# New Material Point Method Contact Algorithms for Improved Accuracy, Large-Deformation Problems, and Proper Null-Space Filtering

John A. Nairn[a,*], Chad C. Hammerquist[b], Grant D. Smith[c]

[a] *Wood Science and Engineering, Oregon State University, Corvallis, OR 97330, USA*
[b] *FracGeo, Woodlands, TX 77380, USA*
[c] *Wasatch Molecular, Salt Lake City, UT 84103, USA*

## Abstract

The material point method (MPM) is increasingly being used in multimaterial contact problems. Despite MPM's advantages for such problems, some aspects of current contact methods are prone to instabilities. The main causes of instabilities are errors in the methods used to calculate normal vector and material separation. This paper presents new methods for these tasks by applying logistic regression to material point clouds contributing to each contact node. The new logistic regression process determines the most probable plane between two materials and finds their separation. The separation calculation explicitly accounts for particle deformation for increased accuracy in large deformation problems. Another improvement to multimaterial MPM is to do contact mechanics corrections twice each time step — first after initial velocity extrapolation to the grid and second after updating grid momenta. Two corrections are needed to find the correct acceleration when updating particle velocities and to extend methods that filter null-space noise to account for contact mechanics. Several examples validate the new methods and demonstrate their advantages over prior methods.

*Keywords:* Material Point Method, Contact Mechanics, Null Space, Computational Mechanics

## 1. Introduction

One advantage of material point method (MPM) modeling [1–3] is its ability to handle dynamically evolving contact between objects [4–9]. Despite this inherent advantage, current multimaterial MPM methods for contact calculations are prone to instabilities. When modeling compaction of particulate composites to large deformation and shock waves through those composites, we noticed four issues in current methods:

1. Significant errors can develop at material interfaces under large compression. We traced these errors to inaccuracies in prior methods for calculating contact normal and material separation needed for accurate contact detection. Here we develop a new approach to these calculations based on machine learning techniques. We describe a logistic regression method that finds the plane separating two material objects in a simulation and finds their separation using distances of local material points to that plane. Importantly, the new methods are the first MPM contact methods to account for particle deformation, which is important in large deformation compaction.
2. By running contact simulations with perfect interfaces, we found that prior multimaterial MPM methods do not revert to single-material MPM. By examining tasks in each multimaterial MPM time step, we determined that each step must resolve contact conditions twice. Once to correct initial velocity extrapolations to the grid to satisfy contact conditions and then a second time to impose contact laws after updating grid momenta.
3. A recent MPM option enhances stability by removing null-space noise from particle velocities in a technique called XPIC($m$) [10]. When this method was applied in multimaterial MPM without regard to contact effects, artifacts were seen at material interfaces in shock physics examples. These artifacts

[*] Corresponding author: `john.nairn@oregonstate.edu`, Tel: +1-541-737-4265, Fax: +1-541-737-3385

were eliminated by revising XPIC($m$) calculations for multimaterial MPM to include a new term that is derived from contact corrections done after initial extrapolations to the grid.

4. In multi-object compaction, errors sometimes occur in contact regions involving three or more objects. These errors are caused by methods used to handle nodes that see three or more materials. We discuss an explicit solution to this problem, but show that in practice, explicit methods have larger errors than prior approximate methods. The recommended approach to handling three or more materials in contact therefore remains similar to prior methods.

This paper describes our solutions to the above issues. Section 2 reviews accepted equations for momenta changes needed to implement contact laws. Instabilities in multimaterial MPM are not caused by these simple equations, but rather by calculations of contact normal and material separation that are inputs to those equations. Section 2 then reviews prior methods for calculating these input terms from grid data. Section 3 describes a new logistic regression method that abandons grid data and instead finds contact normal and material separation using "point clouds" surrounding each contact node. Issues two and three were resolved by identifying new terms in multimaterial MPM that are needed to correctly update particle velocity and filter noise. Issue four is covered by discussing options for handling nodes that see three or more materials. Finally, section 4 provides examples to validate the new methods and to demonstrate their advantages over prior MPM methods.

## 2. Prior Multimaterial MPM Methods

### *2.1. Multimaterial MPM Extrapolations*

MPM uses a dual representation of modeled objects — a Lagrangian view (particles or material points) and an Eulerian view (grid) [1–3]. The particles carry information on the evolving simulation (*e.g.*, momentum, stress, deformation, *etc.*) while the grid is used to solve the momentum equation. In each time step, updated grid information (or Eulerian frame) is mapped to the particle space (or Lagrangian frame) using a matrix, $\mathbf{S}$, of "MPM shape functions." In other words, any vector of nodal quantities, $\boldsymbol{q}$, is mapped to particle quantities, $\boldsymbol{Q}$, using:

$$\boldsymbol{Q} = \mathbf{S}\boldsymbol{q} \tag{1}$$

Here lower case symbols indicate grid values while upper case indicate particle values. Also, subscripted vectors, such as $\boldsymbol{q}_i$ or $\boldsymbol{Q}_p$, indicate a nodal or particle value, while unsubscripted vectors indicate a vector of nodal or particle properties such as $\boldsymbol{q} = (\boldsymbol{q}_1, \boldsymbol{q}_2, \dots)$ or $\boldsymbol{Q} = (\boldsymbol{Q}_1, \boldsymbol{Q}_2, \dots)$. In generalized interpolation MPM (or GIMP), the MPM shape functions are found by convoluting grid shape functions, $N_i(\boldsymbol{x})$, with particle shape functions, $\chi_p(\boldsymbol{x})$ [3]. The elements of $\mathbf{S}$ become:

$$S_{pi} = \frac{\int \chi_p(\boldsymbol{x}) N_i(\boldsymbol{x}) d\boldsymbol{x}}{\int \chi_p(\boldsymbol{x}) d\boldsymbol{x}} \tag{2}$$

The $N_i(\boldsymbol{x})$ functions are typically linear shape functions, although quadratic (or higher order) spline shape functions are also used [11]. $\chi_p(\boldsymbol{x})$ is always 1 within the particle domain and zero elsewhere (*i.e*, $\int \chi_p(\boldsymbol{x}) d\boldsymbol{x} = V_p$ or particle volume).

The first step in MPM calculations is to extrapolate velocity (and other needed quantities) to the grid. Because the number of particles ($N$) and grid nodes ($n$) differ, the non-square $\mathbf{S}$ cannot be inverted. Instead, MPM extrapolates particle velocities using reverse mapping matrix $\mathbf{S}^+$ (first derived by least squares [1]):

$$\boldsymbol{v} = \mathbf{S}^+\boldsymbol{V} \quad \text{where} \quad \mathbf{S}^+ = \mathbf{m}^{-1}\mathbf{S}^T\mathbf{M} \tag{3}$$

where $\mathbf{m}$ is an $n \times n$ lumped mass matrix with nodal mass, $m_i$, on the diagonal, $\mathbf{m}^{-1}$ has $1/m_i$ on the diagonal provided $m_i \neq 0$, and $\mathbf{M} = \mathrm{diag}(\boldsymbol{M})$ is an $N \times N$ diagonal matrix formed from a vector of particle masses, $\boldsymbol{M}$. In code, this extrapolation is done by first extrapolating momenta ($\boldsymbol{p}$ and $\boldsymbol{P}$) and mass:

$$\boldsymbol{p} = \mathbf{S}^T\boldsymbol{P} \quad \text{and} \quad \mathbf{m} = \mathrm{diag}(\mathbf{S}^T\boldsymbol{M}) \quad \Longrightarrow \quad \boldsymbol{v} = \mathbf{S}^+\boldsymbol{V} = \mathbf{m}^{-1}\boldsymbol{p} \tag{4}$$

In multimaterial MPM, each material type, $\alpha$, extrapolates to its own grid velocity field by extrapolating only from particles of material type $\alpha$ [4]. These extrapolations can be written as

$$\boldsymbol{p}^{\alpha 0} = \mathbf{S}^{\alpha T}\boldsymbol{P} \quad \text{and} \quad \mathbf{m}^{\alpha} = \mathrm{diag}(\mathbf{S}^{\alpha T}\boldsymbol{M}) \quad \Longrightarrow \quad \boldsymbol{v}^{\alpha 0} = \mathbf{S}^{+\alpha}\boldsymbol{V} = (\mathbf{m}^{\alpha})^{-1}\boldsymbol{p}^{\alpha 0} \tag{5}$$

where $(\mathbf{m}^\alpha)^{-1}$ has $1/m_i^\alpha$ on the diagonal provided $m_i^\alpha \neq 0$ and

$$S_{pi}^\alpha = \begin{cases} S_{pi} & p \in \alpha \\ 0 & p \notin \alpha \end{cases} \quad \text{and} \quad \mathbf{S}^{+\alpha} = (\mathbf{m}^\alpha)^{-1} \mathbf{S}^{\alpha T} \mathbf{M} \tag{6}$$

Note that non-zero elements of $\mathbf{S}^\alpha$ are identical to non-zero elements of the full $\mathbf{S}$ matrix; the superscript $\alpha$ is used to indicate that shape functions for materials other then $\alpha$ are set to zero in $\mathbf{S}^\alpha$. This notation results in more compact matrix expressions. The superscript zero is added to $\boldsymbol{p}^{\alpha 0}$ and $\boldsymbol{v}^{\alpha 0}$ to indicate extrapolations that ignore contact conditions.

### 2.2. Basic Contact Equations

Particle velocities in multimaterial MPM should follow contact mechanics. But, when those velocities are extrapolated to the grid (using $\boldsymbol{v}^{\alpha 0} = \mathbf{S}^{+\alpha} \boldsymbol{V}$), the grid velocities may not satisfy contact laws. A key task in multimaterial MPM is to correct grid results by imposing momenta changes derived from contact mechanics. Assume that node $i$ receives contributions from only two materials ($\alpha$ and $\beta$) and that initial calculations found the normal vector for their contacting interface, $\hat{\boldsymbol{n}}$, and determined they are in contact. The basic contact equations are derived by changing initially extrapolated momenta by $\Delta \boldsymbol{p}_i^\alpha$ or:

$$\boldsymbol{p}_i^\alpha = \boldsymbol{p}_i^{\alpha 0} + \Delta \boldsymbol{p}_i^\alpha \quad \text{and} \quad \boldsymbol{p}_i^\beta = \boldsymbol{p}_i^{\beta 0} - \Delta \boldsymbol{p}_i^\alpha \tag{7}$$

