Stability analysis of explicit MPM

Song Bai and Craig Schroeder†

†University of California, Riverside

Abstract

In this paper we analyze the stability of the explicit material point method (MPM). We focus on PIC, APIC, and CPIC transfers using quadratic and cubic splines in two and three dimensions. We perform a fully three-dimensional Von Neumann stability analysis to study the behavior within the bulk of a material. This reveals the relationship between the sound speed, CFL number, and actual time step restriction and its dependence on discretization options. We note that boundaries are generally less stable than the interior, with stable time steps generally decreasing until the limit when particles become isolated. We then analyze the stability of a single particle to derive a novel time step restriction that stabilizes simulations at their boundaries. Finally, we show that for explicit MPM with APIC or CPIC transfers, there are pathological cases where growth is observed at arbitrarily small time step sizes. While these cases do not necessarily pose a problem for practical usage, they do suggest that a guarantee of stability may be theoretically impossible and that necessary but not sufficient time step restrictions may be a necessary and practical compromise.

CCS Concepts

• Computing methodologies → Physical simulation;

1. Introduction

The material point method (MPM) was introduced into computer graphics for simulation snow [SSC⁺13]. Since then, MPM has grown significantly in popularity as a method for handling a wide variety of complex phenomena. These materials often involve plasticity, which is challenging to handle implicitly [YSB⁺15,KGP⁺16,TGK⁺17,LLJ22]. This has created interest in explicit MPM formulations, which handle plasticity easily and are quite effective when stiffness is not too high.

Implicit methods are generally favored in graphics due to their stability [BW98]. Unlike explicit methods, which often require small time steps sizes to maintain stability, implicit methods tend to be stable at very large time step sizes and are often run at one (or a few) time steps per frame, though smaller time steps may be required to resolve collisions or improve accuracy. For explicit methods, time steps must be chosen based on stability considerations. Nevertheless, explicit MPM formulations are widely used (and are
the norm outside graphics) since they greatly simply the treatment of plasticity, which is difficult to handle accurately and efficiently with implicit methods. In practice, one often selects a fixed time step size by trial and error that is sufficient to ensure stability for a particular simulation [YSB\textsuperscript{*} 15, KGP\textsuperscript{*} 16]. More recently, there has been interest in using adaptive time step sizes based on the classical CFL restriction for explicit MPM [FHHJ18, SSS20]. The classical CFL restriction requires that the numerical wave speed must be at least as high as the physical wave speed. Although this is not a guarantee of stability, it is necessary [Str04]. A simulation that does not follow its CFL restriction can be expected to explode.

Stable time step selection is a fundamental challenge for numerical simulation, especially MPM [Bra16], and many tools have been developed to analyze stability for entire classes of numerical methods. Von Neumann analysis is the gold standard for proving stability for linear finite difference schemes. It is straightforward to use in this context and provides a guarantee of stability under some circumstances [Str04]. Although Von Neumann analysis assumes a linear numerical scheme and a linear partial differential equation, it can be adapted to nonlinear problems by linearizing about an equilibrium configuration. Indeed, Von Neumann analysis has been applied to MPM a few times, but this has been limited to 1D [NZ20, Gri14a], though [NZ20] does suggest some ways to apply the results to higher dimensions. In this paper, we perform a fully three-dimensional Von Neumann stability analysis on MPM near the rest configuration. This analysis includes the full algorithm with APIC [JSS\textsuperscript{*} 15] or CPIC [HFG\textsuperscript{*} 18] transfers, which has never been done in any dimension. This analysis justifies the use of sound-speed-based time step restrictions and provides insight on how details of the discretization affect the CFL numbers that can be used.

Transfers and averaging between particles and grid have a generally stabilizing effect on MPM. On the other hand, reduced numbers of neighboring particles can often reduce stability. Indeed, MPM simulations are often observed to explode from the boundary rather than the interior. We investigate the effects of boundaries on MPM stability, which naturally leads to the limiting case where a particle becomes isolated. Indeed, a simulation with an isolated particle is actually less stable than a large system of particles. Single particle stability was a special consideration in the development of APIC [JSS\textsuperscript{*} 15]. It was also noted in [SSS20] that fluid simulations could become unstable if particles became isolated. They proposed a time step restriction to avoid the single particle instability in the fluid case, noting that the instability also affects solids. In this paper, we take a closer look at the single particle instability and propose a time step restriction in the solid case.

One of the big surprises we encountered during the analysis in this paper was the destabilizing effects of variable time step sizes on the stability of isolated particles when APIC or CPIC transfers are used. Conventional wisdom suggests the existence of a time restriction below which time step sizes may be chosen at will; this is a fundamental assumption that underlies adaptive time step sizes and time step restrictions. We show that this might not be true for MPM under some circumstances.

2. Related work

The material point method was developed as an extension of early hybrid particle/grid methods [Har64] to handle elastic solids [SCS94, SZS95]. Early hybrid methods were quickly found to suffer from a variety of instabilities, such as ringing [Oku72, Bra88, BL98, BK04, SKB08, Gri14a]. This lead to studies into the stability properties of particle-grid transfers, which lead to the development of better interpolation functions [BK04, WG08]. Although the original PIC was effectively implicit, being a projection-style fluid solver, early MPM methods were all explicit. Implicit formulations of MPM were developed to improve stability and take larger time steps [GW01, GW03, SK04, LS06]. Despite these advantages, explicit methods are very popular in the MPM community due to their simplicity, accuracy, and the need to take small time steps anyway for many applications.
PMF was introduced to the computer graphics community by [SSC*13] as a method for simulating snow. Within the community, PMF has grown significantly in popularity. It is used for phase change [SSJ*14], foams and gels [YSB*15, RGI*15], dry sand [DBD16, KGP*16], wet sand [TGK*17, GPH*18], coupling with rigid bodies [HFG*18], fracture [WFL*19, WDG*19], anisotropic materials [JYM*20, CLR*20], and other complex materials [FLG19].

Early experiences within the community quickly discovered limitations when trying to apply the method to graphics applications, which has lead to efforts within the community to improve upon PMF, such as less noisy and less dissipative transfers [JSS*15, FGG*17], and better time integration [JST17, GSS*15, WLF*20]. Improvements in the method have in turn lead to an expansion in its adoption.

Due to the need for efficiency and stability, PMF was introduced into graphics as an implicit method [SSC*13], and most methods within graphics since have been implicit. However, implicit PMF is significantly more complex to implement [YSB13] than explicit methods within graphics since have been implicit. However, implicit schemes are not pursued here. We do not consider alternative transfer schemes such as GIMP [BK04, WG08, Wal09, NG15], but we will not pursue this here.

