia Yang[a], Greg Miller[b], Pedro Arduino[b], Peter Mackenzie-Helnwein[b,*]

[a]*Graduate Student, Department of Civil and Environmental Engineering, University of Washington, Seattle, WA 98195-2700, United States*
[b]*Faculty, Department of Civil and Environmental Engineering, University of Washington, Seattle, WA 98195-2700, United States*

Abstract

The impact of debris carried by floods or tsunamis can cause severe damage to structures, but the complex phenomena involved are difficult to model. The Material Point Method (MPM) provides one framework for modeling such systems, with the capability of incorporating combined fluid/solid behavior with complex interaction. Conventional MPM uses regular grids with tri-linear interpolation. However, linear functions introduce volumetric locking for (nearly) incompressible materials, posing problems when modeling liquids. To eliminate locking, hybrid formulations similar to those used with finite elements were adapted by Mast et al. [1]. This approach introduced two classes of anti-locking algorithms for nearly incompressible materials: a cell-based and a node-based variant. Both variants filter incompatible strains and stresses, but also affect the stability of the time integration. For hydrodynamic problems the cell-based algorithm is prone to checker-board stress fields, while the node-based algorithm can introduce excessive dissipation. This paper presents a new numerical flux smoothing algorithm to produce smooth stress fields in complex hydrodynamic problems while enhancing numerical stability. The goal is to combine the stability of the node-based anti-locking approach with the cell-based variant's capability to effectively solve hydrostatic problems. The improved algorithm is validated using a hydrostatic problem to isolate and minimize the effect of integration errors. A complex hydrodynamic problem involving an embedded solid block is then used as an example to display the new algorithm's modeling capabilities.

Peer-review under responsibility of the organizing committee of the 1 st International Conference on the Material Point Method.

Keywords: material point method; smoothing; debris; impact; fluid-structure interaction.

1. Introduction

Throughout history, tsunamis have caused severe damage in coastal areas. As coastal populations and the associated infrastructures continue to increase around the world, understanding and managing tsunami effects on structures becomes increasingly important. In the literature, tsunami induced hydrodynamic loads on structures have been widely studied. However, tsunamis can also carry large debris, e.g. shipping containers and boats. Many researchers have experimentally demonstrated that debris carried by water flow (such as in the case of tsunamis) can cause large impact forces on structural components (e.g., [2–4]). These problems have not yet been well-studied numerically, largely be-

* Corresponding author.
E-mail address: pmackenz@uw.edu

1877-7058
Peer-review under responsibility of the organizing committee of the 1 st International Conference on the Material Point Method.

cause the complex interactions between multiple solids and fluids are not easily modeled using typical fluid-oriented or solid-oriented numerical frameworks. The material point method (MPM)[5] provides a unified fluid/solid interaction platform based on updated Lagrangian computational grids, and this enables the modeling of these complex fluid-solid (moving and stationary) interactions.

The standard MPM implementation uses a regular orthogonal grid with tri-linear shape functions. While this is not the only variant in use (see, e.g., [6]), it remains popular for its simplicity. However, the linear shape functions also introduce not only volumetric locking for (nearly) incompressible materials (hence causing problems when modeling liquids), but also integration errors, which arise from the MPM particle discretization. Each of these issues is problematic, and must be addressed to generate useful solutions.

Figure 1(a) illustrates the issue of volumetric locking through filling of a stationary tank using standard MPM [5] and tri-linear shape functions. To eliminate this kind of non-physical behavior, hybrid formulations are generally used in (mixed) finite element methods. This approach was adapted to MPM by Mast et al.[1] as a filter-step between the particle strain update and particle stress update. Two of the three investigated anti-locking algorithms from that study are specifically relevant for fluid modeling: (i) the cell-based formulation; and (ii) the node-based formulation. The characteristics of these algorithms can be demonstrated with simulations modeling water injected into a rigid rectangular box as illustrated in Figure 1. Detailed analysis reveals that these algorithms are able to not only cure the locking problem, but also implicitly stabilize the analysis by smoothing state variable fields and hence dissipating strain energy. In this article, we define *anti-locking* as an energy conserving procedure, while *smoothing* refers to a diffusive, thus, dissipative procedure.

The objective of this paper is to provide an improved smoothing strategy to enable: (i) stable simulation of fluids and fluid-structure interaction; and (ii) realistic analysis when no integration errors are involved. The key target for the proposed smoothing strategy are MPM implementations using multi-linear interpolation functions and single-point-quadrature elements. In the following, the anti-locking strategy applied in this study is first described, and the newly developed numerical flux smoothing algorithms are presented. Validation and application examples are used to demonstrate the effectiveness of the approach, which is further discussed in Section 6.

