Computational Design of Novel Materials

Luke H. Spencer

University of New South Wales at the Australian Defence Force Academy

This report details the final year project of computational design of lattice structures. The aim of this project was to create a base platform for the design of custom and novel lattice structures using computationally expensive evaluations coupled with a multi-objective, multi-concept optimization algorithm. A literature review on space filling polyhedra, lattice structure applications and design optimization of lattice structures is presented. The methods for automated lattice generation of three space filling polyhedral cells is detailed. Automated repeatable evaluation functions of lattice properties using CATIA and ABAQUS are detailed. The integration of lattice generation and evaluations into the optimization algorithm is described. Four lattice optimisation problems are presented with problem formulation and results to validate the novel lattice structure design platform. Discussion of results and recommendations for continuation of this project and future work is included at the end of this report.

Contents

I. Project Outline 1
II. Lattice Generation 2
III. Evaluation Functions 3
   A. CATIA 3
   B. ABAQUS 3
   C. Post Processing 3
IV. Problem Formulation 3
   A. Cube, Single-Objective 4
   B. Cube, Bi-Objective 4
   C. Multi Concept, Bi-Objective 4
   D. Cube, Bi-Objective (with ABAQUS) 5
V. Optimisation Results 6
   A. Cube, Single-Objective 6
   B. Cube, Bi-Objective 6
   C. Multi Concept, Bi-Objective 7
   D. Cube, Bi-Objective (with ABAQUS) 9
VI. Results Discussion 10
VII. Recommendations for Future Work 11
VIII. References 12

APPENDICES
Appendix A. Literature Review
Appendix B. Lattice Generation Methods
Appendix C. Lattice Generation and Evaluation Functions
Appendix D. ABAQUS Evaluation Functions
Appendix E. Post Processing Function Examples

Nomenclature

L = Length of lattice structure Block (y direction)
B = Width of lattice structure block (x direction)
H = height of lattice structure block (z direction)
a = length of cube or length of cuboctahedron (mm)
t = radius of strut for all concepts (mm)
I. Project Outline

Materials have long been used as indicators defining the technological progress of an era i.e. the stone age, the iron age, the silicon age etc. Over the last few decades we have seen the emergence of custom materials i.e. ones specifically designed for a purpose e.g. materials which can withstand extreme temperatures, stress, radiation, and corrosion to the ones which can self-regulate porosity, strength, conductivity and elastic properties. While advanced approximation schemes have been developed, and adopted by several industries e.g. aerospace in particular, there have been limited attempts to embrace the technology in materials design. To identify custom material structures involves solving an optimization problem. Such problems will contain nonlinear objectives and nonlinear constraints with a need for a global optimum solution. To solve such a problem, stochastic optimization methods such as evolutionary algorithms or genetic algorithms must be used. Objective evaluation may involve computationally expensive methods such as CFD or FEA, and the feasible solution space may contain multiple concepts which have different sets of input variables. This means that for the design of custom materials, a multi-objective, multi-concept optimization algorithm with capability of using approximations such as surrogates for the computationally expensive objectives is required.

This project aims to provide a base platform for designing custom lattice structures using multi concept, multi objective optimization algorithm developed by the MDO group. This algorithm uses methods from Infeasibility Driven Evolutionary Algorithm (IDEA) (Hemant K. Singh, 2008) and Reference Vector Guided Evolutionary Algorithm (RVEA) (Ran Cheng, 2016). Specifically, this project aims to provide automatic generation of various lattice structure models, the ability to evaluate mass, surface area, volume, stress and stiffness of the structure. This will then be integrated with a multi-objective, simultaneous multi-concept, optimization algorithm developed by the MDO group to run design optimization for one single objective problem and three bi-objective problems. This will provide a base platform for the computational design of novel/custom materials made from periodic open cell lattice structures. This platform will be open for other evaluations on the lattices to be made such as fluid flow analysis or more complex structural evaluations. These evaluations can then be used to create surrogates and more in-depth optimization for different applications can be made in future work.

II. Lattice Generation

For this project, a rectangular prism block of dimensions L, B and H was used as the basis for the lattice material. The requirement was for space filling polyhedra as in the literature review at Appendix A, able to fill the L x B x H block to be generated in a way that was analysable and 3D printable. This requirement led to the use of CAD software to be used in order to generate the lattice.

