

Glenn Research Center

Usage of ChatGPT for Engineering Design and Analysis Tool Development

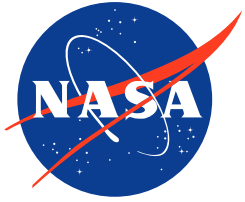
Kristopher Pierson, NASA Glenn Research Center
Matthew Ha, HX5, LLC, NASA Glenn Research Center

2024 AIAA SciTech Forum, Orlando, Florida
January 9th, 2024



This work was supported by:
NASA AATT and TTT Projects

AATT: Advanced Air Transport Technology
TTT: Transformational Tools and Technologies

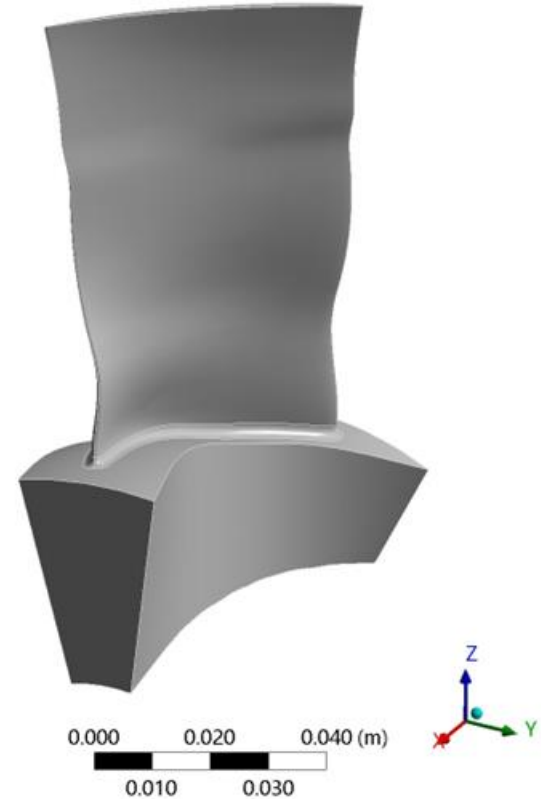


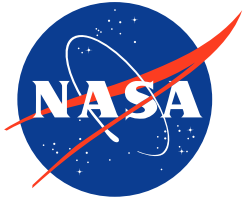
Presentation Outline

This presentation discusses advantages and pitfalls of using ChatGPT to conduct engineering tasks.

The capabilities of ChatGPT are presented in terms of the following examples:

- A simple flow visualization tool is created using ChatGPT. This highlights its programming capabilities.
- An attempt is made to use ChatGPT to script an Ansys DesignModeler geometry import process. It fails to do this and experiences AI hallucinations.
- ChatGPT is used to develop a design system for fan blades for a Turbofan engine.
 - The starting fan design is shown to the right.
 - The method utilized Neural Networks as a surrogate model for the response surface.
 - The design method is a proof-of-concept, which considers only centrifugal loading.
- **Note that the paper contains much more detailed information including prompting and responses from ChatGPT. Those details cannot be shown in the presentation due to time constraints.**



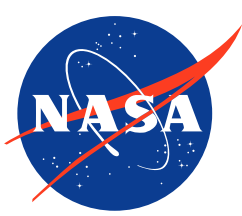


ChatGPT Accolades and Deficiencies

- ChatGPT is a chatbot developed by OpenAI that was released on November 30, 2022.
- GPT-4, which I find much more impressive, was released on March 14th, 2023.
 - ChatGPT based on GPT-4 is available to ChatGPT-Plus subscribers.
 - The GPT-4 version is the one presented in this work.
- GPT-4 has scored impressively on standardized tests.
 - 90th percentile, Uniform Bar Exam; 80th percentile, GRE Quantitative
- All versions of ChatGPT are interactive
 - It is conversational and intuitive, comparable to writing messages to a person
 - It can provide specific coding examples tailored to your application.
- GPT-4 suffers from hallucinations
 - This is a response generated by the model that is false or fabricated, which is confidently presented as fact.
 - All large language models (LLM) today suffer from hallucinations.
- ChatGPT has been successfully utilized to generate tools to analyze CFD results. It also aided in the development of a design methodology, which is discussed in the second half of this presentation.

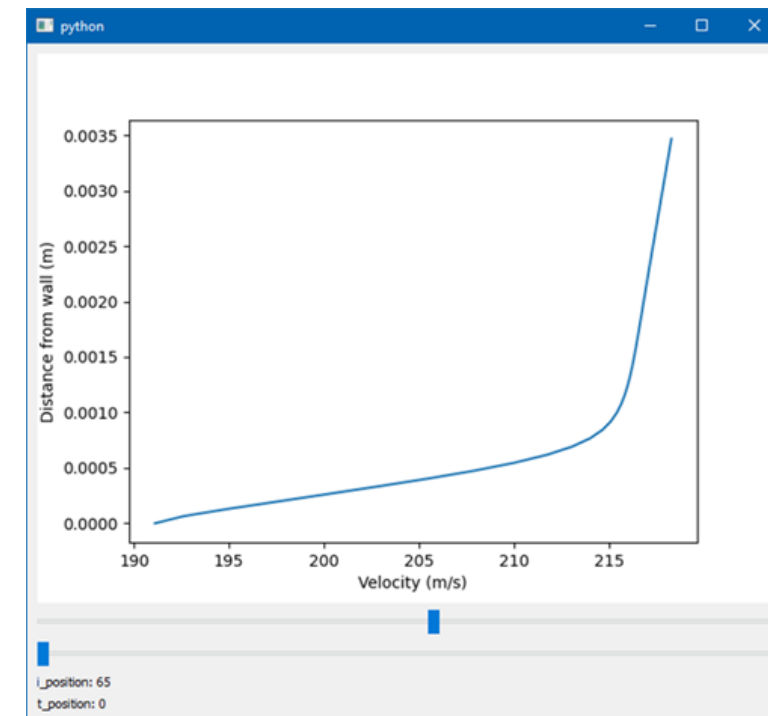
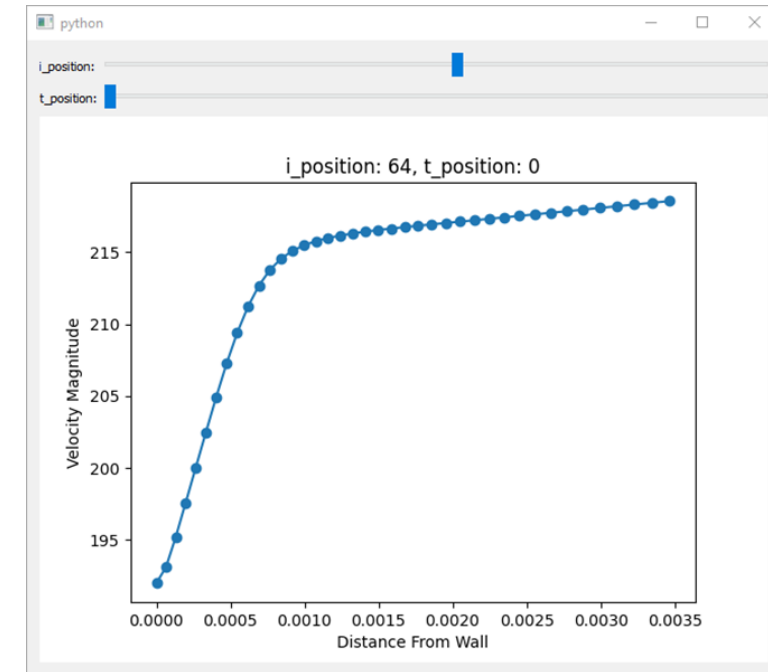


