Mooresville, NC · Alexandria, VA · Huntsville, AL · Barking Sands Kauai, HI · Oklahoma City, OK

*Solutions to Your Most Difficult Problems*

# ML-Enhanced Approach for Topology Optimization Using Cellular Materials

August 12th, 2022

**School:** snell.113@osu.edu | **Corvid:** eric.snell@corvidtec.com

# Motivation

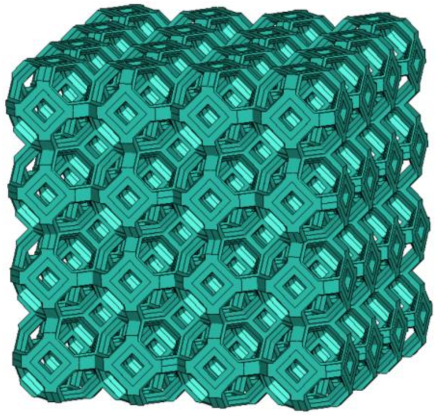
## How can we design cheaper, stronger, lighter parts?

**Existing tools:** FEA, Topology Optimization, AM (L-PBF)

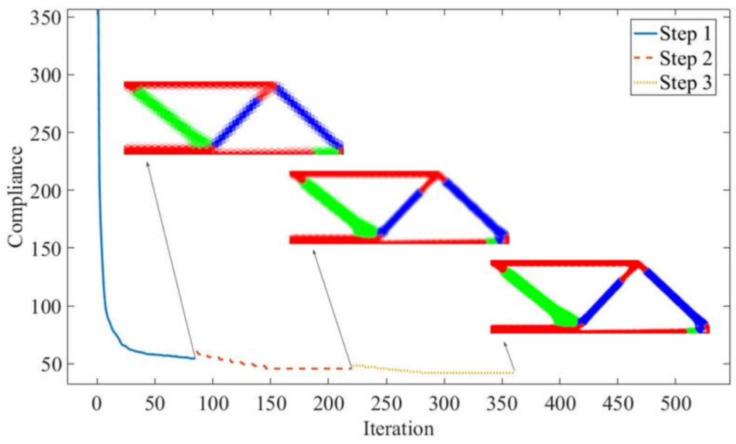
**Idea:** Use small, repeated substructures with partial volume fraction

**Upsides:**
- Reduced mass, material use

**Downsides:**
- Necessitates non-traditional manufacturing and design
- Huge increase in simulation complexity



4 $cm^3$ structure containing 91 tetrakaidecahedra



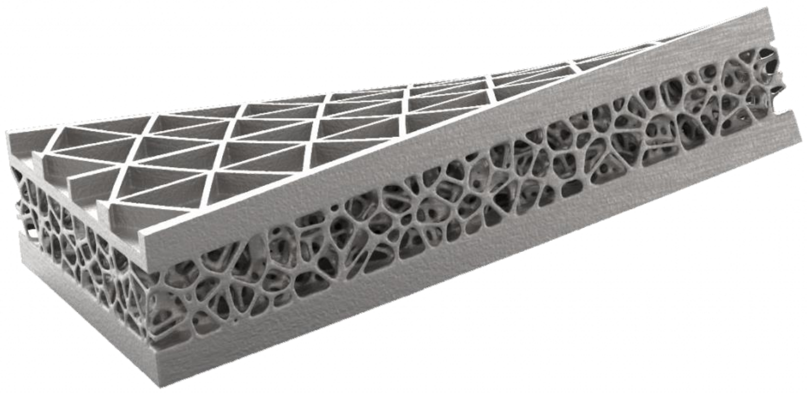
# Potential Applications of Cell. Materials

## Designed parts not limited to void / non-void thresholding

- Use less material (cheaper, lighter weight)
- Highly tailored to desired mechanical response
- **Useful in a variety of defense applications**

## Project focus:

- How can we run simulations and topology optimization using cellular materials?
    - Circumvent expensive computation costs using an approximation

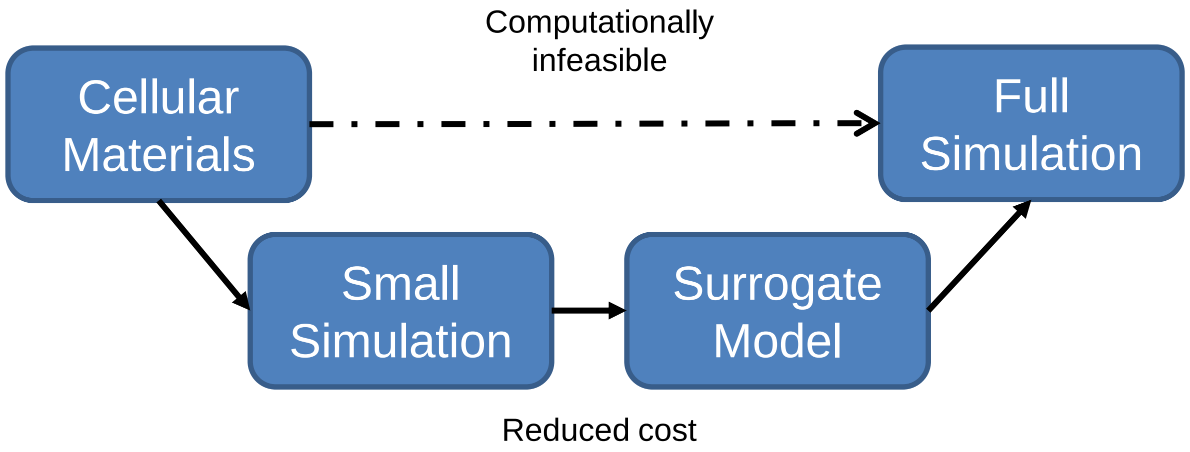


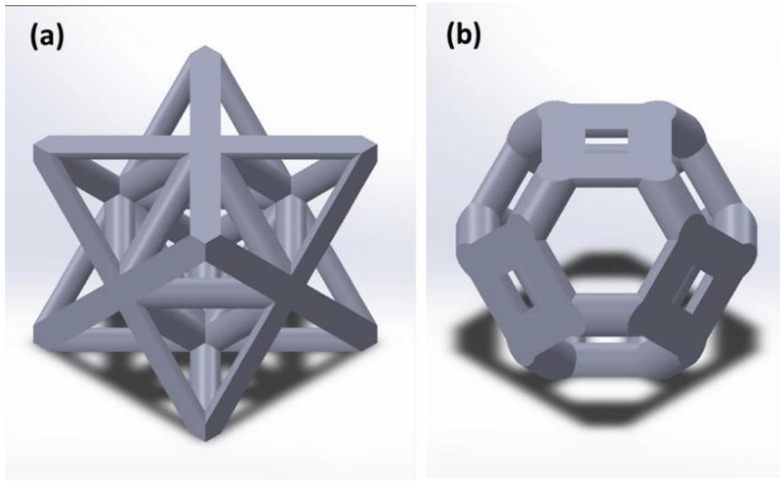
lightweighted part using an organic lattice structure

# Terminology

## Key Terms:

- **Cellular materials** – materials designed to balance reduced density with reasonable mechanical integrity
- **Surrogate models** – models that simulate more expensive models (such as FEA) at a lower cost
- **Homogenization** – treating similar units identically in order to reduce computational costs
- **Topology optimization** – iterative optimization of the distribution of material with a part to minimum compliance (stiffness$^{-1}$)




**Example cellular substructures:**
a) Complex truss
b) Kelvin Cell (tetrakaidecahedron)

# Key Research

## Xiao *et. al*'s hierarchical, multiscale method:

- Measure stress-strain response of aluminum crystal at various temperatures
- Train Single-Layer Feedforward Network (SLFN) to predict failure / non-failure of RVE
- Use model to iteratively predict conditions in lattice structure
  - Element deletion, updated stress

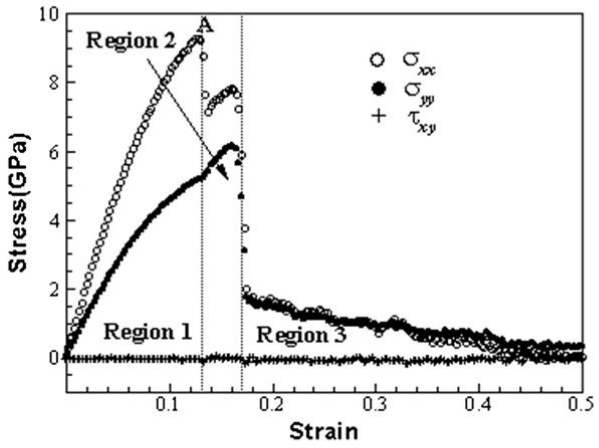



**Fig. 3** Stress–strain relation of an FCC Al crystal in uniaxial tension at 300 K

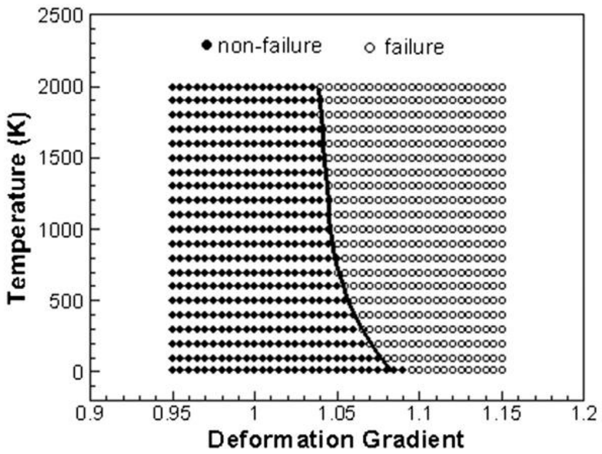
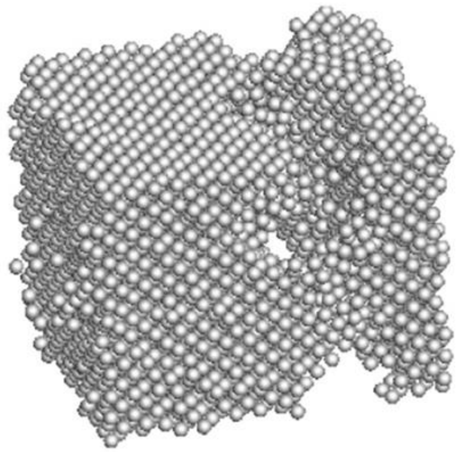



**Fig. 8** The material non-failure/failure interface predicted by machine learning



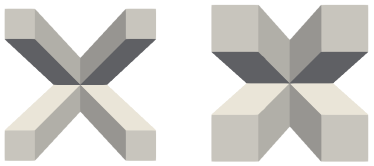
**(a)** $\varepsilon_x = 25\%$

# Approach

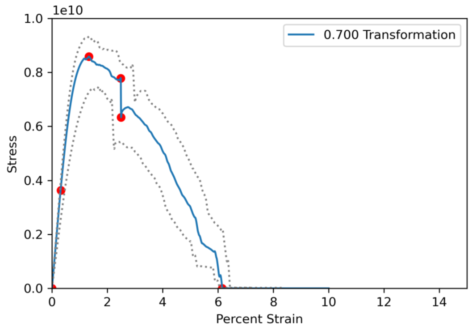
| | Xiao *et. al* | Goal |
|---|---|---|
| **Data** | Stress-strain response of Al crystalline solid | Stress-strain response of simple truss and lattice |
| **Collection Method** | Empirical measurements | Velodyne simulations |
| **Z Variable** | Temperature | Volume fraction |
| **Surrogate Model** | Single-Layer Feedforward Network (SLFN) | ? |
| **Use Case** | Prediction of RVE failure in lattice structure | Stress-strain regression, failure classification for TO |

## SUBSTRUCTURE

VELODYNE



## SURROGATE MODEL



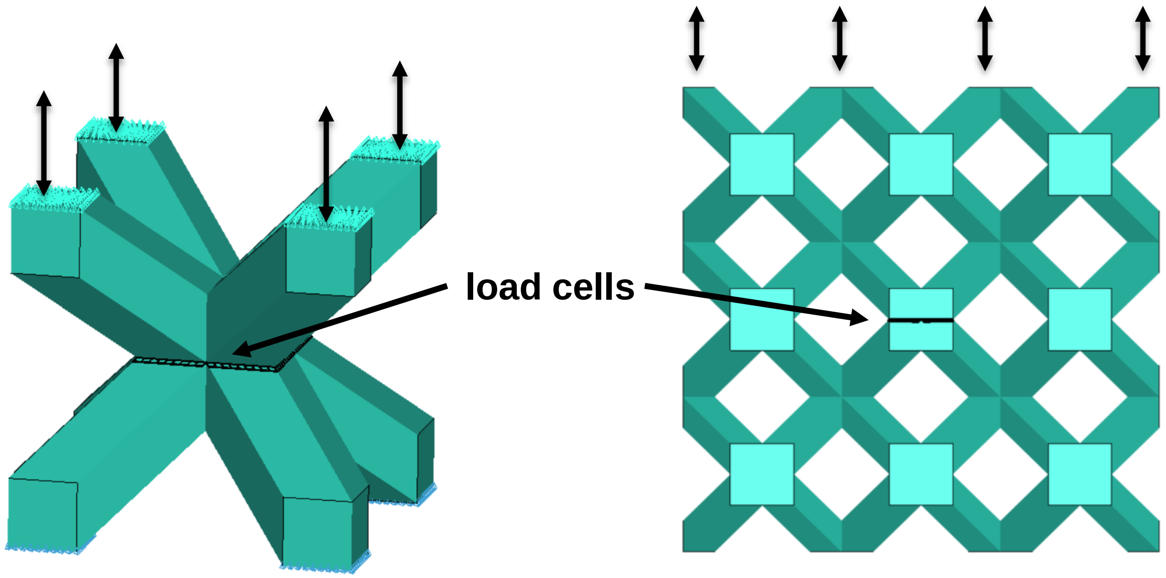
# PART I:

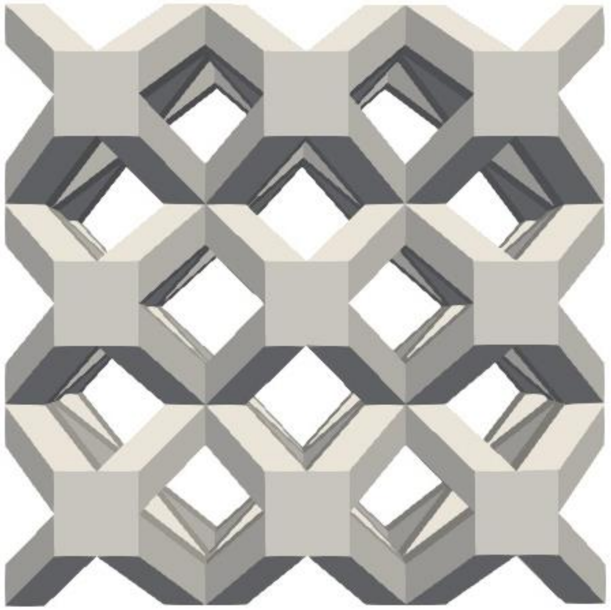
## DATA COLLECTION

# Data Collection

**Quasi-static deformation of unit cells and lattice structures at eight volume fractions**

- Measure stress response to fixed displacement (compression & tension)
- Considerations:
  - displacement speed, lattice size, and mesh size vs. accuracy
  - timesteps in realm of 1e-9



0.352 VF @ 7.5 cm / sec compression

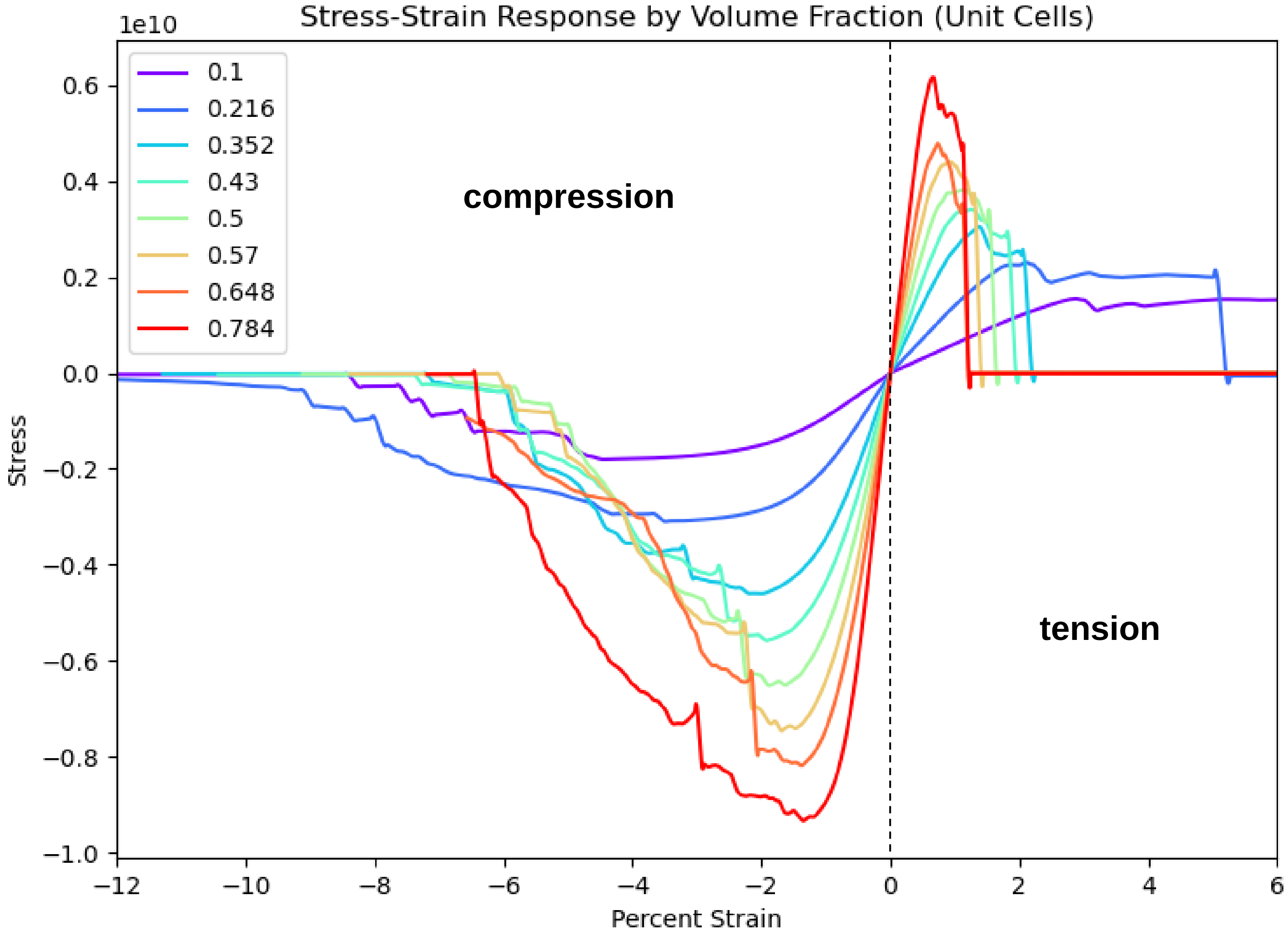
# Single Unit Cells

**Volume Fraction:**

deform 1 cm truss
at 0.5 cm / sec

**0.352**
**compression**

**0.648**
**compression**

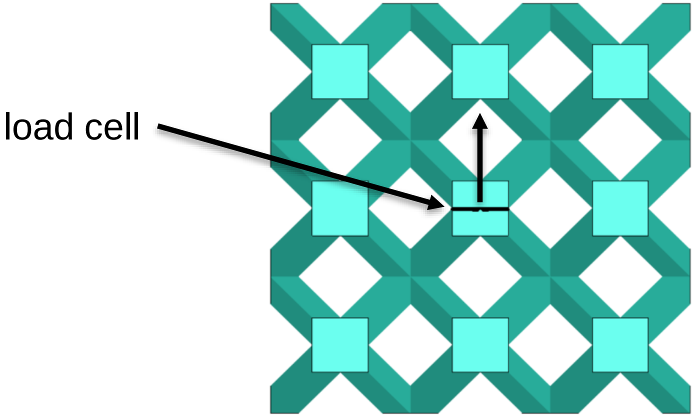


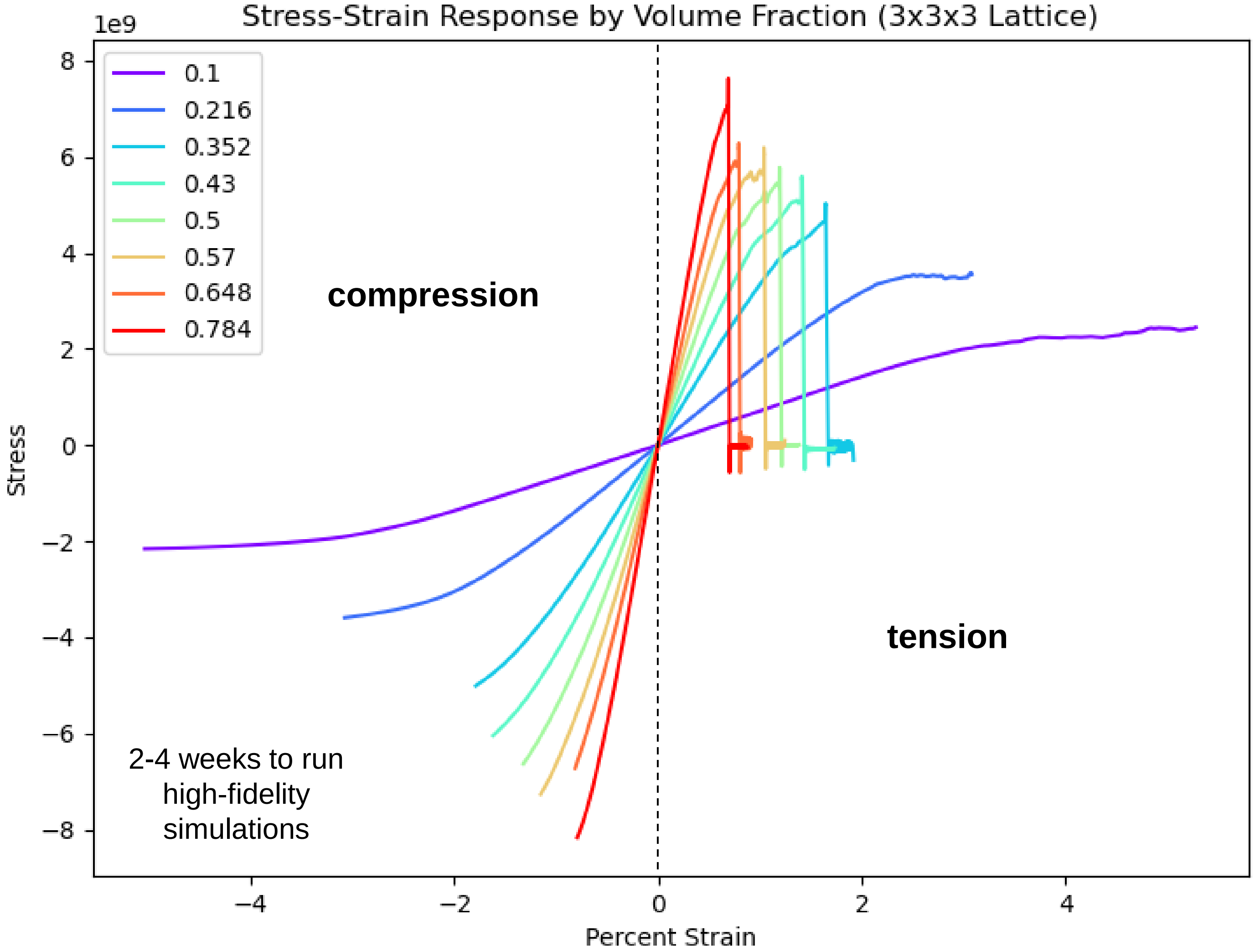
# Lattice Structures

**Volume Fraction:**

deform 3 cm lattice
at 1.5 cm / sec

**0.352**
**compression**
**(clip view)**



# PART II:

## ANALYSIS

# Interpolation

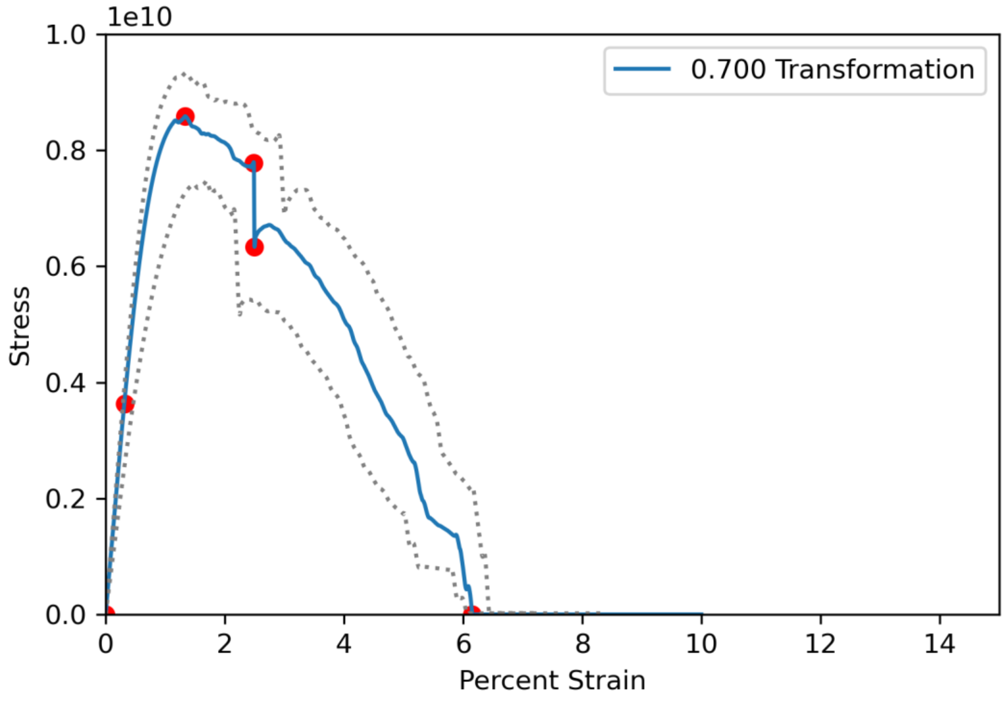
**Knowing how a few structures behave, can we accurately predict failure / non-failure and calculate stress for any volume fraction?**

Interpolate stress-strain response and create models using as few references as possible.

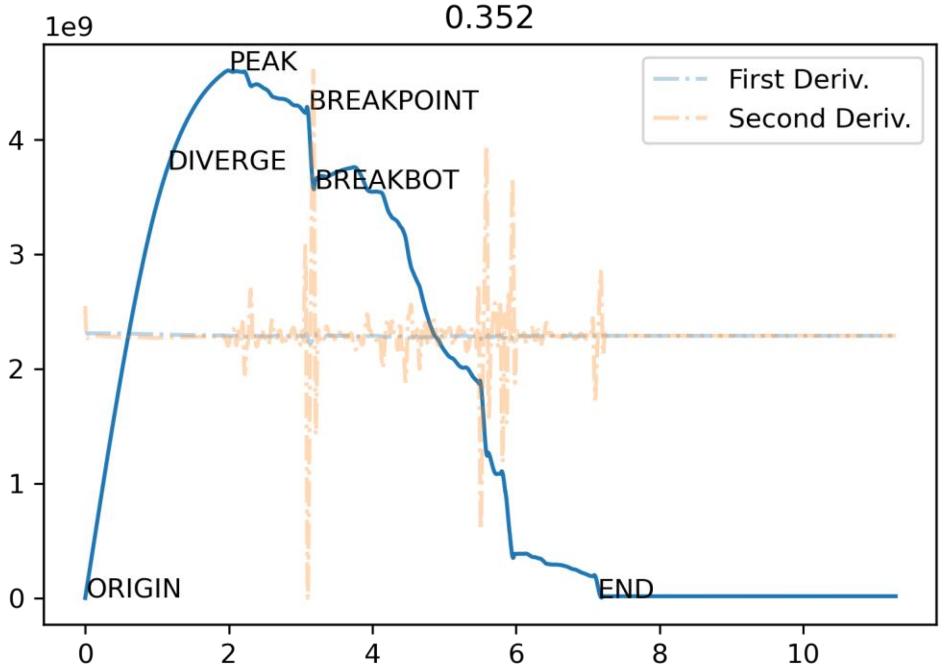
**Method:**

- Model, mesh, and run Velodyne simulations at a few reference volumes, then…
  - Algorithmically locate key points on reference curves (< 5 seconds)
  - Create models for new key point locations (< 3 seconds)
  - Transform data to create curve at requested VF (~ 2.3 seconds)

- Benefits:
  - No need for CAD, meshing, and additional Velodyne runs
  - Save time and money



# Identification of Key Points




| | VF | Point | Strain | Stress | LogStrain | SqrtStress |
|---|---|---|---|---|---|---|
| 0 | 100 | Break Bottom | 6.59875 | 7.647763e+08 | 1.88688 | 27654.6 |
| 1 | 216 | Break Bottom | 7.97500 | 8.869313e+08 | 2.07631 | 29781.4 |
| 2 | 352 | Break Bottom | 3.18800 | 3.571769e+09 | 1.15939 | 59764.3 |
| 3 | 430 | Break Bottom | 2.63475 | 3.982211e+09 | 0.968788 | 63104.8 |
| 4 | 500 | Break Bottom | 2.35225 | 4.930766e+09 | 0.855372 | 70219.4 |

...

