

Automated Post-Processing for Sheet Metal Component Manufacturing

Sehmi, M. S., Christensen, J., Bastien, C., Wilson, A. & Kanarachos, S.

Author post-print (accepted) deposited by Coventry University's Repository

Original citation & hyperlink:

Sehmi, MS, Christensen, J, Bastien, C, Wilson, A & Kanarachos, S 2020, 'Automated Post-Processing for Sheet Metal Component Manufacturing', *Advances in Engineering Software*, vol. 143, 102794.

<https://dx.doi.org/10.1016/j.advengsoft.2020.102794>

DOI 10.1016/j.advengsoft.2020.102794

ISSN 0965-9978

Publisher: Elsevier

**NOTICE: this is the author's version of a work that was accepted for publication in *Advances in Engineering Software*. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in *Advances in Engineering Software*, 143, (2020)
DOI: 10.1016/j.advengsoft.2020.102794**

© 2020, Elsevier. Licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International <http://creativecommons.org/licenses/by-nc-nd/4.0/>

Copyright © and Moral Rights are retained by the author(s) and/ or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This item cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder(s). The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holders.

This document is the author's post-print version, incorporating any revisions agreed during the peer-review process. Some differences between the published version and this version may remain and you are advised to consult the published version if you wish to cite from it.

Automated Post-Processing for Sheet Metal Component Manufacturing

Maninder Sehmi ^{a,*}, Jesper Christensen ^a, Christophe Bastien ^a, Alexis Wilson ^a, Stratis Kanarachos ^a

^a Department of Engineering, Environment and Computing, Coventry University, Coventry, West Midlands, UK

Abstract

Topology optimisation is an increasingly important process used in a variety of industries to improve the designs of manufacturable products. The higher reliance of optimisation software, used for instance in the automotive industry, highlights its importance for designing more efficient and refined mass-produced components. Post-processing of topology optimisation results (e.g. from variable density to manufacturable structures) does however remain a heavily heuristic process where the end-results (and consequently the “efficiency” of the optimised product) can vary significantly as a function of the individual designer/engineer. This “variation” coupled with the often-significant time associated with post-processing makes the use of topology optimisation prohibitive in certain instances.

In this paper, a systematic and repeatable three-step approach to automated post-processing of topology optimisation results for sheet metal manufacturing of automotive components will be introduced. The method, which has been implemented into a software tool, is mesh independent and can handle topology optimisation results in binary as well as variable density formats. The software contains three main steps; namely geometry refinement, re-analysis and manufacturability check. The methodology and software utilise a stencil method, for which the principles are described here. The main objective from this is to generate repeatable refined interpretations of optimisation results. In addition to presenting the actual methodology and software, this paper also investigates different parameter variations; such as geometry update sequence, search radii, stencil shape and type and their influence on the generated post-processed result. Definition of algorithm parameters is provided, together with suggested user-defined settings to enable the derivation of consistent refinements of the topology results.

Keywords

Post-Processing, Topology Optimisation, Semi-autonomous programming, Model refinement, Search stencil, Sheet metal Manufacturing

* Maninder Sehmi (Corresponding Author)
Email Address: sehmim@uni.coventry.ac.uk

1 Introduction

The increasing reliance on Topology Optimisation (TO) for the design of engineered components has allowed for quick generation and manufacturing of complex designs that are uniquely tailored to fulfil a desired function. An example of this is the development of sheet metal components which form an automotive Body-in-White (BIW). Continuous re-design of these individual panels using computational optimisation techniques has allowed for quicker generation of safer, cheaper and more efficient BIW designs, as opposed to purely manual and heuristic analyses. Sheet metal components are commonly used in vehicle structures as they can provide significant structural support under high loads (Machine MFG 2019) and are additionally lightweight, thus improving efficiency.

Despite the improvements in computing power over recent years, the generated topology design must be interpreted by an engineer in order to determine how the design's features can be manufactured. This is achieved through a follow-on procedure known as Post-Processing (PP). PP is a mostly heuristic process and, from user experience, can account for more than 50% of the combined analysis and optimisation time used for a given component. Unlike topology optimisation solvers no standalone automated methodology exists for PP (Sehmi et al. 2018). Subject to the individual who completes the PP, the lack of methodology can cause the final manufacturable solution to vary significantly and can also make the overall process inefficient, ultimately leading to sub-optimal components and systems.

In an attempt to overcome these issues, existing TO solvers have been modified to include model refinement within the topology solver itself such as in the works of (Liu and Ma 2015) and (Yi and Kim 2016). However, these solvers do not separate the TO from the PP; consequently, a variety of important detailed manufacturing features are not considered within these refinements. The lack of consideration towards manufacturing features, e.g. bend angles or stamping tolerances, can prevent the optimised solution from fully representing a suitable design solution.

This paper presents a three-step method to automate PP for TO results, specifically for sheet metal components. The methodology presented is designed to be repeatable and provide a systematic approach to refining TO results. The process, referenced in this paper as Automated Post-Processing (APP), is designed to be mesh-independent and is capable of refining binary and variable density models alike. The APP methodology has been implemented as a computational code in Python programming language, with three main steps: element creation, element deletion and other combinations. The computational methodology includes a mesh-independent stencil method capable of identifying and refining various topological features such as voids and disconnected or "floating" elements. The APP methodology is critically evaluated and compared to competing processes, including potential and existing mesh-dependent refinement methods as well as manual procedures. A series of test cases will demonstrate the versatility of the APP, with tests including the consideration of search radii, stencil shape and type as well as the detection methodology developed for manipulating the recorded data. From this, a critical review of the APP method and its current capabilities will be created, with consideration of further extensions outlined.

The remaining sections of this paper include an overview and justification of the post-processing problem in Section 2 – *The Challenges of Post-Processing Topology Optimisation*, followed by a *Literature Review* in Section 3 highlighting existing methodologies and inspirations for the APP. Section 4 - *Methodology of Automated Post-Processor* overviews how the PP process can be automated, outlining the general processes of the proposed APP method. In Section 5 – *Test Case – Refining Various Topology Features* the capabilities of the APP are tested. Specific variables within the APP will be explored and a critical review of the result quality for selected TO designs will be documented. Finally, Section 6 - *Conclusions and Further Development* showcases potential improvements to be made for the APP.

2 The Challenges of Post-Processing Topology Optimisation

Structural optimisation of a component can be performed using a variety of mathematical methods often starting with TO; a process that finds the optimal distribution of material within a given design volume, subject to an objective, boundary conditions and possibly optimisation constraints. There are many different methods and approaches to TO and equally many ways in which to categorise the various methods. The format of the results generated from a topology solver can differ significantly from software to software. Common variations include the creation of a binary or variable density solution topology (Figure 1), which require different PP approaches in order to create a refined manufacturable solution. Further consideration may need to be taken for other optimisation features such as non-linearity, non-isotropic materials as well as any continuing developments to TO solvers in the future. From a practical viewpoint the actual TO solver used is irrelevant in the context of the APP method of this paper; only the format of the results is important.

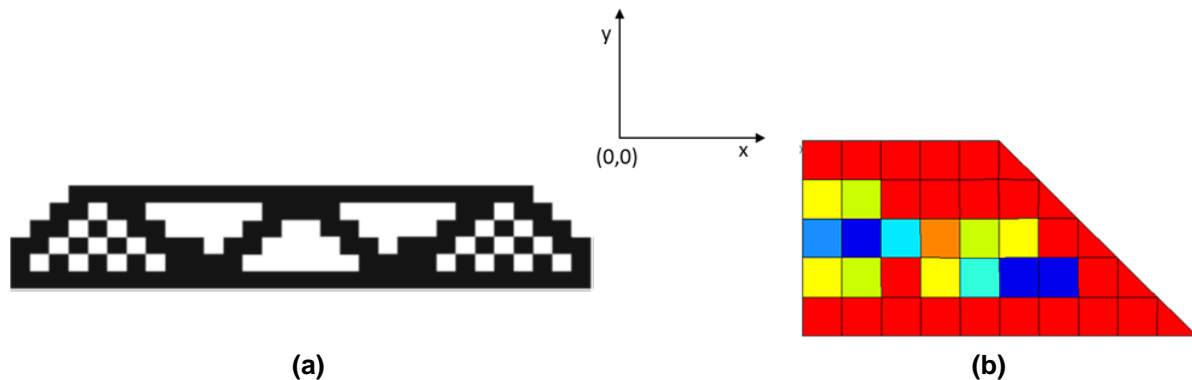


Figure 1 – (a) Binary topology optimisation solution (adapted from Designer.mech.yzu.edu.tw 2017) vs. **(b)** VDM topology optimisation solution.

TO results like those illustrated in Figure 1 are typically obtained from Finite Element (FE) based solvers such as the Bi-directional Evolutionary Structural Optimisation (BESO) or the Variable Density Method (VDM) combined with the Solid Isotropic Material with Penalisation (SIMP) interpolation scheme. These are often available in commercial Finite Element (FE) software; e.g. VDM-SIMP is available in Altair Optistruct.

