 cmes, vol.5, no.6, pp.477-495, 2004

The Generalized Interpolation Material Point Method

S. G. Bardenhagen[1] [2] **and E. M. Kober**[3]

Abstract: The Material Point Method (MPM) discrete solution procedure for computational solid mechanics is generalized using a variational form and a Petrov–Galerkin discretization scheme, resulting in a family of methods named the Generalized Interpolation Material Point (GIMP) methods. The generalization permits identification with aspects of other point or node based discrete solution techniques which do not use a body–fixed grid, i.e. the "meshless methods". Similarities are noted and some practical advantages relative to some of these methods are identified. Examples are used to demonstrate and explain numerical artifact noise which can be expected in MPM calculations. This noise results in non-physical local variations at the material points, where constitutive response is evaluated. It is shown to destroy the explicit solution in one case, and seriously degrade it in another. History dependent, inelastic constitutive laws can be expected to evolve erroneously and report inaccurate stress states because of noisy input. The noise is due to the lack of smoothness of the interpolation functions, and occurs due to material points crossing computational grid boundaries. The next degree of smoothness available in the GIMP methods is shown to be capable of eliminating cell crossing noise.

keyword: MPM, PIC, meshless methods, Petrov–Galerkin discretization.

1 Introduction

The past several decades have brought tremendous advances in computing power and provided fertile ground for the development of the computational sciences. In computational solid mechanics the Finite Element Method (FEM) (see, e.g. [Johnson (1987)]) has been very successfully applied to a wide range of problems with good results. However, body fixed FEM meshes can be difficult and time consuming to generate for complex three–dimensional objects. Further, mesh distortion associated with large deformations compromises solution accuracy, ultimately requiring re–meshing. These difficulties have spurred the development of alternate discretization strategies which avoid mesh distortion by discretizing at points and never maintaining a body–fixed mesh.

Quite a number of "meshless methods" have been developed. Some of the ways in which the methods differ include whether or not a temporary mesh is used in the solution procedure, whether the discretization procedure begins with the differential equations or a weak form, and in the construction and support of the point weighting functions. In a recent review article, [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996)], similarities between Smooth Particle Hydrodynamics, Diffuse Element, Element Free Galerkin, and Reproducing Kernel Particle methods are discussed, and it is found that for certain cases all may considered particular examples of the more general Partition of Unity Method, [Babuška and Mellenk (1997)]. The Partition of Unity method uses a variational form to specialize the discrete approximation in regions of the problem domain using a standard Galerkin discretization scheme. This is in contrast to the development, from a variational form but using a Petrov–Galerkin discretization scheme, of methods such as Adaptive Characteristic Petrov–Galerkin Finite Element, [Demkowicz and Oden (1986)], and Meshless Local Petrov–Galerkin methods, [Atluri and Zhu (2000)]. While there are still more meshless methods, these are singled out because they are derivable from a weak form and therefore share features with the Generalized Interpolation Material Point Method described here.

The Material Point Method (MPM) is one of the latest de-

[1] Dept. of Mechanical Engineering, University of Utah, Salt Lake City, UT 84112, USA, Graduate Engineering Research Center, University of Florida, 1350 Poquito Rd., Shalimar, FL 32579, USA.

[2] Correspondence address: Group T–14, MS B214, Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA.

[3] Group T–14, MS B214, Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA.

velopments in particle–in–cell (PIC) methods. PIC methods were originally used in computational fluid mechanics to model highly distorted fluid flow, [Harlow (1963)]. Subsequent developments advanced the understanding of the algorithm and brought modifications to reduce numerical diffusion in the FLIP algorithm, [Brackbill, Kothe, and Ruppel (1988); Burgess, Sulsky, and Brackbill (1992)]. Fundamental aspects of PIC methods include the interpolation of information between a grid and particles, and precisely which solution variables are ascribed to the grid, and which to the particles. The general trend has been toward keeping more properties on particles. This trend has been continued in the development of MPM, where the ability of the particles, or "material points", to advect naturally Lagrangian constitutive response state variables, has been exploited in application to computational solid mechanics, [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)]. In MPM the grid may be viewed as a temporary computational scratch pad, as the material points carry the complete solution.

Particle methods are well suited for solid mechanics where it is natural to have a reference state and properties which are a function of location in the reference state. Material response is governed by continuum mechanics constitutive models which generate stress based on both the history and current mechanical state. These models are often complex and require the calculation of "internal variables" representing the (history dependent) material state. Lagrangian particles allow easy implementation of these constitutive models, and straight–forward advection of internal variables through the computational grid. MPM has found application in the solution of a wide variety of problems in solid mechanics, including mantle convection, [Lenardic, Moresi, and Mühlhaus (2000)], silo discharge, [Więckowski, Youn, and Yeon (1999)], membrane stretching, [York, Sulsky, and Schreyer (1999)], landfill settlement, [Zhou, Stormont, and Chen (1999)], elastic vibrations, [Sulsky, Chen, and Schreyer (1994)], collisions, [Bardenhagen, Harstad, Maudlin, Gray, and Foster (1998); Sulsky, Chen, and Schreyer (1994); Sulsky and Schreyer (1996); Sulsky, Zhou, and Schreyer (1995)], and the response of granular material, [Bardenhagen and Brackbill (1998); Bardenhagen, Brackbill, and Sulsky (2000b,a); Bardenhagen, Guilkey, Roessig, Brackbill, Witzel, and Foster (2001)].

The derivation of the MPM algorithm has recently been cast in variational, or weak form, [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)], providing a standard setting for the discretization of the governing equations, [Johnson (1987)]. If drawing attention to the similarities between FEM and MPM by deriving MPM from a weak form enhances communication between research communities, it provides a valuable contribution. In addition, however, this setting provides a venue for generalizing the MPM discretization technique, which has not been taken advantage of. PIC methods were developed by considering particles to provide an alternate representation of solution variables on the grid. The particle representation was used to advect these variables through the grid, avoiding (in particular) difficulties associated with interface tracking. In order that grid and particle solutions have the appropriate correspondence, the nature of transferring information between grid and particles had to be carefully attended to. For example, both the MPM algorithm and its predecessor FLIP, [Brackbill and Ruppel (1986)], conserve total mass and momentum in interpolating from particles to grid and back again. In fact, essentially the same governing equations presented in [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)] may be derived, without reference to a variational form, simply by considering conservation of momentum on the computational grid, and conservation of mass and momentum in the interpolation between grid and particle representations of the current solution, [Brackbill and Ruppel (1986)].

Here the full generality of the variational formulation is exploited. The variational form of the governing equations provides a consistent framework for generalizing the MPM discretization technique, and similarities to other meshless methods, [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996); Babuška and Mellenk (1997); Demkowicz and Oden (1986); Atluri and Zhu (2000)], may be identified. The use of smoother representations of discrete material point data allows an entire family of methods to be developed. A significant result of this generalization is smoother representation of particle data on the computational grid. This removes a numerical artifact inherent in the MPM formulation, which can develop when material points fail to register in a self–similar fashion on the computational grid. This situation can be expected to arise regularly in fi-

nite deformation analyses, and is demonstrated to very seriously degrade the accuracy of solutions obtained using MPM. The nature of the general derivation suggests denoting the family of Material Point Method discretization schemes developed here the Generalized Interpolation Material Point (GIMP) methods. It is hoped that the generalization is more deserving of its acronym's formal definition (an attractive trim) than its slang.

In the first section the GIMP methods are derived from a variational form using a Petrov–Galerkin discretization scheme. The specialization to the MPM algorithm is shown. In the second section a GIMP algorithm is developed in which the interpolation functions are in C^1 (as opposed to MPM, for which they are in C^0). Properties of this version, denoted the contiguous particle GIMP method, and more general "fuzzy particle" discretizations, are discussed. Next the performance of MPM and contiguous particle GIMP methods are compared by considering the quasi–static compression of a continuum column, and stress wave propagation in the same bar. A convergence study is reported. Finally, conclusions are drawn.

2 Derivation of the Discrete Equations

In the following derivation of the discrete equations, bold face quantities indicate tensors, ∇ is the gradient operator, and $\cdot$ and $:$ are first order (vector) and second order tensor contractions, respectively. The subscript p is used to index material point variables, and v grid vertex variables. The notation $\sum_p$ and $\sum_v$ is used to denote summation over all material points, and over all grid vertices, respectively.

Of interest in solid mechanics is the deformation and material response of a continuous solid body under prescribed loads and initial conditions, as governed by conservation of mass and momentum. Conservation of mass is satisfied implicitly by leaving discrete particle masses unchanged throughout a computation. Here we develop the discrete version of conservation of momentum, which permits evolution of particle momenta in time. We consider a deformable body acted upon by body forces and subjected to either kinematic or traction boundary conditions everywhere on its surface. The variational form for conservation of momentum may be written

$$\int_\Omega \rho\mathbf{a}\cdot\delta\mathbf{v}\,d\mathbf{x} + \int_\Omega \boldsymbol{\sigma}:\nabla\delta\mathbf{v}\,d\mathbf{x} = \int_\Omega \rho\mathbf{b}\cdot\delta\mathbf{v}\,d\mathbf{x} + \int_{\partial\Omega_{\boldsymbol{\tau}}} \boldsymbol{\tau}\cdot\delta\mathbf{v}\,dS. \tag{1}$$

Here ρ is the current mass density, $\mathbf{a}$ is the acceleration, $\boldsymbol{\sigma}$ is the Cauchy stress, $\mathbf{b}$ is the specific body force, $\boldsymbol{\tau}$ is the boundary traction, and $\delta\mathbf{v}$ is an admissible velocity field. The entire current volume is denoted by Ω with boundary, $\partial\Omega$, which is the union of that part of the boundary on which tractions are prescribed, $\partial\Omega_{\boldsymbol{\tau}}$, and that part on which velocities are prescribed, $\partial\Omega_{\mathbf{v}}$.

The essence of the discretization procedure is to represent a solid material continuum as a collection of body fixed (Lagrangian) particles, or "material points". The terms particle and material point will be used interchangeably throughout this manuscript. Particles are defined by "particle characteristic functions", $\chi_p(\mathbf{x})$. In practice the particle characteristic functions are non zero over a small volume. They define the space occupied, perhaps only partially, by a given particle, and can be thought of as the spatially varying volume fraction of that particle. They are functions of current particle position and, most generally, deformation state.