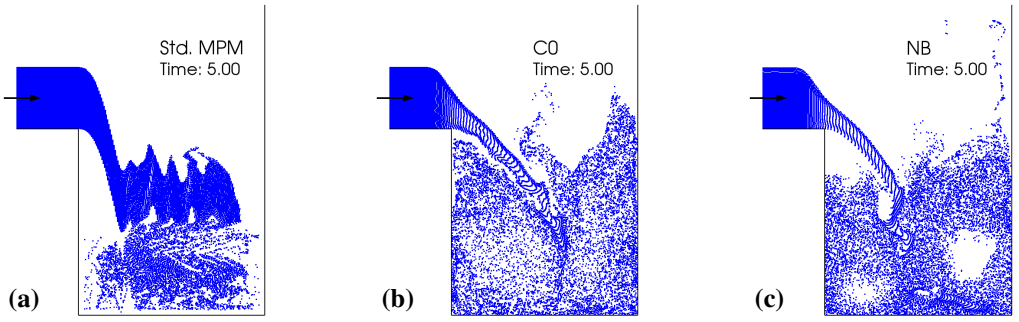

Fig. 1. Comparison of hydro-dynamical simulation snapshots between (a) standard MPM, (b) the cell-based algorithm (C0) and (c) the node-based (NB) algorithm after Mast et al.[1]. The fluid has bulk modulus $K = 2.2\,\mathrm{GPa}$, mass density $\rho = 1000\,\mathrm{kg/m^3}$ and viscosity $\mu = 0.001\,\mathrm{Pa{\cdot}s}$; the box has a width of 3 m; the injection velocity is 1 m/s; and the cell size is 0.1 m by 0.1 m.

2. Anti-locking strategy

The anti-locking strategy employed is based on the Hu-Washizu principle[7,8]. It is expressed as a set of weak form equations:

$$\int_\Omega \rho\delta\tilde{\bar{\boldsymbol{\sigma}}} : \left(\boldsymbol{d} - \tilde{\boldsymbol{d}}\right)\, dV = 0 \quad , \qquad \int_\Omega \delta\tilde{\boldsymbol{d}} : \rho\left(\bar{\boldsymbol{\sigma}} - \tilde{\bar{\boldsymbol{\sigma}}}\right)\, dV = 0 \tag{1}$$

and

$$\int_\Omega \delta\boldsymbol{v}\cdot\rho\boldsymbol{a}\, dV = -\int_\Omega \mathrm{grad}\,\delta\boldsymbol{v} : \rho\tilde{\bar{\boldsymbol{\sigma}}}\, dV + \int_\Omega \delta\boldsymbol{v}\cdot\rho\bar{\boldsymbol{b}}\, dV + \int_{\partial\Omega^\tau} \boldsymbol{\tau}^*\cdot\delta\boldsymbol{v}\, dS \tag{2}$$

where ρ is mass density, $\boldsymbol{v}$ is the velocity field, $\boldsymbol{a}$ is the acceleration field, $\boldsymbol{\tau}^*$ a prescribed surface traction, $\boldsymbol{d} = \frac{1}{2}(\mathrm{grad}\,\boldsymbol{v} + \mathrm{grad}^T\boldsymbol{v})$ is the rate of deformation tensor, $\bar{\boldsymbol{\sigma}} = \boldsymbol{\sigma}/\rho$ is the mass-specific stress, all defined in the physical

space, and $\delta\boldsymbol{v}$, $\delta\tilde{\boldsymbol{d}}$, $\delta\tilde{\boldsymbol{\sigma}}$ are weight functions defined on the background grid. The improved rate of deformation, $\tilde{\boldsymbol{d}}$, and the improved specific stress, $\tilde{\boldsymbol{\sigma}}$, are defined as piecewise constant functions in this study and represent the locking-free solution (one-point quadrature elements). The algorithm is implemented in an updated Lagrangian framework and is very similar to the $[\mathbb{P}_1]^3/\mathbb{P}_0$ interpolation discussed in [9] and thus shares concerns regarding spurious modes (checker-boarding) due to a violation of the LBB-condition [9,10]. Section 3 addresses the issue by introducing numeric dissipation through smoothing.