We consider both 2D and 3D. We also consider PIC and CPIC [HFG*18] transfers, which we are able to do without repeating too much of the analysis. In particular, PIC can be obtained by dropping the $C_p$ portions (or by setting $\xi = 0$). CPIC can be obtained by substituting $\nabla w_p^o \rightarrow \xi \nabla w_p^o(x_i^o - x_p^o)$ everywhere it occurs. We color code the portions that differ for PIC or CPIC to make the differences clear at a glance. We also summarize the conclusions separately for each. We do not consider alternative transfer schemes such as FLIP [BR86, BKR88, SCS94], PIC/FLIP blends [SSC*13], or XPIC [HN17], which would generally be expected to have radically different stability profiles.

3. Stability analysis

Our principal objective in this paper is to examine the stability characteristics of explicit PMF. For this purpose, we begin with the explicit APIC scheme shown in Algorithm 1, adopting the notation used in [JSS*15]. We restrict our analysis to the case of quadratic or cubic splines for the transfer weights. These are the most common choices in graphics and they have the advantage that $C_p^{j+1} = \sum_{i} w_p^o (x_i^o - x_p^o)(x_i^o - x_p^o)^T = \frac{1}{2} I$ takes a particularly simple form, where $\xi = \frac{4}{\Delta x}$ for quadratic and $\xi = \frac{3}{\Delta x}$ for cubic, leading to the simple update rule for $C_p^{j+1}$ used in Algorithm 1 (See [JSS*15] for details). The analysis could be extended to cover other transfer kernels (such as GIMP [BK04, WG08, Wal09, NG15]), but we will not pursue this here.

We consider both 2D and 3D. We also consider PIC and CPIC [HFG*18] transfers, which we are able to do without repeating too much of the analysis. In particular, PIC can be obtained by dropping the $C_p$ portions (or by setting $\xi = 0$). CPIC can be obtained by substituting $\nabla w_p^o \rightarrow \xi \nabla w_p^o(x_i^o - x_p^o)$ everywhere it occurs. We color code the portions that differ for PIC or CPIC to make the differences clear at a glance. We also summarize the conclusions separately for each. We do not consider alternative transfer schemes such as FLIP [BR86, BKR88, SCS94], PIC/FLIP blends [SSC*13], or XPIC [HN17], which would generally be expected to have radically different stability profiles.

3.1. Von Neumann analysis

Von Neumann analysis is the simplest approach for proving stability for linear finite difference schemes under periodic boundary conditions. The idea behind Von Neumann analysis is to take the Fourier transform of the discretized PDE. When this is done, different wave-numbers decouple from one another and grow or decay with different amplification factors $\gamma$. The scheme is stable if all of the amplification factors satisfy $|\gamma| \leq 1$. For simplicity, we ignore the complications surrounding $|\gamma| = 1$ throughout this paper.

**Basic ideas.** To get a general feel for how the procedure is applied in practice, we first apply it to the advection equation $u_t + c u_x = 0$ in 1D, which we discretize with forward-time-forward-space as $u_{i+1}^n - u_i^n + \frac{c}{\Delta x} (u_i^n - u_{i-1}^n) = 0$. We begin by fixing a wave-number $k$ and let $u_i^n = u(i\Delta x, n\Delta t) = g^n e^{ik\Delta x}$. Here, $i$ is the imaginary number when it does not occur in an index, and $g^n$ is a power of $g$. Substituting this into the discretized equation yields $g^{n+1} e^{ik\Delta x} - g^n e^{ik\Delta x} = 0$. Solving for $g$ gives us $g = 1 + \frac{ck}{\Delta x} (1 - e^{ik\Delta x})$. Let $v = \frac{\Delta x}{ck}$ so that $|g|^2 = 1 + 2v(v + 1)(1 - cos(k\Delta x))$. To have $|g| \leq 1$ for all $k$ we must have $v(v + 1) \leq 0$ so that $-1 \leq v \leq 0$. That is, the scheme is only stable if $c \leq \Delta x$ and $\Delta t \leq \frac{\Delta x}{c}$. This classical conclusion mirrors the CFL restriction. Here, $c$ is the advection velocity. Since the scheme only looks right (it uses $u_{i+1}^n$ but not $u_{i-1}^n$), it can only be stable if information moves right to left ($c \leq 0$). Since at each time step information travels only one grid node, the numerical wave speed is $\frac{c}{\Delta x}$, so that $\Delta x \leq c$ is the time step restriction.

**PMF Von Neumann setup.** In the simple example above, our state was a simple scalar $u_i^o$ that lives on a regular grid with per-

![Figure 3: Definitions of intermediates. Note that $b$, $w$, $b$, and $c$ lack grid indices. Since all grid cells are the same, these quantities do not explicitly depend on the grid index. $\bar{b}_q$ is the complex conjugate of $b_q$, so that $B$ is Hermitian.](image)
riors. The situation with MPM is far more complex. The PDE is nonlinear, and there is much more state, which lives on potentially moving particles. In order to apply Von Neumann analysis, we must assume a (quite restrictive) setup. (A1) All cells contain the same number of particles in the same arrangement. (A2) Periodic boundary conditions are used. (A3) All particles are identical; in particular, they have the same mass ($m_p = m$) and volume ($V_p = V_0$). (A4) The configuration is near the rest configuration, so that $u_p^0 \approx 0$ and positions can be considered to be stationary. (A5) The particle distribution is symmetric in the sense that for each grid node $i$ and particles $p$ there is a particle $\mathbf{p}$ such that $(\mathbf{x}_p^i - \mathbf{x}_q^i) = -(\mathbf{x}_p^i - \mathbf{x}_q^i)$. Stated another way, if the grid is mirrored along the $x$, $y$, and $z$ directions, there will be particles in the same locations. This assumption will simplify parts of the analysis by making certain quantities real-valued.

We will use index $p$ to refer to a per-particle quantity ($u_p^n$, $w_p^n$). Since each cell has the same particle layout (by assumption (A1)), we introduce the index $q$ to refer to a particle within some canonical cell. This is useful for quantities that are not constant across particles but are the same for corresponding particles in different grid cells. Each $p$ has a unique $q$ associated with it, and each $q$ corresponds with a $p$ for each grid cell.

**Initial particle state.** The analysis begins by fixing a wavenumber $\mathbf{z}$ and setting the particle state to be of the appropriate form.

$$u_p^n = \varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{u}_p^n + \mathbf{C}_p \varepsilon^{\mathbf{z} z} \mathbf{F}_p^n = I + \varepsilon^n I \varepsilon^{\mathbf{z} z} \mathbf{x}_p.$$ 

Here, $\varepsilon^n$ represents a small perturbation from equilibrium. At equilibrium ($\varepsilon = 0$), the particles are stationary ($\mathbf{u}_p^n = 0$, $\mathbf{C}_p = 0$) and experience no strain ($\mathbf{F}_p^n = I$) (assumption (A4)). In the definitions above, we have also introduced $\mathbf{u}_p$, $\mathbf{C}_p$, and $\mathbf{F}_p$. These provide a magnitude and direction for the perturbation, which may be different within a cell. Note the use of the index $q$. As in the motivating example, $\varepsilon^n$ is a power of the growth factor $g$ and $i$ is the imaginary number when not used as an index.

