Optimization of 3D printed objects

Claudius Taylor, Tom Wilson, Dr. Hong Lin

May 2019

Introduction

In traditional manufacturing, complexity is expensive. The cost of complexity takes the form of work-in-progress, tooling, and time to prototype. These costs can be hard to estimate because they depend on many factors. (Gorguluarslan et al. 2017) In contrast, in three dimensional printing, complexity is free. There is a clear trade off between the strength of an object and the amount of material used to print the object. In this project, we aim to show a method for simultaneously optimizing a 3D printable object for both strength and cost. Three-dimensional printing is ready to become a viable alternative to conventional manufacturing. (Chen and Gabriel 2016)

The goal of this project is to create an end-to-end flow from STL file to optimal G-Code file. In order to achieve this goal, we have to design an experiment, slicing the STL file in many different ways. We chose to focusing on the infill parameters. we collect the metadata from the g-code output and enrich with some previously collected tensile strength data. Then we can model both the strength of the object and the amount of filament used to print the object. With those models in hand, we define a cost function that is proportional to filament used minus strength. Using a differential evolution algorithm to find the minimum cost. Majeed, Lv, and Peng (2018) provides a good basis for the application of big data analytics in the additive manufacturing process.

Blender is an open-sourced design tool that can output .STL files. STL files specify the surface of a three-dimensional object in terms of triangles on the surface of the object. In the first phase, the lattice structure is generated using the mesh information of the structure geometry. (Gorguluarslan et al. 2017) A consequence of triangular faces is that they require significantly less computational time because neither projection of the polyhedron faces onto the appropriate coordinate planes nor reduction using Green's Theorem are necessary. (Eberly 2002)

Slic3r is an open-sourced tool written in perl which can convert an stl model into printing instructions. Gcode format refers to the fact that many robot movements needed for three dimensional printing start with the character 'G'. Slc3r creates horizontal slices (layers) and generates tool-paths needed to fill them. It can also calculate the amount of filament to be extruded.

The result of our optimization shows an optimal layer height at 6, fill density at 46%, fill-every-layers at 5, wall thickness at 5, and nozzle temperature at 277 degrees. The optimal slicing uses 111.5cm3 of filament with a tensile strength 27 MPa compared to 67.2cm3 and 10 MPa for the default slicing parameters.

Workflow

We point our code at a single stl file. Slic3r provides 144 command line parameters, we loop through every combination of high and low settings for about 9 command-line parameters. this results in 2 to the 9th power combinations; a full factorial experiment. For each configuration we loop through the non-empty options and construct a command line string. Once the command string is constructed, we run slic3r and check for successful G-Code generation. We extract metadata from the resulting G-Code file. This metadata output includes the target parameter filament used in both volume (cm^3) and length in mm. During additive manufacturing, a huge amount of real-time big data is generated. (Majeed, Lv, and Peng 2018)

The process starts with an STL file. there are several software packages that are capable of modeling three-dimensional objects in STL format. We used Blender. our utility will slice a single STL file in many ways. After slicing, we use the metadata from the G-Code files and enrich with strength data which was experimentally determined. After optimization the optimal G-Code is constructed into a final command line string and executed by slic3r. Then we can open the G-Code file with software provided by any 3D printer manufacturer.

Our machine learning workflow starts by reading in strength data. We create an input matrix, X and an output vector, y. In this case y is tensile strength and X is every available feature except for strength, elongation, and

roughness. which would not be available prior to testing strength. We avoid circular reasoning. The X matrix is further divided into numeric and non-numeric features. The non-numeric features are dummy encoded and joined back to the numeric features resulting in an X that is fully numeric and appropriate for machine learning. Upon training a random Forest, we extract the feature importance. Based on the future importance, we select a human determined cutoff for the most important features. We selected the top 4. Then iterate and retrain on just those features. Those 4 features are wall thickness, nozzle temperature, infill density and layer height. To validate the model we inspect the predicted vs. actual an accurate model will fall along the Y equals X line (45 degree line). Further inspecting the residual to ensure that it is normally distributed and centered on zero. We follow a similar process for filament usage. First we have to clean the filament used column by splitting apart the volume used and length used. The result is stripped of units and white space until we have an actual number appropriate for a model output vector. which is now our new y. the X in the case of filament used is all of the metadata except for duplicates of the slic3r parameters and any features which have exactly zero variance. Fitting a random Forest gives us feature portents, we cut off the most important features, 3 this time. retrain then inspect predicted vs. actual and residuals should be normally distributed. In this case of filament usage, there is some non normality at the tails but overall our model is appropriate. Analytical models help develop and refine a more complete mental model of complicated data. (Steed et al. 2017)

Defining the cost function involves a lining up inputs of both models, 5 in total, ensuring consistent units and shape. During the cost function we can capture a callback o monitor progress of the optimization. we also have an opportunity to weight the simultaneous optimization. we started with 50/50 weighting. The input to the differential Evolution algorithm is the cost function we just defined along with the boundary condition for each input. we started with the minimum and maximum observed value in our original training data set. the result of our optimization shows that the optimal layer height is 6 fill density is the optimal infill layers is for the optimal wall thickness is 5 the optimal nozzle temperature is 277 degrees. Visualization is a vital component of design problem solving. (Gorguluarslan et al. 2017)

```
; generated by Slic3r 1.2.9

; external perimeters extrusion width = 0.60mm
; perimeters extrusion width = 0.63mm
; infill extrusion width = 0.63mm
; solid infill extrusion width = 0.63mm
; top infill extrusion width = 0.63mm

M107
M104 S200 ; set temperature
G28 ; home all axes
G1 Z5 F5000 ; lift nozzle

M109 S200 ; wait for temperature to be reached
G21 ; set units to millimeters
G90 ; use absolute coordinates
M82 ; use absolute distances for extrusion
G92 E0
G1 Z0.350 F7800.000
G1 E-2.00000 F2400.00000
G92 E0
G1 X66.925 Y67.762 F7800.000
G1 E2.00000 F2400.00000
G1 X68.677 Y66.283 E2.07096 F1800.000
G1 X70.830 Y65.490 E2.14192
G1 X72.009 Y65.384 E2.17858
G1 X127.991 Y65.384 E3.91071
G1 X130.250 Y65.781 E3.98168
G1 X132.238 Y66.925 E4.05264 F1800.000
G1 X133.717 Y68.677 E4.12360
G1 X134.510 Y70.830 E4.19456
```