2.1 Initial Discretization

The particle characteristic functions are required to be a partition of unity in the initial configuration, i.e.

$$\sum_p \chi_p^i(\mathbf{x}) = 1 \quad \forall \quad \mathbf{x}. \tag{2}$$

i means initial

where χ_p^i denotes the particle characteristic functions restricted to their initial positions and undeformed state. In the simplest cases, particle characteristic functions are taken to be initially non–overlapping. However, nothing precludes overlapping, or "fuzzy" particles, as discussed in Section 3.3. Initial particle volumes, V_p^i, are defined by

$$V_p^i = \int_{\Omega^i} \chi_p^i(\mathbf{x})\,d\mathbf{x}, \tag{3}$$

where Ω^i is the initial volume of the continuum body to be discretized.

In addition to initial particle volumes, the material point initial masses, m_p^i, momenta, $\mathbf{p}_p^i$, and stresses, $\boldsymbol{\sigma}_p^i$, must be defined. These properties may be assigned by integrating properties of the continuum against the particle characteristic functions,

$$m_p^i = \int_{\Omega^i} \rho^i(\mathbf{x}) \chi_p^i(\mathbf{x})\, d\mathbf{x}, \tag{4}$$

$$\mathbf{p}_p^i = \int_{\Omega^i} \rho^i(\mathbf{x}) \mathbf{v}^i(\mathbf{x}) \chi_p^i(\mathbf{x})\, d\mathbf{x}, \tag{5}$$

where ρ^i is the continuum body's initial mass density, and $\mathbf{v}^i$ the initial velocity. Note that these definitions result in reduced particle volumes, mass and momenta for boundary particles when $\Omega^i \cap \Omega_p^i < \Omega_p^i$, where Ω_p^i denotes the support of particle characteristic function p in the initial configuration. Particle densities are defined as the ratio of particle mass to particle volume. Note that using this definition, the initial density, $\rho_p^i = m_p^i / V_p^i$, is consistent with the (volume averaged) continuum body's initial density everywhere, including on boundary particles. Similarly, particle velocities are defined as the ratio of particle momentum to particle mass, giving an initial particle velocity, $\mathbf{v}_p^i = \mathbf{p}_p^i / m_p^i$, consistent with the continuum body's. Initial values of particle Cauchy stresses, $\boldsymbol{\sigma}_p^i$, may be assigned

$$\boldsymbol{\sigma}_p^i = \int_{\Omega^i} \boldsymbol{\sigma}^i(\mathbf{x}) \frac{\chi_p^i(\mathbf{x})}{V_p^i}\, d\mathbf{x}, \tag{6}$$

where $\boldsymbol{\sigma}^i(\mathbf{x})$ is the continuum body's initial Cauchy stress. The particle stresses are also consistent with the volume averaged continuum initial stress everywhere.

Using Eqn. 2, the following identities obtain

$$\sum_p m_p^i = \sum_p \int_{\Omega^i} \rho^i(\mathbf{x}) \chi_p^i(\mathbf{x})\, d\mathbf{x} = \int_{\Omega^i} \rho^i(\mathbf{x})\, d\mathbf{x}, \tag{7}$$

$$\sum_p \mathbf{p}_p^i = \sum_p \int_{\Omega^i} \rho^i(\mathbf{x}) \mathbf{v}^i(\mathbf{x}) \chi_p^i(\mathbf{x})\, d\mathbf{x} = \int_{\Omega^i} \rho^i(\mathbf{x}) \mathbf{v}^i(\mathbf{x})\, d\mathbf{x}, \tag{8}$$

i.e., the continuum body initial mass and momentum are conserved exactly in the discretization. Employing the particle characteristic functions in the initial discretization provides exact conservation of total mass and momentum between the continuous system and its discrete representation. However, it also results in the possibility of particles near surfaces having scaled values of volume, mass and momentum relative to interior, or "bulk" particles. In practice, particularly for non–overlapping particle characteristic functions, it may be easier to approximate the spatial extent of a continuous body as a union of the support of particle characteristic functions. In that case, Eqn.s 7 and 8 only hold in the limit of infinite spatial resolution, but the initialization integrals are simplified as $\Omega_p^i \cap \Omega^i = \Omega_p^i \quad \forall \quad p$.

2.2 Discrete Solution Procedure

The main reason to explicitly detail the initial discretization technique is to identify connections to the representation of material point data in the discrete solution procedure. Given a material point property, f_p, a representation consistent with the initial discretization procedure is the sum over the material points,

$$f(\mathbf{x}) = \sum_p f_p \chi_p(\mathbf{x}). \tag{9}$$

The particle characteristic functions are used as a basis for representing particle data throughout the computational domain and determine the degree of smoothness of the spatial variation.

Using Eqn. 9 to develop a continuous representation of the particle density, ρ_p, stress, $\boldsymbol{\sigma}_p$, and rate of change of momentum density, $\dot{\mathbf{p}}_p / V_p$, splits the volume integrals into sums of integrals over particles

$$\begin{aligned} \sum_p \int_{\Omega_p \cap \Omega} \frac{\dot{\mathbf{p}}_p \chi_p}{V_p} \cdot \delta\mathbf{v}\, d\mathbf{x} + \sum_p \int_{\Omega_p \cap \Omega} \boldsymbol{\sigma}_p \chi_p : \nabla \delta\mathbf{v}\, d\mathbf{x} = \\ \sum_p \int_{\Omega_p \cap \Omega} \frac{m_p \chi_p}{V_p} \mathbf{b} \cdot \delta\mathbf{v}\, d\mathbf{x} + \int_{\partial\Omega_{\boldsymbol{\tau}}} \boldsymbol{\tau} \cdot \delta\mathbf{v}\, dS. \end{aligned} \tag{10}$$

where Ω_p denotes the current support of particle characteristic function p, and the current particle volumes are defined by

$$V_p = \int_{\Omega_p \cap \Omega} \chi_p(\mathbf{x})\, d\mathbf{x}, \tag{11}$$

analogous to Eqn. 3.

The other fundamental aspect of PIC methods is the use of a computational grid. In MPM the grid serves as a scratch pad for the solution of conservation of momentum, from which particle states are updated. To complete the discretization procedure, approximations to the admissible velocity fields, or test functions, are introduced in terms of grid vertex quantities and grid shape functions. This step is analogous to the development of FEM discrete equations. However, use of both grid and particle basis functions to represent test functions and trial functions, respectively, is a Petrov–Galerkin method, [Johnson (1987)], and therefore more akin to some of the meshless methods (in particular [Demkowicz and Oden (1986); Atluri and Zhu (2000)]) than the FEM.

The continuous representation, $g(\mathbf{x})$, of grid data, g_v, is then

$$g(\mathbf{x}) = \sum_v g_v S_v(\mathbf{x}). \tag{12}$$

Here $S_v(\mathbf{x})$ is a computational grid shape function, which takes unit value at node v and zero value at the other nodes. Further, the shape functions are required to be a partition of unity, i.e.

$$\sum_v S_v(\mathbf{x}) = 1 \quad \forall \quad \mathbf{x}. \tag{13}$$

As can be seen from Eqn.s 2, 9, 12 and 13, the grid shape functions and particle characteristic functions have similar requirements and serve analogous functions. An important difference is the continuity imposed in practice. While useful discrete equations can be developed using distributions for particle characteristic functions, the grid shape functions are typically in C^0.

Substitution of the grid shape function representation for the admissible velocity fields as in Eqn. 12, and use of the arbitrariness of the admissible velocity fields, yields the discrete governing equations

$$\dot{\mathbf{p}}_v = \mathbf{f}_v^{int} + \mathbf{f}_v^{\mathbf{b}} + \mathbf{f}_v^{\boldsymbol{\tau}}, \tag{14}$$

where

$$\dot{\mathbf{p}}_v = \sum_p \overline{S}_{vp} \dot{\mathbf{p}}_p \tag{15}$$

$$\mathbf{f}_v^{int} = -\sum_p \boldsymbol{\sigma}_p \cdot \overline{\nabla S}_{vp} V_p. \tag{16}$$

$$\mathbf{f}_v^{\mathbf{b}} = \sum_p m_p \mathbf{b} \overline{S}_{vp}, \tag{17}$$

$$\mathbf{f}_v^{\boldsymbol{\tau}} = \int_{\partial\Omega_{\boldsymbol{\tau}}} \boldsymbol{\tau} S_v(\mathbf{x})\, dS, \tag{18}$$

and, for simplicity, the specific body force is assumed to be constant. Here the rate of change of momentum on the grid is denoted by $\dot{\mathbf{p}}_v$, the "internal force" due to stress is denoted by $\mathbf{f}_v^{int}$, and the forces due to body forces and surface tractions are denoted by $\mathbf{f}_v^{\mathbf{b}}$ and $\mathbf{f}_v^{\boldsymbol{\tau}}$ respectively, and

$$\overline{S}_{vp} = \frac{1}{V_p} \int_{\Omega_p \cap \Omega} \chi_p(\mathbf{x}) S_v(\mathbf{x})\, d\mathbf{x}, \tag{19}$$

$$\overline{\nabla S}_{vp} = \frac{1}{V_p} \int_{\Omega_p \cap \Omega} \chi_p(\mathbf{x}) \nabla S_v(\mathbf{x})\, d\mathbf{x}. \tag{20}$$

The functions $\overline{S}_{vp}$ and $\overline{\nabla S}_{vp}$, will be referred to as the weighting, and gradient weighting, functions respectively. Note that both are implicitly functions of grid vertex position $\mathbf{x}_v$ and particle position $\mathbf{x}_p$ as emphasized by the subscripts. The weighting functions are also functions of the integration domain, i.e. the current particle volume.