During the course of the Fourier analysis, we will find it convenient to introduce many intermediate quantities in order to keep expressions manageable. The definitions of all of these intermediates are listed in Figure 3 for easy reference.

**P2G transfers.** We begin the time step by transferring particle state to the grid.

$$\mathbf{m}_p^n = \sum_{p} \mathbf{w}_p^n \mathbf{m}_p = \mathbf{w} \mathbf{m}$$

$$\mathbf{m}_p^n \mathbf{u}_p^n = \sum_{p} \mathbf{w}_p^n \mathbf{m}_p \mathbf{u}_p^n + \mathbf{C}_p \mathbf{u}_p^n$$

$$\mathbf{u}_p^n = \frac{\varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{u}_p^n}{\varepsilon} \sum_{p} \mathbf{w}_p^n \mathbf{u}_q + \mathbf{C}_p \mathbf{u}_p^n$$

Note that the grid velocity follows the same wave form as the state variables ($\varepsilon^{\mathbf{z} z}$ vs. $\varepsilon^{\mathbf{z} z}$); all of the quantities will take this form. The intermediate $\mathbf{b}$ provides the direction for perturbations to $\mathbf{u}_p^n$.

**Forces.** With transfers completed, we must propagate our perturbations through the force computations. We compute forces through the first Piola-Kirchhoff stress tensor $\mathbf{P}$, $\mathbf{M}$ is the unperturbed stress derivative (See Figure 3). $\mathbf{K}_q$ is the direction of stress perturbation. Neglecting higher order terms in $\varepsilon$ (so that we are effectively analyzing linear elasticity), the stress is

$$\mathbf{P}(\mathbf{F}_p^n) = \mathbf{P}(\mathbf{I} + \mathbf{M} : (\mathbf{F}_p^n - \mathbf{I}) = \varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{M} : (\mathbf{F}_q) = \varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{K}_q$$

We next compute grid forces from the stresses

$$\mathbf{f}_p^n = -\sum_{p} \mathbf{v}_p^n \mathbf{P}(\mathbf{F}_p^n) (\mathbf{F}_p^n)^T \nabla \mathbf{w}_{ip}$$

$$= -\sum_{p} \mathbf{v}_p^n \mathbf{P}(\mathbf{F}_p^n) (\mathbf{F}_p^n)^T \nabla \mathbf{w}_{ip}$$

$$= -\varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{K}_q \nabla \mathbf{w}_{ip} = -\varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{c}$$. 

**Grid and particle velocities.** We can now apply forces to the grid.

$$\mathbf{u}_p^{n+1} = \mathbf{u}_p^n + \Delta t (\mathbf{m}_p^n - 1) \mathbf{f}_p^n$$

$$= \frac{\varepsilon^n \varepsilon^{\mathbf{z} z}}{\varepsilon} \mathbf{u}_p^n - \Delta t (\mathbf{m}_p^n - 1) \varepsilon^{\mathbf{z} z} \mathbf{c} = \varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{u}_p^n$$

Transferring back to particles,

$$\mathbf{u}_p^{n+1} = \sum_{i} w_{ip} \mathbf{u}_q^{n+1} = \sum_{i} w_{ip} \varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{u}_q = \varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{u}_q$$

From our original assumption on the form for $\mathbf{u}_p^{n+1}$ we have $\mathbf{u}_p^{n+1} = \mathbf{u}_p \varepsilon^{\mathbf{z} z} \mathbf{u}_p$, so that $\varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{u}_q = \varepsilon^n \varepsilon^{\mathbf{z} z} \mathbf{u}_p$ and thus $\mathbf{u}_q = \mathbf{h} \mathbf{u}_p$. Thus, we see that the particle velocity directions $\mathbf{u}_q$ are aligned but may differ in scale.

**Affine state.** Next, we update our affine state $\mathbf{C}_p^{n+1}$ on particles.

$$\mathbf{C}_p^{n+1} = \varepsilon \sum_{i} \mathbf{w}_{ip} \mathbf{C}_q^{n+1} (\mathbf{x}_p - \mathbf{x}_q)$$

$$\mathbf{C}_p^{n+1} = \varepsilon \sum_{i} \mathbf{w}_{ip} \mathbf{C}_q^{n+1} (\mathbf{x}_p - \mathbf{x}_q) = \varepsilon \mathbf{u}_p^{n+1} \mathbf{C}_p \mathbf{u}_p$$

As with velocities, this must match our definition for $\mathbf{C}_p^{n+1} = \varepsilon \mathbf{C}_q^{n+1} \mathbf{C}_p \mathbf{u}_p^{n+1}$, which leads to $\mathbf{C}_p^{n+1} = \mathbf{C}_p \mathbf{u}_p^{n+1}$. With this simplified form for $\mathbf{C}_p$, we may simplify $\mathbf{b}$ as well.

$$\mathbf{b} = \sum_{i} \mathbf{w}_{ip} (\mathbf{u}_q + \mathbf{C}_q (\mathbf{x}_q - \mathbf{x}_p)) = \sum_{i} \mathbf{w}_{ip} \mathbf{u}_q$$

$$\mathbf{b} = \sum_{i} \mathbf{w}_{ip} \mathbf{u}_q$$

**Deformation gradient.** The final remaining step in the transfer from grid to particles is the deformation gradient.

$$\nabla \mathbf{u}_p^{n+1} = \sum_{ip} \mathbf{u}_p^{n+1} (\nabla \mathbf{w}_{ip})^T = \sum_{ip} \mathbf{w}_{ip} \varepsilon \mathbf{u}_p^{n+1} \mathbf{C}_p \mathbf{u}_p$$

Substituting $\mathbf{F}_p^{n+1}$ in the definition for $\mathbf{F}_p^{n}$ and neglecting higher order terms we have

$$\mathbf{F}_p^{n} = (I + \Delta \nabla \mathbf{u}_p^{n+1})^T = (I + \Delta \mathbf{w}_{ip} \mathbf{u}_p^{n+1} \mathbf{C}_p \mathbf{u}_p)^T$$
Since positions are assumed not to move appreciably (assumption (A4)), we have completed the time step. In the process of keeping the equations relatively short, we have defined many intermediate quantities, which we must now solve for to amplify the solution factor \( g \).

