Data-Driven Design of a Supercompressible Meta-Material

Chenjie Gan, Justin Miller

School of Engineering
BROWN UNIVERSITY

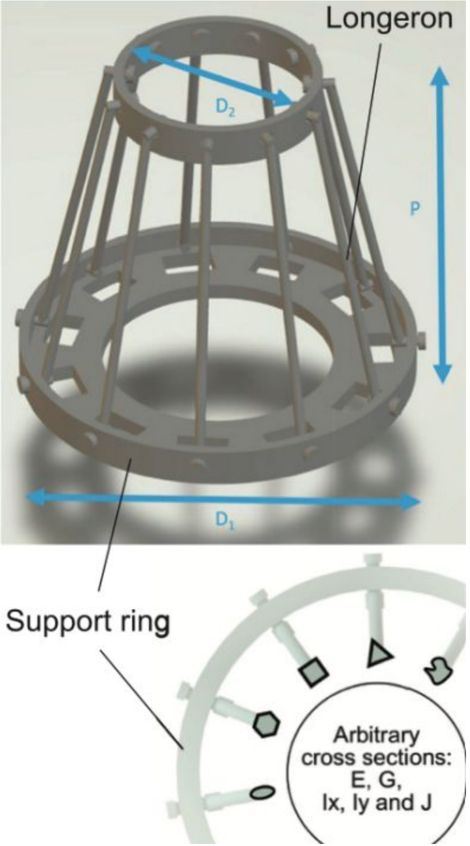
Introduction: Project Goals

- Examining datasets of FEA simulations on coilable structure

- Applying machine learning algorithms and optimization onto the feature space in order to draw conclusions about best designs

- Comparing algorithms on different data, feature counts, and hyperparameters to draw insights on the application of machine learning to mechanical design

Structure of Interest



[1] M.A. Bessa, P. Glowacki, and M. Houlder. Bayesian machine learning in metamaterial design: Fragile becomes supercompressible. Adv. Mater., 0(0):1904845–, October 2019.

Data Characteristics: Features

- Inputs were selected thoughtfully as a range of possible discretized designs for the structure and the output data was obtained from running ABAQUS Simulations

- 3D & 7D datasets, 1000 and 50000 points respectively

3D Input Parameters

expression	parameter name
$\frac{D_1-D_2}{D_1}$	ratio_top_diameter
$\frac{P}{D_1}$	ratio_pitch
$\frac{d}{D_1}$	ratio_d

7D Input Parameters

expression	parameter name
$\frac{D_1-D_2}{D_1}$	ratio_top_diameter
$\frac{P}{D_1}$	ratio_pitch
$\frac{I_x}{D_1^4}$	ratio_Ixx
$\frac{I_y}{D_1^4}$	ratio_Iyy
$\frac{J}{D_1^4}$	ratio_J
$\frac{A}{D_1^2}$	ratio_area
$\frac{G}{E}$	ratio_shear_modulus