Material $\beta$ changes by $-\Delta \boldsymbol{p}_i^\alpha$ to conserve momentum. The momentum change is found by solving:

$$\boldsymbol{v}_i^\beta - \boldsymbol{v}_i^\alpha = \frac{\boldsymbol{p}_i^\beta}{m_i^\beta} - \frac{\boldsymbol{p}_i^\alpha}{m_i^\alpha} = k\hat{\boldsymbol{t}} \quad \Longrightarrow \quad \Delta \boldsymbol{p}_i^\alpha = \Delta \boldsymbol{p}_i^\alpha(0) - m_i^{(red)} k \hat{\boldsymbol{t}} \tag{8}$$

where $\Delta \boldsymbol{p}_i^\alpha(0) = m_i^\alpha(\boldsymbol{v}_i^{(\mathrm{cm})} - \boldsymbol{v}_i^{\alpha 0})$ is the momentum change needed for materials to move with center of mass velocity ($\boldsymbol{v}_i^{(\mathrm{cm})}$), $\hat{\boldsymbol{t}}$ is unit vector perpendicular to $\hat{\boldsymbol{n}}$ chosen such that $\Delta \boldsymbol{p}_i^\alpha(0) \cdot \hat{\boldsymbol{t}} > 0$, and $m_i^{(red)} = m_i^\alpha m_i^\beta / (m_i^\alpha + m_i^\beta)$ is reduced mass. For arbitrary contact law, $k$ is found by equating contact traction to the tangential contact force implied by $\Delta \boldsymbol{p}_i^\alpha$ or

This section edited from published paper to clarify direction of $\hat{\boldsymbol{t}}$ and correct sign error in Eq. (10)

$$S_{slide}(N_c, \boldsymbol{v}_i^\beta - \boldsymbol{v}_i^\alpha, \dots) A_c = f_t^{(C)} = \frac{\Delta \boldsymbol{p}_i^\alpha \cdot \hat{\boldsymbol{t}}}{\Delta t} = \frac{\Delta \boldsymbol{p}_i^\alpha(0) \cdot \hat{\boldsymbol{t}} - m_i^{(red)} k}{\Delta t} \tag{9}$$

or

$$k = \frac{\Delta \boldsymbol{p}_i^\alpha(0) \cdot \hat{\boldsymbol{t}} - S_{slide}(N_c, \boldsymbol{v}_i^\beta - \boldsymbol{v}_i^\alpha, \dots) A_c \Delta t}{m_i^{(red)}} \tag{10}$$

Here $S_{slide}(N_c, \boldsymbol{v}_i^\beta - \boldsymbol{v}_i^\alpha, \dots)$ can be any contact law that gives tangential sliding force as a function of normal compression ($N_c$) and possibly other parameters (*e.g.*, sliding velocity, $\boldsymbol{v}_i^\beta - \boldsymbol{v}_i^\alpha$, or more: $\dots$). The contact compression is found from contact force needed to prevent interpenetration: $N_c = -\Delta \boldsymbol{p}_i^\alpha(0) \cdot \hat{\boldsymbol{n}} / (A_c \Delta t)$. Finally, $A_c$ is contact area and $\Delta t$ is the MPM time step.

The above contact equations are standard in MPM and available in various references [4–9]. Most implementations assume simple Coulomb friction (*i.e.*, $S_{slide} = \min(\mu N_c, \Delta \boldsymbol{p}_i^\alpha(0) \cdot \hat{\boldsymbol{t}} / (A_c \Delta t))$ for slip or stick situations), but the equation easily extends to arbitrary contact laws [9]. Despite consensus on contact calculations, multimaterial MPM often encounters difficulties. One challenge in all numerical modeling of contact is its inherent stiffness transitioning from zero prior to contact to effectively-infinite needed to prevent interpenetration. Sufficient spatial and temporal resolution (if available) can mitigate high stiffness issues. Our hypothesis is that remaining, MPM-specific difficulties are caused either by inaccuracy in the normal vector or by invalid determination of whether or not the two materials are in contact (*i.e.*, the two initial calculations assumed available at the start of this section).

*2.3. Prior Contact Normal and Contact Detection Methods*

Prior start-of-the-art, multimaterial MPM used grid methods to extrapolate volume gradient (or area gradient using particle area, $A_p$, in 2D and axisymmetric) and particle position using [8, 12]:

$$\boldsymbol{g}_i^\alpha = \begin{cases} \sum_{p\in\alpha} \boldsymbol{G}_{ip} V_p & \text{3D} \\ \sum_{p\in\alpha} \boldsymbol{G}_{ip} A_p & \text{2D/axisymmetric} \end{cases} \quad \text{and} \quad \boldsymbol{x}^\alpha = \mathbf{S}^{+\alpha}\boldsymbol{X} \tag{11}$$

where $\boldsymbol{x}$ and $\boldsymbol{X}$ are vectors of grid and particle positions and $\boldsymbol{G}_{ip}$ are GIMP gradient shape functions found by convolution with $\nabla N_i(\boldsymbol{x})$ [3]:

$$\boldsymbol{G}_{ip} = \frac{\int \chi_p(\boldsymbol{x})\nabla N_i(\boldsymbol{x}) d\boldsymbol{x}}{\int \chi_p(\boldsymbol{x}) d\boldsymbol{x}} \tag{12}$$

For a single material, $||\boldsymbol{g}_i^\alpha||\hat{\boldsymbol{n}}^\alpha = \boldsymbol{g}_i^\alpha$ where $\hat{\boldsymbol{n}}^\alpha$ is an outward-directed normal from material $\alpha$. Using information from both materials, the normal from material $\alpha$ to material $\beta$ is best found by averaging similarly-directed normal vectors for the two materials:

$$\hat{\boldsymbol{n}} = \frac{\boldsymbol{g}_i^\alpha - \boldsymbol{g}_i^\beta}{||\boldsymbol{g}_i^\alpha - \boldsymbol{g}_i^\beta||} \tag{13}$$

This "average gradient" method for finding normal vector is denoted as AG [8].

Given normal vector, early MPM contact assumed materials were in contact whenever

$$(\boldsymbol{v}^\beta - \boldsymbol{v}^\alpha)\cdot\hat{\boldsymbol{n}} < 0 \tag{14}$$

which is equivalent to detecting materials approaching each other [4]. But materials moving toward each other may or may not actually be touching. When they are touching, this condition and the definition of $N_c$ implies interfacial compression ($N_c > 0$). When they are not touching, however, the $N_c$ definition still gives $N_c > 0$, but materials that are not touching cannot be in compression. In other words, this condition is necessary, but not sufficient for detecting materials both in contact and in compression.

For accurate contact detection, one must determine if the materials are touching by calculating material separation [8]. One approach is to use extrapolated particle positions $\boldsymbol{x}_i^\alpha$. These extrapolations, however, are for particle centers and must be corrected to find edges of material domains. Figure 1A shows a 1D material $\alpha$ approaching node $i$ where $d = x_{edge}^\alpha - x_i$ is signed distance from the material's edge ($x_{edge}^\alpha$) to node $i$ (at $x_i$). For given shape functions and particle size, $d$ is a function of extrapolated $x_i^\alpha - x_i$ [13]. Figure 1B plots $d(x_i^\alpha - x_i)$ calculated for undeformed particles when using two particles per cell, GIMP or CPDI [14] shape functions, and linear $N_i(\boldsymbol{x})$ grid functions. The plots on the left are for material $\alpha$ while the plots on the right reflect that result for material $\beta$ approaching from the other direction. Material touching is detected when $d(x_i^\beta - x_i) < d(x_i^\alpha - x_i)$ [13]. To apply in 3D, the distance is found from position difference along the normal vector or touching occurs when

$$d\big((\boldsymbol{x}_i^\beta - \boldsymbol{x}_i)\cdot\hat{\boldsymbol{n}}\big) - d\big((\boldsymbol{x}_i^\alpha - \boldsymbol{x}_i)\cdot\hat{\boldsymbol{n}}\big) < 0 \tag{15}$$

The first MPM methods to use calculated separation advocated a simple offset correction that detected contact when

$$(\boldsymbol{x}_i^\beta - \boldsymbol{x}_i)\cdot\hat{\boldsymbol{n}} - (\boldsymbol{x}_i^\alpha - \boldsymbol{x}_i)\cdot\hat{\boldsymbol{n}} = (\boldsymbol{x}_i^\beta - \boldsymbol{x}_i^\alpha)\cdot\hat{\boldsymbol{n}} < \delta_c \tag{16}$$

where $\delta_c$ is a critical separation [7, 8]. This recommendation is equivalent to using linear fits $d(x) = ax \pm b$ and detecting contact when

$$(\boldsymbol{x}_i^\beta - \boldsymbol{x}_i^\alpha)\cdot\hat{\boldsymbol{n}} < \frac{2|b|}{a} \tag{17}$$

The linear fits in Fig. 1B give $\delta_c = 0.92$ or $0.96$ for GIMP or CPDI shape functions with two particle per linear cell dimension, respectively. Previous work found $\delta_c = 0.8$ to give better results with GIMP shape functions [7, 8]. This critical "contact position" method is denoted as the "CP = $\delta_c$" method. A useful improvement is to replace linear fits with higher-order fits to $d(x)$. For example, MPM nanoindentation

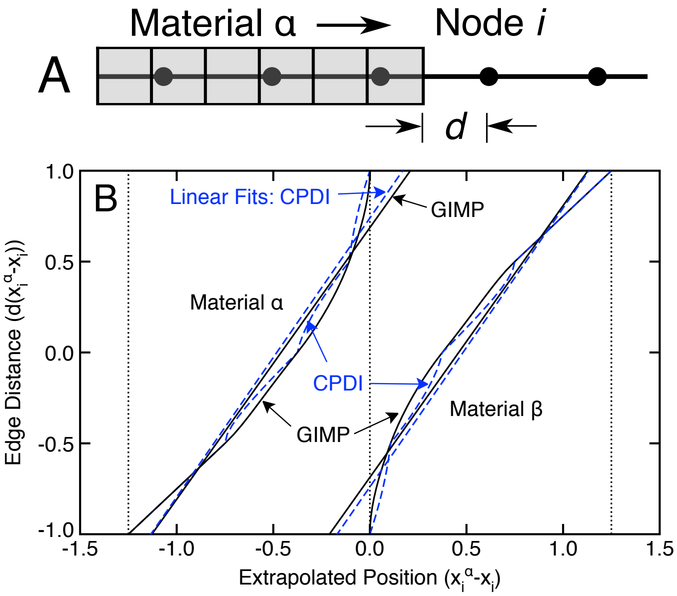

Figure 1: Actual distance from edge of a semi-infinite material domain (or $d(x_i^\alpha - x_i)$) to node $i$ as a function of distance of extrapolated particle centers to node $i$ (or $x_i^\alpha - x_i$) (in units of grid cells). The calculations are for undeformed particles using GIMP (solid curves) or CPDI (dashed curves) shape functions. The straight lines are linear fits to GIMP (solid lines) or CPDI (dashed lines) curves. The curves on the left are for material $\alpha$; the corresponding curves for material $\beta$ approaching from the right give $d(x_i^\beta - x_i)$ as a function of $x_i^\beta - x_i$ and are on the right.

simulations used a power law fit resulting in improved contact calculations [13]. This "$d(x)$ fitting" method is denoted as the DF method (fit from Ref. [13] was used here). Note that all separation calculations depend on normal vector. Thus methods to find normal vector and contact separation are intertwined.

