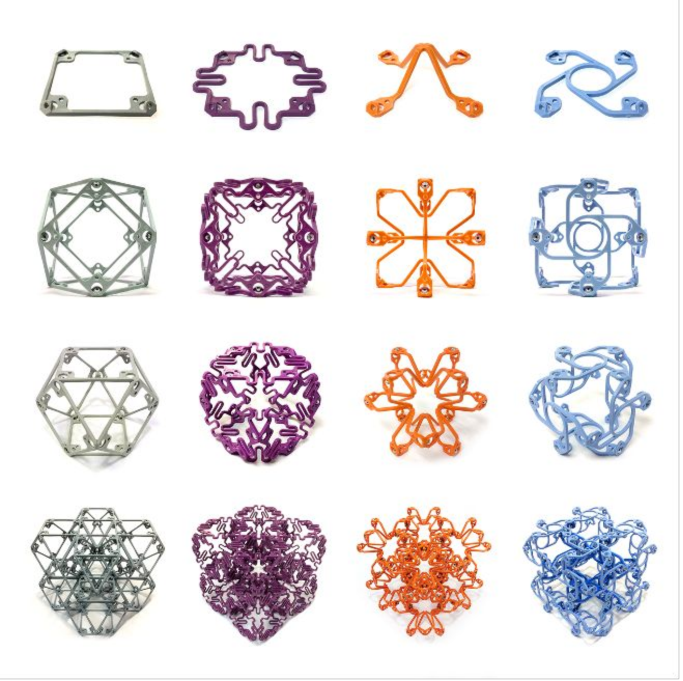
Discrete elements for simulating failure in voxel structures

The Nature of Mathematical Modeling Final Project
May 22, 2023
Miana Smith

What are voxels? *(in this context)*



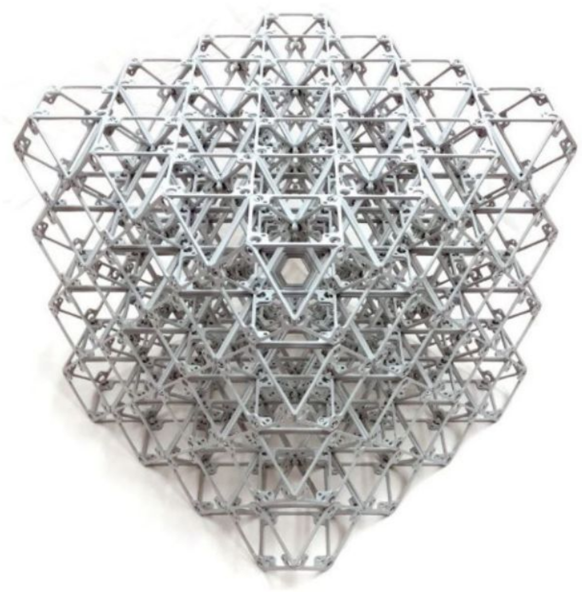
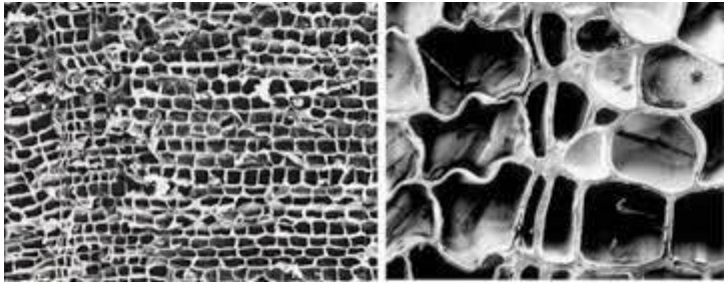
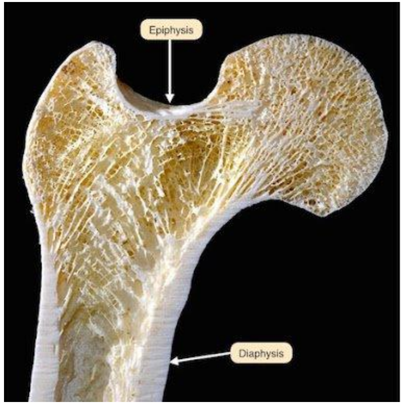


Image sources: https://cba.mit.edu/docs/theses/20.09.jenett.pdf,

Why cellular materials?