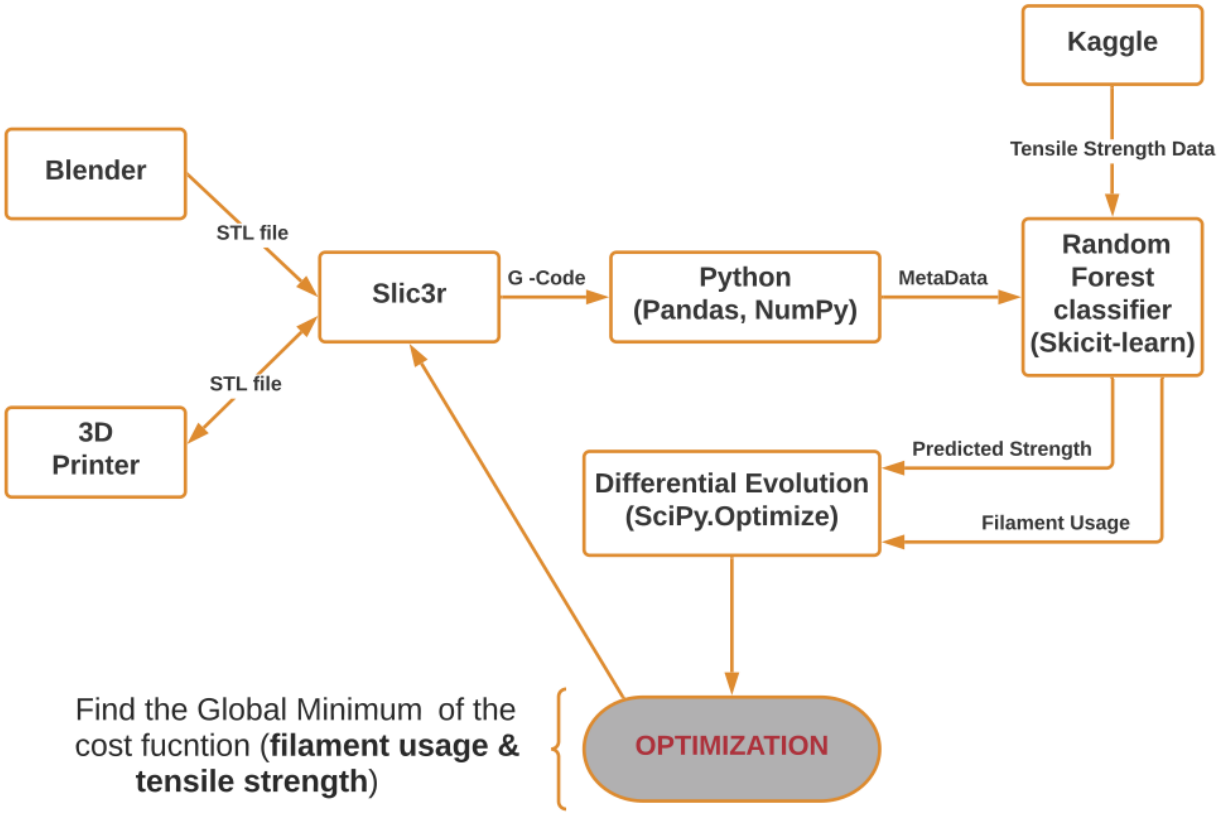


Figure 1: workflow diagram

```
G1 X134.616 Y72.009 E4.23121
...
...
...
; filament used = 15777.7mm (111.5cm3)
; fill_angle = 45
; fill_density = 46.1292%
; fill_pattern = honeycomb
; gap_fill_speed = 20
; infill_every_layers = 4
; infill_extruder = 1
; infill_extrusion_width = 0
; infill_overlap = 15%
; infill_speed = 80
; overhangs = 1
; perimeter_extruder = 1
```

Figure ?? shows an example of the contents of a gcode file.

Simulation and Modeling

Figure 1 shows our workflow.

In this project, we used FlashForge 3D printer to obtain a large cube as a STL file and pass it through a Slic3r to format it as a G-Code. We then model tensile strength from data obtain from Kaggle: (Ahmet Okudan, n.d.). After

applying that model to our metadata, we model filament usage as a function of similar inputs. When we modeled strength, we found that the most important parameter was wall thickness. Extrusion temperature, and fill density were also significantly predictive of tensile strength. Every other parameter we looked at was discarded, the top ten features are summarized here.

We used a random forest to predict tensile strength from the four most important parameters (wall thickness, nozzle temperature, infill density, and layer height). We used a 2nd random forest to predict filament usage from the 3 most important parameters (infill every layers, fill density, and layer height)

Predictors of tensile strength

column	description	units
wall thickness	Number of solid layers on surface	# of layers
nozzle temperature	Extrusion temperature	°C
infill density	percent of volume to fill with pattern	%
layer height	height of each layer	mm
fan speed	speed of cooling fan	% of max
bed temperature	temperature of platform	°C
print speed	speed of robot arm	mm/s
infill pattern	pattern of non-solid layers	Rectilinear or Honeycomb
material	material	ABS or PLA

```
metadata=pd.Dataframe()
for configuration in product(*configurations.values()):
    metarow = pd.Series(configuration,index=configurations.keys())
    cmd=["slic3r"]
    for key,value in zip(configurations.keys(),configuration):
        metarow[key]=value
        if value:
            cmd.append(str(key))
            if not isinstance(value,bool):
                cmd.append(str(value))
    cmd.append(input_file)
    check_output(cmd)
    with open(gcode_file_path) as gcode_file:
        for line in gcode_file.readlines():
            if line.startswith(';'):
                datum = line.strip('; \n').split('=')
                if len(datum)==2:
                    metarow[datum[0]]=datum[1]
        metadata = metadata.append(metarow,ignore_index=True)
```

root mean squared error of 2.05 MPa.

