CS205b/CME306

Lecture 5

1 Time Integration

1.1 Newmark Method

Newmark methods are the most famous of the multivalue methods in computation solids. Multivalue methods are those that take advantage of the fact that higher order derivatives can be computed at lower accuracy to be more efficient. The Newmark method is a two-parameter family of methods:

$$x^{n+1} = x^n + \Delta t v^n + \frac{\Delta t^2}{2}((1-2\beta)a^n + 2\beta a^{n+1})$$
$$v^{n+1} = v^n + \Delta t((1-\gamma)a^n + \gamma a^{n+1}).$$

The specific method depends on the choice of β and γ chosen.

Constant acceleration equations, $\beta = \gamma = 0$

$$x^{n+1} = x^n + \Delta t v^n + \frac{\Delta t^2}{2}a^n \qquad v^{n+1} = v^n + \Delta t a^n$$

Implicit acceleration, $\beta = 1/2$ **and** $\gamma = 1$

$$x^{n+1} = x^n + \Delta t v^n + \frac{\Delta t^2}{2}a^{n+1} \qquad v^{n+1} = v^n + \Delta t \gamma a^{n+1}$$

The velocity update is the first order backward Euler update. The position update can be written as

$$x^{n+1} = x^n + \Delta t \frac{v^n + v^{n+1}}{2},$$

which is just the second order accurate trapezoid rule. This choice is still only first order accurate overall.

There is a theorem that states that second order accuracy is obtained if and only if $\gamma = 1/2$. The following two methods are second order Newmark methods.

Trapezoid rule, $\beta = 1/4$ **and** $\gamma = 1/2$

$$x^{n+1} = x^n + \Delta t v^n + \frac{\Delta t^2}{2}\frac{a^n + a^{n+1}}{2} \qquad v^{n+1} = v^n + \Delta t \frac{a^n + a^{n+1}}{2}$$

This method gets its name since the position equation can also be rewritten as a trapezoid rule update as

$$x^{n+1} = x^n + \Delta t \frac{v^n + v^{n+1}}{2}.$$

Central differencing, $\beta = 0$ and $\gamma = 1/2$

$$x^{n+1} = x^n + \Delta t v^n + \frac{\Delta t^2}{2} a^n \qquad v^{n+1} = v^n + \Delta t \frac{a^n + a^{n+1}}{2}.$$

The name of this method comes from the ability of its updates to be expressed as the central differences

$$\frac{x^{n+1} - x^{n-1}}{2\Delta t} = v^n \qquad \frac{x^{n+1} - 2x^n + x^{n-1}}{\Delta t^2} = a^n.$$

These equations can be obtained by

$$\begin{aligned}
x^{n+1} + x^n &= \left(x^n + \Delta t v^n + \frac{\Delta t^2}{2} a^n \right) + \left(x^{n-1} + \Delta t v^{n-1} + \frac{\Delta t^2}{2} a^{n-1} \right) \\
x^{n+1} - x^{n-1} &= \Delta t v^n + \frac{\Delta t^2}{2} a^n + \Delta t v^{n-1} + \frac{\Delta t^2}{2} a^{n-1} \\
\frac{x^{n+1} - x^{n-1}}{\Delta t} &= v^n + v^{n-1} + \Delta t \frac{a^n + a^{n-1}}{2} \\
\frac{x^{n+1} - x^{n-1}}{2\Delta t} &= v^n \\
x^{n+1} - x^n &= \left(x^n + \Delta t v^n + \frac{\Delta t^2}{2} a^n \right) - \left(x^{n-1} + \Delta t v^{n-1} + \frac{\Delta t^2}{2} a^{n-1} \right) \\
x^{n+1} - 2x^n + x^{n-1} &= \Delta t v^n + \frac{\Delta t^2}{2} a^n - \Delta t v^{n-1} - \frac{\Delta t^2}{2} a^{n-1} \\
\frac{x^{n+1} - 2x^n + x^{n-1}}{\Delta t} &= v^n - v^{n-1} + \frac{\Delta t}{2} (a^n - a^{n-1}) \\
\frac{x^{n+1} - 2x^n + x^{n-1}}{\Delta t} &= v^{n-1} + \Delta t \frac{a^{n-1} + a^n}{2} - v^{n-1} + \frac{\Delta t}{2} (a^n - a^{n-1}) \\
\frac{x^{n+1} - 2x^n + x^{n-1}}{\Delta t^2} &= a^n
\end{aligned}$$

