Notes on Various MPM Extrapolations

John A. Nairn

December 22, 2018

Contents

Chapter 1

MPM Extrapolations - Principles

1.1 The MPM Extrapolation Problem

Let lower case denote grid (or Eulerian frame) quantities and upper case denote particle (or Lagrangian frame) quantities. We define S to be a matrix of grid-based shape functions to extrapolate from nodes to any position (normally to a particle position). Thus:

$$\begin{aligned} \boldsymbol{Q}_p &= \sum_i S_{pi} \boldsymbol{q}_i & (1.1) \\ 1 &= \sum_i S_{pi} & (1.2) \end{aligned}$$

The first extrapolates any nodal quantity, q_i, to a particle quantity, Q_p, and S_{pi} is shape function for node i at position of particle p (note that many MPM papers use S_{ip}, but the reversed order makes more sense with S and transformation from grid to particles). The second requires partition of unity for the shape functions. As a linear transformation

$$\boldsymbol{Q} = \mathsf{S}\boldsymbol{q} \qquad (1.3)$$

where $\boldsymbol{Q}$ is vector of particle properties, $\boldsymbol{q}$ is vector of nodal quantities, and S is the $N \times n$ transformation matrix (N particles and n nodes).

The first task in MPM time step n is to extrapolate from the particles to the grid. We begin by assuming a linear transformation

$$\boldsymbol{v}_i^{(n)} = \sum_p S^+_{ip} \boldsymbol{V}_p^{(n)} \qquad \text{or} \qquad \boldsymbol{v} = \mathsf{S}^+ \boldsymbol{V} \qquad (1.4)$$

where $\boldsymbol{v}$ and $\boldsymbol{V}$ are vectors of nodal and particle velocities, and S^+ is matrix of shape functions that ideally would be the pseudo-inverse of S. To make these transformations invertible, we would require $\mathsf{S}\mathsf{S}^+ = \mathsf{I}_{p\times p}$, but in generally the matrices are not square and therefore cannot be inverted. We resort instead to a pseudo inverse. Current MPM code uses $\mathsf{S}^+ = [m]^{-1}\mathsf{S}^T[M]$ where $[m]$ and $[M]$ are diagonal matrices with nodal and particle masses on the diagonals, respectively.

After extrapolating velocities to the grid MPM time step extrapolates particle stresses to the grid to find nodal forces, $\boldsymbol{f}_i^{(n)}$. The velocities and forces are used to update grid nodal positions and velocities using:

$$\begin{aligned} \tilde{\boldsymbol{x}}_i^{(n+1)} &= \boldsymbol{x}_i + \boldsymbol{v}_i^{(n)} \Delta t + \frac{1}{2} \boldsymbol{a}_i^{(n)} (\Delta t)^2 & (1.5) \\ \tilde{\boldsymbol{v}}_i^{(n+1)} &= \boldsymbol{v}_i^{(n)} + \boldsymbol{a}_i^{(n)} \Delta t & (1.6) \end{aligned}$$

The $\tilde{(\cdot)}$ nomenclature indicate grid position and velocity at the end of time step n, but it is not the grid position and velocity for time step $n+1$. Instead, for time step $n+1$, the grid position is reset to $\boldsymbol{x}_i$ and the velocity is extrapolated to the new grid using updated particle positions and velocities. The $\boldsymbol{a}_i^{(n)}$ is a grid acceleration given by $\boldsymbol{a}_i^{(n)} = \boldsymbol{f}_i^{(n)}/m_i^{(n)}$ where $m_i^{(n)}$ is nodal mass in time step n.

Next, the updated grid values are used to update particle positions and velocity, which can be written generally as

$$\begin{aligned} \boldsymbol{X}_p^{(n+1)} &= \boldsymbol{X}_p^{(n)} + \mathbb{V}_p^{(n)}\Delta t + \frac{1}{2}\mathbb{A}_p^{(n)}(\Delta t)^2 = \boldsymbol{X}_p^{(n)} + \mathbb{V}_p^{(n+1)}\Delta t - \frac{1}{2}\mathbb{A}_p^{(n)}(\Delta t)^2 && (1.7) \\ \boldsymbol{V}_p^{(n+1)} &= \boldsymbol{V}_p^{(n)} + \mathbb{A}_p^{(n)}\Delta t && (1.8) \end{aligned}$$

where $\mathbb{V}_p^{(n)}$ and $\mathbb{A}_p^{(n)}$ are velocity and acceleration extrapolated from the grid to the particle, by methods that are not yet determined or methods that can be potentially be selected to define various types of MPM. Whatever justification is used to select them, however, they are subject the physical consistency restriction that:

$$\mathbb{V}_p^{(n+1)} = \mathbb{V}_p^{(n)} + \mathbb{A}_p^{(n)}\Delta t \tag{1.9}$$