Following the framework of the cell-based algorithm[1], at any n-th time step Equation (2) yields modified nodal internal forces $\boldsymbol{f}_i^{int(n)}$:

$$\boldsymbol{f}_i^{int(n)} = -\sum_p m_p \tilde{\boldsymbol{\sigma}}_p^{(n)} \cdot \operatorname{grad} N_i(\boldsymbol{x}_p^{(n)}) \tag{3}$$

in which i is the node index, p is the particle index, m_p is the particle mass, $\boldsymbol{x}_p$ are the particle coordinates, and N_i represents the shape functions. Particles in any cell c are then updated with the improved fields $\tilde{\boldsymbol{d}}_c$ and $\tilde{\boldsymbol{\sigma}}_c$ at the end of the n-th time step using Equation (1):

$$\tilde{\boldsymbol{d}}_c^{(n+1)} = \frac{1}{m_c}\sum_{p\in\Omega_c} m_p \boldsymbol{d}_p^{(n+1)} \quad , \qquad \tilde{\boldsymbol{\sigma}}_c^{(n+1)} = \frac{1}{m_c}\sum_{p\in\Omega_c} m_p \bar{\boldsymbol{\sigma}}_p^{(n+1)} \tag{4}$$

in which Ω_c represents the domain of cell c and $m_c = \sum_{p\in\Omega_c} m_p$ the mass contained in that domain. The reader is referred to Mast et al.[1] and Yang[11] for a full discussion of these expressions.

3. Limited constant flux smoothing algorithm

MPM uses material points directly as integration points, which like any discretization introduces integration errors. Because the location of these integration points follow the material motions, they are not optimized for integration accuracy. The associated integration errors can result in nonphysical unbalanced nodal loads when the stress fields are discontinuous across cell boundaries. These spurious forces cause high frequency stress field checkerboarding and system destabilization, which is difficult to control with most existing stabilization strategies for one-point quadrature elements. In this paper, following the previous work in Mast et al.[1], we propose an improved smoothing approach based on a limited constant flux smoothing algorithm, separately applied on each state variable at each time step before particle updating. This reduces the effects of the integration errors by means of a diffusive mechanism. Due to space constraints, only the essential idea and implementation of the proposed smoothing algorithm are presented here. A complete derivation and discussion can be found in Yang[7].

3.1. Background theory

In the new proposed algorithm, smoothing is thought of as a conserved quantity transported between cells through their common interface. When this quantity is related to energy, the fundamental idea of the algorithm is to control energy increases to avoid instability, and at the same time control energy dissipation so variable fields will not be excessively smoothed. To achieve this goal, the discontinuous Galerkin method (DGM)[12] is applied for this pseudo transport problem.

DGM can be considered a hybrid method between the finite element method (FEM) and the control volume method (CVM). DGM uses smooth shape functions within cells and allows discontinuity at cell boundaries or surfaces. In each cell, DGM, just like FEM, uses Galerkin weak equations to evaluate rates of change of variables. However it introduces numerical traces or fluxes from CVM to solve interface conditions between cells. The numerical flux is the key for getting stable results using the method. More details about DGM and numerical fluxes can be found in [12–14].

3.2. Cell-wise averaging

In MPM, only incremental values on the grid are used to update state variables at material points to avoid diffusion. However, this can cause problems for the grid's ability to reproduce the variable fields on the particle domain when

particles move from one cell to another. Therefore, an additional cell-wise averaging procedure is included in the proposed algorithm. This procedure is identical to a smoothing effect implicitly included in the cell-based algorithm[1] with constant interpolants.

3.3. Localized smoothing procedure

In order to simplify the smoothing work and maximize the efficiency and flexibility of the algorithm, a smoothing procedure is converted into a series of decoupled one-dimensional problems. This is achieved by first splitting each single multi-dimensional smoothing problem into multiple partitions, each of which represents an independent multi-dimensional smoothing problem. Then the concept of DGM is used to apply fictitious boundary conditions such that the split smoothing problems are localized within pairs of adjacent cells as a series of one-dimensional cases. In the end, the smoothed fields of the split partitions are summed up as the final solution.