1.2 Staggering the Position and Velocity

Define velocity halfway between time steps so that the update

$$v^{n+1/2} = \frac{x^{n+1} - x^n}{\Delta t}$$

is second order accurate. Note that x^n is still defined at the normal times. We can then define v^n by averaging nearby velocities as

$$v^n = \frac{v^{n-1/2} + v^{n+1/2}}{2} = \frac{x^{n+1} - x^{n-1}}{2\Delta t},$$

which is also central differencing for velocity. The update formula for positions is then just

$$x^{n+1} = x^n + \Delta t v^{n+1/2}.$$

Acceleration is also at grid points

$$a^n = a(x^n, v^n) = a\left(x^n, \frac{v^{n-1/2} + v^{n+1/2}}{2}\right)$$

so the update

$$v^{n+1/2} = v^{n-1/2} + \Delta t a^n$$

is also second order accurate. We also have the second order accurate central difference formula

$$\frac{x^{n+1} - 2x^n + x^{n-1}}{\Delta t^2} = a^n.$$

We can combine these formulas to obtain the explicit formula for a half time step

$$v^{n+1/2} = v^n + \frac{\Delta t}{2} a^n,$$

which is then used to update positions. Note that these two steps are equivalent to

$$x^{n+1} = x^n + \Delta t v^n + \frac{\Delta t^2}{2} a^n.$$

Finally, velocities are updated another half step using the implicit update

$$v^{n+1} = v^{n+1/2} + \frac{\Delta t}{2} a^{n+1} = v^{n+1/2} + \frac{\Delta t}{2} a(x^{n+1}, v^{n+1}).$$

The overall velocity update is

$$v^{n+1} = v^{n+1/2} + \frac{\Delta t}{2} a(x^{n+1}, v^{n+1}) = v^n + \frac{\Delta t}{2} a(x^n, v^n) + \frac{\Delta t}{2} a(x^{n+1}, v^{n+1}).$$

which is just the trapezoid rule.

The dependence of acceleration on velocity is often symmetric (e.g., damping forces), so that a fast solver like preconditioned conjugate gradient can be employed.

2 Springs

2.1 Simple Spring

One of the simplest forces one might imagine is the simple spring. A simple spring may be formulated in 1D with one endpoint fixed to the origin and the other of mass m located at x. Let x_0 be the rest length of the spring and k_s and k_d be spring-specific constants. The force applied by the spring is

$$F = -k_s\left(\frac{x}{x_0} - 1\right) - k_d v$$

The constant k_s is the spring constant, which has units of newtons $N = kgms^{-2}$. In 1D, this is just Young's modulus, but in higher dimensions the two differ by a geometry term. The constant k_d is the damping coefficient and has units of kgs^{-1}. We can write the spring as a first order system as

$$\begin{pmatrix} x \\ v \end{pmatrix}' = \begin{pmatrix} 0 & 1 \\ -\frac{k_s}{mx_0} & -\frac{k_d}{m} \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} + \begin{pmatrix} 0 \\ -\frac{k_s}{m} \end{pmatrix}.$$

The eigenvalues of this system are

$$\lambda = -\frac{k_d}{2m} \pm \sqrt{\left(\frac{k_d}{2m}\right)^2 - \frac{k_s}{mx_0}}$$

We can simplify analysis of the eigenvalues by rewriting them in terms of a dimensionless k_{do} as

$$k_d = k_{do}\sqrt{\frac{mk_s}{x_0}} \qquad \lambda = \frac{-k_{do} \pm \sqrt{k_{do}^2 - 4}}{2}\sqrt{\frac{k_s}{mx_0}}.$$

The system is under-damped if $k_{do} < 2$. If $k_{do} = 0$ the system has no damping at all, and its eigenvalues will be pure imaginary. The system is critically damped and has a repeated eigenvalues when $k_{do} = 2$, and the system is over-damped when $k_{do} > 2$. As before, $|\lambda|\Delta t \approx 1$.

2.2 Elastic Materials

By connecting masses and springs, it is possible to simulate elastic materials. In 1D, an elastic material is modeled by interconnecting a line of masses with springs. In 2D, an elastic material is modeled by splitting it into triangles with masses at the vertices and springs along the edges. In 3D, the same can be done using tetrahedra with masses at the vertices and springs along the edges. In the limit of infinite stiffness, these materials will behave as rigid bodies, and in the limit of no stiffness, the particles move without interacting.

Many materials may be modeled quite effectively using only masses and springs. There are other effective ways to model materials, such as the finite element method, which instead computes forces between particles by considering the deformation and material properties of the volumetric elements. Finite elements tend to require fewer elements, but they are also slower. The finite element method has better determined elastic behavior and permits arbitrary constitutive models, including fracture and plasticity. It is also more accurate, though accuracy has little meaning when considering collisions and fracture. We will cover finite elements more later.