The first, ready, existing package found was an open source compilation of MATLAB codes and functions called GEOM3D, authored by David Legland. (Legland, 2016). This package, has functions to create the nodes, edges and faces of space filling polyhedra lattices and to create STL files of these lattices. This is done by creating cylindrical struts centred on edges and spheres at nodes to create a surface file of these lattices. MATLAB has many inbuilt functions for surface generation. This package enabled the ability to create surface files of lattice structures but did not allow for the output of properties: mass, volume, or surface area, or application of materials to the lattice. Investigations were made into converting these STL files to solid files to conduct the evaluations of the lattice structures. This proved not possible as the STL files had internal imperfections which were not apparent from the outside. The STL files generated using GEOM3D are fine for actual manufacture using additive manufacturing but were inadequate for surface to solid conversion.

Other methods of lattice generation in STL format were found, using OpensCAD, FREECAD, and RHINO. These methods could create STL files with no internal imperfections inside of the lattice struts. Meshing and conversion from surface to solid was computationally expensive, relatively inaccurate and was very difficult to automate. This meant that for the purposes of this project they were also discounted. The next step taken was to generate lattice structures straight to solid format. This was conducted using CATIA, the methodology of this process can be seen in Appendix B.

Summary Report 2017, UNSW Canberra at ADFA
Using the CATIA models with their respective design tables, MATLAB code was written. A MATLAB function was created for each lattice type, Cube, Cuboctahedra and Tetrakaidecahedra. These functions take inputs: lattice geometry, unit cell geometry, and lattice material. The functions then convert these inputs to the relevant design table values, and write the values to the design table. The function then creates a COM Server between CATIA and MATLAB, opens the respective cell primitive CATPart file, updates the part to the design inputs from the design table. The function then exports the values for the mass of the lattice block, the surface area and the volume and stores these in arrays in MATLAB. The CATIA COMServer Link was brought to my attention through (Friedrich, 2014). The rest of the commands in the MATLAB Functions were created from MATLAB help or by recording VBA macros on CATIA and converting these VBA functions into working COMServer MATLAB commands. The MATLAB functions created for the Cube lattice, Cuboctahedra lattice and the Tetrakaidecahedra lattice can be found in Appendix C.

III. Evaluation Functions

A. CATIA

The functions for the three, space filling polyhedra (cube, cuboctahedron, and tetrakaidecahedron) take the lattice geometry and generate the lattice as described above. These functions then export the values for the mass of the lattice block, the surface area and the volume and stores these in arrays in MATLAB. The CATIA COMServer Link method was found through (Friedrich, 2014) and was adapted to the purposes of this project. The rest of the commands in the MATLAB Functions were created from MATLAB help or by recording VBA macros on CATIA and converting these VBA functions into working COMServer MATLAB commands. The MATLAB functions created for the Cube lattice, Cuboctahedra lattice and the Tetrakaidecahedra lattice can be found in Appendix C.

B. ABAQUS

The evaluation functions for element stress and node deflection are found through Abaqus using MATLAB, Command line and python scripts. Within the evaluation, nodes and edges are found for the specific solution being evaluated using GEOM3D functions (Legland, 2016) which have been modified to supply the exact node lists and edge lists required. Another function then takes the node and edge lists and other input information from the user such as material, lattice strut diameter, loads applied to the lattice, and outputs required from the Abaqus analysis. This function writes the INP file for the Abaqus analysis in the correct format automatically for any cube lattice that is generated through the optimisation algorithm. This function can be seen at Appendix D. Using the INP file, the Abaqus analysis is completed in the windows shell using commands from MATLAB. The subsequent results files are read and converted to MATLAB arrays using Abaqus2Matlab (G. Papazafeiropoulos, 2017) functions that have been modified for this project. Using this set of functions an evaluation may be conducted on element stresses, and node displacements or any other output generated by Abaqus using standard beam analysis with INP files. This is done with the inputs of nodes, edges, material, strut radius, and lattice loads.

C. Post Processing

After the optimisation run has completed, the output is a large array of data containing the solution number, concept number, variable values, objective value, constraint values, and constraint violations. The number of rows and columns of this array vary depending on the number of concepts, variables, objectives, constraints and the max population assigned to the optimisation run. To evaluate this data, post processing functions were required to sort and find non-dominated solutions, plot Pareto fronts, find hyper volumes of Pareto fronts and other statistical results. These functions were created in collaboration with Multidisciplinary Design Optimization Group members. Examples of these functions may be found in Appendix E.