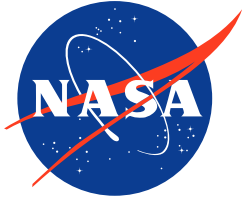
ChatGPT Logo



ChatGPT CFD Flow Visualizer

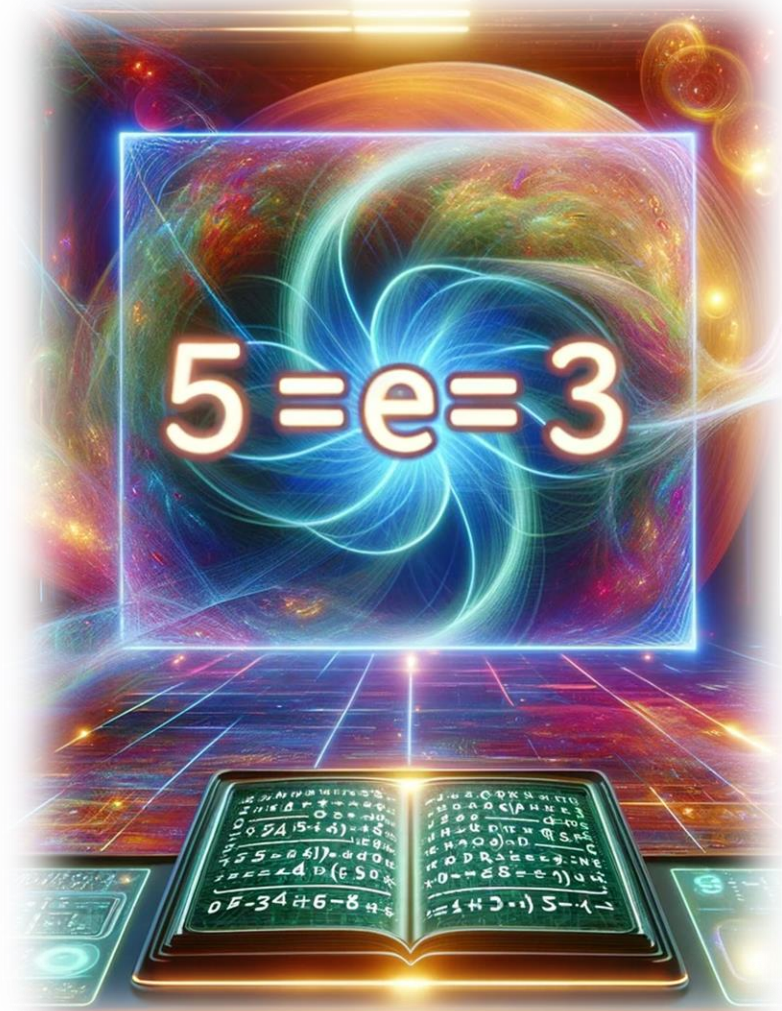
- ChatGPT was prompted with information about the data structure and how the visualizer should function.
 - `VelocityMagnitude[i_position, j_position, t_position]`
 - `DistanceFromWall[i_position, j_position]`
 - ChatGPT was prompted to produce a GUI that will allow the user to toggle `i` and `t` positions, while plotting `j`
- ChatGPT returned a tool that functioned, but it had several issues.
 - It was programmed as an object, but it could not be imported to other scripts
 - 'Velocity Magnitude' and 'Distance From Wall' should be on the other axis.
- A more detailed request was made, which produced a more tool that functioned better
 - The ability to import the tool to other scripts improved its utility significantly.
 - To do this, ChatGPT allowed for the object to take the data as an input during object instantiation.
 - The ChatGPT programming was impressive, complete with comments and meaningful class/method names.





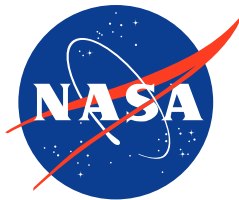
Beware of Hallucinations

- An important part of the design methodology presented next is changing the blade geometry in Ansys.
- ChatGPT was prompted to write a code that would switch out the blade geometry.
 - The tools used were Ansys Workbench / DesignModeler
- Instead, ChatGPT produced code written in made-up programming languages, which were presented in a fashion that make it seem correct to a non-expert.
 - Hallucinated two new programming languages for controlling geometry in DesignModeler
 - Ansys Geometry Interface Language (AGL)
 - ANSYS DesignModeler Scripting Language
- ChatGPT understood the request in terms of the steps that are need to be taken to complete the process.
 - However, it did not know the necessary syntax.
 - This result is likely due to a limited amount of openly available training data available for this type of process.
 - Training on Ansys internal documentation would likely remedy this issue and make ChatGPT an excellent assistant in performing simulations.