The above allows one to find particle updates for any definition of effective velocity and acceleration on the particles. Alternatively, if someone proposes particle updates, those can be reverse-engineered into the effective terms that were assumed:

$$\begin{aligned} \mathbb{A}_p^{(n)} &= \frac{\boldsymbol{V}_p^{(n+1)} - \boldsymbol{V}_p^{(n)}}{\Delta t} && (1.10) \\ \mathbb{V}_p^{(n)} &= \frac{\boldsymbol{X}_p^{(n+1)} - \boldsymbol{X}_p^{(n)}}{\Delta t} - \frac{1}{2}\left(\boldsymbol{V}_p^{(n+1)} - \boldsymbol{V}_p^{(n)}\right) && (1.11) \\ \mathbb{V}_p^{(n+1)} &= \frac{\boldsymbol{X}_p^{(n+1)} - \boldsymbol{X}_p^{(n)}}{\Delta t} + \frac{1}{2}\left(\boldsymbol{V}_p^{(n+1)} - \boldsymbol{V}_p^{(n)}\right) && (1.12) \end{aligned}$$

If needed, the following identities always hold:

$$\begin{aligned} \mathbb{A}_p^{(n)}\Delta t &= \mathbb{V}_p^{(n+1)} - \mathbb{V}_p^{(n)} = \boldsymbol{V}_p^{(n+1)} - \boldsymbol{V}_p^{(n)} \\ \mathbb{V}_p^{(n+1)} - \boldsymbol{V}_p^{(n+1)} &= \mathbb{V}_p^{(n)} - \boldsymbol{V}_p^{(n)} \\ \left(\mathbb{A}_p^{(n)} - \boldsymbol{A}_{g\to p}^{(n)}\right)\Delta t &= \boldsymbol{V}_p^{(n+1)} - \boldsymbol{V}_{g\to p}^{(n+1)} + \boldsymbol{V}_{g\to p}^{(n)} - \boldsymbol{V}_p^{(n)} \end{aligned} \tag{1.13}$$

1.2 Matrix Form

Let upper case vectors be particle properties and lower case grid properties. Let $\mathbb{V}^{(n)}$ and $\mathbb{A}^{(n)}$ be vectors of velocity and acceleration extrapolated to the particle. The task is to find $\mathbb{V}^{(n)}$, $\mathbb{V}^{(n+1)}$ and $\mathbb{A}^{(n)}$ and use them for particle position and velocity updates. Next start with a generalized MPM update, which be used to reverse engineer effective particle velocity and accelerations, as:

$$\boldsymbol{V}^{(n+1)} = \mathsf{P}\boldsymbol{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t \tag{1.14}$$

where P is some projection tensor. In standard FLIP MPM, $\mathsf{P} = \mathsf{I}$, but ideally P would be a filter that removes null space noise from the particle velocities or $(\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)}$ would be the null space velocities. Reverse engineering particle acceleration gives

$$\mathbb{A}^{(n)}\Delta t = \boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)} = \mathsf{S}\boldsymbol{a}\Delta t - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} \tag{1.15}$$

Note that a second interpretation of P is as damping of the extrapolated acceleration or

$$\mathbb{A}^{(n)}\Delta t = \boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)} = \left(\mathsf{S}\boldsymbol{a} - \alpha(\mathsf{P})\boldsymbol{V}^{(n)}\right)\Delta t \tag{1.16}$$

where $\alpha(\mathsf{P}) = \frac{1}{\Delta t}(\mathsf{I}-\mathsf{P})$ is a damping based on current particle velocities. It is a first order term (because of Δt in denominator). The acceleration term in position update is needed to capture this term.

1.3 General Results in Terms of $\mathbb{V}^{(n)}$

If we treat $\mathbb{V}^{(n)}$ as only undetermined quantity and add grid and particle damping to the acceleration, the update equations can be written in terms of eventual assunmption for $\mathbb{V}^{(n)}$ as:

$$\begin{aligned}
\mathbb{A}^{(n)}\Delta t &= \mathsf{S}\boldsymbol{a}\Delta t - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} - \alpha_g \Delta t \mathsf{S}\boldsymbol{v} - \alpha_p \Delta t \boldsymbol{V}^{(n)} && (1.17)\\
\mathbb{V}^{(n+1)} &= \mathbb{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} - \alpha_g \Delta t \mathsf{S}\boldsymbol{v} - \alpha_p \Delta t \boldsymbol{V}^{(n)} && (1.18)\\
\boldsymbol{V}^{(n+1)} &= \mathsf{P}\boldsymbol{V}^{(n)} + \mathsf{S}\boldsymbol{a}\Delta t - \alpha_g \Delta t \mathsf{S}\boldsymbol{v} - \alpha_p \Delta t \boldsymbol{V}^{(n)} && (1.19)\\
\boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \left[\mathbb{V}^{(n)} + \frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t - \frac{1}{2}(\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} - \frac{1}{2}\alpha_g \Delta t \mathsf{S}\boldsymbol{v} - \frac{1}{2}\alpha_p \Delta t \boldsymbol{V}^{(n)}\right]\Delta t && (1.20)
\end{aligned}$$