Grid mass, m_v, and momenta, $\mathbf{p}_v$ are interpolated from the particles to initialize the grid using the weighting functions, i.e.

$$m_v = \sum_p m_p \overline{S}_{vp}, \tag{21}$$

$$\mathbf{p}_v = \sum_p \mathbf{p}_p \overline{S}_{vp}, \tag{22}$$

and grid velocities may then be defined as $\mathbf{v}_v = \mathbf{p}_v / m_v$. Because the grid shape functions are a partition of unity, Eqn. 13, and using the definition of particle volume, Eqn. 11, gives

$$\sum_v \overline{S}_{vp} = 1 \quad \forall \quad \mathbf{x}_v, \mathbf{x}_p, \tag{23}$$

i.e. the weighting functions are also partitions of unity. Using Eqn. 23 it is easily shown that

$$\sum_v m_v = \sum_v \sum_p m_p \overline{S}_{vp} = \sum_p m_p, \tag{24}$$

$$\sum_v \mathbf{p}_v = \sum_v \sum_p \mathbf{p}_p \overline{S}_{vp} = \sum_p \mathbf{p}_p. \tag{25}$$

The general weighting functions preserve the property that mass and momentum are conserved (in total, as in Eqn.s 7–8) in interpolating from particles to the grid.

Eqn. 14 gives the acceleration of the computational grid, $\mathbf{a}_v = \dot{\mathbf{p}}_v / m_v$. All that remains is to use this information to update the particles. Because there may be more particles than grid vertices, a unique relationship between particle and grid variables does not exist in general, i.e. Eqn. 15 is not invertible. Rather, particle updates are defined by

$$\dot{\mathbf{x}}_p = \sum_v \frac{\mathbf{p}_v}{m_v} \overline{S}_{vp} = \sum_v \mathbf{v}_v \overline{S}_{vp}, \tag{26}$$

$$\dot{\mathbf{p}}_p = \sum_v \frac{\dot{\mathbf{p}}_v m_p}{m_v} \overline{S}_{vp} = m_p \sum_v \mathbf{a}_v \overline{S}_{vp}, \tag{27}$$

As emphasized using the above notation, particle positions are updated using (grid) velocities, and particle velocities are updated using (grid) accelerations. Interpolating changes in position and velocity in this way serves to reduce numerical diffusion, [Brackbill, Kothe, and Ruppel (1988); Brackbill and Ruppel (1986)]. Using Eqn. 27, the total change in momentum is the same on both the particles and the grid as required by Eqn. 15,

$$\sum_p \dot{\mathbf{p}}_p = \sum_v \frac{\dot{\mathbf{p}}_v}{m_v} \sum_p m_p \overline{S}_{vp} = \sum_v \dot{\mathbf{p}}_v, \tag{28}$$

where Eqn. 21 was again used. From Eqn.s 15, 17, 21, 22, 26, and 27, it may be seen that the weighting functions, $\overline{S}_{vp}$, are used to interpolate information from grid to particles and back again. Although derived differently, the MPM algorithm and its predecessors, [Brackbill, Kothe, and Ruppel (1988); Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995); Brackbill and Ruppel (1986)], also use the same functions to interpolate to and from the computational grid.

In addition to positions and momenta, particle constitutive response must be updated consistent with the deformation of the grid. Particle strain rates are calculated by constructing a continuous approximation to the grid velocity using grid shape functions, i.e. Eqn. 12. The strain rate, $\dot{\boldsymbol{\varepsilon}}$, may then be calculated,

$$\dot{\boldsymbol{\varepsilon}}(\mathbf{x}) = \frac{1}{2}\left(\nabla \mathbf{v}(\mathbf{x}) + \nabla \mathbf{v}(\mathbf{x})^T\right) = \frac{1}{2}\left(\nabla S_v \mathbf{v}_v + \mathbf{v}_v \nabla S_v\right), \tag{29}$$

where a superscript T indicates the transpose. Particle strain rates, $\dot{\boldsymbol{\varepsilon}}_p$, are determined using a volume weighted average over each particle

$$\dot{\boldsymbol{\varepsilon}}_p = \frac{1}{V_p} \int_{\Omega_p \cap \Omega} \chi_p(\mathbf{x}) \dot{\boldsymbol{\varepsilon}}(\mathbf{x})\, d\mathbf{x} = \sum_v \frac{1}{2}\left(\overline{\nabla S}_{vp} \mathbf{v}_v + \mathbf{v}_v \overline{\nabla S}_{vp}\right). \tag{30}$$

Hence it can be seen from Eqn.s 16 and 30 that the gradient weighting functions, $\overline{\nabla S}_{vp}$, are used to interpolate information to and from the particles. The MPM algorithm, [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)], also uses the same functions to interpolate gradients to and from the computational grid.

Eqn.s 14 – 18, 21, 22, 26, 27, and 30 are identical in form to those presented for MPM, [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)]. The difference is that the weighting functions and gradient weighting functions have been generalized. These generalized weighting and gradient weighting functions may be calculated for any combination of particle characteristic functions and grid shape functions. Examples, using particular choices of particle characteristic functions and grid shape functions, will be given in the following section.

2.3 Observations

In general the weighting functions serve to smooth and distribute data more than the particle or grid basis function representations, Eqn. 9 or 12. This is a consequence of integrating over particle volumes, which smoothes the grid shape functions. The weighting functions, $\overline{S}_{vp}$, have larger support than the grid shape functions in general, and, except for special cases, $\overline{S}_{vp}|_{\mathbf{x}_p = \mathbf{x}_v} < 1$. A consequence of the inequality is that a particle whose position is coincident with a grid vertex will not interpolate data exclusively to that vertex. It is precisely this property which improves performance in handling finite deformations, as demonstrated in Section 4. This property is

also used in many other meshless methods, where, at any given spatial point, data from many particles is required to construct a spatially continuous representation of the data there, [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996); Babuška and Mellenk (1997); Demkowicz and Oden (1986); Atluri and Zhu (2000)]. For these methods the same effect is desired, namely a smoother representation of particle data. However, in the absence of a regular grid, construction of the weighting functions is only achieved at considerable effort and computational cost. The construction of the GIMP weighting functions benefits substantially from the use of a regular grid. They can be determined analytically for common particle characteristic functions and grid shape functions.

Because the GIMP methods are particle methods derived using a Petrov–Galerkin discretization scheme, of all the various meshless methods, they have the most in common with the Meshless Local Petrov–Galerkin (MLPG) Method introduced in [Atluri and Zhu (1998)] and described in more detail in [Atluri and Shen (2002b)]. Both methods generate families of algorithms, the details of which are determined by the specific choices of test and trial functions [Atluri and Shen (2002a)]. However, the MLPG method emphasizes the complete absence of a computational mesh, while the GIMP methods embrace the use of a (spatially fixed) mesh for the simplifications it provides.

It is worth noting that if weighting functions with support beyond nearest neighbor vertices are introduced arbitrarily, difficulties can arise at the boundaries of the computational grid. Particles near the grid boundary may be required to interpolate information to ghost vertices, or different interpolation rules may be required there, in order that the mass and momentum are conserved in interpolating between particles and grid. Other meshless methods, [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996); Babuška and Mellenk (1997); Demkowicz and Oden (1986); Atluri and Zhu (2000)], require special treatment at boundaries on account of the support of the interpolation functions overlapping the computational boundary. A notable exception are some MLPG methods [Atluri and Shen (2002a)]. The shape functions derived for the GIMP methods require no special treatment. Each particle only contributes to vertices whose shape functions overlap its characteristic function. By construction, contiguous particle characteristic functions are completely contained within the computational grid boundaries and hence conserve interpolated quantities to only these vertices.

It is also worth noting similarities with other particle methods, not derived from a variation form, which have been especially successful in plasma simulations, [Birdsall and Langdon (1985)]. In these methods the utility of smoother interpolation has also been recognized. Further, the construction of smoother weighting functions by integrating over particle volumes is well established, [Hockney and Eastwood (1981)]. At this point, however, the similarities end, as once data has been collected at grid points finite difference approaches are used to solve the governing equations. Construction of a sensible method requires a balance of interpolation errors, finite difference errors, and computational complexity. Global conservation of important solution variables such mass and momentum, as in Eqn.s 24 and 25, is obtained by imposing requirements on the interpolating functions. However local grid data errors due to particle disorder are only reduced, not eliminated, by using smoother interpolation functions.

In contrast, GIMP methods can be easily constructed in which local grid data errors are eliminated completely in at least one important situation. Consider the case of a uniformly stressed body, i.e. $\boldsymbol{\sigma}_p = \boldsymbol{\sigma} \quad \forall \quad p$. The internal force, from Eqn. 16 simplifies to

$$\mathbf{f}_v^{int} = -\boldsymbol{\sigma} \cdot \sum_p \overline{\nabla S}_{vp} V_p. \tag{31}$$

If particle characteristic functions which are a partition of unity in the current configuration are chosen, i.e.

$$\sum_p \chi_p(\mathbf{x}) = 1 \quad \forall \quad \mathbf{x}, \tag{32}$$

then Eqn. 31 may be further simplified, using the definition of $\overline{\nabla S}_{vp}$, Eqn. 20,

$$\mathbf{f}_v^{int} = -\boldsymbol{\sigma} \cdot \int_{\Omega_p \cap \Omega} \nabla S_v(\mathbf{x}) = \mathbf{0} \quad \textit{if} \quad \Omega_p \cap \Omega = \Omega_p, \tag{33}$$

i.e. internal forces are identically zero at "internal" grid vertices (away from the material boundary). This represents an important special case where, in the absence of body forces, static equilibrium is maintained only if Eqn. 33 holds.

MPM may be recovered from the more general formulation presented here by appropriate selection of the particle characteristic functions. Specifically, for

$$\chi_p(\mathbf{x}) = \delta(\mathbf{x}-\mathbf{x}_p)V_p, \tag{34}$$

where $\delta(\mathbf{x}-\mathbf{x}_p)$ is the Dirac delta function, the formulation presented in [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)] is recovered exactly. For this case the integrals in Eqn.s 19 and 20 are transformed into evaluations at points $\mathbf{x}_p$, i.e. $\overline{S}_{vp} = S_v(\mathbf{x}_p)$ and $\overline{\nabla S}_{vp} = \nabla S_v(\mathbf{x}_p)$. Note that the MPM discretization scheme represents a special case where grid shape functions are not smoothed in the construction of the weighting functions. Note also that *the particle characteristic functions in Eqn. 34 are not a partition of unity*, Eqn. 32. Hence a uniform stress state on the particles can result in non–zero internal forces on the computational grid. This is examined in Section 4.

