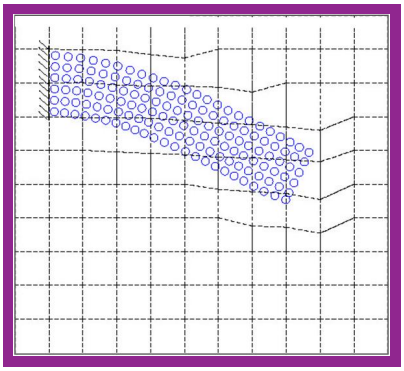
A Primer on the Material Point Method (MPM)

Rebecca Brannon

2013-09-03



ABSTRACT: This document is a beginner's primer on the Material Point Method (MPM), which can be viewed as an extension of an updated-Lagrangian finite-element method (FEM) to better accommodate very complicated geometries (like intricate porous micro-structures), as well as extremely large deformations (as seen in penetration) without requiring any remapping or advection schemes for a material model's internal variables. The MPM further provides, at no additional cost, an automatic no-slip/no-stick material contact at locations not known in advance (as in compression of a porous microstructure or sloshing of a fluid). This primer includes reviews of traditional linear FEM in 1-D, transient nonlinear FEM in 1-D and 2-D. Following each review of traditional FEM, the revisions in the algorithm required to solve the same problem using MPM are explained.

Chapter 1

A Primer on the Material Point Method (MPM)

1.1 The classic starting point: linear quasistatic 1-D bar problem

This section provides a "learn by example" introduction to the Material Point Method (MPM) by showing how to solve the classical quasi-static uniaxial bar equation,

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}x}\left[E(x)A(x)\frac{\mathrm{d}u(x)}{\mathrm{d}x}\right]+f(x)=0 && \text{on domain } \Omega \text{ defined by } 0<x<L \\ &\text{subject to Robin BCs:} && \alpha_0\mathcal{F}(0)+\beta_0 u(0)=\gamma_0 \\ &&\text{and} \quad & \alpha_L\mathcal{F}(L)+\beta_L u(L)=\gamma_L \end{aligned} \tag{1.1}$$

Here, $E(x)$ is the bar stiffness,[1] $A(x)$ is the cross-sectional area of the bar, $f(x)$ is the spatially varying distributed load on the bar[2], and $u(x)$ is the displacement. In the Robin boundary conditions,

$$\mathcal{F}(0) := -E(0)A(0)u'(0) \qquad \text{and} \qquad \mathcal{F}(L) := E(L)A(L)u'(L) \tag{1.2}$$

are the forces at the left and right ends of the bar, respectively (each defined to be positive when pointing to the right, which is why the first one is defined with a negative). The Robin parameters, α_0, β_0, γ_0, α_L, β_L, and γ_L, are user-prescribed constants. These conditions include the following special cases: prescribed displacement,[3] prescribed force,[4] and a spring BC.[5]

This section is broken into three subsections. First, the traditional FEM solution is provided, and then it is converted to an MPM formulation. This section finishes with a step-by-step MPM algorithm. Subsequent sections build up from this simple 1-D bar problem to allow material nonlinearity, first in 1-D and then in 2-D.

The strong form of the governing equation in Eq. (1.1) is multiplied by an arbitrary weight function[6] $w(x)$, and then integrated over the domain from $x = 0$ to $x = L$:

$$\int_0^L E(x)A(x)u''(x)w(x)\mathrm{d}x + \int_0^L f(x)w(x)\mathrm{d}x = 0 \qquad \forall w(x) \tag{1.3}$$

The first term is integrated by parts to give the so-called **weak form** of the governing equation:

$$E(x)A(x)w(x)u'(x)\big|_0^L - \int_0^L EAw'(x)u'(x)\mathrm{d}x + \int_0^L w(x)f(x)\mathrm{d}x = 0 \tag{1.4}$$

Using the definition of boundary forces defined in Eq. (1.2), this may be written

$$\int_0^L E(x)A(x)w'(x)u'(x)\mathrm{d}x = w(0)\mathcal{F}(0) + w(L)\mathcal{F}(L) + \int_0^L w(x)f(x)\mathrm{d}x = 0 \tag{1.5}$$

1.1.1 Traditional FEM formulation of the bar problem

A traditional FEM solution to the weak form of the linear bar problem introduces a set of nodes (which we assume are numbered sequentially from 1 to ν_n, where ν_n denotes the total number of nodes). The nodes are furthermore used to define elements on the finite-element mesh, as explained in any introductory FEM textbook. Below ν_e denotes the total number of elements.

[1] The modulus E is commonly thought to be equal to Young's modulus, which is true for uniaxial stress (where the lateral stress is zero), but it is actually a different modulus, called the constrained modulus, if the loading is uniaxial strain (where the lateral strain is zero)

[2] The distributed load has dimensions of force per unit length. Point loads are accommodated by allowing the distributed load to include Dirac delta contributions.

[3] found by setting the α parameter to zero, $\beta = 1$, and γ equal to the prescribed displacement

[4] by setting $\alpha = 1$, $\beta = 0$, and γ equal to the precribed force

[5] found by setting $\alpha = 1$, β equal to the negative of the spring constant

[6] also called the **trial function**.

The approximate FEM solution for the displacment field is

$$\tilde{u}(x) := \sum_{j=1}^{\nu_\mathrm{n}} \tilde{u}_j N_j(x) \tag{1.6}$$

where $\{N_1(x), N_2(x), \ldots, N_{\nu_\mathrm{n}}\}$ are the nodal basis functions (such as the "tent" functions constructed from linear shape functions), and $\{\tilde{u}_1, \tilde{u}_1, \ldots, \tilde{u}_{\nu_\mathrm{n}}\}$ are the nodal displacements.[7]

The weight function in the weak formulation is similarly expanded as

$$w(x) := \sum_{i=1}^{\nu_\mathrm{n}} w_i N_i(x) \tag{1.7}$$

In the finite-element method, the nodal basis functions are required to satisfy the following properties:

Kronecker property: $$N_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{1.8a}$$

Partition of unity: $$\sum_{i=1}^{\nu_\mathrm{n}} N_i(x) = 1 \qquad \forall x \tag{1.8b}$$

Linear completeness: $$\sum_{i=1}^{\nu_\mathrm{n}} x_i N_i(x) = x \qquad \forall x \tag{1.8c}$$

FEM compact support: $N_i(x) = 0$ $\forall x$ falling in an element not including node i (1.8d)

Weak-form integrability: $N_i(x)$ must be continuous (1.8e)

The reason for the last condition is explained in introductory FEM textbooks.