Introducing $\boldsymbol{A}^{(0)}_{damp} = \alpha_g \mathsf{S}\boldsymbol{v} + \alpha_p \boldsymbol{V}^{(n)}$, the general updates become

$$\begin{aligned}
\mathbb{A}^{(n)}\Delta t &= -(\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} + \left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t && (1.21)\\
\mathbb{V}^{(n+1)} &= \mathbb{V}^{(n)} - (\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} + \left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t && (1.22)\\
\boldsymbol{V}^{(n+1)} &= \mathsf{P}\boldsymbol{V}^{(n)} + \left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t && (1.23)\\
\boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \left[\mathbb{V}^{(n)} - \frac{1}{2}\boldsymbol{V}^{(n)} + \frac{1}{2}\left(\mathsf{P}\boldsymbol{V}^{(n)} + \left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t\right)\right]\Delta t && (1.24)\\
&= \boldsymbol{X}^{(n)} + \left[\mathbb{V}^{(n)} + \frac{\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}}{2}\right]\Delta t && (1.25)
\end{aligned}$$

Errors between Lagrangian position change and velocities are:

$$\begin{aligned}
\frac{\boldsymbol{X}^{(n+1)} - \boldsymbol{X}^{(n)}}{\Delta t} &= \mathbb{V}^{(n)} - \frac{1}{2}(\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)} + \frac{1}{2}\left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t && (1.26)\\
\frac{\boldsymbol{V}^{(n)} + \boldsymbol{V}^{(n+1)}}{2} &= \frac{1}{2}(\mathsf{I}+\mathsf{P})\boldsymbol{V}^{(n)} + \frac{1}{2}\left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t && (1.27)\\
\text{(error)} &= \mathbb{V}^{(n)} - \boldsymbol{V}^{(n)} && (1.28)
\end{aligned}$$

Once the projection operator is chosen, the above update determines the particle velocity update, but the position update depends on what is assumed for the effective velocity. The following sections give possible methods. Lastly, some result suggested second order terms, which appear only in position update, can cause instabilities (radial compression in axisymmetry). Truncating second terms might help, but the choice of how to truncate is not clear. The truncation can also be implement by altering the choice of $\mathbb{V}^{(n)}$, which might be the best tactic.

1.3.1 Assume $\mathbb{V}^{(n)} = \boldsymbol{V}^{(n)}$

A tempting choice is to select $\mathbb{V}^{(n)} = \boldsymbol{V}^{(n)}$, because that choice eliminates the error between particle position change and its average velocity. The updates become:

$$\boldsymbol{V}^{(n+1)} = \mathsf{P}\boldsymbol{V}^{(n)} + \left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t \tag{1.29}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \left[\frac{\boldsymbol{V}^{(n)} + \boldsymbol{V}^{(n+1)}}{2}\right]\Delta t \tag{1.30}$$

Whenever tried, this method gives poor results. The issue appears to be that particle velocities develop noise and that noise causes this position update to be very poor.

1.3.2 Assume $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$

The problem with $\mathbb{V}^{(n)} = \boldsymbol{V}^{(n)}$ was recognized in early days of FLIP modeling. The correction suggested by Brackbill (learned from after dinner talk at MPM Workshop in Oak Ridge in 2018) was essentially to use $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$. The updates become:

$$\boldsymbol{V}^{(n+1)} = \mathsf{P}\boldsymbol{V}^{(n)} + \left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t \tag{1.31}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \left[\mathsf{S}\boldsymbol{v} + \frac{\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}}{2}\right]\Delta t \tag{1.32}$$

Using $\tilde{\boldsymbol{v}} = \boldsymbol{v} + \boldsymbol{a}\Delta t$ leads to

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \left[\mathsf{S}\tilde{\boldsymbol{v}} + \frac{\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}}{2} - \mathsf{S}\boldsymbol{a}\Delta t\right]\Delta t \tag{1.33}$$

The error in Lagrangian velocities (with sign change) is now

$$(\text{error}) = \boldsymbol{V}^{(n)} - \mathbb{V}^{(n)} = \boldsymbol{V}^{(n)} - \mathsf{S}\boldsymbol{v} = (\mathsf{I} - \mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)} \tag{1.34}$$

This error suggest a path to improve FLIP is to damp the zeroth-order error by using

$$\mathbb{A}^{(n)} = \mathsf{S}\boldsymbol{a} - \alpha(\mathsf{I} - \mathsf{S}\mathsf{S}^{+})\boldsymbol{V}^{(n)} \tag{1.35}$$

This approach is essentially the PIC method (if $\alpha = 1$ and $\mathsf{P} = \mathsf{S}\mathsf{S}^{+}$). In other words, PIC is a method that damps out error in Lagrangian velocity.

A common error by those using either FLIP (with $\mathsf{P} = \mathsf{I}$) or PIC (with $\mathsf{P} = \mathsf{S}\mathsf{S}^{+}$) is to think the velocity update changes but the position is the same. Because the $\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}$ depends on P, however, the position update for PIC differs from the position update for FLIP.

