Topological and structural optimization of simply supported beam under uniform distributed load

Jackson Sammartino
EGM6366
December 11, 2025

***Abstract* – Given an initial design domain of a simply supported 8 m x 1 m plate with thickness 0.02 m under uniform downward distributed load of 20 kN/m, the topology and structural form was optimized using MATLAB programming and Abaqus finite element analysis software. The objective was to minimize weight while not exceeding a von Mises stress of 250 MPa. A symmetry model was used in a topology code to generate the conceptual shape, which was then parameterized with four design variables. Latin hypercube sampling was used to generate 20 samples, which underwent FEA simulation. The resulting weight and stress data was fit to the four design variables using polynomial response surfaces and Kriging surrogate models, and cross-validated. Using the *fmincon* function in MATLAB, an optimum design was computed and refined through a trial-and-error approach, resulting in a final structure with weight of 1647 N (73% decrease from initial) and maximum stress of 216 MPa.**

***Index Terms* – finite element analysis, structural optimization, surrogate modeling, topology optimization**

I. Introduction

The goal of this project was to design a simply supported beam, under a uniform distributed load, from a given domain (Fig. 1), such that the topology of the structure minimized compliance, the dimensions of the final shape minimized weight (cost), and the von Mises stress anywhere in the structure did not exceed a given limit of σ_{allow} = 250 MPa. Note that a simplified model was used to take advantage of symmetry and reduce computational effort. Figure 1 shows the initial model and topology, with the symmetry boundary condition being imposed by rollers on the left side, and the distributed force P = 20 kN/m.

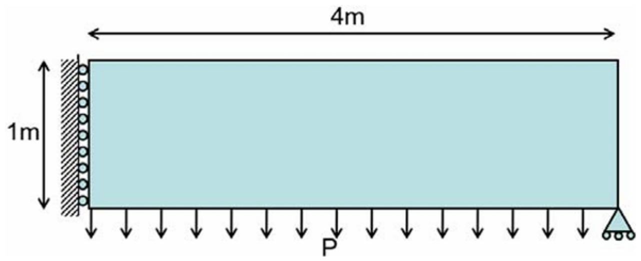

Fig. 1: Symmetrical model of the initial design domain and boundary conditions [1]. Note that the thickness *t* was fixed at 0.02 m into the page.

To accomplish this goal, first, topology optimization was performed in MATLAB using a modified verson of the 99-line "top.m" code written by Dr. Ole Sigmund [2]. Using a design-for-manufacturing (DFM) philosophy, a conceptual design shape was chosen from the topology optimization results. Briefly stated, DFM prioritizes large feature sizes and simple geometry to make manufacturing faster and cheaper.

Then, four shape variables were chosen for weight optimization, and their low and high bounds were set based on the constraints of the design domain (Fig. 1). Using a Latin hypercube sampling (LHS) method in MATLAB, 20 sample designs were generated, programmed into Abaqus finite element software, and simulated under the given loading (Fig. 1). Key output variables were the weight of the design *W* and the maximum von Mises stress σ.

From these results, quadratic polynomial response surfaces (PRS) and Kriging surrogate models were fit to *W* and σ, cross-validated, and used to search for an optimum design. If the optimum design was different from sample locations, then the surrogates were rebuilt and the process redone with the optimum as a sample itself [1]. A final optimized design was chosen which satisfied all constraints and minimized weight.

II. Procedure and Results

Topology Optimization in MATLAB

The topology of the structure was chosen such that it minimized compliance with a fixed volume fraction *f* of 0.4 (~40% of the design domain was filled with material). As shown in Table I below, this code was ran many times with different input parameters in MATLAB using the "top.m" code structure written by Dr. Ole Sigmund, modified (Appendix) for a uniform distributed load [2]. The number of elements in the horizontal direction *nelx* was always four times the number of elements in the vertical direction *nely* (Fig. 1), but they were varied along with the minimum feature size r_{min} and penalization power *p* using a DFM approach.

Table I: Topology Optimization Input Parameters

Iteration	nelx	nely	p	r_{min}
1	80	20	3	1.5
2	160	40	3	1.5
3	100	25	3	2
4	**100**	**25**	**3**	**2.5**
5	100	25	2	2.5
6	100	25	4	2.5

Table 1: Some example iterations of input parameters used for topology optimization with top.m MATLAB code [2]. Iteration 4 was selected.

The first iteration (Fig. 2), with input parameters recommended by the author of the code, produced a decent result, with no small or sharp features and some clearly identifiable shape design parameters. This was the smallest element density that worked in this manner, whereas increasing the element density rapidly increased computation time and worsened designs from a DFM perspective (Fig. 3). Still, there were some smaller features that were not ideal, so the minimum size filter was increased to 2, then to 2.5, resulting in the final topology (Fig. 4) used for the next stage of the design process. The penalization power was also changed as an exercise but affected the convergence of the algorithm. Note that AI software was used only to remove background borders from MATLAB (Figs. 2-4) [3].