Let $\hat{\boldsymbol{N}}(x)$ denote an $\nu_\mathrm{n} \times 1$ column array of the nodal basis functions. For future use, let $\hat{\boldsymbol{G}}$ denot the array of spatial gradients of the nodal basis functions. Similarly, let $\hat{\boldsymbol{u}}$ and $\hat{\boldsymbol{w}}$ denote $\nu_\mathrm{n} \times 1$ column arrays of nodal displacements and nodal weights, respectively:

$$\hat{\boldsymbol{N}}(x) := \begin{bmatrix} N_1(x) \\ N_2(x) \\ \vdots \\ N_{\nu_\mathrm{n}}(x) \end{bmatrix} \qquad \hat{\boldsymbol{G}}(x) := \begin{bmatrix} N_1'(x) \\ N_2'(x) \\ \vdots \\ N_{\nu_\mathrm{n}}'(x) \end{bmatrix} \qquad \hat{\boldsymbol{u}} := \begin{bmatrix} \tilde{u}_1 \\ \tilde{u}_2 \\ \vdots \\ \tilde{u}_{\nu_\mathrm{n}} \end{bmatrix} \qquad \hat{\boldsymbol{w}} := \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{\nu_\mathrm{n}} \end{bmatrix} \tag{1.9}$$

Then Eqs. (1.6) and (1.7) may be written in matrix form as[8]

$$\tilde{u}(x) = \underbrace{\underbrace{\hat{\boldsymbol{N}}^T(x)}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{u}}}_{\nu_\mathrm{n}\times 1}}_{1\times 1} \qquad \text{and} \qquad w(x) = \underbrace{\underbrace{\hat{\boldsymbol{w}}^T}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{N}}(x)}_{\nu_\mathrm{n}\times 1}}_{1\times 1} \tag{1.10}$$

The superscript "T" denotes the transpose, which turns a column array into a row array and hence reverses the matrix dimensions as indicated. The end result is a 1×1 matrix, which is simply a single number. Let us also define a "reaction array" $\hat{\boldsymbol{\mathcal{F}}}$ having all zero components except in the first and last components as

[7] The reader is presumed to be familiar enough with the finite-element method that these statements require no detailed explanations clarification.

[8] Commutativity of scalar multiplication allows us to put the nodal basis array either on the left (with a transpose) or on the right (without transpose) as done here. In anticipation of upcoming parts of the analysis, we find it convenient to have the displacement array placed on the right with the weight array on the left.

follows:

$$\hat{\boldsymbol{\mathcal{F}}} := \begin{bmatrix} \mathcal{F}(0) \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \mathcal{F}(L) \end{bmatrix} \qquad \text{which allows writing} \qquad w(0)\mathcal{F}(0) + w(L)\mathcal{F}(L) = \hat{\boldsymbol{w}}^T \hat{\boldsymbol{\mathcal{F}}} \tag{1.11}$$

Making these substitutions into Eq. (1.5) gives

$$\int_0^L E(x)A(x) \underbrace{\underbrace{\underbrace{\hat{\boldsymbol{w}}^T}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{G}}(x)}_{\nu_\mathrm{n}\times 1}}_{\nu_\mathrm{n}\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{G}}^T(x)}_{1\times\nu_\mathrm{n}}}_{1\times 1} \underbrace{\hat{\boldsymbol{u}}}_{\nu_\mathrm{n}\times 1} \mathrm{d}x = \underbrace{\underbrace{\hat{\boldsymbol{w}}^T}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{\mathcal{F}}}}_{\nu_\mathrm{n}\times 1}}_{1\times 1} + \int_0^L \underbrace{\underbrace{\hat{\boldsymbol{w}}^T}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{N}}(x)}_{\nu_\mathrm{n}\times 1}}_{1\times 1} f(x)\mathrm{d}x \qquad \forall \hat{\boldsymbol{w}} \tag{1.12}$$

where (recall) $\hat{\boldsymbol{G}}(x)$ denotes the array of shape function gradients,

Since this must hold $\forall \hat{\boldsymbol{w}}$, the premultiplication by $\hat{\boldsymbol{w}}^T$ may be removed. Also, recognizing that $\hat{\boldsymbol{u}}$ does not vary with x, it may be removed from the first integral, giving

$$\hat{\hat{\boldsymbol{K}}}\hat{\boldsymbol{u}} = \hat{\boldsymbol{\mathcal{F}}} + \hat{\boldsymbol{f}} \tag{1.13}$$

where

$$\textbf{Global Stiffness:} \quad \underbrace{\hat{\hat{\boldsymbol{K}}}}_{\nu_\mathrm{n}\times\nu_\mathrm{n}} := \int_0^L E(x)A(x) \underbrace{\underbrace{\hat{\boldsymbol{G}}(x)}_{\nu_\mathrm{n}\times 1} \underbrace{\hat{\boldsymbol{G}}^T(x)}_{1\times\nu_\mathrm{n}}}_{\nu_\mathrm{n}\times\nu_\mathrm{n}} \mathrm{d}x \tag{1.14}$$

and

$$\textbf{Body force array:} \quad \underbrace{\hat{\boldsymbol{f}}}_{\nu_\mathrm{n}\times 1} := \int_0^L f(x) \underbrace{\hat{\boldsymbol{N}}(x)}_{\nu_\mathrm{n}\times 1} \mathrm{d}x \tag{1.15}$$

Counting unknowns and equations to confirm solvability

Equation (1.13) represents a set of ν_n equations involving $\nu_\mathrm{n} + 2$ unknowns:

$$\textbf{A total of } \nu_\mathrm{n} + 2 \textbf{ unknowns}: \qquad u_1, u_2, \ldots, u_{\nu_\mathrm{n}}, \quad \mathcal{F}(0), \mathcal{F}(L) \tag{1.16}$$

Not only are the nodal displacements unknown, so are the boundary forces, $\mathcal{F}(0)$ and $\mathcal{F}(L)$, located at the first and last components of $\hat{\boldsymbol{\mathcal{F}}}$. To achieve a solvable system, Eq. (1.13) must be supplemented with the two boundary conditions in Eq. (1.1). Namely, noting that $u(0) = u_1$ and $u(L) = u_{\nu_\mathrm{n}}$,

$$\begin{aligned} \alpha_0\mathcal{F}(0) + \beta_0 u_1 &= \gamma_0 \\ \alpha_L\mathcal{F}(L) + \beta_L u_{\nu_\mathrm{n}} &= \gamma_L \end{aligned} \tag{1.17}$$

Together, Eqs. (1.13) and (1.17) form a set of $\nu_\mathrm{n} + 2$ equations solvable for the $\nu_\mathrm{n} + 2$ unknowns! Specific methods of solution, covered in a first course on the finite-element method, are not discussed in this manuscript.