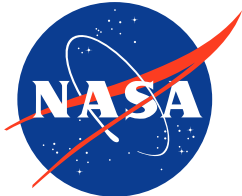


Design Method

Created with the help of ChatGPT



Glenn Research Center



Baseline Model and Analysis Type

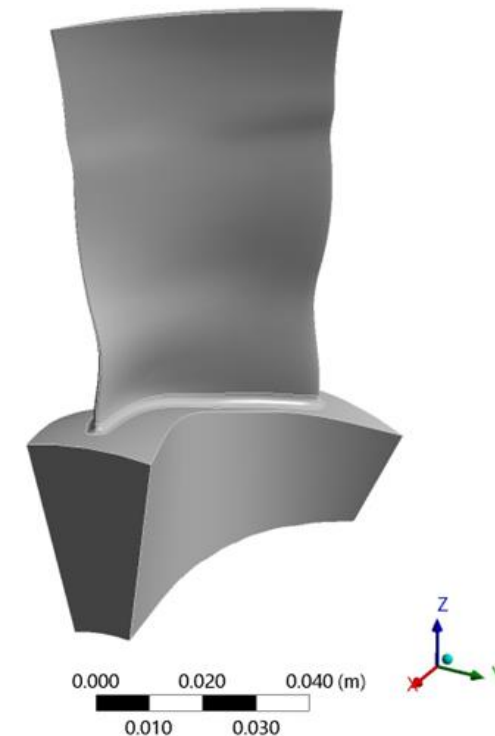
- Geometry for the model is generated using T-Blade3, a general parametric 3D blade geometry builder.
 - The disk is modeled as a thick wedge.
- The baseline blade model is the Tail Cone Thruster Design 7-28-8
 - This design is available on the T-Blade3 website cited in paper.
 - Blade is 150 [mm]; rotational speed is 21,581 [RPM]
- A slight alteration is made to the model on the T-Blade3 website; all lean and sweep control points are set to zero.
 - This reduced the FS from ~1.4 to ~1.0
- The blade is made of titanium.
- The initial factor of safety is 1.01

Not Considered

Flutter
Forced Response
Aero Efficiency
Bird Strike
and more

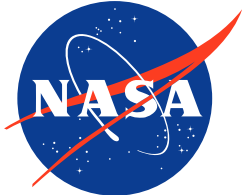
Considered

Centrifugal Load

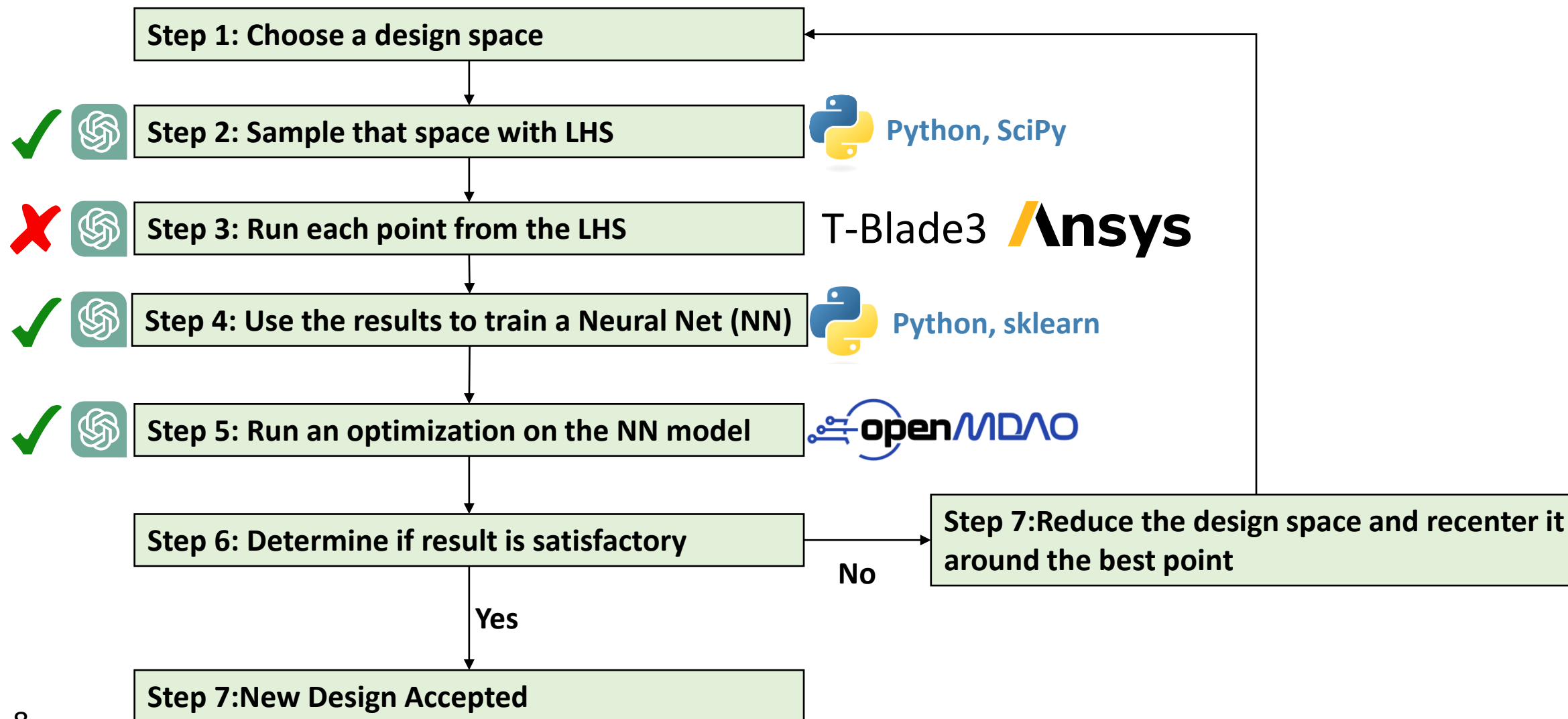


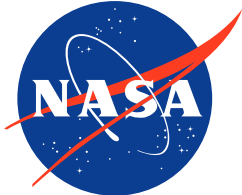
Ti-6Al-4V Properties

Density [kg/m ³]	7,850
Elastic Modulus [GPa]	200
Poisson Ratio	0.3
Tensile Strength [MPa]	250