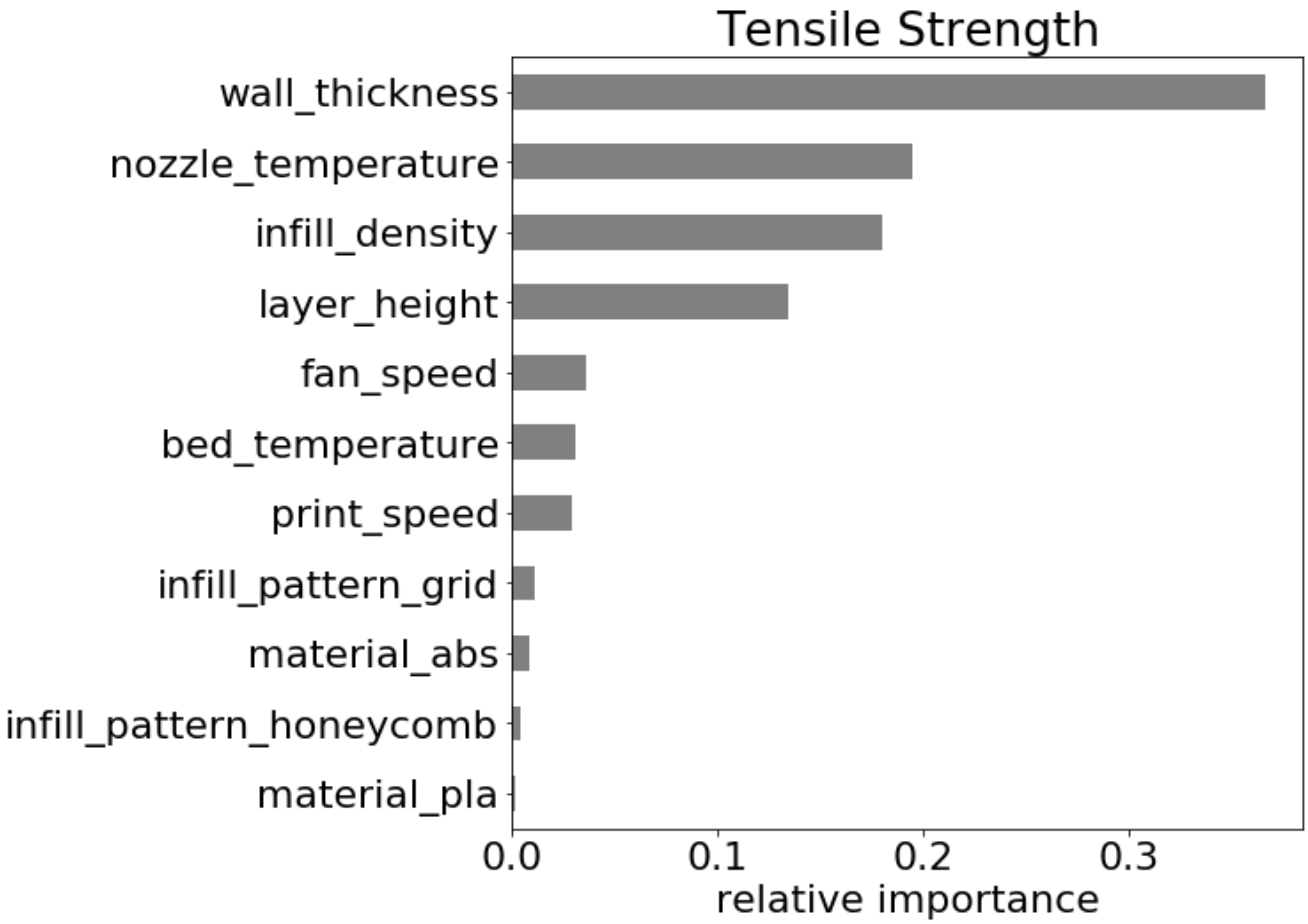


Figure 2: relative importance for tensile strength

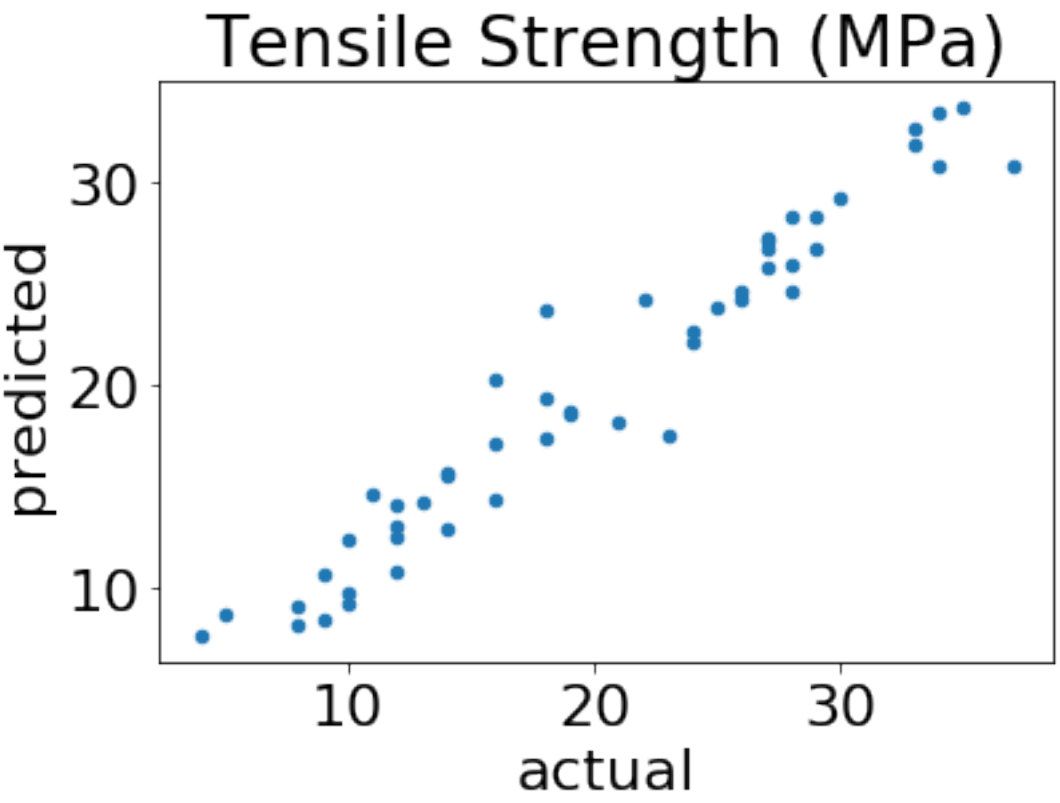


Figure 3: predicted tensile strength vs actual tensile strength

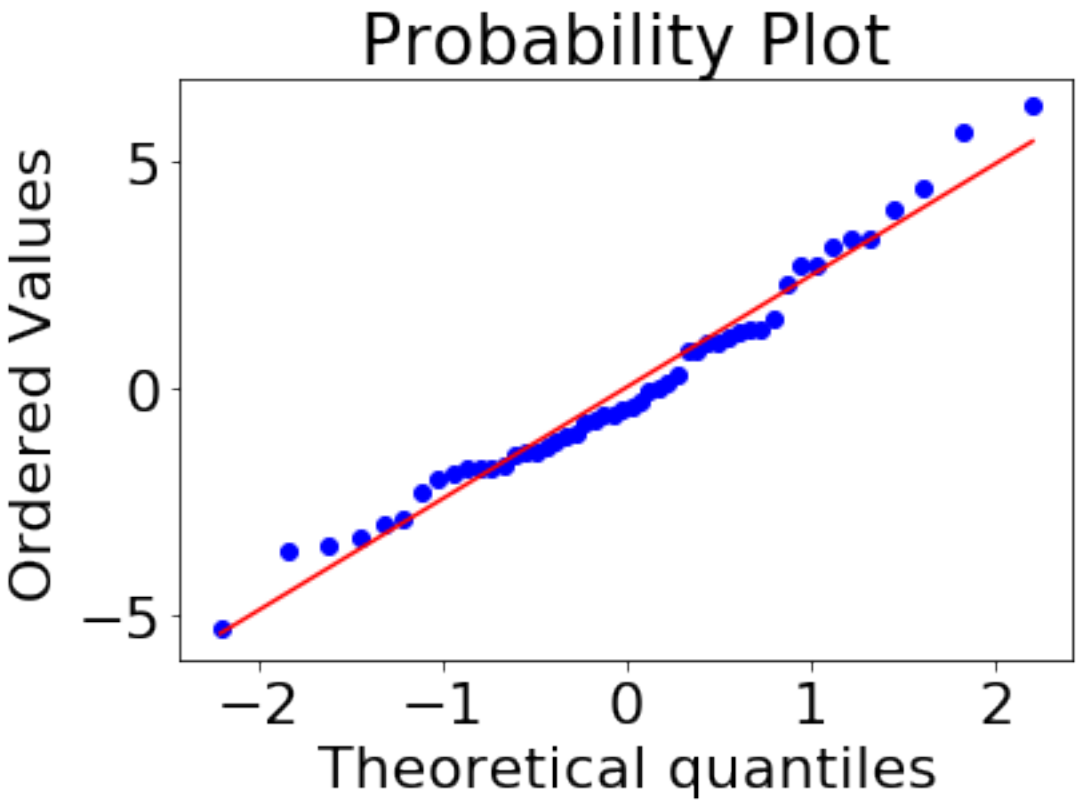


Figure 2 shows our the relative importance of the top 10 feature for predicting tensile strength.

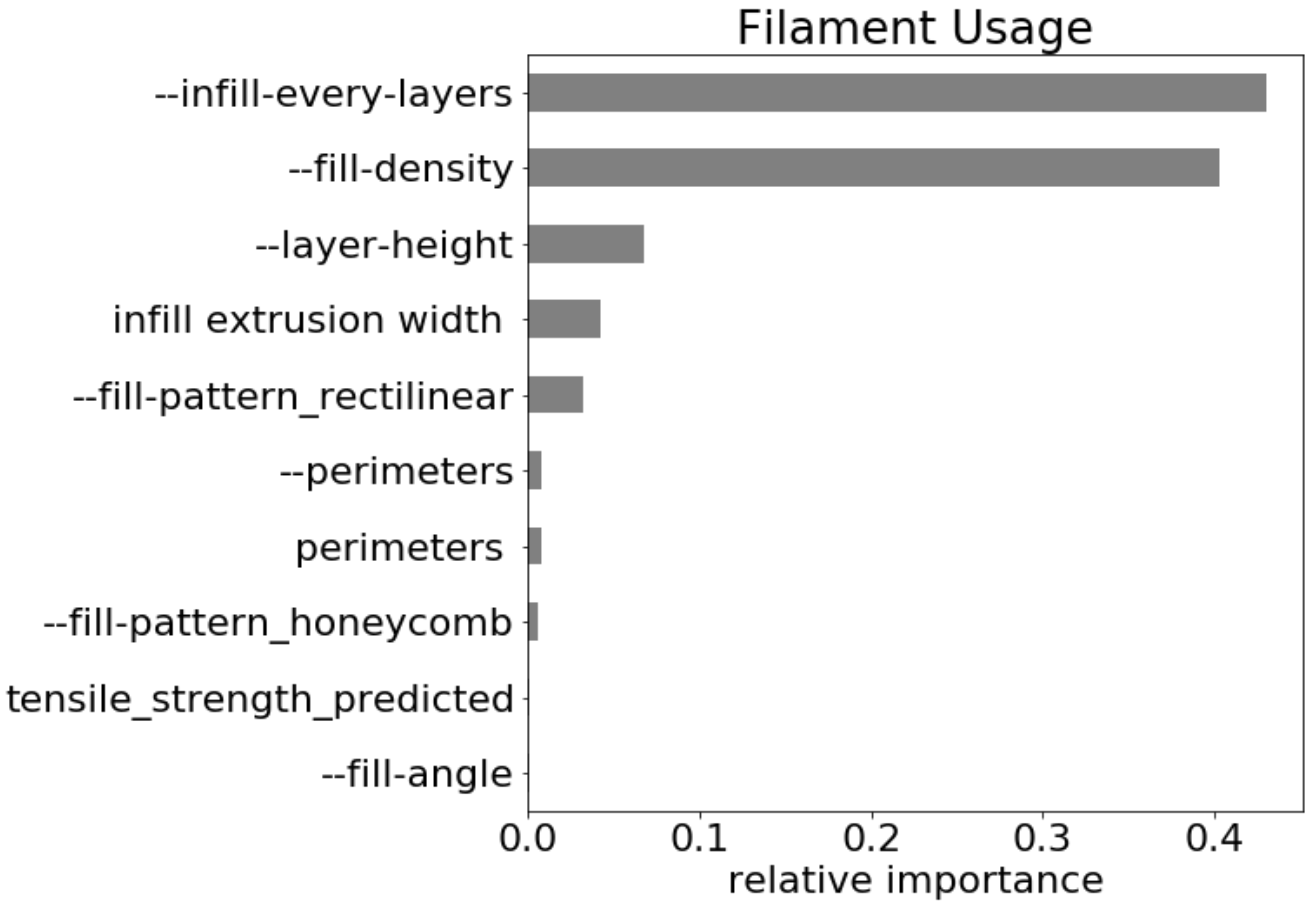


Figure 4: relative importance for predicting filament usage

Figure 3 shows predicted tensile strength plotted against actual tensile strength

Predictors of Filament usage

column	description	units
infill every layers	Infill every N layers	#
fill density	volume to fill with pattern	%
layer height	height of each layer	mm
infill extrusion width	filament flow width	mm
fill pattern	fill pattern	fill pattern
perimeters	horizontal solid layers	perimeters
fill angle	horizontal infill pattern	degrees

Figure 4 shows our the relative importance of the top 10 feature for predicting filament usage.