FEM integral evaluation

In the finite-element method, a large fraction of the work goes into evaluation of the integrals in Eqs. (1.14) and (1.15) . Since there are two integrals to be evaluated, let's discuss them generically by letting $\zeta(x)$ denote the integrand so that the task is to evaluate

$$\int_{\Omega} \zeta(x)\mathrm{d}x \tag{1.18}$$

where Ω represents the integration domain spanning from x=0 to x=L. For the stiffness integral, the integrand is $\zeta(x) = E(x)A(x)\hat{\boldsymbol{G}}(x)\hat{\boldsymbol{G}}^T(x)$. For the force integral, $\zeta = f(x)\hat{\boldsymbol{N}}(x)$.

In a conventional FEM code, each integral of the form in Eq. (1.18) is evaluated by breaking the integral over the entire domain Ω (defined by $0 < x < L$) into the sum of integrals over element domains:

$$\int_{\Omega} \zeta(x)\mathrm{d}x = \sum_{e=1}^{\nu_\mathrm{e}} \int_{\Omega_e} \zeta(x)\mathrm{d}x \tag{1.19}$$

Here, Ω_e represents the e^{th} element domain, and ν_e is the number of elements. Because finite elements fully tessellate the domain (i.e., they cover it without gaps or overlaps), this reformulation of the integral entails no error or loss of generality. Once the integral over the *entire* domain has been recast this way in terms of element integrals, a conventional FEM integrator will then employ Gauss integration to evaluated each element integral, thus ultimately giving

$$\int_{\Omega} \zeta(x)\mathrm{d}x = \sum_{e=1}^{\nu_\mathrm{e}} \int_{\Omega_e} \zeta(x)\mathrm{d}x \approx \sum_{e=1}^{\nu_\mathrm{e}} \sum_{g=1}^{\nu_\mathrm{g}(e)} \int_{\Omega_e} \zeta(x_{ge})\omega_{ge}\mathrm{d}x \tag{1.20}$$

where $\nu_\mathrm{g}(e)$ denotes the number of gauss points used on the e^{th} element, x_{ge} is the location of the g^{th} Gauss point on the e^{th} element, and ω_{ge} is the associated Gauss weight factor. As seen, this method of evaluating the FEM integrals requires function evaluations at a finite number of locations (the Gauss points). Therefore, the FEM code user does not have to write computer functions for the material, structural, and loading functions: stiffness $E(x)$, area $A(x)$, and body force $f(x)$. Instead, the FEM code user only needs to supply arrays containing values of these functions at the Gauss points!

Having reviewed the finite-element method for the simple 1-D bar equation, we are now in a position to convert the formulation to become an MPM method.

1.1.2 Converting the 1-D linear bar FEM code to an MPM code

Like an FEM formulation, the Material Point Method (MPM) solves the weak form of the governing equations on an overlaid "helper" grid. The grid has nodes and elements that play the same role in an MPM formulation as in an FEM formulation.[9] Accordingly, most of the equations in the preceding FEM section continue to apply for the MPM, with only minor exceptions as explained below. Recognizing this similarity of MPM to FEM can greatly facilitate revising an existing FEM code to include MPM as an option.

In the previously discussed traditional FEM formulation, all problem data[10] are specified at the Gauss points, which (by definition of what it means to be a Gauss point) must be placed at locations that have a specific connection to the element and node topology. For an MPM formulation, on the other hand, all problem data are saved at so-called so-called material points,[11] which, as illustrated in Fig. 1.1, are not required to have any connection whatsoever to the topology of the overlaid mesh.

One difficulty in describing MPM formulations is the introduction of many more countable entities in the algorithm. Like the FEM formulation, you still have a countable number of nodes on the grid. In the MPM, you also have a countable number of material points. To alleviate some notational clutter in upcoming equations, it is fairly common (*cf.,* [1]) for MPM formulations to adopt the convention that the letter of the alphabet used in indices (subscripts) defines the intended meaning as follows:

[9] In MPM formulations, an element is often alternatively referred to as a **cell**, but its meaning is still unchanged.
[10] namely, values of material, structural, and forcing functions: stiffness $E(x)$, area $A(x)$, and body force $f(x)$.
[11] called **markers** by some MPM researchers.

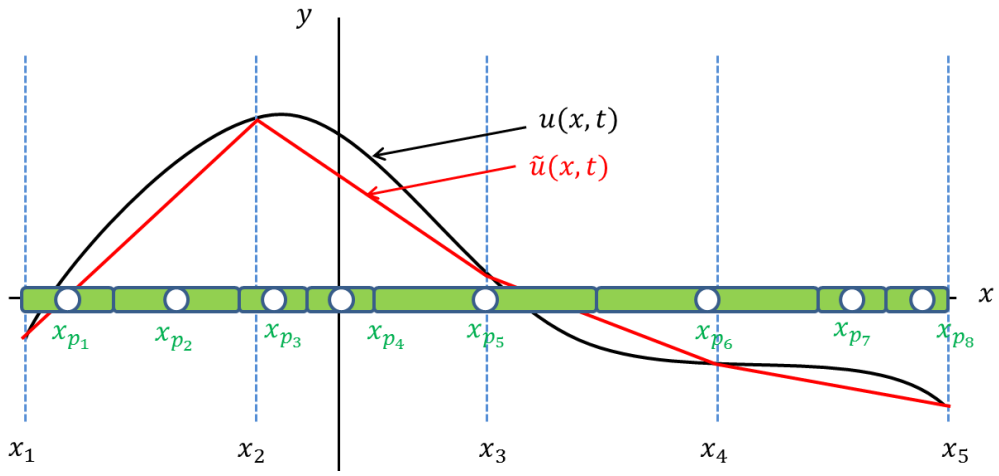


Figure 1.1: A depiction of MPM particles (hollow dots) and their associated domains (green pill boxes) on a 1-D domain. As shown here, the sizes of the MPM particle domains do not need to have any connection to the underlying topology of the grid (where the nodes on the mesh are located at the blue dashed lines). Here, the location of a particle is shown with a double subscript, but references to the p^{th} particle location will hereafter be written as x_p while the i^{th} grid node location is written x_i (*i.e.,* the subscript symbol p or i will indicate particle or node, respectively.