IV. Problem formulation

For this project an optimisation of the lattice structure using the lattice generation functions, and evaluation functions described above was conducted. The optimisation was conducted using MDO groups multi objective, multi concept optimisation algorithm. To test the functions with this optimisation algorithm four problems were run. The first problem is a single objective optimisation for minimum weight with a single cube concept. The second optimisation was a bi-objective optimisation problem for minimum weight and maximum surface area for a single cube concept. The third optimisation was a bi-objective problem for minimum weight and maximum surface area for three concepts: cube, cuboctahedron and tetrakaidecahedron. The fourth and final optimisation was the same as problem two with extra constraints on maximum stress allowable in the struts and maximum deflection allowable in the nodes of the lattice. These four problem formulations are set out in detail below with objectives, constraints, variables, variable ranges, and the number of runs completed for each problem.
Each of the problems had the following global parameters:
1. \( L = 800 \text{ mm} \)
2. \( B = 800 \text{ mm} \)
3. \( H = 200 \text{ mm} \)
4. Material Variable options are Steel and Aluminium.

A. **Cube Single Objective**

Objectives: \( F_1 = \text{Max Surface Area (this was conducted by minimising inverse area)} \ (1/\text{mm}^2) \)

Runs: 21

Max population: 300

Table 1: Cube Concept Formulation

<table>
<thead>
<tr>
<th>Variables</th>
<th>Concept 1: Cube</th>
<th>Variable type</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>( a )</td>
<td>Continuous</td>
</tr>
<tr>
<td>X1 Lower Bound</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>X1 Upper bound</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>X2</td>
<td>( t )</td>
<td>Continuous</td>
</tr>
<tr>
<td>X2 Lower Bound</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>X2 Upper Bound</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>X3</td>
<td>Material</td>
<td>Categorical</td>
</tr>
</tbody>
</table>

Constraints within the optimisation algorithm are setup so that a negative value indicates a feasible solution and a positive value indicates an infeasible solution. Equation 1 below shows the geometry constraint for the cube concept and ensures that cell struts do not overlap.

\[
G_1 = t - \frac{a}{2}
\] (1)

B. **Cube Bi Objective**

Within this optimisation problem the cube concept is used. The same variables, bounds and constraints used in the single objective problem are used for the Bi-Objective problem.

Objectives: \( F_1 = \text{Max Surface Area (this was conducted by minimising inverse area)} \ (1/\text{mm}^2) \)
\( F_2 = \text{Minimum weight (kg)} \)

Runs: 21

Max population: 300

C. **Multi Concept Bi Objective**

Within this optimisation problem the cube, cuboctahedron and tetrakaidecahedron concepts are used. The same variables, bounds and constraints in the cube single objective problem are used for this bi-objective problem with the addition of the variables, bounds and constraints for the other two concepts shown below.

Objectives: \( F_1 = \text{Max Surface Area (this was conducted by minimising inverse area)} \ (1/\text{mm}^2) \)
\( F_2 = \text{Minimum weight (kg)} \)

Runs: 11

Max population: 300

Table 2: Cuboctahedron Concept Formulation

<table>
<thead>
<tr>
<th>Variables</th>
<th>Concept 2: Cuboctahedron</th>
<th>Variable type</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>( a )</td>
<td>Continuous</td>
</tr>
<tr>
<td>X1 Lower Bound</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>X1 Upper bound</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>X2</td>
<td>( t )</td>
<td>Continuous</td>
</tr>
<tr>
<td>X2 Lower Bound</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>X2 Upper Bound</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>X3</td>
<td>Material</td>
<td>Categorical</td>
</tr>
</tbody>
</table>
The below equation describes the geometry constraint for the cuboctahedron concept. Equation 2 was created to ensure that the radius of cell struts do not become too large that the struts overlap.

\[ G1 = t - \frac{\sqrt{3}}{6} \times \frac{a}{\sqrt{2}} \]  

(2)