Figure 5 shows predicted tensile strength plotted against actual tensile strength

root mean squared error of $17.2cm^3$

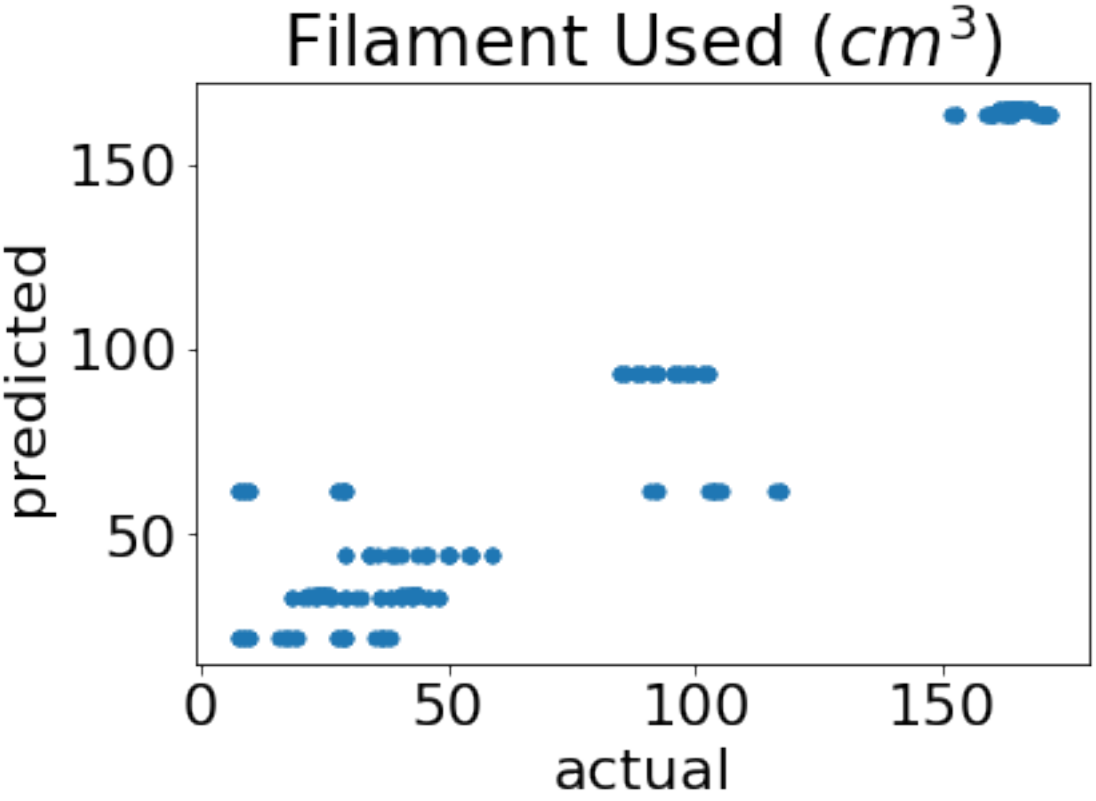
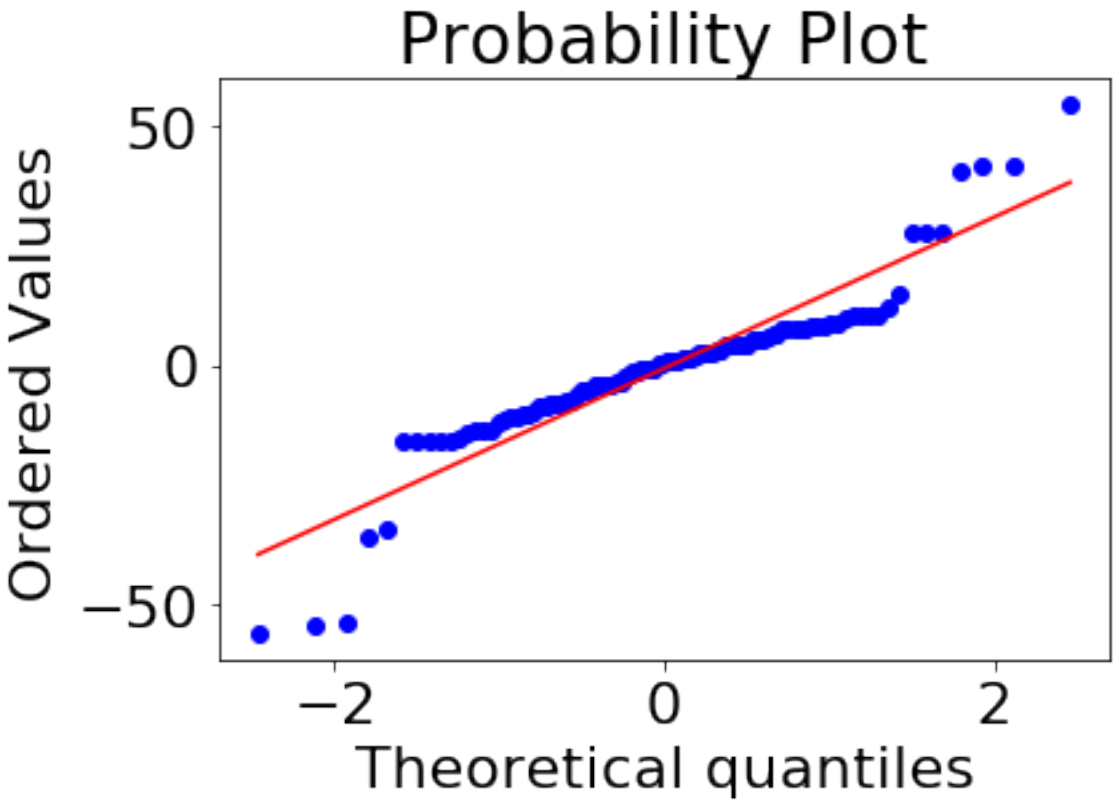


Figure 5: predicted filament used vs actual filament used



For filming usage, we found that the number of skipped layers, in-fill density and layer height are significantly important. Every other future was discarded. We used a 2nd random Forest to predict filament usage from these three parameters.

Our findings are consist with Chen and Gabriel (2016) who found thickness of Fill, Fill Rate, Extruder Speed, and Extruder Head Temperature have an effect on tensile strength.