- Index i stands for a grid node, and $\sum_i$ refers to a sum over all grid nodes. For example, x_i refers to the location of the i^{th} grid node.
- Index p stands for an MPM particle, and $\sum_p$ refers to a sum over all MPM particles. For example, x_p refers to the location of the p^{th} grid node. This convention supercedes the double subscripting notation used in Fig. 1.1, which can always be reinstated (as needed) to refer to particular particles.[12]
- Index e stands for a grid element (sometimes called a "cell").
- Index c stands for an MPM particle "corner." The name **corner** is inherited from terminology of 2-D formulations, where a particle corner is a vertex of a polygon-shaped domain for the particle. In 1-D, each particle has only two corners, located at $x_p \pm r_p$, where r_p is the **particle radius** (equal to half of the particle length).[13] Also, the double-indexed position, x_{cp} refers to the c^{th} corner of the p^{th} particle. It is important to write this with the c written before the p, as we intend to later assign a different meaning to x_{pc}.
- The MPM formulation, derived below, will introduce a double-index quantity, ϕ_{ip}, equal to an average of the grid's i^{th} nodal basis function over the p^{th} particle domain.
- Similarly, the double-index quantity $\boldsymbol{G}_{ip}$ will be introduced below to refer to an average of the i^{th} nodal basis function's gradient over the p^{th} particle domain.

How is MPM different from FEM?

An existing FEM code for the linear bar problem can be revised to accommodate MPM as an option by going to any point in the FEM source code where spatial integrals are evaluated using the standard FEM-style integration given in Eq. (1.20), and there adding an "`if MPM`" branch to permit evaluation of the same integrals in a different way (using particle data as described below). After this MPM-integration branch,

[12] For example, the location of the 29^{th} particle would be denoted $x_{p_{29}}$ in order to distinguish it from the 29^{th} grid node location, which would be denoted $x_{i_{29}}$

[13] Some extensions to MPM might allow the particle location x_p to be located somewhere other than the particle centroid, but it is assumed that you can make appropriate revisions in that case.

you would then return to the existing FEM coding to finish solving for nodal displacements on the grid as usual.[14] Once nodal displacements are found on the grid, they are simply mapped to the particles by evaluating Eq. (1.6) at $x = x_p$. Alternative mappings from the grid to particles are discussed below.

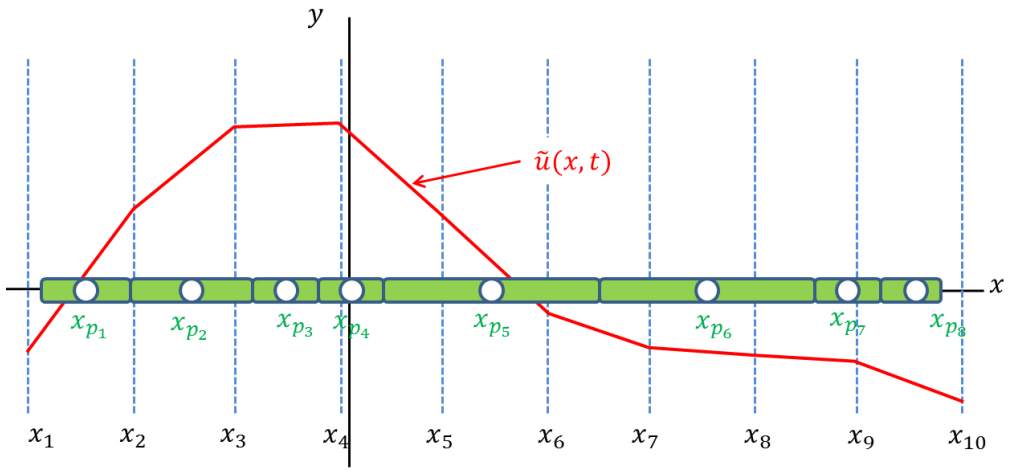


Figure 1.2: The same particle topology as in Fig. 1.1, but with a different overlaid grid. In this case, there is no grid node at the body's boundary. This additional complication with the Material Point Method (or, for that matter, any particle method that uses an overlaid grid) is avoidable for small-deformation problems where it is always possible to place overlay grid nodes at the boundary. Algorithmic adjustments for large displacements are deferred until later.

Any code that describes fields using standard FEM nodal basis functions, which means they satisfy all of the conditions in Eq. (1.8), is properly regarded to be a finite-element code even if it employs non-standard methods (like MPM) to evaluate the FEM nodal integrals. Later, we define a variant of MPM that actually redefines the nodal basis functions to adaptively match the underlying particle topology,[15] thus making such a method no longer properly a finite-element method.[16]

MPM-style evaluation of FEM-style nodal integrals

All of the equations previously given for the standard FEM solution to the bar equation continue to apply, except that the FEM-style summation over elements in Eq. (1.19) is replaced with a summation over particle domains:

$$\int_\Omega \zeta(x)\mathrm{d}x = \sum_p \int_{\Omega_p} \zeta(x)\mathrm{d}x \tag{1.21}$$

[14] Actually, you might also need an "`if MPM`" branch in the construction of the boundary force array $\hat{\mathcal{F}}$, depending on the MPM particle placements. When the physical boundary does not coincide with a grid boundary, as in Fig. 1.2, the construction of $\hat{\mathcal{F}}$ is a bit more complicated. In that case, it might have nonzero components at more than the first and last locations in the array, but it will still depend on $\mathcal{F}(0)$ and $\mathcal{F}(L)$, thus making no change in the solvability of the system of $\nu_\mathrm{n}+2$ equations for $\nu_\mathrm{n}+2$ unknowns, and only minor changes in the solution procedure itself. For the simple 1-D small-displacement problems, we presume for simplicity that the boundary of the physical domain coincides with grid nodes, as in Fig. 1.1, hence making the boundary force array the same as it was in the previous classical FEM formulation. Evaluation of the boundary term for large-displacement problems is discussed later.

[15] This method, discussed later, stretches the nodal basis functions in regions having elongated MPM particles; it even effectively deletes some of the nodal basis functions by setting them to zero, but the key is that partition of unity is nevertheless preserved.