Finally, it is noted that the Petrov–Galerkin method gives "lumped mass" governing equations directly (Eqn. 14). This is a consequence of the consistent use of particle and grid basis functions, for trial and test functions respectively. In the variational procedure presented in [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)], a full mass matrix is derived. The full mass matrix is then diagonalized or "lumped" for computational efficiency to give the discrete governing equations typically solved in practice. It is interesting that the Petrov–Galerkin method avoids this additional step, which typically lacks justification beyond computational tractability.

3 Example GIMP Methods

In this section examples will be given for several GIMP methods. Attention will be focused on the results obtained for various selections of particle characteristic functions χ_p. Grid shape functions S_v are identical for all cases considered, and are chosen consistent with those used in practice in MPM, [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)]. In one dimension, the shape functions are the piecewise linear "tent" functions

$$S_v(x) = \begin{cases} 0 & x - x_v \le -L, \\ 1 + (x-x_v)/L & -L < x - x_v \le 0, \\ 1 - (x-x_v)/L & 0 < x - x_v \le L, \\ 0 & L < x - x_v. \end{cases} \tag{35}$$

See also Fig. 1, where L is the cell spacing. For simplification a uniform grid is assumed, i.e. $L_v = L \;\; \forall \;\; v$. In more than one dimension, the shape functions are constructed as products of these (one–dimensional) tent functions, i.e. in three–dimensions $S_v(\mathbf{x}) = S_{v1}(x_1)S_{v2}(x_2)S_{v3}(x_3)$ where x_i are the components of $\mathbf{x}$ in the grid directions. An analogous multiplicative decomposition is available for the particle characteristic functions.

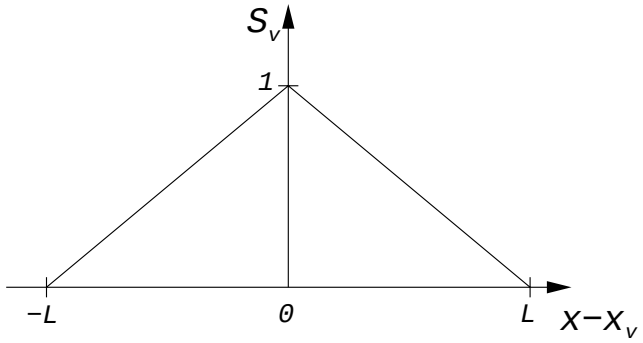

Figure 1 : One dimensional "tent" grid shape function used in all GIMP Methods presented here.

While straight–forward conceptually, implementation of GIMP methods with finite, deforming particles in three dimensions does suffer a practical complication. Performing the integrations to determine the weighting functions requires integration over the current support of the particle characteristic functions, as they deform and rotate relative to the computational grid. This may need to be done numerically in general. The investigation in the following section is performed in one–dimension for simplicity. In one–dimension the weighting functions can be determined analytically.

For this reason it is worth investigating the performance of approximate GIMP algorithms, where particle deformations are not tracked. For these algorithms all complications associated with integrating over the current support of the particle characteristic functions are obviated

and the weighting functions may be calculated analytically, but at the cost of errors associated with the particle characteristic functions not forming a partition of unity. The following section investigates the performance of MPM and contiguous particle GIMP algorithms with and without tracking particle deformations.

3.1 Material Point Method

As mentioned in the previous section the original MPM discrete equations may be obtained from the principle of virtual work by selecting the Dirac delta function for the particle characteristic functions as in Eqn. 34, [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)]. For this case $\overline{S}_{vp} = S_v(\mathbf{x}_p)$ and $\overline{\nabla S}_{vp} = \nabla S_v(\mathbf{x}_p)$. This formulation has the virtue that is computationally efficient because a given particle interpolates information only to the vertices of the grid cell it is contained in, and a grid vertex interpolates information only to particles in adjacent cells. However, this also results in the interpolation being strongly dependent on the registration of the particles on the grid, potentially changing abruptly as particles cross cells.

Because these particle characteristic functions are not a partition of unity, they are not suitable for use in the initial discretization procedure presented in Section 2.1. This is easily overcome by selecting other particle characteristic functions for the initial discretization, or simply using another initial discretization procedure altogether. However, this situation points out an inconsistency wherein the material points mass and volume are assigned as required by the initial mass density of the continuum body on one hand, but then are assumed to represent infinitesimal points on the other. It is certainly more intuitively appealing to use the same particle characteristic functions to prescribe material point properties in the initial discretization, and as basis functions for the discrete solution procedure.

3.2 Contiguous Particles GIMP Method

The simplest choice of particle characteristic functions (of finite extent) in one–dimension is the combination of step functions, $H(x)$ ($H(x) = 0$ if $x < 0$ and $H(x) = 1$ if $x > 0$),

$$\chi_p(x) = H(x-(x_p-l_p)) - H(x-(x_p+l_p)). \tag{36}$$

Here $2l_p$ is the current particle size. The initial size, $2l_p^i$, is determined by dividing the cell spacing L by the number of particles per cell. This selection of particle characteristic functions defines "contiguous particles", i.e. contiguous regions of non–overlapping support Ω_p. The particle characteristic functions may also be written,

$$\chi_p(x) = \begin{cases} 1 & \text{if} \quad x \in \Omega_p, \\ 0 & \text{otherwise.} \end{cases} \tag{37}$$

This generalization is the simplest finite particle generalization in the sense that it retains the grid shape functions used in the original implementation of MPM [Sulsky, Chen, and Schreyer (1994); Sulsky, Zhou, and Schreyer (1995)], but replaces particle mass points with particle volumes. It represents the very next degree of smoothness obtainable in the family of GIMP methods (with tent grid shape functions).

Note that for this case the weighting and gradient weighting functions, Eqn.s 19 and 20, simplify to

$$\overline{S}_{vp} = \frac{1}{2l_p}\int_{x_p-l_p}^{x_p+l_p} S_v(x)\,dx, \tag{38}$$

$$\overline{\nabla S}_{vp} = \frac{1}{2l_p}\int_{x_p-l_p}^{x_p+l_p} \nabla S_v(x)\,dx, \tag{39}$$

and $\overline{\nabla S}_{vp}(x_p) = \partial \overline{S}_{vp}(x_p)/\partial x_p$, where the dependence on particle position is indicated explicitly. Specifically, using the piecewise linear tent function for grid shape function $S_v(x)$, Eqn. 35, gives from Eqn. 38,

$$\overline{S}_{vp} = \begin{cases} 0 & x_p - x_v \le -L - l_p, \\ \frac{(L+l_p+(x_p-x_v))^2}{4Ll_p} & -L-l_p < x_p-x_v \le -L+l_p, \\ 1+\frac{x_p-x_v}{L} & -L+l_p < x_p-x_v \le -l_p, \\ 1-\frac{(x_p-x_v)^2+l_p^2}{2Ll_p} & -l_p < x_p-x_v \le l_p, \\ 1-\frac{x_p-x_v}{L} & l_p < x_p-x_v \le L-l_p, \\ \frac{(L+l_p-(x_p-x_v))^2}{4Ll_p} & L-l_p < x_p-x_v \le L+l_p, \\ 0 & L+l_p < x_p-x_v. \end{cases} \tag{40}$$

See also Fig. 2. The result is a weighting function with support in adjacent cells and in next nearest neighbor

cells. This specialization has the advantage that it develops weighting functions in C^1 with a minimal amount of additional complexity, both theoretically and computationally. It will be referred to as the "contiguous particle GIMP method". The increased support does result in an increase in computational effort. The grid vertex quantities appearing in the governing equations (Eqn. 14) are accumulated by summing over particles. The contribution of a material point's data to a grid vertex is determined by the distance between its centroid and the grid vertex. In the contiguous particle GIMP method additional grid vertices to which a given material point contributes data must be located.

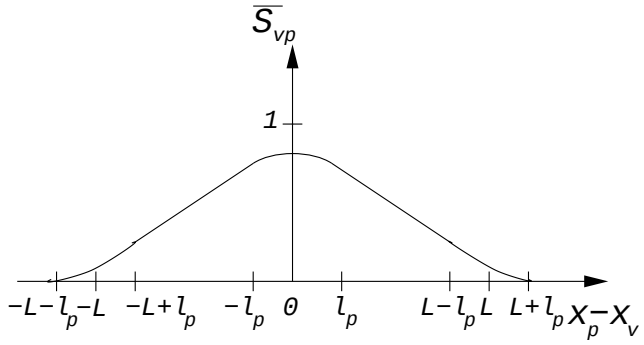

Figure 2 : Contiguous particle GIMP weighting function in one dimension.

An approximation to the exact weighting functions, which provides considerable simplification in practice, is provided by not tracking current particle volumes, i.e. $l_p(t) = l_p^i \quad \forall \quad p,t$. This is a good approximation for small deformations. However, errors develop when particle volumes fail to remain contiguous, or overlap the computational boundaries, during finite deformations. For example, when a particle's (approximate) characteristic function overlaps the computational grid boundary, the mass and momentum corresponding to the fraction of the particle characteristic function which lies outside the grid are not interpolated to the grid. Hence errors in conservation of mass and momentum between grid and particles are made. Examples are given in Section 4.

3.3 Fuzzy Particle GIMP Methods

While not examined in detail, some aspects and possible merits of overlapping particle characteristic functions, or "fuzzy particles", are considered here. Eqn. 32 constrains the form of the particle characteristic functions, but two possibilities immediately present themselves. These are illustrated in Fig 3 for "tent" particle characteristic functions in one–dimension, in the case where two particles initially occupy each cell. Note that for one particle per cell the two possibilities in Fig. 3 converge.

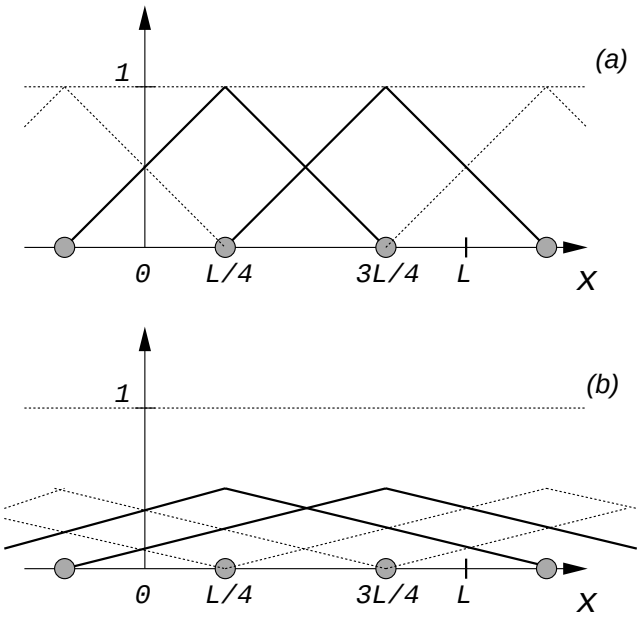

Figure 3 : Two possible fuzzy particle discretization schemes illustrated in one–dimension for two particles per cell. All particle characteristic functions which contribute to the continuous representation of particle data for $0 \le x \le L$ are shown, with particle characteristic functions centered between 0 and L in bold.