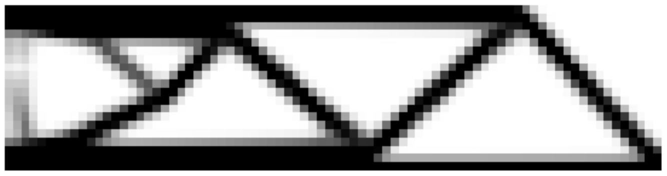
Fig. 2: Initial topology optimization results from Table I [2], [3].

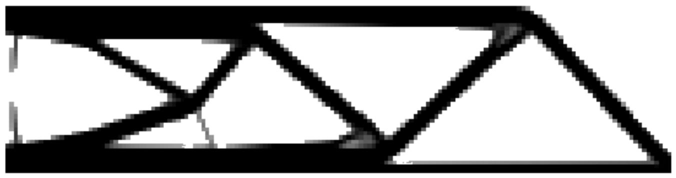
Fig. 3: Example of fine mesh topology optimization results from Table I, which was not favorable over coarser meshes [2], [3].

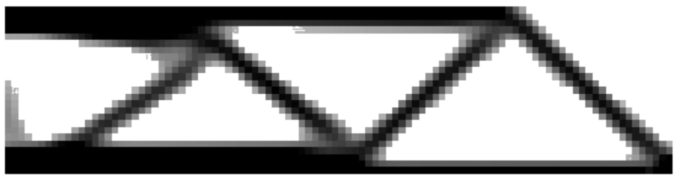
Fig. 4: Final topology selected from DFM perspective [2], [3].

Design Parameterization

Using the final topology optimization result (Fig. 4), design parameterization was performed to identify which features could be modified during weight optimization. Four shape variables were chosen (Fig. 5) to balance design flexibility with surrogate model complexity: the width of the two larger sections on the top and bottom of the left end (A), the width of the two smaller sections on the top and bottom of the right end (B), the width of all four diagonal sections (C), and the radius of all eleven interior fillets (D). The following Table II shows the four design variables and their bounds, which were tested and set based on geometric compatability (fully defined sketch) in the Abaqus CAD software, manufacturing limits, consideration of the intended volume fraction, or general engineering judgement. Evidently with these chosen variables, laser cutting appeared to be the best way to manufacture the beam topology. Accordingly, the minimum size for fillet radius (D) was set to 0.014 m, which is 0.7 times the material thickness 0.02 m [4].

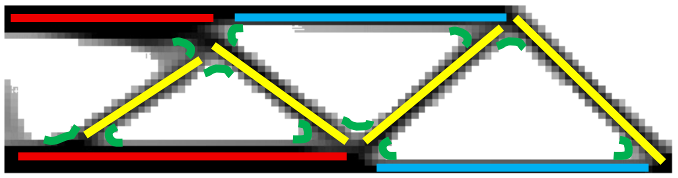
Fig. 5: Four design variables labeled on final topology [2], [3]. Note that for other dimensions, hand measurements (protractor and ruler) and scaling to the given design domain (Fig. 1) were performed using Figure 4.

Table II: Design Variables Summary

Symbol	**Lower Limit (m)**	**Upper Limit (m)**
A (red)	0.19 (1)	0.3 (1)
B (blue)	0.05 (2)	0.2 (1)
C (yellow)	0.07 (1)	0.2 (1)
D (green)	0.014 (2)	0.1 (2)

Table 2: Summary of the four chosen design variables and their limits, set based on (1) constraints of Abaqus CAD fully defined sketch geometry or (2) manufacturing or weight considerations.

The following Figures 6 and 7 respectively show examples of low-limit and high-limit designs in accordance with Table II.

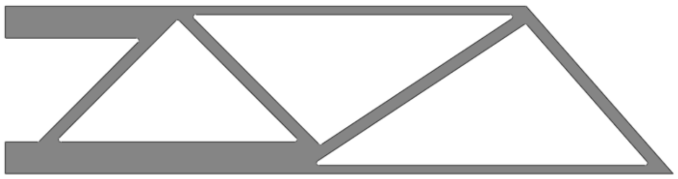
Fig. 6: example of design with all design variables at their low limit (Table II).