[16] As explained later, these redefined nodal basis functions no longer impose Eq. (1.8a) or Eq. (1.8d), but they retain the properties of partition of unity and linear completeness as long as the particle domains tesselate the domain (*i.e.,* without gaps or overlaps). If the particles do not form a tesselation, but approach one as the number of particles is increased, then retention of partition of unity and linear completeness on the grid ensures convergence of the solution to the correct solution.

where (recall) Ω refers to the entire domain from x=0 to x=L, and Ω_p refers to particle domains, shown as pillboxes in Fig. 1.1.

Let V_p denote the "volume" of the p^{th} particle (which is simply the particle length in the 1-D bar example). Then, without loss in generality, the above equation may be written

$$\int_\Omega \zeta(x)\mathrm{d}x = \sum_p V_p \overline{\zeta_p} \tag{1.22}$$

where

$$\overline{\zeta_p} := \frac{1}{V_p}\int_{\Omega_p} \zeta(x)\mathrm{d}x \tag{1.23}$$

This shows that an MPM-style integration over Ω boils down to the need to find the average $\overline{\zeta_p}$ of the function $\zeta(x)$ over the p^{th} particle domain Ω_p. Variants of MPM are distinguished by how they compute such averages. As pointed out by Bardenhagen and Kober [1], with further generalization and clarification by Sadeghirad *et al.* [3], all commonly used MPM approximate integration methods can be unified as

GENERALIZED APPROXIMATE AVERAGE

$$\frac{1}{V_p}\int_{\Omega_p} \zeta(x)\mathrm{d}x \approx \frac{1}{V_p^*}\int_{\Omega_p^*} \zeta_p^*(x)\omega^*(x)\mathrm{d}x \qquad \text{where} \qquad V_p^* := \int_{\Omega_p^*} \omega^*(x)\mathrm{d}x \tag{1.24}$$

where $\omega^*(x)$ denotes a weighting function, the selection of which (among other things) distinguishes variants of MPM formulations, as discussed below. The **weighted particle volume** V_p^* is generally different from the *actual* particle volume V_p, and it is the actual volume that must still be used in Eq. (1.22). The generalized approximate average permits the integration domain Ω_p^* to be selected differently from the actual particle domain Ω_p. Choosing to change the integration domain is not particularly useful in 1-D problems, but we will see that it is almost essential in 2-D and 3-D problems. How can we "get away with" changing the integration domain? The key is that we are *finding averages*, which makes this revision very reasonable; specifically, discretization refinement corresponds to successively smaller particle domain sizes, making the average of any continuous field over an approximate (presumably more convenient) particle domain converge to the average over the actual domain, as long as all domains contain the material point about which the average is taken. Complications for discontinuous integrands are discussed below. Notice that the generalized average in Eq. (1.24) allowed a revised integrand $\zeta^*(x)$ to take the place of the actual integrand, $\zeta_p(x)$. To explain the motivation for (and sensibility of) this aspect of the generalized average, we will now discuss specific choices for the weight function, and other approximations used in this unified particle averaging formula.

Early MPM formulations treated the body as if it were made from point masses[4]. Bardenhagen and Kober [1] pointed out that this somewhat unsavory assumption is equivalent to the following selections in the generalized approximate average of Eq. (1.24):

$$\Omega_p^* = \Omega_p \tag{1.25a}$$

$$\zeta_p^*(x) = \zeta(x) \tag{1.25b}$$

$$\omega^*(x) = \delta(x - x_p) \tag{1.25c}$$

where $\delta(x - x_p)$ is the Dirac delta function centered at x_p. By properties of the Dirac delta (noting that $x_p \in \Omega_p$), these choices therefore give the following approximation for the particle domain average:

The single-point average: $$\frac{1}{V_p}\int_{\Omega_p} \zeta(x)\mathrm{d}x \approx \zeta(x_p) \tag{1.26}$$

In other words, the choices in Eq. (1.25) are equivalent to taking the average of $\zeta(x)$ over the particle domain to be approximately the integrand evaluated at the particle! This is equivalent to single-point Gauss

integration on the particle domain. It is also equivalent to the following alternative choices in the generalized average:

$$\Omega_p^* = \Omega_p \tag{1.27a}$$

$$\zeta_p^*(x) = \zeta(x_p) \qquad = \text{constant} \tag{1.27b}$$

$$\omega^*(x) = T_p^*(x) \tag{1.27c}$$

where $T_p^*(x)$ is the so-called **tophat** function defined by

$$T_p^*(x) := \begin{cases} 1 & \text{for } x \in \Omega_p^* \\ 0 & \text{for } x \notin \Omega_p^* \end{cases} \tag{1.28}$$

These choices, which correspond to replacing the spatially varying integrand $\zeta(x)$ by a function that is constant over the particle, give the same result as the Dirac choices in Eq. (1.25). The single-point average is a very good approximation if the integrand $\zeta(x)$ does not vary significantly over Ω_p, but we now explain that this is not generally the case for most MPM formulations. To judge the accuracy of such an approximation, we must recall the specific forms of the $\zeta(x)$ functions actually needed in the nodal grid integrals: Eq. (1.18),

$$\text{For the stiffness integral, } \zeta(x) = E(x)A(x)\hat{\boldsymbol{G}}(x)\hat{\boldsymbol{G}}^T(x) \tag{1.29a}$$

$$\text{For the force integral, } \zeta(x) = f(x)\hat{\boldsymbol{N}}(x) \tag{1.29b}$$

Deciding if the single-point average in Eq. (1.26) will give acceptably accurate results requires us to judge if these functions can be assumed to be approximately constant over the particle domain, at least in the limit as the discretization is refined. These considerations are as follows:

FORCE INTEGRAL, Eq. (1.29a): If linear shape functions are used, the nodal basis functions $\hat{\boldsymbol{N}}$ will be piecewise linear (tent functions). Then their gradients in $\hat{\boldsymbol{G}}$ will be discontinuous at grid cell boundaries. Accordingly, even though $E(x)$ and $A(x)$ might be reasonably approximated to be constant over a particle domain, such an approximation is unreasonable for $\hat{\boldsymbol{G}}$ if the particle domain is not contained entirely within a grid cell. For example, the single-point average is expected to be moderately accurate for the second particle in Fig. 1.1, because that particle is contained entirely within a grid cell. The error of the single-point average for the third particle in that figure is expected to be quite high because that particle straddles two grid cells, thus causing the integrand function $\zeta(x)$ to vary discontinously over the particle despite its small size. We call this a **cell-straddling error**.