In general, there are two different strategies for processing of (topology) optimisation, namely to influence the results “within” the solver or refinement of results post solving. As an example of the former, optimisation constraints could be added by defining minimum and maximum member sizes etc. as e.g. done in (Norato et al. 2016). These methods often restrict the actual optimisation which may lead to undesirable reductions in the solution space. In fact, TO is most often employed in order to maximise the performance of a given part, consequently the solution space should not be unnecessarily constrained. Furthermore, the performance of computationally efficient optimisation solvers may dramatically decrease if finer geometry details have to be considered during the optimisation, thereby retracting from the general purpose of TO. Traditionally, TO methods determine overall load path location and do not simultaneously optimise/analyse “finer” geometry details such as stress concentration factors around voids. If a post-processor considering sheet metal manufacturing was to be integrated directly into a specific TO solver it would be necessary to consider the finer details at that

stage, most probably leading to reduced optimisation efficiency and unnecessary restriction of the solution space. Furthermore, it is unlikely that the overall optimal topology will change significantly as a function of considering the finer geometric details/manufacturing constraints such as bend radii. Finally, with the constant evolvement of new TO algorithms and solvers, e.g. considering dynamic loading, non-linear and non-isotropic material behaviours, it is convenient to separate the TO and PP steps.

The second strategy for PP is to conduct it after the TO is completed. This approach enables the development of a universal methodology/tool generally compatible with any TO solver including hybrid and refined versions of any given methodology. This approach allows the full solution space to be utilised; generally maximising the benefits of TO but may create more “challenging” results from a PP and manufacturability viewpoint. Therefore, the APP method presented in this paper will be employed after TO has been completed. In general, PP of TO results should consist of three main steps as detailed in Figure 2:

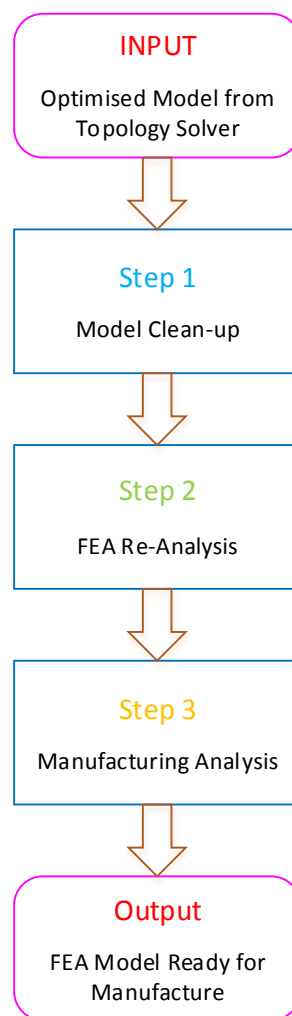


Figure 2 –Steps performed by proposed APP

As an example of the above 3 steps, consider the topology optimisation of a cantilever beam subject to a maximum displacement constraint with an objective of minimising mass. In step

1, topological features such as “holes” must be identified, Figure 3(a), and refined for manufacturability, Figure 3 (b).

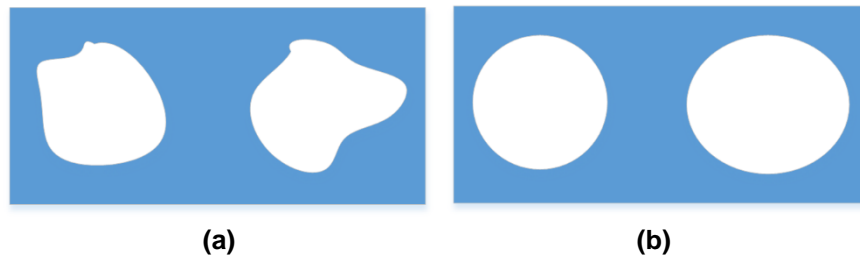


Figure 3 – (a) TO results (b) Refined topology for manufacturing

In step 2, Figure 2, FE analysis of the geometry in Figure 3(b) is completed to ensure the maximum displacement constraint is not violated. In step 3 the manufacturability of the design in Figure 3(b) is checked e.g. via a FE-based stamping or punching analysis. Steps 2 and 3 may be interchanged and may include new or additional constraints (relative to the TO) such as maximum stress or plasticity limits from the punching process. Steps 2 and 3 could be completed utilising pre-existing tools such as batch-meshing and even shape or size optimisation, perceivably making these steps “less challenging” compared to step 1. With that in mind, focus will now turn to step 1, Figure 2.

As previously stated the overall aim of step 1 is to provide a topology suitable for performance/structural and manufacturing analysis in steps 2 and 3 respectively. This is achieved by firstly identifying and secondly refining the topological features, as illustrated in Figure 4.

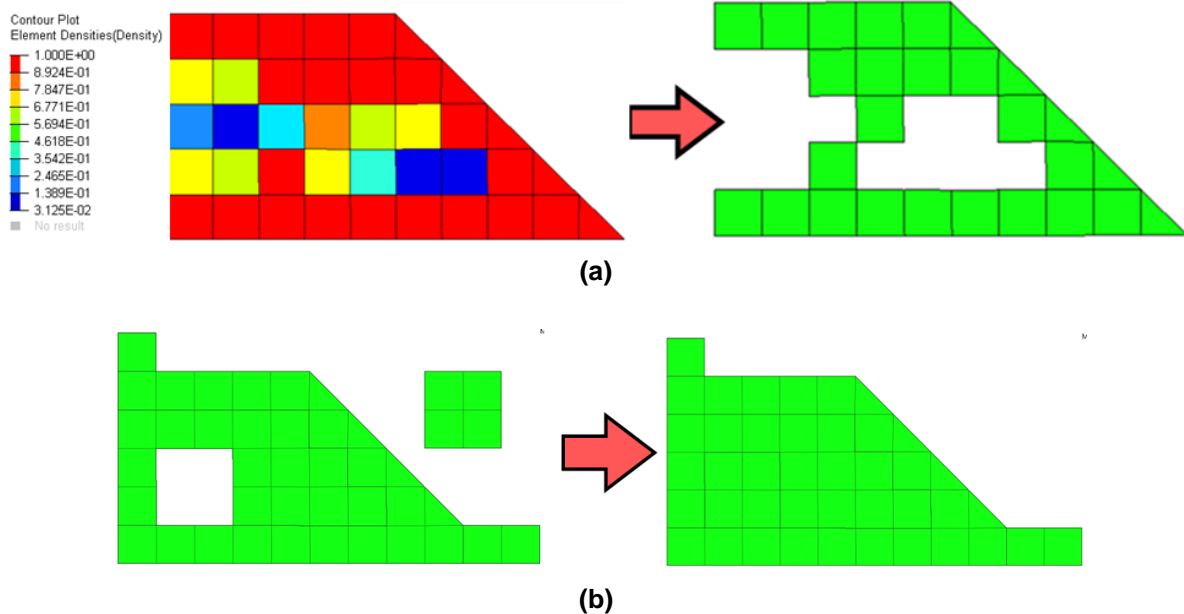


Figure 4 – (a) conversion of a VDM solution to a binary solution and (b) removal of floating elements and filling “structurally insignificant” voids

These features may include, but are not limited to, the removal of voids or “detached” components which are deemed insignificant in either structural or design importance and the conversion of Variable Density Method (VDM) models to binary components. The definition of

an “insignificant feature” will vary between applications and should ideally be fully defined through a set of explicit parameters. The parameter values may be defined through a number of ways including manufacturing standards, tolerances, company guidelines or even personal experience. Allowing the user to define these parameters introduces an appropriate level of adaptivity and flexibility into the PP, but the fixed implementation of those parameters into the APP ensures a consistent method for the removal of “insignificant features”, consequently providing repeatable PP results. Several methodologies can be used to obtain the desired balance between adaptivity, flexibility, consistency and repeatability of the APP, including geometry tracking and image recognition techniques as shall be subsequently analysed, evaluated and compared. Before this is completed it is however important to consider the desired level of automation of the PP, primarily focusing on step 1 (Figure 2).

2.1 Level of Automation in Post-Processing

Aside from the use of integrated PP-TO solvers, the PP refinement of TO designs is almost entirely manual. This procedure typically involves the use of analysis (FE) or Computer Aided Design (CAD) software to update topological features, ensuring a mature design suitable for manufacturing. Generally, for a discretised (FE) model, the individual elements will be repositioned, deleted or added in certain sections of the optimised topology through manual model editing. This process of editing elements would typically follow a series of design guidelines for the component, whilst also maintaining the topology’s structural performance. This procedure is therefore very time-consuming and may pose issues with consistency (due to user interpretation), quality and (optimised) structural performance of the refined solution due to variations of potential prerequisites.

As an example consider the task of determining whether to delete or retain each of the two holes in Figure 3(a). Firstly, what “shapes” are the holes; e.g. square, elliptical or circular? Assuming the two holes are defined as being circular the next step is to determine whether or not they are above or below a threshold (circular) radius value; but what is the radius of the two “circular” holes in Figure 3(a)? Based on this example it is clear to see how subjective (and unrepeatable) PP can be, also that no two holes or features in TO will be identical, for example due to factors such as mesh size used. Consistency and repeatability could be introduced by defining a fixed set of rules for e.g. “measuring” holes. In the interest of minimising errors and real-world time consumption whilst implementing consistency, flexibility and quality of results, several, if not all, steps involved in the PP process could be automated using computer software. This includes the potential to significantly augment refinement processes by creating interactive environments (i.e. augmented reality (Yew et al. 2014) (Nee et al. 2012)) or integrating these refinement steps into machining tools as interactive systems. Table 1 provides an overview of desirable PP features, where the author ranks the potential of manual, semi-automated and fully-automated methodologies to enable the features either 1 (low), 2 (medium) or 3 (high).