## Break point selection:

Consider proximity to peak stress as well as value of second derivative

- Number of extrema to consider
- Savitzky-Golay filter width
- Proximity threshold

## Benefits:

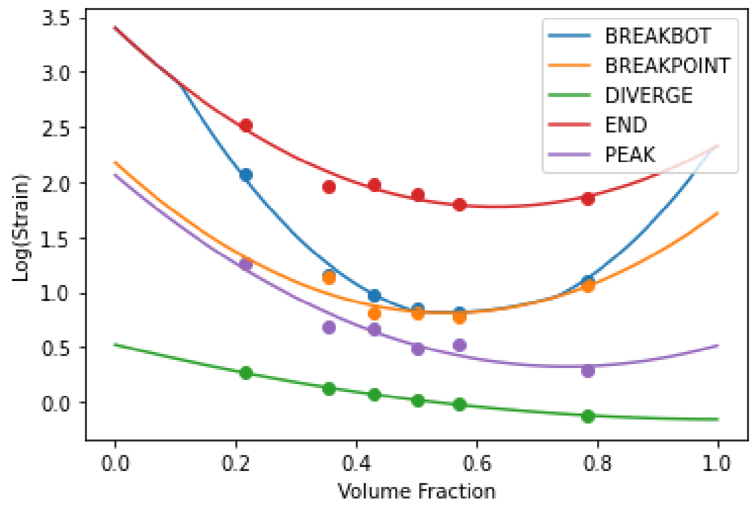
Algorithmic design rather than black-box approach allows for manual enforcement of point ordering, custom parameters

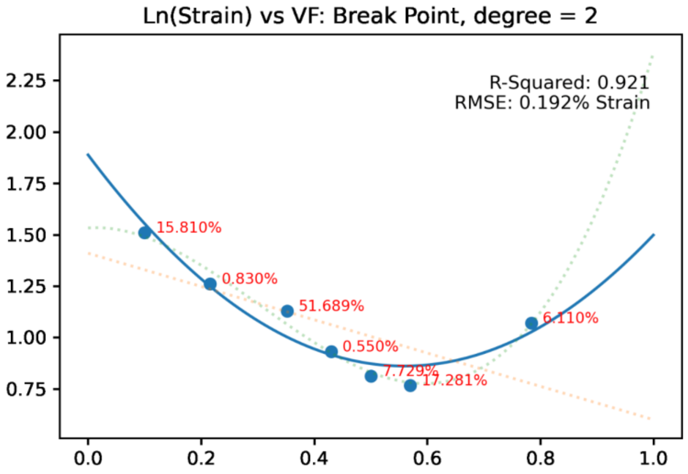
# Polynomial Regression

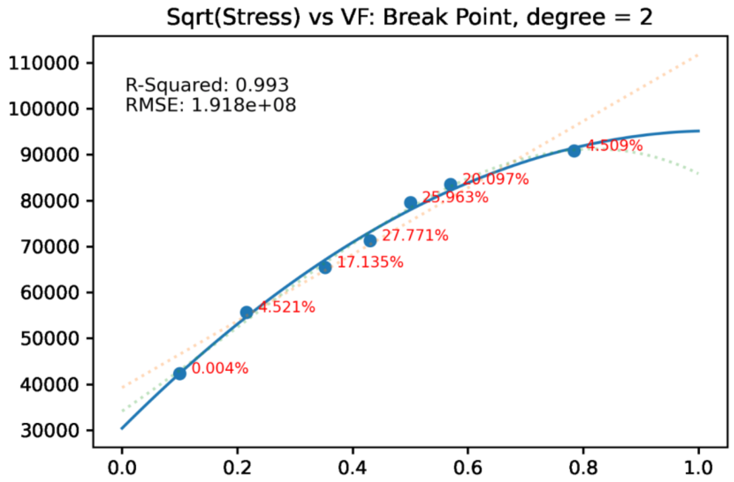
**For each point, we can do two polynomial regressions, one for stress and the other for strain.**

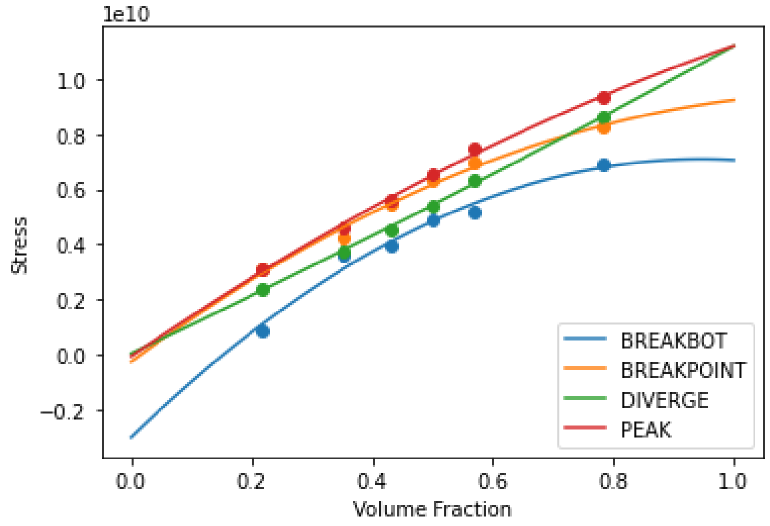
## Findings:

- Stress responses for key points are well behaved
- Strain responses are decent
  - RMSE < 0.2% strain for peak stress and break point
- Enforce prediction ordering

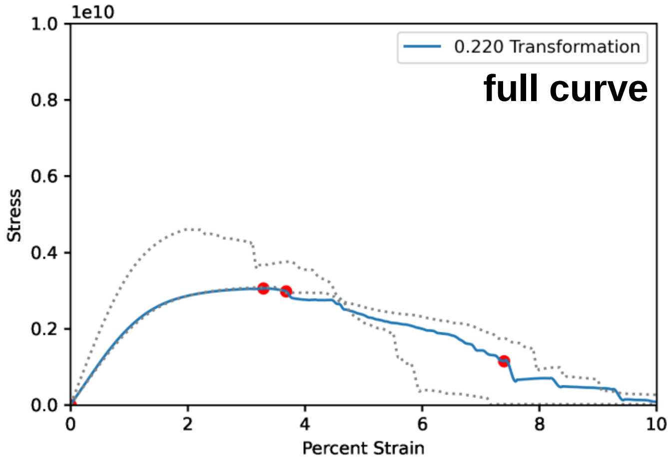


# Interpolated Stress-Strain Curves

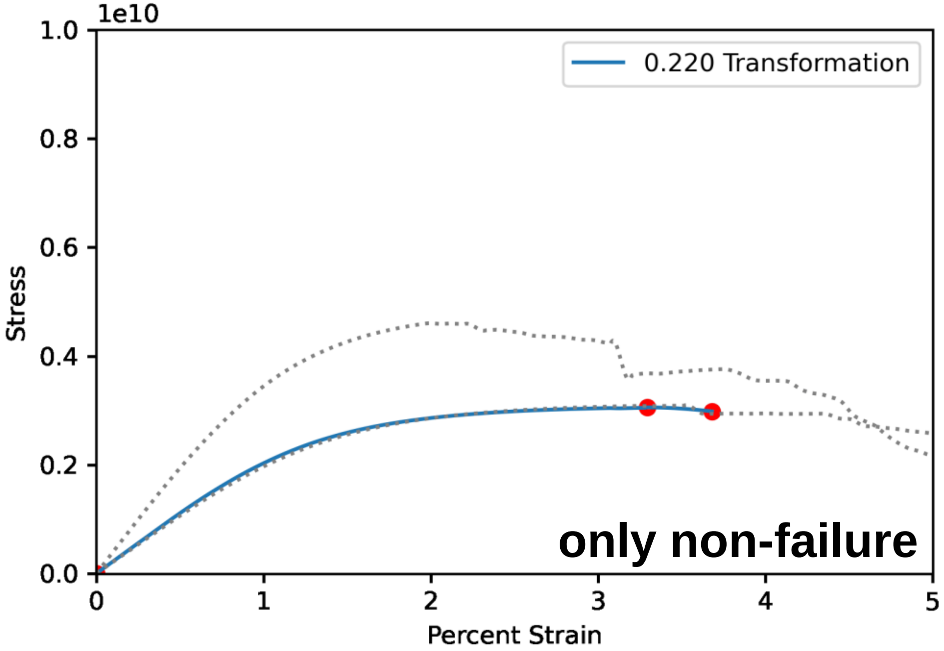



## Method:

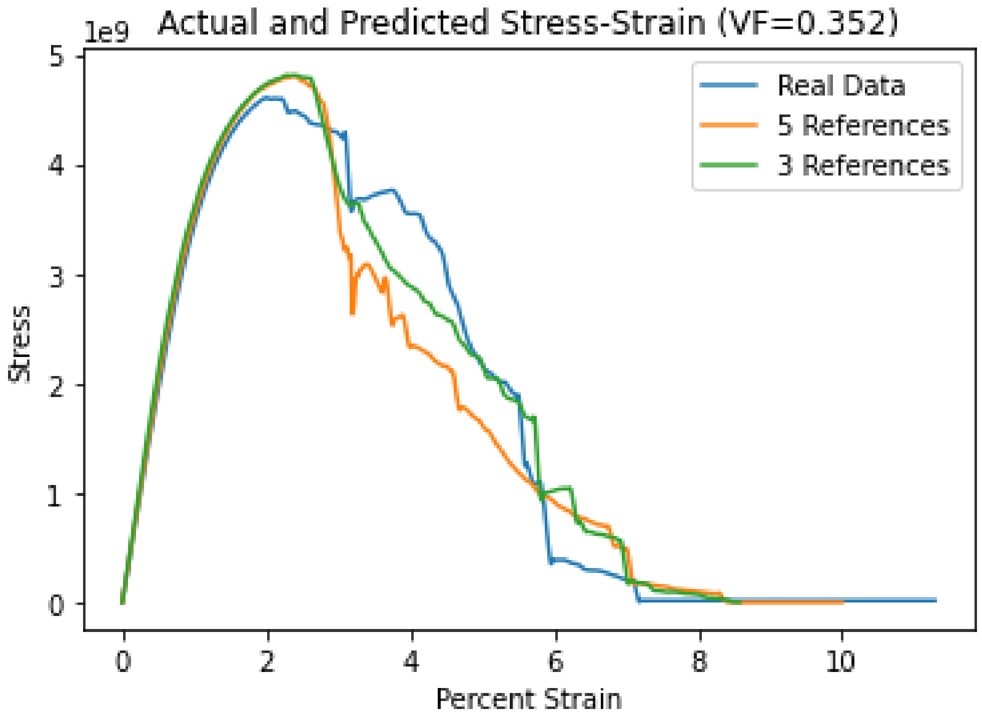
- Create stress and strain models for each key point
- Recalculate key points at new VF
- Take weighted average of two nearest reference curves and transform onto new key points

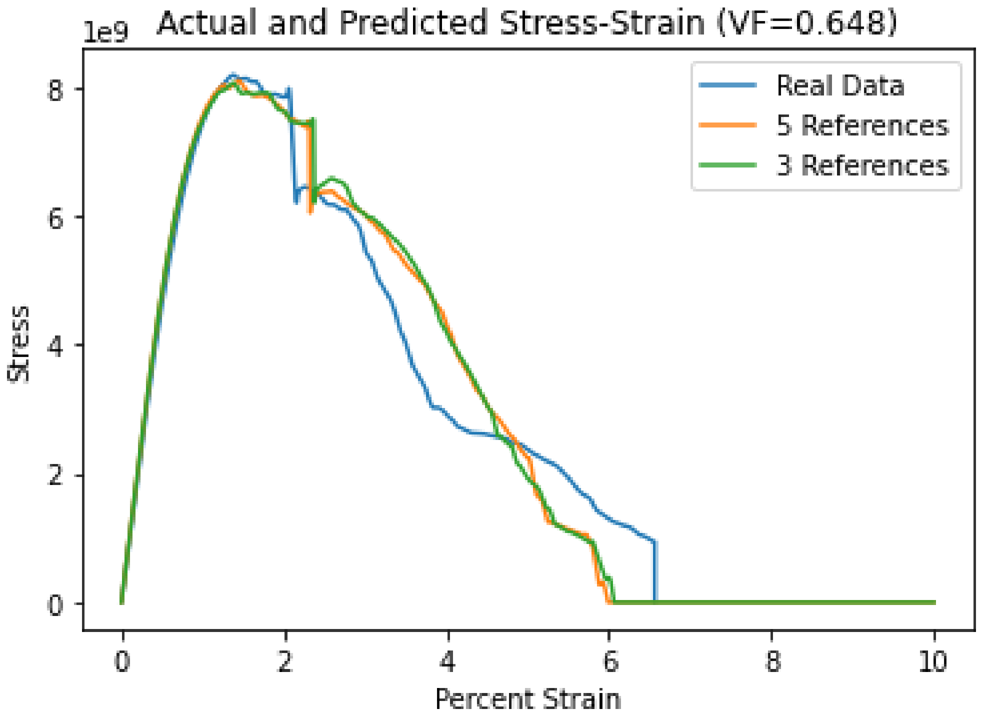
## Result:

- Interpolated curves allow us to:
  - Classify failure / non-failure for any intermediate volume fraction
  - In case of non-failure, predict stress response



# Prediction by Number of Reference Curves

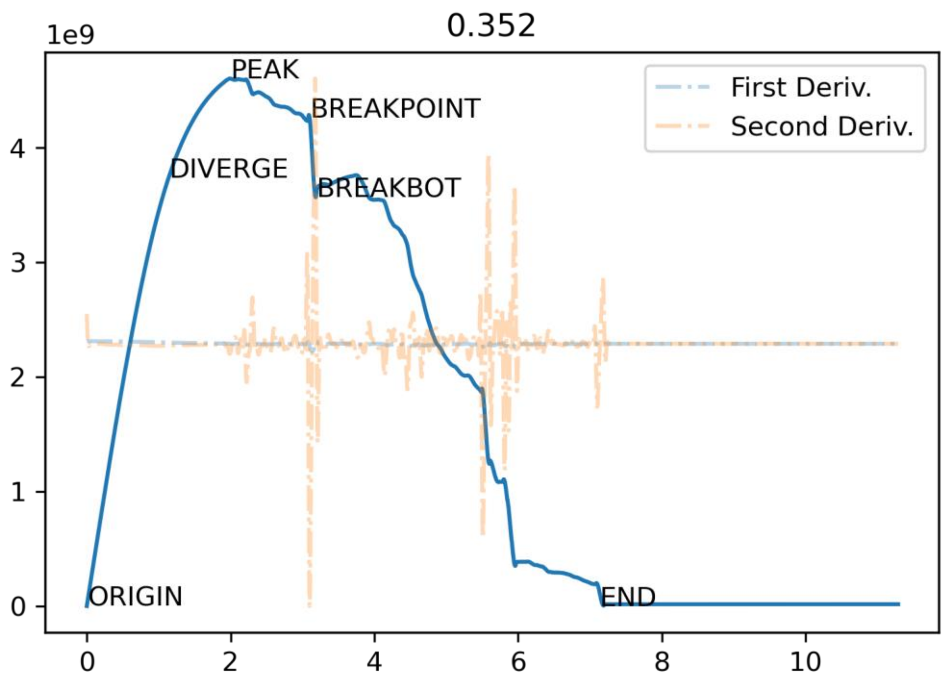


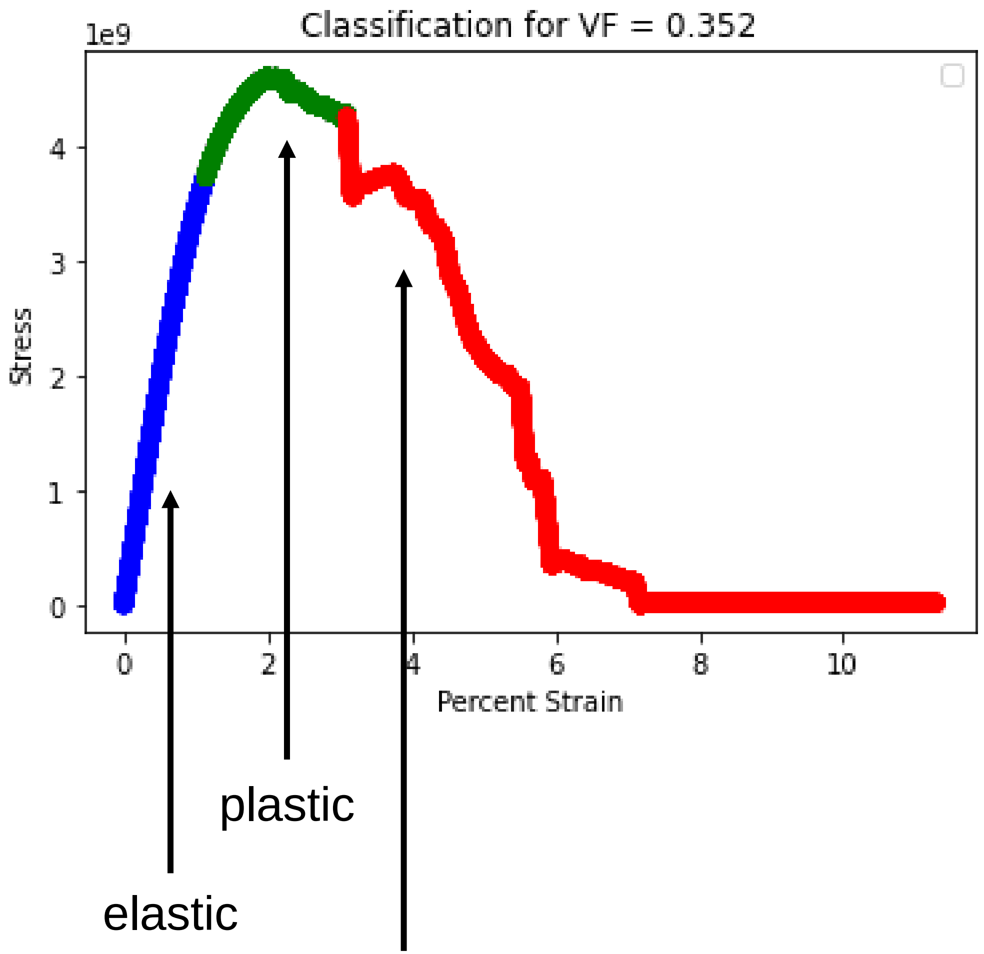




**3 References:** 0.216, 0.5, 0.784
**5 References:** 0.216, 0.430, 0.5, 0.570, 0.784

Three reference values appears sufficient.

# Classification Using Curves



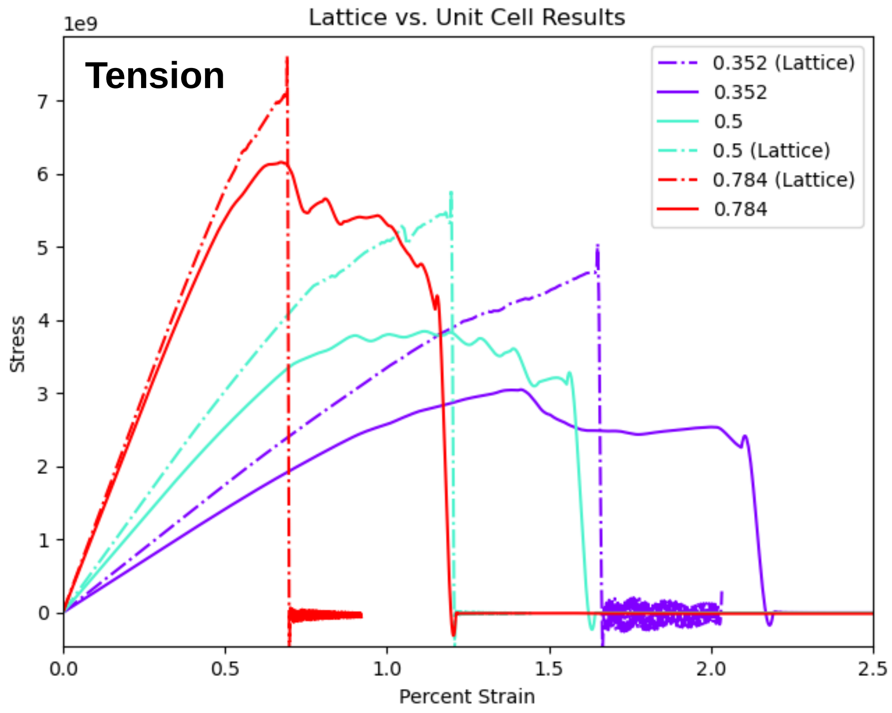



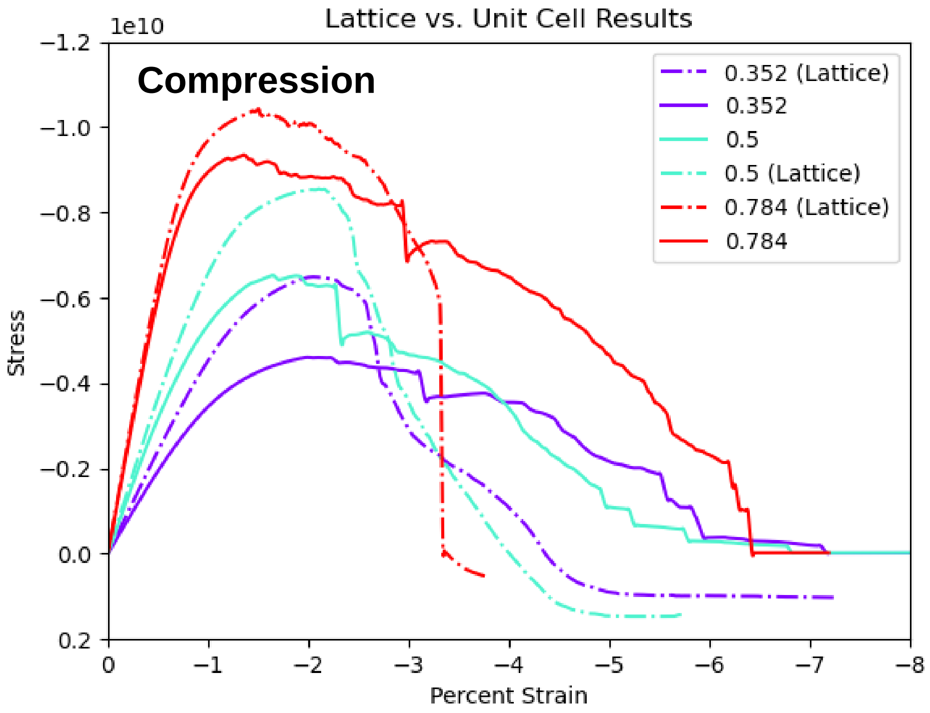

0.352 VF, compression @ 0.5 cm / sec

# Lattice vs. Unit Cell Results

**The stress-strain response is different in lattice structures—can we use our single results to predict the far more expensive lattice results?**

- Is there a consistent transformation from single → lattice?

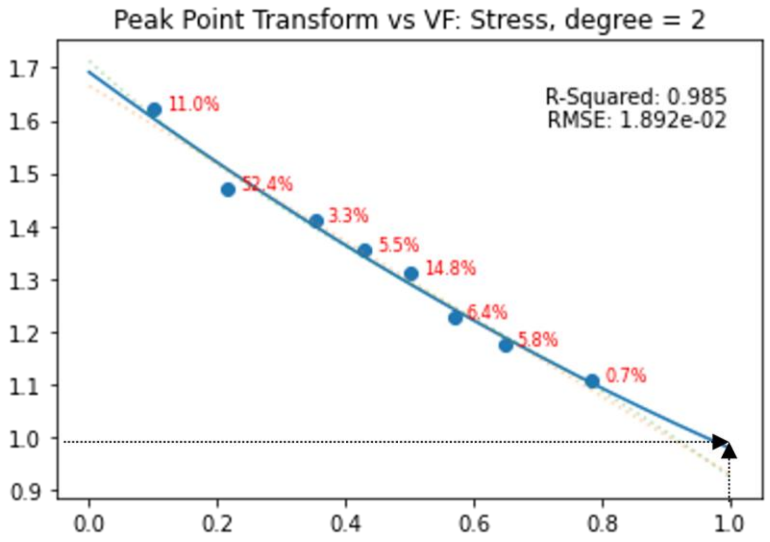


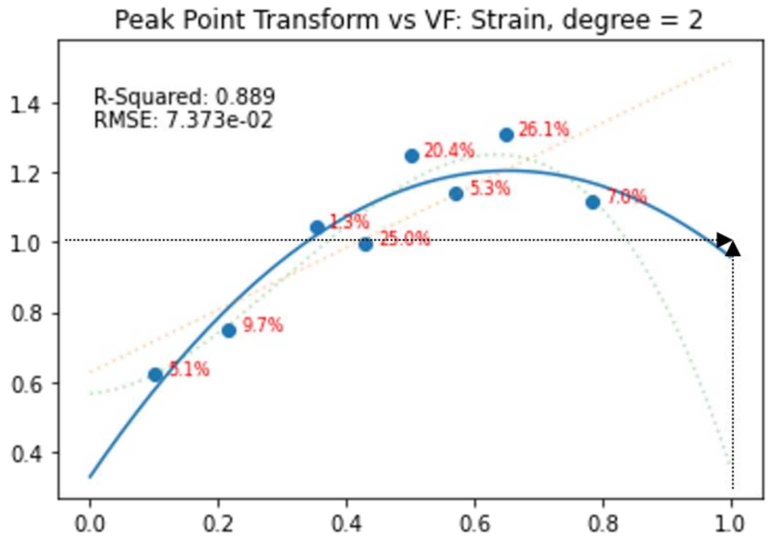
# Transformation: Single → Lattice

## Data from:

- 7.5 cm / second lattice compression vs. 0.5 cm / second unit cell compression




At a VF of 1.0, there is no functional difference between a “lattice” and a unit cell (cube), so we can use (1,1) as a reference.




- Can determine stress transformations, but strain is more complex—too high of a compression speed?
  - Need more than 1 run to define transformation
- If we can determine this relationship for one substructure, does this apply for other substructures?