2.3 1D Mass Spring System

We begin with a 1D mass spring system consisting of a line of n points with mass m and $n-1$ identical springs connecting them. For a particular spring, let x_l, x_r, v_l, and v_r be the position and velocity for the left and right ends of the spring. Let $\Delta x = x_r - x_l$ and $\Delta v = v_r - v_l$. As with the simple spring with a fixed endpoint, the force expected will be

$$F = -k_s\left(\frac{\Delta x}{x_0} - 1\right) - k_d\Delta v.$$

Note that translating the spring and examining it at a different point in space do not affect the force that it applies, so we may observe the spring from the position x_l of the left endpoint moving at its velocity v_l. From this vantage point, the spring looks *similar* to the simple spring with its non-fixed endpoint at location Δx with velocity Δv. (The system is different, though, in that neither endpoint is fixed. In particular, it will respond differently to the force applied by the spring. This does not affect the force the spring exerts, though.) Note that this is the force the spring applies to the right endpoint. The force applied to the left endpoint is $-F$ as required by Newton's

third law. The frequency of the system is λ in Hertz (s^{-1}), and the sound speed is $c = x_0\lambda$, in ms^{-1}. The sound speed looks like

$$c = x_0\sqrt{\frac{k_s}{mx_0}} = \sqrt{\frac{k_s x_0}{m}}.$$

To get more accuracy, we may want to refine this discretization. We would like to double the number of points to $2n$, which increases the number of springs to $2n-1 \sim 2(n-1)$. To keep the total mass constant, we must replace the n nodes with mass m with $2n$ nodes with mass $\hat{m} = m/2$. To keep the length constant, we must replace the the $n-1$ springs of length x_0 with $2n-1$ springs of length $\hat{x_0} = \frac{n-1}{2n-1}x_0 \sim x_0/2$. The Young's modulus k_s is a characteristic of the material and does not change. Ignoring the fencepost problem, the sound speed $\hat{c} = \sqrt{k_s x_0/m} = c$ then remains the same, and the frequency λ doubles to

$$\hat{\lambda} = 2\lambda \sim 2\sqrt{\frac{k_s}{mx_0}}.$$

Note that halving the spring length but keeping the same sound speed forces the frequency to double, since information must travel through twice as many springs in the same amount of time. Since $\Delta t\lambda \approx 1$, the stable time step size Δt is halved to $\hat{\Delta t} = \Delta t/2$ due to the refinement. The fraction x/x_0 that the material compresses or stretches does not change under the refinement, since both x and x_0 are both halved. The resulting elastic force and acceleration for a refined spring are

$$\hat{F} = -k_s\left(\frac{x/2}{x_0/2} - 1\right) = F \qquad \hat{a} = \frac{F}{m/2} = 2a.$$

In particular, the force is unchanged, but the acceleration of the individual particles doubles. Finally k_{do} is a property of the material and should not change, so that

$$\hat{k_d} = k_{do}\sqrt{\frac{k_s m/2}{x_0/2}} = k_d.$$

In summary, if the resolution of the discretization is doubled, the various parameters of the system change as follows:

$$\hat{m} = \frac{m}{2} \qquad \hat{x_0} = \frac{x_0}{2} \qquad \hat{k_s} = k_s \qquad \hat{k_d} = k_d$$

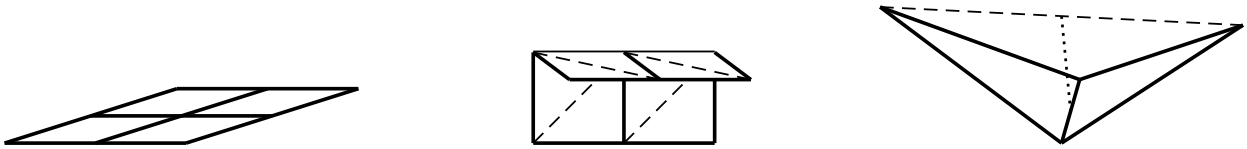
$$\hat{c} = c \qquad \hat{\lambda} = 2\lambda \qquad \hat{\Delta t} = \frac{\Delta t}{2} \qquad \hat{F} = F \qquad \hat{a} = 2a$$

2.4 Zero Length Spring

Simply setting $x_0 = 0$ in the simple spring model causes problems. However, when modeling materials, this value is reached through the limit $n \to \infty$, $x_0 \to 0$, and $m \to 0$.