No.	Desired Features	Level of Automation		
		Manual	Semi-Automated	Fully-Automated
		Potential Ease of Implementation		
1	Low complexity for integrating multiple manufacturing methods	3	2	2
2	Low level of user involvement	1	2	3
3	Overall PP time	1	2	3
4	Repeatability of process	1	2	3
5	Consideration of individual TO result	3	3	3
6	Easy result validation (re-analysis) process	1	2	3
Total		10	13	17

Table 1 – Comparison of Post-Processing approaches

Although the individual scores in Table 1 are somewhat subjective it clearly highlights the disadvantages of manual methods particularly in terms of time spent on the process and its lack of structure. The allure of a fully automated process includes minimal user input/staff time, “rapidly” obtaining repeatable and fully manufacturable results. Developing such a process (robust software) would however require substantial resources and would simultaneously require “engineering judgement” of results to be implemented directly and fully into the software. Regardless of whether or not the fully automated process is the end-goal, a semi-automated process would be a sensible starting point. Furthermore, substantial improvements to the PP step including significant time-reductions, repeatability and consistency of results can be obtained via a semi-automated process. Therefore, the APP presented in this paper will take the form of a semi-automated process. The next section is a critical review and evaluation of existing methodologies which could and/or has been used for automation of PP.

3 Literature Review

This section provides an overview of candidate methodologies for identifying and refining topological features specifically for step 1 of the PP methodology presented in Figure 2, Section 2. The discussion is divided into mesh-dependent and mesh-independent methods, with examples from recent publications analysed and criticised accordingly. A majority of the concepts discussed within this section have been discussed in the review paper (Sehmi et al. 2018), with all outlined content relating to current methodologies used in optimisation post-processing.

3.1 Geometry Tracking (Mesh-Dependent Methods)

Most, but not all, TO solvers utilise Finite Element Analysis (FEA) to assess the performance of individual optimisation iterations with the end-result defined as an FE model. It is therefore convenient to use this discretised geometry as the starting point for an automated PP procedure. An example of this can be to record and relate element data (e.g. design variable status') for a discretised meshed component and using information such as associated element, node or edge positions to firstly identify the unique topological features before determining how the model should be refined. This type of process, regardless of the specific features identified, can be labelled a mesh-dependent process as it relies solely on this data and no external (non-mesh) features for the refinement.

An example of mesh-dependent refinement applied to a discretised TO solution file can be found in (Lee et al. 2011). Here, the inclusion of Bézier curves is implemented to generate a die shape for a tube drawing process. This die shape is then used for the generation of a drawn shape represented as an FEA solution, which in turn is used to determine the likelihood of fracture. If the likelihood is high the Bézier curve die shape is updated along with the FEA until a converged, non-fracture solution is obtained. This process can be seen as the closest practical (manufacturable) use of mesh-independent TO refinement in recent publications, with PP integrated into the TO iteration loop.

Another example of mesh-dependent refinement is from (Lin and Chao 2000), which takes a greyscale (VDM) TO solution and translates this to a binary FEA model. A parameterised design is then created using this updated model by incorporating shape optimisation to smooth out edges. This again integrates the TO and PP processes, making the procedure reliant on using a discretised FEA TO model.

From Section 2.1 it is desired that the PP step is separated from a TO solver for the APP. Currently, no suitable standalone mesh-dependent PP process exists for TO refinement, whether a discretised model is used or not (Sehmi et al. 2018). In the following section, attention is turned to mesh-independent processes that do not rely on specific FEA data.

3.2 Image Recognition Techniques and Alternative (Mesh-Independent) Refinement Processes

An emerging branch of optimisation and refinement methods incorporates image recognition techniques. Image recognition is defined as the ability to identify and detect an object in a digital format and is used in features such as security surveillance and factory automation (MathWorks 2019). These methods, when integrated with topology optimisation processes, involve detecting the presence of CAD components and may not necessarily consider or involve the use of a finite element mesh, inciting a level of mesh-independency. Two prominent variations considered in this section include the Level Set Method (LSM) and Isogeometric Analysis (IGA).

Level Set Method (LSM)

Developed by Stanley Osher and James Sethian (Osher and Sethian 1988), LSM was initially intended to be a method of recognising topological features of components. It has more recently been integrated into topology optimisation solvers to create a binary refined solution. As explained by (van Dijk et al. 2013), LSM is able to read models with a variety of geometry mapping methods. These include the traditional FEA discretised material mesh, a grid representation of the design with structural boundaries, and a density-based format (Figure 5). LSM is designed to identify borders of a component in a geometric plane, determining whether solid sections lie within or outside of this region.

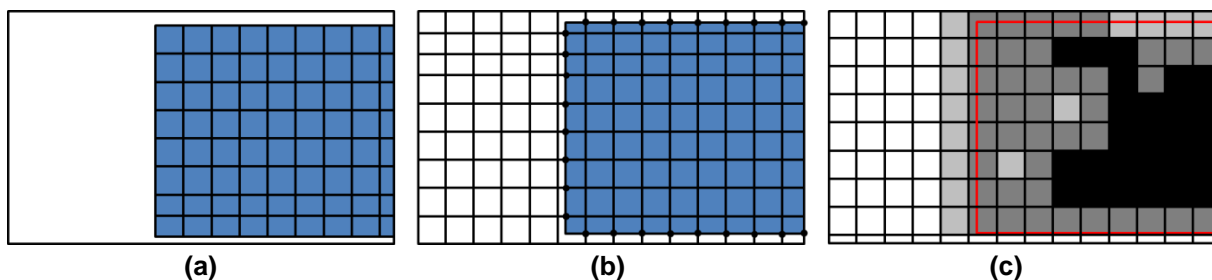


Figure 5 – Geometry types that can be read using LSM: **(a)** Finite Element mesh **(b)** Grid over the design space with geometry boundaries **(c)** Variable Density plot of material (adapted from van Dijk et al. 2013)

The LSM is able to locate and define border locations for input component geometry. This process has been considered for the refinement of TO models. (Challis 2009) shows this in an integrated LSM-TO solver, with the LSM refining the design during the iteration process. This process, as with current mesh-dependent methods (Section 3.1), has not been able to separate the PP process into a standalone program.

A more practical application of an integrated LSM-TO solver is shown by (Kang and Wang 2012). Here, after TO is performed for an individual iteration, an additional LSM step is introduced to locate updated topological boundaries and relocate the position of a proposed feature hole in accordance to the new topology (Figure 6).

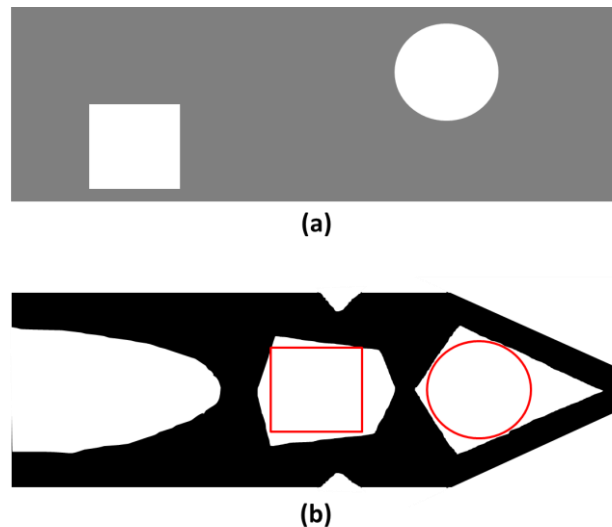


Figure 6 – (a) initial proposed hole locations prior to TO and LSM (b) new proposed locations of feature holes based on integrated TO and LSM run (adapted from (Kang and Wang 2012))

Isogeometric Analysis (IGA)

A more recent alternative methodology to that of traditional FEA is IGA. This process disregards the use of a discrete mesh and instead calculates material stresses using a continuous CAD geometry. This geometry is typically generated using Non-Uniform Rational Basis Splines (NURBS) or alternatively Bézier curves. Using this continuous geometry, the IGA process will place a “projected” grid over the component in a method of discretising the model without needing to create a mesh. The process will then perform an FE analysis, similar to those performed for discretised models (Lovadina et al. 2014).

After performing IGA, the NURBS-generated solution can be optimised using a process called Trimmed Surface Analysis (TSA) (Kang and Youn 2016). TSA uses the stress data provided by the IGA and uses this to “cut” the geometry such that a reduced model is obtained which contains the higher, more structurally important stresses, generating a 2D CAD solution. It is important to emphasise that IGA is not a post-processing refinement method but instead needs TSA to create a refined optimised model. Despite potentially creating a more refined (smoother) design than a discretised mesh, no specific manufacturing features are considered within this procedure. In its current format IGA can only be used on relatively “simple” geometry and loading, with FEA retaining its superiority for more complex (and industry-relevant) models.

Alternative Refinement Methods

Recent variations of mesh-independent TO refinement methods have incorporated features not seen within commercially available TO solvers. The most notable of these includes a database storage method where a set of pre-constructed CAD models are used as the final solution to a TO structure with similar topology (Liu and Ma 2016). This process will however drastically limit the number of unique solution designs which can be created and may inadvertently produce sub-optimal designs due to the limited number of solutions. Features such as machine learning could however be adapted for database designs by automatically generating new structures with similar features, thus removing some of the issues caused if

using a limited number of designs. Alternatives also include manipulating the manufacturing tools as opposed to changing the design itself. (Lee et al. 2012) demonstrates how a manufacturing die can be re-positioned in relation to the topology optimised result, thus influencing the shape of a cast metal part when manufactured. Whereas both processes have the advantage of reducing PP CPU time, both will ultimately be limited by the number of potential solutions stored within their databases, in which continuous manual updates to the programs are inevitably required.