FORCE INTEGRAL, Eq. (1.29b): As long as the applied distributed load $f(x)$ is continuous, Eq. (1.8e) ensures that the integrand in Eq. (1.29b) is also continuous. Accordingly, it would be increasingly accurate to assume that $\zeta(x)$ in that equation to be approximately constant over a small particle domain. In other words, the primary source of error with the Dirac averaging scheme arises in the stiffness integral, not the distributed force integral.

Eliminating the cell-straddling error

The cell-straddling error may be dramatically reduced by making the following choices for the generalized average:

$$\Omega_p^* = \Omega_p \tag{1.30a}$$

$$\text{For the stiffness integral, } \zeta_p^*(x) = E(x_p)A(x_p)\hat{\boldsymbol{G}}(x)\hat{\boldsymbol{G}}^T(x) \tag{1.30b}$$

$$\text{For the force integral, } \zeta_p^*(x) = f(x_p)\hat{\boldsymbol{N}}(x) \tag{1.30c}$$

$$\omega^*(x) = T_p^*(x) \tag{1.30d}$$

where $T_p^*(x)$ is the tophat function defined previously. Note that the approximate $\zeta_p^*(x)$ functions are the same as the exact $\zeta(x)$ functions except that the *physical* field functions[17] are treated as constant over each particle domain, given by their values at the particle. Substituting Eq. (1.27) into Eq. (1.24), and then using Eq. (1.22) in Eqs. (1.14) and (1.15) gives

APPROXIMATE MPM PARTICLE STIFFNESS

$$\hat{\hat{\boldsymbol{K}}} \approx \sum_p E(x_p)A(x_p) \int_{\Omega_p} \hat{\boldsymbol{G}}(x)\hat{\boldsymbol{G}}^T(x)\mathrm{d}x \tag{1.31}$$

and

APPROXIMATE MPM FORCE

$$\hat{\mathcal{F}} \approx \sum_p f(x_p) \int_{\Omega_p} \hat{\boldsymbol{N}}(x)\mathrm{d}x \tag{1.32}$$

In these forms, the remaining integrals involve only the nodal basis functions or their gradients, hence allowing these integrals to be evaluated exactly over each particle domain, thereby significantly increasing the accuracy of the result in comparison to the single-point method.

1.1.3 MPM Algorithm and data management for the 1-D bar

The following describes a data structure that I don't think is actually used in any MPM codes, but which I think ought to be considered in light of the fact that we want to move towards support of particles that stretch across multiple grid cells. This data scheme will, I think, work well with CPDI-style integral evaluations.

To revise an existing MPM linear bar code to accommodate an MPM option, you must create a new data array containing the locations and lengths of each material point. To fold the MPM option smoothly into an existing FEM code, you should anticipate the possibility of material point domains stretching across arbitrarily large expanses of grid cells. Accordingly, you should bypass the standard FEM element loop, and instead perform a loop directly over particles. You will need to create a new ragged array, let's call it $\hat{\hat{\boldsymbol{Y}}}$, for which Y_{pn} contains the ID of the n^{th} node having a support domain intersecting the approximate particle domain Ω_p^*. If, for example, we choose $\Omega_p^* = \Omega_p$ and if we use standard linear shape functions, then the ragged array corresponding to Fig. 1.1 would be

$$\hat{\hat{\boldsymbol{Y}}} = \begin{bmatrix} 1 & 2 & \\ 1 & 2 & \\ 1 & 2 & 3 \\ 2 & 3 & \\ 2 & 3 & 4 \\ 3 & 4 & 5 \\ 4 & 5 & \\ 4 & 5 & \end{bmatrix} \tag{1.33}$$

Here, for example, the third row corresponds to the third particle, and that particle has a domain partily in the first element and partly in the second element. Since the nodes for these element are numbered 1, 2, and 3, those are the nodal values associated with (*i.e.,* affected by) this particle. The fourth particle, on the other hand resides entirely within the second element. Thus, since this element is bounded by nodes 2 and 3, these are the numbers that go into the forth row of the above matrix. Similarly, the list corresponding to

[17] namely, stiffness $E(x)$, area $A(x)$, and distributed load $f(x)$

Fig. 1.2 would be

$$\hat{\hat{\boldsymbol{Y}}} = \begin{bmatrix} 1 & 2 & & \\ 1 & 2 & 3 & 4 \\ 3 & 4 & & \\ 3 & 4 & 5 & \\ 4 & 5 & 6 & 7 \\ 6 & 7 & 8 & 9 \\ 8 & 9 & 10 & \\ 9 & 10 & & \end{bmatrix} \tag{1.34}$$

Need to finish this. Again, I'm not sure if I am describing the optimal data management scheme. We ought to think carefully about this, not allowing any existing MPM codes to prevent us from seeing the optimal structure, especially since we want particles to be able to stretch across multiple cells (which could cause significant parallelization problems if we aren't careful)

1.2 Reformulation of the linear bar problem to treat the constitutive model as a 'black box' in preparation to handle material nonlinearity

There are two forms of nonlinearity in realistic mechanics problems:

- **Geometric nonlinearity** arises from large motions of the body so that now there is a significant difference between the initial and deformed locations of a point in the body. When large motions are possible, one must reconsider the problem functions, such as the distributed load $f(x)$ to decide if x represents the initial or deformed location. For now, we will ignore geometric nonlinearity by assuming that displacements are small enough that we don't need to worry if x is the initial or deformed position – the two are approximately equal to each other. Neglecting geometric nonlinearity also frees us from having to consider complications such as the meaning of strain[18]
- **Material nonlinearity** arises when the stress is not a linear function of strain, or possibly not even a function of strain at all! In this case, Hooke's law ($\sigma = E\varepsilon$) no longer applies, and we must go back to the strong form of the governing equations to remove all remnants of Hooke's law. In subsequent sections, we will assume that the material constituive model is a "black box" function $\sigma(\varepsilon)$ giving the stress as some unknown, generally nonlinear function of strain.