Necessary and sufficient criteria for contact detection are when the materials are approaching each other (Eq. (14)) *and* when separation is less than zero (Eq. (15)). The first implies the interface is in compression while the second ensures the materials are actually touching. A simple validation example reveals that the velocity criterion alone (Eq. (14)) is poor while combining it with a separation criterion can be acceptable. Figure 2A shows two identical materials loaded in uniaxial compression. The materials were modeled using a neo-Hookean, hyperelastic material with strain energy function [15, 16]:

$$W = \frac{1}{2}G\left(\frac{I_1}{J^{2/3}} - 3\right) + \frac{1}{2}K\left(\frac{1}{2}(J^2 - 1) - \ln J\right) \tag{18}$$

where $K$ and $G$ are bulk and shear moduli, $J$ is determinant of deformation gradient $\mathbf{F}$, and $I_1$ is first invariant of $\mathbf{F}\mathbf{F}^T$. The simulations used $K = 2450$ MPa and $G = 940$ MPa. The top was pushed at constant velocity equal to 0.05% of the material's wave speed (to approach quasi-static loading) while the bottom was fixed. The material dimensions were $40 \times 100$ mm; the grid used $5 \times 5$ mm cells with four particles per cell in 2D plane-strain calculations with GIMP shape functions. The contact was modeled using Coulomb friction with friction coefficient $\mu = 0.2$. For this validation the materials started with a 7.5 mm (or 1.5 cells) gap.

Figure 2B plots the average axial stress until slightly after theoretical contact time. The first simulations situated material $\beta$ such that initial contact plane lies on a grid line (solid curves). Clearly the velocity criterion alone ($\Delta \boldsymbol{v} < 0$ curve) is unacceptable. Contact is always detected too early. The CP $= 0.92$ is better, but still too early. The empirically-derived CP $= 0.80$ improves the result further while the DF method is nearly exact. The second simulations situated material $\beta$ such that initial contact plane lies in the middle of a grid cell (dashed curves). Now, all grid-based methods falsely detect contact too soon. At first, the false detection only occurs at some nodes, which accounts for the slower change in stress after contact. The likely errors are in the normal vector. An error in normal vector shifts $(\boldsymbol{x}_i^\alpha - \boldsymbol{x}_i^\beta) \cdot \hat{\boldsymbol{n}}$ causing both CP $= \delta_c$ and DF methods to misinterpret the required separation for contact.

All grid-based methods are problematic. Even if more improvements could be made, they are still

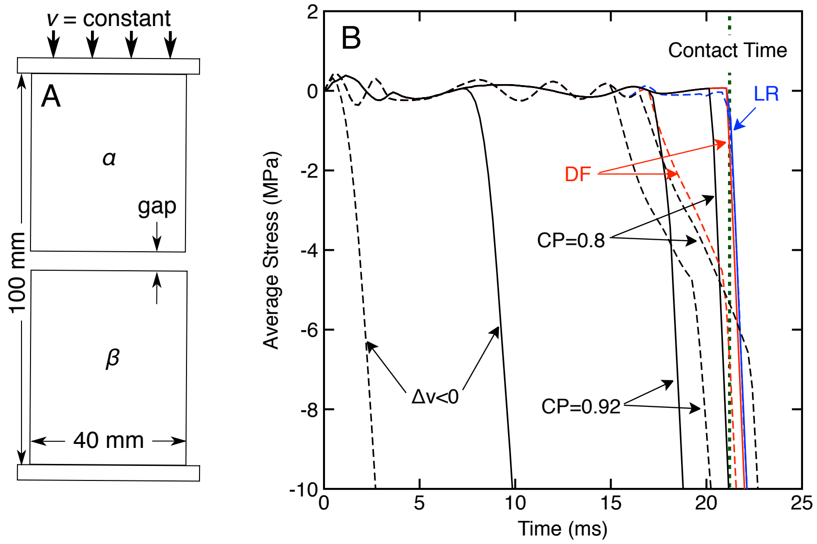

Figure 2: A. Two identical materials in axial compression. B. Average axial particle stress as a function of time until just after indicated theoretical "contact time" to close the initial gap for several contact detection methods (as labeled). The solid lines make contact on a grid line while dashed lines make contact in the middle of a grid cell.

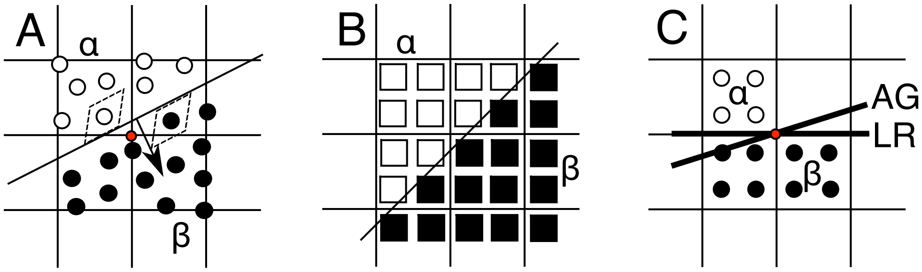

Figure 3: A. A material point cloud (showing point centers and sample domains for two points) that sees node in the middle with contact plane and normal vector. B. Material points at an angle to a contact plane showing corners of some material points crossing the contact plane. C. Calculation of contact plane using AG or LR methods near edge of material $\alpha$.

unsatisfying because they depend on shape functions and particle size. The next section derives a new approach to finding both normal and contact separation that does not rely on grid extrapolations and works for any shape functions or particle size. The new methods use logistic regression methods (denoted as LR method). Figure 2 shows that LR method gets essentially exact contact time for contact at grid lines and at mid cell locations (the solid and dashed LR curves mostly overlap).

## 3. Revised Multimaterial MPM Methods

### *3.1. Logistic Regression Methods (LR)*

The mass and momentum extrapolations identify the "material contact nodes," which are nodes that see two or more materials. The first task in non-grid methods (*e.g.* the LR method) is for each contact node to compile a list of nearby material points that define a "point cloud" around that node (see Fig. 3A). The goal is use point cloud data to find the best plane (or line in 2D) separating the materials. A vector normal to that plane is the contact normal. The material points closest to the plane define the material separation. Note that point clouds are compiled as needed and associated with grid nodes; this approach does not need to maintain nearest-neighbor lists for material points.

The contact problem in MPM is equivalent to a problem that arises artificial intelligence tasked to calculate the separation between two point clouds. One solution to that problem has been termed a support vector machine (SVM) [17]. The goal of an SVM is to find the most-probable line separating the two point clouds and then the "support vector" is defined as the distance between the two closest points normal to the separation line. One SVM solution method redefines it as a constrained optimization problem with number of variables equal to number of points [17] in the point cloud (∼16 in 2D or ∼64 in 3D MPM). An alternate approach to obtain nearly the same results as an SVM is to formulate the problem using logistic regression [18] that can be solved by non-linear least squares (NLLS) with four unknowns in 3D (or three in 2D) [17]. In brief, logistic regression is a statistical model that uses the sigmoidal logistic function to penalize points as a function of their distance from a preferred separation line rather than a simple linear regression function of distance. We found this probabilistic approach to give more reliable identification for the plane separating the two point clouds, especially near edges of the material point domains.

For 3D problems, define $\boldsymbol{x}_p = (X_{p,x}, X_{p,y}, X_{p,z}, 1)$ where $(X_{p,x}, X_{p,y}, X_{p,z})$ is the position of material point $p$. The equation for the desired separating plane is

$$\boldsymbol{x} \cdot \boldsymbol{\phi} = 0 \tag{19}$$

where $(\phi_1, \phi_2, \phi_3)$ is vector normal to the plane and $\phi_4$ is an offset. Consider an $N$-point cloud consisting of two materials of type $\alpha$ and $\beta$ and define $c_p = -1$ or 1 if point $p$ is material class $\alpha$ or $\beta$, respectively. To use logistic regression, we minimize the error

$$\Omega = \sum_{p=1}^{N} w_p (f(\boldsymbol{x}_p, \boldsymbol{\phi}) - c_p)^2 + \sum_{j=1}^{4} \lambda_j \phi_j^2 \quad \text{where} \quad f(\boldsymbol{x}, \boldsymbol{\phi}) = \frac{2}{1 + e^{-\boldsymbol{x}\cdot\boldsymbol{\phi}}} - 1 \tag{20}$$

is a logistic function (scaled for use of $c_p = -1$ or 1), $w_p$ are weighting factors, and $\lambda_j$ are penalty values that are used to improve convergence. The logistic function penalizes particles close to the plane resulting in calculation of the "most probable" plane separating the two material classes. NLLS starts with an initial guess $\boldsymbol{\phi}^{(0)}$ and finds updated $\boldsymbol{\phi}^{(k+1)}$ using:

$$\boldsymbol{\phi}^{(k+1)} = \boldsymbol{\phi}^{(k)} + \left(\mathbf{J}^T \mathbf{W} \mathbf{J} + \boldsymbol{\Lambda}\right)^{-1} \left(\mathbf{J}^T \mathbf{W} (\boldsymbol{c} - \boldsymbol{f}(\boldsymbol{\phi}^{(k)})) - \boldsymbol{\Lambda} \boldsymbol{\phi}^{(k)}\right) \tag{21}$$

where $\mathbf{J}$ is an $N \times 4$ Jacobian with elements $J_{pj} = \partial f(\boldsymbol{x}_p, \boldsymbol{\phi}^{(k)})/\partial \phi_j$, $\mathbf{W}$ is an $N \times N$ diagonal matrix with $w_p$ on the $p^{th}$ diagonal element, $\boldsymbol{\Lambda}$ is an $4 \times 4$ diagonal matrix with $\lambda_j$ penalty on the $j^{th}$ diagonal element, $\boldsymbol{c}$ is an $N$-vector of $c_p$ material classes, and $\boldsymbol{f}(\boldsymbol{\phi}^{(k)})$ is an $N$-vector of $f(\boldsymbol{x}_p, \boldsymbol{\phi}^{(k)})$ values. The NLLS equation is iterated until a convergence criterion is met. 2D problems use the same equations but eliminate $X_{p,z}$, $\phi_3$, and $\lambda_3$. Readers planning to implement these methods should refer to the appendix for more details including selection of weights ($w_p$), penalty values ($\lambda_j$), and convergence criteria. Readers evaluating results of this method can assume the details in the appendix provide an NLLS solution for the vector normal to the most-probable separation plane as $\boldsymbol{n} = (\phi_1, \phi_2, \phi_3)$.