Methods that utilise geometry manipulation but are not currently used for TO refinement can include Geometric Iterative Methods (GIMs). GIMs are geometric programs which are able to locally modify nodal positions of a NURBS curve in order to update their location or curvature (Lin et al. 2017). These processes are heavily focussed on geometry curve generation but could be considered for PP, with refinement being potentially considered on a local level through an iterative approach.

Table 2 summarises the above mentioned key mesh-independent refinement features that currently exist.

No.	Feature Considered	Mesh-Independent Variations			
		LSM	IGA with TSA	Database Storage	GIMS
1	Uses discretised input model	✓		✓	
2	Uses VDM input model	✓	✓		
3	Potential to consider multiple manufacturing methods	✓	✓		
4	Refinement is based around a manufacturing method(s)	N/A	N/A	✓	
5	Used within existing TO solvers	✓	✓	✓	
6	Currently available as a geometry refinement processes separate from TO				✓
7	Modifies and uses input topology for output solution file	✓	N/A		

Table 2 – Overview of available mesh-independent refinement techniques

The mesh-independent variations summarised in Table 2 indicate that no current features allow for both the refinement of VDM and discretised solutions whilst also separating this process from an initial TO process. GIMs is the only refinement process that is not currently integrated into a TO solver but it is also limited to only refining CAD lines and not TO solution files. Currently LSM, IGA with TSA and Database Storage can refine TO solution files but not separately from the TO solvers they are integrated with. There is a high importance towards ensuring TO solutions can be refined separately from an existing TO solver to enable the inclusivity of refinement of multiple file types using a single refinement process.

3.3 Mesh-Dependent vs. Mesh-Independent Methods

Step 1 of the proposed APP (Figure 2 – Section 2) aims to refine a TO solution by utilising a standalone unique refinement methodology not provided by existing PP methods. An important feature to consider for this step is whether the APP should display a degree of mesh-independence from the solution FEA file. Table 3 highlights the advantages and disadvantages of using either process.

	Advantages	Disadvantages
Mesh-Dependent Method	<ul style="list-style-type: none"> - Relatively low CPU time as less data is read - Can call all solution data from a single (FE) solution file 	<ul style="list-style-type: none"> - Only reads element and node data (does not consider void space) - Geometrical/feature distance is harder to define - Refinement only considered on a local level, relative to specific features
Mesh-Independent Method	<ul style="list-style-type: none"> - Potential to refine CAD models other than FEA/meshed solutions - Considers geometrical spacing (distance) of members from one another - Modification of feature search parameters allows as little or much data to be recorded as required - Improved accuracy over mesh dep. (Section 5) 	<ul style="list-style-type: none"> - Relatively high CPU time (compared to mesh dependent) due to extra parameters read and created - Possibly uses a more complex method due to the additional location parameters.

Table 3 – Use of a Mesh-Dependent vs. a Mesh-Independent Method for Post-Processing

Since it is desired that step 1 of the APP should refine a variety of TO solution files, it is a worthy assumption that the file formats of these solutions will differ, with some solvers not even generating a discretised FEA model (Hughes et al. 2005). Instead, suitability may be drawn to searching for features to remove or add that are not mesh-dependent, such as the presence of solids or voids within a searchable workspace. A mesh-independent post-processor will allow for the consideration of manufacturing features in a relative geometrical space instead of only on an element and node only basis. It is therefore desired that a mesh-independent process is used for step 1 of the APP. However, it should be made apparent that as no available comparative method exists for the proposed APP, it may be desirable to generate a similar mesh-dependent variant of step 1 as a means of comparison of the method's capabilities. Comparison and testing of the two variations are shown in Sections 4.3, and 5, respectively.

3.4 Summary of Available Standalone Post-Processing Methods

Table 4 summarises recent practical applications of PP of TO solutions as discussed in Sections 3.1 and 3.2. Examples considered involve either geometry refinement methods or integrated TO-PP solvers, with a focus towards adapting these processes to the proposed mesh-independent APP. It should be noted that none of the methodologies presented in Table 4 are available as standalone PP processes and in some way need to be implemented into TO solvers.

Reference	Automated Process	Includes Model Refinement	Includes Manufacturing Methods	Suitable for Sheet Metal Manufacturing	Binary Solution Created
Variable Density Method	✓				
Heuristic Optimisation	✓				✓
129-Line Level Set Topology Optimisation (Challis 2009)	✓				✓
Isogeometric Analysis with Trimmed Surface Analysis (Kang and Youn 2016)	✓	✓		✓	✓
3D Machined Database Optimisation (Liu and Ma 2015)	✓	✓	✓		✓
Die shape design of tube drawing process (Lee et al. 2012)	✓		✓	✓	
Geometric Iterative Methods (Lin, et al. 2018)	✓	✓		✓	
"Identifying boundaries in topology optimization results using basic parametric features" (Yi and Kim 2016)	✓	✓		✓	

Table 4 – Overview of Current Refinement Solvers and Programs

A majority of the highlighted procedures from Table 4 consider the integration of PP refinement into TO solvers. Additional examples include works from (Koguchi and Kikuchi 2006), (Tang and Chang 2001) and (Parvizian et al. 2012) which include refinement within the topology optimisation solver, with (Hsu and Hsu 2004) and (Larsen and Jensen 2009) which both feature automated refinement of optimisation models. These processes, however, all include the refinement of a meshed topology solution, whereas it is desired that an automated post-processing program can be performed for a variety of solution files (no just meshed solutions). Currently, to the authors' knowledge, no methods directly involve a standalone PP process refining specifically TO solution files.

As different optimisation solvers create different model outputs, it is desired that the APP is separate from any TO solver so that it can refine a variety of optimisation file types. Furthermore, only two available solvers consider specific manufacturing methods to refine the solution for manufacture, with only (Lee et al. 2012) implementing manufacturing of sheet metal components. This lack of consideration shows a significant gap in the variety of manufacturable solutions that can be automatically generated by software, instilling the understanding that specific features and manufacturing processes are only able to be post-processed manually.

In summary, it is shown from Table 3 and Table 4 that no standalone post-processor of TO solution files exists. Because of this, there are no repeatable refinement methods as any comparative processes do not run separately from their integrated TO solver. These processes are not parameter driven, meaning that any refinement that takes place in the existing TO solvers is not guided to specific user-defined criteria, reducing the consistency and user guidance to produce a desired refined topology. Additionally, as these processes do not include standalone post-processing, the refinement is limited to refining one type of TO solution file and is not a universal solution for TO refinement overall.

4 Methodology of Automated Post-Processor (APP)

This section introduces development of the unique standalone semi-automated APP. The initial development will address the key setbacks discussed in Section 3, with an initial focus on the refinement of FE-based TO.

4.1 Overview of Automated Post-Processor (APP)

Based on the discussions of the previous sections the main feature choices for the APP of this paper are listed below along with the primary justifications:

- **Stand-alone processor:** To avoid unnecessarily restricting the TO solution space or reducing TO algorithm efficiency whilst maximising application versatility of the APP.
- **Mesh-independent topology feature search method:** Minimising the influence of FE mesh size upon PP results. Note the mesh size may influence the TO results themselves.
- **Semi-automated post-processor:** This paper only focuses on automating parts of the PP process in order to explore and validate the selected steps in detail. Using a modular programming approach this enables features to be added or removed; e.g. catering for different manufacturing constraints, input formats or even the extension to a fully automated approach.
- **Solution file format:** For optimisation of automotive components FEA is most often used to assess structural performance, manufacturing analysis etc. Therefore, the input and output files for the APP will be in FE format.

Implemented Features

It is desired that the proposed APP is able to refine TO solutions of varying file formats and variations in complexity. In its initial development, it is desired that a version of the APP to be tested under suitable loadcases should be able to include the following features, with additional complexity to its refinement being able to be implemented within future developments (see Section 6):

- Add and remove material (elements) from a 2D (x-y axes) TO solution model
- Incorporate methodology similar to that of the LSM, with focus on keeping the fundamental APP methodology mesh-independent
- The APP will initially attempt to refine TO solutions with regular shaped meshes
- Whereas the APP is designed to be mesh-independent and able to refine varying meshed and non-meshed file types, focus will be made for the code to refine meshed TO results.

Considering these essential features and the three-step approach to PP outlined in section 2, a flowchart of the overall APP steps can be found in Figure 7. Please note Figure 7 refers to a “stencil method”, which will be explained in detail in section 4.2.1.