Another potential situation where a zero-length spring might be used is to connect two objects. One solution to this problem is to place the connection points an arbitrary distance $x_0/2$ inside each object. Letting s be the amount the joint is separated, the distance between the ends of the spring is $x = x_0 + s$, and its derivative is where $v = x' = s'$. The spring force is

$$F = -k_s\left(\frac{x}{x_0} - 1\right) - k_d v = -k_s\left(\frac{x_0 + s}{x_0} - 1\right) - k_d v = -\frac{k_s}{x_0}s - k_d v.$$



(a) Edge springs do not prevent shear.

(b) Edge and shear springs do not prevent bending.

(c) Bending springs between two triangles.

Figure 1: Mass spring model for cloth.

The resulting system is then the same as before, except that it lacks the inhomogeneous term, and is

$$\begin{pmatrix} s \\ v \end{pmatrix}' = \begin{pmatrix} 0 & 1 \\ -\frac{k_s}{mx_0} & -\frac{k_d}{m} \end{pmatrix} \begin{pmatrix} s \\ v \end{pmatrix}.$$

We can now choose k_s and x_0, or we could just factor out the arbitrary parameter x_0 into the spring coefficient with

$$\hat{k_s} = \frac{k_s}{x_0} \qquad F = -\hat{k_s}x - k_d v \qquad k_d = k_{do}\sqrt{mk_s}.$$

2.5 2D Mass Spring System in 3D

Cloth may be modeled as a 2D surface that lives in 3D. The cloth surface might be discretized as a Cartesian grid of masses and springs, with masses at the corners connected along the edges by springs. The springs effectively restrict stretching and compression along the axes.

This cloth discretization lacks forces that are able to combat shear in the cloth. If a piece of cloth were modeled as a single square, one could fold the cloth into a line that is twice the edge length. Shearing the entire Cartesian grid of cloth has the same effect, illustrated in Figure 1(a). One way of solving this problem is to add springs along the diagonals of the squares in the cloth grid. One might choose to place springs on all of the left diagonals, all of the right diagonals, some mixture of left and right diagonals, or both left and right diagonals. Shear forces are different from stretching forces. Cloth tends to be very resistant to stretch but shears relatively easily, so shear springs are typically much weaker than edge springs. An alternative approach would be to use finite elements for the cloth, but for now we shall stick to springs.

This cloth model still has a major problem. This panel of cloth has no resistance to bending along the lines of springs that run along the axial directions, as in Figure 1(b). This may be fixed with additional springs that correct bending, such as the version obtained here by connecting opposing vertices of adjacent triangles (the shear springs create triangles). Bending forces in cloth are also typically very weak.

These bending springs have a couple potential limitations. If the desired angle between two triangles is nonzero, the bending spring will be unable to apply a force that would tend to correct the bend if the two triangles are coplanar or are bent in the wrong direction. Another issue with this model is that bending forces become very weak when the triangles are nearly coplanar. These two issues can be alleviated by connecting the shared edge between the triangles to the bending spring with yet another spring. This configuration is shown in Figure 1(c).

This setup works well in practice but may fail to apply forces in the right direction for some configurations of the cloth triangles. Note that the two triangles (formed by edge and shear springs)

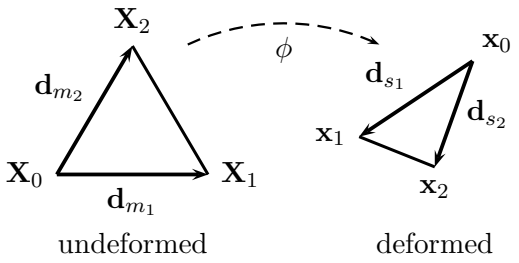


Figure 2: Undeformed and deformed triangle edges.

along with the original bending spring form a tetrahedron. achieving the desired lengths of these six springs is equivalent to achieving the desired shape for this tetrahedron. The tetrahedron may become inverted from its desired shape or degenerate. The second bending spring applies forces between a pair of opposing edges. In addition to this spring, there are two other edge-edge connections possible as well as four point-face possibilities (altitude springs). Between these seven possible springs locations, there will always be at least one choice (except in highly degenerate colinear configurations) that is capable of applying a suitable restoring force to the tetrahedron.