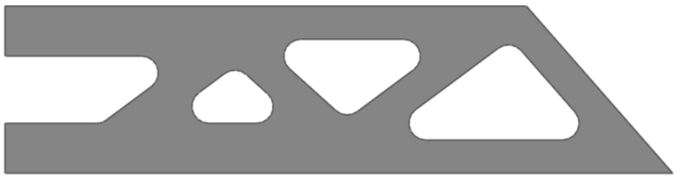
Fig. 7: example of design with all design variables at their high limit (Table II).

Design of Experiment and Finite Element Analysis

After the topology was parameterized with four design variables, surrogate modeling was done for the weight W and the maximum von Mises stress σ in the structure. To start, a sample size of 20 was generated using Latin hypercube sampling (LHS) with the *lhsdesign* function in MATLAB (Appendix), then scaled according to the upper and lower limits of each variable (Table II). Each design was implemented in Abaqus CAD software and simulated under the given loading (Appendix). The resulting maximum von Mises stress was used as a response variable, as well as the weight of the design, where density ρ was 7835 kg/m^3 and g was 9.81 m/s^2 [1].

$$W = mg = \rho V g = \rho A t g = (1537.227)A$$

After surrogate fitting, cross-validation was performed to determine whether more samples were required or not. Table III below shows the 20 LHS samples used, Table IV shows the resulting response variables, and Figure 8 shows a general case of the deformed geometry and stress distribution.

Table III: 20 LHS Sample Inputs

Sample	A (m)	B (m)	C (m)	D (m)
1	0.1969	0.1711	0.1381	0.0707
2	0.2829	0.1670	0.1779	0.0822
3	0.2022	0.0641	0.1514	0.0360
4	0.2166	0.1019	0.1155	0.0487
5	0.2239	0.0698	0.1701	0.0913
6	0.2674	0.1038	0.0834	0.0941
7	0.2331	0.1834	0.1643	0.0305
8	0.2346	0.1341	0.1582	0.0332
9	0.2759	0.1501	0.1235	0.0566
10	0.2539	0.1473	0.1042	0.0224
11	0.2930	0.1566	0.0996	0.0229
12	0.2017	0.1856	0.0957	0.0399
13	0.2464	0.1288	0.1149	0.0640
14	0.2645	0.0812	0.1818	0.0972
15	0.2576	0.1165	0.0722	0.0180
16	0.2838	0.0743	0.1933	0.0475
17	0.2110	0.1931	0.0794	0.0697
18	0.2437	0.0513	0.1953	0.0851
19	0.1920	0.0931	0.1343	0.0607
20	0.2956	0.1234	0.1466	0.0742

Table 3: Samples generated from MATLAB *lhsdesign* function.

Table IV: Response Variables from FEA

Sample	W (N)	Max σ (MPa)
1	2873	185.5
2	3374	99.67
3	2344	272.8
4	2399	242.0
5	2677	115.9
6	2509	238.2
7	3128	141.0
8	2861	219.5
9	2918	216.5
10	2670	225.9
11	2824	244.6
12	2652	209.1
13	2675	222.2
14	2921	97.14
15	2321	285.6
16	2912	117.8
17	2686	218.9
18	2766	143.0
19	2404	303.4
20	3009	188.5

Table 4: Two response variables, weight W and maximum von Mises stress σ (must be less than 250 MPa in the final design), for surrogate modeling. Each sample number corresponds to its row in Table 3.

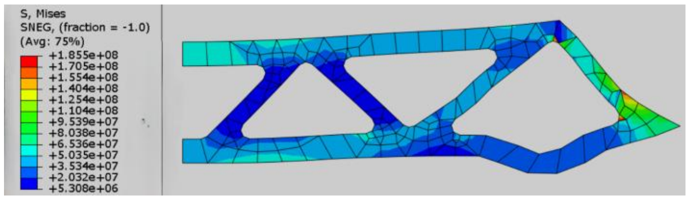

Fig. 8: Sample 1 upscaled deformed geometry and stress distribution (Pa) [3].

Polynomial Response Surfaces

It was decided that a quadratic polynomial response surface (PRS) was the minimum complexity of model form to capture the influence of the four design variables on each other. As such, 15 coefficients were determined from the data in Tables III and IV using a simple linear regression algorithm in MATLAB, for both the surrogate weight and surrogate stress (Appendix). A leave-one-out method of cross validation was performed, and percent relative errors e_{rel} were calculated. Table V below shows the entirely of the PRS process. The three columns on the left show the coefficients β for each variable term (A, B, C, and D were defined in Figure 5 and Table II) in the PRS, while the three right colums show the cross-validation relative error (%) for each sample.