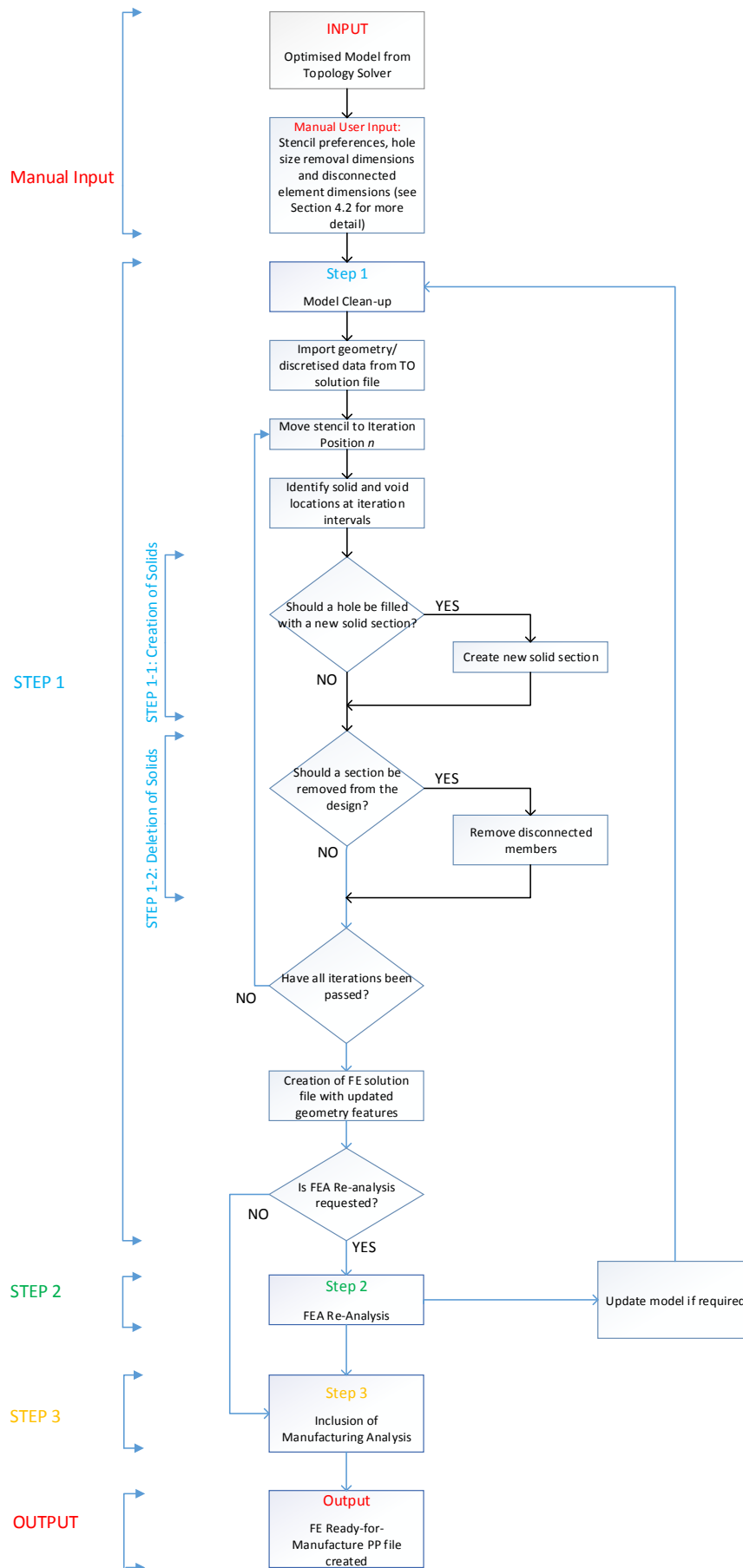


Figure 7 – Flowchart overview for individual steps in APP

In step 1, an initial model clean-up is performed by firstly identifying (global) topological features. These are then refined, with initial considerations of manufacturing constraints to ensure a “coherent” geometry representative of the original TO solution is generated. This may involve removing or adding elements to the structure so that issues such as material checkerboarding (from VDM input formats) are eliminated.

Following the refinement, there are two options, to proceed to step 3 (Figure 7) or verify the performance of the refined structure via FEA, subject to the original load case(s) and performance constraints defined in the TO step. If the latter option is chosen and the refined structure meets the performance constraints the process will continue to step 3. If the constraints are violated the “severity” of the violation will be determined and step 1 repeated. At this stage the user may be prompted on how they wish to proceed. As previously discussed, it is generally possible to complete steps 2 and 3 of the APP methodology using pre-existing commercial software applications such as batch meshing in Altair HyperWorks (Altair 2019). The remainder of this paper will therefore predominately focus on step 1 (Figure 7).

4.2 The Automated Post-Processor Stencil Method

One of the desired key-attributes of the APP is that it should be mesh-independent, despite that input and output format will both be in FE formats; i.e. in the format of geometry discretised through a finite number of elements. In this context the term mesh-independency does however relate to the topology identification and refinement of step 1. It is desired that this step remains independent of the input mesh/element size, including models with multiple or varying element sizes. In other words, if two models of identical (global) topology, but with different mesh sizes are post-processed by the APP, the two results should be consistent.

The desired mesh-independent process possesses some similarities to the previously outlined LSM. Both methods propose a search method that can identify specific geometry features of an input geometry or image. Reasoning for not directly incorporating the LSM process in the APP is that a general LSM procedure identifies geometric features on a global level and does not allow for the search space distances to be modified during a particular run. This aims to be avoided when using the APP, in which a series of modifiable search distances can be used to optimally locate specific features. LSM does not inherently consider the modification of geometry, only initially recording boundaries of geometric features. It is desired that a process can be created to modify these boundaries on a local level for specific features, a consideration not present in the method by Kang and Wang (Section 3.2). This is more likely to be achieved by ensuring a separation of the TO from the PP step.

With these considerations, two sub-processes can be outlined for step 1 of the APP:

Step 1-1: VDM to Binary Solution

This optional step is introduced to cater for input files which contain continuous design variables from TO solvers using the VDM method. The purpose of step 1-1 is to convert the continuous design variables of the VDM solution into discrete ones; i.e. a binary format. This is simply achieved by deleting any elements with a density below a threshold set by the user, thus obtaining a binary (solid/void) format as illustrated in Figure 8.

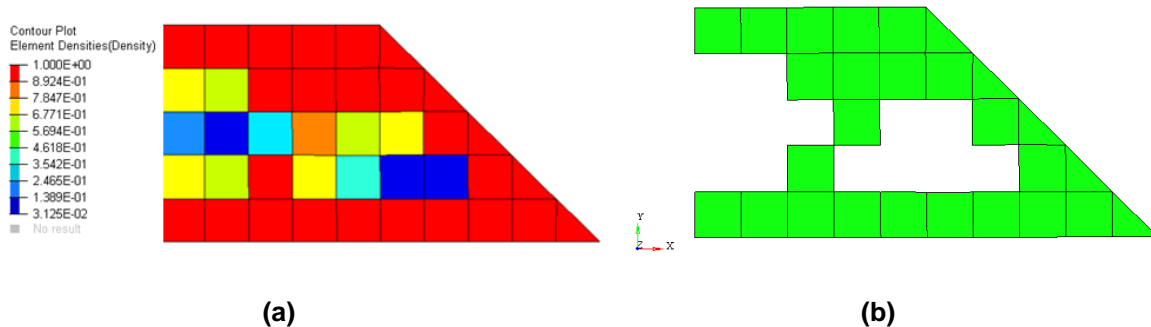


Figure 8 – (a) Input VDM results file (b) Output Binary result using APP Step 1-1 using a density threshold of 0.8

Step 1-2: Element Creation/Deletion (Stencil Method)

Based on parameter values (x_i) set by the user, step 1-2 has two distinct characteristics namely the ability to:

- A. Create new elements (filling voids) removing features that are enclosed by other solids (elements). For example, a “circular” hole with a diameter smaller than x_1 will be “filled in” by creating new elements.
- B. Delete solid (element) members that are disconnected from the “main” topology; e.g. a “group” of elements with a surface area or volume less than x_2 will be deleted.

In order to obtain useful and relevant results; characteristics A and B must be accompanied by a series of rules, in which a certain combination of topology/geometry will result in element creation/deletion. Before these rules can be applied, a method must however be established to systematically read and record the geometrical data.

Inspired by the iterative nature of GIMs, the boundary identification of the Level Set Method and the necessity to consider neighbourhoods, the authors propose to use a stencil method adaptation. In its simplest form a stencil method consists of a kernel which reads and updates “elements” whilst moving in a fixed pattern; i.e. grid. Stencil methods are most commonly used in connection with finite difference solvers (as opposed to finite element solvers) and are also widely applied to solve e.g. optimisation, computational fluid dynamics and partial differential equation problems. To the best of the authors’ knowledge, the method has however not previously been applied to post-processing of TO. In order to successfully implement a stencil method there are three main aspects to consider:

1. Stencil (kernel) shape
2. Iteration process
3. Topology updating tools and process

Each of the three aspects will be discussed in subsequent sections. For clarity this will be done with reference to a two-dimensional geometrical space (Figure 1), but the methods and principles can straightforwardly be expanded into three-dimensional geometrical space.

4.2.1 Stencil Shape

The iterative stencil or kernel can in principle assume any shape. Generally, there are four key parameters to consider, namely the number of search points, their location relative to the centre of the stencil, as well as the shape and dimension of the search area associated with each search point. Figure 9 illustrates a range of 2D stencil shapes.

