Notes

• Some example values for common materials: (VERY approximate)
  • Aluminum: \(E = 70 \text{ GPa}\) \(v = 0.34\)
  • Concrete: \(E = 23 \text{ GPa}\) \(v = 0.2\)
  • Diamond: \(E = 950 \text{ GPa}\) \(v = 0.2\)
  • Glass: \(E = 50 \text{ GPa}\) \(v = 0.25\)
  • Nylon: \(E = 3 \text{ GPa}\) \(v = 0.4\)
  • Rubber: \(E = 1.7 \text{ MPa}\) \(v = 0.49\)
  • Steel: \(E = 200 \text{ GPa}\) \(v = 0.3\)

Finite Volume Method

• We discretize integral equation (complete with boundary integrals) over finite volumes
• Typically Voronoi regions (from Delaunay triangulation) or quads from grid
• Each finite volume has a vertex in the middle
  • Assume volume quantities are constant (equal to value at vertex)
  • Assume deformation affine in each triangle (constant strain \(\rightarrow\) constant stress)
  • [evaluate each term]
  • Exact choice of control volumes not critical - constant times normal integrates to zero

2D FVM strain

• Triangle with corners \(i, j, k\)
• Coordinates \(x_i = X(p_i), x_j = X(p_j), x_k = X(p_k)\)
• Assume affine in this triangle: \(X(p) = Ap + b\) (so \(\partial X / \partial p\))
• Then \(x_i - x_j = (Ap_i + b) - (Ap_j + b) = A(p_i - p_j)\)
• And \(x_i - x_k = A(p_i - p_k)\)
• Let \(\Delta x = [x_i - x_j \mid x_i - x_k]\) and \(\Delta p = [p_i - p_j \mid p_i - p_k]\)
• Thus \(A \Delta p = \Delta x\)
• Then \(A = \Delta x \Delta p^{-1}\)
• Can precompute and store \(\Delta p^{-1}\)

2D FVM stress calculation

• Look at single triangle \(i,j,k\) again
• Write part of path integral of \(\sigma n\) for volume \(i\)
• Take constant \(\sigma\) out
• Note integral of normals around closed curve is 0
• Switch to integrating normals along triangle boundary
• So force on \(i\) due to this triangle’s strain is
  • \(F_i = -\sigma ((x_i - x_k) \cdot ((x_j - x_i) \cdot) / 2\)
• Use the integral of normal=0 over triangle:
  • \(F_i = -\sigma (x_k - x_i) \cdot / 2\)
Path independence

- Note: trick involving switching path integral to triangle boundary works independent of the choice of finite volume
  - Force we compute only depends on where the finite volumes cut the triangle edges - always assume midpoint
- Choice of finite volume does affect body forces and mass calculation
- Might want to connect to centroids instead of circumcentres
  - always well defined (even if not Delaunay)
  - formulas much simpler: sum 1/3 triangle area

Boundary Conditions

- Actually easier to think about numerically
  - Control volumes automatically include boundary
  - Free boundary: do nothing
  - Specified traction: integrate over decomposed boundary mesh
    - E.g. wind forces...
  - Specified displacement (position): just set x to what it's supposed to be

Finite Element Method

- #1 most popular method for elasticity problems (and many others too)
- FEM originally began with simple idea:
  - Can solve idealized problems (e.g. that strain is constant over a triangle)
  - Call one of these problems an element
  - Can stick together elements to get better approximation
- Since then has evolved into a rigorous mathematical algorithm, a general purpose black-box method
  - Well, almost black-box...

Modern Approach

- Galerkin framework (the most common)
- Find vector space of functions that solution (e.g. X(p)) lives in
  - E.g. bounded weak 1st derivative: $H^1$
- Say the PDE is $L[X]=0$ everywhere (“strong”)
- The “weak” statement is $\int Y(p)L[X(p)]dp=0$ for every $Y$ in vector space
- Issue: $L$ might involve second derivatives
  - E.g. one for strain, then one for div sigma
  - So $L$, and the strong form, difficult to define for $H^1$
- Integration by parts saves the day
Weak Momentum Equation

- Ignore time derivatives - treat acceleration as an independent quantity
- We discretize space first, then use “method of lines”: plug in any time integrator

\[
L[X] = \rho \ddot{X} - f_{\text{body}} - \nabla \cdot \sigma
\]

\[
\int_{\Omega} L[X] = \int_{\Omega} Y(\rho \ddot{X} - f_{\text{body}} - \nabla \cdot \sigma)
\]

\[
= \int_{\Omega} Y \rho \ddot{X} - \int_{\Omega} Yf_{\text{body}} - \int_{\Omega} Y \nabla \cdot \sigma
\]

\[
= \int_{\Omega} Y \rho \ddot{X} - \int_{\Omega} Yf_{\text{body}} + \int_{\Omega} \sigma \nabla Y
\]

Making it finite

- The Galerkin FEM just takes the weak equation, and restricts the vector space to a finite-dimensional one
  - E.g. Continuous piecewise linear - constant gradient over each triangle in mesh, just like we used for Finite Volume Method
- This means instead of infinitely many test functions \( Y \) to consider, we only need to check a finite basis
- The method is defined by the basis
  - Very general: plug in whatever you want - polynomials, splines, wavelets, RBF’s, …