**Table 3: Tetrakaidecahedron Concept Formulation**

<table>
<thead>
<tr>
<th>Variables</th>
<th>Concept 3: Tetrakaidecahedron</th>
<th>Variable type</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>s</td>
<td>Continuous</td>
</tr>
<tr>
<td>X1 Lower Bound</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>X1 Upper bound</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>X2</td>
<td>b</td>
<td>Continuous</td>
</tr>
<tr>
<td>X2 Lower Bound</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>X2 Upper Bound</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>X3</td>
<td>0</td>
<td>Continuous</td>
</tr>
<tr>
<td>X3 Lower Bound</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>X3 Upper bound</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>X4</td>
<td>t</td>
<td>Continuous</td>
</tr>
<tr>
<td>X4 Lower Bound</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>X4 Upper bound</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>X5</td>
<td>Material</td>
<td>Categorical</td>
</tr>
</tbody>
</table>

The equations below 3 to 6 describe the geometry constraints for concept 3. Equations 3 and 4 come directly from (Sullivan et al., 2008). These ensure that the tetrakaidecahedron cell does not become elongated in the x and y direction but only in the z direction. Equation 5 and 6 were created to ensure that the radius of cell struts do not become too large that the struts overlap.

\[ G1 = \frac{b}{s} - 2 \times \sqrt{2} \]  

(3)

\[ G2 = -\varnothing + \sin^{-1}\left(\frac{\sqrt{2} \times b}{5 \times s} + \frac{\sqrt{2}}{10} \times \sqrt{10 - \left(\frac{b^2}{s^2}\right)}\right) \]  

(4)

\[ G3 = t - \frac{b}{2} \]  

(5)

\[ G4 = t - \frac{s \times \sin(2\varnothing)}{2} \]  

(6)

**D. Cube Bi Objective (with Abaqus)**

This problem formulation contains the same concept variables and bounds as the cube bi-objective problem. The addition of Abaqus analysis meant that a force was applied to top nodes of the lattice block of 400 N.

Objectives:  
F1 = Max Surface Area (this was conducted by minimising inverse area) \((1/\text{mm}^2)\)  
F2 = Minimum weight (kg)

Runs:  
1

Max population: 150

Constraints for this problem contained the same geometry constraints for the cube concept as in Equation 1. There were two other constraints added that constrained stress and deflection for the lattice block. These are shown in Equation 7 and Equation 8. These limits were chosen from a single feasible solution evaluation.

\[ G2 = \text{Displacement} - 0.25 \]  

(7)

\[ G3 = \text{Von Mises} - 100 \]  

(8)
V. Optimization Results

A. Cube Single Objective

For this optimisation, a single concept (cube cell) was optimised for maximum surface area with constraints and variable ranges as described in the problem formulation. Table 4 below shows the maximum, mean, median, minimum and standard deviation of the inverse area for the 21 runs of the optimisation algorithm. The cell size, and strut size of the lattice (X1, and X2) are also shown for the non-dominated solution of the median run. As a single objective optimisation there is only one non-dominated solution for this problem therefore the Pareto front is trivial. The optimal lattice structure for the median run is shown below in Figure 1.

Table 4: Cube Single Objective Results

<table>
<thead>
<tr>
<th>Maximum inverse area</th>
<th>Mean Inverse area</th>
<th>Median Inverse area</th>
<th>Minimum Inverse area</th>
<th>Standard Deviation Inverse area</th>
<th>X1 (median)</th>
<th>X2 (median)</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.7458×10^-7</td>
<td>1.6271×10^-7</td>
<td>1.6168×10^-7</td>
<td>1.6105×10^-7</td>
<td>2.9863×10^-9</td>
<td>53.168</td>
<td>9.998</td>
<td>1.6168×10^-7</td>
</tr>
</tbody>
</table>

Figure 1: Maximum Surface Area Cube Lattice

B. Cube Bi-Objective

For this optimisation, a single concept (cube cell) was optimised for maximum surface area and minimum weight with constraints and variable ranges as described in the problem formulation. Table 5 below shows the maximum, mean, median, minimum and standard deviation of the hyper volume for the 21 runs of the optimisation algorithm. The cell size, strut size, material and objective values of the lattice (X1, X2, X3, F1 and F2) are also shown for three solutions from the median run in Table 6. These three solutions were chosen as the outer solutions of the Pareto front and one knee solution, the first being minimum area and minimum weight, solution two being maximum mass and maximum surface area and solution three being the knee solution as a trade-off of the two objectives. The Pareto front of the median run can be seen in Figure 2. The three lattice structures for the median run are shown below in Figure 3, Figure 4, and Figure 5.