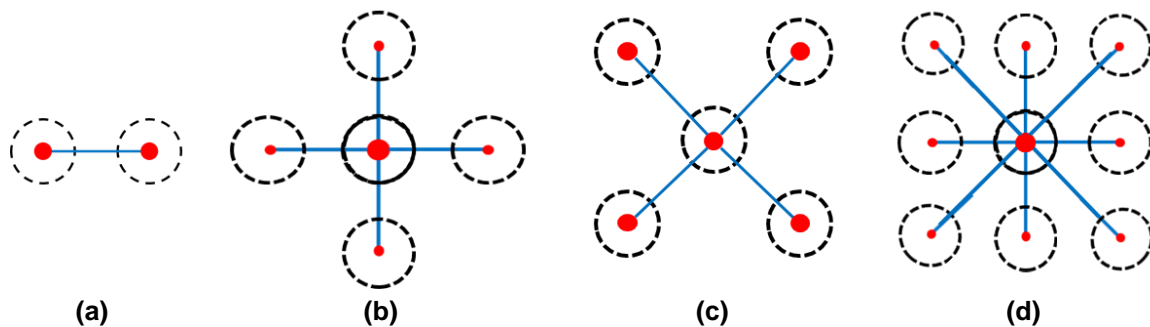


Figure 9 – Example 2D stencil variations: (a) two-horizontal (b) plus-shape (c) x-shape and (d) box-shape

The simplest stencil shape (Figure 9 a) requires less data to be read and stored leading to a reduction in CPU and memory requirements when compared to the more complex stencil in Figure 9 d. A potential disadvantage of the simpler stencil shapes is the reduction in “topology resolution” i.e. geometric details stored for a given grid point. The effects of different stencil shapes will be explored in the subsequent case studies.

Figure 10 illustrates a two-dimensional stencil with four search points (SP), the iteration centre is denoted IC, the search point areas are circular and defined by the search radius (SR) and the distance of an offset search point from the IC is noted as the Arm Length (AL).

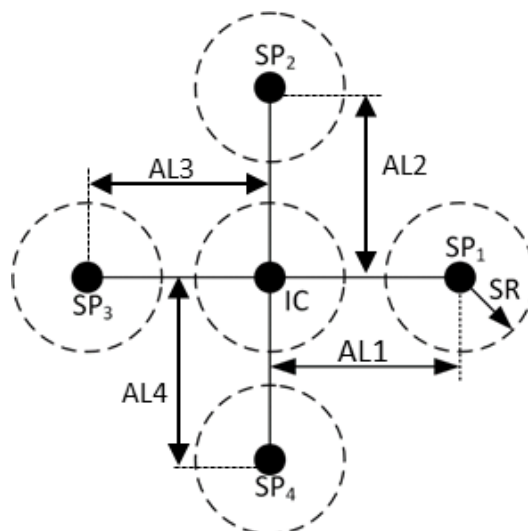


Figure 10 – Visual representation of two-dimensional stencil with 4 search points

The stencil in Figure 10 will search and record data (e.g. solid/void) for each of the four locations (arms) and the IC for each iteration; i.e. as the IC moves through grid point locations. The detection and status decision rules for the IC will be subsequently discussed in Section 4.2.3.

An alternative to using the fixed arm stencils illustrated in Figure 10 is to use a variable arm stencil that modifies the arm lengths, which works as follows. When starting the search at a given grid point all 4 ALs are identical according to a pre-set value. Consider the scenario where IC, SP3 and SP4 register “solid” values whereas SP1 and SP2 register “voids”. In this case AL1 is extended up until SP1 registers a solid point or an upper arm length limit AL_{i_max} is reached. The process is then repeated for SP2.

One potential benefit of the variable arm length stencil is that it requires less data to be stored, as the data recorded can be used to “skip” grid points thus reducing CPU time. Comparisons of these two variations are also discussed in Section 5.

4.2.2 Detection of solid/void at individual search points

The detection of a solid or void location is governed by the SR, in combination with the Cartesian equation of a circle according to equation (1).

$$SR^2 \geq (x_1 - x_0)^2 + (y_1 - y_0)^2 \quad (1)$$

As an example, consider the search point (SP) located at (x_0, y_0) and the element centre P (representing a solid point) located at (x_1, y_1) as illustrated in Figure 11.

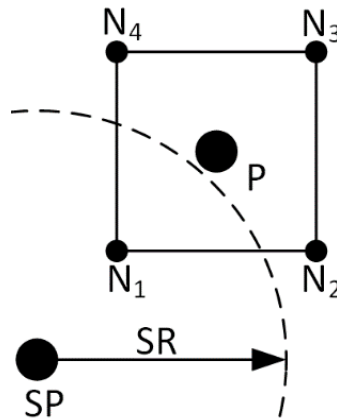


Figure 11 - locating geometrical features for a stencil search point

The determination of solid/void locations can be completed using element geometry, for example element centres and equation (1). The method is robust for determining solid/voids for meshes with consistent element size and minor element distortion, subject to appropriate selection of AL, SR and indeed grid size. The robustness of the method does however dramatically decrease with increased mesh irregularities; i.e. varying element size and distortion. As an example, consider Figure 11 where the element centre P is outside the SR, but a significant part of the element clearly lies within the SR. To alleviate this “uncertainty” it must firstly be decided if the above scenario (Figure 11) constitutes a solid or a void location.

Assuming that it constitutes a solid location, the issue could be resolved by considering nodal locations as opposed to element centres.

A high number of variations of the search areas could be introduced; for example, AL or SR could be changed for each SP individually. Alternative search area shapes could also replace the SR, for example by using a rectangular area which would create a search “box” instead of a circle. For clarity this paper utilises a circular search with identical AL and SR values for all SPs as illustrated in Figure 10. The effects of varying the specific values of AL and SR will be investigated in the subsequent case studies. Focus will however be made towards the ability for the APP to refine topologies with regular, rectangular meshes, with subsequent test cases (Section 5) focussing on these regular meshes. Consideration of testing regular meshes is taken due to the increased complexity and number of variables needed to consider for variable mesh sizes. It is expected that refinement of irregular meshes is possible for the APP if consideration to varying iterations and stencil shapes is considered. For this paper, features such as regular iteration intervals (Section 4.2.3) and the four stencil shape variations (Figure 9) will be considered alongside regular meshes.

4.2.3 Determination of Iteration Centre solid/void status

Assuming that a solid/void status has now been recorded (using equation (1)) for each of the 4 individual SP of the stencil illustrated in Figure 10, a decision must now be made on the status (solid/void) of the IC before the stencil moves to the next grid point. In situations where all 4 SP record the same status, i.e. solid or void, the decision is straightforward, but the decision is not so straightforward when this is not the case. Many factors including AL and SR values, the number of SP as well as the shape of the stencil and its iteration method influence the stencil level rules for determining the status at the IC (grid point).

Rules determining IC status can be categorised into those who favour solids (FS) and those who favour voids (FV). Fraction rules can then be set up to determine the IC status at any given grid point as follows:

$$FS: \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i} \geq Th \Rightarrow IC = 1 \quad \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i} < Th \Rightarrow IC = 0 \quad (2)$$

$$FV: \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i} > Th \Rightarrow IC = 1 \quad \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i} \leq Th \Rightarrow IC = 0 \quad (3)$$

In equations (2) and (3) SP_{solid} is the number of SP statuses determined as solid, n denotes the total number of SP and Th represents a user defined threshold value between 0 and 1. Note that equation (2) favours solids (FS) whereas equation (3) favours voids (FV). As a demonstrative example consider a threshold value of 0.5 and the scenario illustrated in figure 12, where the number of SP_{solid} equals two and n equals 4.

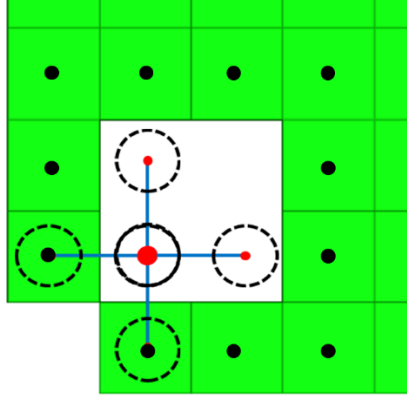


Figure 12 – Stencil locating element centres to determine presence of solids.

According to equation (2), IC will be set equal to 1 (solid) whereas for equation (3) IC will be set to zero (void). Note that equations (2) and (3) do not utilise the status of IC, but this could however be added; equations (4) and (5):

$$FS: \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i + sf * IC} \geq Th \Rightarrow IC = 1 \quad \vee \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i + sf * IC} < Th \Rightarrow IC = 0 \quad (4)$$

$$FV: \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i + sf * IC} > Th \Rightarrow IC = 1 \quad \vee \frac{\sum SP_{Solid}}{\sum_{i=1}^n SP_i + sf * IC} \leq Th \Rightarrow IC = 0 \quad (5)$$

In 4 and 5 sf represents (a potential) scale factor to reduce or increase the weighting of the IC status read. If the IC status is read as solid it would add to the SP_{solid} count of the numerator including any sf multiplication. Returning to the example of Figure 11, assuming sf equals 1.0 and Th equals 0.5, IC would be set to 0 (void) for both rules in 4 and 5 as 2/5 is less than the Th value.

Note that the above rules do not consider the relative locations of the solid void SPs; e.g. if SP1 and SP4 reading solids would not be any different to SP1 and SP3 reading solids (Figure 9). Furthermore, the rules do not consider any influence of variable arm lengths (Figure 10), this is primarily because the variable stencils introduced are intended to detect “void sizes” and skipping individual grid points to reduce CPU time.