3 Finite Element Method

3.1 Geometric Calculation of Strain

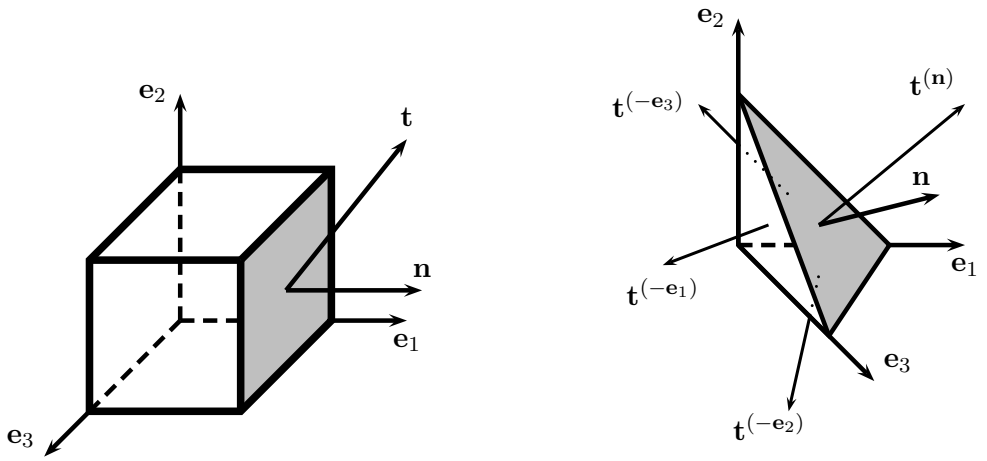
This section uses a significant amount of text from Teran et al., Finite Volume Methods for the Simulation of Skeletal Muscle, 2003, with permission of the author.

A deformable object is characterized by a time dependent map ϕ from undeformed material coordinates $\mathbf{X}$ to deformed spatial coordinates $\mathbf{x}$. We use a tetrahedron mesh and assume that the deformation is piecewise linear, which implies $\phi(X) = \mathbf{F}\mathbf{X} + \mathbf{b}$ in each tetrahedron. The Green strain is defined as $\mathbf{G} = (\mathbf{F}^T\mathbf{F} - \mathbf{I})/2$. Note that $\mathbf{G}$ defined in this way is rotation invariant, since applying a rotation to the deformed object results in a new transformation $\hat{\phi}(X) = R\mathbf{F}\mathbf{X} + R\mathbf{b}$, and $\hat{\mathbf{G}} = (R\mathbf{F})^T(R\mathbf{F}) - \mathbf{I} = \mathbf{F}^T\mathbf{F} - \mathbf{I} = \mathbf{G}$. Given $\mathbf{G}$, the original R can be recovered up to rotation and reflection, so little useful information is lost by constructing $\mathbf{G}$.

For simplicity, consider two spatial dimensions where each element is a triangle. Figure 2 depicts a mapping ϕ from a triangle in material coordinates to the resulting triangle in spatial coordinate. We define edge vectors for each triangle as $\mathbf{d}_{m_1} = \mathbf{X}_1 - \mathbf{X}_0$, $\mathbf{d}_{m_2} = \mathbf{X}_2 - \mathbf{X}_0$, $\mathbf{d}_{s_1} = \mathbf{x}_1 - \mathbf{x}_0$, and $\mathbf{d}_{s_2} = \mathbf{x}_2 - \mathbf{x}_0$. Note that $\mathbf{d}_{s_1} = (\mathbf{F}\mathbf{X}_1 + \mathbf{b}) - (\mathbf{F}\mathbf{X}_0 + \mathbf{b}) = \mathbf{F}\mathbf{d}_{m_1}$ and likewise $\mathbf{d}_{s_2} = \mathbf{F}\mathbf{d}_{m_2}$ so that $\mathbf{F}$ maps the edges of the triangle in material coordinates to the edges of the triangle in spatial coordinates. Thus, if we construct 2×2 matrices $\mathbf{D}_m$ with columns $\mathbf{d}_{m_1}$ and $\mathbf{d}_{m_2}$, and $\mathbf{D}_s$ with columns $\mathbf{d}_{s_1}$ and $\mathbf{d}_{s_2}$, then $\mathbf{D}_s = \mathbf{F}\mathbf{D}_m$ or $\mathbf{F} = \mathbf{D}_s\mathbf{D}_m^{-1}$. Using this definition of $\mathbf{F}$ the green strain is $\mathbf{G} = (\mathbf{D}_m^{-T}\mathbf{D}_s^T\mathbf{D}_s\mathbf{D}_m^{-1} - \mathbf{I})/2$, which can be rewritten to obtain