The first possibility, Fig. 3(a), is characterized by particles only overlapping their nearest neighbors, and particle characteristic functions uniquely representing the grid data at one or more points. For these cases $f(x_p) = f_p$ in Eqn. 9 and $\Omega_p^i \le 2L$. A second possibility is illustrated in Fig. 3(b) and is characterized by particle characteristic function overlap with all neighbors initially within some region and $f(x_p) \neq f_p$. In Fig. 3(b) this case is illustrated for $\Omega_p^i = 2L$. This possibility is akin to that generally preferred in various meshless methods where basis function support is typically taken to be much larger than particle spacings and hence many particles typically contribute to continuous representation of particle data at any given spatial point, [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996); Babuška and Mellenk

(1997); Demkowicz and Oden (1986); Atluri and Zhu (2000)].

Both overlapping particle scenarios are a natural extension of the piecewise constant representation of particles (contiguous particle GIMP) to piecewise linear representations. Nearest neighbor only overlap tends to localize particle data while constant overlap tends to distribute it. Further analysis/experimentation is needed to determine the merits of each possibility. However, their complementary functionality is suggestive of utility in particle addition and deletion algorithms in conjunction with computational grid refinement and coarsening.

4 Example Calculations

In this section numerical solution artifacts associated with the properties of the interpolation scheme used to transfer information between particles and grid are demonstrated. Numerical solutions using two different particle characteristic functions, as detailed in the previous section, are compared. One solution is obtained using the MPM algorithm, where material points are taken as infinitesimal points. For this case $\overline{S}_{vp} = S_v(x_p)$ where $S_v(x)$ is given by the piecewise linear tent function as in Eqn. 35 or Fig. 1. This solution is labeled "MPM" in Figs. 4, 5, 6, and 7. Another solution is obtained using contiguous particle GIMP, where particle characteristic functions are given by Eqn. 37, and $\overline{S}_{vp}$ is given by Eqn. 40 or Fig. 2. This solution is label "GIMP". For both of these solutions particle volumes are not tracked, i.e. $l_p(t) = l_p^i \quad \forall \quad p,t$.

Two one–dimensional examples are examined. Both simulate uniaxial compression of a bar or column, the first quasi–statically, the second dynamically. In both examples the bar has unit initial mass density, ρ_0, and an initial length, Δ_0, of 50. Material response is determined by a one–dimensional hyperelastic model

$$\sigma = E(F-1), \tag{41}$$

where F is the deformation gradient in one dimension. Young's Modulus, E, is taken to be 10^6, giving a wave speed, c, of 1000. Any consistent set of units suffices. For the first two subsections the bar is discretized using 50 computational cells, $L = 1$, and two material points per cell. The last two subsections consider variations in cell size and the number of particles per cell. The time for a wave to propagate the length of the bar, or the "wave transit time" is $\Delta_0/c = .05$. The explicit time stepping algorithm used in all numerical solutions is governed by the CFL stability condition which demands that for linear systems the time step, Δt, satisfy $\Delta t \leq L/c = 10^{-3}$. For all (nonlinear) calculations presented here the time step is reduced by an additional factor of ten, i.e. $\Delta t = 10^{-4}$, both to assure numerical stability and to increase the accuracy of the computations.

Results are reported at specific stages in the deformation by plotting stresses at material points. Attention is focused on the appearance of computational artifacts associated with particles crossing cell boundaries, and the effect on material point stresses. While the resolution of the computation is commensurate with the grid cell size, constitutive response is computed on material points only. In order that inelastic constitutive response be accurately simulated, material point stresses must vary smoothly. In both cases examined here average deformations are small, but are sufficient to cause particles to cross cell boundaries. Of course cell crossing artifacts exhibited at this relatively coarse discretization occur at yet smaller strains if the spatial resolution is increased. This is examined in more detail at the end of the section.

4.1 Quasi–static Compaction

The quasi–static compaction case simulates the response of a column of material to a slowly increasing compressive body force b (e.g. gravity). The magnitude of the body force is increased linearly with time during the computations. In order to obtain good quasi–static solutions, the total simulation time is taken equal to 40 wave transit times. Results are depicted in Figs. 4 and 5, where vertical lines indicate cell boundaries. Each material point position and stress is indicated with open triangles for the solution obtained using MPM, and with open circles for contiguous particle GIMP. Dashed lines connecting the material points emphasize non–uniformity in the numerical solutions.

Analytical solutions for static equilibrium at any given body force may be easily found. These solutions are

$$\sigma(x) = E\left\{\sqrt{\frac{2\rho_0 b}{E}(\Delta - x) + 1} - 1\right\} \qquad 0 \leq x \leq \Delta, \tag{42}$$

where x denotes position in the current configuration, and Δ is the current length of the column. This solution is

plotted for guidance as a solid line in Figs. 4 and 5. The current position of each point may be calculated, in particular, the current length of the column is given by

$$\Delta = \Delta_0 + \frac{\rho_0 b}{2E}\Delta_0^2. \tag{43}$$

This equation is used to select the magnitude of the final body force, b_f, such that the end displacement is one grid cell at the end of the simulation, i.e. $\Delta - \Delta_0 = -1$. For this case $b_f = -800$. Results are reported at various column weights per unit cross–sectional area, W, defined by

$$W = \rho_0 |b| \Delta_0. \tag{44}$$

For the parameters chosen, the weight of the column at the end of the simulation is $W_f = 40,000$.

Fig. 4(a) depicts solutions early in the deformation, before any material points have crossed cell boundaries. For this case both numerical solutions and the analytical solution essentially overlie one another. Fig. 4(b) depicts the situation shortly after the first cell crossings. The exact solution predicts that at this column weight the left particles initially in cells for which $x_v \geq 30$ will have crossed their left cell boundaries, resulting in 3 material points in the cell occupying $29 \leq x \leq 30$. These cell crossings have substantially perturbed the MPM solution, both for $x \geq 29$, where the cell crossings have occurred, and in the remainder of the column, due to propagation of the cell crossing disturbances.

It is not difficult to understand cell crossing noise. Consider uniformly stressed material, as discussed in Section 2.3, but in only one dimension. Because of the discontinuous nature of the gradient of the weighting function used in MPM, a force imbalance develops when uniformly stressed material points register non–uniformly on the grid, i.e. different numbers of particles are in adjacent cells. This force imbalance is proportional to the particle stress multiplied by the difference between the number of particles on each side of a grid vertex. The contiguous particle GIMP gradient weighting functions reduce these artificial force imbalances substantially. As gaps or overlaps in particle characteristic functions develop (as material points move relative to each other), the particle characteristic functions are no longer a partition of unity and artificial force imbalances can still develop. However, these force imbalances are proportional to the stress multiplied by the ratio of the gap or overlap volume to the initial particle volume, and only develop when the gaps or overlaps register non–uniformly on the grid.

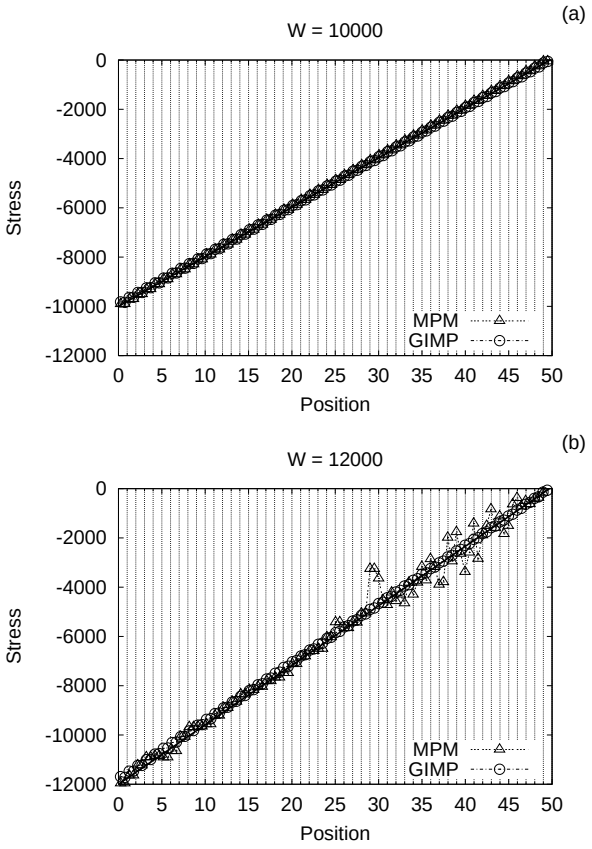

Figure 4 : Numerical and analytical solutions to the quasi–static compression of a continuum bar due to a slowly increasing body force *b*. The MPM solution material point stresses are indicated with open triangles and the contiguous particle GIMP solution material point stresses with open circles. Vertical lines indicate cell boundaries. The solid diagonal lines indicate analytical solutions. Solutions are obtained at increasing column weights.