The **IC Status Rule** (ICSR) parameter will be used to determine the solid/void status of the iteration position the stencil currently occupies (positioned at the stencil’s iteration centre – IC). In this paper, an accompanying process defined as the **Search Point Status Rule** (SPSR) will be used to determine the solid/void status of an individual (local) search point on the stencil, using the variations described in equations (2), (3), (4) and (5). When using the SPSR rule for each search point, a ratio of solids/voids will be determined and recorded. If this value is larger than a user-defined value labelled, **Solid/Void Ratio Threshold** (SVRTH), a solid is recorded at this point. Conversely for the ICSR rule, equations (2) and (3) will be used to determine the overall IC ratio for the stencil location. If this ratio is above another user-defined parameter, **Solid/void Total Ratio Threshold** – STRTH, the overall IC location is treated as solid. It is important to note that the user must consider modifying this parameter when using different stencil shapes. For instance, if a five-point stencil is used, the user may want more than 50% of the recorded search point values to be solid when declaring the overall IC solid. In this case it is expected that a three out of five (0.6) ratio is established for a five-point stencil

but a two out of four (0.5) ratio is established for a four-point stencil etc. The effects of these parameters will be extensively investigated in the subsequent case studies (section 5.5). Additionally, if the local and global thresholds are exactly equal to the SVRTH and/or STRTH values respectively, additional user-defined parameters labelled **Favour Solid Status at Point** (FSSP) and **Favour Solid status for Iteration Centre**, (FSIC) are referenced. This allows the user to favour the status of a solid or a void for an individual search point or the overall status if this threshold is exactly matched during the SPSR and ICSR process. Furthermore, the user can also define whether the iteration centre, IC, location should be included within the status determination or not by referencing another defined parameter, **Iteration Centre INCLuded**, ICINCL. A list of all these parameters and their functions is provided in Table 5, section 4.2.5.

4.2.4 Topology Updating tools and Process

Before the specific tools to update the topology are introduced, the overall process, i.e. the stencil iteration and the topology updating methodology must firstly be defined.

Iteration process

Although the way in which the stencil iterates through the grid may influence the resulting topology this will not be explicitly explored in this paper. For the 2D case studies presented the grid will be equally spaced according to the user defined grid spacing parameters GS-X (horizontal) and GS-Y (vertical), as well as the inclusion of only regular meshes in the subsequent test cases. The stencil will iterate from left to right and bottom to top as illustrated in Figure 13. Additionally, the iterations will start at an “offset value”, which will be determined by the input values provided for GS-X and GS-Y. The starting iteration coordinate position is determined as shown in equation (6):

$$((x_{\min} - GS-X), (y_{\min} - GS-Y), 0) \tag{6}$$

Where x_{\min} is the smallest x-coordinate position in the provided topology and y_{\min} is the smallest y-coordinate position.

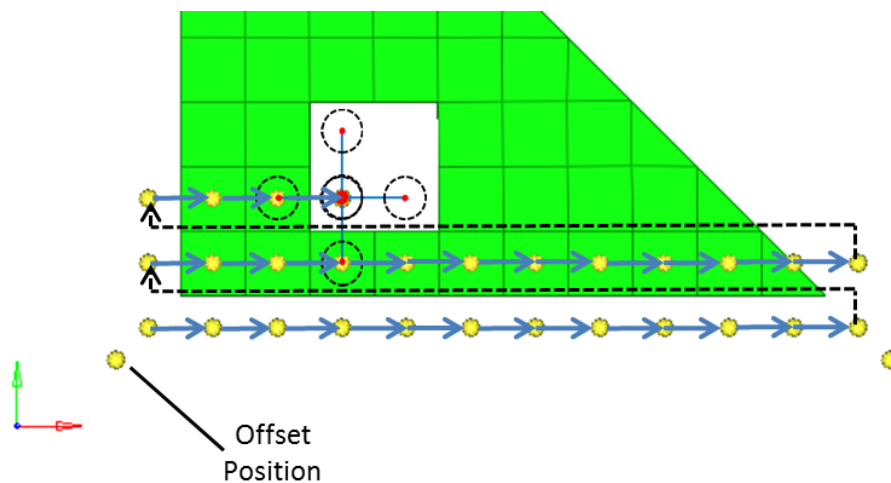


Figure 13 – Stencil iteration

Another significant factor in the topology updating process is the sequence in which the void and solid statuses of grid points are updated and elements created/deleted. This can either be once the stencil has finished its iterations and recorded the data for all grid points or it can be done as the stencil iterates throughout the grid. The former will be referred to as post-iterative (PI) whereas the latter will be referred to as mid-iterative (MI), with the results of these differences highlighted in section 5.10.2. As an example, consider the scenario illustrated in Figure 13. Subject to the IC status determination rules (section 4.2.3) the PI will record voids at 4 search points (SP), whereas the MI may record a different outcome because the “void” may be filled in as the stencil iterates from left to right and bottom to top. The effects MI and PI have on the quality and accuracy on the topology results will be discussed in Section 5.

Based on the above rules and the PI or MI process, the topology represented by finite elements can now be updated through the creation or deletion of individual elements. In particular, the focus of this will be to prevent issues such as material checkerboarding by “filling in” the void spaces where appropriate. Consideration will be made to refining geometric features whilst avoiding adding “too many” elements into a structure thereby strongly changing the load paths of the topology optimised design.

Element creation

To demonstrate the creation of elements (in step 1-2) the FV rule (not considering the IC scale factor) in equation (3) is utilised with the Th value set to 0.5. Now consider the starting scenario with the topology and stencil located as in Figure 13. At this grid point, a void is detected at the IC, with solids detected at SP3 and SP4 as illustrated in Figure 14.

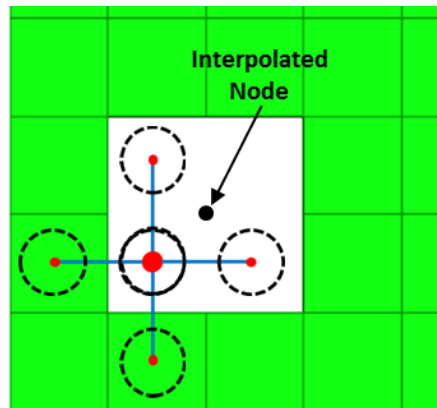


Figure 14 – Stencil iteration

With a void detected the APP will search for an enclosed “hole” using a database of element centres, nodal locations and surrounding stencil grid data. Note that the specific rules for updating depend on a number of additional parameters including the selection of MI or PI, stencil shape and whether a fixed or variable stencil is used. The final parameter is the user-defined **M**inimum **H**ole-**S**ize in the “**X**” (horizontal) and “**Y**” (vertical) directions, MHS-X and MHS-Y respectively. Any enclosed voids (determined by void grid points) smaller than MHS-X and/or MHS-Y are filled in, assuming the stencil status determines a void(s) within this space to add elements to. Where possible, new elements are created using existing nodes and if required by extrapolation or interpolation thereof. The elements are created with the aim of maximising the Jacobian, creating equal side lengths and internal angles as well as an

average element size dictated by surrounding elements as illustrated in Figure 14. This methodology can also be used to create triangular elements and quadrilateral elements, filling the same void space that quadrilateral-only element would initially cover. Testing of variations to symmetric 2D elements such as triangular elements and distorted elements will be explored in further developments of the APP and will not be initially considered for this paper. Additionally, it should be understood that the development of an element creation process is such that the APP can refine features which are not intended to be specific design features i.e. material checkerboarding.

In certain instances, the user may want to include specific features such as dedicated holes for wires passing through a metal sheet. The APP will identify these like any other void and will either leave this or fill in the void as a function of the MHS value. This value can of course be adjusted to suit the specific hole size desired, however this will subsequently be applied to the entire structure and may consequently have a detrimental effect on the APP's performance. In order to address such instances a switch could be applied in reference to the grid-points enabling "local override" of specific parameters. Although the implementation of local override is relatively straightforward it is not of primary interest and will not be further discussed in this paper.

Element deletion

The element deletion works in a similar way to the element creation; namely by measuring the "dimensions" of connected solids. Grid point data is utilised to determine connected and unconnected areas. Just like the creation of elements, the specific rules for deleting elements depend on a number of additional parameters including the selection of MI or PI, stencil shape and whether a fixed or variable stencil is used. The final parameters are the user-defined **M**inimum **S**ection **S**ize in the horizontal (MSS-X) and vertical (MSS-Y) directions, as well as a **D**istance measure between the "**M**ain topology section" and the **D**isconnected section (DMD). Any disconnected sections/area smaller than MSS-X and/or MSS-Y at a distance equal to or greater than DMD are removed. Determination of a "disconnected" structure will be whether the DMD values and MSS-X/Y values record solid ratios less than or equal to these values. If the number of solid ratios in x and y directions is greater than these values at a given iteration position, the solid recorded will be determined as "connected" to the structure.

4.2.5 Automated Post-Processor Input Variables

All main APP parameters, controllable by the user, have now been defined. For clarity, these are listed and described in Table 5. The influence of each parameter upon the post-processed topology will be investigated in Section 5.

Variable name	Description
SS	Selection of stencil shape; available ones are defined in Figure 9, Section 4.2.1
ST	Stencil type; i.e. fixed or variable arm length, see Figure 10, Section 4.2.1. Note that this parameter directly controls the number of search points (SP)
AL_i	Stencil arm length for SP i , see Figure 10.
AL_{i_max}	Maximum stencil arm length for variable arm length stencils.
SR	Search Point Range Radius
SPSR	This parameter uses either of the four rules (equations (2), (3), (4) or (5)) to determine the status of an individual stencil search point (SP)
ICSR	This parameter uses either equation (2) or equation (3) to determine the status of the overall stencil iteration centre (IC)
GS-X/Y	Grid spacing parameters; these define the vertical and horizontal distances between grid points, see Figure 13.
MI / PI	Mid-or post-iteration updates. This switch determines whether elements are created/deleted as the stencil iterates through the grid or after the stencil iterations are complete.
MHS-X/Y	Minimum allowable hole size; holes less than this size will be filled in.
MSS-X/Y	Minimum section size; disconnected areas/elements less than this size will be deleted.
DMD	Minimum distance between the main topology and the disconnected topology section(s). Sections less than MSS-X/MSS-Y at a distance equal to or greater than DMD are removed.
SVRTH	Ratio of solids/(total solids and voids) to determine the solid/void status for an individual SR
STRTH	Ratio of solids/(total solids and voids) to determine the solid/void status for the IC position, using the SR results
FSSP	If the Th lands equal to the SVRTH ratio, the user can determine whether the SR status should be favoured as solid or void
FSIC	If the Th lands equal to the STRTH ratio, the user can determine whether the overall IC status should be favoured as solid or void
ICINCL	Identifies whether the IC search radius is included in the stencil solid/void status or not (default is to have it included).