Table V: Two Quadratic PRS Summaries

Term	β_W	β_σ	Sample	e_W (%)	e_σ (%)
1	-0.0477	0.0995	1	0.3788	2.6871
A	5.0789	-0.8688	2	0.9921	-83.810
B	6.1995	0.5442	3	-2.2471	-20.953
C	8.3795	0.2699	4	-0.1111	-1.1530
D	1.6450	-0.3001	5	0.5941	-38.742
A^2	-1.4473	1.7826	6	1.0346	-26.575
B^2	-2.7755	-2.5565	7	-0.8417	56.972
C^2	0.2709	2.6414	8	0.5276	-28.781
D^2	5.8046	-3.9400	9	-0.2640	-1.3406
AB	1.0155	1.3436	10	0.4379	8.6551
AC	-8.7641	-3.2788	11	-0.5876	-6.6124
AD	-0.1735	3.0143	12	0.5107	-36.623
BC	-3.4530	-1.6167	13	-0.1132	-0.6830
BD	-0.1238	0.0608	14	-1.0157	32.615
CD	-2.2106	-0.4779	15	-0.2076	1.6202
-	-	-	16	0.3451	-77.201
-	-	-	17	-0.4803	27.651
-	-	-	18	0.4933	28.722
-	-	-	19	-0.5728	22.193
-	-	-	20	0.3906	7.7997

Table 5: Two PRS coefficients and their cross-validated performances. Note that the coefficients of the two surrogates are times 10^3 and 10^4 for weight W and stress σ, respectively. The units of relative error are in percent (%).

Using the two PRS (Table V) with the *fmincon* function in MATLAB (Appendix) with the four design variables, their upper and lower bounds (Table II), the maximum stress inequality constraint of 250 MPa, and weight as the objective function, the algorithm efficiently returned an optimized design as summarized in Table VI below. Figure 9 shows objective weight function versus design iteration from *fmincon*. Note that this design had all variables at their low limit. However, due to high error in the stress PRS, the Abaqus FEA simulation of this design returned a maximum stress of 337.9 MPa (not shown), which violated the given constraint.

Table VI: Optimum PRS Design Summary (Erroneous σ)

A (m)	B (m)	C (m)	D (m)	W (N)	σ (MPa)
0.19	0.05	0.07	0.014	1659	175.7

Table 6: Optimum design, according to the weight and stress PRS used with *fmincon* in MATLAB. Note that all dimensions are at their low bound and the stress constraint was not satisfied in the FEA simulation.

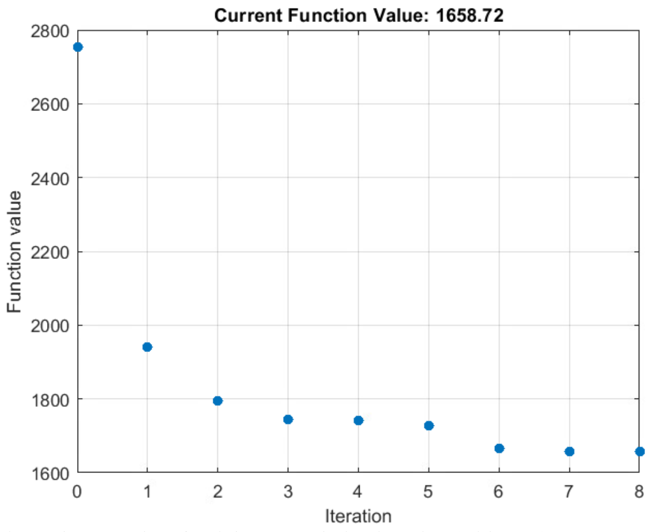

Fig. 9: *fmincon* plot of weight versus design iteration (with PRS) [3].

Kriging Surrogate Models

Two Kriging surrogate models were similarly fit to the data for the weight and maximum stress of the structure using a Gaussian regression process *fitrgp* function in MATLAB (Appendix). The models were then used in *fmincon* to find the optimum design (Table VII). The standard deviation *s* of the predictions at the optimum design was used to as a metric to evaluate the validity of the Kriging models (Table VIII). Details of *fmincon* convergence are shown in Figure 10 to the right. Simulation in Abaqus revealed a maximum stress of 242.6 MPa (satisfied constraint) and weight of 1925 N.

Table VII: Optimum Kriging Design Summary

A (m)	B (m)	C (m)	D (m)	W (N)	σ (MPa)
0.2164	0.05	0.07	0.1	1979	250.0

Table 7: Optimum design, according to the weight and stress Kriging surrogates used with *fmincon* in MATLAB.

Table VIII: Kriging Prediction Errors

-	Value	StDev	Error (%)
Weight W	1979 N	21.8033 N	1.102
Stress σ	250.0 MPa	49.6972 MPa	19.88

Table 8: Standard deviation (error) of the optimum Kriging predictions.