Fig. 5(a) depicts solutions for a still larger column weight. The MPM solution has degraded substantially. The spurious forces developed due to the low degree of continuity in interpolation are overwhelming the "physical" forces in the system and destroying the solution. The GIMP solution remains smooth, although a slight difference between it and the analytical solution has developed. In Fig. 5(b) only traces of the MPM solution remain. The GIMP solution remains smooth, but fur-

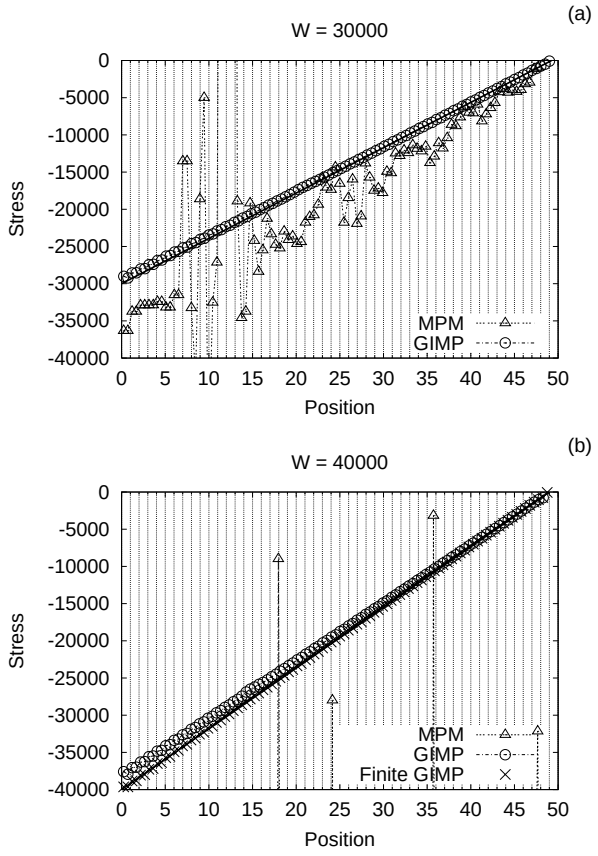

Figure 5 : As in Fig. 4, but for still higher column weights. In (b), the Finite GIMP solution material point stresses are indicated with x's.

ther under predicts the full weight of the column (the stress at $x = 0$). Because the GIMP solution uses initial particle widths throughout the computation, as a particle approaches the boundary its characteristic function begins to overlap the edge of the computational grid. The overlapping portion information is not interpolated to the grid. Therefore in the GIMP algorithm, the solution on the grid is one for slightly reduced column mass/weight as discussed in Section 3.2.

For this final weight one more numerical solution, labeled "Finite GIMP" is presented in Fig. 5(b). The Finite GIMP solution overlies the analytical one. In Finite (contiguous particle) GIMP, particle widths are tracked and used in the interpolation functions $\overline{S}_{vp}$. Hence the particle characteristic functions remain a partition of unity throughout the deformation. Particle characteristic function overlap at the boundary is eliminated and the full weight of the column is reflected in the solution.

In summary, the MPM algorithm can be very noisy. Neighboring material points may experience computational artifacts resulting in radically different deformation histories, compromising the accuracy of stress evaluations, especially for history dependent, inelastic materials. Material point stress oscillations may even become strong enough to overwhelm the physical forces in the system and destroy the solution, as seen in this example.

4.2 Dynamic Compaction

The second case of interest is the dynamic compaction of the same bar. For this case the bar is given an initial velocity to the left, impacting a rigid wall (the computational boundary). The initial velocity, v_0, is taken to be $c/50$. In this case, for the same end displacement, $\Delta - \Delta_0 = -1$, there isn't time for stress equilibrium to take place, rather the solution is a stress wave. The simulation is run for one wave transit time. A Richtmyer–VonNeumann artificial viscosity term is added to smear the shock over several computational cells, [VonNeumann and Richtmyer (1950)], and a linear term is added to damp out ringing behind the shock. The general form of the artificial viscosity pressure, q, is

$$q = \begin{cases} c_1 \rho c |\dot{\varepsilon} L| + c_2 \rho (\dot{\varepsilon} L)^2 & \dot{\varepsilon} < 0, \\ 0 & \text{otherwise}, \end{cases} \tag{45}$$

where c_1 and c_2 are dimensionless coefficients. Analogous to the constitutive response, the artificial viscosity term is evaluated at the particles. It is then used to augment the particle stresses, i.e. $\sigma_p \to \sigma_p - q_p$. For the calculations presented here, coefficients $c_1 = .2$ and $c_2 = 2.0$ were used.

Figs. 6 and 7 show the stress state in the bar, as given by both numerical solutions, at the same end displacements as in the quasi–static example. Again each material point position and stress is indicated with open triangles for the solution obtained using MPM, and open circles for GIMP. Vertical lines indicate cell boundaries. The exact solution is a discontinuity propagating to the right with speed c. In front of the shock the material is uncompressed and stress free, while behind it material is uniformly compressed. Behind the shock front the deformation gradient is equal to the ratio of the current length of compressed material to its initial length

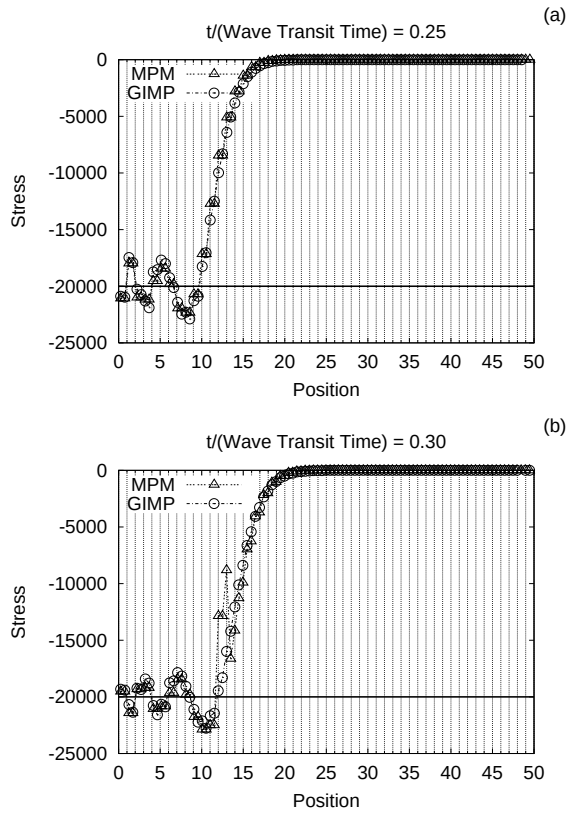

Figure 6 : MPM and GIMP numerical and analytical solutions for the dynamic compaction of a continuum bar. The MPM solution material point stresses are indicated with open triangles and the contiguous particle GIMP solution material point stresses with open circles. Vertical lines indicate cell boundaries. The solid horizontal lines indicate the analytical solution for the stress behind a sharp shock (a discontinuity). Solutions are obtained at various fractions of the wave transit time.

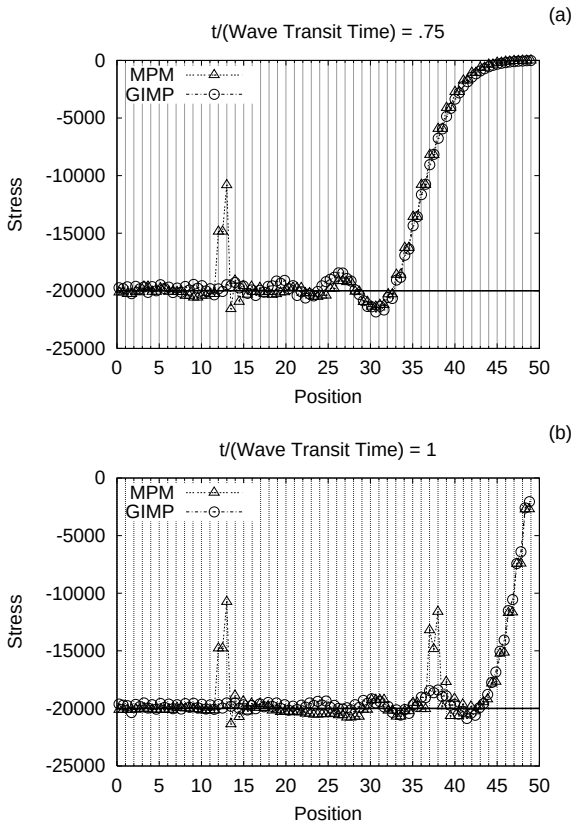

Figure 7 : As in Fig. 6, but at later times.

$$F = \frac{ct - v_0 t}{ct} = 1 - v_0/c. \tag{46}$$

Using the selected material properties, and Eqn. 41, the stress behind the shock may be calculated to be -20000. This value is indicated with the solid horizontal lines in Figs. 6 and 7.

Fig. 6(a) depicts solutions early in the deformation, before any material points have crossed cell boundaries. For this case both numerical solutions are similar. However, close inspection reveals that stresses are uniform within cells in the MPM solution, while in the GIMP solution particle stress variation within cells is more consistent with the smeared shock front. Fig. 6(b) depicts the situation shortly after the first cell crossing behind the shock. Note that cell crossings ahead of the shock have no effect, because the stress is zero there. From Eqn. 46 it may be found that the first cell crossing occurs at the vertex $x_v = 13$ resulting in 3 material points in the cell occupying $12 \leq x \leq 13$. As in the quasi–static case, the cell crossing substantially perturbs the MPM solution, but not the GIMP solution.

Fig. 7(a) depicts solutions after the shock has propagated across 75% of the bar. The stress disturbance in the MPM solution is locked in as required to maintain equilibrium on the grid for non–uniform particle registration. The GIMP solution is much smoother. In Fig. 7(b) the shock has reached the end of the bar, resulting in another cell with three material points as evident in the MPM solution.

As for the quasi–static compression case presented earlier, the Finite contiguous particle GIMP method may be applied in this case to further improve the discrete solu-

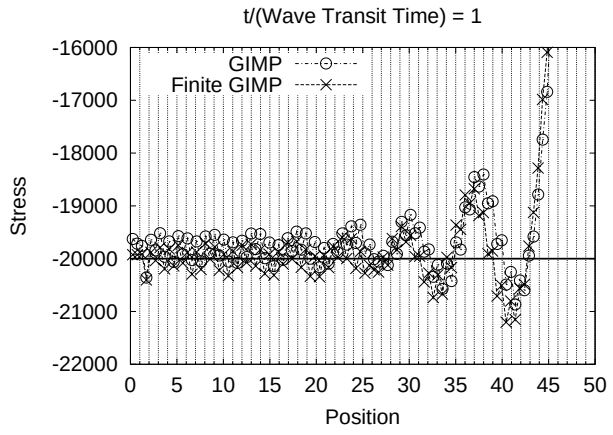

Figure 8 : Various GIMP numerical solutions for the dynamic compaction of a continuum bar after the stress wave has reached the end of the bar. The contiguous particle GIMP solution material point stresses are indicated with open circles, and the Finite contiguous particle GIMP solution material point stresses are indicated with x's. Vertical lines indicate cell boundaries. The solid horizontal line indicates the analytical solution for the stress behind a sharp shock.

tion. Fig. 8 depicts contiguous particle GIMP and Finite contiguous particle GIMP (labeled "Finite GIMP") solutions at the end of the simulations, when the compression wave has propagated across the entire bar. The range over which stress is plotted is greatly reduced in order to allow the difference between the solutions to be distinguished. It is found that, as for the quasi–static compression case, information is lost to the grid when finite deformations are not accounted for. It should be noted that MPM does not suffer from information loss to the grid unless a particle leaves the computational domain. Tracking current particle volumes corrects the MPM solutions as well, but does not alleviate cell crossing noise.