Design Method Flowchart





ChatGPT and Design Optimization

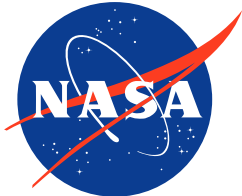
- ChatGPT explained existing OpenMDAO code
 - OpenMDAO is an open-source multidisciplinary analysis and optimization Python package written at NASA GRC
 - In this work, OpenMDAO is used to run cases and to perform optimization on the Neural Network model of the design space.
 - ChatGPT provided an excellent explanation of the existing code. It was even able to reference OpenMDAO source code.
- ChatGPT wrote the code to sample the design space
 - ChatGPT provided Python code that produced Latin Hypercube Sampling (LHS) of the design space using the SciPy package.
 - Code snippet show to the right; another, more impressive example of ChatGPT coding is in the paper's appendix.
- Recommended the usage of Neural Networks (NN)
 - ChatGPT recommended that NNs be used to represent the response surface.
 - It also wrote the python code for training the NN.

```
import numpy as np
import pandas as pd
from scipy.stats.qmc import LatinHypercube

# Define bounds
# Define your center points here
c1, c2, c3, c4, c5, c6 = [0, 0, 0, 0, 0, 0]
# Define your range values here
r1, r2, r3, r4, r5, r6 = [1, 1, 1, 1, 1, 1]
bounds = [
    [c1-r1, c1+r1],
    [c2-r2, c2+r2],
    [c3-r3, c3+r3],
    [c4-r4, c4+r4],
    [c5-r5, c5+r5],
    [c6-r6, c6+r6]
]

# Generate LHS sample
lhs_obj = LatinHypercube(d=6)
samples = lhs_obj.random(n=1000)

# Adjust the samples to the specified bounds
samples_adjusted = np.array(bounds)[: , 0] + samples *
(np.array(bounds)[: , 1] - np.array(bounds)[: , 0])
```



Design Variables and Objective Function

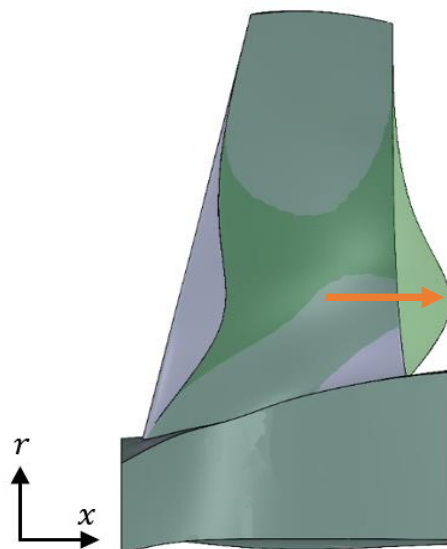
- Design Variables are lean and sweep control points.
 - 25%, 50% and 100% radial span
- The lean and sweep control points shift the blade profile in the $m'_s - \theta$ coordinate system.
 - Specifics about this coordinate system can be found in the paper.
 - The example below shows the effect of changing the control point values

- The factor of safety (FS) is calculated by:

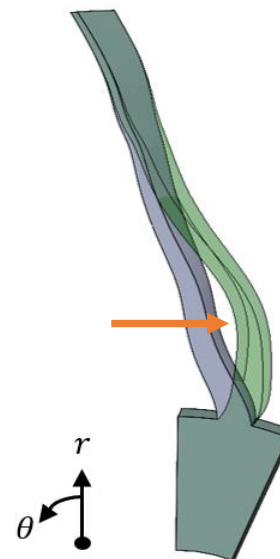
$$FS = \frac{\sigma_{ys}}{\sigma_{vm,max}}$$

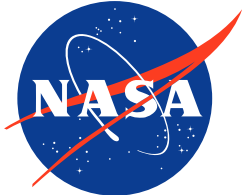
- σ_{ys} : Yield Strength of Titanium
- $\sigma_{vm,max}$: Maximum von-Mises stress in model
- Objective is to maximize FS through optimization of lean and sweep control points (3 each).

Increasing Sweep Control Point @ 25% Span

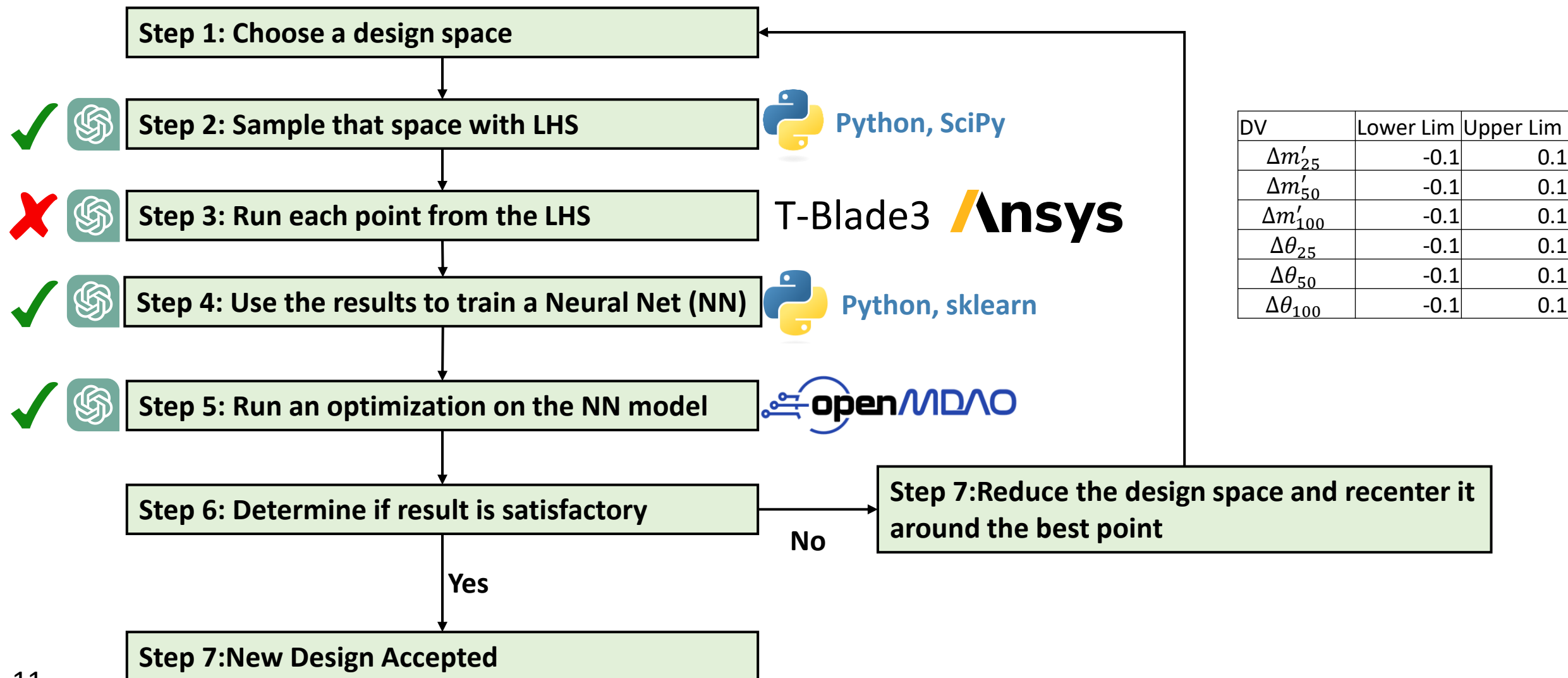


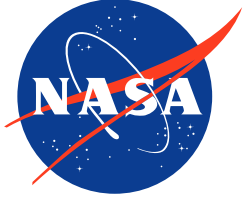
Increasing Lean Control Point @ 25% Span