3.4. Controlled smoothing rocedure

In the proposed algorithm, the smoothing procedure is applied locally between pairs of adjacent cells as one-dimensional problems. Assume q^c represents a constant variable field over the domain of cell c, and $\hat{q}^c = q^c + \Delta q^c$ is the corresponding smoothed variable field. The employed algorithm computes the incremental values Δq^+ and Δq^- of two adjacent cells caused by a localized smoothing procedure as

$$\Delta q^+ = \gamma \frac{m^-}{m^- + m^+} \left(q^- - q^+\right) \qquad \text{and} \qquad \Delta q^- = \gamma \frac{m^+}{m^- + m^+} \left(q^+ - q^-\right) \,. \tag{5}$$

Here superscripts + and − are cell indices representing cells at positive and negative sides of the interface, respectively; $m^\pm$ give the total mass of cells at two sides of a given interface; and γ is a flux tuning coefficient relative to the flux limiters: $0 \le \gamma \le 1$. It is worth noting here that the use of superscripts ± differs from common usage in the literature, e.g. [12], in which + and − indicate limits approaching from positive and negative sides, respectively.

Equation (5) is derived to give the extreme energy dissipation and hence maximum numeric stability with $\gamma = 1$, and implies the smoothing algorithm can continue dissipating energy until the variable field approaches a global constant function (i.e., $q^+ = q^-$ at any interfaces) if γ is defined as any non-zero constant. This characteristic can easily ruin a simulation when a global constant function is not enough to describe the desired variable field. For instance, a linearly varying pressure field in a hydrostatic problem. Therefore, the concept of a flux limiter, also called a slope limiter [14], is used to tune down or turn off the fluxes if the variable field is considered reasonable. Two limiters are applied sequentially here.

3.4.1. Oscillation limiter

The first limiter blocks all fluxes between cells unless local high-frequency oscillation occurs, which is treated as nonphysical or spurious behavior in the analysis. For each cell c, and its neighbors l and r located at the negative and positive side of c, the condition indicating when a cell is undergoing high-frequency oscillation is given by

$$\left(q^r - q^c\right)\left(q^c - q^l\right) < 0 \,, \tag{6}$$

which means the slopes have different orientations. If and only if this condition is satisfied, the tuning coefficients γ for fluxes between cell c and its neighbors are evaluated with the linear limiter; otherwise, $\gamma = 0$.

3.4.2. Linear limiter

Piecewise constant interpolation introduces errors predicting variable values at cell edges. The linear limiter is designed to estimate these errors and adjust the constant fluxes accordingly.

For each cell c, and its neighbors l and r located at the negative and positive side of c, the slope of the variable field can be estimated as

$$S^c = \begin{cases} \left(q^r - q^l\right) / \left(x_o^r - x_o^l\right) \,, \text{ if both } l \text{ and } r \text{ exist;} \\ \left(q^c - q^l\right) / \left(x_o^c - x_o^l\right) \,, \text{ if only } l \text{ exists; and} \\ \left(q^r - q^c\right) / \left(x_o^r - x_o^c\right) \,, \text{ if only } r \text{ exists.} \end{cases} \tag{7}$$

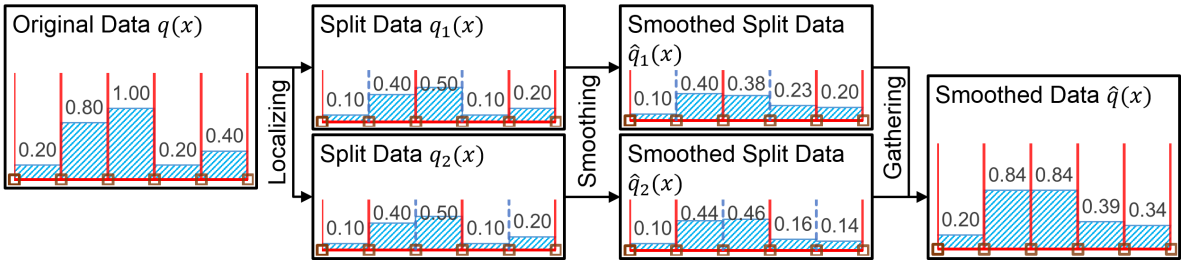

Fig. 2. 1D illustration of the limited constant flux smoothing algorithm. Consider the second cell from the left as reference cell. Then, split 1 interacts with the cell to its right, split 2 with the cell to its left.

x_o^r, x_o^c and x_o^l are mass centers of cell r, c and l, respectively. Then, the allowable error e_a^c for cell c at interface is

$$e_a^c = (x_m - x_o^c)S^c \tag{8}$$

in which x_m is an interface coordinate and is approximated by $x_m = (m^- x_o^+ + m^+ x_o^-)/(m^- + m^+)$. Eventually, the tuning coefficient can be evaluated as

$$\gamma = \min\left\{\max\left\{\frac{(q^- - q^+) + (e_a^- - e_a^+)}{(q^- - q^+)}, 0\right\}, 1\right\} . \tag{9}$$