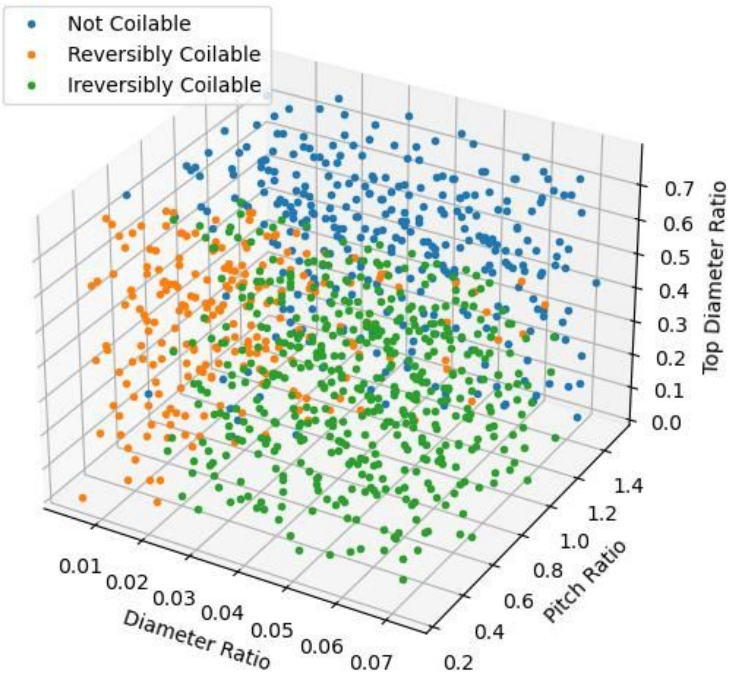
Output Parameters

expression	parameter name
coilability	coilable
σ_{crit}	sigma_crit
E_{abs}	energy

Data Characteristics: Preview

- Overview of feature space and categories
- Reveals organized DOE (Sobol Sequence)
- Guides choice of ML Models
  - Visualizable decision boundary
  - Apparent overlap
  - Intuitive feel of feature space

Feature Space Visualization

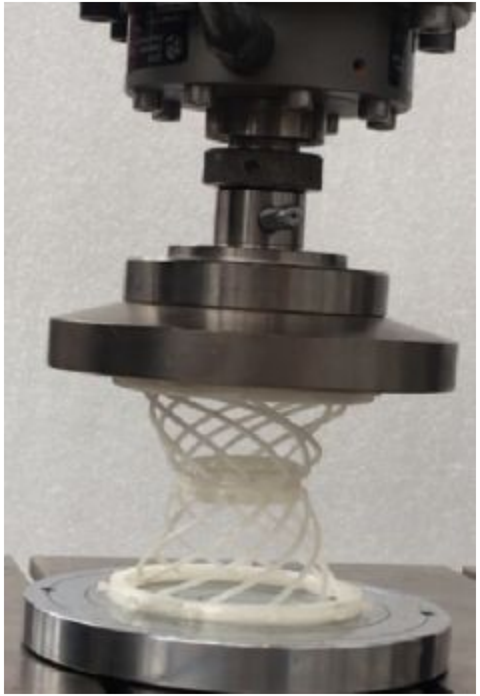


Model Selection: Overview

Goals in Model Selection for Meta-Material Design

- Quantitative:
    - Classification Correctness
    - Regression Accuracy
- Qualitative:
    - Interpretability
    - Physical Plausibility

Physical Representation of Model



Model Selection: Classification

- C Support Vector Classifier (RBF Kernel):
  - A strong, versatile Bayesian model that can find many types of separable barriers
- Random Forest:
  - A deterministic model that incorporates bagging and excels at decision making tasks
- Neural Network Classifier:
  - A complex, deterministic model with growing relevance in many fields
  - Expected to struggle on low dimensional, low data tasks but to excel on harder datasets

Results: Classification - 3D (3 Class)

- Despite being very different models, SVM and Random Forest are tied at the best accuracy metric

- However, if one choice had to be made the group would opt to use the Bayesian-focused SVM

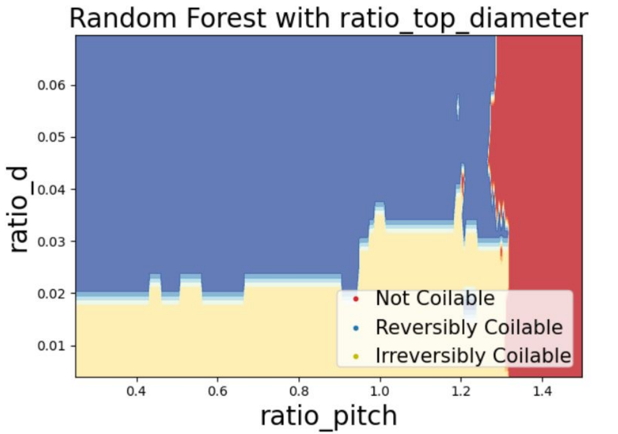
- Neural Network underperforms at this task