1.3 Nonlinear small-deformation transient 1-D bar problem

For the 1-D bar problem with material nonlinearity, the governing equations are

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}x}\left[A(x)\sigma(x)\right] + f(x) = 0 && \text{on domain } \Omega \text{ defined by } 0 < x < L \\ &\sigma(x) = \xi(\varepsilon(x), x) && \text{subject to Robin BCs:} \\ &\varepsilon(x) = \frac{\mathrm{d}u(x)}{\mathrm{d}x} && \alpha_0 \mathcal{F}(0) + \beta_0 u(0) = \gamma_0 \\ & && \alpha_L \mathcal{F}(L) + \beta_L u(L) = \gamma_L \end{aligned} \tag{1.35}$$

Here, $\xi(\varepsilon, x)$ is a "blackbox" nonlinear elastic constitutive model giving stress from strain ε. Note that the linear bar problem discussed in the previous section is recovered by choosing $\xi(\varepsilon, x) = E(x)\varepsilon$. In the Robin boundary conditions,

$$\mathcal{F}(0) := -A(0)\sigma(0) \qquad \text{and} \qquad \mathcal{F}(L) := A(L)\sigma(0) \tag{1.36}$$

[18] Is it change in length divided by initial length or current length? This question is moot when there is negligible change in length.

As in the previous linear bar analysis, these forces are defined to be positive when pointing to the right, in the direction of increasing x, while positive stresses correspond to tension (thus accounting for the negative in the first force definition, since a tensile force on the left end of the bar would point to the left). All other variables in these equations are defined as they were in Eq. (1.1). In what follows, we progress through the same sort of analysis steps as in the linear case, making appropriate adjustments for material nonlinearity.

The strong form of the governing equation in Eq. (1.35) is multiplied by an arbitrary weight function $w(x)$, and then integrated over the domain from $x = 0$ to $x = L$:

$$\int_0^L \frac{\mathrm{d}}{\mathrm{d}x}\left[A(x)\sigma(x)\right] w(x)\mathrm{d}x + \int_0^L f(x)w(x)\mathrm{d}x = 0 \qquad \forall w(x) \tag{1.37}$$

The first term is integrated by parts to give the so-called **weak form** of the governing equation:

$$w(x)A(x)\sigma(x)\big|_0^L - \int_0^L w'(x)A(x)\sigma(x)\mathrm{d}x + \int_0^L w(x)f(x)\mathrm{d}x = 0 \qquad \forall w(x) \tag{1.38}$$

Using the definition of boundary forces defined in Eq. (1.2), this may be written

$$\int_0^L w'(x)A(x)\sigma(x)\mathrm{d}x = w(0)\mathcal{F}(0) + w(L)\mathcal{F}(L) + \int_0^L w(x)f(x)\mathrm{d}x = 0 \tag{1.39}$$

1.3.1 Traditional FEM formulation of the nonlinear bar problem

As in the linear case, the approximate FEM solution for the displacement field is

$$\tilde{u}(x) := \sum_{j=1}^{\nu_\mathrm{n}} \tilde{u}_j N_j(x) \tag{1.40}$$

and the weight function is still expanded as it was in the linear case,

$$w(x) := \sum_{i=1}^{\nu_\mathrm{n}} w_i N_i(x) \tag{1.41}$$

In the nonlinear finite-element method, the nodal basis functions are still required to satisfy the properties listed in Eq. (1.8). The arrays $\hat{\boldsymbol{N}}(x)$, $\hat{\boldsymbol{G}}(x)$, $\hat{\boldsymbol{u}}$, $\hat{\boldsymbol{w}}$, and $\hat{\boldsymbol{\mathcal{F}}}$ are defined the same as they were in the linear case. Accordingly, the array form of Eq. (1.42) becomes

$$\int_0^L \underbrace{\hat{\boldsymbol{w}}^T}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{G}}(x)}_{\nu_\mathrm{n}\times 1} A(x)\sigma(x)\mathrm{d}x = \underbrace{\underbrace{\hat{\boldsymbol{w}}^T}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{\mathcal{F}}}}_{\nu_\mathrm{n}\times 1}}_{1\times 1} + \int_0^L \underbrace{\underbrace{\hat{\boldsymbol{w}}^T}_{1\times\nu_\mathrm{n}} \underbrace{\hat{\boldsymbol{N}}(x)}_{\nu_\mathrm{n}\times 1}}_{1\times 1} f(x)\mathrm{d}x \qquad \forall \hat{\boldsymbol{w}} \tag{1.42}$$

where (recall) $\hat{\boldsymbol{G}}(x)$ denotes the array of shape function gradients. Since this must hold $\forall \hat{\boldsymbol{w}}$, the premultiplication by $\hat{\boldsymbol{w}}^T$ may be removed, giving

$$\boxed{\hat{\boldsymbol{f}}^\mathrm{int} + \hat{\boldsymbol{f}}^\mathrm{ext} = \hat{\boldsymbol{0}}} \tag{1.43}$$

where

$$\boxed{\textbf{Internal force:} \quad \hat{\boldsymbol{f}}^\mathrm{int} := -\int_0^L \hat{\boldsymbol{G}}(x)A(x)\sigma(x)\mathrm{d}x} \tag{1.44}$$

and

$$\textbf{External force:} \quad \hat{\boldsymbol{f}}^{\text{ext}} := \hat{\boldsymbol{\mathcal{F}}} + \int_0^L \hat{\boldsymbol{N}}(x) f(x) \mathrm{d}x \tag{1.45}$$

These equations state that the sum of all forces must be zero. In particular, the forces applied by external agents (the distributed body force and the boundary forces) must be balanced exactly by the body's internal resistance to those forces.

Counting unknowns and equations to confirm solvability

The internal force involves an unknown stress field, $\sigma(x)$. Therefore, all components of the internal force array are unknowns. These internal forces would become known if the values of their integrands were known at Gauss points. Thus, stresses at Gauss points become part of the list of unknowns. Stresses at Gauss points may be found from strain at Gauss points via the constitutive law, making strain at Gauss points added to the list of unknowns. Strain at Gauss points are defined to equal the displacement gradient at the Gauss points, which is found if we know the nodal displacements. Thus nodal displacements must be added to the list of unknowns. As was the case in the linear problem, the boundary forces, $\mathcal{F}(0)$ and $\mathcal{F}(L)$, are unknown.

Overall, the counting of equations and unknowns for the nonlinear bar problem is summarized as follows.