3.5. Implementation

To better illustrate the algorithm, a one-dimensional example with a piecewise constant variable field (described with function $q(x)$) is given in Figure 2. The individual steps for the limited constant flux algorithm are

i. *Localizing/Splitting:* The purpose of splitting a smoothing problem into multiple partitions is to allow each cell to smooth with each of its neighbors, one at a time. To do so, the (minimum) number of partitions is equal to the (maximum) number of surfaces ($n = 6$), edges ($n = 4$) or ends ($n = 2$) of a three-, two- or one-dimensional cell, respectively. Then $q(x)$ is split equally into n partitions,

$$q_m(\boldsymbol{x}) = q(\boldsymbol{x})/n\,, \quad m = 1, \ldots, n\,, \tag{10}$$

i.e., $q(\boldsymbol{x}) = \sum_{m=1}^{n} q_m(\boldsymbol{x})$.
Example: The localization step in Figure 2 creates identical partitions (splits) of the original data.

ii. *Smoothing:* For purposes of implementation, to prevent $\Delta_q := q^- - q^+ = 0$ in Equation (9) and save computational cost, Equation (5) are rewritten as

$$\Delta q^+ = \gamma_o \frac{m^-}{m^- + m^+}\Delta_q^M \qquad \text{and} \qquad \Delta q^- = -\gamma_o \frac{m^+}{m^- + m^+}\Delta_q^M\,, \tag{11}$$

in which γ_o is a tuning coefficient related to the oscillation limiter only, and Δ_q^M includes the effects of the linear limiter:

$$\Delta_q^M = \text{sign}\,(\Delta_q)\min\left\{\max\left\{\text{sign}\,(\Delta_q)\left[\Delta_q + (e_a^- - e_a^+)\right], 0\right\}, |\Delta_q|\right\} \tag{12}$$

Here sign$(\cdot)$ is a function returning -1, 0 and 1 when the input value is negative, zero and positive, respectively.
Example: The smoothing step in Figure 2 assumes equal mass in every cell. The oscillation limiter blocks the flux between the left two cells in split 1.

iii. *Gathering:* In this last step, smoothed solutions from each split are gathered and combined back into a single smoothed solution

$$\hat{q}(x) = \sum_{m=1}^{n} \hat{q}_m(x)\,. \tag{13}$$

Example: The gathering step in Figure 2 recombines all (both) splits to the smoothed field, $\hat{q}(x)$. The diffusive character of smoothing is evident in both examples.

4. Validation example

A rectangular tank half filled with fluid at rest in stable equilibrium (i.e., the initial conditions should match the solution at any time, t) is used to study the performance of the new proposed smoothing algorithm. The simulation is simplified as a plane strain problem, in which the fluid has a bulk modulus $K = 2.2\,\text{GPa}$, mass density $\rho = 1000\,\text{kg/m}^3$ and viscosity $\mu = 0.001\,\text{Pa}\cdot\text{s}$. Acceleration due to gravity is $g = 10\,\text{m/s}^2$. The tank has a width of 3.2 m and a height of 6.4 m.

For all plane-strain models considered in this validation, the boundary condition for the tank's inner surfaces is

$$\boldsymbol{v}_i^{(n+1)} \cdot \boldsymbol{n}_i = 0, \text{ if } \boldsymbol{v}_i^{(n+1)} \cdot \boldsymbol{n}_i < 0 \tag{14}$$

and unconstrained otherwise. $\boldsymbol{v}$ is the velocity, $\boldsymbol{n}$ is the unit normal to the inner surface of the tank; a superscript $(n+1)$ indicates time t_{n+1}; and a subscript i represents node i on the grid. The rotation angle θ_g of the computational grid is selected as 0° (which is aligned with the rectangular tank) and 45° relative to the tank floor. $\theta_g = 45^\circ$ is used to intentionally introduce integration errors into the simulation so that capabilities of the smoothing algorithms in handling destabilizing sources derived from integration errors can be tested. To do so, an enhanced boundary treatment developed by Yang[11] is used to decouple the grid geometry from the boundary geometry. The numbers of particles per cell (ppc) and cells per length (cpl) are used to quantify any level of refinement. Equivalent ppc and cpl values obtained for models with $\theta = 0^\circ$ are used to indicate refinement levels for models with $\theta_g = 45^\circ$, in which grid lines are not parallel to the edges of the square water block and hence cells do not necessarily have the same number of particles. Simulation duration is taken as 3 seconds (simulated physical time), which allows the fluid block to have visible motion, if any, regardless of the cause of this non-physical behavior.