High performance, mass efficient

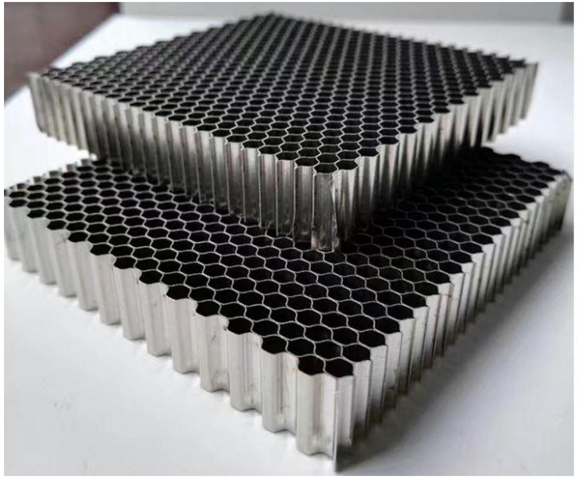




Micrograph of cork

Bone internal structure

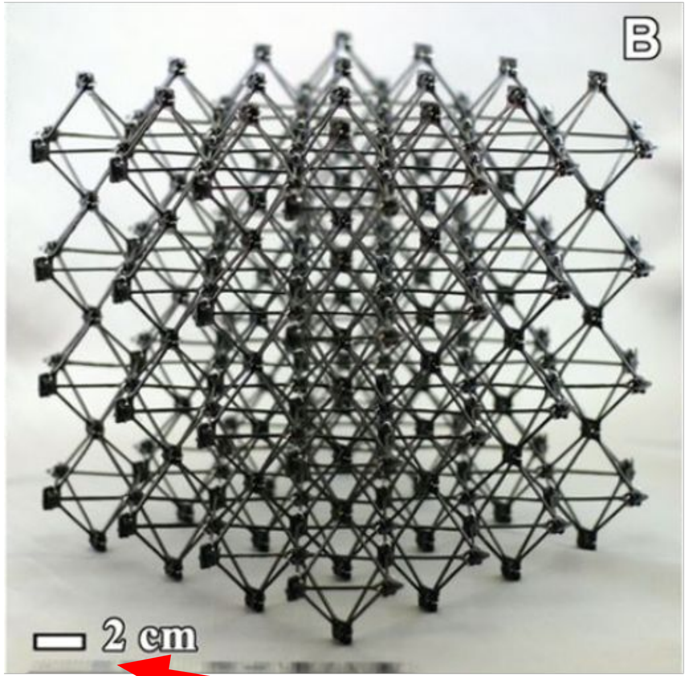
Truss bridge

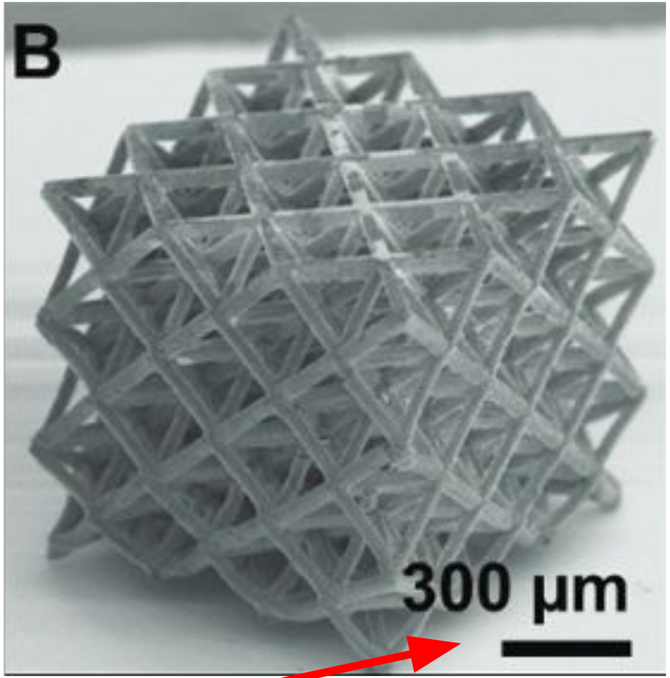


Aluminum honeycomb

Images from: https://qph.cf2.quoracdn.net/main-qimg-f162086a70ea86b759f2f7214e714ed7-lq, https://bioresources.cnr.ncsu.edu/resources/the-rationale-behind-cork-properties-a-review-of-structure-and-chemistry/, https://www.enr.com/articles/38496-the-worlds-ten-longest-continuous-truss-bridges, https://www.dmcrf.com/aluminum-honeycomb-core/

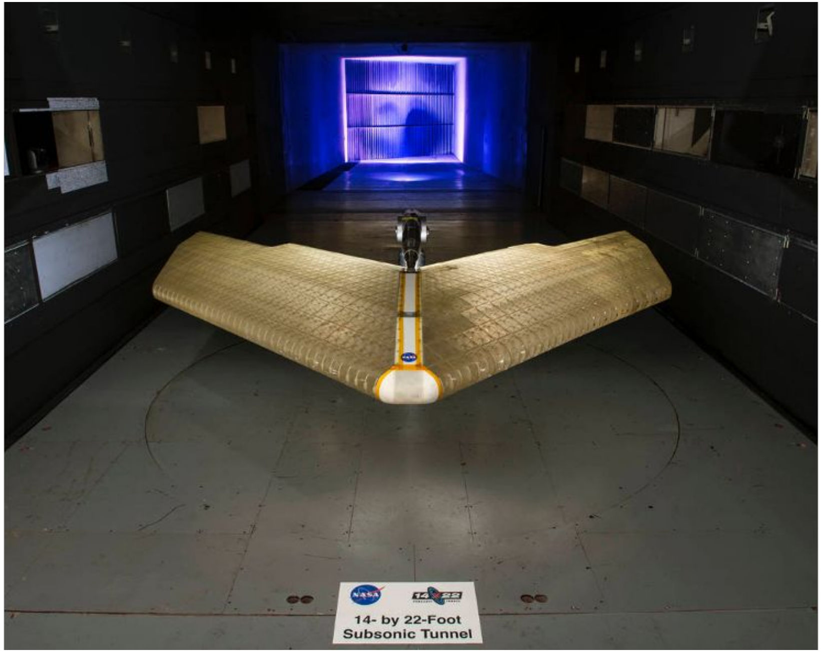
Why voxels?





versus

Example prior voxel applications



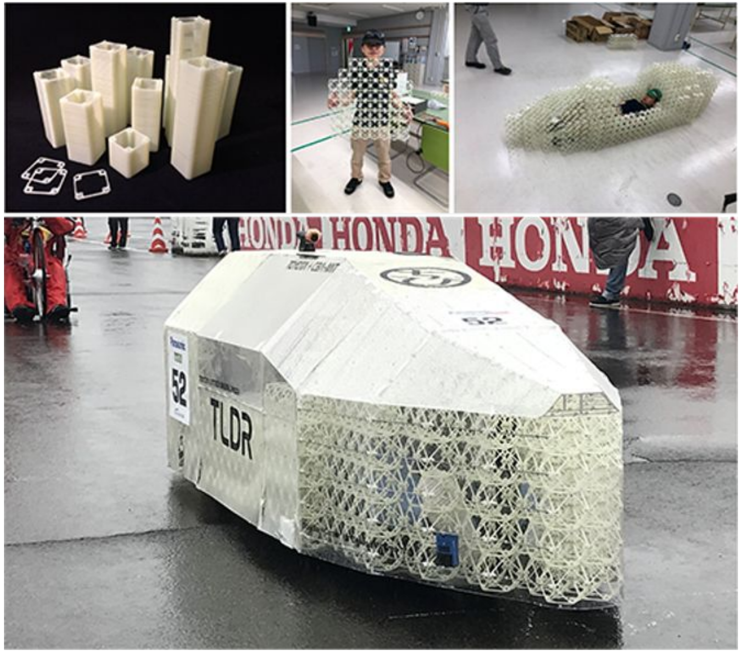
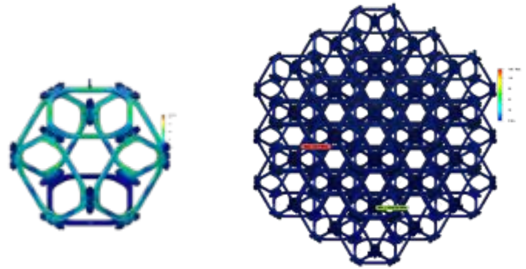


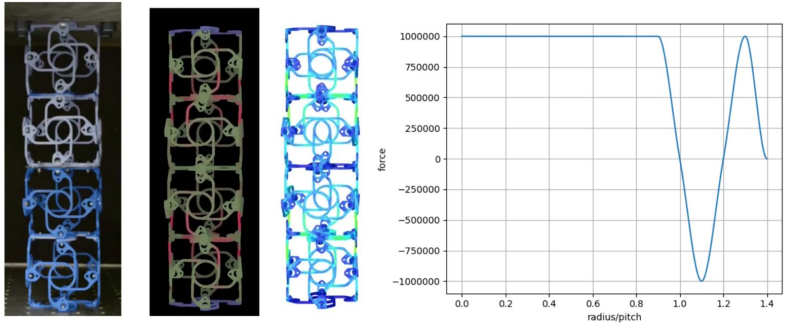
Image sources: https://www.nasa.gov/feature/ames/madcat, https://cba.mit.edu/media/pictures.html