**Eliminating unknown vectors and matrices.** When we started, we assumed \( V^0 \), \( m \), and \( \Delta t \) were known. The grid and particle locations \( (x^p_i, x^q_j) \) were fixed in a regular arrangement, and the weights are known. We also fixed a wave-number \( \mathbf{z} \), which for now we assume is known. This allows us (in principle at least) to directly calculate \( h_q, e_q, b, k_q, \) and \( M \). The quantities \( b \) and \( F_q \) are expressed in terms of other quantities. The matrix and vector equations we have left are the definitions of \( K_q, \mathbf{c}, \mathbf{u} \). We can eliminate \( \mathbf{u} \) by introducing a new scalar \( \beta \) and substituting in \( \mathbf{b} \).

\[
\mathbf{u} = \frac{bnm - \Delta t V^0}{m(w-g)} \quad \mathbf{c} = \frac{\Delta t V^0}{m(b - w g)} \quad \mathbf{e} = \frac{\beta \mathbf{c}}{\beta \mathbf{c} + \mathbf{u}}
\]

Next, we eliminate \( F_q \) and \( K_q \) from \( \mathbf{c} \), dropping into indexing notation to deal with tensors.

\[
\mathbf{c} = \sum_p M \left( \mathbf{t} \mathbf{K}^T \mathbf{q} \right) \mathbf{w} p e^{i(s^q_j - x^q_j) \cdot z} = \sum_p \left( M \cdot (\mathbf{t} \mathbf{K}^T \mathbf{q}) \right) \mathbf{w} p e^{i(s^q_j - x^q_j) \cdot z}
\]

\[
c_i = \sum_p M_{ijkl} u_k \sum_p \mathbf{w} p e^{i(s^q_j - x^q_j) \cdot z} = \sum_p M_{ijkl} u_k \sum_p \mathbf{w} p e^{i(s^q_j - x^q_j) \cdot z}
\]

\[
c_i = \sum_p M_{ijkl} u_k B_{ij} = \mathbf{u} A \mathbf{c}
\]

We can now eliminate \( \mathbf{c} \) and \( \mathbf{u} \) from the formula for \( \mathbf{u} \) giving

\[
\mathbf{u} = \beta \mathbf{c} = \beta \mathbf{e} \mathbf{Au}
\]

We are left with the eigenvalue problem \( \mathbf{A} \mathbf{u} = \sigma \mathbf{u} \), where \( \mathbf{u} \) is an eigenvector with eigenvalue \( \sigma \). Note that all of the pieces for \( \mathbf{A} \) can be computed, so that \( \mathbf{A}, \sigma, \) and \( \mathbf{u} \) are now available. We also have \( \sigma \mathbf{e} = 1 \).

**Eliminating the scalars.** We have now computed all of the scalar and matrix quantities. We are however left with the intermediate scalars \( \tau \) and \( \beta \), which we must eliminate to get \( g \), which is ultimately what tells us whether we are stable. \( \sigma \) was computed by solving an eigenvalue problem and is assumed to be available. The remaining equations are

\[
\tau = \frac{\Delta g}{g - 1} \quad \beta = \frac{\Delta t V^0}{m(b - w g)} \quad \sigma \tau = 1.
\]

Eliminating \( \tau \) and \( \beta \) we get a quadratic for \( g \),

\[
g^2 + \left( \frac{\Delta t V^0}{m w} - \frac{b}{w} - 1 \right) g + \frac{b}{w} = g^2 + 2rg + s = 0.
\]

What remains is to examine the roots \( g \).

---

**Real or complex.** The quantities \( w, b, \xi, M, s, \) and \( c \) are always real. Under the symmetry assumption (A5), for each grid node \( i \) and each particle \( p \) there is a particle \( \mathbf{p} \) such that \( (x^p_i - x^i) = -(x^p_i - x^i) \). Note that \( w^p_i = w^p_i \) but \( \mathbf{w}^p_i \) is not. Then

\[
h_q = \sum_i w^p_i e^{i(s^q_j - x^q_j) \cdot z} = \sum_i w^p_i e^{-(s^q_j - x^q_j) \cdot z} = h_q.
\]

Similarly, we also have \( k_q = -h_q, \xi_q = -\xi_q \). From this we conclude that \( \mathbf{B} \) is real and thus symmetric. Then \( \mathbf{A} \) is also real and symmetric, so that its eigenvalues \( \sigma \) and eigenvectors \( \mathbf{u} \) are real. Note that \( \mathbf{A} \) is symmetric so that \( \sigma \) is real. Since the equilibrium configuration should be stable (a consequence of assumption (A4)), \( \sigma > 0 \). From this, \( r \) is also real.

**Stability.** If \( r^2 < s \) then \( g \) is complex and the roots are a complex conjugate pair whose product is \( g \sigma = |g|^2 = s \). One can show that \( s \leq 1 \), so that the complex case is always stable. We conclude that the limit of stability (where \( |g| = 1 \)) is reached when \( g \) is \( \pm 1 \). The case \( g = 1 \) leads to \( c = 0 \) or \( \Delta t = 0 \), which only requires a nonnegative time step. The case \( g = -1 \) yields our time step restriction

\[
c \leq \frac{2(s + 1)}{\sigma} \quad \Delta t \leq \frac{\nu w c}{V^0} \leq \frac{\nu w c}{\sigma} \frac{2(s + 1)}{\sigma},
\]

which must be true for all \( z \). The dependence on \( z \) is entirely within the second factor, where both \( s \) and \( \sigma \) depend on \( z \).

**Isotropic.** For an isotropic constitutive model near the rest configuration, \( M_{ijkl} = \lambda \delta_{ij} \delta_{kl} + \mu (\delta_{ik} \delta_{jl} + \delta_{il} \delta_{jk}) \). Then \( A = (\lambda + \mu) \mathbf{B} \) has the same eigenvectors as \( \mathbf{B} \). Since the trace is the sum of the eigenvalues, the eigenvalues of \( \mathbf{A} \) are bounded by \( \lambda + 2 \mu \) with equality in the rank-one case. This leads to the bound

\[
\Delta t \leq \frac{\nu w c}{\lambda + 2 \mu} \frac{2(s + 1)}{\sigma} = \frac{f \Delta t}{v}. \tag{1}
\]

The first factor is independent of \( z \), the number of particles, or their placement. It is in fact just inverse of the sound speed \( c \). The second term \( f \) is dependent only on \( z \), the splines, and the particle
Assumptions (A1) and (A2) are essential to generalize. We performed our analysis using the five rather favorable particle coverage, the same simulation setup is stable at a maximum velocity magnitude grew by a factor of 10. We used binary search to identify the largest stable CFL number for each simulation. The time step restriction based on Von Neumann stability analysis proposed in the more difficult case of solids. In this section, we derive a time step restriction for solids in the vicinity of the rest configuration. In [SSS20], an effective time step restriction for the single particle instability was proposed in the case of fluids. In that paper, no solution was proposed in the more difficult case of solids. In this context, this assumption is not as limiting as it might seem. Particles that are at risk of becoming isolated tend to be near the surface, where they would not be expected to experience significant strain.