First Order

Some simulations (or maybe only radial compression in axisymmetry) are better by shifting the effective velocity to middle of time step and removing damping or by choosing $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v} + (1/2)\mathsf{S}\boldsymbol{a}$, which is grid velocity at midpoint of time step without damping. The resulting position update is:

$$\begin{aligned}\boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\tilde{\boldsymbol{v}} + \frac{\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}}{2} - \frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t\right]\Delta t && (1.36)\\ &= \boldsymbol{X}^{(n)} + \left[\mathsf{S}\tilde{\boldsymbol{v}} - \frac{1}{2}\boldsymbol{V}^{(n)} + \frac{1}{2}\mathsf{P}\boldsymbol{V}^{(n)} - \frac{1}{2}\boldsymbol{A}^{(0)}_{damp}\right]\Delta t && (1.37)\end{aligned}$$

1.3.3 Assume $\mathbb{V}^{(n)} = \mathsf{P}\boldsymbol{V}^{(n)}$

An alternative view of Brackbill's suggestion is that it not revealing proper FLIP but rather recognizing that FLIP is inherently unstable, but can be fixed by using a smoothed velocity in the position update. If a good projection P is found, perhaps the best definition of MPM follows from $\mathbb{V}^{(n)} = \mathsf{P}\boldsymbol{V}^{(n)}$ or the particle velocity with null space removed (if P is non-null-space projector). The particle updates become:

$$\begin{aligned}
\boldsymbol{V}^{(n+1)} &= \mathsf{P}\boldsymbol{V}^{(n)} + \left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t && (1.38)\\
\boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \left[\frac{3\mathsf{P}\boldsymbol{V}^{(n)} - \boldsymbol{V}^{(n)}}{2} + \frac{1}{2}\left(\left(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\right)\Delta t\right)\right]\Delta t && (1.39)\\
&= \boldsymbol{X}^{(n)} + \left[\mathsf{P}\boldsymbol{V}^{(n)} + \frac{\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}}{2}\right]\Delta t && (1.40)
\end{aligned}$$

The errors between Lagrangian position update and average Lagrangian velocity are:

$$(\text{error}) = (\mathsf{I} - \mathsf{P})\boldsymbol{V}^{(n)} \tag{1.41}$$

or error proportional to null space terms. Note that this error is likely smaller then error when $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$. Note that for FLIP, $\mathsf{P} = \mathsf{I}$ and this approach reduces to approach in section 1.3.1 while for PIC, $\mathsf{P}\boldsymbol{V}^{(n)} = \mathsf{S}\boldsymbol{v}$ and this approach reduces to approach in section 1.3.2.

First Order

This method too might benefit from shifting to midpoint of the time step, which changes only the position update, or:

$$\begin{aligned}
\mathbb{V}^{(n)} &= \mathsf{P}\boldsymbol{V}^{(n)} + \frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t && (1.42)\\
\boldsymbol{X}^{(n+1)} &= \boldsymbol{X}^{(n)} + \left[\mathsf{P}\boldsymbol{V}^{(n)} + \frac{\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}}{2} + \frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t\right]\Delta t && (1.43)
\end{aligned}$$

1.3.4 Energy Dissipation

The energy change on one particle between state n and $n+1$ is effective force times distance or:

$$\Delta E_p = \frac{M_p \Delta \boldsymbol{V}_p}{\Delta t}\cdot\Delta\boldsymbol{X}_p = M_p\mathbb{A}^{(n)}_p\Delta t\cdot\left(\mathbb{V}^{(n)}_p + \frac{1}{2}\mathbb{A}^{(n)}_p\Delta t\right) \tag{1.44}$$

For a non-dissipative, FLIP update, the energy change is

$$\Delta E_{p,FLIP} = M_p\boldsymbol{A}_{g\to p}\Delta t\cdot\left(\boldsymbol{V}_{g\to p} + \frac{1}{2}\boldsymbol{A}_{g\to p}\Delta t\right) \tag{1.45}$$

where $\boldsymbol{A}_{g\to p}$ and $\boldsymbol{V}_{g\to p}$ are the p^{th} elements of $\mathsf{S}\boldsymbol{a}$ and $\mathsf{S}\boldsymbol{v}$, respectively. The difference $\Delta E_{p,FLIP} - \Delta E_p$ is energy dissipated (positive for loss of energy) by use of an effective velocity or

$$\Delta E_{p,disp} = M_p\Delta t\left(\boldsymbol{A}_{g\to p}\cdot\boldsymbol{V}_{g\to p} - \mathbb{A}^{(n)}_p\cdot\mathbb{V}^{(n)}_p + \frac{\Delta t}{2}\left(|\boldsymbol{A}_{g\to p}|^2 - |\mathbb{A}^{(n)}_p|^2\right)\right) \tag{1.46}$$