Other methods for simulating voxels



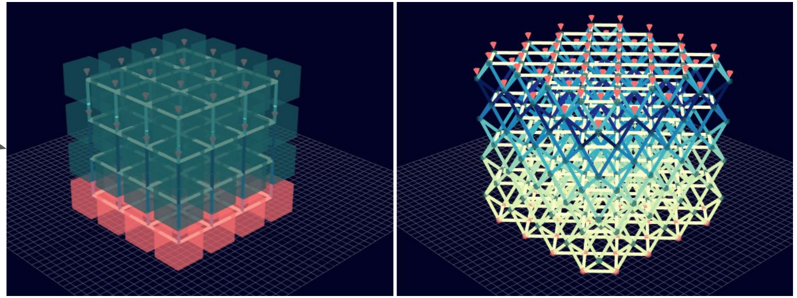
Existing PDEs, FEM based tools

- E.g. Fusion 360’s FEA environment



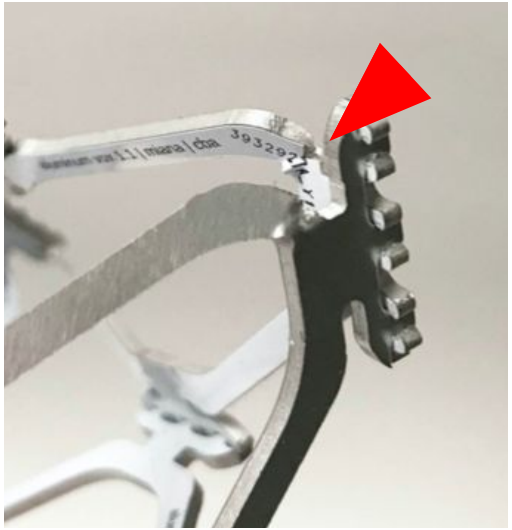
CBA tools

- Erik and Neil’s discrete element methods
- Amira’s Metavoxels

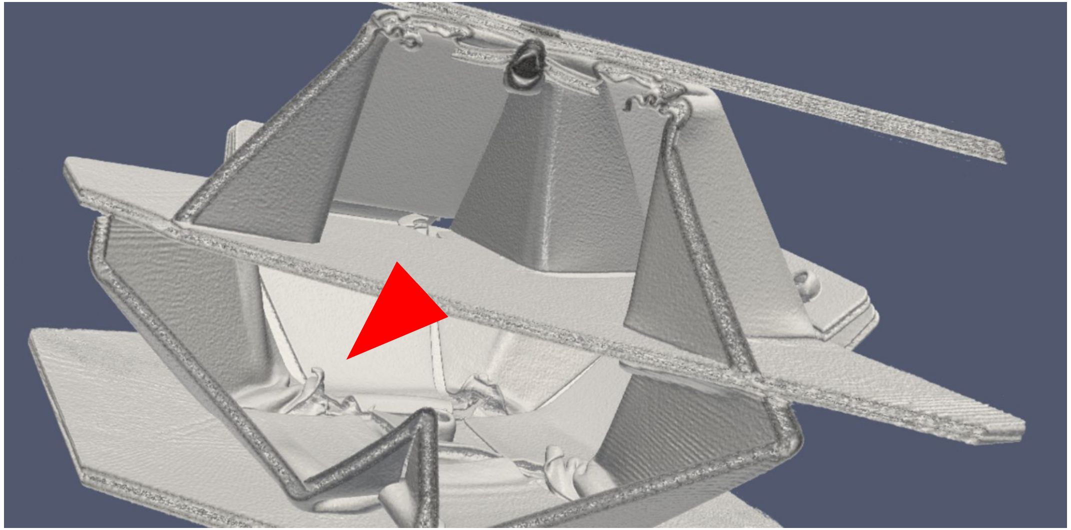


Images from: https://gitlab.cba.mit.edu/amiraa/metavoxels, https://fab.cba.mit.edu/classes/864.23/text/Discrete_Elements.pdf

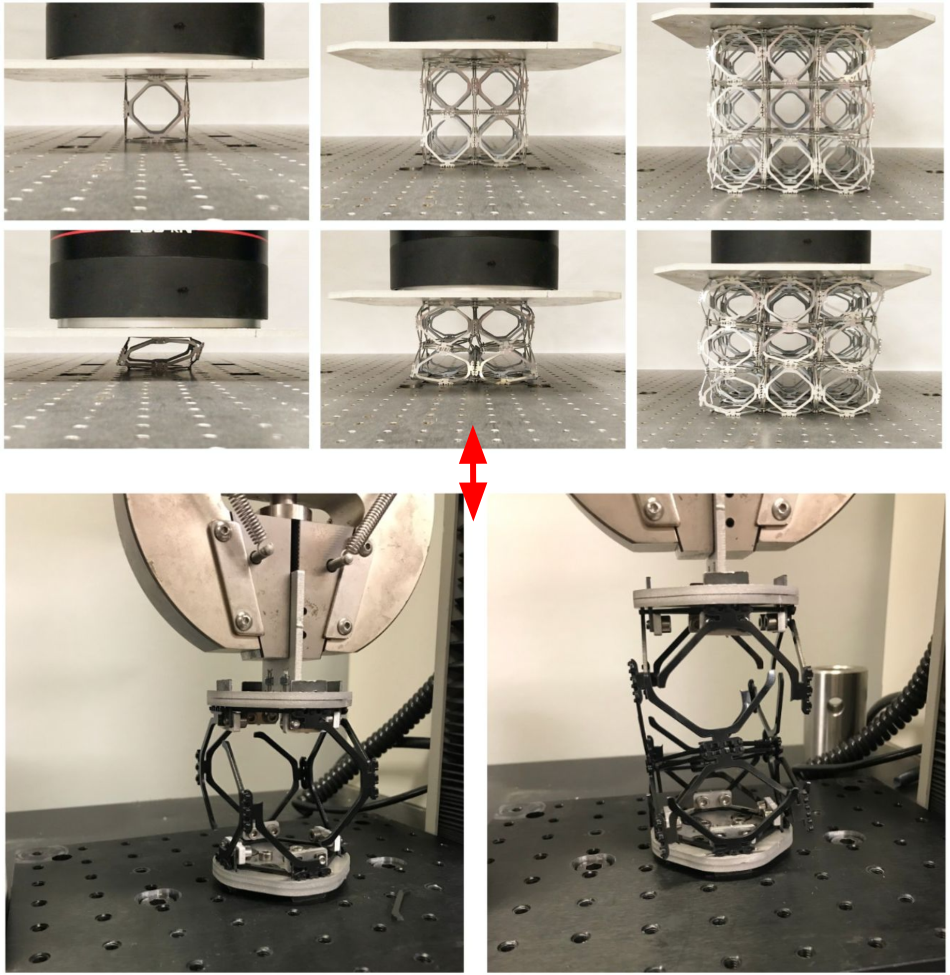
Problem: how do things fail?