Our strategy for deriving a time step restriction is to consider the stability of perturbations from the rest configuration. The momentum of an isolated particle is constant in the absence of outside forces, so that its velocity does not change. Ignoring grid-related effects, the velocity of the particle is decoupled from the rest of the dynamics. In practice, the location of a particle relative to the grid can affect its stability, with some placements within a cell requiring a smaller time step for stability. A particle traveling through the grid would experience a more complex trajectory. To handle this, we evaluate the stability of a stationary particle at an arbitrary location within a cell and then select a time step size that is stable for all such locations.

**Stability formulation.** At the beginning of each time step, the isolated particle has state \( m_p, \mathbf{x}_p, \mathbf{v}_p, \mathbf{C}_p, \) and \( \mathbf{F}_p \). The mass \( m_p \) does not change. Since we are assuming that the particle is stationary, \( \mathbf{v}_p = 0 \) and \( \mathbf{x}_p \) is constant. We assume a small perturbation from the rest configuration, so that \( \mathbf{C}_p = \epsilon \mathbf{A}^0 \) and \( \mathbf{F}_p = \mathbf{I} + \epsilon \mathbf{E}^0 \). After the time step, we will have a new state \( \mathbf{C}_{p+1} = \epsilon \mathbf{A}^{n+1} \) and \( \mathbf{F}_{p+1} = \mathbf{I} + \epsilon \mathbf{E}^{n+1} \). The new state variables \( \mathbf{C}^{n+1}, \mathbf{F}^{n+1} \) are related to the original \( \mathbf{E}^0, \mathbf{F}_p \) by a matrix \( \mathbf{N} \). Since the changing portion of the state consists of two \( 3 \times 3 \) matrices, \( \mathbf{N} \) will be an \( 18 \times 18 \) matrix. A stable time step size is one such that the spectral radius (largest eigenvalue magnitude) of \( \mathbf{N} \) is no larger than 1.

**P2G and grid update.** We begin by transferring the particle’s state to the grid, which yields

\[
m_i^t = m_p w_{ip}^n, \quad \mathbf{u}^t = \mathbf{C}_p (\mathbf{x}^t - \mathbf{x}_p^n) = \epsilon \mathbf{A}^0 (\mathbf{x}_p^n - \mathbf{x}_p^n).
\]
Next, we compute particle forces. Let $\mathbf{M} = \frac{\partial^2 r}{\partial \mathbf{F}^2}$ and $\mathbf{K} = \mathbf{M} \cdot : \mathbf{E}^0$ so that $\mathbf{P} (\mathbf{F}^0_p) = \varepsilon \mathbf{M} \cdot : \mathbf{E}^0 = \varepsilon \mathbf{K}$. Neglecting higher order terms,

$$t^i_p = -\sum_r \frac{V_p^i (\mathbf{P} (\mathbf{F}_p^0)) (\mathbf{F}_p^0)^T \nabla w^r_p}{w^r_p} = -\varepsilon \nabla \cdot \mathbf{K} \nabla w^r_p.$$

Finally, we can compute the new grid velocities

$$\tilde{u}_i^{n+1} = u_i^n + \Delta (m_i^n)^{-1} t_i^n = \varepsilon \mathbf{A}^n (x^r_i - x_i^n) - \frac{\varepsilon \Delta t \nabla \nu^0}{w^r_p} \mathbf{K} \nabla w^r_p.$$

**G2P transfers.** A quick calculation shows that $u^p_{n+1} = \sum_r w^r_p \tilde{u}_i^{n+1} = 0$, so that particles do not move. Let

$$R = \frac{1}{\varepsilon} \sum \nabla w^r_p (\nabla w^r_p)^T.$$

We now update the deformation gradient on particles.

$$\nabla u^p_{n+1} = \sum_i \tilde{u}_i^{n+1} (\nabla w^r_p)^T = \varepsilon \mathbf{A}^n \sum_i (x^r_i - x_i^n) (\nabla w^r_p)^T = -\frac{\varepsilon \Delta t \nabla \nu^0}{m_p} \mathbf{K} \sum_i \nabla w^r_p (\nabla w^r_p)^T$$

$$= \varepsilon \mathbf{A}^n - \frac{\varepsilon \Delta t \nabla \nu^0}{m_p} \mathbf{K}.$$

$$F^p_{n+1} = (1 + \varepsilon \Delta t \mathbf{u}^p_{n+1}) F^p_n$$

$$= (I + \varepsilon \Delta t \mathbf{A}^n) - \frac{\varepsilon \Delta t \nabla \nu^0}{m_p} \mathbf{K} R + \mathbf{E}^n$$

$$E^n_{n+1} = \nabla \mathbf{A}^n - \frac{\varepsilon \Delta t \nabla \nu^0}{m_p} (\mathbf{M} \cdot : \mathbf{E}^0) R + \mathbf{E}^n$$

Finally we update the affine state.

$$C^p_{n+1} = \sum w^r_p \tilde{u}_i^{n+1} (x^r_i - x_i^n)^T$$

$$= \varepsilon \mathbf{A}^n \sum w^r_p (x^r_i - x_i^n)^T (x^r_i - x_i^n)^T$$

$$= -\frac{\varepsilon \Delta t \nabla \nu^0}{m_p} \mathbf{K} \sum \nabla w^r_p (x^r_i - x_i^n)^T = \varepsilon \mathbf{A}^n - \frac{\varepsilon \Delta t \nabla \nu^0}{m_p} \mathbf{M} : \mathbf{E}^n$$

$$A^n_{n+1} = A^n - \frac{\varepsilon \Delta t \nabla \nu^0}{m_p} \mathbf{M} : \mathbf{E}^n$$

Let $\tilde{M} = \frac{\varepsilon \Delta t \nu^0}{m_p} \mathbf{M}$. Then, we can express these as a matrix equation using indexing notation

$$\begin{pmatrix} \Delta A^0_{i,j} \\ F^p_{n+1} \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\
\end{pmatrix}$$

$$\begin{pmatrix} \Delta A^0_{i,j} \\ F^p_{n+1} \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\ \varepsilon \Delta t \nabla \nu^0 \\
\end{pmatrix}$$

We define $\mathbf{N}$ to be the resulting $18 \times 18$ matrix (8 × 8 in 2D).