Model Limitations

There are many failure modes in 3d printing, our approach is limited to meta data available in gcode files. The following table describes several failure modes that must be considered outside of optimization.

failure	effect	countermeasure
overhang	collapse	add support structures
shrinkage	deformation at corners	increase fan speed
pillowing	sagging infill	increase fill density
stringing	unintended structure	decrease retraction speed

While support structures are outside the scope of this project, Gorguluarslan et al. (2017) provides a framework to optimize support structures for additive manufacturing.

Optimization

In order to find a minimum cost we can ask a differential evolution algorithm to modulate these five parameters and search the space. Here we've shown how fill density collapses towards an Optimum just under 50% fill density. Looking at a default slicing rectilinear infill pattern and a single wall infill density about 13%, we compared it to our Optimum slicing infill about 50%.

The analysis of the results concludes that the process capability indices (Cp and Cpk), can be improved and at the same time optimal parameters can be identified using Six Sigma DMAIC (Define, Measure, Analyze, Improve, and Control) approach which is a win-win situation. (Chen and Gabriel 2016)

A BESO(Bidirectional Evolutionary Structural Optimization)-based optimization process is used to find the optimum struts' thickness distribution. (Tang et al. 2017)

The heterogeneous lattice structure optimized by the proposed method has a better performance compared to the homogeneous lattice structure. (Tang et al. 2017) Finally, the process can be optimized to obtain a larger feasible area for design and optimization. (Tang et al. 2017)

Relations between strut dimensions and mechanical properties can provide a feedback to lattice simulation and optimization model with more accurate material properties. (Tang et al. 2017)

In order to further improve the optimization of the support structures, the objective function (that only includes the material volume) could be extended, by taking into account the support removal and finishing costs.(Gorguluarslan et al. 2017)

The first example, i.e. the cantilever beam example, is used to show that the proposed framework can produce an optimized structure with minimal computational cost by considering the minimum manufacturing limit. (Gorguluarslan et al. 2017)

The second example is used to investigate the minimum diameter value that can be fabricated using the SLS process that will be used for the fabrication of the optimized lattice-based pillar structure. (Gorguluarslan et al. 2017)

The third example was a real-world application of the lattice structure optimization for a seat-bottom frame with a larger number of design variables when compared to the previous examples. (Gorguluarslan et al. 2017)

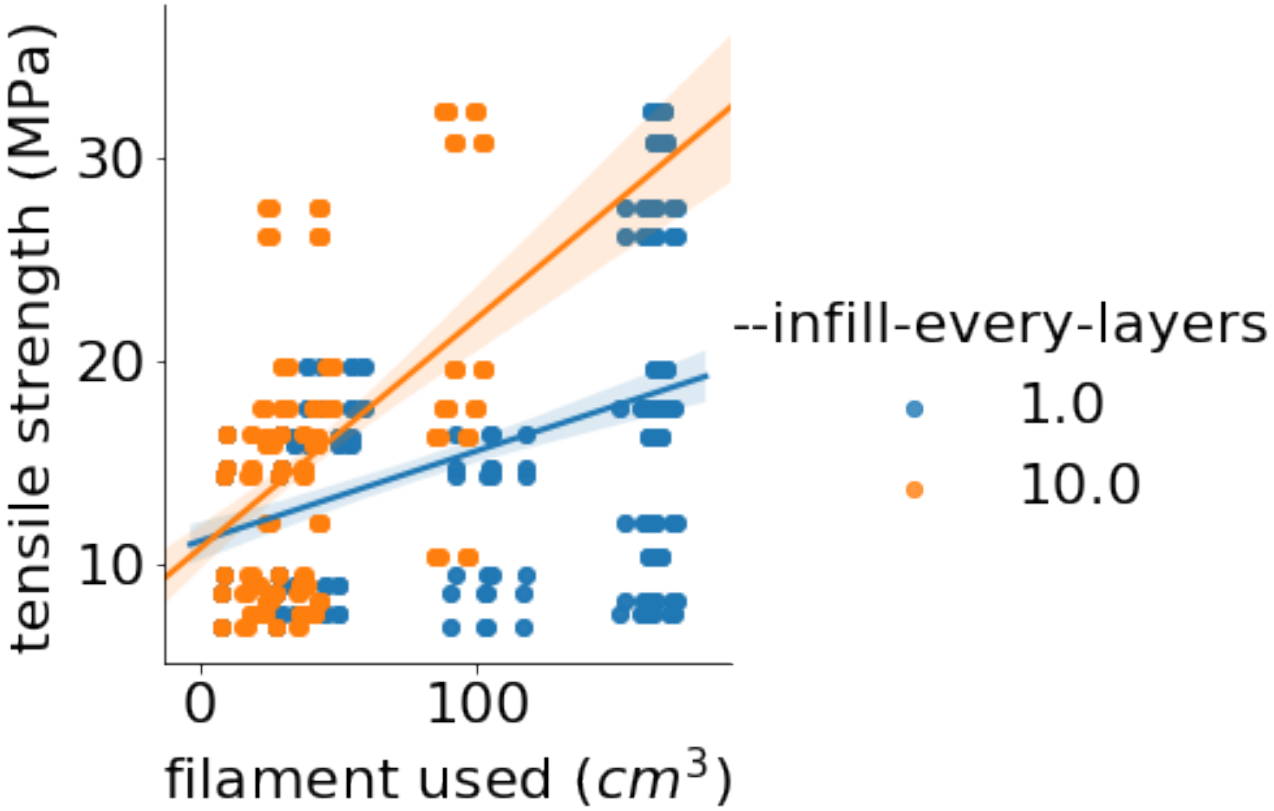


Figure 6: predicted filament used vs actual filament used

Moreover, it is shown in the second and third examples that the two-phase optimization framework can successfully find an optimized structure that can be fabricated once the minimum value is known for the specific additive manufacturing machine. (Gorguluarslan et al. 2017)

Gorguluarslan et al. (2017) has shown that using a genetic algorithm performs better traditional generation strategies.

It is also shown that the optimized structure has better performance compared to alternative existing methods. (Gorguluarslan et al. 2017)

Looking at a plot of strength vs filament usage we see an opportunity for optimization in the top left corner where using less filament results in reasonably strong object.

Figure 6 shows tensile strength plotted against filament usage.

Figure 7 shows how the progress of the optimization algorithm as it converges on the optimal fill density.