4.3 Sensitivity to Particle Discretization

To assess the effect of particle density used in the numerical simulations, the wave propagation example was also run with 1 and 4 particles/cell initial discretizations for both MPM and contiguous particle GIMP algorithms. The results, along with the 2 particle/cell results presented in Section 4.2, are depicted in Fig. 9, at the end of the simulations. In comparing results for MPM, Fig. 9(a), and contiguous particle GIMP, Fig. 9(b), it is evident that regardless of the number of particles used, the GIMP solutions result in much smoother variation of particle stresses. The MPM solutions suffer large variations in particle stresses due to cell crossings and consequent non–uniform registration of the particles on the grid. The frequency of these large particle stress variations is strongly dependent on the number of particles used. This is expected and due to the direct relationship between the number of particles used and the details of their non–uniform registration on the grid. Although the magnitude of the large variations in particle stresses is slightly reduced by using more particles, the increase in frequency of these variations results in a noisier solution overall.

It is worth noting, from Eqn. 40, that as $l_p \to 0$, then $\overline{S}_{vp} \to S_v(\mathbf{x}_p)$, i.e. the GIMP weighting functions tend toward the MPM ones as more particles are used. Shorter wavelength noise consistent with this limiting behavior can be seen in the GIMP solutions for 4 particles/cell in Fig. 9(b). Because increasing the particle density is generally expected to increase solution smoothness in PIC methods, the result is unexpected. However, as it suggests the use of fewer particles/cell is advantageous, it is of significant practical importance. Computational effort and storage requirements are, of course, strongly influenced by the number of particles used, especially in three–dimensions.

4.4 Convergence

In order to gain further insight into the performance of the various algorithms investigated in this section, the dependence of solution quality on grid spacing was also investigated. Because an exact solution is available for the case of a column quasi–statically compressed under gravity, and the numerical solutions are not complicated by artificial viscosity, this case was chosen for further study.

Of continuing interest is the quality of the solution on the particles. An error measure was chosen which compares particle stresses to the exact solution at current particle locations. Specifically,

$$\text{Error} = \sum_p \frac{|\sigma(x_p) - \sigma_p| 2 l_p}{W \Delta_0} \tag{47}$$

where $\sigma(x)$ is the exact solution given by Eqn. 42, and x_p, σ_p, and l_p are particle positions, stresses and dimensions,

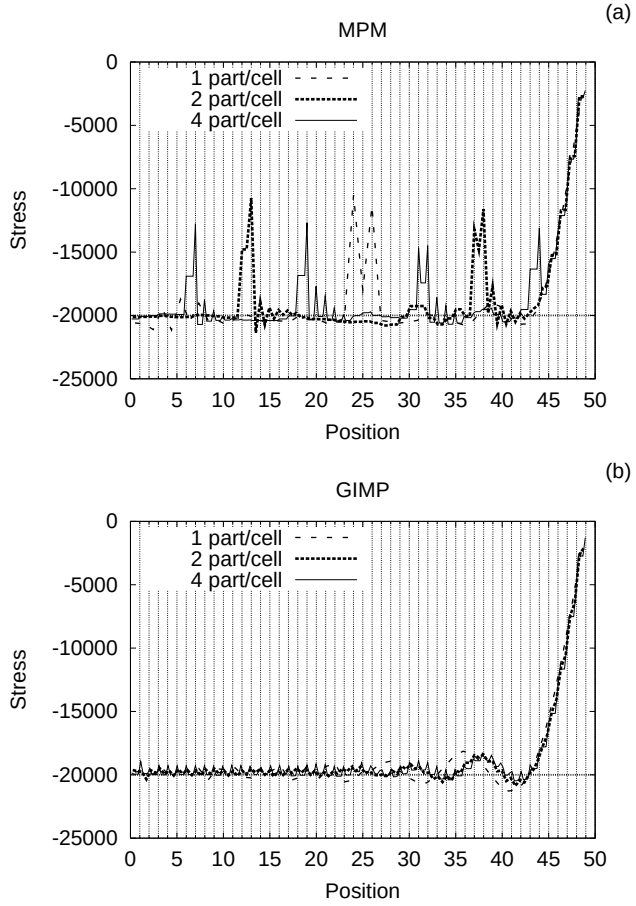

Figure 9 : MPM (a) and contiguous particle GIMP (b) solutions of the dynamic compaction of a continuum bar after the stress wave has reached the end of the bar. Different solutions have different numbers of particles/cell. Vertical lines indicate cell boundaries. The lightly dashed horizontal lines indicate the analytical solution for the stress behind a sharp shock.

respectively, from numerical simulations. The summation is normalized by the weight of the column multiplied by the initial length, to give a unitless error measure.

Gravity was applied slowly to the system in the numerical simulations, as described in Section 4.1, to obtain quasi–static results. However, because the numerical solutions track the system's dynamics, there will always be some variation relative to the analytical (static) solution, Eqn. 42, on account of stress waves. In order to get similar contributions to the error calculations due to dynamics for all cases considered, the same time step size was used regardless of cell size. This results in the same temporal discretization of the applied loading for all numerical simulations.

Errors are reported in Fig. 10, for cell sizes spanning three orders of magnitude, for the MPM, GIMP and Finite GIMP algorithms. The number of particles per cell in the initial configuration, labeled "p/c" in the figure's legend, is varied between one and four. The dashed line indicates first–order convergence with cell size, and the solid line indicates second–order convergence (in this norm, i.e. Eqn. 47). It should be noted that the data from Fig. 4(a) have been used in the error calculations, resulting in the filled and open circles on the vertical line corresponding to a cell size of 1 (both are clustered with the majority of the data for this cell size and are difficult to distinguish). This time, early in the computation ($W = 10000$), was chosen because the solution quality depicted in Fig. 4(a) looks good for both algorithms. As discussed in Section 4.1, it corresponds to a time for this cell size which is prior to any cell crossing events. For cell sizes 2.5 and larger, there are no cell crossing events for any of the particle densities considered. Except for two points, discussed specifically below, the errors for cell sizes of one or larger are all closely clustered, regardless of the algorithm or particle density. For cell sizes less than one, the errors exhibit trends strongly dependent on the numerical algorithm.

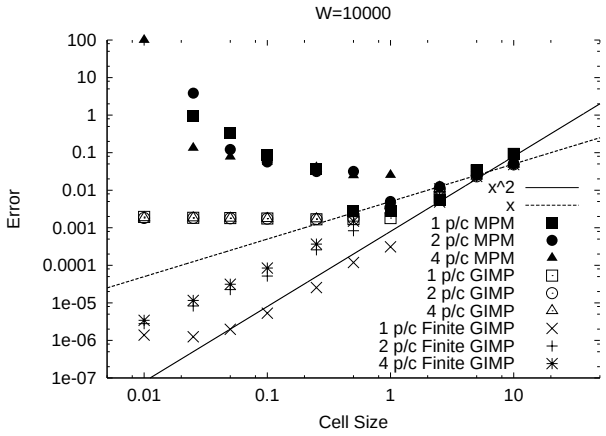

Figure 10 : Results of a convergence study for the MPM, GIMP and Finite GIMP algorithms, for various particle densities. Error relative to the exact solution is plotted against computational cell size. The computation is of the quasistatic compression of a column under gravity, identical to that depicted in Fig. 4(a).

The error calculations for the MPM algorithm are depicted with filled symbols. In all cases examined the error begins to increase with decreasing cell size after some minimum cell size is reached. This behavior is the result of the excitation, by cell crossing noise, and subsequent unstable growth, of solution instabilities. Specifically, PIC methods have been known to exhibit an instability, denoted the "finite grid instability", which originates because the particles can support solution variation wavelengths which are unresolvable on the computational grid. This instability has been shown to depend on the degree of smoothness of the weighting functions, the Mach number of the problem, and the numerical integration scheme used, [Brackbill (1988)]. The low speed, explicit calculations presented here would be expected to be very susceptible to the finite grid instability. The dramatic increase in error with decrease in cell size is due to unstable growth of perturbations to the solution created by cell crossing noise. Unfortunately, there is no way to decouple the deleterious effect the finite grid instability has on the solution from the positive effect decreasing the cell size has. The reason is simply that, for the same deformation, decreasing the cell size results in increasing the number of cell crossings and corresponding noise. Increasing the resolution will always ultimately result in cell crossings, no matter how small the deformation. An example of a numerical solution in which cell crossing noise has grown unstably, resulting in large error, may be seen in Fig. 4(c).

Further examination of Fig. 10 confirms that cell crossing noise drives the MPM algorithm solution error. Increasing the particle density decreases the magnitude of the cell crossing noise. The MPM algorithm data in Fig. 10 are consistent with this, at least until the error becomes large. After a minimum is reached (which depends on both cell size and particle density), error accumulates more slowly with increased particle density. Further evidence for the importance of the role of cell crossing noise is found in the unit cell size MPM algorithm error calculations. The four particle per cell calculation has substantially more error than the one or two particle per cell calculations. It is the only MPM calculation which has had cell crossings at this column weight and grid resolution.

The GIMP algorithm simulation errors are plotted with open symbols. This algorithm is found to be convergent over a larger range of cell sizes than MPM, as would be expected on account of dramatically reduced cell crossing noise inherent in the contiguous particle GIMP algorithm. Although difficult to discern in Fig 10, it is found that one particle per cell computations have consistently less error than the other GIMP simulations. This result is consistent with the fact that the GIMP interpolation functions are smoothest when only one particle is used per cell, as discussed in the previous subsection. Ultimately the error saturates with decreasing cell size, independent of the particle density. The numerical results continue to converge, but to the solution for a column with slightly reduced mass, due to boundary effects, as discussed in conjunction with Fig. 4. For this algorithm the combination of smoother interpolation and reduced cell crossing noise prevents the finite grid instability from dominating the quality of the calculations.