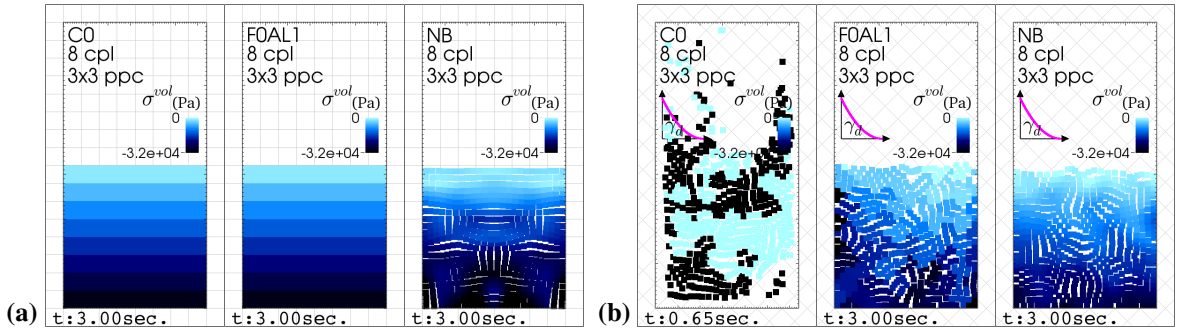

Fig. 3. Comparison of simulation snapshots between C0, F0AL1 and NB algorithms with grid orientation θ_g = (a) 0° and (b) 45°.

The first test is designed to show a strong dissipative characteristic of the node-based algorithm (NB)[1]. Grid orientation is selected as $\theta_g = 0^\circ$ to fit the geometry of the water tank, and 8 cells per length (of the square fluid block) and 3×3 particles per cell are used. Figure 3(a) compares the simulation results using the cell-based approach (C0), the limited constant flux algorithm (F0AL1), and the node-based approach (NB) at $t = 3\,\text{s}$. It shows both C0 and F0AL1 can keep the same stress field till the end of the simulations at 3 seconds. In contrast, NB over-smooths the stress field and changes the equilibrium status even though it generates the desired continuous stress field. This artificial change of the stress field causes global unbalanced downward forces and hence introduces a non-physical reduction of volume (height).

Next, the same setup as the first test is used except the grid is rotated by 45° counterclockwise around the left bottom corner of the fluid tank. Figure 3(b) shows the cell-based algorithm (C0) is not able to sufficiently control the destabilization caused by integration errors due to unfavorable placement of the particles for integration purposes, and hence the simulation fails, showing dramatic numeric instabilities. Consistent with its design, the limited constant flux algorithm (F0AL1) stabilizes the simulation through artificial diffusion and dissipation of energy, very similar to the node-based algorithm (NB). Both F0AL1 and NB algorithms suffer from non-physical loss of volume. However, with the flux limiter, F0AL1 significantly reduces the non-physical volume (height) changes as presented in the time history graphs of the equivalent height, h_{eq}, in Figure 4(a). Figure 4(b) shows that this non-physical behavior can be controlled by increasing the number of particle per cell (ppc), (i.e., by reducing the integration error). However, Figure 4(c)

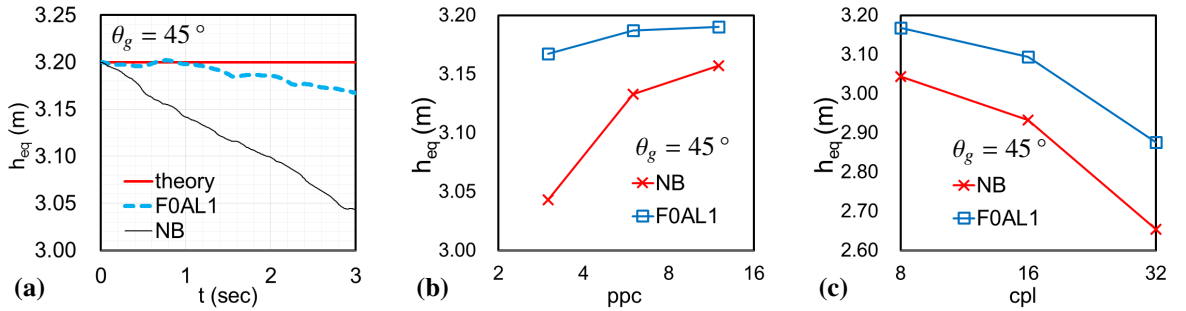

Fig. 4. Equivalent height h_{eq}: (a) time history (8 cpl, 3^2 ppc), (b) influence of ppc refinement (8 cpl); and (c) influence of cpl refinement (3 ppc).

demonstrated for this test problem that refining the grid at constant ppc increases the sources of destabilization, which triggers more energy dissipation for stabilization, and, consequently, exacerbates the non-physical behavior.