Aluminum cuboct voxel failure at 50% strain



CT scan showing buckling in a plate lattice unit cell (credit: A. Parra Rubio)

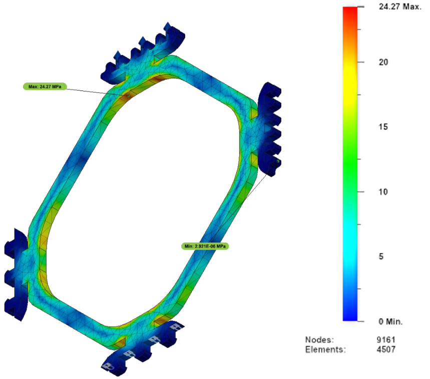


Why discrete elements?

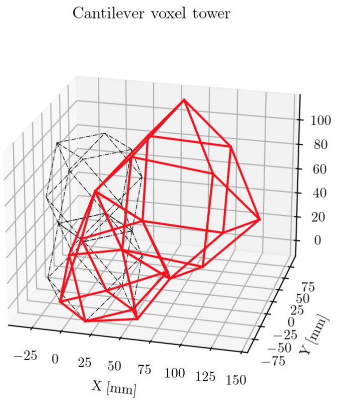
Existing FEM:

- These tools cover the linear response well
- Vary in how well nonlinear behavior is handled (if at all)

→ Existing tools are not good at predicting how things fail



E.g, Fusion's non-linear static stress test fails to converge for the above example (linear static load case)



Frame3DD struggling with large displacements

This project

Problem 1:

Extend the prior DEM beam bending problem to 3D and adapt for easy geometry modification

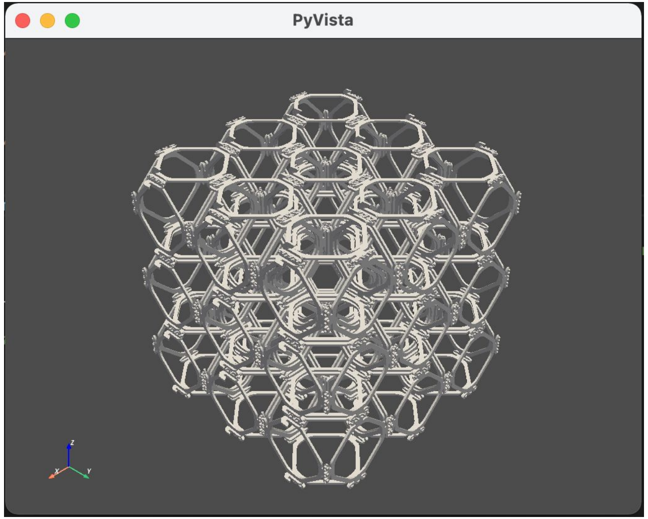
Problem 2:

Model tensile and compressive failure modes for a single voxel

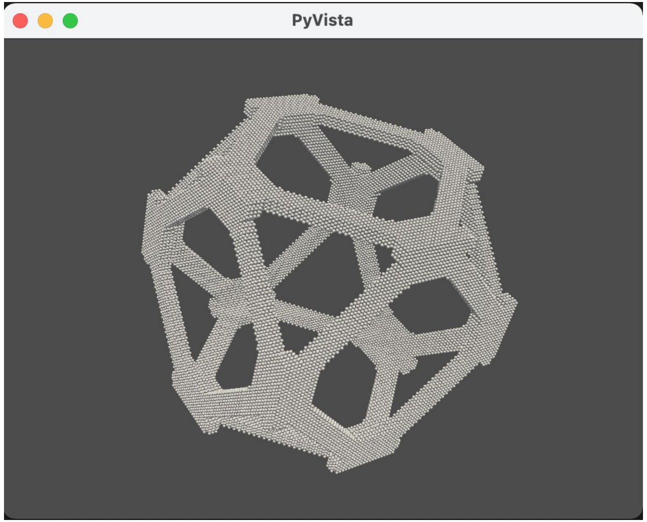
- Compare against real data
- Compare linear response against FEM and simplified beam models

The approach

Geometry: Pyvista handles intaking an stl (or other standard 3D representation) and populating it with particles.



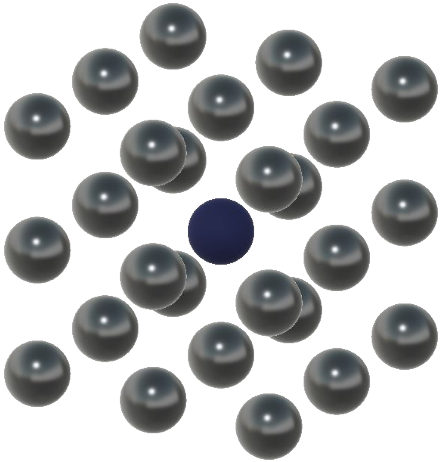
3x3x3 voxel structure mesh



1 voxel, N = 20622 particles

The approach

Particle grid:

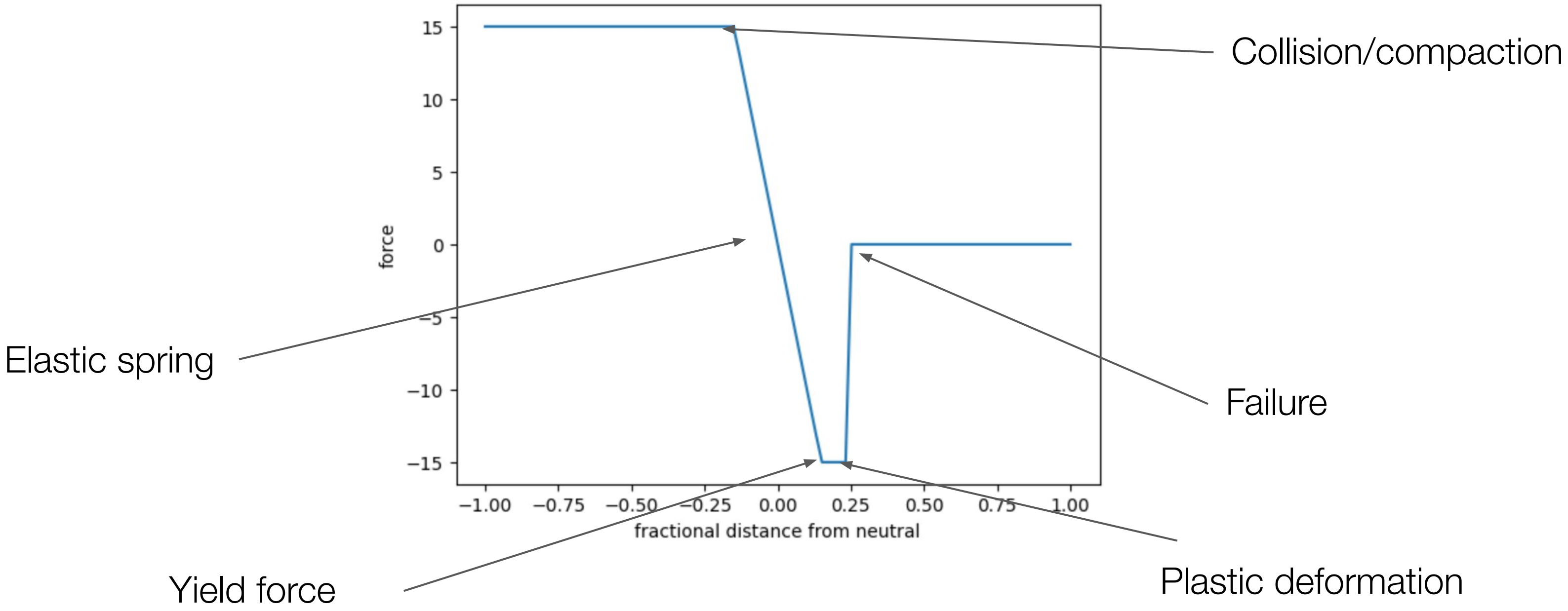


→ Store the indices of the nearest neighbors for each particle → no updating (assuming relatively rigid system, w/o large self-collision)

→ Initial distance is neutral distance for the system

The approach

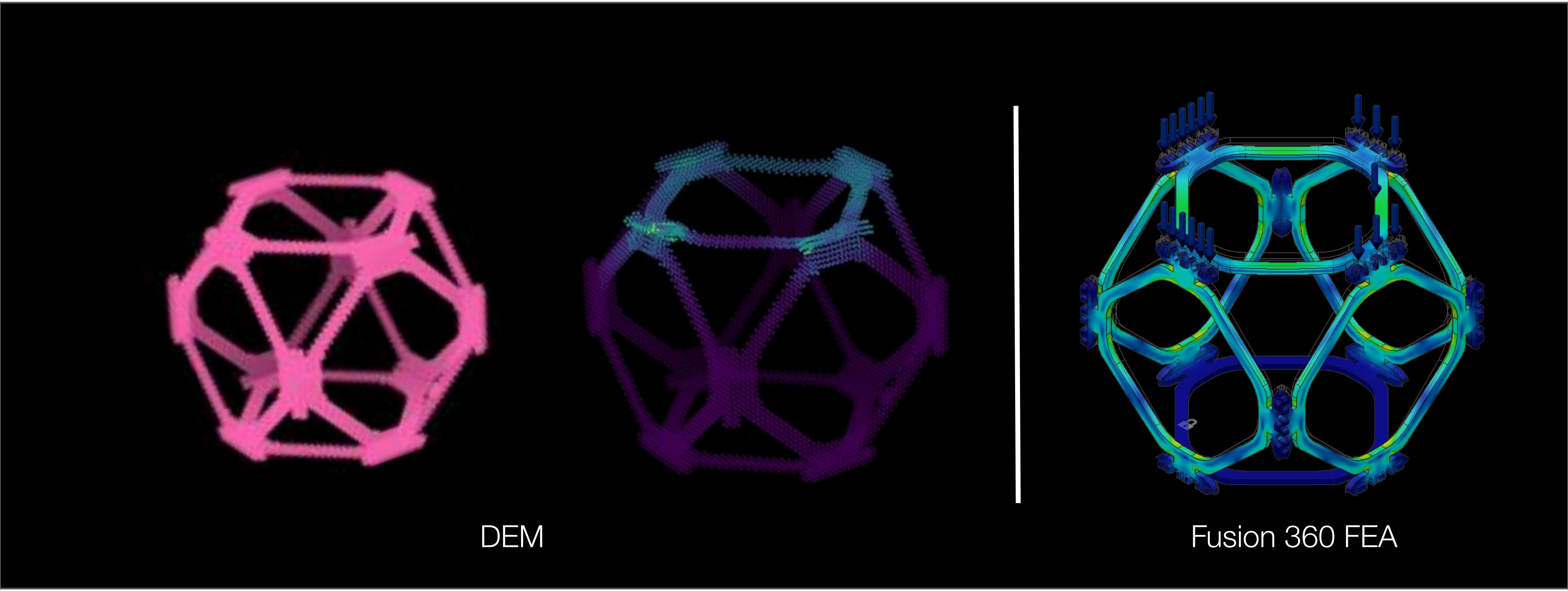
Force law: piecewise linear, e.g:



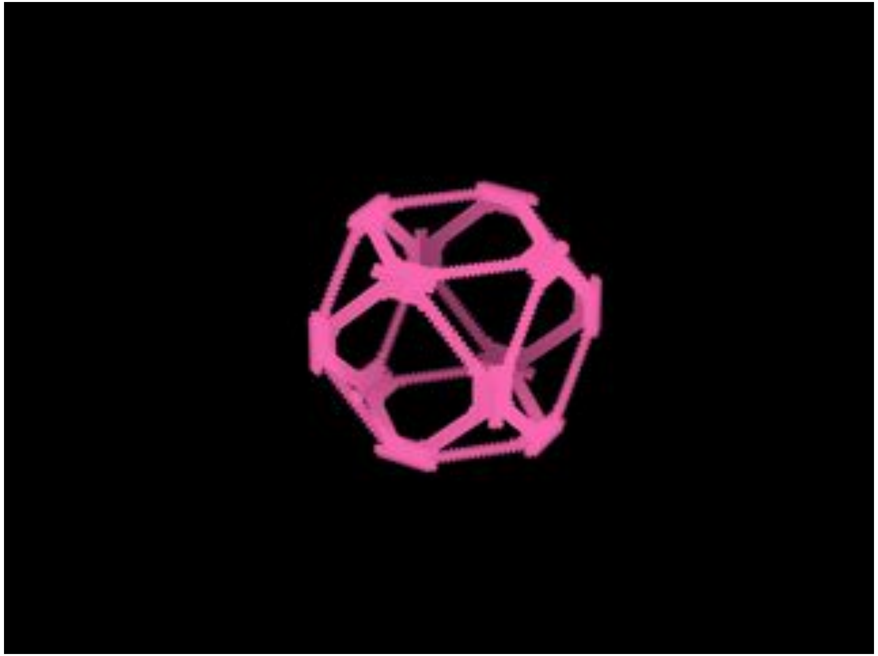
To apply constant increasing force or to apply constant increasing displacement?