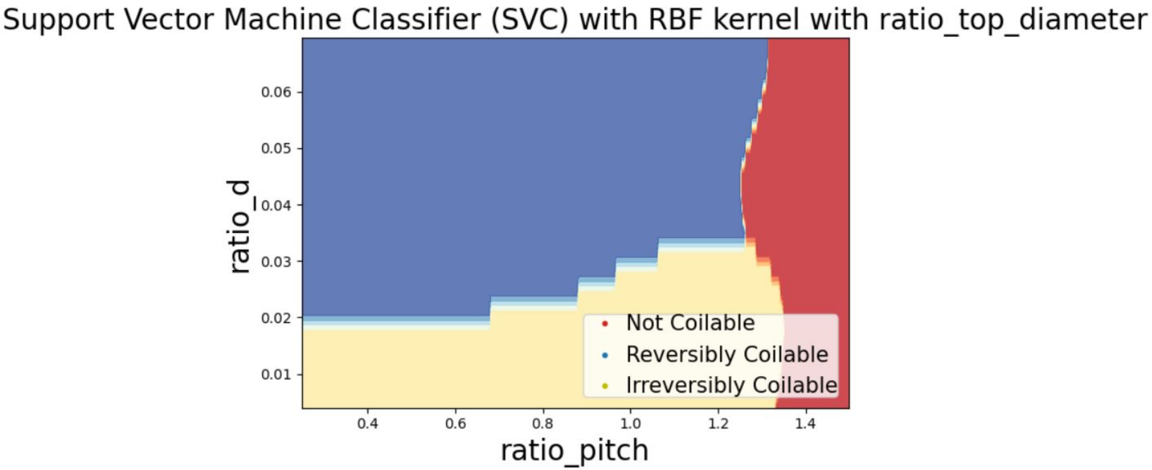
3D 3-Class Accuracy Table

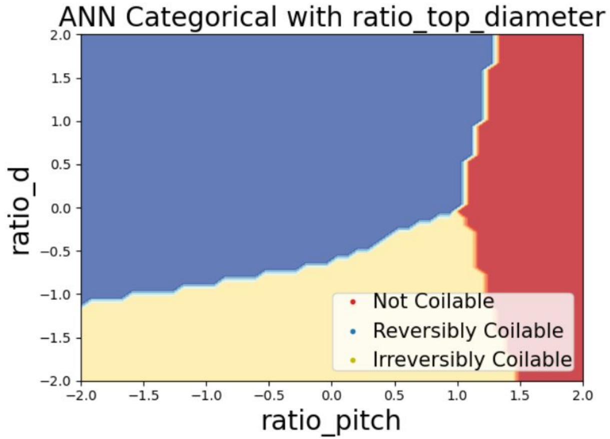
	Accuracy
SVM (RBF Kernel)	0.852
Random Forest	0.852
ANN (two hidden layers)	0.820

Model Comparison: Classification

- Decision Boundary plots reveal more of the story







Results: Classification - 3D (Binary)

3D 2-Class Accuracy Table

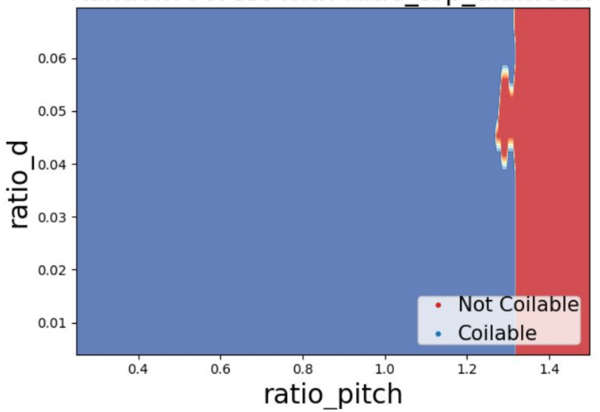
	Accuracy
SVM (RBF Kernel)	0.952
Random Forest	0.960
ANN (two hidden layers)	0.952