Table 5: Cube Bi-Objective Hyper Volume Results

<table>
<thead>
<tr>
<th>Maximum HV</th>
<th>Mean HV</th>
<th>Median HV</th>
<th>Minimum HV</th>
<th>Standard Deviation HV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1596</td>
<td>1.1495</td>
<td>1.1508</td>
<td>1.1344</td>
<td>0.00787</td>
</tr>
</tbody>
</table>

Table 6: Cube Bi-Objective Selected Solutions Results

<table>
<thead>
<tr>
<th>Solution:</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum area and minimum mass</td>
<td>73.7583</td>
<td>1.04432</td>
<td>aluminium</td>
<td>0.6081</td>
<td>2.3442e-06</td>
</tr>
<tr>
<td>Maximum mass and maximum area</td>
<td>51.3668</td>
<td>8.9811</td>
<td>steel</td>
<td>221.9793</td>
<td>1.8057e-07</td>
</tr>
<tr>
<td>Knee Solution</td>
<td>57.9221</td>
<td>3.6865</td>
<td>aluminium</td>
<td>12.7382</td>
<td>4.0739e-07</td>
</tr>
</tbody>
</table>
C. Three Concept Bi-Objective

For this optimisation, three concepts (cube, cuboctahedron, and tetrakaidecahedron) were optimised for maximum surface area and minimum weight with constraints and variable ranges as described in the problem formulation. Table 7 below shows the maximum, mean, median, minimum and standard deviation of the hyper volume for the 11 runs of the optimisation algorithm. The cell size, strut size, material and objective values of the lattice (X1, X2, X3, F1 and F2) are also shown for four solutions from the median run in Table 8. These four solutions were chosen as the outer solutions of the Pareto front for each of the concepts present. The Pareto front
of the median run can be seen in Figure 6. The four lattice structures for the median run are shown below in Figure 7, Figure 8, Figure 9 and Figure 10.

Table 7: Multi Concept Bi-Objective Hyper Volume Results

<table>
<thead>
<tr>
<th></th>
<th>Maximum HV</th>
<th>Mean HV</th>
<th>Median HV</th>
<th>Minimum HV</th>
<th>Standard Deviation HV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2039</td>
<td>1.2011</td>
<td>1.2005</td>
<td>1.1981</td>
<td>0.0021156</td>
<td></td>
</tr>
</tbody>
</table>

Table 8: Multi Concept Bi-Objective Selected Solutions Results

<table>
<thead>
<tr>
<th>Solution:</th>
<th>Concept</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max mass</td>
<td>Cube</td>
<td>73.7584</td>
<td>4.0373</td>
<td>Aluminium</td>
<td>17.9952</td>
<td>3.1868e-07</td>
</tr>
<tr>
<td>Min mass</td>
<td>Cube</td>
<td>90.2118</td>
<td>1.0311</td>
<td>Aluminium</td>
<td>0.4791</td>
<td>2.9335e-06</td>
</tr>
<tr>
<td>Max mass</td>
<td>Cuboctahedron</td>
<td>52.8698</td>
<td>7.8043</td>
<td>Steel</td>
<td>358.4177</td>
<td>1.1173e-07</td>
</tr>
<tr>
<td>Min mass</td>
<td>Cuboctahedron</td>
<td>85.6322</td>
<td>2.8129</td>
<td>Aluminium</td>
<td>9.1659</td>
<td>4.3171e-07</td>
</tr>
</tbody>
</table>

Figure 6: Median Hypervolume Run Pareto Front for Multi Concept: Cube (Concept 1), Cuboctahedron (Concept 2), Tetrakaidecahedron (Concept 3)

Figure 7: Minimum Mass Cube Lattice

Figure 8: Maximum Mass Cube Lattice
D. Cube Bi-Objective (with Abaqus)

For this optimisation, a single concept (cube cell) was optimised for maximum surface area and minimum weight with constraints and variable ranges as described in the problem formulation. The Pareto front of the single run can be seen in Figure 11. The Pareto front for this optimisation achieved a hyper volume of 1.1743. The cell size, strut size, material and objective values of the lattice (X1, X2, X3, F1 and F2) are also shown for three solutions in Table 9. These three solutions were chosen as the outer solutions of the Pareto front and one knee solution, the first being minimum area and minimum weight, solution two being maximum mass and maximum surface area and solution three being the knee solution as a trade-off of the two objectives.
Table 9: Cube Bi-Objective (With Abaqus) Selected Solutions Results