Given the LR solution for the vector normal to the converged plane, $\boldsymbol{n}$, the contact unit normal is $\hat{\boldsymbol{n}} = \boldsymbol{n}/||\boldsymbol{n}||$. The separation between the materials is independent of $\phi_4$, but selecting $\phi_4 = -\boldsymbol{x}_i \cdot \boldsymbol{n}$ defines a contact plane that passes through node $i$. The signed distance of each material's edge to the plane through node $i$ is found from the particles of each material class closest to the plane:

$$d\big((\boldsymbol{x}_i^\alpha - \boldsymbol{x}_i) \cdot \hat{\boldsymbol{n}}\big) = \max_{p\in\alpha} (\boldsymbol{X}_p \cdot \hat{\boldsymbol{n}} + R_p) - \boldsymbol{x}_i \cdot \hat{\boldsymbol{n}} \tag{22}$$

$$d\big((\boldsymbol{x}_i^\beta - \boldsymbol{x}_i) \cdot \hat{\boldsymbol{n}}\big) = \min_{p\in\beta} (\boldsymbol{X}_p \cdot \hat{\boldsymbol{n}} - R_p) - \boldsymbol{x}_i \cdot \hat{\boldsymbol{n}} \tag{23}$$

Here $R_p$ is distance from particle $p$'s center to its deformed edge along $\hat{\boldsymbol{n}}$. Contact occurs by Eq. (15) when

$$\min_{p\in\beta} (\boldsymbol{X}_p \cdot \hat{\boldsymbol{n}} - R_p) - \max_{p\in\alpha} (\boldsymbol{X}_p \cdot \hat{\boldsymbol{n}} + R_p) = d_{LR} < 0 \tag{24}$$

Notice that the contact condition ($d_{LR} < 0$) is independent of $\boldsymbol{x}_i$, but distances to node $i$ are easily calculated if needed (*e.g.*, for penalty function methods in rigid contact based on distance to the node [19, 20]).

One advantage of LR methods is that $R_p$ allows them to account for particle deformation — an option unavailable in prior grid methods. If particles start as rectangular cuboids (rectangles in 2D), $R_p$ could be the distance from particle center to the edge of the deformed parallelepipeds. Figure 3B shows a situation where this approach would cause contact along angled planes to detect contact sooner than along planes parallel to grid lines. A slightly better approach, therefore, is to imagine an ellipsoid inscribed in the initial particle and set $R_p$ to the distance along $\hat{\boldsymbol{n}}$ from particle center to the deformed ellipsoid. This $R_p$ is given by affine transformation of the undeformed ellipsoid using particle $p$'s deformation gradient, $\mathbf{F}_p$:

$$\frac{1}{R_p} = \sqrt{\frac{n_{0,x}^2}{R_{0,x}^2} + \frac{n_{0,y}^2}{R_{0,y}^2} + \frac{n_{0,z}^2}{R_{0,z}^2}} \tag{25}$$

where $\boldsymbol{n}_0 = \mathbf{F}_p^{-1}\hat{\boldsymbol{n}}$ is normal vector transformed to the initial configuration ($\boldsymbol{n}_0$ is not a unit vector) and $\boldsymbol{R}_0$ is vector of undeformed particle radii. For cubical particles, this result simplifies to $R_p = R_{0,x}/||\boldsymbol{n}_0||$.

LR methods fix situations that confound grid methods. Figure 3C, for example, shows a node near a corner of material $\alpha$ but a full edge of material $\beta$. At the center node, the volume gradient from material $\alpha$ will find a normal at 45° while material $\beta$'s normal will be vertical. When averaged by Eq. (13), the separation line will be titled about 18°. The LR method in this situation converges to the preferred, horizontal plane. Despite the absence of material $\alpha$ in the upper-right grid cell, the LR solution remains horizontal. Prior methods often deteriorate near edges of material domains; this example suggests LR methods will do better.

Next, consider thin material domains near an edge or sandwiched between two other materials. A thin domain means less than two cells of material points beyond the contact node. Because the correction functions in Fig. 1 assume each material is a semi-infinite domain, they are wrong for thin domains [13]. Furthermore, extrapolated grid data are unable to detect or correct for thin domains. In contrast, LR methods, can find good normal and separation for any thickness material. Section 4 gives more examples that demonstrate LR improvements over grid methods.

### *3.2. Multimaterial MPM Time Step*

Particle to grid extrapolations in MPM conserve momentum, but when using multiple materials, they will not conserve contact physics. We assert that multimaterial MPM should correct initial velocity extrapolations for contact physics (justification for this correction is given below). We define a contact-corrected extrapolation as:

$$\boldsymbol{v}^\alpha = \boldsymbol{v}^{\alpha 0} + \Delta\boldsymbol{v}^\alpha = \mathbf{S}^{+\alpha}\boldsymbol{V} + \Delta\boldsymbol{v}^\alpha \tag{26}$$

where $\Delta\boldsymbol{v}^\alpha$ are the velocity changes found by contact calculations, which are related to contact momenta changes described above by $\Delta\boldsymbol{p}^\alpha = \mathbf{m}^\alpha\Delta\boldsymbol{v}^\alpha$. Compared to single material mode:

$$\boldsymbol{p} = \sum_\alpha \boldsymbol{p}^\alpha = \sum_\alpha \boldsymbol{p}^{\alpha 0} \quad \text{and} \quad \mathbf{m} = \sum_\alpha \mathbf{m}^\alpha \tag{27}$$

The equality to sums of both $\boldsymbol{p}^\alpha$ and $\boldsymbol{p}^{\alpha 0}$ assumes the contact calculations conserve momenta.

MPM codes differ on when they update particle stresses and stains [21, 22]. When updating them using initially-extrapolated velocities, the next step is to extrapolate grid velocities to find particle $p$'s velocity gradient:

$$\nabla\boldsymbol{V}_{p\in\alpha} = \sum_i \boldsymbol{v}_i^\alpha \otimes \boldsymbol{G}_{pi} \tag{28}$$

Because all multimaterial MPM codes should implement Eq. (26), these grid to particle extrapolations will correctly be using contact-corrected grid values that correspond to particle $p$'s material type.

Next, the nodal forces are found by extrapolating forces to the grid. Each material extrapolates its own force, $\boldsymbol{f}^{\alpha 0}$, as:

$$\boldsymbol{f}^{\alpha 0} = \boldsymbol{f}_{int}^\alpha + \boldsymbol{f}_{ext}^\alpha \tag{29}$$

where $\boldsymbol{f}_{int}^\alpha$ are internal forces due to Cauchy stress on $\alpha$ particle and $\boldsymbol{f}_{ext}^\alpha$ are external forces due to body forces and tractions on $\alpha$ particles. The reader is referred to other MPM papers for force extrapolations details (*e.g.*, Refs. [1–3]). The detail used here is that superscript "0" indicates forces that ignore contact; the final forces used to update particles will need to add contact forces.

The next multimaterial MPM task is when updating grid momenta. First, each material velocity field is updated to provisional momenta, $\boldsymbol{p}^{\alpha 0+}$, by using forces that ignore contact:

$$\boldsymbol{p}^{\alpha 0+} = \boldsymbol{p}^{\alpha} + \boldsymbol{f}^{\alpha 0}\Delta t = \boldsymbol{p}^{\alpha 0} + \Delta\boldsymbol{p}^{\alpha} + \boldsymbol{f}^{\alpha 0}\Delta t \tag{30}$$

Like initial extrapolations, the updated momenta must also be corrected for contact using:

$$\boldsymbol{p}^{\alpha +} = \boldsymbol{p}^{\alpha 0+} + \Delta\boldsymbol{p}^{\alpha +} = \boldsymbol{p}^{\alpha 0} + \Delta\boldsymbol{p}^{\alpha} + \Delta\boldsymbol{p}^{\alpha +} + \boldsymbol{f}^{\alpha 0}\Delta t = \boldsymbol{p}^{\alpha} + \Delta\boldsymbol{p}^{\alpha +} + \boldsymbol{f}^{\alpha 0}\Delta t \tag{31}$$

The final task is to update particle velocity. In multimaterial mode MPM, a standard FLIP update [23, 24] is

$$\boldsymbol{V}^{(n+1)} = \boldsymbol{V}^{(n)} + \mathbf{S}\boldsymbol{a}^{\alpha}\Delta t \quad \text{where} \quad \boldsymbol{a}^{\alpha} = (\mathbf{m}^{\alpha})^{-1}(\boldsymbol{f}^{\alpha 0} + \boldsymbol{f}^{\alpha C}) \tag{32}$$

is material $\alpha$'s acceleration found from extrapolated forces, $\boldsymbol{f}^{\alpha 0}$, and any forces on material $\alpha$ implied by contact calculations, $\boldsymbol{f}^{\alpha C}$. The question remains — how is $\boldsymbol{f}^{\alpha C}$ calculated? The two choices suggested by particle updates in Eq. (31) are:

$$\boldsymbol{f}^{\alpha C,0} = \frac{\Delta\boldsymbol{p}^{\alpha} + \Delta\boldsymbol{p}^{\alpha +}}{\Delta t} \quad \text{or} \quad \boldsymbol{f}^{\alpha C,1} = \frac{\Delta\boldsymbol{p}^{\alpha +}}{\Delta t} \tag{33}$$

$\boldsymbol{f}^{\alpha C,0}$ are contact forces that convert initially-extrapolated momenta, $\boldsymbol{p}^{\alpha 0}$, to final momenta, while $\boldsymbol{f}^{\alpha C,1}$ are contact forces that convert contact-corrected initial momenta, $\boldsymbol{p}^{\alpha}$, to final momenta. The correct option is the one that reverts to single material MPM when all interfaces are perfect. For perfect interfaces, $\Delta\boldsymbol{p}^{\alpha} = \mathbf{m}^{\alpha}\mathbf{m}^{-1}\boldsymbol{p} - \boldsymbol{p}^{\alpha 0}$ and $\Delta\boldsymbol{p}^{\alpha +} = \mathbf{m}^{\alpha}\mathbf{m}^{-1}\boldsymbol{p}^{+} - \boldsymbol{p}^{\alpha 0+}$; these result in each material's momentum matching the single-material momentum (*i.e.*, $\boldsymbol{p}^{\alpha} \to \boldsymbol{p}$ and $\boldsymbol{p}^{\alpha +} \to \boldsymbol{p}^{+}$). Substituting these perfect-interface results into material $\alpha$'s acceleration for each contact force option leads to two acceleration options:

$$\boldsymbol{a}^{\alpha,0} = \mathbf{m}^{-1}\boldsymbol{f} + \frac{\Delta\boldsymbol{v}^{\alpha}}{\Delta t} = \boldsymbol{a} + \frac{\Delta\boldsymbol{v}^{\alpha}}{\Delta t} \quad \text{or} \quad \boldsymbol{a}^{\alpha,1} = \mathbf{m}^{-1}\boldsymbol{f} = \boldsymbol{a} \tag{34}$$

Clearly, $\boldsymbol{a}^{\alpha,1}$ is preferred because when interfaces are perfect, material $\alpha$'s acceleration correctly reverts to the single-material mode acceleration $\boldsymbol{a}$. For a physical interpretation, we note that initial contact corrections given by $\Delta\boldsymbol{p}^{\alpha} = \mathbf{m}^{\alpha}\Delta\boldsymbol{v}^{\alpha}$ are corrections required to conserve contact physics on the grid. In other words, those corrections are part of the extrapolations and not part of the current time step's contact force. Instead, a simulation should use only $\Delta\boldsymbol{p}^{\alpha +}$ to find contact force ($\boldsymbol{f}^{\alpha C} = \boldsymbol{f}^{\alpha C,1}$) and acceleration ($\boldsymbol{a}^{\alpha}$). Note that the required $\Delta\boldsymbol{p}^{\alpha +}$ term is not correctly calculated unless initial extrapolations were corrected first with $\Delta\boldsymbol{p}^{\alpha}$. In other words, $\Delta\boldsymbol{p}^{\alpha}$ is not only needed for updating particle stresses and strains with initially grid extrapolations (*i.e.*, when using Eq. (28)), it is needed in all MPM time steps. Fortunately, when corrections are done twice each time step, the contact calculations work for any method chosen to update particle stresses and strains.