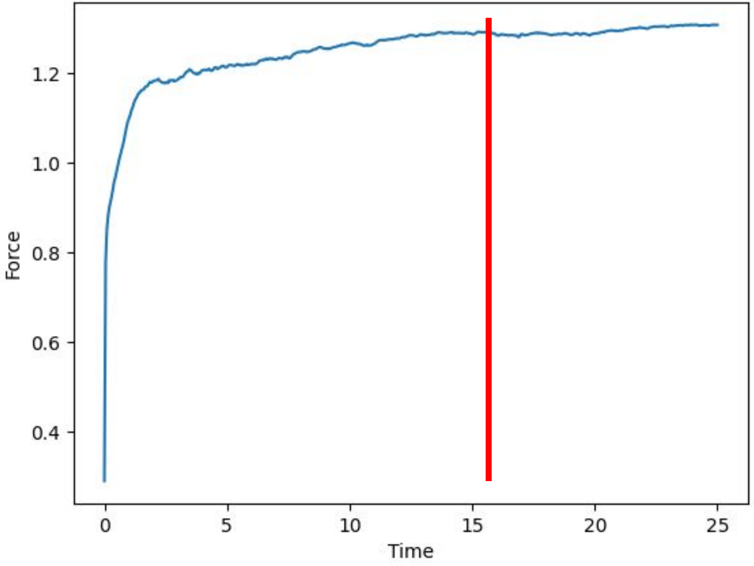
→ What measurement to extract/from where to generate force/displacement graphs?

Compression (low displacement), 7128 particles, uniform force applied over top surface



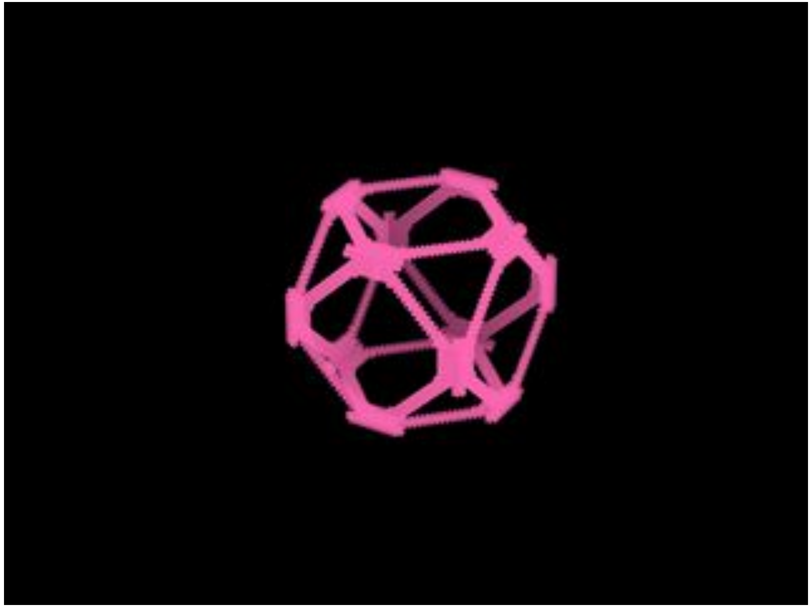
Constant displacement, long flat plastic region