<table>
<thead>
<tr>
<th>Solution: Minimum area and minimum mass</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>95.6808</td>
<td>1.2218</td>
<td>aluminium</td>
<td>0.7129</td>
<td>2.3358x-06</td>
</tr>
<tr>
<td>Maximum mass and maximum area</td>
<td>52.9162</td>
<td>9.9324</td>
<td>steel</td>
<td>274.7537</td>
<td>1.6339e-07</td>
</tr>
<tr>
<td>Knee Solution</td>
<td>55.8504</td>
<td>3.7777</td>
<td>aluminium</td>
<td>14.7934</td>
<td>3.6055e-07</td>
</tr>
</tbody>
</table>

VI. Results Discussion

Within this project the way that the lattice is generated is by creating a single primitive cell of the concept and then this is patterned in the x, y, and z directions to fill the lattice block of L, B, and H dimensions. When the single cell is created it will have its own length width and height, the amount of times that the cell is replicated in each direction is the number that will fit inside the block dimensions. This space filling technique means that the lattice block created generally will not fill the full dimensions but will fill it as much as possible while only generating whole cells and not partial lattice cell geometry. This can be seen in the cube single objective problem where surface area is to be maximised. From the problem formulation the ideal cell geometry would be a = 50 and t = 10 to generate the most and largest lattice struts. As can be seen in the results the optimal solution was a = 53.168mm and t = 9.998mm. It can be seen from this that the optimisation algorithm and the way the lattice block is generated that the upper and lower limits of the variables are not able to be reached. As the lattice block is 800x800x200, the most cells in the x and y direction would be 16 and 4 in the z direction. As 50mm cell size is not able to be achieved a maximum of 15x15x3 cells can be generated. Therefore 53.168mm is very close to the largest possible size before the lattice generation drops to 14x14x3. This means that the integration of the lattice generation and evaluation into the optimisation algorithm is working as expected and outputting reasonable results for a single concept and single objective.

For the cube bi-objective problem the variable of material starts to play a part in affecting the solution’s weight objective. Obviously aluminium is lighter that steel and this can be seen on the Pareto front where solutions on the right outer extreme (maximum mass) are made of steel and the other end are all made of aluminium. As mass is to be minimised it is expected that the knee solution is made of aluminium and this is the case as shown in the results. The objective values of the cube bi-objective problem correlate with those of the cube single objective problem where the maximum surface area achieve in the single objective problem was not exceeded by the cube bi-objective problem. This is due to the mass objective making those solutions dominated in the sorting process.

For the multi concept bi-objective problem the most obvious observation to be made is that there are only two concepts present on the Pareto front of the median run. The cube solutions (concept 1) are also in the left-hand side while the cuboctahedron solutions (concept 2) are on the right-hand side. This is due to the variable ranges of each of the concepts. The variable ranges shown for the tetrakaidecahedron in the problem formulation only allow for cells of this type to be generated that are at minimum: 102.65 mm high and 82.42 mm wide and maximum of: 346.41 high and 227.279 mm wide. For this reason, the tetrakaidecahedron lattice block does not create enough cells within the L, B, H dimensions to maximise surface area and the cube and cuboctahedron solutions become dominant. Within the Pareto front for the multi concept problem the cuboctahedron seems to be the heavier of the two remaining concepts. There are feasible cube lattice blocks that can achieve the same mass as the cuboctahedron but these are dominated due to the increase in surface area of the cuboctahedron solutions. As can be seen in Figure 6 there is some small overlap in the cube and cuboctahedron solutions. Another observation to be made about the multi concept optimisation is the difference in hyper volume values compared to that of the cube bi objective problem. As can be seen from the results the mean hypervolume of the multi concept is 0.05169 larger than the cube bi objective problem. In this context the larger hyper volume means that the non-dominated solution space is larger and more converged towards the two objectives. This can be explained by the addition of the cuboctahedron concept which increased the mass and surface area ranges.