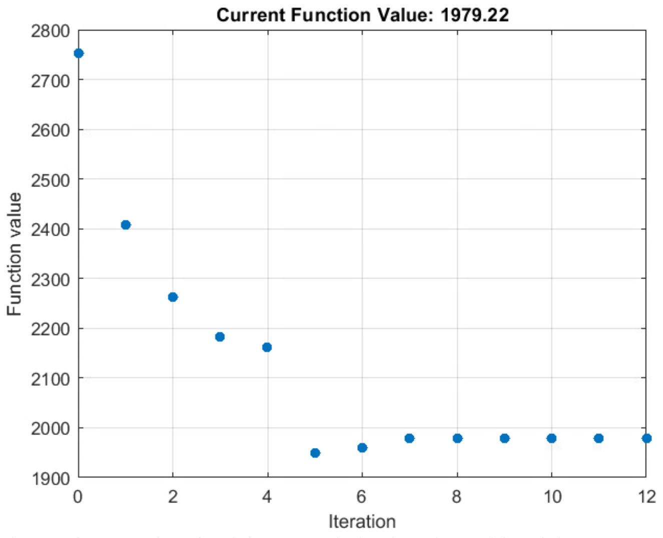

Fig. 10: *fmincon* plot of weight versus design iteration (with Kriging) [3].

Final Search for Optimum Design

Both the PRS and Kriging surrogates for weight were satisfactory; but, the Kriging stress surrogate were much favorable in terms of uncertainty and also consistency with FEA results. So, the Kriging method was used to refine the optimum design. The solved design (Table VII), with corrected stress of 242.6 MPa from FEA results, was used as a sample itself, and the process redone [1]. The only change in the output was in design variable A, which was reduced from 0.2164 m to 0.2026 m. However, the FEA simulation of this design violated the stress constraint (392.8 MPa).

From this point, a trial-and-error approach was taken to further reduce the weight of the structure while satisfying the stress constraint. Observations were made regarding the higher sensitivity of stress to changes in the design variables B and C, while dimension A did not have nearly as much effect. The size of internal fillets D only had adverse effects when very small, as stress concentrations and mesh distortion occurred around intersection points of multiple fillets. Table IX and Figure 11 below show the details of the final optimum design, while Figure 12 shows the fully-defined CAD sketch of the structure for replicability.

Table IX: Final Design Summary

A (m)	B (m)	C (m)	D (m)	W (N)	σ (MPa)
0.15*	0.05	0.07*	0.1*	1647	216.0

Table 9: Final optimum design, based on the Kriging surrogates and refined with a trial-and-error approach. *Note: dimension A was lowered to 0.15 m (below low bound of 0.19 m), dimension C was changed to 0.1 m but only for the leftmost diagonal section, and dimension D was lowered to its minimum value of 0.014 m for the 5 leftmost fillets only.

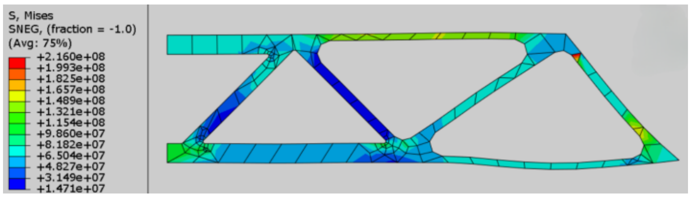

Fig. 11: FEA simulation of the final optimized design. Note that units are in Pa, and the stress constraint is oversatisfied (216 MPa versus 250 MPa).

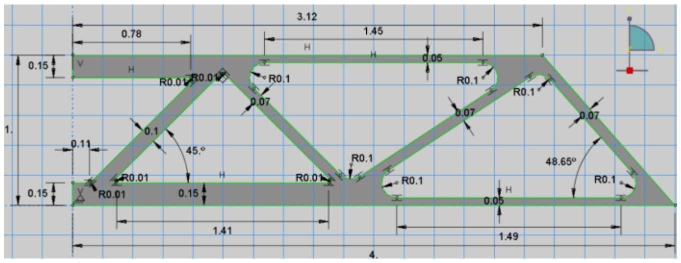

Fig. 12: fully-defined CAD sketch of final optimized design. Note that units are in m.

III. Discussion

Quality of PRS Models

The surrogate PRS for weight was an excellent fit, with only a few samples having cross-validation errors exceeding 1% (Table V). This can be attributed to the simple scaling between the geometry of the design and its area, which is directly proportional to the weight. Perhaps a quadratic PRS was even unnecessary for this simpler response variable. However, the surrogate PRS for maximum stress was an inconsistent, often poor fit. Many samples had cross-validation errors under 5%, but others exceeded 50% (Table V). This was a major limitation, and caused the PRS method to produce an optimum design that exceeded the maximum stress constraint when simulated in Abaqus FEA. The maximum stress in the structure was very sensitive to the four design variables and their relationships with each other. More random samples are recommended for building a more accurate quadratic stress PRS. If that still proves ineffective, then a more complicated model form should be used for the PRS.