For the usual case of $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$, a simpler (and more general) derivation is

$$\begin{aligned}
\frac{\Delta E_{p,disp}}{M_p} &= \boldsymbol{A}_{g\to p}\Delta t \cdot \left(\boldsymbol{V}_{g\to p} + \frac{1}{2}\boldsymbol{A}_{g\to p}\Delta t\right) - \Delta\boldsymbol{V}_p \cdot \left(\boldsymbol{V}_{g\to p} + \frac{\Delta\boldsymbol{V}_p}{2}\right) && (1.47)\\
&= (\boldsymbol{A}_{g\to p}\Delta t - \Delta\boldsymbol{V}_p)\cdot\boldsymbol{V}_{g\to p} + \frac{1}{2}(\boldsymbol{A}_{g\to p}\Delta t - \Delta\boldsymbol{V}_p)\cdot(\boldsymbol{A}_{g\to p}\Delta t + \Delta\boldsymbol{V}_p) && (1.48)\\
&= (\boldsymbol{A}_{g\to p}\Delta t - \Delta\boldsymbol{V}_p)\cdot\left(\boldsymbol{V}_{g\to p} + \frac{1}{2}(\boldsymbol{A}_{g\to p}\Delta t + \Delta\boldsymbol{V}_p)\right) && (1.49)\\
&= (\boldsymbol{A}_{g\to p}\Delta t - \Delta\boldsymbol{V}_p)\cdot\left(\frac{1}{2}\boldsymbol{A}_{g\to p}\Delta t + \frac{\Delta\boldsymbol{X}_p}{\Delta t}\right) && (1.50)
\end{aligned}$$

1.4 Current MPM Methods

1.4.1 Extrapolation of Particle Property to the Grid

Sulsky derived the MPM method using a weighted least square assumption. Assuming particle velocities extrapolated from grid velocities is given by $\boldsymbol{V}^{(n+1)}_{g\to p} = \sum_i S_{pi}\boldsymbol{v}^{(n)}_i$, find the $\boldsymbol{v}^{(n)}_i$ to minimize the difference of this velocity with initial particle velocity ($\boldsymbol{V}^{(n)}_p$) weighted by the particle mass:

$$\begin{aligned}
\Omega &= \sum_p M_p\big(\boldsymbol{V}^{(n)}_p - \boldsymbol{V}^{(n)}_{g\to p}\big)^2 = \sum_p M_p\big(\boldsymbol{V}^{(n)}_p - \sum_i S_{pi}\boldsymbol{v}^{(n)}_i\big)^2 && (1.51)\\
0 = \frac{d\Omega}{d\boldsymbol{v}^{(n)}_j} &= -2\left(\sum_p S_{pj}M_p\boldsymbol{V}^{(n)}_p - \sum_i\sum_p M_pS_{pj}S_{pi}\boldsymbol{v}^{(n)}_i\right) && (1.52)\\
\sum_i m^{(n)}_{ji}\boldsymbol{v}^{(n)}_i &= \sum_p S_{pj}P^{(n)}_p && (1.53)\\
\mathsf{m}\boldsymbol{v} &= \mathsf{S}^T\boldsymbol{P} && (1.54)\\
\boldsymbol{v} &= \mathsf{m}^{-1}\mathsf{S}^T\boldsymbol{P} && (1.55)
\end{aligned}$$

where $m^{(n)}_{ji} = \sum_p M_pS_{pj}S_{pi}$ is an element of the full mass matrix m and $\boldsymbol{P}$ is a vector of particle momenta. If we replace m by the diagonal lumped mass matrix with $\boldsymbol{m}^{(n)}_i = \sum_j m^{(n)}_{ij} = \sum_p M_pS_{pi}$, this result reduces to

$$S^+_{ip} = \frac{M_pS_{pi}}{\boldsymbol{m}^{(n)}_i} \qquad (1.56)$$

Note that the GIMP derivation by Bardenhagen and Kober gets the same result without the needing to assert a lumped mass matrix (this fact may or may not be significant).

1.4.2 FLIP Particle Updates

FLIP is recovered by setting $\mathsf{P} = \mathsf{I}$. The particle velocity update is always

$$\boldsymbol{V}^{(n+1)} = \boldsymbol{V}^{(n)} + (\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp})\Delta t \qquad (1.57)$$

The position update can take two forms

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \begin{cases} \left(\mathsf{S}\boldsymbol{v} + \frac{\boldsymbol{V}^{(n+1)} - \boldsymbol{V}^{(n)}}{2}\right)\Delta t & \mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v} \\ \left(\frac{(\boldsymbol{V}^{(n)} + \boldsymbol{V}^{(n+1)})}{2}\right)\Delta t & \mathbb{V}^{(n)} = \boldsymbol{V}^{(n)} = \mathsf{P}\boldsymbol{V}^{(n)} \end{cases} \qquad (1.58)$$

The first version works well. The second one differs from standard FLIP by $\boldsymbol{V}^{(n)}-\mathsf{S}\boldsymbol{v}$ in the position update. This scheme does not work well.