- In binary classification (between coilable and not coilable), there is a significant improvement to accuracy
- This improvement was driven by a significant overlap between reversibly and irreversibly coilable behaviors
- Since all models now have extremely high performance, the group would still pick the most explainable one, which is still SVM

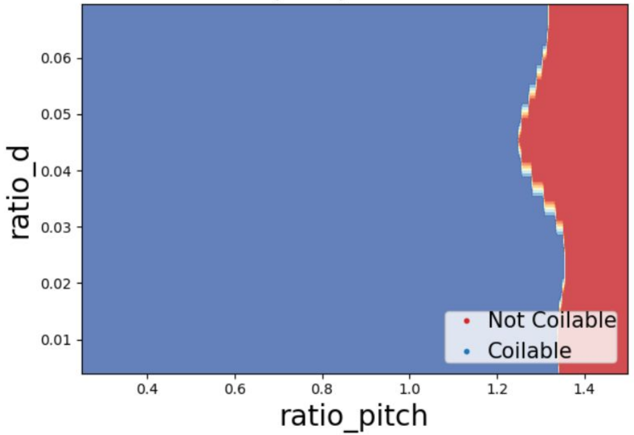
Model Comparison: Classification

- Model complexities (and limitations) can be seen more clearly in binary classification

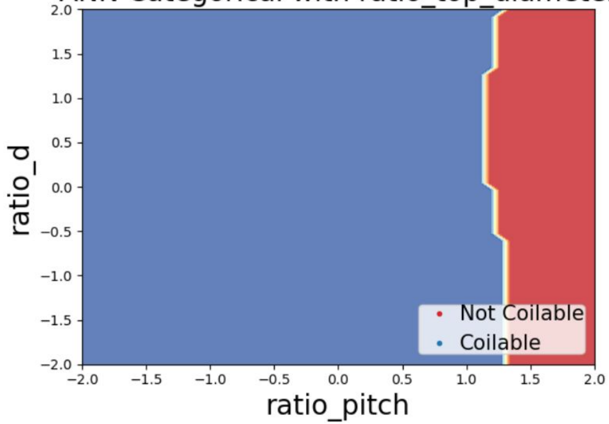
Random Forest with ratio_top_diameter



Support Vector Machine Classifier (SVC) with RBF kernel with ratio_top_diameter



ANN Categorical with ratio_top_diameter



Results: Classification - 7D (Binary)

- On 7D data, the accuracy for SVM and Random Forest is similar to the 3D binary task

- The Neural Network sees the most improvement, added complexity in the problem suits it well

- The group would choose the neural network here, but with a grain of salt

7D 2-Class Accuracy Table

	Accuracy
SVM (RBF Kernel)	0.958
Random Forest	0.952
ANN (two hidden layers)	0.970

Model Selection: Regression

- Gaussian Process Regression (Matern Kernel):
  - A very powerful Bayesian model known for its strong handling of uncertainties
- Ridge Regression:
  - A simple, mostly Bayesian model that incorporates point estimates / regularization
- Neural Network Regression:
  - A complex, deterministic model with growing relevance in many fields
  - Expected to struggle on low dimensional, low data tasks but to excel on harder datasets

Results: Regression - 3D

- For the 3D Dataset, the regression has very good results across all models
- Neural Network performance lags slightly behind the two simpler models

3D Regression Metric Table

	R Squared	MSE (Scaled)
GPR (Matern)	0.999328	0.000624
Ridge Regression (Degree 5)	0.999332	0.000621
ANN (two hidden layers)	0.998623	0.001314

- The similarities between GPR & Ridge are expected due to the model formation

Results: Regression - 7D

- 7D Regression also shows great results

7D Regression Metric Table

	R Squared	MSE (Scaled)
GPR (Matern)	0.964031	0.032061
Ridge Regression (Degree 7)	0.978428	0.017931
ANN (two hidden layers)	0.984236	0.014808

- On increased dimensions & data, Neural Network takes the #1 spot

Comparisons: Dimensionality

- 3D , 1000 Data Points:
    - Strong Bayesian Models thrive (SVM, GPR)
    - They enjoy both the best accuracy and also best explainability / physical roots