### *3.3. Multimaterial XPIC(m) Particle Updates*

Although FLIP updates are standard for MPM, they are prone to noise [10, 25, 26]. The numerical issue is that the number of particles is typically larger than the number of active nodes. This mismatch leads to a potentially large null space in particle to grid mappings [27]. Over time, null space modes on the particles can grow and degrade simulation results. An enhanced MPM update can be written as:

$$\boldsymbol{V}^{(n+1)} = \mathbf{P}\boldsymbol{V}^{(n)} + \mathbf{S}\boldsymbol{a}^{\alpha}\Delta t \tag{35}$$

where $\mathbf{P}$ is a projection operator that filters null-space noise from $\boldsymbol{V}^{(n)}$. As described in Ref. [10], one effective projection, called XPIC($m$) for eXtended Particle In Cell method of order $m$, can be written as

$$\mathbf{P}\boldsymbol{V}^{(n)} = \big(\mathbf{I} - (\mathbf{I} - \mathbf{S}\mathbf{S}^{+})^{m}\big)\boldsymbol{V}^{(n)} = \boldsymbol{V}^{(n)} - (\mathbf{I} - \mathbf{S}\mathbf{S}^{+})^{m-1}(\boldsymbol{V}^{(n)} - \mathbf{S}\boldsymbol{v}) \tag{36}$$

The resulting update was shown to filter null-space noise while minimizing unwanted dissipation [10].

The simplest extension of XPIC($m$) to multimaterial MPM would be to replace $\mathbf{S}^{+}$ with $\mathbf{S}^{+\alpha}$, which would also change $\boldsymbol{v}$ in Eq. (36) to $\boldsymbol{v}^{\alpha 0}$. This approach works well for many contact problems, but artifacts occur at interfaces in shock physics simulations. Because we showed initial extrapolations must be corrected for contact, we propose that multimaterial XPIC($m$) should substitute $\boldsymbol{v}^\alpha$ instead of $\boldsymbol{v}^{\alpha 0}$ into Eq. (36). Making this substitution and using $\mathbf{S}^{+\alpha}$ in place of $\mathbf{S}^+$, an XPIC($m$) update for multimaterial MPM becomes:

$$\mathbf{P}\boldsymbol{V}^{(n)} = \big(\mathbf{I} - (\mathbf{I} - \mathbf{S}\mathbf{S}^{+\alpha})^m\big)\boldsymbol{V}^{(n)} + (\mathbf{I} - \mathbf{S}\mathbf{S}^{+\alpha})^{m-1}\mathbf{S}\Delta\boldsymbol{v}^\alpha \tag{37}$$

The first term is prior XPIC($m$) applied individually to each material velocity field. The second term reflects changes needed for contact physics. After this revised velocity projection, particle positions update by methods in Ref. [10] (importantly, PIC and XPIC($m$) position updates differ from the FLIP position update). Notice that XPIC(1) reduces to:

$$\mathbf{P}\boldsymbol{V}^{(n)} = \mathbf{S}\big(\mathbf{S}^{+\alpha}\boldsymbol{V}^{(n)} + \Delta\boldsymbol{v}^\alpha\big) = \mathbf{S}\boldsymbol{v}^\alpha \quad \text{and} \quad \boldsymbol{V}^{(n+1)} = \mathbf{S}\boldsymbol{v}^{\alpha+} \qquad \text{for each } \alpha \tag{38}$$

which is a standard PIC update [28] that simply maps the updated grid result for each material to the particles. Like FLIP, the multimaterial PIC update correctly reverts to single material mode when all interfaces are perfect. PIC's problem is that it often dissipates too much energy. Higher order XPIC($m$) methods greatly reduce that unwanted dissipation.

A revised XPIC($m$) implementation needs minor changes to Ref. [10] methods as follows:

$$\begin{aligned}
\mathbf{P}\boldsymbol{V}^{(n)} &= m\mathbf{S}\mathbf{S}^{+\alpha}\boldsymbol{V}^{(n)} + \mathbf{S}\Delta\boldsymbol{v}^\alpha - \sum_{k=2}^{m}\binom{m}{k}(-\mathbf{S}\mathbf{S}^{+\alpha})^k\boldsymbol{V}^{(n)} + \sum_{k=1}^{m-1}\binom{m-1}{k}(-\mathbf{S}\mathbf{S}^{+\alpha})^k\mathbf{S}\Delta\boldsymbol{v}^\alpha && (39)\\
&= \mathbf{S}\left(m\mathbf{S}^{+\alpha}\boldsymbol{V}^{(n)} + \Delta\boldsymbol{v}^\alpha - \sum_{k=2}^{m}(-1)^k(\mathbf{S}^{+\alpha}\mathbf{S})^{k-1}\left(\binom{m}{k}\mathbf{S}^{+\alpha}\boldsymbol{V}^{(n)} + \binom{m-1}{k-1}\Delta\boldsymbol{v}^\alpha\right)\right) && (40)\\
&= m\mathbf{S}(\boldsymbol{v}^\alpha - \boldsymbol{v}^{*\alpha}) && (41)
\end{aligned}$$

where

$$\boldsymbol{v}^{*\alpha} = \sum_{k=2}^{m}(-1)^k\left(\boldsymbol{v}_k^{*\alpha} + \frac{m-1}{m}\Delta\boldsymbol{v}_{k-1}^{*\alpha}\right) \tag{42}$$

with

$$\begin{aligned}
\boldsymbol{v}_k^{*\alpha} &= \frac{1}{m}\binom{m}{k}(\mathbf{S}^{+\alpha}\mathbf{S})^{k-1}\boldsymbol{v}^\alpha = \frac{m+1-k}{k}\mathbf{S}^{+\alpha}\mathbf{S}\boldsymbol{v}_{k-1}^{*\alpha} && (43)\\
\Delta\boldsymbol{v}_k^{*\alpha} &= \frac{1}{m-1}\binom{m-1}{k}(\mathbf{S}^{+\alpha}\mathbf{S})^{k-1}\Delta\boldsymbol{v}^\alpha = \frac{m-k}{k}\mathbf{S}^{+\alpha}\mathbf{S}\Delta\boldsymbol{v}_{k-1}^{*\alpha} && (44)
\end{aligned}$$

The recursion relations start with $\boldsymbol{v}_1^{*\alpha} = \boldsymbol{v}^\alpha$ and $\Delta\boldsymbol{v}_1^{*\alpha} = \Delta\boldsymbol{v}^\alpha$. This derivation replaced $\mathbf{S}^{+\alpha}\boldsymbol{V}^{(n)}$ in Eq. (40) with $\boldsymbol{v}^\alpha - \Delta\boldsymbol{v}^\alpha$ and exploited fact that $\Delta\boldsymbol{v}_m^{*\alpha} = 0$. XPIC($m$) implementation calculates $\boldsymbol{v}^{*\alpha}$ in a separate task and uses that result in particle updates. The only difference is that multimaterial XPIC($m$) uses $\boldsymbol{v}^{*\alpha}$ in place of $\boldsymbol{v}^*$ in single material XPIC($m$) [10]. Calculation of $\boldsymbol{v}^{*\alpha}$ requires storing $\Delta\boldsymbol{v}^\alpha$ found in initial contact calculations for use in the XPIC($m$) task.

Although the above results extend XPIC($m$) to multimaterial MPM, the XPIC($m$) updates are not identical to single material MPM for perfect interfaces. For example, consider XPIC(2). In single material MPM, the velocity projection can be written as:

$$\mathbf{P}\boldsymbol{V}^{(n)} = (2\mathbf{I} - \mathbf{S}\mathbf{S}^+)\mathbf{S}\boldsymbol{v} = \mathbf{S}\left[\boldsymbol{v} + (\mathbf{I} - \mathbf{S}^+\mathbf{S})\boldsymbol{v}\right] \tag{45}$$

In multimaterial XPIC(2) with perfect interfaces (*i.e.*, $\boldsymbol{v}^\alpha = \boldsymbol{v}$ and $\Delta\boldsymbol{v}^\alpha = \boldsymbol{v} - \boldsymbol{v}^{\alpha 0}$) the velocity projection reduces to:

$$\mathbf{P}\boldsymbol{V}^{(n)} = (2\mathbf{I} - \mathbf{S}\mathbf{S}^{+\alpha})\mathbf{S}\boldsymbol{v} - \mathbf{S}\boldsymbol{v} + \mathbf{S}\boldsymbol{v}^{\alpha 0} = \mathbf{S}\left[\boldsymbol{v}^{\alpha 0} + (\mathbf{I} - \mathbf{S}^{+\alpha}\mathbf{S})\boldsymbol{v}\right] \tag{46}$$

Subtracting multimaterial from single material XPIC(2), the difference is:

$$\Delta\mathbf{P}\boldsymbol{V}^{(n)} = \mathbf{S}(\mathbf{S}^+ - \mathbf{S}^{+\alpha})(\boldsymbol{V}^{(n)} - \mathbf{S}\boldsymbol{v}) \tag{47}$$

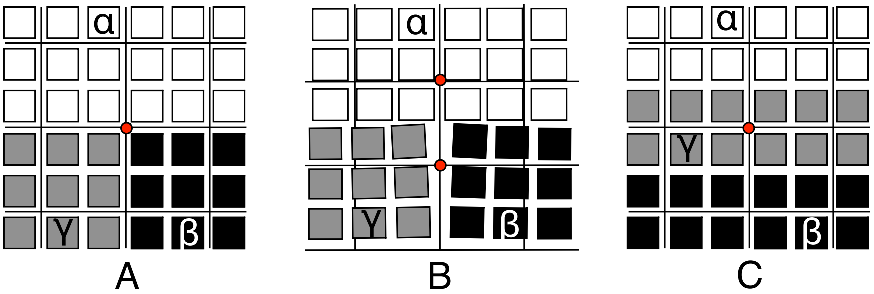

Figure 4: A. Three materials interacting with the indicated central node. B. Results of compression by 5% (half a cell in this problem) using the lumping method for the indicated 3+ nodes. C. Example of a 3+ node with a thin material sandwiched between two other materials.