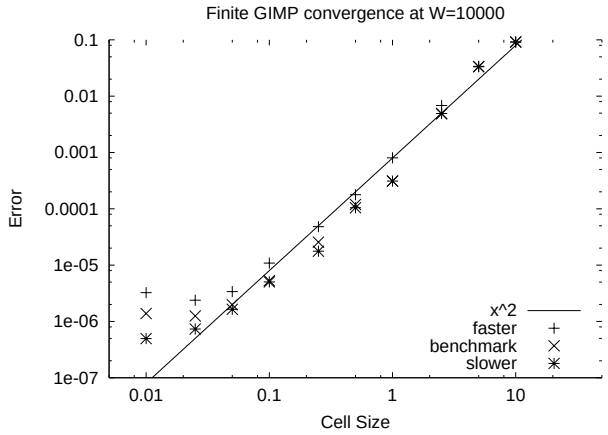

Figure 11 : Results of a convergence study for the Finite GIMP algorithm using one particle per cell, for various loading rates. Error relative to the exact solution is plotted against computational cell size. Gravity is applied five times faster, and 4 times slower, than for the results in Figs. 4 and 10.

The Finite GIMP simulation errors are plotted with various crossed–line symbols. This algorithm is found to converge the most rapidly over the entire range of cell sizes considered. Convergence rates are consistent with the smoothness of the weighting functions, which are quadratic when only one particle is used per cell. The spread between one particle per cell results and those obtained with more particles is most evident for cell sizes less than or equal to one and again suggests that one particle per cell may be the optimal discretization scheme.

Ultimately the error saturates with decreasing cell size in the one particle per cell Finite GIMP computations. The numerical simulations resolve dynamic stress wave propagation in the column. Gravity is applied very slowly in order to reduce the magnitudes of the stress waves, but they cannot be completely eliminated. Calculations in which gravity was applied five time faster and four times more slowly resulted in the error saturating at larger and smaller values, respectively, as seen in Fig. 11. This suggests the primary contribution to the error at the finest grid sizes is the difference between the quasi–static numerical solution and the exact (static) solution, i.e. the error saturation is caused by resolving the dynamics.

5 Conclusions

A family of PIC methods has been derived from a variational form using a Petrov–Galerkin discretization scheme. This family of methods has been named the Generalized Interpolation Material Point (GIMP) methods to indicate their relation to the Material Point Method (MPM), [Sulsky, Zhou, and Schreyer (1995)], which can be derived as a special case. This generalized framework allows comparison to other discretization schemes which also do not use a body–fixed mesh, such as Smooth Particle Hydrodynamics, Element Free Galerkin, and Reproducing Kernel Particle methods, [Belytschko, Krongauz, Organ, Fleming, and Krysl (1996)], the Partition of Unity Method, [Babuška and Mellenk (1997)], Adaptive Characteristic Petrov–Galerkin Finite Element, [Demkowicz and Oden (1986)], and Meshless Local Petrov–Galerkin methods, [Atluri and Zhu (2000)]. Two practical advantages of the GIMP methods stand out it this context. The first is the advantage of using a spatially fixed mesh in generating the weighting functions used to develop continuous representations of discrete data. The second is the lack of any special treatment required near computational boundaries due to weighting function overlap.

The GIMP methods were investigated because of concern about the noise inherent in MPM. Constitutive models are implemented only on the particles in all GIMP methods. In evaluating constitutive response on material points, there is an implicit assumption that neighboring points have similar histories. Computational artifacts result in nonphysical input to the material point constitutive models, and in large variance between adjacent material points and/or oscillations. These artifacts result in unphysical stress states, poor solution quality, and even complete breakdown of the solution, as illustrated in this manuscript for elastic materials. These artifacts will result in erroneous evolution of history variables in inelastic constitutive models. If there is sufficient dissipation to prevent a complete breakdown, solution divergence may go undetected. In MPM, material point stresses can be strongly perturbed in this manner when particles cross cells. For this reason, it is suggested that the MPM algorithm only be used for infinitesimal deformation problems, defined in practice by the absence of cell crossings. For finite deformation problems, greater continuity in interpolation between material points and the computational grid is required for accurate solutions. The very next degree of smoothness available in the family of GIMP methods, dubbed the contiguous particle GIMP method, is demonstrated to perform decidedly better.

Acknowledgement: The authors would like to acknowledge helpful discussions with Dr. V. Mousseau, T–Division, Los Alamos National Laboratory. This work was supported by the U.S. Department of Energy through the Center for the Simulation of Accidental Fires and Explosions, under grant W-7405-ENG-48, and by the Air Force Office of Scientific Research, through the Research Institute for Autonomous Precision Guided Systems, under grant AFOSR–F49620–00–1–0288 .

References

Atluri, S. N.; Shen, S. (2002a): The meshless local petrov–galerkin (MLPG) method: a simple & less–costly alternative to the finite element and boundary element methods. *CMES: Computer Modeling in Engineering & Sciences* , vol. 3 no. 2, pp. 11–52.

Atluri, S. N.; Shen, S. (2002b): *The meshless local petrov–galerkin (MLPG) method.* Tech Science Press.

Atluri, S. N.; Zhu, T. (1998): A new meshless local petrov–galerkin (MLPG) approach in computational mechanics. *Comp. Mech.*, vol. 22, pp. 117–127.

Atluri, S. N.; Zhu, T. (2000): New concepts in meshless methods. *Int. J. Num. Meth. Eng.*, vol. 47, pp. 537–556.

Babuška, I.; Mellenk, J. M. (1997): The partition of unity method. *Int. J. Num. Meth. Eng.*, vol. 40, pp. 727–758.

Bardenhagen, S. G.; Brackbill, J. U. (1998): Dynamic stress bridging in granular material. *J. Appl. Phys.*, vol. 83, pp. 5732–5740.

Bardenhagen, S. G.; Brackbill, J. U.; Sulsky, D. (2000): Numerical study of stress distribution in sheared granular material in two dimensions. *Phys. Rev. E*, vol. 62, pp. 3882–3890.

Bardenhagen, S. G.; Brackbill, J. U.; Sulsky, D. (2000): The material point method for granular materials. *Comput. Meth. Appl. Mech. Eng.*, vol. 187, pp. 529–541.

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

Bardenhagen, S. G.; Harstad, E. N.; Maudlin, P. J.; Gray, G. T.; Foster, J. C. (1998): Viscoelastic models for explosive binder materials. *Shock Compression of Condensed Matter – 1997*, pp. 281–284.

Belytschko, T.; Krongauz, Y.; Organ, D.; Fleming, M.; Krysl, P. (1996): Meshless methods: an overview and recent developments. *Comp. Meth. Appl. Mech. Eng.*, vol. 139, pp. 3–47.

Birdsall, C. K.; Langdon, A. B. (1985): *Plasma Physics via Computer Simulation*. McGraw–Hill.

Brackbill, J. U. (1988): The ringing instability in particle–in–cell calculations of low–speed flow. *J. Comp. Phys.*, vol. 75, pp. 469–492.

Brackbill, J. U.; Kothe, D. B.; Ruppel, H. M. (1988): FLIP: a low dissipation, particle–in–cell method for fluid flow. *Comp. Phys. Comm.*, vol. 48, pp. 25–38.

Brackbill, J. U.; Ruppel, H. M. (1986): FLIP: a method for adaptively zoned, particle–in–cell calculations in two dimensions. *J. Comp. Phys.*, vol. 65, pp. 314–343.

Burgess, D.; Sulsky, D.; Brackbill, J. U. (1992): Mass matrix formulation of the FLIP particle–in–cell method. *J. Comp. Phys.*, vol. 103, pp. 1–15.

Demkowicz, L.; Oden, J. T. (1986): An adaptive characteristic petrov–galerkin finite element method for convection–dominated linear and nonlinear parabolic problems. *J. Comp. Phys.*, vol. 67, pp. 188–213.

Harlow, F. H. (1963): The particle–in–cell computing method for fluid dynamics. *Meth. Comp. Phys.*, vol. 3, pp. 319.

Hockney, R. W.; Eastwood, J. W. (1981): *Computer Simulation Using Particles*. McGraw–Hill.

Johnson, C. (1987): *Numerical solution of partial differential equations by the finite element method*. Cambridge University Press.

Lenardic, A.; Moresi, L.; Mühlhaus, H. (2000): The role of mobile belts for the longevity of deep cratonic lithosphere: the crumpled zone model. *Geophys. Res. Lett.*, vol. 27, pp. 1235–1238.

Sulsky, D.; Chen, Z.; Schreyer, H. L. (1994): A particle method for history–dependent materials. *Comput. Meth. Appl. Mech. Eng.*, vol. 118, pp. 179–196.

Sulsky, D.; Schreyer, H. L. (1996): Axisymmetric form of the material point method with applications to upsetting and Taylor impact problems. *Comput. Meth. Appl. Mech. Eng.*, vol. 139, pp. 409–429.

Sulsky, D.; Zhou, S.-J.; Schreyer, H. L. (1995): Application of a particle–in–cell method to solid mechanics. *Comp. Phys. Comm.*, vol. 87, pp. 236–252.

VonNeumann, J.; Richtmyer, R. D. (1950): A method for the numerical calculation of hydrodynamic shocks. *J. Appl. Phys.*, vol. 21, pp. 232.

Więckowski, Z.; Youn, S.-K.; Yeon, J.-H. (1999): A particle–in–cell solution to the silo discharging problem. *Int. J. Num. Meth. Eng.*, vol. 45, pp. 1203–1225.

York, A. R.; Sulsky, D.; Schreyer, H. L. (1999): The material point method for simulation of thin membranes. *Int. J. Num. Meth. Eng.*, vol. 44, pp. 1429–1456.

Zhou, S.; Stormont, J.; Chen, Z. (1999): Simulation of geomembrane response to settlement in landfills by using the material point method. *Int. J. Analyt. Meth. Geomech.*, vol. 23, pp. 1977–1994.