Linear Triangle Elements

- Simplest choice
- Take basis \( \{ \phi_i \} \) where \( \phi_i(p) = 1 \) at \( p_i \) and 0 at all the other \( p_j \)'s
  - It’s a “hat” function
- Then \( X(p) = \sum_i x_i \phi_i(p) \) is the continuous piecewise linear function that interpolates particle positions
- Similarly interpolate velocity and acceleration
- Plug this choice of \( X \) and an arbitrary \( Y = \phi_j \) (for any \( j \)) into the weak form of the equation
- Get a system of equations (3 eq. for each \( j \))

The equations

\[
\int_{\Omega} \phi_j \sum_i \rho \ddot{x}_i \phi_i - \int_{\Omega} \phi_j f_{\text{body}} + \int_{\Omega} \sigma \nabla \phi_j = 0
\]

\[
\sum_i \int_{\Omega} \rho \phi_j \phi_i \ddot{x}_i = \int_{\Omega} \phi_j f_{\text{body}} - \int_{\Omega} \sigma \nabla \phi_j
\]

• Note that \( \phi_j \) is zero on all but the triangles surrounding \( j \), so integrals simplify
• Also: naturally split integration into separate triangles
Change in momentum term

- Let $m_{ij} = \int \rho \phi_i \phi_j$
- Then the first term is just $\sum_i m_{ij} \ddot{x}_i$
- Let $M = [m_{ij}]$: then first term is $M \ddot{x}$
- $M$ is called the mass matrix
  - Obviously symmetric (actually SPD)
  - Not diagonal!
- Note that once we have the forces (the other integrals), we need to invert $M$ to get accelerations

Body force term

- Usually just gravity: $f_{\text{body}} = \rho g$
- Rather than do the integral with density all over again, use the fact that $\phi_i$ sum to 1
  - They form a “partition of unity”
  - They represent constant functions exactly - just about necessary for convergence
- Then body force term is $gM1$
- More specifically, can just add $g$ to the accelerations; don’t bother with integrals or mass matrix at all

Stress term

- Calculate constant strain and strain rate (so constant stress) for each triangle separately
- Note $\nabla \phi_j$ is constant too
- So just take $\alpha \nabla \phi_j$ times triangle area
- [derive what $\nabla \phi_j$ is]
- Magic: exact same as FVM!
  - In fact, proof of convergence of FVM is often (in other settings too) proved by showing it’s equivalent or close to some kind of FEM

The algorithm

- Loop over triangles
  - Loop over corners
  - Compute integral terms
    - only need to compute $M$ once though - it’s constant
  - End up with row of $M$ and a “force”
- Solve $Ma = f$
- Plug this a into time integration scheme
Lumped Mass

- Inverting mass matrix unsatisfactory
  - For particles and FVM, each particle had a mass, so we just did a division
  - Here mass is spread out, need to do a big linear solve - even for explicit time stepping
- Idea of lumping: replace $M$ with the “lumped mass matrix”
  - A diagonal matrix with the same row sums
  - Inverting diagonal matrix is just divisions - so diagonal entries of lumped mass matrix are the particle masses
  - Equivalent to FVM with centroid-based volumes

Consistent vs. Lumped

- Original mass matrix called “consistent”
- Turns out its strongly diagonal dominant (fairly easy to solve)
- Multiplying by mass matrix = smoothing
- Inverting mass matrix = sharpening
- Rule of thumb:
  - Implicit time stepping - use consistent $M$ (counteract over-smoothing, solving system anyways)
  - Explicit time stepping - use lumped $M$ (avoid solving systems, don’t need extra sharpening)

Locking

- Simple linear basis actually has a major problem: locking
- Notion of numerical stiffness
  - Instead of thinking of numerical method as just getting an approximate solution to a real problem,
  - Think of numerical method as exactly solving a problem that’s nearby
  - For elasticity, we’re exactly solving the equations for a material with slightly different (and not quite homogeneous/isotropic) stiffness
- Locking comes up when numerical stiffness is MUCH higher than real stiffness

Locking and linear elements

- Look at nearly incompressible materials
- Can a linear triangle mesh deform incompressibly?
  - [derive problem]
- Then linear elements will resist far too much: numerical stiffness much too high
- Numerical material “locks”
- FEM isn’t really a black box!
- Solutions:
  - Don’t do incompressibility
  - Use other sorts of elements (quads, higher order)
Quadrature

• Formulas for linear triangle elements and constant density simple to work out
• Formulas for subdivision surfaces (or high-order polynomials, or splines, or wavelets…) and varying density are NASTY
• Instead use “quadrature”
  • i.e. numerical approximation to integrals
• Generalizations of midpoint rule
  • E.g. Gaussian quadrature (for intervals, triangles, tets) or tensor products (for quads, hexes)
• Make sure to match order of accuracy [or not]

Accuracy

• At least for SPD linear problems (e.g. linear elasticity) FEM selects function from finite space that is “closest” to solution
  • Measured in a least-squares, energy-norm sense
• Thus it’s all about how well you can approximate functions with the finite space you chose
  • Linear or bilinear elements: O(h²)
  • Higher order polynomials, splines, etc.: better

Other elements

• Not so obvious ones:
  • Isoparametric elements (meshes with curved edges)
  • Radial-basis functions (mesh-free methods)
  • Mixed element meshes (triangles and quads together)
  • Embedded elements
  • Special-purpose elements (e.g. for cracks)