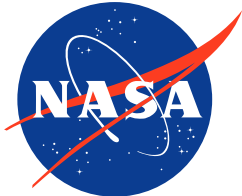
Design Method Flowchart





Neural Network Recommendation

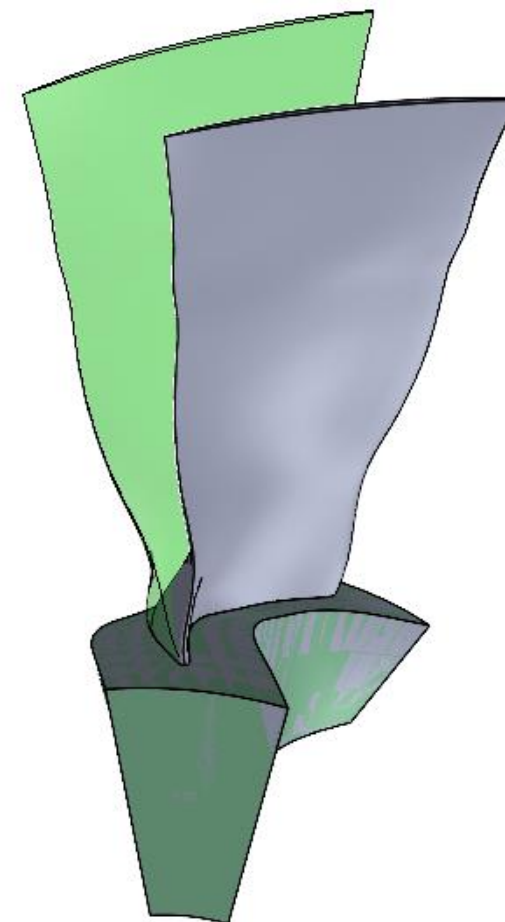
- While working on a work-flow, the type of surrogate model to use was a point of difficulty.
- Initially, polynomial regression was used to try to model the response surface, but this did not work well.
- ChatGPT was prompted for recommendations on substitutions for polynomial regression
 - It responded with two useful suggestions: Random Forest Regression and Neural Networks
- ChatGPT was able to program the training of these models, which allowed them to be explored with little effort.
- This allowed testing within a couple of hours of the recommendation.
 - The first test made it clear that the NNs had great potential for this application.
 - Random Forest Regression worked well at providing good estimates of FS values, but the response is discontinuous, which makes it less desirable than NN.



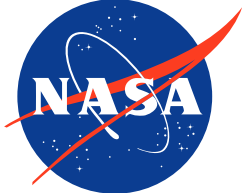
Optimization Results

- The final design showed a dramatic improvement over the starting design
 - Through modification of only the lean and sweep control points, the FS is increased 5X
- The design space has a very high level of variable interaction, the neural network captured this well.
- An issue with Ansys DesignModeler interpreting the STEP file cause approximately 20% failure of cases.
 - The method showed a great deal of robustness despite this issue
 - I will be moving to different tools going forward.

Iteration	Initial	1	2	3
$\Delta m'_{25}$	0	0.0006	-0.0566	-0.0562
$\Delta m'_{50}$	0	-0.0654	-0.1022	-0.1017
$\Delta m'_{100}$	0	-0.09	-0.1079	-0.1083
$\Delta \theta_{25}$	0	-0.0278	-0.0953	-0.0949
$\Delta \theta_{50}$	0	-0.0855	-0.153	-0.1526
$\Delta \theta_{100}$	0	-0.09	-0.1549	-0.1554
FoS, NN	N/A	2.141	3.264	3.504
FoS, FEA	1.017	2.742	5.163	5.176



Initial Design: Grey; FS=1.017
Final Design: Green; FS=5.176



Lessons Learned and Next Steps

- ChatGPT proved to be an excellent aid, but I only currently recommend people use it for topics which they already have some familiarity with.
 - ChatGPT makes mistakes, the user should always confirm factual accuracy, especially when querying about topics outside of their knowledge zone.
- ChatGPT is great at offering recommendations and writing code.
 - The recommendation and programming of the NN were instrumental in the development of the presented design method.
- The design methodology created with the aid of ChatGPT can be applied to all sorts of design optimization.
 - Changes are being made to run cases in parallel to achieve faster turn-around time.
 - **Multidisciplinary design optimization is our next step, where an Aerodynamic NN is paired with a Structural NN.**
- Various aspects of the NN training and architecture should be explored
- Sampling methods should also be explored
 - Orthogonal-Array LHS
 - This may help capture variable interactions better.

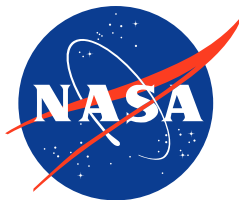


Thank You!
Questions?

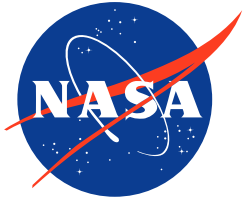


Glenn Research Center

Additional Information



Glenn Research Center



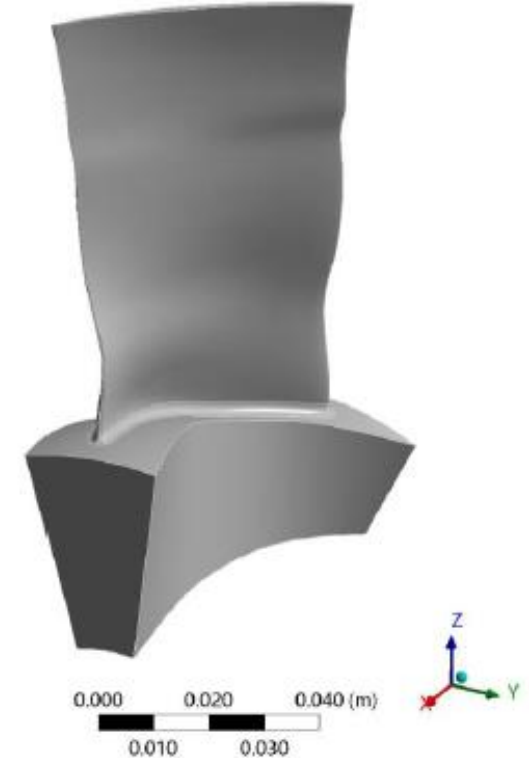
Baseline Model

- The baseline blade model is the Tail Cone Thruster Design 7-28-8
 - This design is available on the T-Blade3 website cited in paper.
- Blade Radius is approximately 150 [mm]
 - This is a scale model representative of one that could be tested in a wind tunnel.