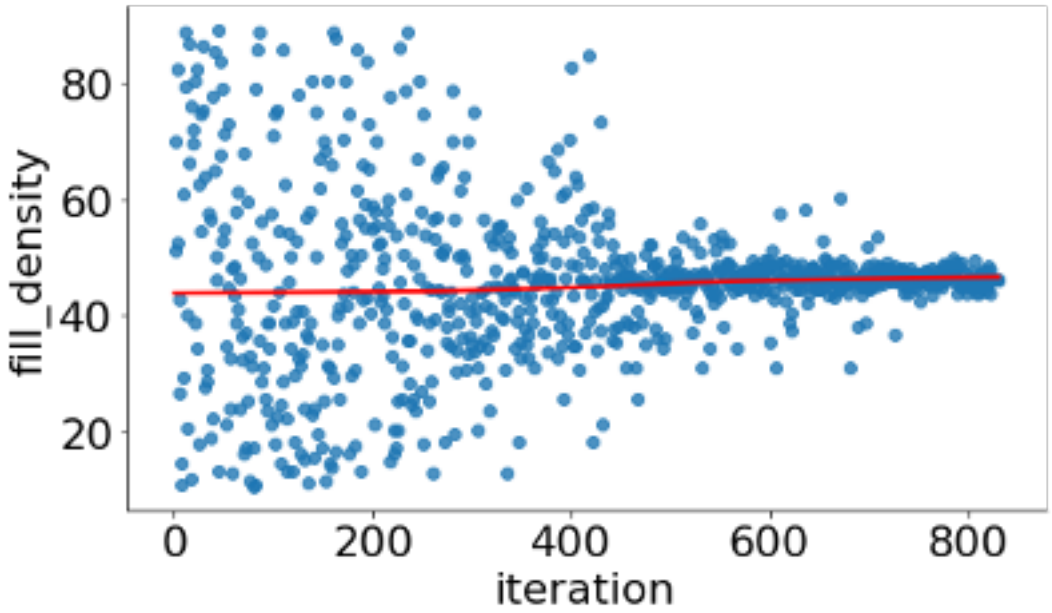


Figure 7: predicted filament used vs actual filament used

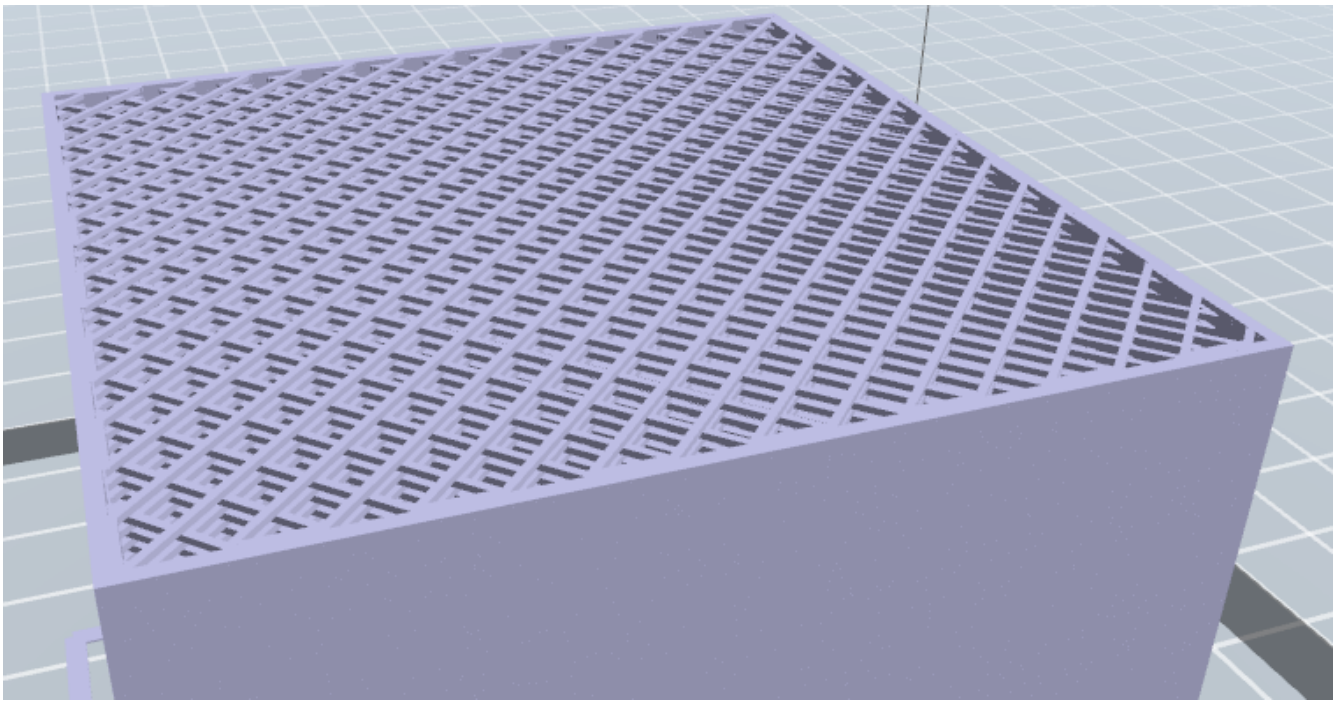


Figure 8: predicted filament used vs actual filament used

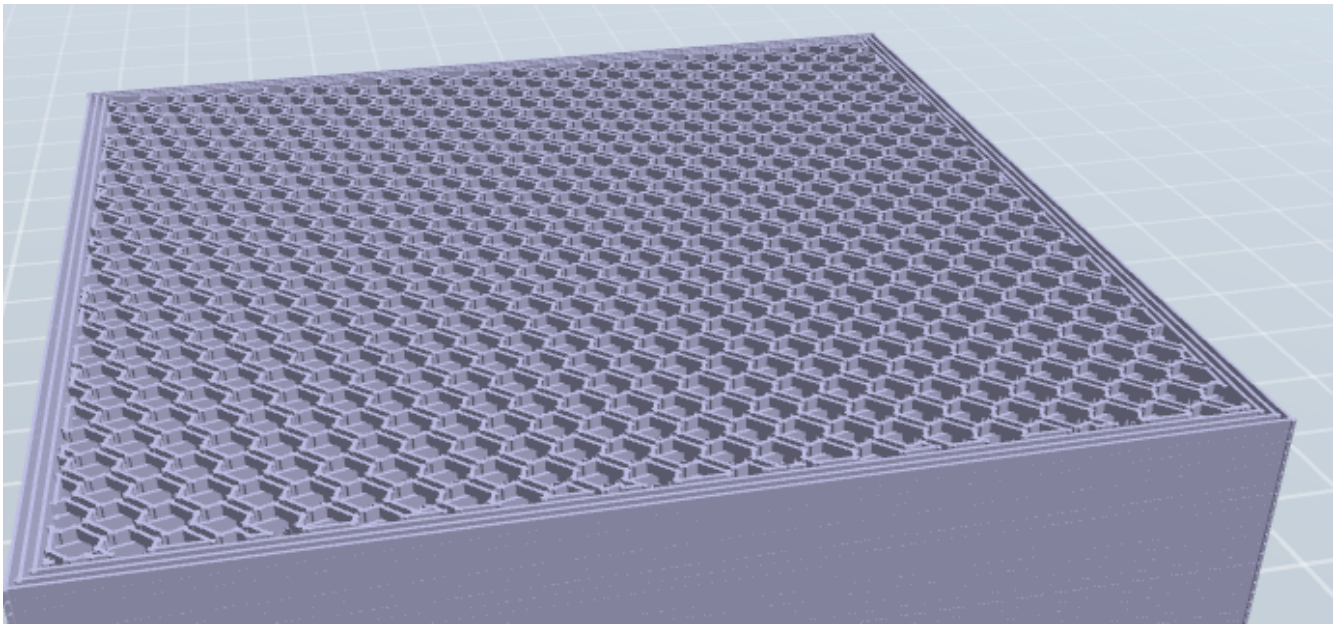


Figure 9: predicted filament used vs actual filament used

A default slicing:

Optimal slicing

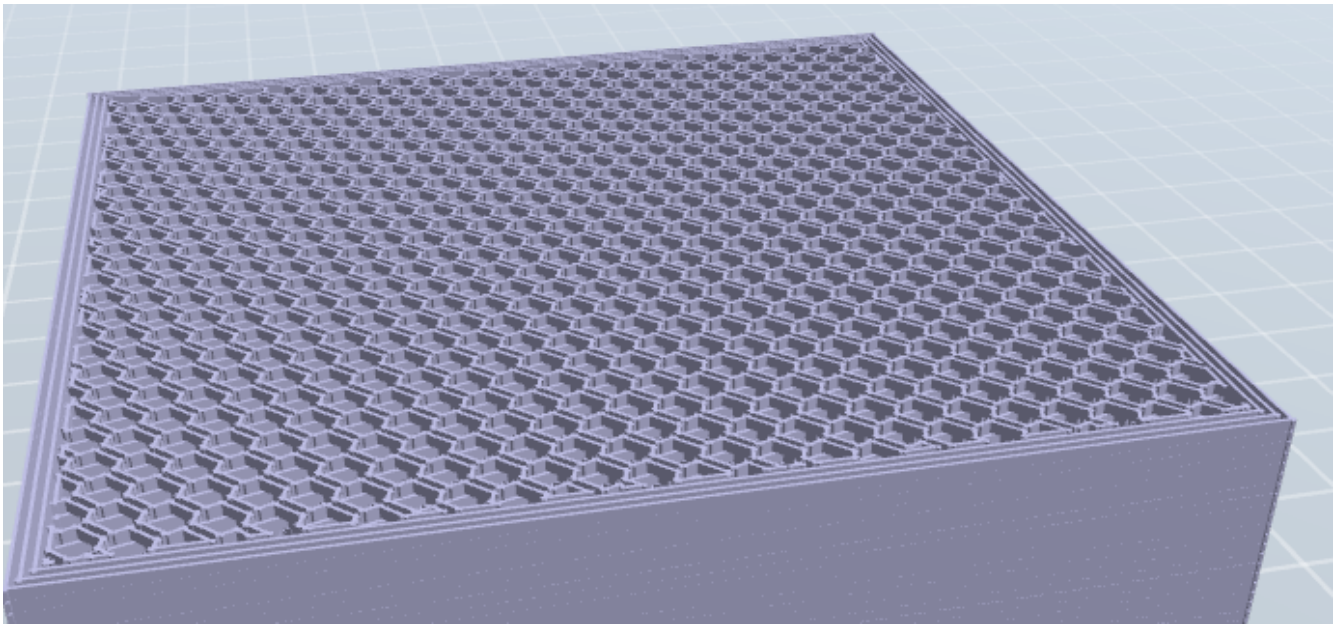


Figure 8 shows a cross sectional view of default slicing settings.

Figure 9 shows a cross sectional view of our optimal slicing settings.

Summary

This paper discusses a machine learning method namely Random Forest to evaluate the cost function of filament used to print objects and the resulting tensile strength of that object. The Random Forest resulted in lower cost.

For filament usage, we found that the number of skipped layers, infill density and layer height are important.

Acknowledgments

The completion of this project could not have been possible without the support and understanding of our family. We would also like to thank the Department and a special thanks also to Dr. Hong Lin.

References

Ahmet Okudan, TR/Selcuk University. n.d. "3D Printer Dataset for Mechanical Engineers." *Kaggle*. https://www.kaggle.com/afumetto/3dprinter/version/3.