$$\mathbf{D}_m^T\mathbf{G}\mathbf{D}_m = \frac{1}{2}(\mathbf{D}_s^T\mathbf{D}_s - \mathbf{D}_m^T\mathbf{D}_m) = \frac{1}{2}\left[\begin{pmatrix} \mathbf{d}_{s_1}\cdot\mathbf{d}_{s_1} & \mathbf{d}_{s_1}\cdot\mathbf{d}_{s_2} \\ \mathbf{d}_{s_1}\cdot\mathbf{d}_{s_2} & \mathbf{d}_{s_2}\cdot\mathbf{d}_{s_2} \end{pmatrix} - \begin{pmatrix} \mathbf{d}_{m_1}\cdot\mathbf{d}_{m_1} & \mathbf{d}_{m_1}\cdot\mathbf{d}_{m_2} \\ \mathbf{d}_{m_1}\cdot\mathbf{d}_{m_2} & \mathbf{d}_{m_2}\cdot\mathbf{d}_{m_2} \end{pmatrix}\right]$$

in order to emphasize that we are simply measuring the change in the dot products of each edge with itself and the other edge.



(a) Cauchy stress for unit cube.

(b) Cauchy stress in an arbitrary direction.

Figure 3: Cauchy stress tensor $\boldsymbol{\sigma}$.

In three spatial dimensions, $\mathbf{D}_m$ and $\mathbf{D}_s$ are 3×3 matrices with columns equal to the edge vectors of the tetrahedra, and $\mathbf{D}_m^T \mathbf{G} \mathbf{D}_m$ is a measure of the difference between the dot products of each edge with itself and the other two edges. Note that $\mathbf{D}_m^{-1}$ can be computed and stored for efficiency.

3.2 Cauchy Stress

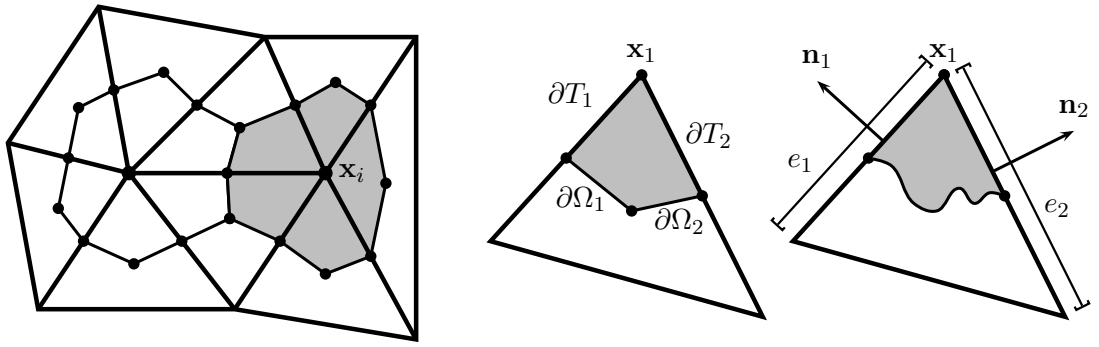
The stress an object is experiencing may be described by considering the relationship between a direction $\mathbf{n}$ and the traction (force per unit area) $\mathbf{t}^{(\mathbf{n})}$ applied to the plane cross section with normal $\mathbf{n}$. Consider a small cube inside a larger volume of material. The surrounding material applies stress to the cube through its faces, as illustrated in Figure 3(a). Forces in the direction of $\mathbf{n}$ are compressive or expansive forces. Forces orthogonal to $\mathbf{n}$ are sheer forces. Let $\mathbf{t}^{(\mathbf{e}_i)}$ be the traction applied to the plane of the unit cube with normal $\mathbf{e}_i$. Then, we can write these tractions in terms of the basis as

$$\mathbf{t}^{(\mathbf{e}_1)} = \boldsymbol{\sigma}_{11}\mathbf{e}_1 + \boldsymbol{\sigma}_{21}\mathbf{e}_2 + \boldsymbol{\sigma}_{31}\mathbf{e}_3 \qquad \mathbf{t}^{(\mathbf{e}_2)} = \boldsymbol{\sigma}_{12}\mathbf{e}_1 + \boldsymbol{\sigma}_{22}\mathbf{e}_2 + \boldsymbol{\sigma}_{32}\mathbf{e}_3 \qquad \mathbf{t}^{(\mathbf{e}_3)} = \boldsymbol{\sigma}_{13}\mathbf{e}_1 + \boldsymbol{\sigma}_{23}\mathbf{e}_2 + \boldsymbol{\sigma}_{33}\mathbf{e}_3$$

That is, $\mathbf{t}^{(\mathbf{e}_i)} = \boldsymbol{\sigma}\mathbf{e}_i$. This defines a 3×3 tensor $\boldsymbol{\sigma}$ called the Cauchy stress tensor. This definition of $\boldsymbol{\sigma}$ is given in terms of the axis-aligned normal directions $\mathbf{e}_i$.