- 7D , 50000 Data Points:
    - Neural Network thrives on both Classification and Regression
    - Enjoy best accuracy while maintaining relatively short training times
    - However, explainability could be an issue for some applications

Comparisons: Classification vs. Regression

- An interesting artifact is the difference between classification and regression behavior as the dataset & dimensions increase

- For classification, the 7D dataset results in better or roughly the same classification accuracy as the 3D dataset

- For regression, the 7D dataset results in worse metrics than the 3D dataset

Comparisons: Scalability

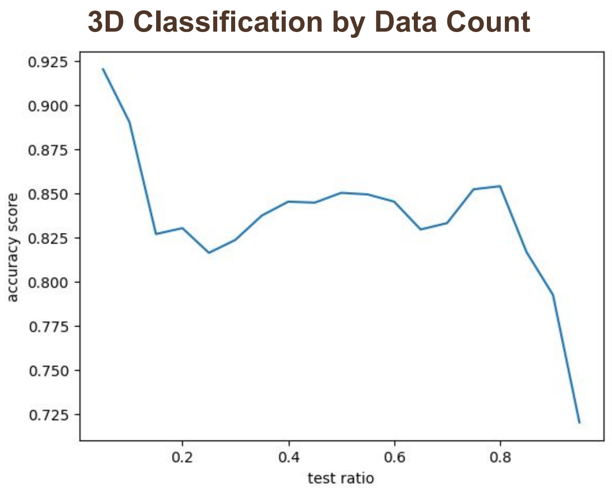
- Bayesian models and Deterministic models behave different ways when scaling from 3D to 7D datasets, and from 7D datasets to beyond

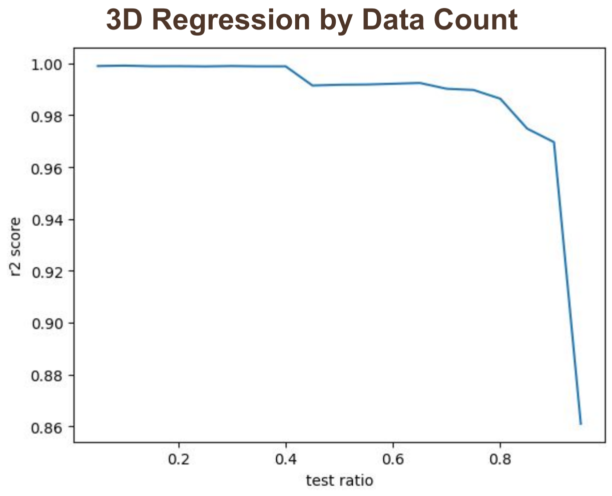
- Bayesian:
  - Runtime swells, especially for GPR
  - Performance sees diminishing returns very quickly

- Deterministic:
  - Runtime increases, but is manageable
  - Performance does see some diminishing returns
  - But for complex models (like NNs), added dimensions now add more opportunity

Evaluating Better Models: Data Split

- Generally aligns with assumption: More Data for is Better
- Accuracy is low for both SVC and GPR due to insufficient training data, not capture the complexity and variability of the data
- The accuracy of SVC and GPR increases as training ratio increasing