# PART III:

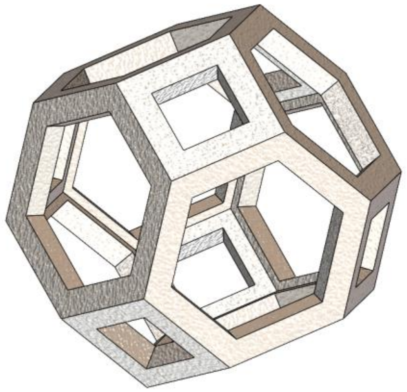
## SUMMARY

# Surrogate Modeling

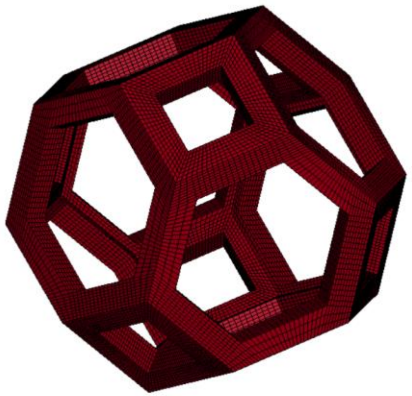
## DATA CREATION:

- Multiple VFs
- Compression, tension

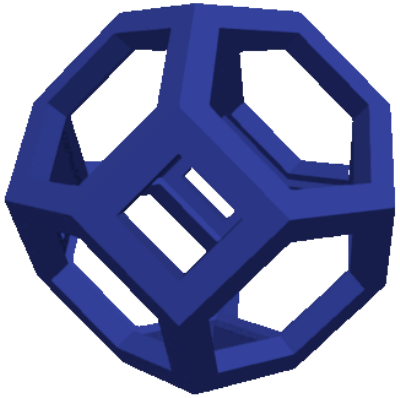
**CAD**



**Cubit, LS-PrePost**



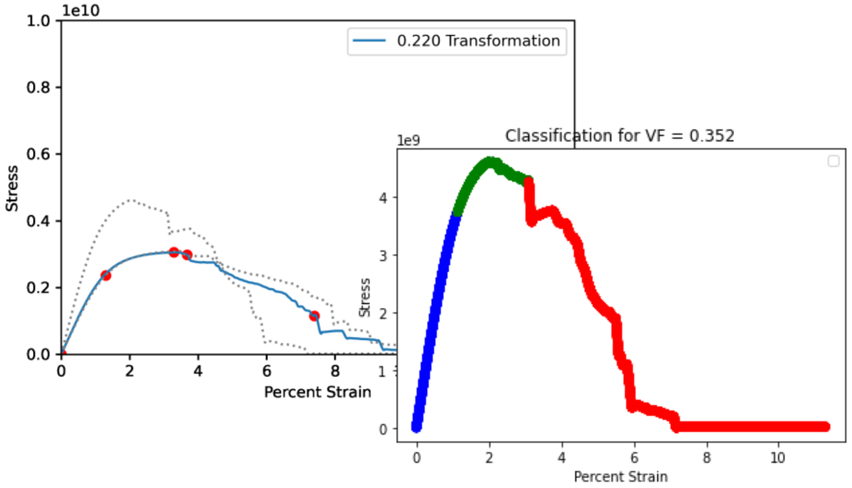
**Deformation**



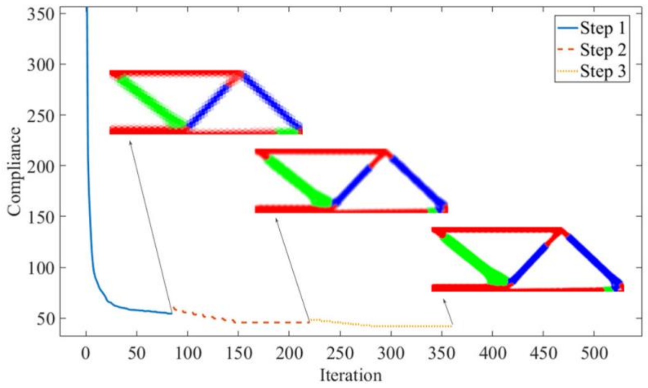
## MODEL CREATION:

- Interpolation
- Classification
- Model evaluation

**Build Model**



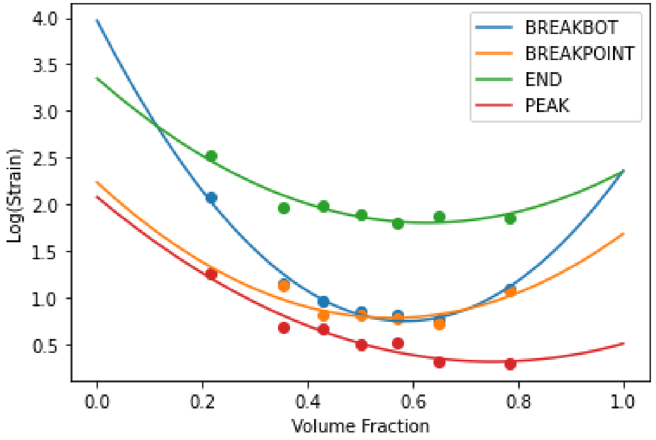

**Employ Model**



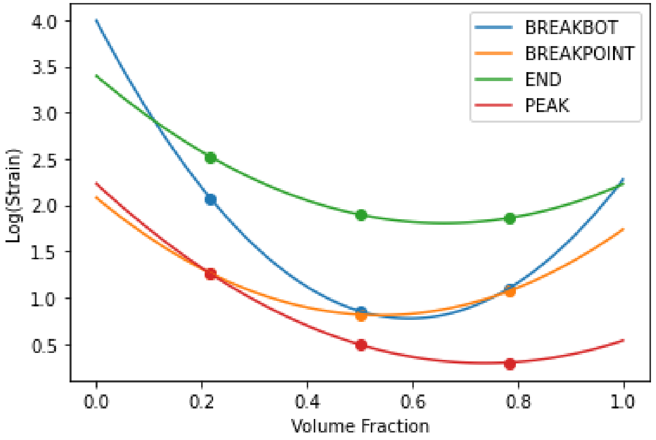
# Limitations

- Need for many quasi-static runs → repetitive, time-consuming process
- Runs affected by mesh quality, other factors
- Using fewer references could mean unrepresentative data
- Key point selection can be unclear
- Failure is relative
  - Is element deletion valid for failure mode?
    - Substructure hasn't ceased to exist

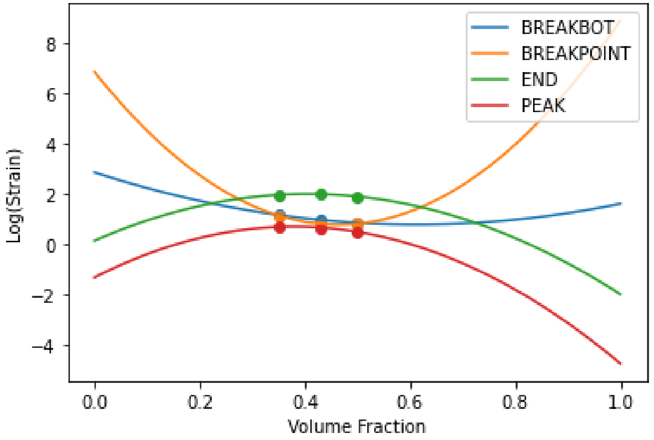
**Full Results**




**Representative ✓**




**Unrepresentative ✗**



# State of the Project

**Accurate surrogate model for stress regression and failure / non-failure classification for a simple truss substructure**

- Framework to:
  - Measure stress-strain response of a unit cell and lattice structure
  - Interpolate stress-strain results
  - Determine accuracy of interpolations
  - Classify failure / non-failure
  - Build surrogate model for cellular material stress-strain response

- Currently unable to accurately predict lattice results given unit cell results
- Haven't implemented model into topology optimization software
  - Proof-of-concept for surrogate modeling for cellular materials

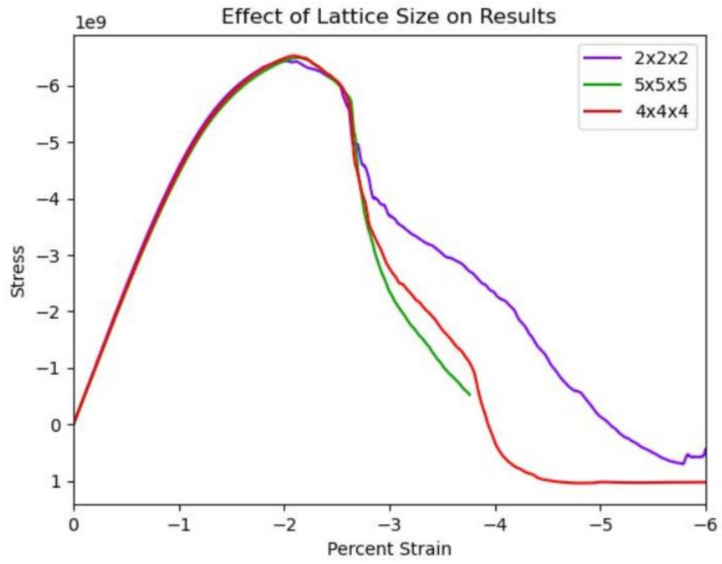
# Next Steps

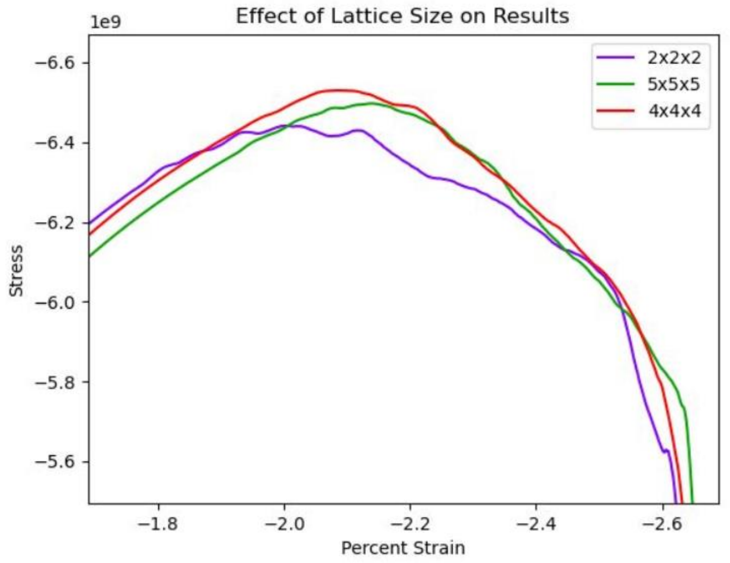
## Next Steps:

- **Obtain higher resolution compression lattice results**
  - Lower deformation speeds, larger lattices
  - $5cm^3$ @ 2.5cm / second → ~28 days for 5% deformation
- **Determine accuracy of transforming single → lattice**
  - Is there a consistent transformation regardless of substructure type?
- **Test efficacy of approach on other structures**
- **Implicit solver vs. explicit, quasi-static approach?**
- **Implementation of surrogate models:**
  - Interface with topology optimization software

## Long Term:

- Develop dataset to train a NN on failure point identification
- Create database of substructures and their surrogate models in topology optimization software
- Evaluate output using cellular materials vs. traditional topology optimization



# PART IV:

## OOP WRAPPERS FOR VELODYNE

# Issues Encountered

## Need many similar Velodyne runs with slightly different parameters

- Scripted .card creation and automatic run submissions from .k file
  - Parse load cell elements from LS-DYNA file
  - Ask user for deformation, speed, number of nodes on HPC, etc.

## Large amount of data from various sources

- How to pull in requested data for analyses?
- How to efficiently track job statuses?

~130 runs

# Thought Process

## Goal:

- Batch analysis of run results
- Efficient, streamlined tracking of HPC runs

## Process:

- Interpret a folder with Velodyne output files as a “Job” object
- Create Python wrapper to:
  - Provide access and mutate methods
  - Allow scripts to access simulation information in uniform fashion

## Caveats:

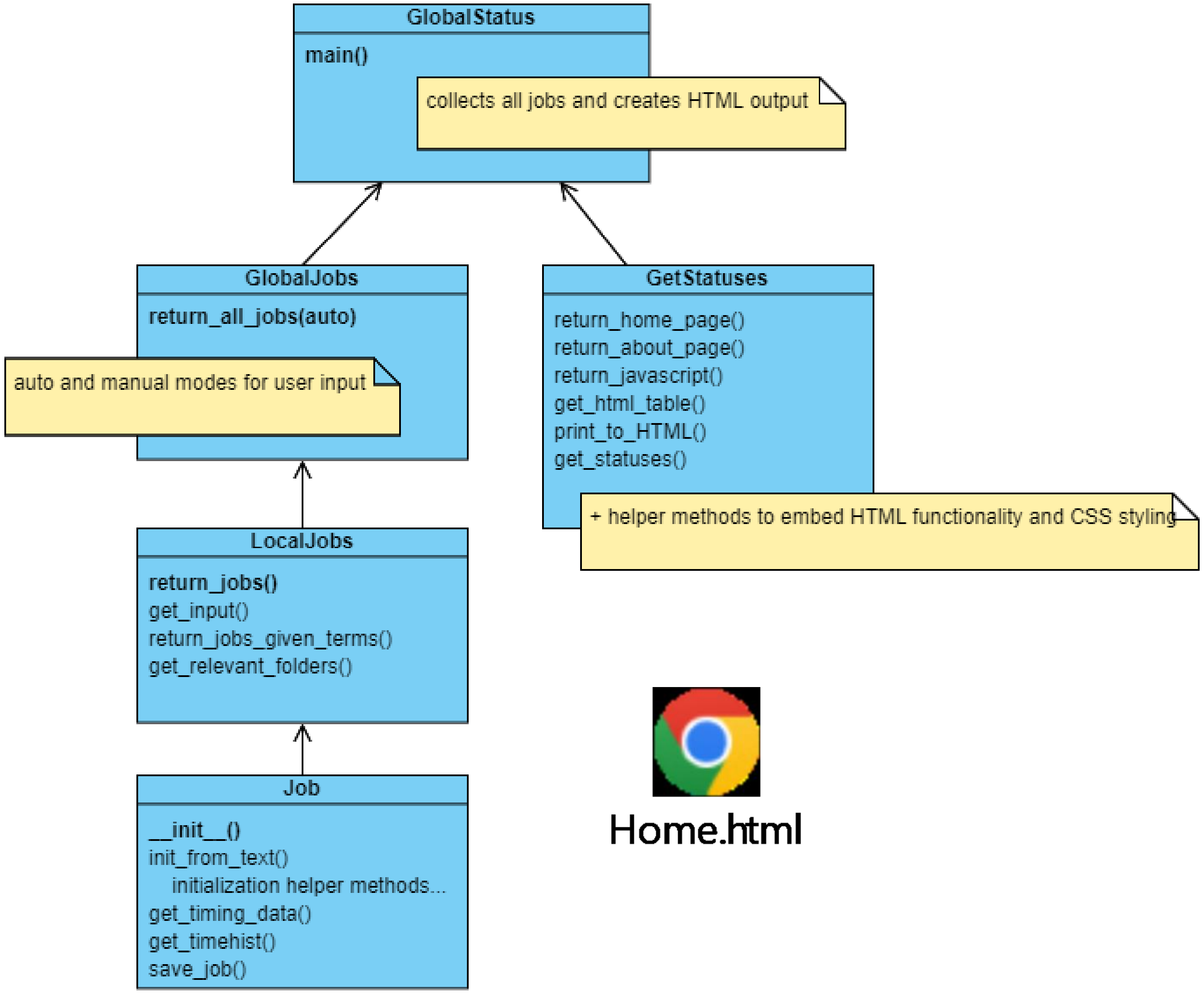
- Restarts and cancellations
- Generic but useful
- Adhere to efficient programming practices