Chen, Joseph C, and Victor Samuel Gabriel. 2016. "Revolution of 3d Printing Technology and Application of Six Sigma Methodologies to Optimize the Output Quality Characteristics." In *2016 Ieee International Conference on Industrial Technology (Icit)*, 904–9. IEEE.

Eberly, David. 2002. "Polyhedral Mass Properties (Revisited)." *Geometric Tools, LLC, Tech. Rep.*

Gorguluarslan, Recep M, Umesh N Gandhi, Yuyang Song, and Seung-Kyum Choi. 2017. "An Improved Lattice Structure Design Optimization Framework Considering Additive Manufacturing Constraints." *Rapid Prototyping Journal* 23 (2). Emerald Publishing Limited: 305–19.

Majeed, Arfan, Jingxiang Lv, and Tao Peng. 2018. "A Framework for Big Data Driven Process Analysis and Optimization for Additive Manufacturing." *Rapid Prototyping Journal*. Emerald Publishing Limited.

Steed, Chad A, William Halsey, Ryan Dehoff, Sean L Yoder, Vincent Paquit, and Sarah Powers. 2017. "Falcon: Visual Analysis of Large, Irregularly Sampled, and Multivariate Time Series Data in Additive Manufacturing." *Computers & Graphics* 63. Elsevier: 50–64.

Tang, Yunlong, Guoying Dong, Qinxue Zhou, and Yaoyao Fiona Zhao. 2017. "Lattice Structure Design and Optimization with Additive Manufacturing Constraints." *IEEE Transactions on Automation Science and Engineering*, no. 99. IEEE: 1–17.