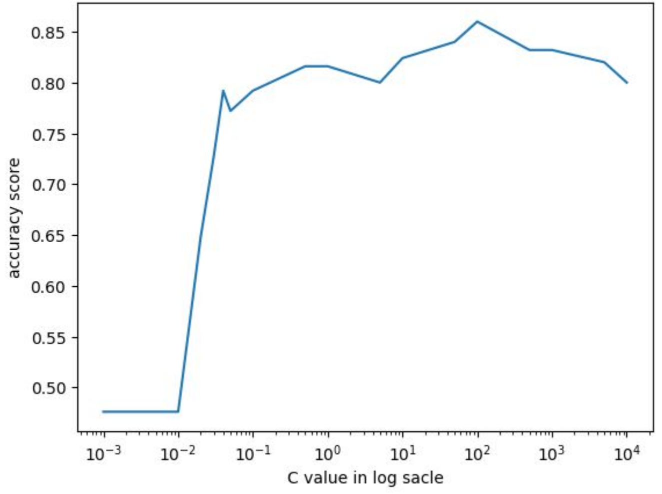


School of Engineering
BROWN UNIVERSITY

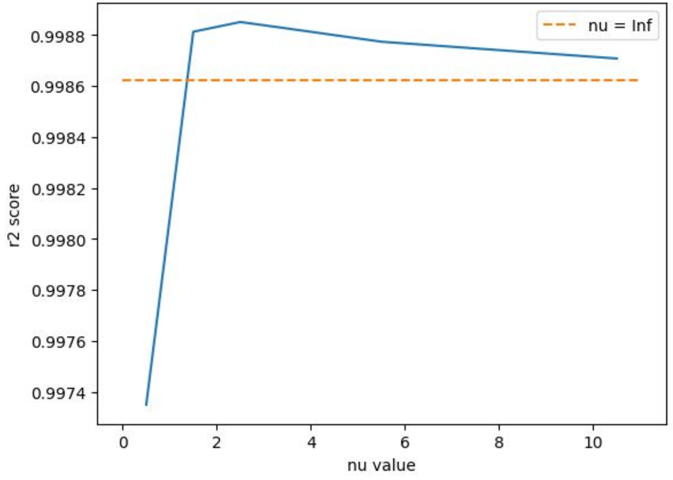
Evaluating Better Models: Hyperparameters

- C in SVC helps balance the model's need to correctly classify training data against the desire to keep the model simple enough to be effective on unseen data.
- A higher value of C reduces the regularization effect, allowing the model to focus more on fitting the training data as accurately as possible
- For GPR with Matern kernel, nu is controlling the smoothness of the function that the model will predict.
- Nu = 2.5 gives best results pointing out the target function’s smoothness should around this value

SVC with different C value



GPR with different nu value



Optimization: preprocess - initial points

Raw dataset → **Drop NAN data** → Preprocessed dataset → **SVC classifier** → Predicted dataset → **Sequence by stress or energy** → Best points in dataset

	ratio_d	ratio_pitch	ratio_top_diameter	coilable	sigma_crit	energy
0	0.004	0.25	0	1	0.0004942	0.00027953
1	0.0385	0.875	0.4	0		
2	0.05575	0.5625	0.2	2	23.052002	14.2853201
3	0.02125	1.1875	0.6	0	0.6937755	
4	0.029875	0.71875	0.5	1	2.88796194	
5	0.064375	1.34375	0.1	0	36.6998566	
6	0.047125	0.40625	0.7	2	24.3035964	
7	0.012625	1.03125	0.3	1	0.06897521	
8	0.0169375	0.640625	0.75	0	0.43959869	
9	0.0514375	1.265625	0.35	0		
10	0.0686875	0.328125	0.55	2	79.3113645	39.971166

Drop

Algorithms	crit/(KPa); (r_top_diameter, r_pitch, r_d)	Eabs/(KJ m-3); (r_top_diameter, r_pitch, r_d)
Part 1 dataset maximum	84.408; (0.072, 0.976, 0.477)	3.732; (0.034, 0.321, 0.758)
Search Point in Dataset	4.969; (0.032, 0.449, 0.665)	3.732; (0.034, 0.321, 0.758)

Optimization: preprocess - objective function

```python
def objective_function(x, target_output):
    # Ensure that the input is classified as '1' by the SVC
    x_scale_class = scaler_x_class.transform([x])
    x_scale = scaler_x_regression.transform([x])

    if svc.predict(x_scale_class) != 1:
        return float('inf')
        # return 100000

    # Predict with GPR and get the desired output
    gpr_result = gpr.predict(x_scale, return_std=False)
    result = scaler_y_regression.inverse_transform(gpr_result)
    sigma = result[0][0]
    energy = result[0][1]
    if target_output == 'sigma':
        sigma_converge_space_NM.append(sigma)
        return -sigma  # Negative because we are using a minimization function
    elif target_output == 'energy':
        energy_converge_space_NM.append(energy)
        return -energy
```

with SVC classifier, return infinity value when not coilable or coilable but yield;

with GPR model, calculate critical stress and absorbed energy with input point

return negative stress or energy as the value of objective function for further minimize process