Quality of Kriging Models

For similar reasons as the PRS, the Kriging weight surrogate was an excellent fit (1.102% error), but stress was much more challenging to model (19.88%) as shown in Table VIII above. Still, the uncertainty of the Kriging stress prediction for the optimum design was lower than the PRS error, which was why the Kriging method was chosen to iterate upon. The large uncertainty in the stress surrogate still limited the iteration process greatly, as it only took one iteration to violate the stress constraint. This is why manual methods were taken to further reduce the weight of the structure while not exceeding the limit of 250 MPa. Again, more random samples would mitigate this issue. Additionally, testing different values of the hyperparameters θ_k could improve results. Since all the design variables were on a similar scale and represented similar physical quantities, isotropic Kriging was used (same hyperparameter for all four variables). Anisotropic Kriging may be considered.

Comparison and Final Evaluation of Designs

The PRS and Kriging methods gave different optima, the latter of which was then refined manually with a trial-and-error approach. The following Table X and Figures 13-16 show the evolution of the design domain.

Table X: Design Evolution

Design	Weight (N)	Max σ (MPa)
Initial	6149	40.04
PRS	1566	337.9
Kriging	1925	242.6
Final	1647	216.0

Table 10: Performance of various designs in the design process.

Fig. 13: Initial design domain shape.

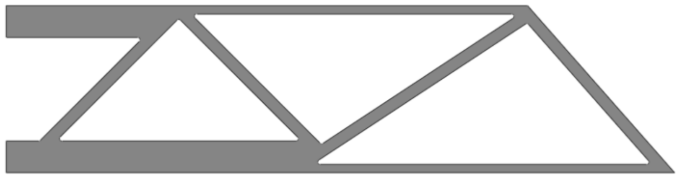

Fig. 14: PRS optimum design shape (stress constraint violated). Note that every design variable is at its minimum value.

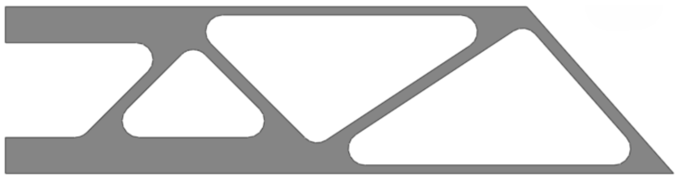

Fig. 15: Kriging optimum design shape (stress constraint satisfied).

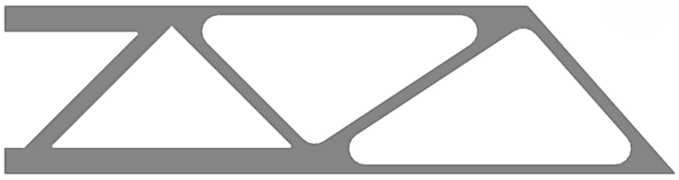

Fig. 16: Final optimum design shape (weight reduced further from Fig. 15). Note the large decrease of five fillets on the left side, and the small increase in width of the leftmost diagonal section.

This project resulted in a successful design that satisfied the stress constraint (216 MPa < 250 MPa) while reducing weight by over 73% (comparing Fig. 16 to Fig. 13). Still, many improvements could made, and it is very likely that the weight of the structure could be reduced further. The greatest limitation is in the stress surrogates for both PRS and Kriging models, which exhibited significant error compared to the weight surrogates (Tables V and VIII). More samples and different model parameters are recommended. Another important change would be in design parameterization: as evidenced by the final trial-and-error process (Table IX), more design variables are needed to effectively reduce the weight in different sections of the structure. The upper and lower limits of the design variables can also be expanded, specifically in the low range, to further reduce weight. CAD implementation was challenging and could be improved further with better design parameterization.

Appendix

Modification to Topology Optimization MATLAB Code

Three lines were changed in the initial “top.m” code, to change the loading from a concentrated downward force at the top left of the structure, to a uniform downward force distributed across the bottom nodes [2].

```
bottomNodes = (0:nelx) + 1;
bottomDOFs = 2*bottomNodes;
F(bottomDOFs) = -1/(nelx+1);
```

Latin Hypercube Sampling MATLAB Code

The following code used the *lhsdesign* function in MATLAB to generate 20 samples for use in surrogate models.

```
clc; clear;
n = 20;
nvar = 4;
X = lhsdesign(n, nvar);
% Scale variables to lower and upper bounds
samples = zeros(n, nvar);
lower = [0.19, 0.05, 0.07, 0.014];
upper = [0.3, 0.2, 0.2, 0.1];
for ii = 1:1:nvar
  samples(:, ii) = X(:, ii)*(upper(ii) -
  lower(ii)) + lower(ii);
end
```