For the usually case of $\mathbb{V}^{(n)}=\mathsf{S}\boldsymbol{v}$ and FLIP result of $\boldsymbol{A}_{g\to p}-\mathbb{A}_p^{(n)}=\boldsymbol{A}_{p,damp}^{(0)}$, the energy dissipation is

$$\Delta E_{p,disp}=M_p\boldsymbol{A}_{p,damp}^{(0)}\Delta t\cdot\left(\tilde{\boldsymbol{V}}_{g\to p}-\frac{1}{2}\boldsymbol{A}_{p,damp}^{(0)}\Delta t\right) \tag{1.59}$$

B. Modified FLIP Particle Updates #1

Many codes use an apparently first order position update based on updated grid velocity extrapolated to the particle and combine that with a standard FLIP velocity update, which for undamped case is:

$$\boldsymbol{V}^{(n+1)} = \boldsymbol{V}^{(n)}+\mathsf{S}\boldsymbol{a}\Delta t \tag{1.60}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)}+\mathsf{S}\tilde{\boldsymbol{v}}\Delta t=\boldsymbol{X}^{(n)}+(\mathsf{S}\boldsymbol{v}+\mathsf{S}\boldsymbol{a}\Delta t)\Delta t \tag{1.61}$$

We can reverse engineer the effective particle terms as

$$\mathbb{V}^{(n)}=\mathsf{S}\boldsymbol{v}+\frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t,\qquad \mathbb{V}^{(n+1)}=\mathsf{S}\boldsymbol{v}+\frac{3}{2}\mathsf{S}\boldsymbol{a}\Delta t,\qquad \text{and}\qquad \mathbb{A}^{(n)}=\mathsf{S}\boldsymbol{a} \tag{1.62}$$

This approach is like section 1.4.2 except velocity is taken from end of time step rather then middle (position update adds full acceleration rather than half the acceleration). This scheme works reasonably well in many codes and is same as first order option suggested in section 1.3.2. It is ignoring second order terms, but some problems (*e.g.*, radial compression of disk) appear more stable with second order terms ignored. This shifting of $\mathbb{V}^{(n)}$ by half the acceleration might be a option in other methods too.

A. Modified FLIP Particle Updates #2

Some MPM papers (at least one by Berzins) adopt the strategy of using the updated particle velocity in a first order update of the particle position. The (undamped) updates in this approach are:

$$\boldsymbol{V}^{(n+1)} = \boldsymbol{V}^{(n)}+\mathsf{S}\boldsymbol{a}\Delta t \tag{1.63}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)}+\boldsymbol{V}^{(n+1)}\Delta t=\boldsymbol{X}^{(n)}+\left[\boldsymbol{V}^{(n)}+\mathsf{S}\boldsymbol{a}\Delta t\right]\Delta t \tag{1.64}$$

This approach looks like an end-point approximation to second order position update while FLIP with $\mathbb{V}^{(n)}=\boldsymbol{V}^{(n)}$ is a midpoint, second-order position update in this same scheme. We can reverse engineer the effective particle terms as

$$\mathbb{V}^{(n)}=\boldsymbol{V}^{(n)}+\frac{1}{2}\mathsf{S}\boldsymbol{a}\Delta t,\quad \mathbb{V}^{(n+1)}=\boldsymbol{V}^{(n)}+\frac{3}{2}\mathsf{S}\boldsymbol{a}\Delta t,\quad \text{and}\quad \mathbb{A}^{(n)}=\mathsf{S}\boldsymbol{a}\ (\text{or } \mathsf{P}=\mathsf{I}) \tag{1.65}$$

This approach is like section 1.3.1 except velocity is taken from end of time step rather then middle (for the position update). Neither of these approaches work well.

1.4.3 PIC Particle Updates

PIC is recovered by setting $\mathsf{P}=\mathsf{S}\mathsf{S}^+$. The particle velocity update is always

$$\boldsymbol{V}^{(n+1)}=\mathsf{S}\boldsymbol{v}+(\mathsf{S}\boldsymbol{a}-\boldsymbol{A}_{damp}^{(0)})\Delta t=\mathsf{S}\tilde{\boldsymbol{v}}-\boldsymbol{A}_{damp}^{(0)}\Delta t \tag{1.66}$$

The position update looks similar to above, but combines options for $\mathbb{V}^{(n)}$ differently:

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \left\{ \begin{array}{ll} \left(\mathsf{S}\boldsymbol{v} + \frac{\boldsymbol{V}^{(n+1)}-\boldsymbol{V}^{(n)}}{2}\right)\Delta t & \mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v} = \mathsf{P}\boldsymbol{V}^{(n)} \\ \left(\frac{(\boldsymbol{V}^{(n)}+\boldsymbol{V}^{(n+1)}}{2}\right) & \mathbb{V}^{(n)} = \boldsymbol{V}^{(n)} \end{array} \right. \tag{1.67}$$

Also, the position update differ from FLIP because the update velocity used in position update differs.