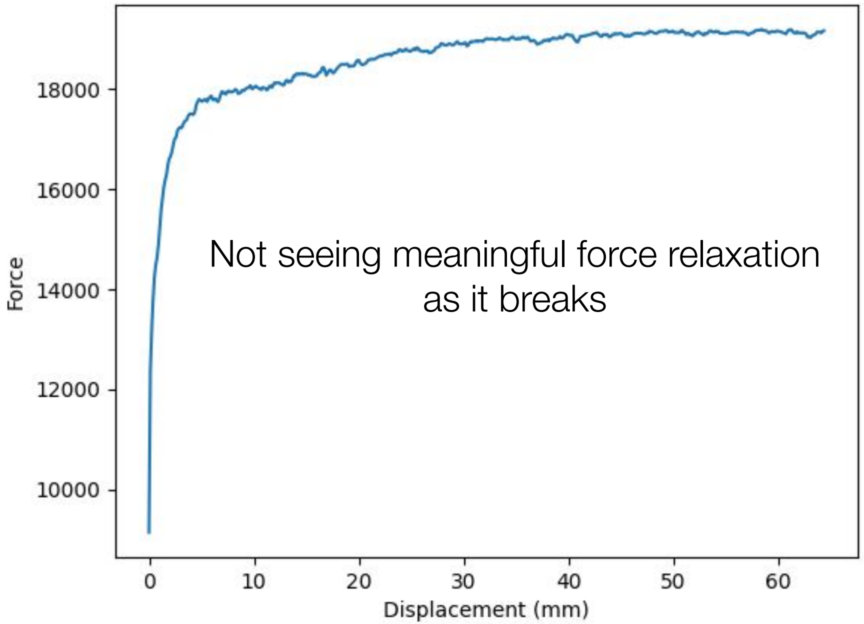




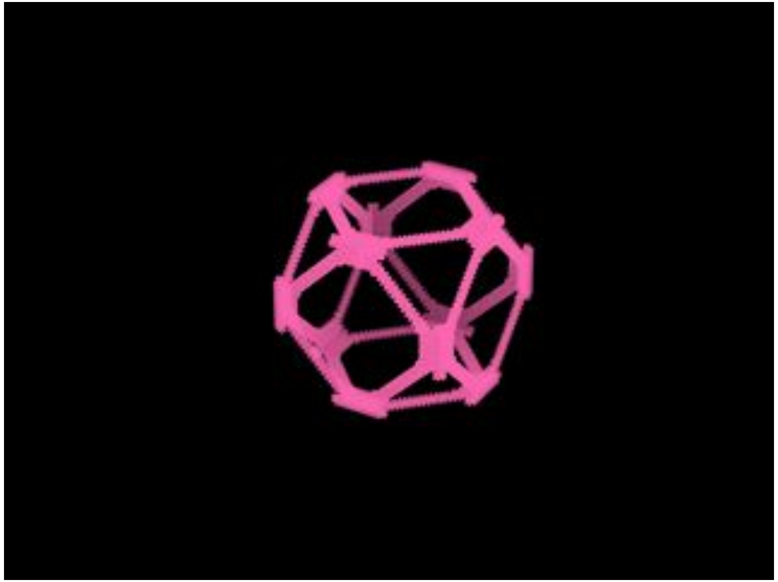
*axes have meaningless units

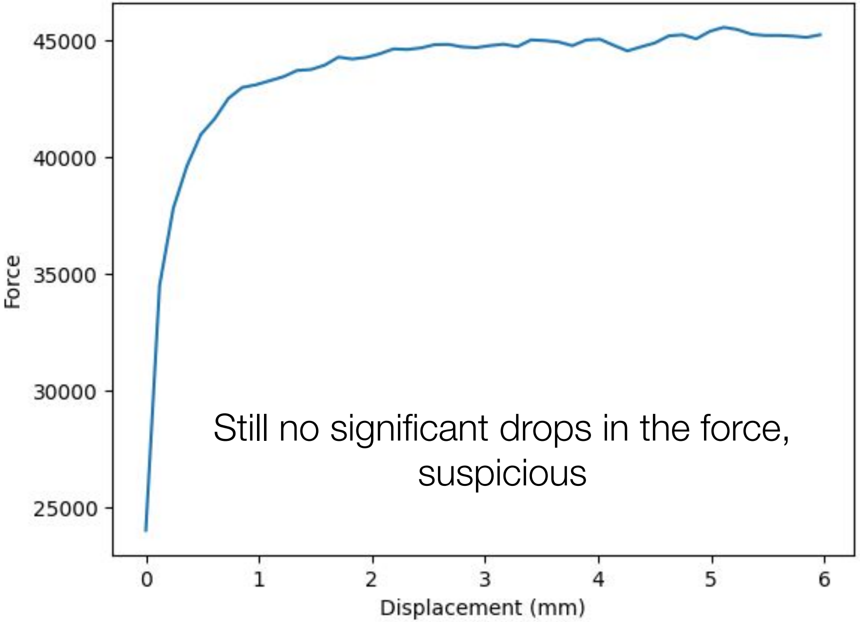
Same force law to beam failure



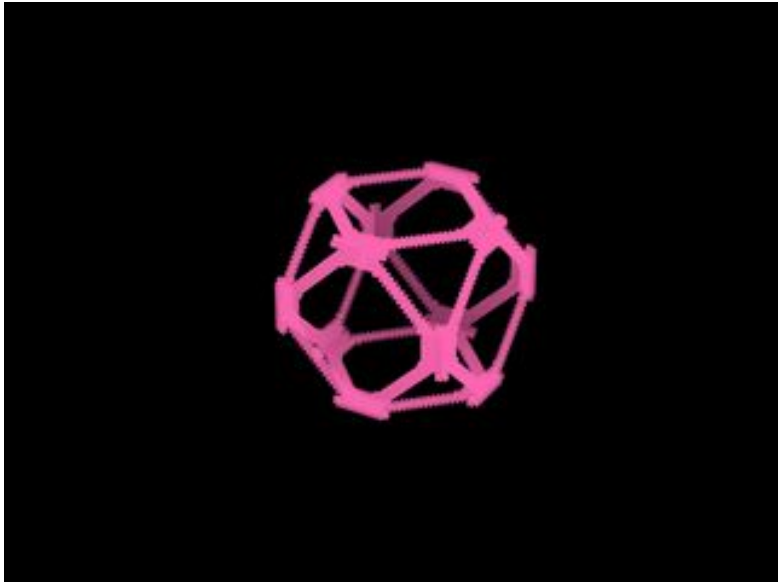


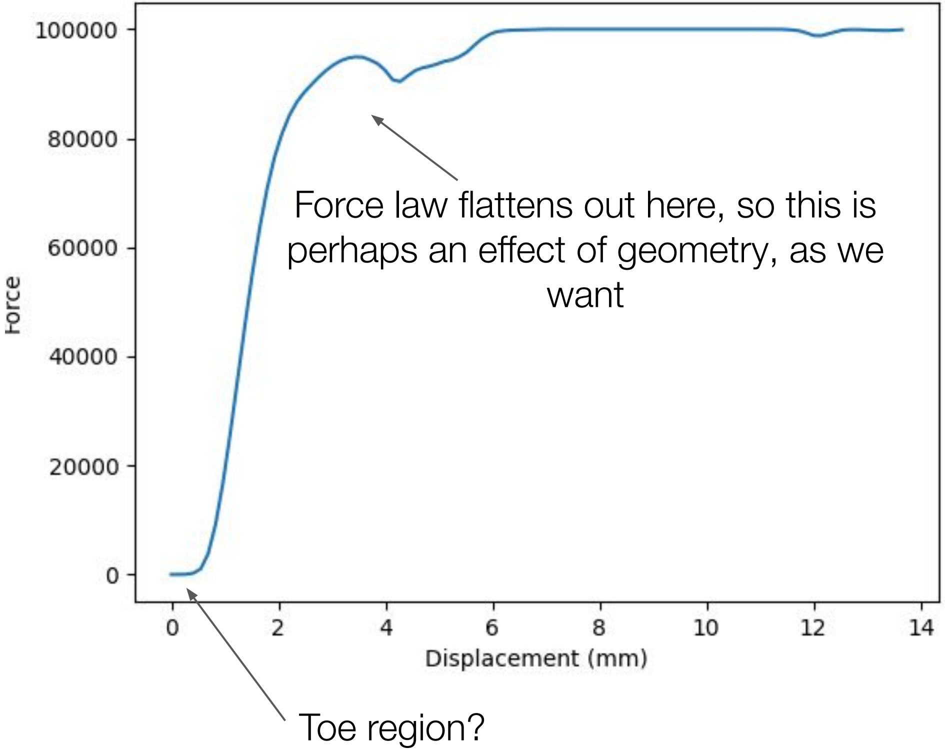
Reducing the length of the plastic region