Although not zero, this difference is related to a difference between simulation results expressed on the particles ($\boldsymbol{V}^{(n)}$) and those extrapolated from the grid ($\mathbf{S}\boldsymbol{v}$). In good simulations, this difference will be small. In brief, multimaterial XPIC($m$) filtering is slightly different than single material filtering. Overall, the benefits of enhanced stability out weight any small differences. Keeping the differences small, however, requires multimaterial-corrected XPIC($m$). If single-material XPIC($m$) is used instead, the error changes to:

$$\Delta \mathbf{P}\boldsymbol{V}^{(n)} = -\mathbf{S}(\mathbf{S}^{+} - \mathbf{S}^{+\alpha})\mathbf{S}\boldsymbol{v} \tag{48}$$

This error is larger than for multimaterial XPIC($m$) because it scales with total velocity rather than a difference between two similar velocities.

### *3.4. Three or More Materials in Contact*

In simulations with more than two material types, a single contact node may see three or more materials. These “3+” nodes may cause numerical problems. The number of 3+ nodes can be reduced by higher resolution, but no resolution can eliminate 3+ nodes where multiple materials meet at a point (see Fig. 4A). We tried three strategies for 3+ nodes. First, a node with $n$ materials has $n(n-1)$ contacting surfaces. In principal, LR methods can find normal and separation for each pair and then handle them separately using standard methods. Even if LR calculations remain accurate, however, momenta changes added to material $\alpha$ for contact with material $\beta$ will conflict with changes added for contact with material $\gamma$. The result is that handling contact pairs in isolation will no longer satisfy contact laws. This approach did not work well.

The second strategy was to simultaneously solve $n(n-1)$ contact law conditions. Starting with three materials and conserving momentum, the revised momenta are:

$$\boldsymbol{p}_i^{\alpha} = \boldsymbol{p}_i^{\alpha 0} + \Delta\boldsymbol{p}_i^{\alpha}, \quad \boldsymbol{p}_i^{\beta} = \boldsymbol{p}_i^{\beta 0} + \Delta\boldsymbol{p}_i^{\beta}, \quad \text{and} \quad \boldsymbol{p}_i^{\gamma} = \boldsymbol{p}_i^{\gamma 0} - \Delta\boldsymbol{p}_i^{\alpha} - \Delta\boldsymbol{p}_i^{\beta} \tag{49}$$

The momenta changes should be found by using Eq. (8) for each contact pair with $\hat{\boldsymbol{t}}$ for that pair's interface. But, the resulting three equations with only two unknowns cannot be solved to satisfy all contact laws while conserving momentum. If one or more of the pairs are not in contact, however, the remaining contact laws can be satisfied. For example, if material $\gamma$ is a thin domain between materials $\alpha$ and $\beta$ (Fig. 4C), $\gamma$ will be in contact with $\alpha$ and $\beta$, but $\alpha$ will not be in contact with $\beta$. The resulting two simultaneous equations can be solved explicitly.

A third option is to model each material $\alpha$ as if in contact with all other materials lumped into a virtual material $\beta$ [8, 9] where

$$\boldsymbol{p}_i^{\beta} = \sum_{k \notin \alpha} \boldsymbol{p}_i^{k0} \quad \text{and} \quad m_i^{\beta} = \sum_{k \notin \alpha} m_i^{k} \tag{50}$$

Each material is treated separately and the momentum change is added only to that material. If material pairs are using different contact laws, the calculations can use the law with the other material having the most volume. This approximate result will not exactly conserve momentum. It is, however, the simplest option.

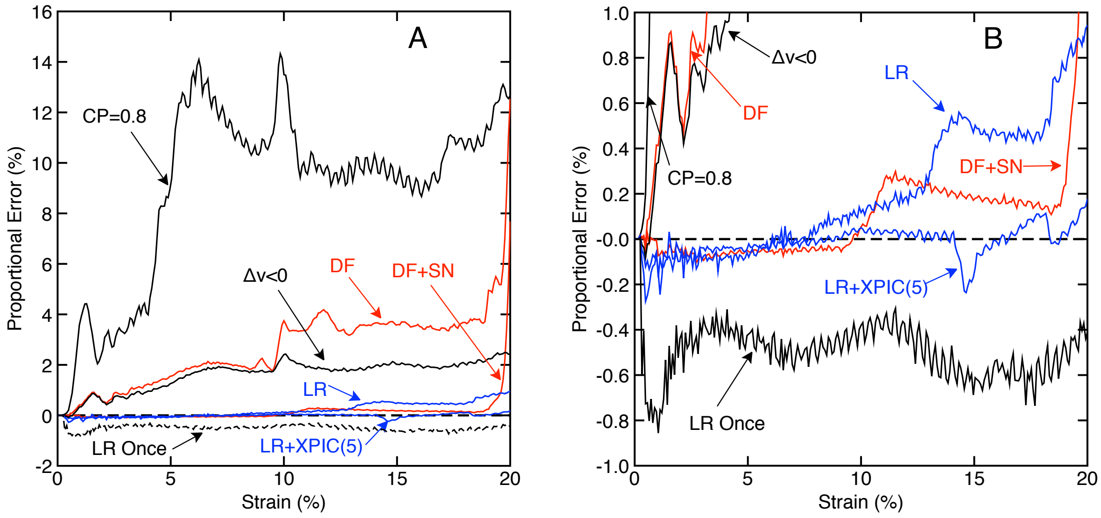

Figure 5: Proportional error ((MM-SMM)/SMM in %) in average axial stress for uniaxial compression of two identical materials as a function of compression strain using various methods for contacting normal and separation. A. All results. B. Zoomed in view of the results for error range ±1%.

Figure 4B shows results after axial compression of Fig. 4A to 5% strain using the lumping method. As expected the indicated contact nodes will get angled normals for materials $\beta$ and $\gamma$ causing the interface to rotate and separate near 3+ nodes. All other regions are handled well and global forces are close to single material MPM. To see if explicit modeling of 3+ nodes is better, we used the following algorithm. For any 3+ node, use LR methods to determine which material pairs are in contact. If only one pair or two pairs are in contact, calculate momenta change by single pair methods or by simultaneous solution of two contact laws. If three or more pairs are in contact, use the lumping method. This algorithm resolved all 3+ node issues for the simple example in Fig. 4A (figure not shown because identical to Fig. 4A but shifted half a cell). The discussion section compares lumping method to explicit handling of 3+ nodes in a more complex, real-world problem and reaches a different conclusion.

## 4. Results and Discussion

### *4.1. Validation Runs*

Contact method accuracy to high strain was evaluated by repeating the simulation in Fig. 2A up to 20% compression starting with the two materials initially in contact at a grid line. Because the two materials are identical and loading is axial, multimaterial MPM should get the same answer as single material MPM. Differences between single and multimaterial MPM reflect errors in the contact methods. Figure 5A plots those differences (as percent error) for various methods (Fig. 5B plots the same results zoomed in to ±1% error).

The "CP=0.8" method worked at small strain, but at about 0.7% strain the errors grew to about 4%. At about 5% strain the errors grew to 12% and remained high. Visually the material domains started to interpenetrate, especially at the edges. These errors are caused by inaccuracies in the AG method for normals and by the CP contact-detection criterion (Eq. (16)). To separate these two errors, we ran a simulation with velocity criterion alone (Eq. (14)). Although this criterion is necessary, but not sufficient, this problem is known to remain in contact making this condition both necessary and sufficient. Because the velocity criterion alone does not even calculate material separations, differences between "$\Delta v < 0$" errors and "CP=0.8" errors reflect errors associated with finding material separation using a constant offset and those errors were large.

Because of the large strain in this example, the interface moves about two cells (10 mm) thereby sampling all possible contact locations relative to grid lines. The constant offset method (CP=0.8) was shown above to vary in accuracy depending on location of the contacting plane. The DF method should reduce or eliminate those errors. This expectation is confirmed by the "DF" curve. Because "DF" errors were close to "$\Delta v < 0$" errors, the remaining limitation in grid-based methods must be calculation of the normal. To confirm this hypothesis, the DF method was run again with the normal fixed to $\hat{\boldsymbol{n}} = (0, -1, 0)$. By eliminating errors in normals, the resulting "DF+SN" (for "specified normal") was excellent. The errors were very small, although eventually increased after 18% strain.

The two methods to improve grid-based methods — use $\Delta v < 0$ alone or specify the normal — are unacceptable solutions. Using $\Delta v < 0$ alone is accurate when interfaces start in contact and remain in contact, but has resolution dependent errors on any separation. A specified normal only works for a single interface with constant orientation during a simulation. The logistic regression method seeks to improve on grid methods by replacing them with non-grid, point-cloud methods. The "LR" curve shows the method is stable and accurate to 20% strain. The errors increased slightly at the highest strain, but never exceeded 1%. Visual inspection revealed slight errors on the edges. These effects could be eliminated visually (and reduced numerically) by using XPIC($m$) noise reduction. The "LR-XPIC(5)" curve compares LR method using XPIC(5) to results from single material MPM using XPIC(5). The maximum error was 0.24%.

For efficiency, one is tempted to skip the contact corrections after initial extrapolations. Calculations show that the second calculation would then approximately find the sum $\Delta\boldsymbol{p}^{\alpha,tot} = \Delta\boldsymbol{p}^{\alpha} + \Delta\boldsymbol{p}^{\alpha+}$. But, this effort to maximize efficiency has three flaws. First, a single contact calculation cannot partition total momentum change into $\Delta\boldsymbol{p}^{\alpha}$ and $\Delta\boldsymbol{p}^{\alpha+}$. As a result, it would not match single material mode when interfaces are perfect and could not add the new term in multimaterial XPIC($m$). Second, when updating particle stresses before the momentum update, the particle velocity gradients would be inaccurate (*i.e.*, Eq. (28) would use an uncorrected velocity). Third, whenever contact laws are nonlinear (*e.g.*, any contact law besides frictionless), calculations of normal force, $N_c$, based on $\Delta\boldsymbol{p}^{\alpha,tot}$ would be using the wrong contact forces. To verify the need for two contact calculations in each time step, the LR simulation was repeated using only one. The "LR Once" curve shows the errors are larger. Additional calculations showed that $\Delta\boldsymbol{p}^{\alpha}$ and the consequences of avoiding its calculation, both increase at higher loading rates. In brief, general MPM codes should always use two contact calculations per time step.

### *4.2. Multi-Object Compression*

For a more complex contact problem, we compressed a box arbitrarily filled with elliptical objects (see Fig. 6A). The initial configuration was drawn in a 280×560 pixel bit mapped file and MPM particles were created by translating pixel values to material points. The pixel values were shaded to represent 20 different materials (albeit all with the same properties) to guarantee all contact would be between different material classes. All materials were hyperelastic (see Eq. (18)) with $K = 78090$ and $G = 27600$ MPa and density $\rho = 2.78$ g/cm$^3$ (*i.e.*, aluminum but without plasticity). The initial box was 1×2 mm and it was uniaxially compressed by 75% to $1 \times 0.5$ mm at 100 m/sec (about 2% of the material's wave speed). By rescaling the image, the problem was discretized into MPM simulations at various resolutions. By tracking average axial stress, the results for 280 and 560 material points along the vertical axis were the same indicating convergence. The simulations here used the the lower resolution (280) resulting in $0.0143 \times 0.0143$ mm cells (with four material points per cell). The calculations used CPDI shape functions [14] and multimaterial XPIC(5). The 3+ nodes where handled using the lumping method

Figures 6B and C show the final compressed state (enlarged compared to A) using the DF method (B) and the LR method (C). The interfaces in the DF simulation showed many material points of one ellipse penetrating into other ellipses. In contrast, the LR simulation had visually clean interfaces with no interpenetration. The interpenetration allowed by DF methods caused the global response to be about 20% softer than LR methods. The DF simulation reached 84 GPa. The LR results were smoother and reached 105 GPa. The CP=0.8 results (not shown) were similar to DF results but had slightly more interpenetration and softening (reached only 80 GPa or 25% softer).