**Simplifying the system.** The matrix $\mathbf{N}$ is defined in terms of two unitless quantities $\tilde{M}$ and $\mathbf{R}$. $\tilde{M}$ depends on forces and time step size, while $\mathbf{R}$ encodes the dependence on particle position. In the case of CPIC, $R = I$, so that particle position does not matter. In the case of APIC, $\mathbf{R}$ is diagonal. Let $a = \frac{3}{4}$, so that

$$R_{11} = \frac{3}{3 - 4a}$$

$$R_{11} = -\frac{2(9a^4 - 18a^3 + 2a^2 + 7a + 2)}{(3a^3 - 6a^2 + 4)(3a^3 - 3a^2 - 3a - 1)} \quad 0 \leq a \leq 1/2(quad)$$

The other entries $R_{22}$ and $R_{33}$ are similar. Note that the entries are decoupled. That is, $R_{11}$ depends on $x$ but not $y$ or $z$. Let $r_j = R_{ij}$ be the diagonal entries. For CPIC, $r_1 = 1$. For APIC with quadratic splines, $1 \leq r_j \leq 2$. For APIC with cubic splines, $1 \leq r_j \leq 1.0546$. We denote the relevant upper bound by $r$ so that $1 \leq r_j \leq r$.

Assuming an isotropic constitutive model, we have $M_{ijkl} = \lambda \delta_{ij} \delta_{kl} + \mu \delta_{ik} \delta_{jl} + \mu \delta_{il} \delta_{jk}$. We define new unitless scalars $p$ and $s$ so that $s = \frac{\varepsilon \Delta t \nu^0}{m_p}$ and $\lambda = \rho \mu$. The scalar $p = \frac{2 \varepsilon \Delta t \nu^0}{m_p}$ is a function of the Poisson’s ratio; it is an unknown positive constant for our purposes. $\mathbf{N}$ now depends on only $s, p, r$.

**Stability as a polynomial problem.** The characteristic polynomial $P(\lambda; s, p, r) = 0$ is known. The denominator is

$$P(\lambda; s, p, r) = (1 - \lambda^2 q(\lambda, s, p, r))$$

$$S(\lambda; s, r_1, r_2) S(\lambda; s, r_2, r_3) S(\lambda; s, r_1, r_3)$$

$$S(\lambda; s, r_1, r_2) = \lambda^2 + (r_1 s + r_2 - 2) \lambda - r_1 r_2 + 2 s + 1,$$
where $Q(\lambda; s, p, r_i)$ is a polynomial of degree 6 in $\lambda$ (the polynomial is omitted here for brevity but is provided in the technical document). The goal is to find

$$s^* = \min_{s \leq r/s} P(\lambda; s, p, r_i) = 0, |\lambda| = 1, s > 0.$$  

The time step restriction is then readily derived from $s^*$. Note that the factors of $P(\lambda; s, p, r_i)$ can be treated separately. The $(1 - \lambda)^6$ factor does not impose a time step restriction and can be discarded. There are now two high level cases to consider: $S(\lambda; s, r_i, r_j) = 0$ or $Q(\lambda; s, p, r_i) = 0$.

Case $S(\lambda; s, r_i, r_j) = 0$. We break this case into three further cases: $\lambda = \pm 1$ and complex conjugate $\lambda$. We first consider the case of complex conjugate solutions. Note that $(\lambda - \kappa)(\lambda - \bar{\kappa}) = \lambda^2 - (\kappa + \bar{\kappa})\lambda + \kappa\bar{\kappa}$, so that the square of the magnitude of the eigenvalues are just the constant term of the quadratic. That is, we need $1 = |\lambda|^2 = -sr_f - sr_j + 2s + 1$ which leads to $s(2 - r_i - r_j) = 0$. This case does not lead to a useful time step restriction ($s = 0$ implies $\Delta t = 0$). The case $\lambda = 1$ leads to $0 = S(1; s, r_i, r_j) = 2s$, which also does not lead to a useful solution. The case $\lambda = -1$ implies $0 = S(-1; s, r_i, r_j) = 2s - 2(s - r_i - r_j)s$, which leads to a meaningful time step restriction $s^* \leq A$ where

$$s^* \leq \frac{r_i}{2r_i} = \frac{1}{2} = A. \quad (2)$$

Case $Q(\lambda; s, p, r_i) = 0$. Next, we consider the case where $Q(\lambda; s, p, r_i) = 0$. As before, we consider $\lambda = \pm 1$ and complex conjugate $\lambda$ separately. $0 = Q(1; s, p, r_i)$ has only the solution $s = 0$. The complex conjugate case is very complex and does not lead to the slightest time step restriction, so we omit it here. The case is addressed in the technical document. This leaves the case $Q(-1; s, p, r_i) = 0$, which is a cubic polynomial in $s$, which could in theory be solved for $s(p, r_i, r_j, r_k)$. This would then need to be minimized over all feasible $r_i, r_j, r_k$ to produce the bound $s^*(p)$. This approach is infeasible, but there is a simpler way. Note that $p$ is constant and $r_i$ are independent. If $r_k$ is not at a bound, then $0 = \frac{\partial}{\partial r_k} Q(-1; s, p, r_i) = \frac{\partial Q}{\partial r_k}(-1; s, p, r_i)$. From this, we have $s^* = \min_{s \leq r/s} Q(-1; s, p, r_i)$. For each $r_k$, we must choose between $r_k = 1, r_k = r, r_k = \frac{1}{2}(1 - s/p, r_i) = 0$. This leads to many cases, but we can reduce these somewhat. The $r_k$ are equivalent, so permutations need not be considered.

**Partially unconstrained.** Consider that at least one $r_i$ is unconstrained (say, $r_1$). Then, $Q(-1; s, p, r_i) = -s^2(1 - s/p, r_i)$. Eliminating $r_1, s, p$ from this system of equations leads to the equation $(2r_2s - 2r_2s - 2s - 2)^2 = 0$, which implies $2r_2s - 2s - 2 = 0$ or $2r_2s - 2s - 2 = 0$. Both of these lead back to (2). We are left with the case that all $r_i$ are at their bounds 1 or $r$.