In summary, the MPM enhanced by the proposed smoothing (stabilizing) algorithm has been tested with a hydrostatic problem, water in a rectangular tank. The results show that (1) unlike C0 and F0AL1, NB includes a spurious mode and cannot sustain a static equilibrium state for water even if there is no integration error in the calculation; (2) the integration error induced by the MPM particle forms is a source of destabilization, which caused the C0 simulations to become unstable; (3) both F0AL1 and NB can handle the destabilization caused by the integration error through introducing algorithmic diffusion and thus dissipation; (4) F0AL1 offers a control mechanism for the algorithmic diffusion that NB does not possess; (5) increasing the number of particles per cell controls the integration error (i.e., the destabilizing source); and (6) grid refinement can also introduce destabilization due to motion of particles and cell crossing, and thus will not necessarily increase the accuracy of a simulation.

5. Application to dynamic fluid-solid interaction

A complex fluid-solid-interaction simulation is used to demonstrate the capability of the limited constant flux smoothing algorithm (F0AL1) to handle integration-error-induced destabilization. The problem consists of a floating square block, free to move in a tank as shown in Figure 5(a). The tank will be filled with fluid, forcing interaction with the tank's walls, as well as fluid-solid interaction with the block. Fluid is injected from the left at a constant height of 1.0 m at an initial velocity of $v_0 = 1.0\,\mathrm{m/s}$. The inflow of water continues for a duration of 6.0 s, after which the inflow ceases completely. The mass density of the fluid matches that of water: $\rho_w = 1000\,\mathrm{kg/m^3}$, while that of the solid is $\rho_s = 300\,\mathrm{kg/m^3}$. To decrease computational time by controlling the speed of sound in the fluid, a bulk modulus smaller than that of water is used, $K = 20\,\mathrm{MPa}$, and the viscosity is $\mu = 0.001\,\mathrm{Pa{\cdot}s}$. The solid block measures $0.4\,\mathrm{m} \times 0.4\,\mathrm{m}$ with elastic modulus $E_s = 1\,\mathrm{MPa}$ and Poisson's ratio $\nu_s = 0.25$. The tank has a width of 3 m and a height of 3 m on the left side, while offering infinite height on the right. The regular computational grid is made of square cells of $0.1\,\mathrm{m} \times 0.1\,\mathrm{m}$.

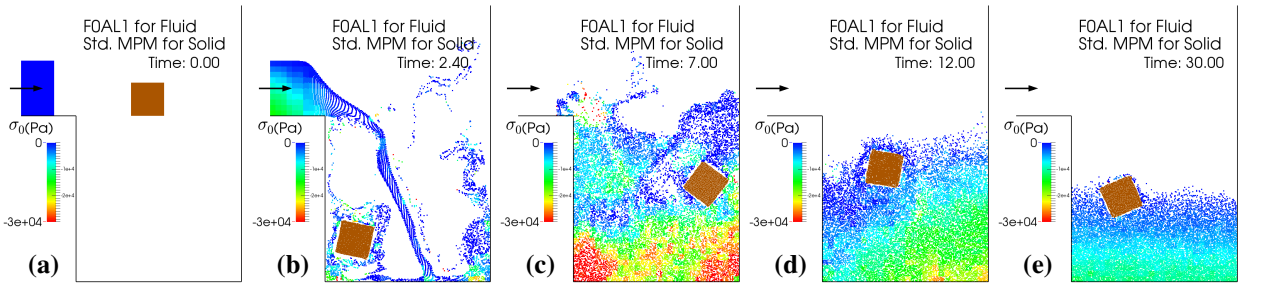

Fig. 5. Simulated means stress (negative pressure) distribution in the fluid using the limited constant numerical flux smoothing (F0AL1). (a) Initial conditions, (b)–(e) after 2.4 s, 7.0 s, 12.0 s, and 30.0 s, respectively.

The solid block is released at $t = 0$ s and falls under gravity, then is dragged to the left and lifted up from the bottom by the injected water as shown in Figure 5(b). The injection causes turbulence dragging the block left and right under water (Figure 5(c)) until the block floats up to the surface (Figure 5(d)) and stays afloat to the end of the simulation (Figure 5(e)).