For the usually case of $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$ and PIC result of:

$$(\boldsymbol{A}_{g\to p} - \mathbb{A}_p^{(n)})\Delta t = \boldsymbol{V}^{(n)} - \boldsymbol{V}_{g\to p} + \boldsymbol{A}_{p,damp}^{(0)}\Delta t \tag{1.68}$$

the energy dissipation is

$$\Delta E_{p,disp} = M_p(\boldsymbol{V}^{(n)} - \boldsymbol{V}_{g\to p} + \boldsymbol{A}_{p,damp}^{(0)}\Delta t)\cdot\left(\tilde{\boldsymbol{V}}_{g\to p} - \frac{\boldsymbol{V}^{(n)} - \boldsymbol{V}_{g\to p}}{2} - \frac{1}{2}\boldsymbol{A}_{p,damp}^{(0)}\Delta t)\right) \tag{1.69}$$

In OSParticulas, crack planes and surfaces update using PIC methods (because velocity is PIC too) and it does not have damping (maybe it should). Using $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$, the crack velocity and particle updates are:

$$\boldsymbol{V}^{(n+1)} = \mathsf{S}\boldsymbol{v} + \mathsf{S}\boldsymbol{a}\Delta t = \mathsf{S}\tilde{\boldsymbol{v}} \tag{1.70}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \left(\frac{3}{2}\mathsf{S}\boldsymbol{v} + \frac{\mathsf{S}\boldsymbol{a}\Delta t - \boldsymbol{V}^{(n)}}{2}\right)\Delta t \tag{1.71}$$

$$= \boldsymbol{X}^{(n)} + \left(\frac{3}{2}\mathsf{S}\tilde{\boldsymbol{v}} - \mathsf{S}\boldsymbol{a}\Delta t - \frac{\boldsymbol{V}^{(n)}}{2}\right)\Delta t \tag{1.72}$$

A. Modified PIC Particle Updates #1

Some papers (at least first from Disney), modified the velocity update, but not the common, first-order position update based on extrapolated grid velocities. This style particle update for PIC (with no damping) becomes

$$\boldsymbol{V}^{(n+1)} = \mathsf{S}\tilde{v} \tag{1.73}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \mathsf{S}\tilde{\boldsymbol{v}}\Delta t = \boldsymbol{X}^{(n)} + (\mathsf{S}\boldsymbol{v} + \mathsf{S}\boldsymbol{a}\Delta t)\Delta t \tag{1.74}$$

We can reverse engineer

$$\mathbb{V}^{(n)} = \frac{1}{2}\big(\mathsf{S}\tilde{\boldsymbol{v}} + \boldsymbol{V}^{(n)}\big), \quad \mathbb{V}^{(n+1)} = \frac{1}{2}\big(3\mathsf{S}\tilde{\boldsymbol{v}} - \boldsymbol{V}^{(n)}\big), \quad \text{and} \quad \mathbb{A}^{(n)} = \frac{1}{\Delta t}\big(\mathsf{S}\tilde{\boldsymbol{v}} - \boldsymbol{V}^{(n)}\big) \tag{1.75}$$

In terms of initial grid velocity and grid acceleration, this approach gives

$$\mathbb{V}^{(n)} = \boldsymbol{V}^{(n)} + \frac{1}{2}\left(\mathsf{S}\boldsymbol{a} - \frac{\boldsymbol{V}^{(n)} - \mathsf{S}\boldsymbol{v}}{\Delta t}\right)\Delta t = \mathsf{S}\boldsymbol{v} + \frac{1}{2}\left(\mathsf{S}\boldsymbol{a} + \frac{\boldsymbol{V}^{(n)} - \mathsf{S}\boldsymbol{v}}{\Delta t}\right)\Delta t \tag{1.76}$$

$$\mathbb{V}^{(n+1)} = \boldsymbol{V}^{(n)} + \frac{3}{2}\left(\mathsf{S}\boldsymbol{a} - \frac{\boldsymbol{V}^{(n)} - \mathsf{S}\boldsymbol{v}}{\Delta t}\right)\Delta t = \mathsf{S}\boldsymbol{v} + \frac{1}{2}\left(3\mathsf{S}\boldsymbol{a} - \frac{\boldsymbol{V}^{(n)} - \mathsf{S}\boldsymbol{v}}{\Delta t}\right)\Delta t \tag{1.77}$$

$$\mathbb{A}^{(n)} = \mathsf{S}\boldsymbol{a} - \frac{\boldsymbol{V}^{(n)} - \mathsf{S}\boldsymbol{v}}{\Delta t} \tag{1.78}$$

These results can be compared to PIC method used in `OSParticulas` in the previous section. They do not work well.