- Design I
- A slight sweep c
 - Thi
- The blade
- A mesh
- The initi

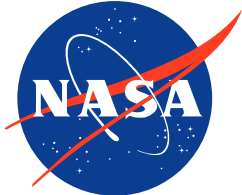


ean and



Ti-6Al-4V Properties

Density [kg/m ³]	7,850
Elastic Modulus [GPa]	200
Poisson Ratio	0.3
Tensile Strength [MPa]	250

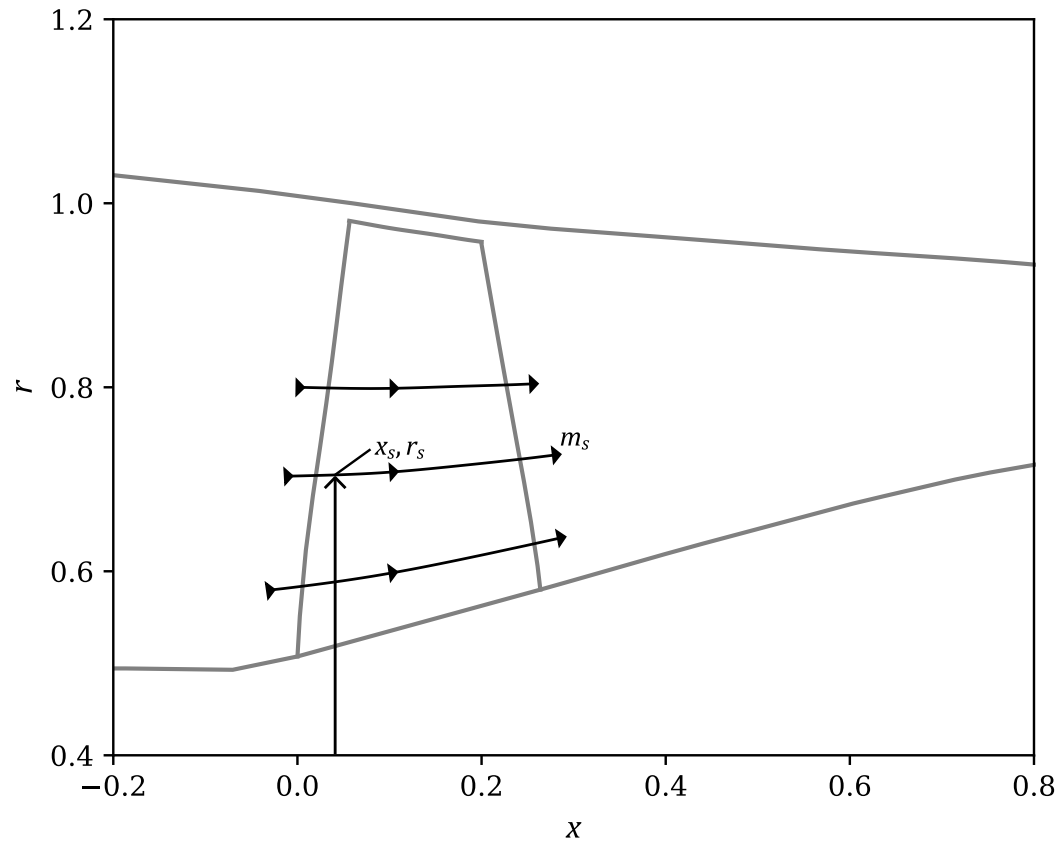
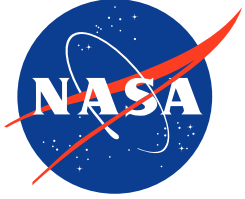


LHS and NN

- ***Might move this to the end***
- Latin Hypercube Sampling is a method of sampling an N-dimensional space such that each dimension is sampled uniformly.
 - This does not guarantee that combinations of variables are sampled
- To improve the sampling, a method of reducing centered discrepancy is utilized.
 - Centered discrepancy is the largest difference between the number of points in any subdomain and the expected number of points for an even sample.
 - Pushes the sample to a more uniform distribution
- After the sample points are simulated, the results are used to train a multilayer perceptron (MLP) model using scikit learn's MLPRegressor class
 - For the third iterations, it was necessary to add early stopping criteria.
 - Early stopping prevents over-training by separating the data into a training set (85%) and a validation set (15%)
 - This is a good practice that will be utilized for all future NN training.

	A			
B	X			
		X		
				X
			X	

MLPRegressor Parameters	
Hidden Layer Size	300, 200
Activation	Relu
Solver	Adam
Alpha	0.001
Learning Rate	Constant
Random State	1
Max Iterations	500