Figure 5 also shows the computed mean stress (negative of pressure) for the same states. Throughout the simulation, free surface conditions ($p = 0$ MPa) are satisfied. The pressure field in the fluid is reasonably smooth, showing only minor artifacts from the underlying grid. Peak pressure corresponds close to the hydrostatic pressure of 30 kPa at $t = 7.0$ s, 20−25 kPa at $t = 12$ s, and 15 kPa at $t = 30$ s.

Figure 5(e) also illustrates the remaining issue of smoothing algorithms: the cell-based, node-based, and the limited constant flux algorithms all artificially dissipate strain energy over time when integration errors exist, and this results in compaction of the particles. The amount of fluid in the analysis should fill the tank to 2/3 of its height. This is approximately the case at $t = 12$ s, but is noticeable violated by $t = 30$ s. This side-effect is also visible for quasi static states after a large number of time steps (3×10^7 steps at $t = 30$ s). However, the task of eliminating this error remains open for future improvements.

Overall, these result demonstrate that the limited constant flux algorithm is able to stabilize a complex hydrodynamic analysis involving splashing fluid and complex fluid-solid interaction, and hence is capable of handling tsunami-debris-structure interactions, albeit with some non-physical loss of fluid volume.

6. Summary and conclusion

This article introduced a clarification regarding the concepts of anti-locking and smoothing algorithms for MPM applications, adding a distinction between these in regards to the previous work by Mast et al.[1]. This led to the introduction of a limited constant flux smoothing algorithm to combine the beneficial features of both the cell-based and node-based smoothing algorithms introduced in [1]. The newly introduced algorithm was shown to improve numeric stability of the MPM algorithm by mitigating the side-effects of numeric integration errors and cell crossings.

The validation example and an application to a coupled fluid-solid interaction problem demonstrate both improved stability and accuracy of the limited constant flux algorithm relative to the previous approaches. Further work remains to manage artificial volume creep that could be of significance for long duration simulations.

References

[1] C. M. Mast, P. Mackenzie-Helnwein, P. Arduino, G. R. Miller, W. Shin, Mitigating kinematic locking in the material point method, Journal of Computational Physics 231 (2012) 5351–5373.
[2] T. Hiraishi, K. Haruo, E. Saitoh, Experimental study on impulsive force of drift body due to tsunami flow, Journal of Earthquake and Tsunami 04 (2010) 127–133.
[3] H. Ko, D. Cox, H. Riggs, C. Naito, Hydraulic Experiments on Impact Forces from Tsunami-Driven Debris, Journal of Waterway, Port, Coastal, and Ocean Engineering (2014) 4014043.
[4] P. Piran Aghl, C. Naito, H. Riggs, Full-Scale Experimental Study of Impact Demands Resulting from High Mass, Low Velocity Debris, Journal of Structural Engineering 140 (2014) 4014006.
[5] D. Sulsky, S.-J. Zhou, H. L. Schreyer, Application of a particle-in-cell method to solid mechanics, Computer Physics Communications 87 (1995) 236–252.
[6] L. Lim, A. Andreykiv, R. Brinkgreve, Pile penetration simulation with material point method, in: M. Hicks, J. Dijkstra, M. Lloret-Cabot, M. Karstunen (Eds.), Installation Effects in Geotechnical Engineering, CRC Press, 2013, pp. 24–30. doi:10.1201/b13890-5.
[7] H.-C. Hu, On some variational principles in the theory of elasticity and the theory of plasticity, Acta Physica Sinica 10 (1954) 259–290.
[8] K. Washizu, Variational methods in elasticity and plasticity, 3 ed., Pergamon Press, Oxford, 1982.
[9] E. Süli, A brief excursion into the mathematical theory of mixed finite element methods, Lecture Notes, University of Oxford (2013) 24–29.
[10] K. J. Bathe, The inf – sup condition and its evaluation for mixed finite element methods, Computer and Structures 79 (2001) 243–252.
[11] W.-C. Yang, Study of Tsunami-Induced Fluid and Debris Load on Bridges using the Material Point Method, Ph.D. thesis, University of Washington, 2016.
[12] B. Cockburn, C.-W. Shu, Runge-Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems, Journal of Scientific Computing 16 (2001) 173–261.
[13] R. J. LeVeque, Numerical methods for conservation laws, Birkhauser Verlag, Basel, 1992.
[14] R. J. LeVeque, Finite volume methods for hyperbolic problems, Cambridge University Press, Cambridge, 2002.