**Boundary cases.** There are four boundary cases (depending on how many of the $r_i$ are 1 and $r$). $0 = Q(-1; s, p, r_i, r_j, r_k, r_l) = -s^2(6r_1s - 3ps + 4r_1s - 2s - 4)$. The middle factor leads to (2). The last factor produces the restriction $s^* \leq B$ where

$$s^* \leq \frac{4}{(2r_1r_j - 3p + 2)} = \frac{4}{(2r_1 - 1)(3p + 2)} = B, \quad (3)$$

which is more strict than (2) since $0 < B < A$. $0 = Q(-1; s, p, r_i, r_j, r_k, r_l) = -4(2r_2s - 2s - 2)(s, p, r_i, r_j, r_k, r_l) = -4(s, p, r_i, r_j, r_k, r_l) = -4(s - 1)(2s - 1)(3s + 2)^5 + (4s - 8p + 6p - 8r_1s + 8r_2s + 8).$ The factor $(2r_2s - 2s - 2)$ leads to (2). The last factor $u(s, p, r_i, r_j, r_k, r_l)$ is a quadratic in $s$ where $u(s, p, r_i, r_j, r_k, r_l) = 0$, and $u(s, p, r_i, r_j, r_k, r_l) > 0$ for all $s \leq r_i, r_j, r_k, r_l$ and $s > p$. We conclude that any roots of $u(s, p, r_i, r_j, r_k, r_l)$, if they are real, must be larger than $B$. Thus, $s^* \leq B$ from (3) is the time step restriction for the single particle instability. Noting $s = \frac{\xi \Delta E_{P}^0}{m_p}$ we have our final single-particle time step restriction

$$\Delta t \leq \sqrt{\frac{m_p V_i^0}{\xi (r - \frac{1}{2})(\mu + \frac{1}{2} \xi)}}, \quad (4)$$

where $\xi$ is introduced to summarize the PIC case ($\xi = 0$) and APIC/CPIC cases ($\xi = k$). The PIC and 2D cases are handled in the technical document but omitted here. Figure 1 shows the single particle stability regions for PIC, APIC, and CPIC.

**Alternating time step sizes.** Conventional wisdom suggests that a simulation will be stable for any combination of time step sizes provided all of the time step sizes are below some critical value. This assumption is fundamental to the idea of using adaptive time step sizes. At least when simulating stationary isolated particles, this assumption is not true. There appear to be sequences of arbitrarily small time step sizes such that long term growth is observed.

We can demonstrate the problem of variable time step sizes by simulating a single stationary particle using CPIC transfers and $A^n = a^n I$ and $E^n = e^n I$, where $a^n$ and $e^n$ are scalars at time step $n$. Assuming an isotropic constitutive model, $M : E^n = (3\lambda + 2\mu) e^n I$. Letting $m = \frac{3\lambda + 2\mu}{2\mu} (3\lambda + 2\mu)$ and recalling that $R = I$ for CPIC we have

$$A^{n+1} = A^n - \frac{\xi \Delta t V_0}{m_p} M : E^n = a^n I - \Delta t m a^n I$$

$$E^{n+1} = \Delta t A^n - \frac{\xi \Delta t V_0^2}{m_p} (M : E^n) R + E^n = \Delta t a^n I - \Delta t^2 m e^n I + e^n I$$

We can express this as a matrix

$$(d^n + 1) = (1 - \Delta t \frac{\Delta t}{\Delta t^2 m} e^n I + e^n I). \quad (5)$$
If we take two steps with different time steps $\Delta t_0 < \Delta t_1$ then
\[
\begin{pmatrix}
\frac{a_{n+2}}{e_{n+2}} \\
\frac{\Delta t_0}{\Delta t_1}
\end{pmatrix} = \begin{pmatrix}
1 & -\Delta t_0^m \\
1 - \Delta t_0^2 & 0
\end{pmatrix} \begin{pmatrix}
1 & -\Delta t_1^m \\
1 - \Delta t_1^2 & 0
\end{pmatrix} \begin{pmatrix}
a_n \\
e_n
\end{pmatrix}.
\]

If we alternate between these two time steps sizes, then the eigenvalues $\lambda_0, \lambda_1$ of $B$ determine the stability. The matrix in (5) has determinant 1, so that $\lambda_0 \lambda_1 = \det(B) = 1$. If the eigenvalues are complex, then $\lambda_0 = \overline{\lambda_1}$ and $|\lambda_0| = |\lambda_1| = 1$, which is stable. If the eigenvalues are real and $\lambda_0 \neq \lambda_1$, then $0 < \lambda_0 < 1 < \lambda_1$ or $0 > \lambda_0 > -1 > \lambda_1$, which is unstable. The simulation is thus stable when $|\lambda_1 + \lambda_2| = |\text{tr}(B)| < 2$.

Alternating time step bound. Let $\Delta t_0 = k \Delta t_1$ with $0 < k < 1$. The case $\text{tr}(B) \leq 2$ leads to $\Delta t_1 \leq \frac{k+1}{k \sqrt{2}}$. This time step restriction is tightest when $k \rightarrow 1$ and leads to $\Delta t_1 \rightarrow \frac{T}{\sqrt{2}}$, where $T$ is the single particle instability time step restriction (4). The case $\text{tr}(B) \geq -2$ leads to $\Delta t_1 \leq \frac{k+2}{k \sqrt{2}}$. This stability requirement fails for all $\Delta t_1$ in an interval that includes $\Delta t_1 = \frac{k+1}{k \sqrt{2}}$. As $k \rightarrow 1$, $\Delta t_1 \rightarrow \frac{T}{\sqrt{2}}$. Plugging in the limit itself shows it to be stable for any $k$, so the bound is tight. Surprisingly, if we simply alternate between two nearly equal time step sizes, we must reduce the time step size by a factor of $\sqrt{2}$.

Cycled time steps have sharp teeth. Optimizing the trace while obeying the single particle instability criterion shows that the largest magnitude eigenvalue of $B$ occurs when $3 \Delta t_0 = \Delta t_1 = T$. (That is, alternate between taking the single particle time step bound and a time step one-third the size.) With these, $B$ has eigenvalues $-\frac{1}{2}$ and $-3$. Every two time steps, the state grows in magnitude by a factor of three. This growth rate is fast enough for a particle that separates from the main bulk to explode before colliding with an obstacle or other particles.

Longer cycles. What happens for longer cycles of time step sizes? The analysis above can in principle be repeated for any number of time step sizes. Cycling 3 nearly equal time step sizes admits unstable time step sizes smaller than (4) by a factor of $2 - \varepsilon$. Cycling a sequence of 49 time steps at $\Delta t$ followed by a step at $0.99 \Delta t$ is unstable for $\Delta t = 0.0314175$. For $99$+1 time steps, $\Delta t = 0.0157089T$ is unstable. Although this cycling scheme seems contrived, this is exactly what happens when using a fixed time step size that does not evenly divide the frame. It seems unlikely that the single particle instability can be absolutely avoided with APIC or APIC with variable time step sizes. Of course, the situation is not quite as bleak as that; the unstable eigenvalues in these two extreme cases are only $1 - 1.00002$ and $-1.00005$, which would not be noticed in practice.

Figure 1 shows the effects of non-constant time step sizes on the stability of a single particle.

4. Numerical results

In the analysis above, we derived two time step restrictions. The first was based on Von Neumann analysis, and the second was based on a single particle instability. In this section we probe at the relationships between these time step restrictions and the time step sizes at which actual simulations go unstable. In particular, both time step restrictions are sharp in the sense that there are simulations that go unstable at that time step size.