FEA Details

For replicability of results, Abaqus FEA procedures are described here. If not specified, default settings were used. SI units (kg, m, N, Pa) were always used. The Part was a 3D deformable planar shell of approximate size 10. The Material was an isotropic elastic type, with density 7835, Young’s modulus 200E9, and Poisson’s ratio 0.3, applied to a homogenous shell Section of thickness 0.02. In the initial Step, an x-symmetry boundary condition was applied to the entire left edge, while a vertical roller (fixed y-displacement) was applied to the bottom right point. In the loading Step, a uniform shell edge load was applied along the bottom edge, with a value of negative 20000 (downward) per unit undeformed length. Standard S4R shell elements and meshing settings were used. The Job was written and submitted, and the maximum von Mises stress was taken as a response variable. For the weight response variable, a mass properties Query was used and then multiplied by gravitational acceleration (9.81 m/s^2).

Acknowledgements

The author would like to thank Professor Nam-Ho Kim at the University of Florida for a very challenging but enjoyable year of solid mechanics (courses EGM6352 and EGM6365). This is likely the last academic assignment the author will ever complete before graduating on Saturday, December 13, 2025.

References

[1] N.H. Kim, “Structural optimization and sensitivity analysis module,” course EGM6365 lectures and assignments, University of Florida, Gainesville, FL, USA, fall semester 2025.

[2] O. Sigmund, “A 99 line topology optimization code written in MATLAB,” MATLAB code script, *Journal of Structural and Multidisciplinary Optimization*, Volume 21, pages 120-127, 2001.

[3] OpenAI, “ChatGPT,” artificial intelligence model used only for removing colored backgrounds from figures, accessed 2025.

[4] Acute Laser, “Considerations when laser cutting metal: minimum dimensions for laser cutting,” website article, 2023.

Additional appendices for PRS and Kriging MATLAB codes:

```
clc; clear;
% 2x quadratic PRS surrogates with 4 design variables and 20 samples
samples = [0.1969 0.1711 0.1381 0.0707;
    0.2829 0.1670 0.1779 0.0822;
    0.2022 0.0641 0.1514 0.0360;
    0.2166 0.1019 0.1155 0.0487;
    0.2239 0.0698 0.1701 0.0913;
    0.2674 0.1038 0.0834 0.0941;
    0.2331 0.1834 0.1643 0.0305;
    0.2346 0.1341 0.1582 0.0332;
    0.2759 0.1501 0.1235 0.0566;
    0.2539 0.1473 0.1042 0.0224;
    0.2930 0.1566 0.0996 0.0229;
    0.2017 0.1856 0.0957 0.0399;
    0.2464 0.1288 0.1149 0.0640;
    0.2645 0.0812 0.1818 0.0972;
    0.2576 0.1165 0.0722 0.0180;
    0.2838 0.0743 0.1933 0.0475;
    0.2110 0.1931 0.0794 0.0697;
    0.2437 0.0513 0.1953 0.0851;
    0.1920 0.0931 0.1343 0.0607;
    0.2956 0.1234 0.1466 0.0742];
w = [2873 3374 2344 2399 2677 2509 3128 2861 2918 2670 2824 2652 2675 2921 2321
2912 2686 2766 2404 3009]';
sig = [185.5 99.67 272.8 242.0 115.9 238.2 141.0 219.5 216.5 225.9 244.6 209.1
222.2 97.14 285.6 117.8 218.9 143.0 303.4 188.5]';
x1 = samples(:,1); x2 = samples(:,2); x3 = samples(:,3); x4 = samples(:,4);
X = [ones(20,1) x1 x2 x3 x4 x1.^2 x2.^2 x3.^2 x4.^2 x1.*x2 x1.*x3 x1.*x4 x2.*x3
x2.*x4 x3.*x4];
beta_w = (X'*X)\(X'*w);
beta_sig = (X'*X)\(X'*sig);
% Leave-one-out cross validation
pred_w = zeros(20,1); pred_sig = zeros(20,1);
err_w = zeros(20,1); err_sig = zeros(20,1);
w_init = w; sig_init = sig; samples_init = samples;
init_beta_w = beta_w; init_beta_sig = beta_sig;
for ii = 1:1:20
    w_true = w(ii);
    w(ii) = [];
    sig_true = sig(ii);
    sig(ii) = [];
    val = samples(ii,:);
    samples(ii,:) = [];
    v1 = samples(:,1); v2 = samples(:,2); v3 = samples(:,3); v4 = samples(:,4);
    X = [ones(19,1) v1 v2 v3 v4 v1.^2 v2.^2 v3.^2 v4.^2 v1.*v2 v1.*v3 v1.*v4
v2.*v3 v2.*v4 v3.*v4];
    beta_w = (X'*X)\(X'*w);
    beta_sig = (X'*X)\(X'*sig);
    valx = [1 val(1) val(2) val(3) val(4) val(1)^2 val(2)^2 val(3)^2 val(4)^2
val(1)*val(2) val(1)*val(3) val(1)*val(4) val(2)*val(3) val(2)*val(4)
val(3)*val(4)];
    pred_w(ii) = valx*beta_w;
    pred_sig(ii) = valx*beta_sig;
    err_w(ii) = (w_true - pred_w(ii))/w_true;
    err_sig(ii) = (sig_true - pred_sig(ii))/sig_true;
    % Reset for next iteration
    w = w_init; sig = sig_init; samples = samples_init;
end
```