# Job.py

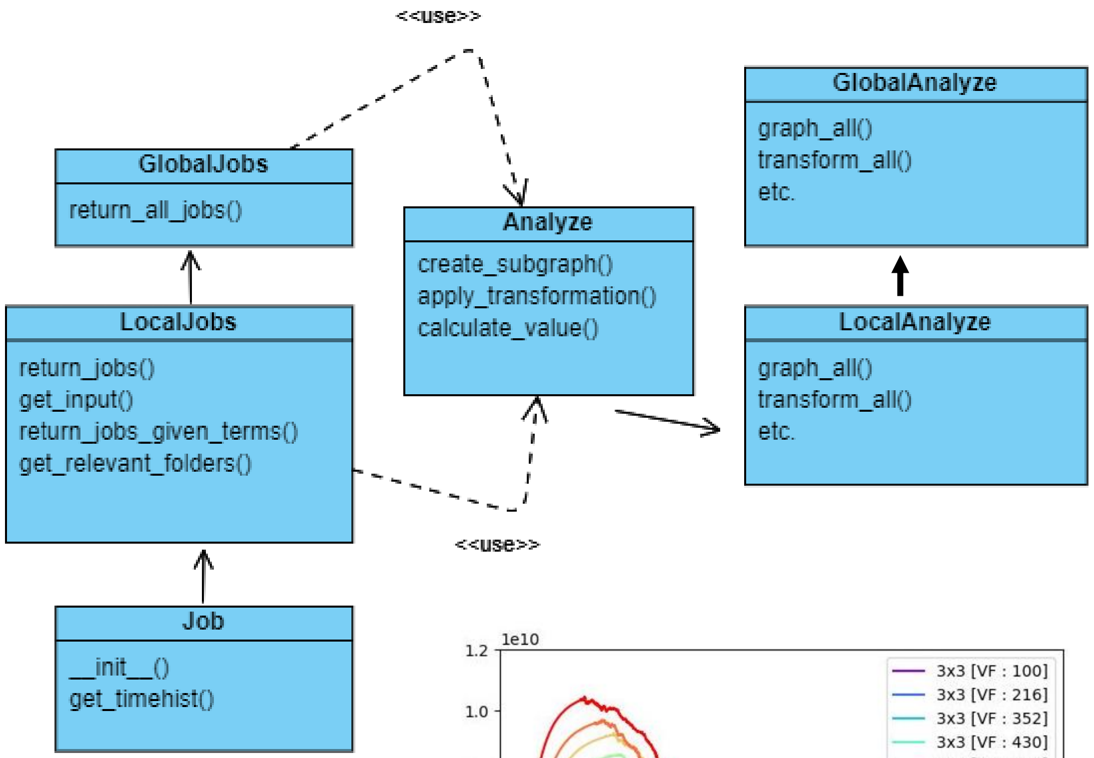
- **Stores important information in public instance variables**
- **Parses relevant information, such as:**
  - Start time (timestep.h5)
  - Estimated end time
  - HPC run code (.o file)
  - Run description (.card file)
- **Provides uniform access to data as pandas DataFrames:**
  - job.get_timehist()
  - job.get_simon()
  - job.get_deletion()
- **Handles restart and cancelation cases**
- **Minimal memory requirements**

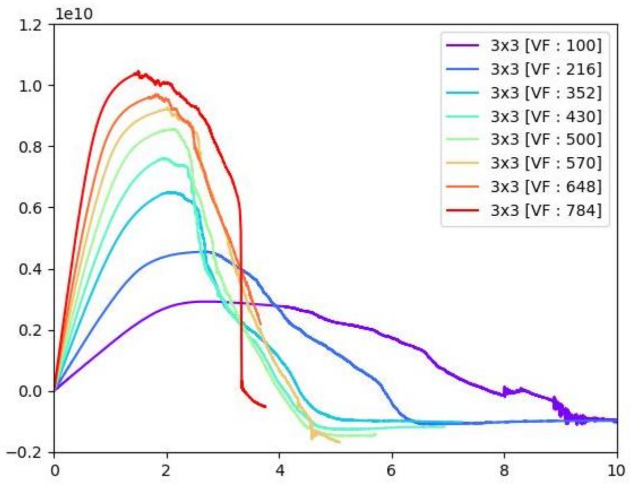
```python
def get_description(self):
    # Attempts to read in the description from the first 25 lines of the velodyne card file.
    # This method searches for a starting point as "Problem Title"
    # and ending point as "Title" or "Subtitle."
    # If the start and end are not found, it'll return an explanation of why it was not found.
    description = None
    try:
        # Searches for velodyne.card file within directory
        file_name = self.directory_name + "/velodyne.card"
        lines = self.get_first_n_lines(file_name, 25)
        extracted = False
        extracting = False
        for line in lines:
            if extracting:
                # Looks for ending line if already started
                if "title" in line.lower():
                    extracting = False
                    extracted = True
                    break
                description = description + line + "\n"
            # Looks for starting line
            if "problem title" in line.lower():
                extracting = True
                description = ""
        # Remove problematic characters to put in a single-line string.
```

# Example Structure: Job Tracking



# Example Structure: Analysis




```
[esnell@corvidpost3 Unit_Cells]$ python ../Python_Scripts/globalanalyze.py

Starting analysis hub for global job data.

Available options are:

    > SS: 'STRESS-STRAIN CURVE'
    > HG: 'HOURGLASS vs. TOTAL ENERGY'
    > DEL: 'DELETIONS'

Enter value[s] corresponding to desired analysis: SS

Enter search terms, separated by spaces:        3x3
Enter excluded terms, separated by spaces:      Velodyne250

    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_784.
    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_430.
    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_648.
    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_352.
    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_100.
    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_500.
    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_570.
    > 1 job loaded from /beegfs/interns/esnell/Unit_Cells/Reg3x3_216.

Running stress/strain analysis for 01_Reg3x3_100_Velodyne750...

    > Successfully extracted LoadCell data.

Running stress/strain analysis for 01_Reg3x3_216_Velodyne750...

    > Successfully extracted LoadCell data.

Running stress/strain analysis for 01_Reg3x3_352_Velodyne750...
```

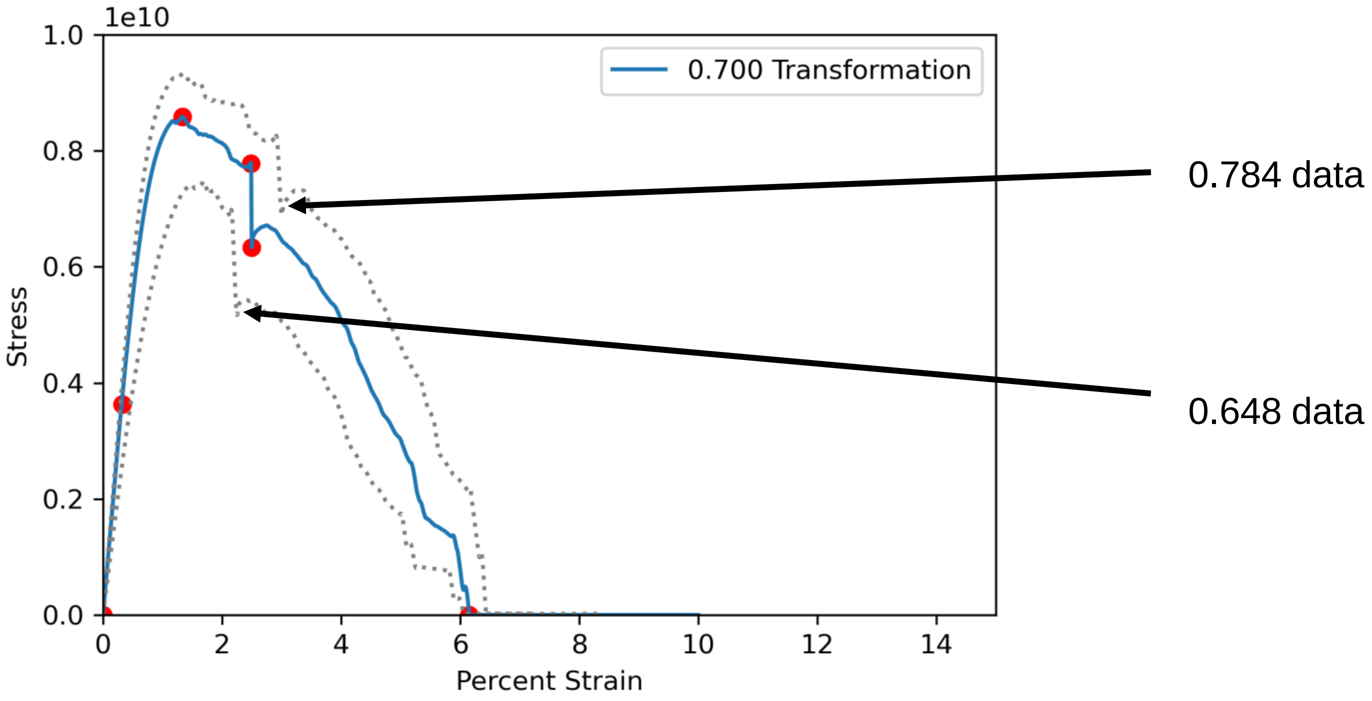
# SPECIAL THANKS TO:

## AARON WARD, ANGELO CRISTOBAL

# APPENDIX

# Interpolated Result -- Example

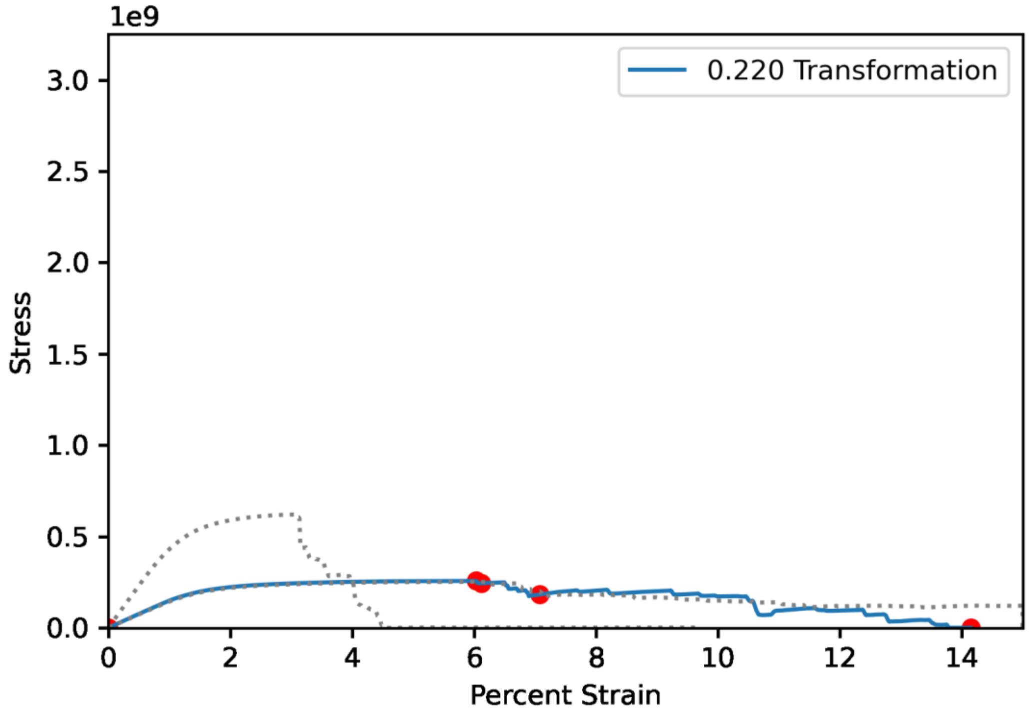
- **If we can estimate the location of key points on the new curve, we can apply a piecewise transformation of real data onto the estimated points:**



# Interpolated Results

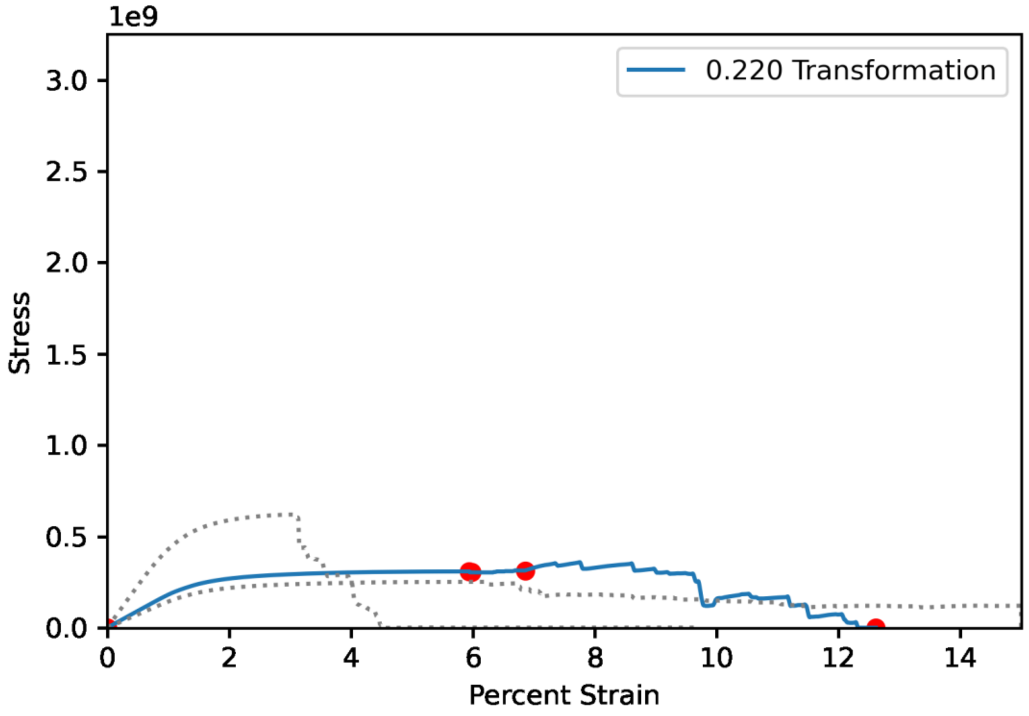
overfitted (5th degree polynomial)