We can extend the Cauchy stress tensor's application to an arbitrary direction by considering a tetrahedron as situated in Figure 3(b). We begin by computing the areas for the four faces. Assume the shaded face has area A. The areas of the other three faces can then be expressed as $(\mathbf{n} \cdot \mathbf{e}_1)A$, $(\mathbf{n} \cdot \mathbf{e}_2)A$, and $(\mathbf{n} \cdot \mathbf{e}_3)A$. The tractions for the three orthogonal faces are $\mathbf{t}^{(-\mathbf{e}_1)} = -\boldsymbol{\sigma}\mathbf{e}_1$, $\mathbf{t}^{(-\mathbf{e}_2)} = -\boldsymbol{\sigma}\mathbf{e}_2$, and $\mathbf{t}^{(-\mathbf{e}_3)} = -\boldsymbol{\sigma}\mathbf{e}_3$. The traction of the shaded triangle is $\mathbf{t}^{(\mathbf{n})}$. Because the total force on the tetrahedron should cancel out (assuming the object is in equilibrium), the sum of the forces on the faces (traction times area) should sum to zero

$$A\mathbf{t}^{(\mathbf{n})} - (\mathbf{n} \cdot \mathbf{e}_1)A\boldsymbol{\sigma}\mathbf{e}_1 - (\mathbf{n} \cdot \mathbf{e}_2)A\boldsymbol{\sigma}\mathbf{e}_2 - (\mathbf{n} \cdot \mathbf{e}_3)A\boldsymbol{\sigma}\mathbf{e}_3 = 0.$$



(a) FVM integration regions.

(b) Integration over a triangle.

Figure 4: Region around a node.

Dividing off A and rewriting the dot products using transposes yields

$$\mathbf{t}^{(\mathbf{n})} = \boldsymbol{\sigma}\mathbf{e}_1(\mathbf{e}_1^T\mathbf{n}) + \boldsymbol{\sigma}\mathbf{e}_2(\mathbf{e}_2^T\mathbf{n}) + \boldsymbol{\sigma}\mathbf{e}_3(\mathbf{e}_3^T\mathbf{n}).$$

Factoring and simplifying finishes off the derivation

$$\mathbf{t}^{(\mathbf{n})} = \boldsymbol{\sigma}(\mathbf{e}_1\mathbf{e}_1^T + \mathbf{e}_2\mathbf{e}_2^T + \mathbf{e}_3\mathbf{e}_3^T)\mathbf{n} = \boldsymbol{\sigma}\mathbf{I}\mathbf{n} = \boldsymbol{\sigma}\mathbf{n}.$$

In particular, the traction on a plane with unit normal $\mathbf{n}$ is $\mathbf{t}^{(\mathbf{n})} = \boldsymbol{\sigma}\mathbf{n}$.

Two important identities are obtained by considering the Cauchy stress along with conservation of momentum. The first is sometimes called the Cauchy equation of motion and is derived from conservation of linear momentum. The second is symmetry, which is obtained from the first equation and conservation of angular momentum.

$$\rho\mathbf{v}' = \nabla \cdot \boldsymbol{\sigma} + \rho\mathbf{f} \qquad \boldsymbol{\sigma} = \boldsymbol{\sigma}^T.$$

Here, $\mathbf{f}$ is the body force (force per unit mass).

3.3 Finite Element Method (FEM)

FEM provides a simple and geometrically intuitive way of integrating the equations of motion, with an interpretation that rivals the simplicity of mass-spring systems. However, unlike masses and springs, an arbitrary constitutive model can be incorporated into FEM. In the deformed configuration, consider dividing up the continuum into a number of discrete regions each surrounding a particular node. Figure 4(a) depicts two nodes each surrounded by a region. Suppose that we wish to determine the force on the node $\mathbf{x}_i$ surrounded by the region Ω. Ignoring body forces for brevity, the force can be calculated as

$$\mathbf{f} = \frac{D}{Dt}\int_\Omega \rho\mathbf{v}\,d\mathbf{x} = \oint_{\partial\Omega} \mathbf{t}\,dS = \oint_{\partial\Omega} \boldsymbol{\sigma}\mathbf{n}\,dS$$

where ρ is the density, $\mathbf{v}$ is the velocity, and $\mathbf{t}$ is the surface traction on $\partial\Omega$ (the force applied over the area). The last equality comes from the definition of the Cauchy stress on $\mathbf{t} = \boldsymbol{\sigma}\mathbf{n}$.