- NUMBER OF EQUATIONS: $2\nu_{\text{n}} + 2 + 2\nu_{\text{G}}$
  (a) ν_{n} linear equations from force balance: $\hat{\boldsymbol{f}}^{\text{int}} + \hat{\boldsymbol{f}}^{\text{ext}} = \hat{\boldsymbol{0}}$
  (b) ν_{n} linear equations from Gauss evaluation of the $\hat{\boldsymbol{f}}^{\text{int}}$ integrals as linear combinations of the stresses at Gauss points.
  (c) 2 linear equations from boundary conditions: $\alpha_0 \mathcal{F}(0) + \beta_0 u_1 = \gamma_0$ and $\alpha_L \mathcal{F}(L) + \beta_L u_{\nu_{\text{n}}} = \gamma_L$
  (d) ν_{G} nonlinear equations from stresses at Gauss points determined from the strains at the Gauss points, via the the constitutive (material) model: $\sigma(x_g) = \xi(\varepsilon_g, x_g)$ where subscript g ranges over the number of Gauss points and x_g refers to the location of the g^{th} Gauss point.
  (e) ν_{G} linear equations from the definition of strain at a Gauss point: $\varepsilon(x_g) = u'(x_g) = \hat{\boldsymbol{u}}^T \hat{\boldsymbol{G}}(x_g)$
- NUMBER OF UNKNOWNS: $2\nu_{\text{n}} + 2 + 2\nu_{\text{G}}$
  (a) ν_{n} Internal forces: $f_1^{\text{int}}, \ldots, f_{\nu_{\text{n}}}^{\text{int}}$
  (b) 2 Boundary forces: $\mathcal{F}(0), \mathcal{F}(L)$
  (c) ν_{G} stresses at Gauss points: $\sigma_1, \ldots, \sigma_{\nu_{\text{G}}}$
  (d) ν_{G} strains at Gauss points: $\varepsilon_1, \ldots, \varepsilon_{\nu_{\text{G}}}$
  (e) ν_{n} nodal displacements: $u_1, \ldots, u_{\nu_{\text{n}}}$

The count of equations equals the count of unknowns, making a solution potentially possible. In this list, the only nonlinear set of equations comes from the constitutive model, but nonlinearity anywhere requires the governing system of equations to be solved iteratively, which is the subject of a numerical analysis class. Our goal of demonstrating solvability is accomplished.

1.3.2 Traditional FEM formulation for the transient nonlinear 1-D bar

1.3.3 Converting the 1-D transient nonlinear FEM solver to an MPM solver

1.3.4 MPM Algorithm for the 1-D transient nonlinear bar

Steve, I didn't get to this section yet, but the previous sections should address many of your questions. The basic algorithm for transient FEM is...

Governing equation:

$$\boxed{\hat{\boldsymbol{f}}^{\text{int}} + \hat{\boldsymbol{f}}^{\text{ext}} = \hat{\hat{\boldsymbol{M}}}\hat{\boldsymbol{a}}} \tag{1.46}$$

where $\hat{\boldsymbol{a}}$ is the nodal acceleration, the internal and external force vectors are defined the same as in the previous section except, for transient dynamics, they are *known* at the beginning of the timestep from the initial conditions. The consistent mass matrix is

$$\hat{\hat{\boldsymbol{M}}} = \int_0^L \rho(x)\hat{\boldsymbol{N}}\hat{\boldsymbol{N}}^T \mathrm{d}x \qquad \text{That is, } M_{ij} = \int_0^L \rho(x) N_i(x)\hat{\boldsymbol{N}}_j(x)\mathrm{d}x \tag{1.47}$$

A lumped mass matrix is defined to be a diagonal mass matrix with the i^{th} diagonal component defined to be the i^{th} row sum, so that

$$m_i = \int_0^L \rho(x) N_i(x)\mathrm{d}x \tag{1.48}$$

When the mass matrix is lumped (and hence diagonal), its inverse is trivial. Thus, recalling that both internal and external nodal forces are known at the beginning of a time step, solving the force balance gives nodal accelerations $\hat{\boldsymbol{a}}^0$ at the 0^{th} time step.

In general, given $\hat{\boldsymbol{a}}^n$ at the beginning of a generic n^{th} time step, and given velocity $\hat{\boldsymbol{v}}^n$ at the beginning of the step, the velocity at the end of the step is, with explicit time integration,

$$\hat{\boldsymbol{v}}^{n+1} = \hat{\boldsymbol{v}}^n + \hat{\boldsymbol{a}}^n \Delta t \tag{1.49}$$

and position at the end of the step is, by Taylor series,

$$\hat{\boldsymbol{x}}^{n+1} = \hat{\boldsymbol{x}}^n + \hat{\boldsymbol{v}}^n \Delta t + \frac{1}{2}\hat{\boldsymbol{a}}^n \Delta t^2 \tag{1.50}$$

With this velocity and position (and hence displacement) updated to the end of the step, you can compute the velocity gradient (which is the strain rate in 1-D problems) or the displacement gradient (which is the strain in 1-D problems). One or both of these is typically required by the constitutive model to update the stress $\sigma(x,t)$ at a Gauss point (or at an MPM particle) to the end of the time step, thus setting all initial conditions to start the cycle over to update the state through time to the next step.

WARNING: with this sort of explicit time integrator, the timestep must be set to a value about 1/10 as large as the amount of time it takes an acoustic wave to traverse a grid cell. For small-deformation linear elasticity, acoustic waves travel at a speed given by $\sqrt{E/\rho}$.

1.4 Nonlinear transient 2-D bar problem

1.4.1 Traditional FEM formulation for the transient nonlinear 2-D bar

1.4.2 Converting the 2-D transient nonlinear FEM solver to an MPM solver

1.4.3 MPM Algorithm for the 2-D transient nonlinear bar

1.5 Intrinsic no-interpenetration bonus attribute of MPM

Bibliography

[1] S.G. Bardenhagen and E.M. Kober. The generalized interpolation material point method. *CMES - Computer Modeling in Engineering and Sciences*, 5:477–495, 2004.

[2] A.R. Khoei and K. Karimi. An enriched-fem model for simulation of localization phenomenon in cosserat continuum theory. *Computational Materials Science*, 44(2):733 – 749, 2008. Has graphic QuadAndTriangleMesh.png (need permission).

[3] A. Sadeghirad, R.M. Brannon, and J.Guilkey. Second-order convected particle domain interpolation (cpdi2) with enrichment for weak discontinuities at material interfaces. *International journal for Numerical Methods in Engineering*, 2012.

[4] D. Sulsky, A. Chen, and H.L. Schreyer. A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118:179–196, 1994.