- unpredictable movement of key points

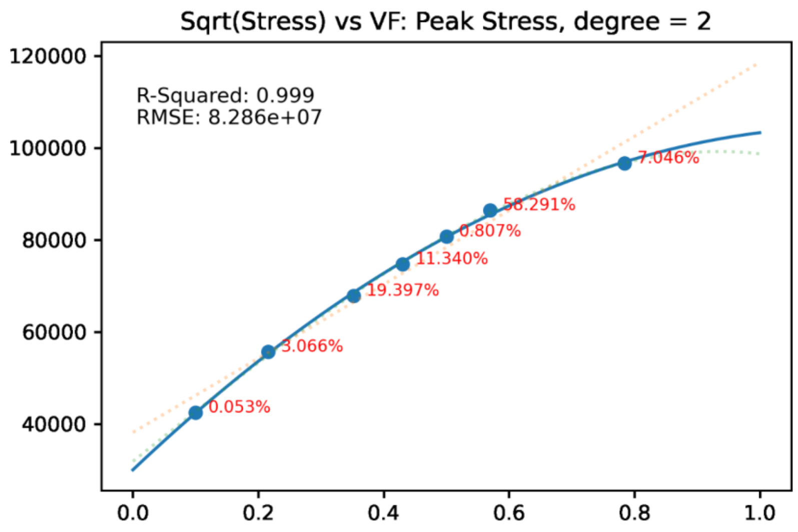



2nd degree polynomial

- transformation of points is smoother

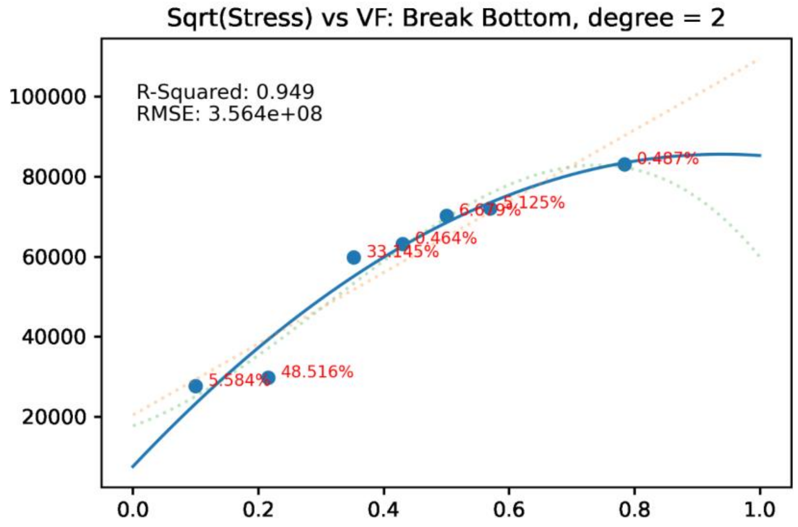


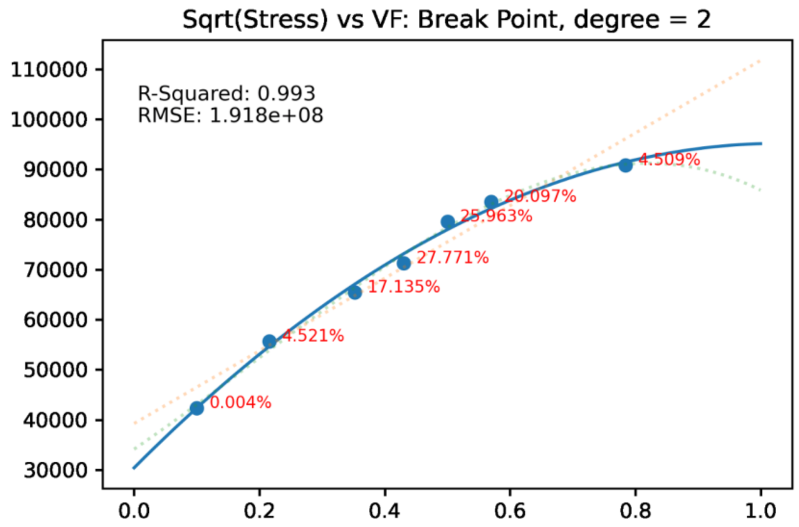
# Interpolating Stress



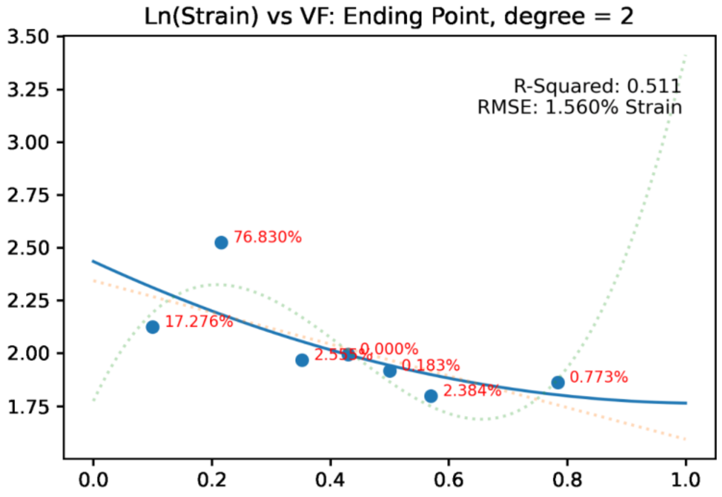

Red text indicates individual contribution to RMSE, and dashed lines are first- and third-degree polynomials.

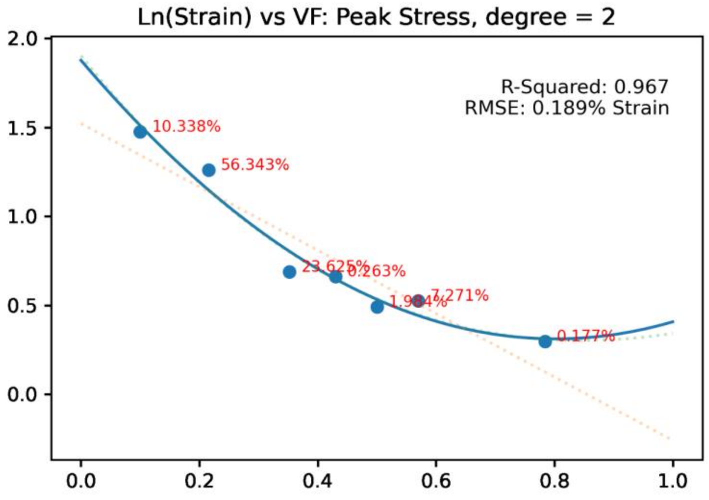
Quadratic approximation is quite strong for stress.

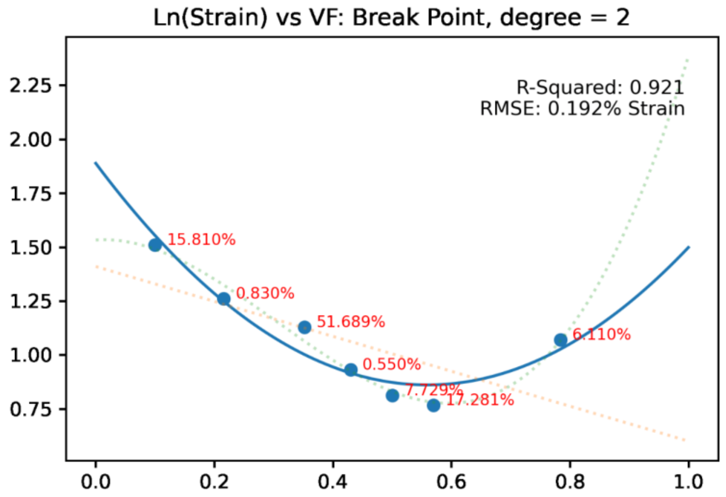


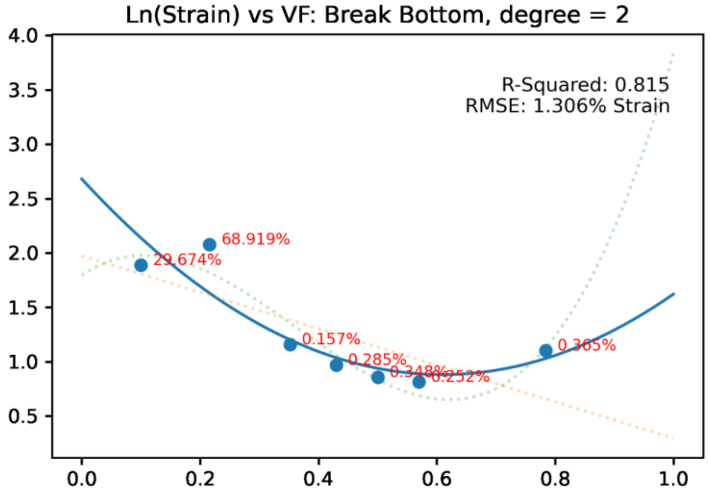



# Interpolating Strain



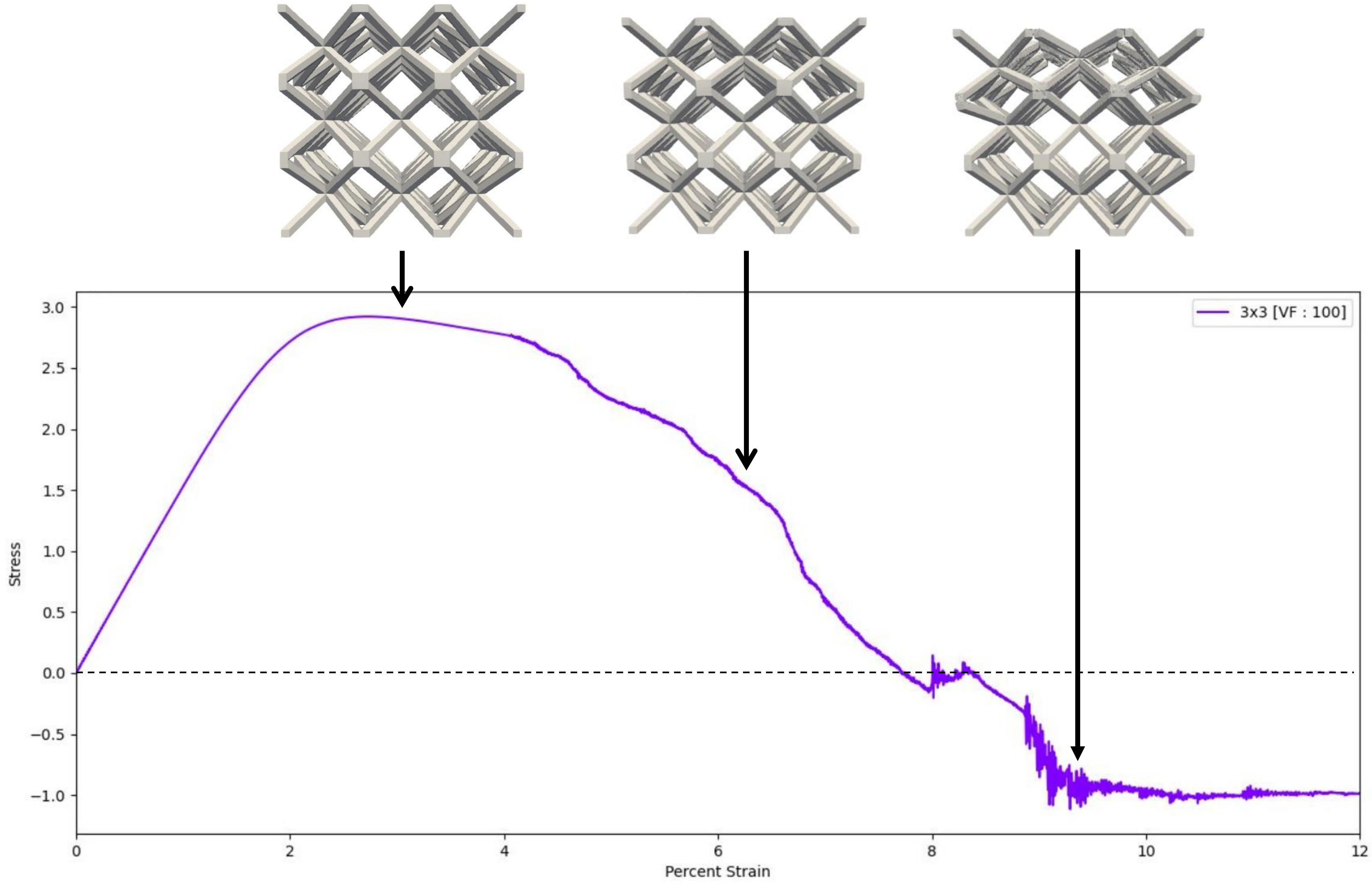




RMSE values for the most important metrics—break point and peak stress are quite low at 0.19% strain.

We can reasonably approximate the stress and strain at which an estimated volume fraction structure will break.