A few observations about the two restrictions are worth pointing out. The first difference is that the Von Neumann stability prediction is related to the classical CFL restriction and depends on the P-wave modulus $(k + 2 \mu)$. The single particle time step (4), however, depends on the bulk modulus $(k + 2 \mu)$. For completeness, (2) depends on the shear modulus $\mu$, which also happens to be equal to the S-wave modulus. That means the ratio between the two depends on the Poisson’s ratio and is not a simple fixed number (and thus a simple rule such as “use CFL 0.8” is not adequate). In our simulations, we always use the correct sound speed for our CFL (not the linearization derived from the Von Neumann stability analysis). With this, the single particle time step size usually falls between around CFL number 0.45 and 0.95, but it can be larger than 1.

Effects of boundaries. In Figure 2, we show a simple rotating circle of radius 0.3 in 2D. This circle is run with a Neo Hookean constitutive model with Young’s modulus $10^3$ and Poisson’s ratio 0.3. The angular velocity is 0.4. We use APIC transfers and quadratic splines so that the classical CFL agrees with the results of the Von Neumann analysis. We begin by running this simulation with CFL 0.9. While this simulation does not explode when run at CFL 1, the velocities do occasionally twitch. This suggests that the simulation...
briefly exceeds its stable step size in certain local areas of the simulation for brief periods of time. The simulation is fully stable at CFL 0.8. What is perhaps a bit unexpected is that if we simply change the geometry by removing all particles except the outer 0.015 of the circle (leaving the other particles in exactly the same places), the simulation explodes. The stable step size actually depends on the geometry, not merely factors such as particle spacing or stiffness. We can “fix” the unstable simulation by reducing the CFL number to 0.7. However, if we prune out more particles (leaving only 0.05 of the circle), the simulation explodes even at this CFL. Reducing the step size to the single particle time step size (CFL number 0.59 in this case) stabilizes the simulation. If run at a step size 5% larger, the simulation eventually explodes.

This simulation illustrates an important feature of MPM: neighboring particles have a stabilizing influence on MPM simulations. In the absence of boundaries, the simulation above would have been stable at CFL 0.9 and even CFL 1.0. The mere existence of a boundary (where there are fewer neighboring particles) reduces the stable time step size significantly. As the number of neighboring particles is reduced gradually by thinning the geometry, the stable time step size drops. In the limit as more and more particles are removed, we eventually end up with isolated particles, which by definition obey the single particle time step restriction. One may view the single particle time step size as a worst case scenario.

Figure 5 shows a similar experiment with colliding shells. As the shell gets thinner, the simulation requires smaller time step sizes. Generally, 3D simulations appear to be somewhat less susceptible to the effects of boundaries, likely since particles typically have more neighbors.

Instability due to boundaries. Instability when following the CFL is also occasionally observed even under seemingly innocuous conditions. In Figure 7, a sphere bounces between objects without following the proposed time step restriction, leading to instabilities and eventually an ejected particle. Obeying (4) stabilizes the simulation.

Instability due to corners. Corners of objects have even fewer neighboring particles than normal boundary particles, which can cause them to be especially sensitive. In Figure 6, the corner of a block becomes unstable, shown red. A particle is eventually ejected from the corner. Enforcing the single particle stability criterion stabilizes the simulation.

Instability due to fracture. Fracturing causes particles to have fewer neighbors than normal, which may reduce their stability. In Figure 8, a sphere fractures against an obstacle, causing particles to become more separated and correspondingly less stable. This simulation is unstable and explodes if the single particle time step is not followed. If the time step restriction is followed, the simulation behaves normally.

Sand. Figure 9 shows two sand spheres colliding. This simulation uses a standard Drucker Prager sand model. Plasticity can help to stabilize simulations, especially ones that might otherwise be unstable due to the single particle time step restriction. This instability is primarily a feedback loop between compression and expansion. Drucker Prager plasticity yields on expansion, breaking the feedback loop. This does, however, lead to divergence of the plastic component of the deformation gradient, which is not required for simulation. Because of this, simulations of sand generally do not become unstable on isolated particles.

Stiffer objects. The proposed step size properly accounts for stiffness and works fine for stiff objects, as shown in Figure 10. If run without the proposed restriction, instabilities are observed in the velocities on this simulation. In this case, the instabilities do not affect the appearance of the object, since we are not rendering velocity.

5. Conclusions and Limitations

In this paper, we have analyzed the stability of explicit MPM simulations in two complementary ways. Von Neumann analysis describes the stability of particles in a bulk medium. The single particle instability describes the stability of particles in the limit where particles have no neighbors and serves as a bound on the destabilizing effects of object boundaries. These complementary restrictions seem to work well at stabilizing non-isolated particles. Both analyses fully account for the discretization details of the algorithm, are practical, and are straightforward to implement. Both analyses are based on a linearization about the rest configuration, so the time step sizes may not be as reliable away from the rest configuration. Nevertheless, we have found them to be quite effective in practice.

Isolated particles are a significant problem for explicit MPM. If time steps are fixed, (4) solves the problem. Otherwise, there does not seem to be a practical way to stabilize the simulation using time step restrictions alone. The time step restrictions from [SSS20] can help somewhat, but they may take effect after some damage has already been done. A pragmatic solution is to simply detect isolated particles and reset (or decay) their state \((C^p \leftarrow 0, F^p \leftarrow 1)\). Alternatively, one may prefer to retain the skew-symmetric part of \(C^p\) to conserve angular momentum (similar to an APIC-RPIC blend.)
but only for isolated particles. It seems reasonable to assume that isolated particles should not be experiencing significant strain. This does not eliminate the need to follow (4), but it can mitigate the effects of variable time step sizes. Implicit time integration is another option for improving stability.

The time step restrictions in this paper are tight (simulations exist that require them) but do not guarantee stability. In fact, it appears that no guaranteed stable time step size exists for APIC or CPIC that does not depend on the positions of the MPM particles. In particular, an attempt at using full nonlinear stability analysis to construct a guaranteed stable time step size with APIC or CPIC transfers is unlikely to succeed. It is unclear how limiting this is in practice. Bulk MPM simulations do not seem to exhibit such pathological behavior provided particles do not become isolated. Further, isolated particles are easy to identify, so a simple corrective procedure could be employed to stabilize them. In practice, we have found the single particle time step restriction along with the restrictions from [SSS20] to be quite effective.

6. Acknowledgments

The simulations in this paper were rendered using SideFX Houdini. We would like to thank Tamar Shinar for many helpful discussions relating to this paper. This work was funded by NSF - 2006570.

References

[BK04] BARDENHAGEN S., KOBEL E.: The generalized interpolation material point method. Comp Mod in Eng and Sci 5, 6 (2004), 477–496. 2, 3


Figure 10: The time step restriction properly accounts for stiffness and works well for stiff objects.