Evaluation of the boundary integral requires integrating over the two segments interior to each incident triangle. Figure 4(b) depicts one of these incident triangles along with interior segments labeled $\partial\Omega_1$ and $\partial\Omega_2$. Since $\boldsymbol{\sigma}$ is constant in each triangle and the integral of the local unit normal over any closed region is identically zero (from the divergence theorem), we have

$$\oint_{\partial\Omega_1} \boldsymbol{\sigma}\mathbf{n}\, dS + \oint_{\partial\Omega_2} \boldsymbol{\sigma}\mathbf{n}\, dS + \oint_{\partial T_1} \boldsymbol{\sigma}\mathbf{n}\, dS + \oint_{\partial T_2} \boldsymbol{\sigma}\mathbf{n}\, dS = 0$$

where ∂T_1 and ∂T_2 are depicted in the figure. Note that the integral of $\boldsymbol{\sigma}\mathbf{n}$ over $\partial\Omega_1$ and $\partial\Omega_2$ can be replaced by the integral of $-\boldsymbol{\sigma}\mathbf{n}$ over ∂T_1 and ∂T_2. That is, for each triangle, we can integrate over the portions of its edges incident to $\mathbf{x}_i$ instead of the two interior edges $\partial\Omega_1$ and $\partial\Omega_2$. Moreover, even if $\partial\Omega_1$ and $\partial\Omega_2$ are replaced by an arbitrary path inside the triangle, see Figure 4(b), we can replace the integral over this region with the integral over ∂T_1 and ∂T_2.

We choose an arbitrary path inside the triangles that connects the midpoints of the two edges incident on $\mathbf{x}_i$. Then the surface integrals are simply equal to $-\boldsymbol{\sigma}\mathbf{n}_1 e_1/2$ and $-\boldsymbol{\sigma}\mathbf{n}_2 e_2/2$ where e_1 and e_2 are the edge lengths of the triangles. Thus, the force contribution to node $\mathbf{x}_i$ from this triangle is

$$\mathbf{f} = -\frac{1}{2}\boldsymbol{\sigma}(e_1\mathbf{n}_1 + e_2\mathbf{n}_2).$$

In three spatial dimensions, we divide the volume of a tetrahedron among its four corners in such a way that the face areas are divided equally among their three corners. This is analogous to insisting in 2D that edges of the triangle be cut through their midpoints. Extending the above argument from 2D to 3D, we are able to express the force contribution from a tetrahedron to its vertices as

$$\mathbf{f} = -\frac{1}{3}\boldsymbol{\sigma}(a_1\mathbf{n}_1 + a_2\mathbf{n}_2 + a_3\mathbf{n}_3),$$

where a_1, a_2, and a_3 are the areas of three faces surrounding the node $\mathbf{x}_i$ under consideration, and $\mathbf{n}_1$, $\mathbf{n}_2$, and $\mathbf{n}_3$ are the faces' normals.

Given an arbitrary stress $\boldsymbol{\sigma}$, regardless of the method in which it was obtained, we obtain a force on the nodes in the following fashion. For each face j, compute the traction as $\boldsymbol{\sigma}\mathbf{n}_i$, and multiply by the face area a_i to obtain the total force for the face $\boldsymbol{\sigma}a_i\mathbf{n}_i$. Then, distribute one third of this force to each of the face's three vertices. Note that the forces applied to the four faces sum to zero, so only three need be computed directly.

3.4 Piola-Kirchhoff Stress

Often, application of a constitutive model (a relationship between stress and strain for a material) will result in a second Piola-Kirchoff stress, $\mathbf{S}$, which can be converted to a Cauchy stress via $\boldsymbol{\sigma} = J^{-1}\mathbf{F}\mathbf{S}\mathbf{F}^T$ where $J = \det(\mathbf{F})$. Using this equality and the identity $a\mathbf{n} = J\mathbf{F}^{-T}A\mathbf{N}$, we can write

$$\mathbf{f} = -\frac{1}{3}\mathbf{P}(A_1\mathbf{N}_1 + A_2\mathbf{N}_2 + A_3\mathbf{N}_3)$$

where $\mathbf{P} = \mathbf{F}\mathbf{S}$ is the first Piola-Kirchhoff stress tensor, the A_i are the areas of the undeformed tetrahedron faces incident to $\mathbf{X}_i$ and the $\mathbf{N}_i$ are the normals to those undeformed faces. Since the A_i and $\mathbf{N}_i$ do not change during the computation, we can precompute and store these quantities. Then the force contribution to each node can be computed as $\mathbf{f}_i = P\mathbf{b}_i$, where the $\mathbf{b}_i$ are precomputed. As before, only three of these need to be computed directly since total force on the four faces sums to zero.