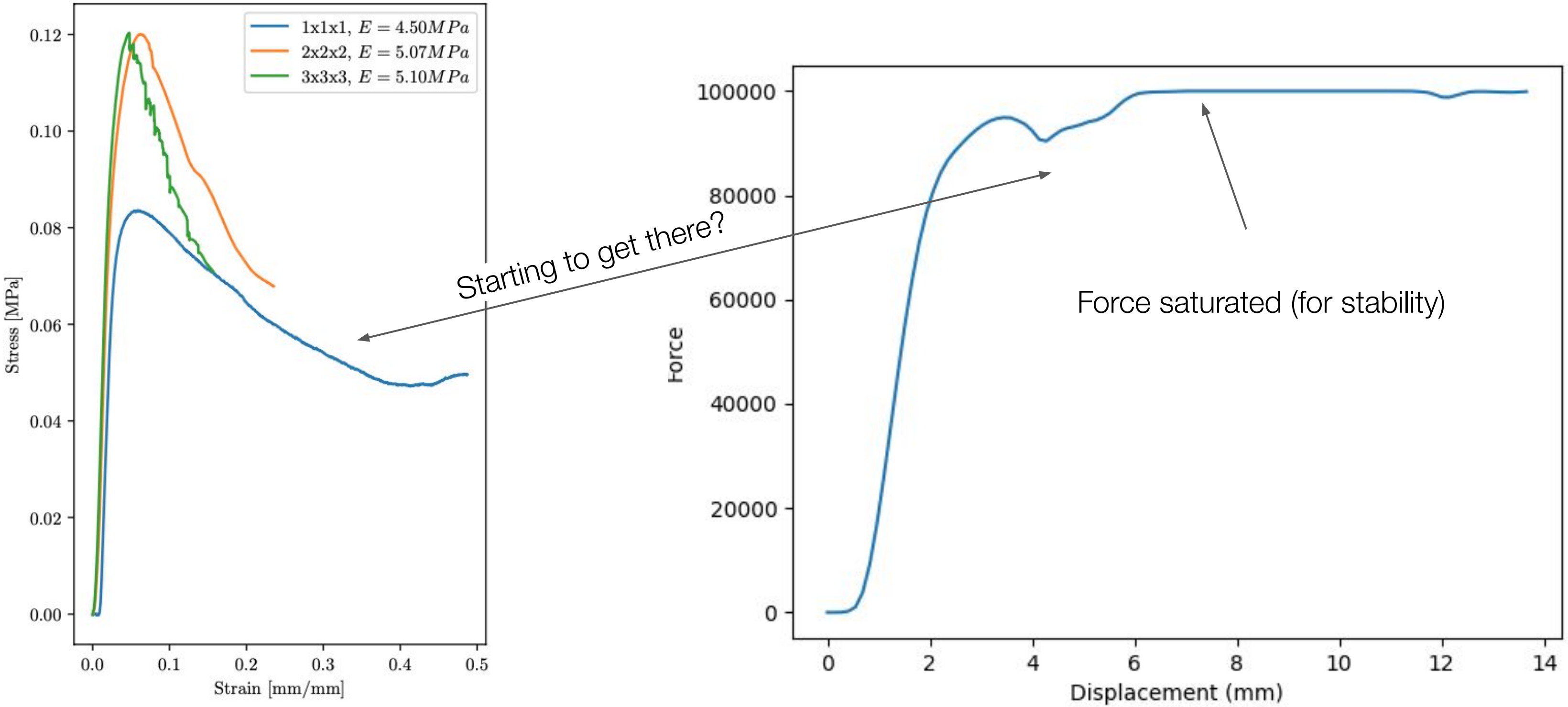




Extending elastic region

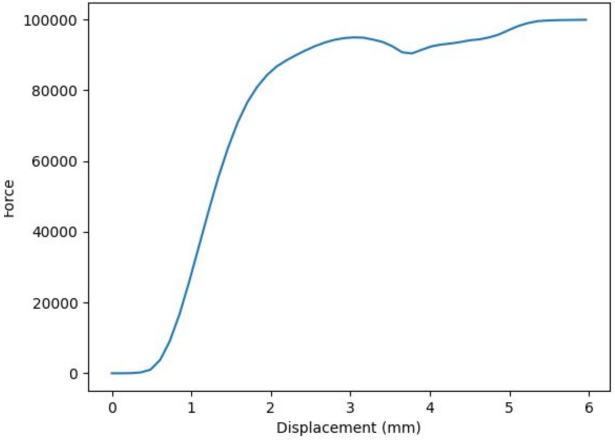




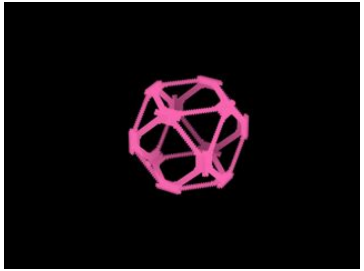


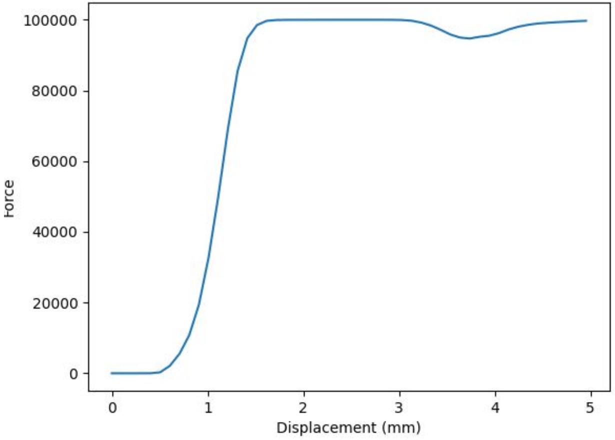
Measured engineering stress-strain for aluminum voxels (NxNxN)

Increasing particle count

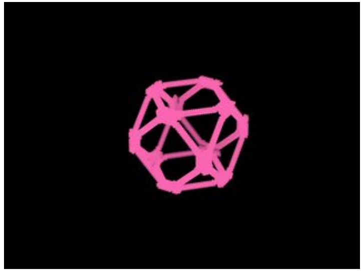


7128 particles





13812 particles



System goes unstable at ~20000 particles w/o substantial changes to behavior

What works? What doesn't?

Basic simulation environment is set up, but force laws are not sufficiently close to reality to be useful

Force law requires a lot of tuning, is not automated

Particle counts are too low, need to switch to GPUs (larger particle counts are failing currently anyways)

Next steps

Run larger particle counts on GPU

Bin searching for adaptive nearest neighbors (to handle later self collisions, in e.g. buckling)

More automatic force law generation from data

Multi-material system?

What does it do?

Discrete element simulation of voxels, aimed at examining failure modes

Who's done what beforehand?

There are a lot of tools for simulating these structures, but not so many focused specifically on failure. The DEM work from Erik and Neil in the DEM chapter is the most direct relationship.

What materials and components were used?

This is written in Python and makes use of taichi for acceleration, pyvista for handling meshes, numpy/scipy for various array calculations, and matplotlib for visualization.

What processes were used?

This mostly draws on the discrete elements chapter, though that chapter in turn does draw on the ones before it.

What questions were answered?

A lot of my personal questions about setting up a discrete element method were answered. Working toward an answer for “is DEM suitable for simulated buckling/fracture/etc in voxel structures?”

How was it evaluated?

Mostly qualitatively currently, comparing against real measured results

What are the implications?

If it worked this could be a useful prototyping tool for voxel design.