For the cube bi-objective problem with Abaqus analysis, the results can be seen within Figure 11 to be very similar to the bi objective problem without the Abaqus analysis. Upon review of the constraint violation data there were 11 cases of stress and deflection violation out of the 150 solutions. The optimisation algorithm is set up such that when a constraint is violated, the objective values are set to maximum values. This is to ensure that computational time is not wasted on infeasible solutions. During post processing of data and results these solutions were evaluated to check their dominance, and some were found to be on the Pareto front when the constraints were not considered. To further compare the cube bi-objective problems, the stress and deflection constraints were applied to the median hypervolume run of the cube bi-objective problem that was initially run with only geometry.
constraints. From the Pareto front of this run, there were 10 out of 79 solutions that violated these constraints. These solutions were all low mass and low surface area solutions with strut radius less than 1.3 mm. Other solutions within this non-dominated list had constraint violation values that were very small showing that the Pareto front was lies on the constraint boundaries.

There are no effects from cube geometry constraint at equation 1 due to variable ranges discounting the constraint. The geometry constraint on the cuboctahedron at equation 2, limited the solution space. The maximum strut radius when the cuboctahedron cell is at the lower limit of \( a = 40 \text{ mm} \) is \( t = 8.165 \text{ mm} \). When the cuboctahedron cell is at maximum cell size of \( a = 90 \text{ mm} \) the maximum strut size is \( t = 18.37 \text{ mm} \). This constraint had violations within the constraint violation data due to the variable ranges. The geometry constraints on the tetrakaidecahedron angle \( \theta \) had violations which made infeasible solutions within this concept. As this concept was not within the non-dominated solution set this constraint violation did not affect the final results.

From the results the lattice generation, and evaluation system has been integrated into the optimisation algorithm and calculated optimal solutions to the four problems that make physical sense and are reasonable.

VII. Recommendations for Future Work

From this project there is clear potential for further work and improvements to be made for this engineering tool. The main potential is for generalised modelling and optimisation tool for 3D printable lattice structures that may be optimised for multiple objectives. The current baseline which includes the lattice generation functions and objective evaluation functions described in this report may be a very useful tool in novel material research. The following are recommendations for improvements and additions that were outside of the scope of this project due to time and resource constraints.

The first recommended addition to this project is the inclusion of the other two concepts: cuboctahedron and the tetrakaidecahedron within the Abaqus beam analysis. This would require the node and edge lists to be generated like that of the cube concept with GEOM3D functions. The edges need to be sorted into groups depending on the orientation of the edge. The INP formatting function then need to be adjusted to include these concepts. These concepts could then be used within the multi concept, multi objective optimisation with Abaqus evaluation of stress and deflection.

The second recommended improvement for this tool is for more lattice concepts to be included. These lattice concepts would be the other space filling polyhedra described in the literature review such as the Rhombic dodecahedron or the tetrahedron. This would provide more options for the optimisation algorithm and might provide improved solutions depending on which objectives are being minimised/maximised.

Another recommendation is to increase the range and generalisation of the tool by generating lattice cells with other strut section types. Within this project the lattice struts had a full circular section so that the struts were solid cylinders. It is possible within the layout and setup of the lattice generation to incorporate other strut sections. Within the Abaqus Beam analysis there are default section types that can be analysed: box, circular, hexagonal, rectangular, I-section, L-section, pipe, and trapezoidal. There is also an option for an arbitrary section that may be user defined. These Abaqus section options have set variable inputs to define the section which would then become variables for the optimisation algorithm. (Dassault_Systems, 2014) This would require more intensive setup time for the CATIA lattice generation and evaluation. A similar method as described in Appendix B may be used.

Within this project the way that the lattice is generated means that the lattice block of \( L, B \) and \( H \) is filled as much as possible given the size of the individual cell. Another way this could be done is to have the number of cell replications in the \( x, y \) and \( z \) directions as variables. This method could provide a method of generating smaller lattice blocks with the dimensions \( L, B \) and \( H \) being the upper limit. In cases where the function of the material is more important than the dimensions this could provide a much larger span of solutions and may find more optimal results.

With the above recommendations this tool could be used for a very wide range of applications and is very easily modified to incorporate new evaluations, objective functions and constraint functions. It has the potential to be used for material design and premanufacture evaluation of 3D printable lattice materials.
References


Summary Report 2017, UNSW Canberra at ADFA