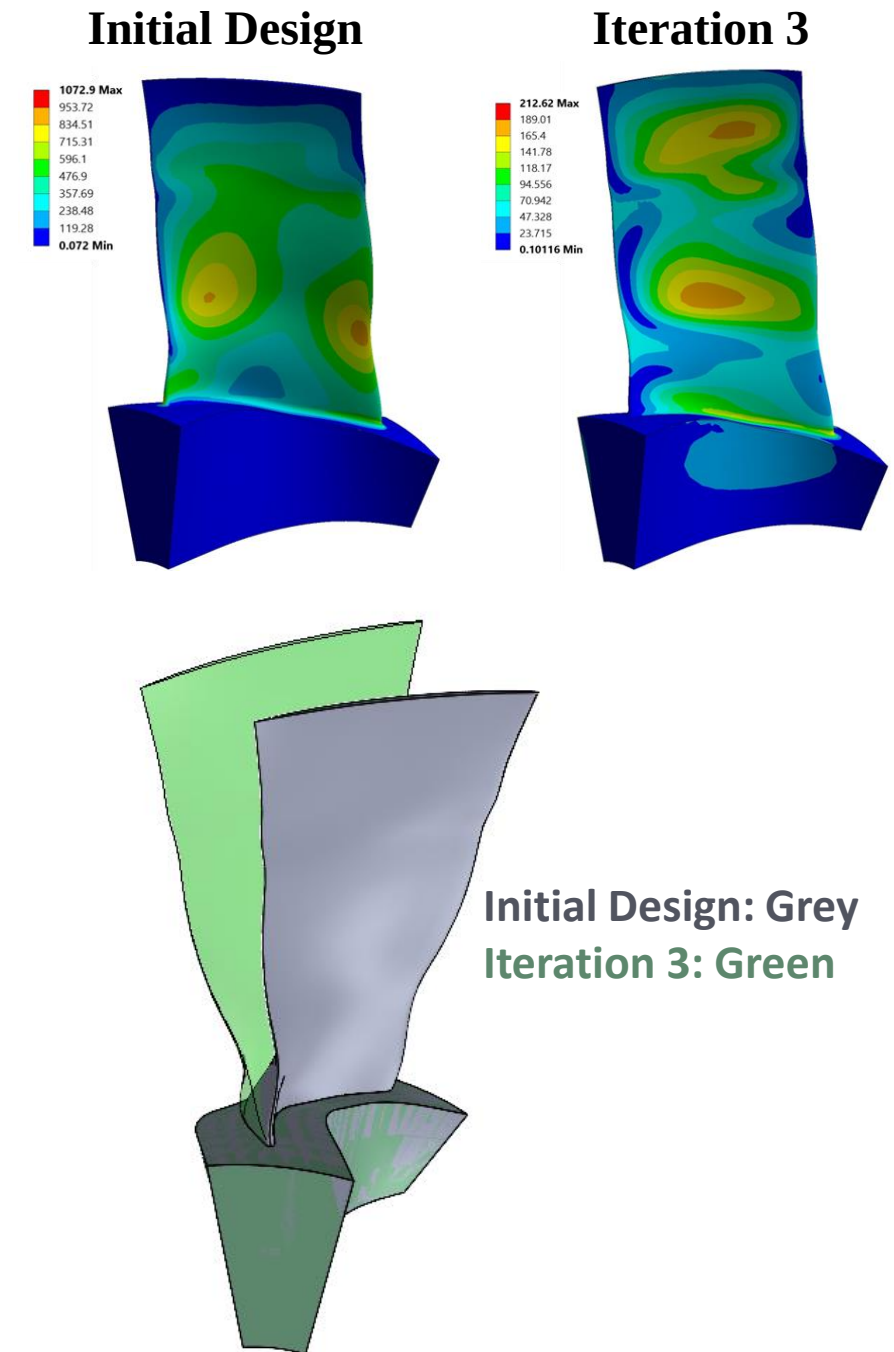
$$dm'_s = \frac{dm_s}{r_s}$$

$$m'_s = \int \frac{\sqrt{(dx_s)^2 + (dr_s)^2}}{r_s}$$

Machine Learning Based Optimization Methodology

Iteration	Initial	1	2	3	4
$\Delta m'_{25}$	0	0.0006	-0.0566	-0.0562	0.0007
$\Delta m'_{50}$	0	-0.0654	-0.1022	-0.1017	0.08802
$\Delta m'_{100}$	0	-0.09	-0.1079	-0.1083	-0.1174
$\Delta \theta_{25}$	0	-0.0278	-0.0953	-0.0949	-0.1396
$\Delta \theta_{50}$	0	-0.0855	-0.153	-0.1526	-0.1852
$\Delta \theta_{100}$	0	-0.09	-0.1549	-0.1554	-0.1595
FoS, NN	N/A	2.141	3.264	3.504	N/A
FoS, FEA	1.017	2.742	5.163	5.176	5.504
New NN FoS	N/A	2.849	4.983	4.954	5.122

- The first three iterations were run using the flow chart on the previous page
- Iteration 4 was conducted by combining data from the other three iterations
 - The NN architecture was also changed
 - NN architecture should be a future research topic
- The row 'New NN FoS' contains results from the prediction using the NN with all 3 sets of data and the updated architecture



Machine Learning Based Optimization Methodology

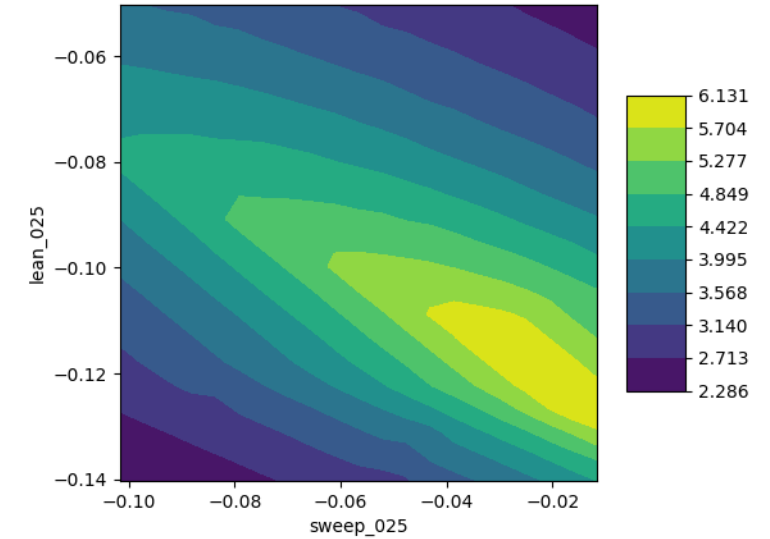
This method has numerous advantages:

- The cases for training can be run in a massive parallel operation.
 - The only limitation is licenses and compute power.
- The method is robust
 - Restart and simulation failure do not cause significant issue
- The data can be stacked.
 - Each iteration can add data to the stack
 - Over time, data from many different designs could be combined to train a more sophisticated model
- The neural net tends to smooth the data, so it works well with a gradient based optimization scheme
 - Genetic Algorithms also work well
- The data can be visualized
 - An added benefit of the NN is the ability to visualize the design space

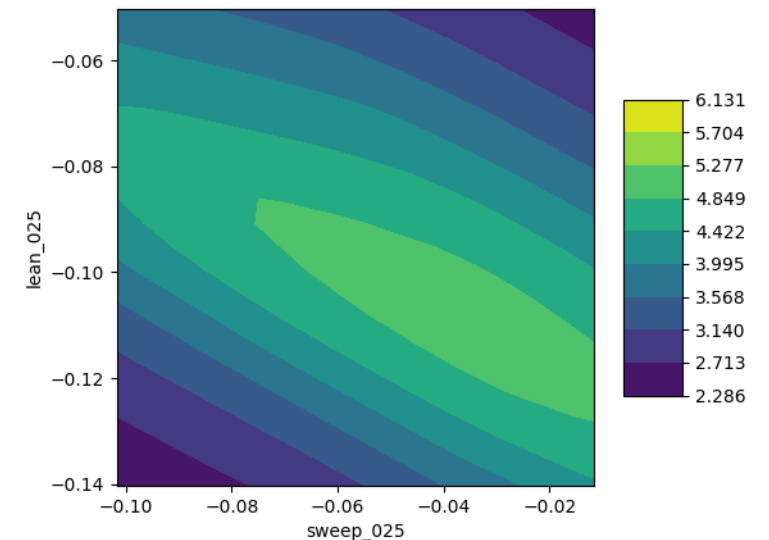
Next Steps:

- Add thickness as a DV
- Add high temperature material properties
- Change to Nastran to achieve quicker, parallel runs

FEA Map of Space



NN Map of Space



```

import sys
from PyQt5.QtWidgets import QApplication, QMainWindow, QVBoxLayout, QSlider, QLabel, QWidget
from PyQt5.QtCore import Qt
import matplotlib.pyplot as plt
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as FigureCanvas

class InteractivePlot(QMainWindow):
    def __init__(self, VelocityMagnitude, DistanceFromWall):
        super().__init__()

        self.VelocityMagnitude = VelocityMagnitude
        self.DistanceFromWall = DistanceFromWall

        self.init_ui()

    def init_ui(self):
        # Create a QWidget as the central widget
        central_widget = QWidget(self)
        self.setCentralWidget(central_widget)

        # Create a QVBoxLayout
        vbox = QVBoxLayout()

        # Create and configure the plot canvas
        self.figure, self.ax = plt.subplots()
        self.canvas = FigureCanvas(self.figure)
        vbox.addWidget(self.canvas)

        # Create sliders for i_position and t_position
        self.i_position_slider = QSlider(Qt.Horizontal)
        self.i_position_slider.setMinimum(0)
        self.i_position_slider.setMaximum(len(self.VelocityMagnitude) - 1)
        self.i_position_slider.valueChanged.connect(self.update_plot)
        vbox.addWidget(self.i_position_slider)

        self.t_position_slider = QSlider(Qt.Horizontal)
        self.t_position_slider.setMinimum(0)
        self.t_position_slider.setMaximum(len(self.VelocityMagnitude[0, 0]) - 1)
        self.t_position_slider.valueChanged.connect(self.update_plot)
        vbox.addWidget(self.t_position_slider)

        # Create labels for i_position and t_position
        self.i_position_label = QLabel()
        vbox.addWidget(self.i_position_label)
        self.t_position_label = QLabel()
        vbox.addWidget(self.t_position_label)

        # Set the layout
        central_widget.setLayout(vbox)

        # Initial plot
        self.update_plot()

    def update_plot(self):
        # Get the current i_position and t_position from sliders
        i_position = self.i_position_slider.value()
        t_position = self.t_position_slider.value()

        # Update the labels
        self.i_position_label.setText(f'i_position: {i_position}')
        self.t_position_label.setText(f't_position: {t_position}')

        # Clear the previous plot
        self.ax.clear()

        # Plot the new data
        velocity_data = self.VelocityMagnitude[i_position, :, t_position]
        distance_data = self.DistanceFromWall[i_position, :]
        self.ax.plot(velocity_data, distance_data)

        # Add labels
        self.ax.set_xlabel('Velocity (m/s)')
        self.ax.set_ylabel('Distance from wall (m)')

```

I have data that I would like to plot in an interactive matplotlib graph. I'd like the interactive plot to be controlled using PyQt5. Please write a Python code to do this based on the following information.

The data to plot has the following format:

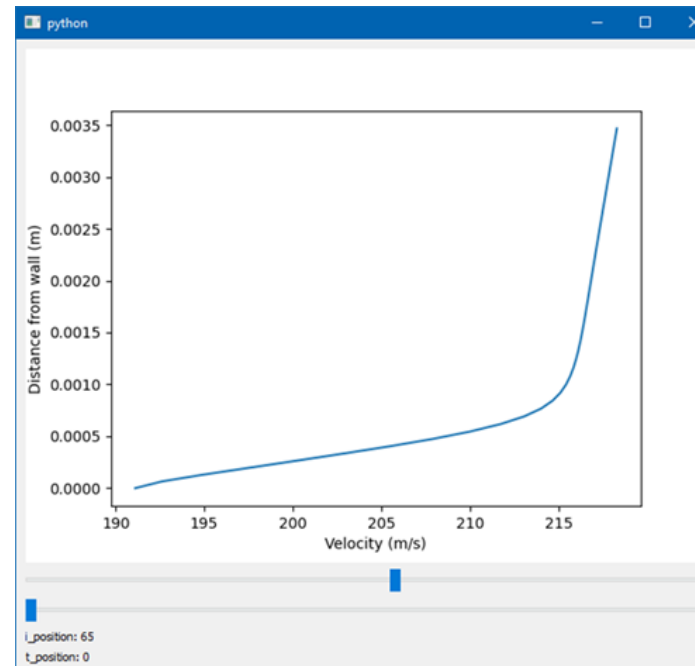
VelocityMagnitude[i_position, j_position, t_position]

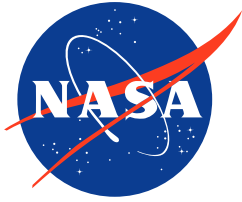
It also has a corresponding set of data, which is the distance from the wall.

DistanceFromWall[i_position, j_position]

I would like this to be its own class that takes inputs for VelocityMagnitude and DistanceFromWall as its inputs. I want to be able to toggle i_position and t_position in the GUI and plot all j points. The GUI should print the i_position and t_position to the screen. The toggles should be a horizontal sliders. The velocity variable should be plotted on the x-axis, and the distance should be plotted on the y-axis. Please add labels to them including units, this is all in standard mks. I would like the plot to be a line plot without markers.

I also want this to be easy to import, so please make a separate function that takes care of all the extra PyQt stuff like the show method.





ChatGPT and Design Optimization

- ChatGPT explained existing OpenMDAO code
 - OpenMDAO is an open-source multidisciplinary analysis and optimization Python package written at NASA GRC
 - In this work, OpenMDAO is used to run cases and to perform optimization on the Neural Network model of the design space.
 - ChatGPT provided an excellent explanation of the existing code. It was even able to reference OpenMDAO source code.
- ChatGPT wrote the code to sample the design space
 - ChatGPT provided Python code that produced Latin Hypercube Sampling (LHS) of the design space using the SciPy package
- Recommended the usage of Neural Networks (NN)
 - ChatGPT recommended that NNs be used to represent the response surface.
 - It also wrote the python code for training the NN.