1.4.4 Combined FLIP/P Particle Updates

A combined fraction β of FLIP (using $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$) and $(1-\beta)$ of method with P (using either $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$ or $\mathbb{V}^{(n)} = \mathsf{P}\boldsymbol{V}^{(n)}$) can be expressed as follows

$$\boldsymbol{V}^{(n+1)} = \beta\boldsymbol{V}^{(n)} + (1-\beta)\mathsf{P}\boldsymbol{V}^{(n)} + (\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp})\Delta t \tag{1.79}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \begin{cases} \left(\mathsf{S}\boldsymbol{v} + \frac{\boldsymbol{V}^{(n+1)}-\boldsymbol{V}^{(n)}}{2}\right)\Delta t & \mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v} \\ \left(\beta\mathsf{S}\boldsymbol{v} + (1-\beta)\mathsf{P}\boldsymbol{V}^{(n)} + \frac{\boldsymbol{V}^{(n+1)}-\boldsymbol{V}^{(n)}}{2}\right)\Delta t & \mathbb{V}^{(n)} = \mathsf{P}\boldsymbol{V}^{(n)} \end{cases} \tag{1.80}$$

Note that this velocity update can be re-expressed as FLIP method with damping term that depends on P by replacing $\boldsymbol{A}^{(0)}_{damp}$ with a new effective damping term called $\boldsymbol{A}_{damp}$:

$$\boldsymbol{V}^{(n+1)} = \boldsymbol{V}^{(n)} + (\mathsf{S}\boldsymbol{a} - \boldsymbol{A}_{damp})\Delta t \tag{1.81}$$

$$\boldsymbol{A}_{damp} = (1-\beta)\frac{(\mathsf{I}-\mathsf{P})\boldsymbol{V}^{(n)}}{\Delta t} + \boldsymbol{A}^{(0)}_{damp} \tag{1.82}$$

$$\boldsymbol{A}^{(0)}_{damp} = \alpha_g\mathsf{S}\boldsymbol{v} + \alpha_p\boldsymbol{V}^{(n)} \tag{1.83}$$

$$\tag{1.84}$$

Note the for mixed FLIP/PIC and for $\mathbb{V}^{(n)} = \mathsf{S}\boldsymbol{v}$, the above general mixing simplifies down to methods once used in OSParticulas:

$$\boldsymbol{A}_{damp} = (1-\beta)\frac{\boldsymbol{V}^{(n)} - \mathsf{S}v}{\Delta t} + \alpha_g\mathsf{S}\boldsymbol{v} + \alpha_p\boldsymbol{V}^{(n)} \tag{1.85}$$

$$= \alpha_g^{(tot)}\mathsf{S}\boldsymbol{v} + \alpha_p^{(tot)}\boldsymbol{V}^{(n)} \tag{1.86}$$

$$\alpha_g^{(tot)} = \alpha_g - \frac{(1-\beta)}{\Delta t} \quad \text{and} \quad \alpha_p^{(tot)} = \alpha_p + \frac{(1-\beta)}{\Delta t} \tag{1.87}$$

Using the "tot" form and updated grid velocity gives:

$$\boldsymbol{V}^{(n+1)} = \boldsymbol{V}^{(n)} + \Big(\big(1 + \alpha_g^{(tot)}\Delta t\big)\mathsf{S}\boldsymbol{a} - \alpha_g^{(tot)}\mathsf{S}\tilde{\boldsymbol{v}} - \alpha_p^{(tot)}\boldsymbol{V}^{(n)}\Big)\Delta t \tag{1.88}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \mathsf{S}\tilde{\boldsymbol{v}}\Delta t - \Big(\big(1 - \alpha_g^{(tot)}\Delta t\big)\mathsf{S}\boldsymbol{a} + \alpha_g^{(tot)}\mathsf{S}\tilde{\boldsymbol{v}} + \alpha_p^{(tot)}\boldsymbol{V}^{(n)}\Big)\frac{(\Delta t)^2}{2} \tag{1.89}$$

Reverting to direct terms gives

$$\boldsymbol{V}^{(n+1)} = \boldsymbol{V}^{(n)} - (1-\beta)\big(\boldsymbol{V}^{(n)} - \mathsf{S}v\big) + \big(\mathsf{S}\boldsymbol{a} - \alpha_g\mathsf{S}\boldsymbol{v} - \alpha_p\boldsymbol{V}^{(n)}\big)\Delta t \tag{1.90}$$

$$= \beta\boldsymbol{V}^{(n)} + (1-\beta)\mathsf{S}v + \Big(\mathsf{S}\boldsymbol{a} - \boldsymbol{A}^{(0)}_{damp}\Big)\Delta t \tag{1.91}$$

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \left(\mathsf{S}\boldsymbol{v} - \frac{(1-\beta)\big(\boldsymbol{V}^{(n)} - \mathsf{S}v\big)}{2}\right)\Delta t + \big(\mathsf{S}\boldsymbol{a} - \alpha_g\mathsf{S}\boldsymbol{v} - \alpha_p\boldsymbol{V}^{(n)}\big)\frac{(\Delta t^2)}{2} \tag{1.92}$$