Continued appendices for PRS and Kriging MATLAB codes:

```
beta_w = init_beta_w; beta_sig = init_beta_sig;
% Optimize using fmincon
func_w = @(x) [1, x(1), x(2), x(3), x(4), x(1)^2, x(2)^2, x(3)^2, x(4)^2,
x(1)*x(2), x(1)*x(3), x(1)*x(4), x(2)*x(3), x(2)*x(4), x(3)*x(4)]*beta_w;
func_sig = @(x) [1, x(1), x(2), x(3), x(4), x(1)^2, x(2)^2, x(3)^2, x(4)^2,
x(1)*x(2), x(1)*x(3), x(1)*x(4), x(2)*x(3), x(2)*x(4), x(3)*x(4)]*beta_sig;
lowerb = [0.19 0.05 0.07 0.014];
upperb = [0.3 0.2 0.2 0.1];
x0 = (lowerb+upperb)/2;
obj = @(x) func_w(x);
cons = @(x) deal(func_sig(x) - 250, []);
options = optimset('PlotFcns','optimplotfval','LargeScale','off');
[xopt, wopt] = fmincon(obj, x0, [], [], [], [], lowerb, upperb, cons, options);
sigopt = func_sig(xopt);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc; clear;
samples = [0.1969 0.1711 0.1381 0.0707;
    0.2829 0.1670 0.1779 0.0822;
    0.2022 0.0641 0.1514 0.0360;
    0.2166 0.1019 0.1155 0.0487;
    0.2239 0.0698 0.1701 0.0913;
    0.2674 0.1038 0.0834 0.0941;
    0.2331 0.1834 0.1643 0.0305;
    0.2346 0.1341 0.1582 0.0332;
    0.2759 0.1501 0.1235 0.0566;
    0.2539 0.1473 0.1042 0.0224;
    0.2930 0.1566 0.0996 0.0229;
    0.2017 0.1856 0.0957 0.0399;
    0.2464 0.1288 0.1149 0.0640;
    0.2645 0.0812 0.1818 0.0972;
    0.2576 0.1165 0.0722 0.0180;
    0.2838 0.0743 0.1933 0.0475;
    0.2110 0.1931 0.0794 0.0697;
    0.2437 0.0513 0.1953 0.0851;
    0.1920 0.0931 0.1343 0.0607;
    0.2956 0.1234 0.1466 0.0742;
    0.2164 0.05 0.07 0.1];
% Last sample ^ is added after one iteration of fmincon >>
w = [2873 3374 2344 2399 2677 2509 3128 2861 2918 2670 2824 2652 2675 2921 2321
2912 2686 2766 2404 3009 1979]';
sig = [185.5 99.67 272.8 242.0 115.9 238.2 141.0 219.5 216.5 225.9 244.6 209.1
222.2 97.14 285.6 117.8 218.9 143.0 303.4 188.5 242.6]';
% Use Regression Gaussian Process function (aka ordinary Kriging)
krig_w = fitrgp(samples, w, 'Basis', 'constant');
krig_sig = fitrgp(samples, sig, 'Basis', 'constant');
% Optimize using fmincon
lowerb = [0.19 0.05 0.07 0.014];
upperb = [0.3 0.2 0.2 0.1];
x0 = (lowerb+upperb)/2;
obj = @(x) predict(krig_w, x);
cons = @(x) deal(predict(krig_sig, x) - 250, []);
options = optimset('PlotFcns','optimplotfval','LargeScale','off');
[xopt, wopt] = fmincon(obj, x0, [], [], [], [], lowerb, upperb, cons, options);
[w_hat, w_std] = predict(krig_w, xopt);
[sig_hat, sig_std] = predict(krig_sig, xopt);
```