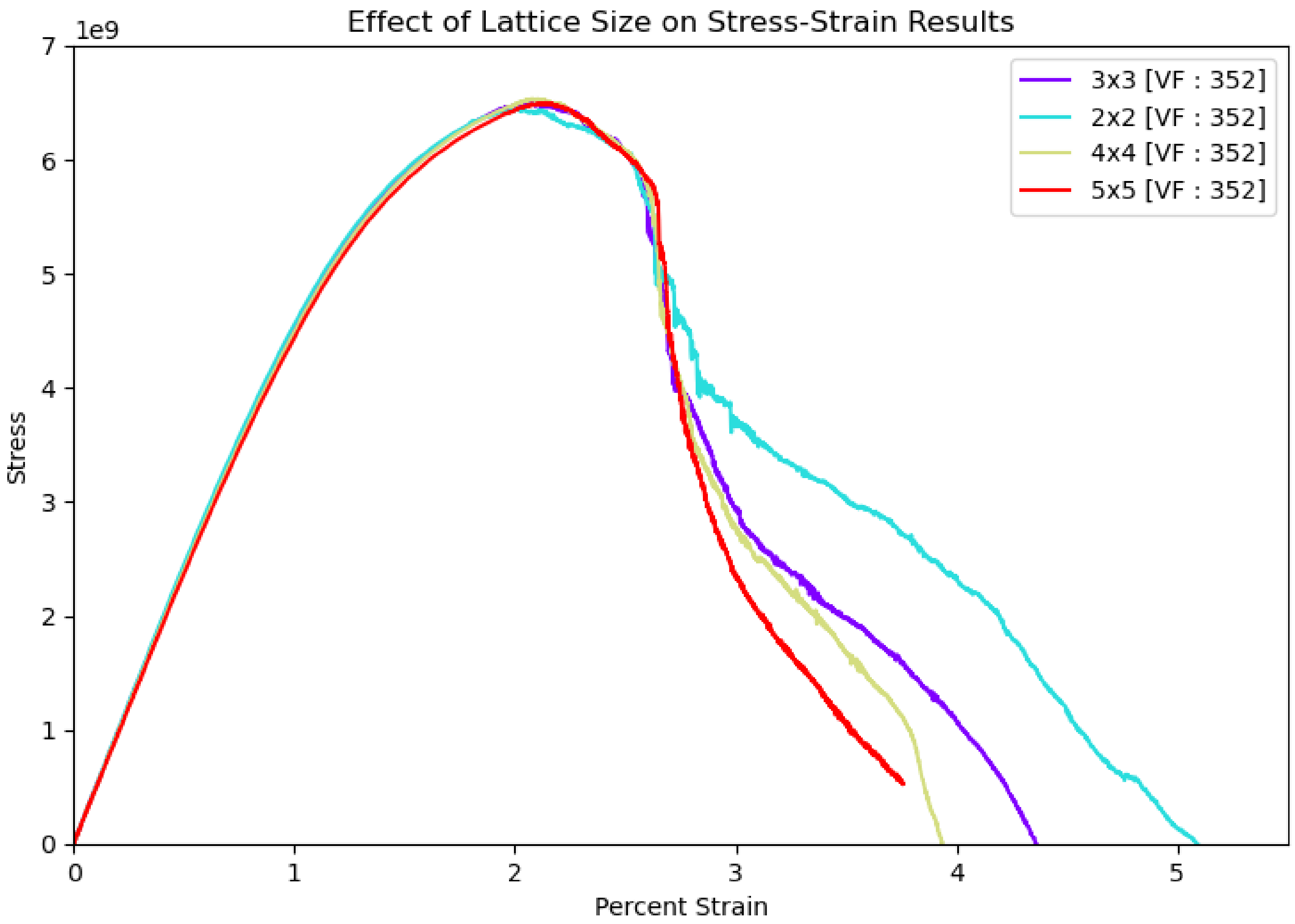
Behavior after element deletion begins in earnest is less predictable.

# Data Collection



# Effect of Lattice Size on Results

- **As we increase lattice size, computation time increases significantly**
  - A 5x5x5 lattice has 15.625 times more elements than a 2x2x2 lattice
    - Takes about a week to run 5x5x5 lattice with high quality mesh to 4% deformation @ 750% speed
    - ~6 months to run 7x7x7 lattice with high quality mesh to 5% deformation @ 100% speed
  - Bigger lattice → increased visual clarity
  - Smaller lattice → ability to run at slower speed



# Approach Comparison

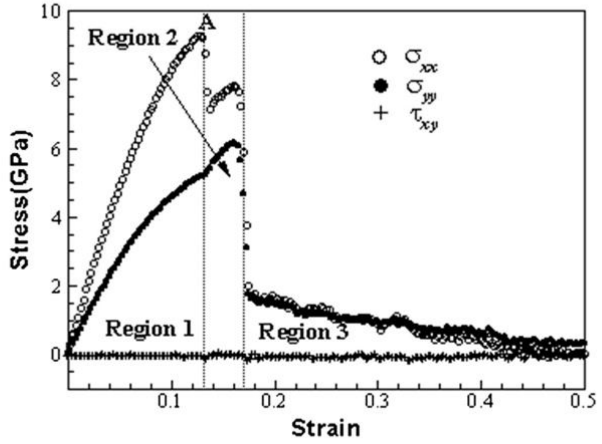



**Fig. 3** Stress–strain relation of an FCC Al crystal in uniaxial tension at 300 K

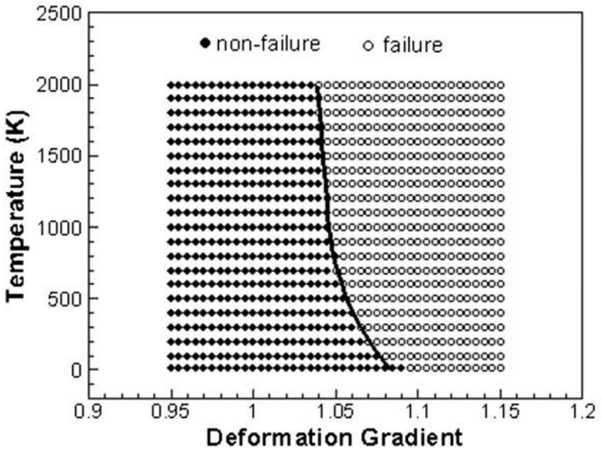
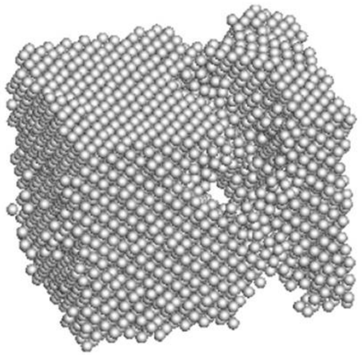
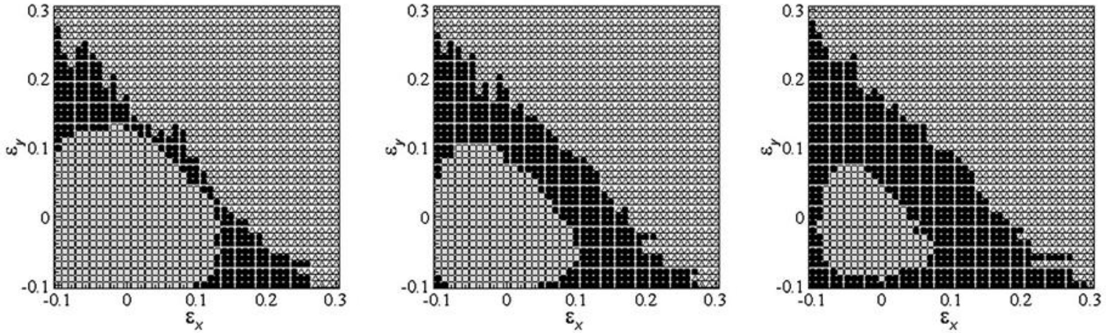



**Fig. 8** The material non-failure/failure interface predicted by machine learning

| | Xiao *et. al* | Goal |
|---|---|---|
| **Data** | Stress-strain response of Al crystalline solid | Stress-strain response of cellular substructure / lattice |
| **Collection Method** | Empirical measurements | Velodyne simulations |
| **Z Variable** | Temperature | Volume Fraction |
| **Model** | Single-Layer Feedforward Network (SLFN) | ? |
| **Use Case** | Prediction of RVE failure in lattice structure | Stress-strain prediction, failure / non-failure for TO |

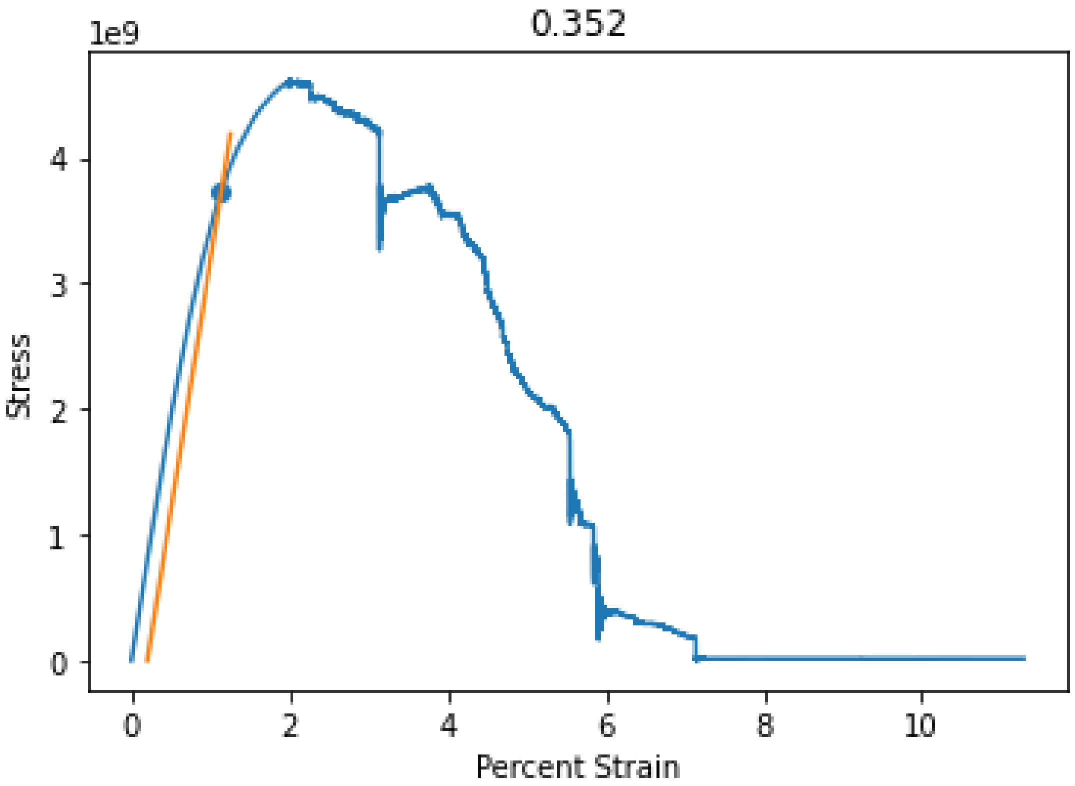


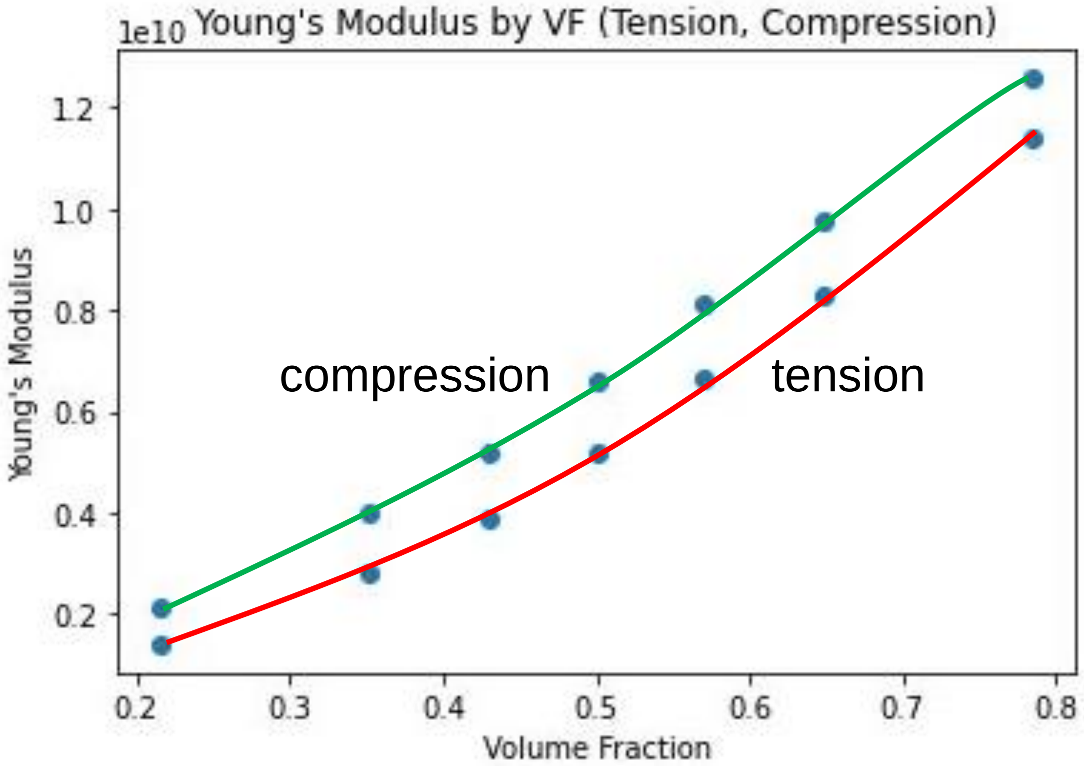
(a) $\varepsilon_x = 25\%$




(a) $\varepsilon_{xy} = 0\%$ (b) $\varepsilon_{xy} = 4\%$ (c) $\varepsilon_{xy} = 6\%$

# Young’s Moduli



# Preliminary Lattice Results

## Lattice results @ 7.5cm / sec

- Inconsistent peak location
- Variable relationship with unit cell results
- **Unclear break point**
- Difficult to fit an accurate model to predict failure / non-failure and interpolate results

Lower deformation rate → higher quality results, hopefully more clear failure behavior