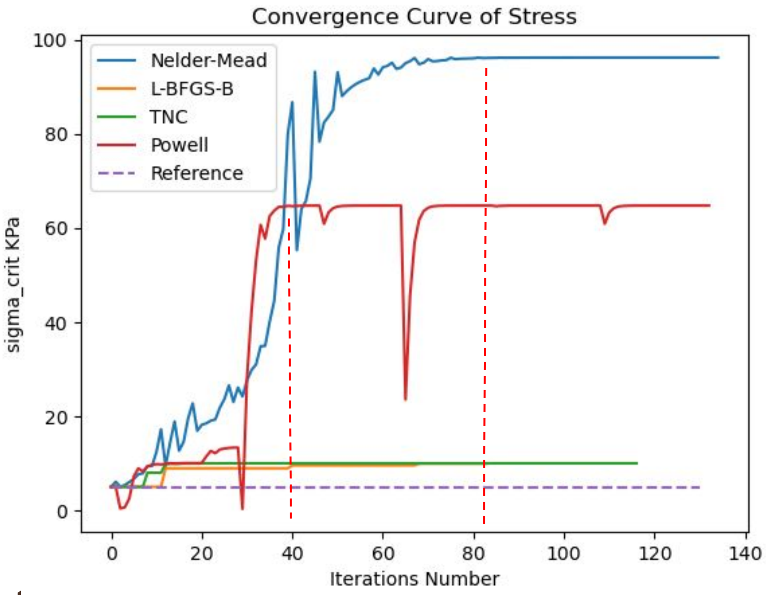
Optimization: basic algorithms – Nelder-Mead & L-BFGS-B

Algorithms	crit/(KPa); (r_top_diameter, r_pitch, r_d)	Eabs/(KJ m-3); (r_top_diameter, r_pitch, r_d)
Nelder-Mead method	118.431; (0.076, 0.977, 0.526)	47.589; (0.063, 0.686, 0.669)
L-BFGS-B method	9.902; (0.038, 0449, 0.665)	9.231; (0.042, 0.321, 0.758)

- Nelder-Mead: simplest algorithm without gradient descent or line search
- L-BFGS-B: efficient and well-used algorithm with gradient descent and line search
- Question: Why optimum value so low? Gradient? Line search?
- Candidates for further exploration:
  - Powell: derivative-free; line search
  - TNC: gradient based; without line search

Optimization: further exploration for gradient based and line search – Powell & TNC

Algorithms	crit/(KPa); (r_top_diameter, r_pitch, r_d)	Eabs/(KJ m-3); (r_top_diameter, r_pitch, r_d)
Nelder-Mead method	**118.431**; (0.076, 0.977, 0.526)	**47.589**; (0.063, 0.686, 0.669)
L-BFGS-B method	**9.902**; (0.038, 0449, 0.665)	**9.231**; (0.042, 0.321, 0.758)
Powell method	**65.394**; (0.057, 0.526, 0.773)	**38.898**; (0.057, 0.561, 0.760)
TNC method	**9.971**; (0.038, 0.449, 0.665)	**9.277**; (0.042, 0.321, 0.759)



- Gradient based search: dominates the poor performance
- Line search: negative role for optimum value; improvement for time efficiency

Conclusion

- Best critical bulk stress design can be 118 KPa, a best absorbed energy design can achieve 48 KJ per cubic meter.

- **there is often no single answer when it comes to machine learning!**
  Dataset, Models, Hyperparameters, Optimizers

Questions?

School *of* Engineering
BROWN UNIVERSITY