The LR methods are computationally more intensive than DF or CP = 0.8 because they need a numerical solution at contact nodes (but can omit some extrapolations). For compaction of ellipses, the LR simulation took about 55% longer than DF or CP = 0.8 simulations. But this simulation is an exception — it includes contact at a large fraction of the active nodes. Most multimaterial MPM simulations will have a lower fraction of contact nodes and therefore have a lower computational cost for LR methods. The cost of avoiding LR methods would be less accurate results.

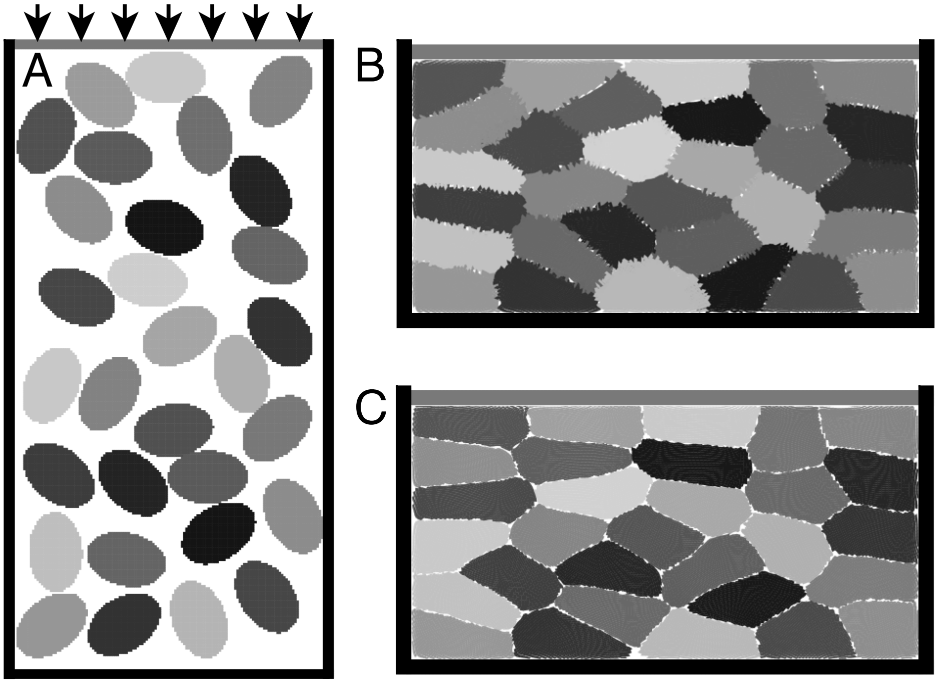

Figure 6: A. Model problem with arbitrarily placed elliptical objects placed in a 1 × 2 mm box and compressed from the top with a piston moving at 100 m/sec. B and C: Final particle shapes after compressing the box by 75% using DF method (B) or the LR method (C). Note that for clarity, B and C are enlarged compared to A. The vertical walls or fixed or distances between vertical, black walls is 1 mm in all drawings.

We repeated the LR simulations using explicit handling of 3+ nodes whenever possible. Despite success in simple problems, handling 3+ nodes in this real-world problem caused artifacts near locations where multiple ellipses met. In real-world problems, one or more of the materials at 3+ nodes may have only a few material points (or even just one). Because explicit 3+ node methods examine the contacting pairs separately, they likely lose accuracy when point clouds for any pair are too small and will be unreliable if each material has only one point. In contrast, the lumped method always uses the entire point cloud around each node. Our hypothesis is that inaccuracies in LR methods when some pairs have a small number of material points are larger than errors caused by approximations in the lumping method. Our current recommendation, therefore, is to use the lumping method to handle all 3+ nodes. Because explicit handling of 3+ nodes is not recommended, the involved equations are not provided. The subject should, however, be revisited if alternative methods are developed.

LR methods accurately find material separation, but does that mean $d_{LR} < 0$ alone is a necessary and sufficient condition for contact? We tried this "displacement-only" approach and the results for compacting elliptical particles were not good. The problem is that $d_{LR} < 0$ may occur when materials are moving apart and calculations of $N_c$ would imply interfacial tension. This situation is better handled by letting the materials move apart rather than modifying contact laws for an interface that is not in compression. In other words, either $(\boldsymbol{v}^\beta - \boldsymbol{v}^\alpha)\cdot\hat{\boldsymbol{n}} > 0$ or $d_{LR} > 0$ imply materials are *not* in contact, which is equivalent to requiring *both* $(\boldsymbol{v}^\beta - \boldsymbol{v}^\alpha)\cdot\hat{\boldsymbol{n}} < 0$ and $d_{LR} < 0$ to detect contact.

### *4.3. Shock Waves*

The need to calculate $\boldsymbol{\Delta v}^\alpha$ and use it to implement multimaterial XPIC($m$) is clearly demonstrated by simulation of a shock impact. We modeled a 50 × 2.5 mm nickel bar confined by walls on top, bottom and left and then impacted with an impact speed on the right of 1840 m/sec (40% of the material's bulk wave speed). The grid used 0.25 × 0.25 mm cells with four particles per cell. The nickel was modeled using the hyperelastic shear term in Eq. (18) but replaced the bulk modulus term with a Mie–Grüneisen equation of

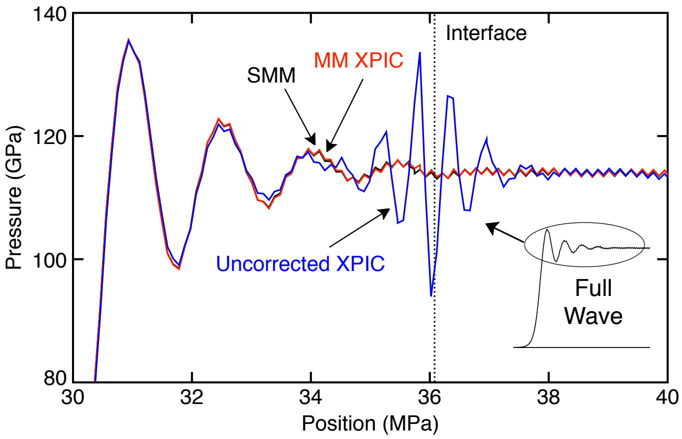

Figure 7: Zoomed in view of particle stresses at the edge of a pressure wave induced by shock impact on nickel (full wave shown in the inset). "SMM" and "MM XPIC" are single material mode and multimaterial mode XPIC($m$) derived here. The two curves mostly overlap. "Uncorrected XPIC" omits the new multimaterial correction in the XPIC($m$) task.

state [29] for pressure:

$$p = \begin{cases} K\frac{\eta\left(1-\frac{1}{2}\gamma_0\eta\right)}{(1-S_1\eta)^2} + \rho_0\gamma_0 U & \eta < 0 \\ K\eta & \eta \ge 0 \end{cases} \tag{51}$$

where $\eta = 1 - J$ is relative compression, $\gamma_0$ and $S_1$ are two material properties, and $U$ is internal energy per unit mass. For nickel, we used $K = 188$ GPa, $\gamma_0 = 2$, $S_1 = 1.44$, and $\rho_0 = 8.87$ g/cm$^3$ [29]. The material was also modeled using plasticity, but those properties had no effect in these 1D, confined shock conditions simulated here (and thus not listed). To complete the shock physics the modeling added a pressure $q$ when in compression described as "artificial viscosity" [29, 30] and given by:

$$q = \rho\Delta x|D_{kk}|\big(A_1C + A_2\Delta x|D_{kk}|\big) \tag{52}$$

where $\rho$ is current density, $|D_{kk}|$ is trace of the velocity gradient, $C$ is current bulk wave speed, and $\Delta x$ is cell size in the MPM grid. $A_1$ and $A_2$ are two parameters that control ringing in the shock front; we used (by trial an error) $A_1 = 0.4$ and $A_2 = 4.0$. For improved results, we used XPIC(5) and CPDI shape functions based on quadratic spline grid function (*i.e.*, replace linear $N_i(\boldsymbol{x})$ terms in CPDI shape function equations [14] with quadratic spline $N_i(\boldsymbol{x})$ terms [11]). The XPIC(5) method reduced noise while quadratic spline CPDI reduced particle-to-particle oscillations within grid cells.

First, we validated this shock modeling by showing that simulations as a function of impact speed in single material MPM very accurately reproduced the theoretical Hugoniot curves [31] for pressure, temperature, and shock wave speed as a function of impact speed. Next, we split the bar into two identical materials joined by a perfect interface and modeled in multimaterial MPM. Figure 7 magnifies the leading edge of the shock wave (moving to the left) soon after it has passed the material interface. The single material (SMM) and multimaterial (MM XPIC) results are virtually identical. As explained above, perfect-interface, multimaterial XPIC($m$) is not identical to single-material XPIC($m$). Here a very small difference is seen near the interface, but otherwise the two curves overlap well. In contrast, the "Uncorrected XPIC" curve that omits the $\Delta\boldsymbol{v}^\alpha$ term in the XPIC($m$) task causes large oscillations in particle values near the interface. This result confirms that multimaterial MPM should always calculate $\Delta\boldsymbol{v}^\alpha$ after initial extrapolations because it is needed for proper particle acceleration (see Eq. (33)) and to add a new term to XPIC($m$) calculations.

## 5. Conclusions

MPM is an excellent platform for modeling multimaterial problems with dynamically evolving contact situations. The revised methods derived above extend multimaterial MPM to handling such problems with greater accuracy, into regions of larger deformation, and with greater stability (by using corrected XPIC($m$)). While the basic contact calculations done to implement contact laws remain unchanged, prior MPM codes should make these changes to optimize multimaterial methods:

1. Use logistic regression to find contact normal and material separation. The two tasks required for this change are to build lists of material points interacting with each contact node and to implement NLLS solution to the logistic regression equations.
2. Implement contact corrections twice each time step — once immediately after extrapolating particle velocities to the grid to find $\Delta\boldsymbol{v}^\alpha$ and once after updating grid momenta to find $\Delta\boldsymbol{p}^{\alpha+}$. This latter term is needed to find the correct material $\alpha$ acceleration and its calculation depends on the initial extrapolations being corrected with $\Delta\boldsymbol{v}^\alpha$.
3. For codes that implement XPIC($m$) methods, use the $\Delta\boldsymbol{v}^\alpha$ calculated in the previous task to extend the filtering to multimaterial XPIC($m$). Code that use other methods to filter null-space noise would likely need new, but different, terms to account for contact.

### Acknowledgements

This work was funded by a Small Business Technology Transfer (STTR) contract #FA8651-17-C-0076 and a Small Business Innovation Research (SBIR) contract #FA8651-18-C-0061 both with Eglin Air Base.

### Appendix

The following gives implementation details for LR methods. The Jacobian can be written as $J_{ij} = \psi(k,\boldsymbol{x}_i)x_{i,j}$ where $k$ means as a function of $\boldsymbol{\phi}^{(k)}$, $x_{i,j}$ is $j^{th}$ component of $\boldsymbol{x}_i$, and

$$\psi(k,\boldsymbol{x}) = \frac{2e^{-\boldsymbol{x}\cdot\boldsymbol{\phi}^{(k)}}}{\left(1+e^{-\boldsymbol{x}\cdot\boldsymbol{\phi}^{(k)}}\right)^2} \tag{53}$$

We can write the $k$-$l$ element of $\mathbf{J}^T\mathbf{W}\mathbf{J}$ as

$$\left(\mathbf{J}^T\mathbf{W}\mathbf{J}\right)_{kl} = \sum_{p=1}^{N} J^T_{kp}w_pJ_{pl} = \sum_{p=1}^{N}\psi(k,\boldsymbol{x}_p)^2 w_p x_{p,k}x_{p,l} \tag{54}$$

Starting with 2D, the full matrix is:

$$\left(\mathbf{J}^T\mathbf{W}\mathbf{J}+\mathbf{\Lambda}\right) = \begin{bmatrix} \lambda_1 + \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,1}^2 & \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,1}x_{p,2} & \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,1} \\ \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,1}x_{p,2} & \lambda_2 + \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,2}^2 & \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,2} \\ \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,1} & \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p x_{p,2} & \lambda_3 + \sum_p \psi(k,\boldsymbol{x}_p)^2 w_p \end{bmatrix} \tag{55}$$

Extension to 3D just adds entries for $z$ components (by analogy). The vector term is:

$$\left(\mathbf{J}^T\mathbf{W}(\boldsymbol{c}-\boldsymbol{f}(\boldsymbol{\phi}^{(k)})) - \mathbf{\Lambda}\boldsymbol{\phi}^{(k)}\right)_j = -\lambda_j\phi_j^{(k)} + \sum_{p=1}^{N}\psi(k,\boldsymbol{x}_p)w_p\big(c_p - f(\boldsymbol{x}_p,\boldsymbol{\phi}^{(k)})\big)x_{p,j} \tag{56}$$

These results are substituted into Eq. (21) to complete the solution.

Note that if the logistic function is replaced by $f(\boldsymbol{x},\boldsymbol{\phi}) = \boldsymbol{x}\cdot\boldsymbol{\phi}$, the above analysis reduces to linear least squares fit to a plane through the point cloud that converges in one iteration. We first tried this linear regression method. It was better than prior contact methods, but still had errors in certain situations (*e.g.*, still tilted in Fig. 3C). Converged logistic regression methods are needed for best results. By selecting $\boldsymbol{\phi}^{(0)} = 0$, the first iteration can be shown equivalent to linear regression. In other words, the recommended method is to use linear regression to find $\boldsymbol{\phi}^{(1)}$ and then use NLLS with logistic regression until convergence to the “most probable” plane separating the two materials. We also note the the algorithm presented here assumes all contact information is thrown out at the end of each time step or the next step has to re-find all normals. If memory is available, it might be beneficial to store normal vectors on each time step and reuse them as initial guess for NLLS calculations on the next time step.

The NLLS method can weight material points by varying $w_p$. We tried several weighting schemes, but none improved on simply choosing $w_p = 1$. The penalty terms $(\lambda_j)$ can be adjusted to help convergence. After numerous trials, we recommend penalty values $\boldsymbol{\lambda} = 10^{-7}(\Delta x)^2(1, 1, 1, 0)$ where $\Delta x$ is grid cell size in the simulation. In other words, the components of the normal vector have penalty values while the offset does not.

The recommended convergence criterion is $1 - \hat{\boldsymbol{n}}^{(k+1)} \cdot \hat{\boldsymbol{n}}^{(k)} < \epsilon$ where $\hat{\boldsymbol{n}}^{(k)}$ is unit normal calculated from $\boldsymbol{\phi}^{(k)}$ and $\epsilon$ is a chosen tolerance. Note that:

$$1 - \hat{\boldsymbol{n}}^{(k+1)} \cdot \hat{\boldsymbol{n}}^{(k)} = 1 - \cos(\Delta\theta) \approx \frac{1}{2}(\Delta\theta)^2 \tag{57}$$

where $\Delta\theta$ is change in normal vector angle between steps $k$ and $k + 1$. In other words, convergence is based solely on normal vector direction. We used $\epsilon = 10^{-5}$, which translates to $\Delta\theta$ error 0.0044 radians. The NLLS iterations normally converged in a few steps, but some cases converged slowly. Slow convergence is often caused by a large mismatch between materials in the point cloud such as one material with a much fewer material points. Most slow-convergences cases still find acceptable normals. To guard against needless iterations, we limited the calculations to a maximum of 15 iterations (and used the final step results if the limit was reached).

## References

## References

[1] D. Sulsky, Z. Chen, H. L. Schreyer, A particle method for history-dependent materials, Comput. Methods Appl. Mech. Engrg. 118 (1994) 179–186.

[2] D. Sulsky, S.-J. Zhou, H. L. Schreyer, Application of a particle-in-cell method to solid mechanics, Comput. Phys. Commun. 87 (1995) 236–252.

[3] S. G. Bardenhagen, E. M. Kober, The generalized interpolation material point method, Computer Modeling in Engineering & Sciences 5 (2004) 477–496.

[4] S. G. Bardenhagen, J. U. Brackbill, D. Sulsky, The material point method for granular materials, Computer Methods in Applied Mechanics and Engineering 187 (2000) 529–541.

[5] S. G. Bardenhagen, J. E. Guilkey, K. M. Roessig, J. U. Brackbill, W. M. Witzel, J. C. Foster, An improved contact algorithm for the material point method and application to stress propagation in granular material, Computer Modeling in Engineering & Sciences 2 (2001) 509–522.

[6] X.-F. Pan, A.-G. Xu, G.-C. Zhang, P. Zhang, J.-S. Zhu, S. Ma, X. Zhang, Three-Dimensional Multi-mesh Material Point Method for Solving Collision Problems, Communications in Theoretical Physics 49 (2008) 1129–1138.

[7] V. Lemiale, A. Hurmane, J. A. Nairn, Material point method simulation of equal channel angular pressing involving large plastic strain and contact through sharp corners, Computer Modeling in Eng. & Sci. 70 (1) (2010) 41–66.

[8] J. A. Nairn, Modeling of imperfect interfaces in the material point method using multimaterial methods, Computer Modeling in Engineering and Sciences 92 (3) (2013) 271–299.

[9] J. A. Nairn, S. G. Bardenhagen, G. S. Smith, Generalized contact and improved frictional heating in the material point method, Computational Particle Mechanics 5 (3) (2018) 285–296. doi:10.1007/s40571-017-0168-1.

[10] C. C. Hammerquist, J. A. Nairn, A new method for material point method particle updates that reduces noise and enhances stability, Computer Methods in Applied Mechanics and Engineering 318 (2017) 724–738.

[11] M. Steffen, P. C. Wallstedt, J. E. Guilkey, R. Kirby, M. Berzins, Examination and analysis of implementation choices within the material point method (MPM), Computer Modeling in Engineering & Sciences 31 (2) (2008) 107–127.

[12] J. A. Nairn, J. E. Guilkey, Axisymmetric form of the generalized interpolation material point method, Int. J. for Numerical Methods in Engineering 101 (2015) 127–147.

[13] C. C. Hammerquist, J. A. Nairn, Modeling nanoindentation using the material point method, Journal of Materials Research 33 (2018) 1369–1381.

[14] A. Sadeghirad, R. M. Brannon, J. Burghardt, A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations, Int. J. Num. Meth. Engng. 86 (12) (2011) 1435–1456.

[15] R. W. Ogden, Non-Linear Elastic Deformations, Ellis-Harwood, New York, 1984.

[16] O. C. Zienkiewicz, R. L. Taylor, The Finite Element Methods for Solid and Structural Mechanics, Elsevier Butterworth-Heinemann, Oxford, UK, 2000.

[17] K. P. Murphy, Machine Learning: A Probabilistic Perspective, The MIT Press, Cambridge, MA, 2012.

[18] J. Zhang, R. Jin, Y. Yang, A. G. Hauptmann, Modified logistic regression: An approximation to SVM and its applications in large-scale text categorization, in: Proceedings of the Twentieth International Conference on Machine Learning, Washington DC, 2003.

[19] J. Ma, D. Wang, M. F. Randolph, A new contact algorithm in the material point method for geotechnical simulations, International Journal for Numerical and Analytical Methods in Geomechanics 38 (11) (2014) 1197–1210.

[20] 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).

[21] S. G. Bardenhagen, Energy conservation error in the material point method, J. Comp. Phys. 180 (2002) 383–403.

[22] J. A. Nairn, Material point method calculations with explicit cracks, Computer Modeling in Engineering and Sciences 4 (6) (2003) 649–664.

[23] J. U. Brackbill, H. M. Ruppel, FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions, Journal of Computational Physics 65 (2) (1986) 314 – 343.

[24] J. Brackbill, D. Kothe, H. Ruppel, FLIP: A low-dissipation, particle-in-cell method for fluid flow, Computer Physics Communications 48 (1) (1988) 25 – 38.

[25] C. Jiang, C. Schroeder, A. Selle, J. Teran, A. Stomakhin, The affine particle-in-cell method, ACM Trans ACM Trans Graph 34 (4) (2015) 51:1–51:10.

[26] C. Jiang, C. Schroeder, J. Teran, An angular momentum conserving affine-particle-in-cell method, Journal of Computational Physics in press.

[27] C. Gritton, M. Berzins, R. M. Kirby, Improving accuracy in particle methods using null spaces and filters, in: Proceedings of the 4th International Conference on Particle-Based Methods - Fundamentals and Applications, PARTICLES 2015, International Center for Numerical Methods in Engineering, 2015, pp. 202–213.

[28] F. Harlow, The particle in cell computing method for fluid dynamics, Methods in Computational Physics 3 (1964) 319.

[29] M. L. Wilkens, Computer Simulation of Dynamic Phenomena, Springer-Verlag, New York, 1999.

[30] J. Von Neumann, R. D. Richtmyer, A method for the numerical calculation of hydrodynamic shocks, J. Appl. Phys. 21 (1950) 232–237.

[31] C. A. Forest, Isoentrope energy, Hugoniot temperature, and the Mie-Gruneisen equation of state, AIP Conference Processinds 370 (1) (1996) 31–34.