Table 5 – List of user-input variables before running automated post-processing (APP)

4.3 Comparative Method – Mesh-Dependent Post-Processing Code (MDPP) and Extendible Stencil

As highlighted in the literature review, no comparative methods exist for a standalone automated post-processor. To benchmark the performance of the APP a suitable comparative method has been created solely relying on mesh data. This mesh-dependent process is similar in its general functionality to that of the APP, wherein it uses a series of recorded data to determine whether elements should be created or removed from the topology. The mesh-dependent method differs however through its lack of a stencil search method: as it only reads element and node data and does not concern itself with geometric spacing, only the data provided from the mesh is manipulated. This would in theory produce results that are generated in a less computationally expensive manner than that of the APP as the major stencil iteration step has been removed. It was found through preliminary testing that the development of a mesh-dependent APP has proven to be unreliable in its ability to accurately create or delete elements. This is due to the lack of data such as relative element areas and distances of elements/nodes from other nearby members. Whereas this information can be found through manipulating the provided coordinate data, the mesh-independent APP proves to do this in a more systematic way through the use of the iteration spacing. This lack of

iteration spacing in the MDPP can cause elements with large distortions to be created, which can create additional structural issues if an area limit for each element is not provided. It is therefore considered that the MDPP will not be tested any further for this paper and focus will only be made towards the mesh-independent APP.

Another comparative method previously mentioned in section 4.2.3 is the ability to extend individual stencil arms, AL , when certain criteria are met during an iteration. An example can be that if a void is detected at an iteration point, several AL lengths could be increased until the new locations of their respective search radii (SR) read solids. This would determine that enclosed voids are present and then begin creating elements for this enclosed area. Whereas this is a desired additional feature of the APP, it can be seen as a very similar method to the mid-iteration (MI) approach, with some additional updates needed into how the stencil reads data. Instead of providing this additional complexity to the MI approach, this paper will focus only on the use of a static stencil moving in an iterative manner, with no modifications made to it between these iterations. This feature should however be considered for future development of the APP.

5 Test Cases – Refining Topological Features

The methodology presented in Section 4 covers the processes of step 1 in the APP. This involves refining a topology optimised solution such that it can be re-analysed for structural performance whilst ensuring manufacturability (steps 2 and 3, Figure 2, Section 2). This section will focus on a series of test cases for step 1 and showcase the APPs ability to create and delete elements for structures in a two-dimensional geometrical space. Various user-defined parameters will be modified and monitored, with general trends and user recommendations identified within each sub-section. The overarching aim of this case study will be to define a series of explicit guidelines for the APP default parameters and provide suitable recommendations as to which combinations should be used to create a refined structure to the user's recommendations.

5.1 Test Case overview

A total of eight Test Cases (TC), incorporating 2,451 scenarios/models, are considered to explore the influence of individual parameters (listed in Table 5) upon the refined topology. Table 6 contains the default APP setup used for all test cases unless otherwise specified. Due to the large number of models produced for this case study, it is impractical to overview the setup parameters of each individual model. Therefore, an overview of each variation will be outlined in each test case in table form. Further information regarding each model's setup is available on request.

Variable name	Value
SS	2 (Plus-shape)
ST	2.0 (Fixed ALi length)
AL _i	AL1 = AL2 = AL3 = AL4 = 1.0 (unit length)
AL _i max	N/A
SR	SR1 = SR2 = SR3 = SR4 = 0.6 (unit length)
SPSR	Equations (2), (3), (4) and (5)
ICSR	Equations (2) and (3)
GS-X/Y	1.0 units (X), 1.0 units (Y)
MI / PI	PI
MHS-X/Y	1.0 (unit length)
MSS-X/Y	1.0 (unit length)
DMD	1.0 (unit length)
SVRTH	0.6
STRTH	0.6
FSSP	1 (Favour solids)
FSIC	1 (Favour solids)
ICINCL	1 (IC included in ratio)

Table 6 – Default APP setup for test cases

The individual test cases can be organised into three key considerations: those that refer specifically to element creation or element deletion, and those that consider combinations of these or other parameters. Table 7 contains a descriptive overview of the individual test case.

Test Case (TC)			
Type	#	Purpose: To determine how...	Variables
Element Creation	1	Different combinations of the ALi, GS-X iteration spacing and the size of a void space affect the APP's ability to create elements	ALi (X) GS-X
	2	Modifying the stencil search radius, SR, affects element creation	SR ICINCL
	3	Prioritising iteration status to favour either solids or voids affects element creation	FSSP FSIC STRTH SVRTH
	4	Changing SS affects iteration status and element creation	SS
Element deletion	5	Different combinations of the ALi, GS-X iteration spacing and the size of a void space affect the APP's ability to delete elements	ALi (X) GS-X
	6	Modifying the stencil search radius, SR, affects element deletion	SR ICINCL
Other	7	Switching between MI and PI affects APP results.	MI/PI
	8	The APP determines what parameters will allow for a hole to be filled and what should be considered a hole	MI/PI MHS-X MHS-Y

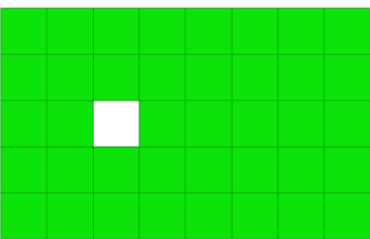
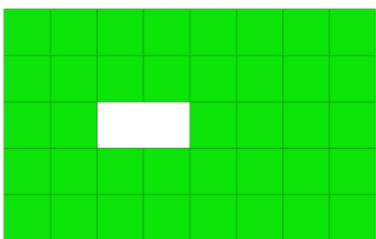
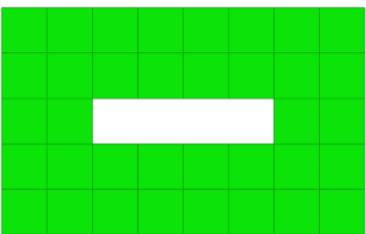



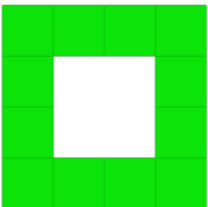
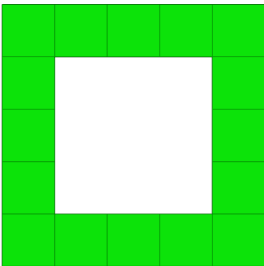
Table 7 – Test Case overview

For the purpose of completing the 8 test cases, the bank of topology optimised models (TOM) listed in Table 8 will be used. Suitable models used for the test cases will generally consist of “ideal” elements i.e. no element distortions or warpage will be present, with all elements being symmetric and of similar element size. Despite the apparent simplicity of the models in Table 8 they are defined to explore the capabilities of the APP, providing a fundamental understanding of the methodology including individual parameters through the test case results. Subject to mesh size, the models in Table 8 could also represent finer details of more complex geometries, such as the holes illustrated in Figure 3(a). The “challenges” of the test case models are therefore representative of much more complicated post-processing tasks.

All TOMs illustrated in Table 8 utilise equilateral quadruple symmetric (square) elements with an element size of 1.0 units. This will remain constant for all models throughout the case study, including any elements created by the APP itself. Table 6 lists default APP parameters used for the case study, in individual test cases these will be modified to either half, double or quadruple the defaults in order to explore the individual or combined effects upon post-processing results.

Thirteen specific models are considered for this case study, with an initial focus taken to TOMs 1-3 for element creation and TOMS 4-6 for element deletion. TOMs 7-13 are specifically considered for TC8 which varies the enclosed void shape geometry for elements to be created, in order to identify the limitations and capabilities of the APP for these designs. Some models

used consist of very few elements, such as TOMs 4 and 7. The purpose of this case study is to highlight the changes made to fundamental design shapes and element configurations in order to predict and interpret the behaviour of the APP. It should also be considered that these configurations will still be directly transferrable to more complex topologies, with multiple or combinations of these models being part of a larger topological structure. An example of this can be referred to Figure 3(a), in which smaller imperfections that include structures from Table 8 are present in much larger geometry. In order to test that the APP is able to perform refinement on a model whilst considering the multiple features described in the individual test cases, a subsequent test will be performed on a more complex model to consider both element creation and deletion for various sections. This example (section 5.11) will look specifically for whether these refinement features can be used in a single model and whether the result will be as expected by the user. Further details of the individual test cases can be found in the subsequent sections.

TOM #	Illustration	TOM #	Illustration
1		2	
3		4	
5		6	
7		8	

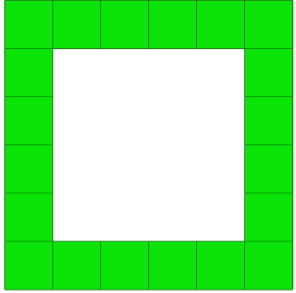
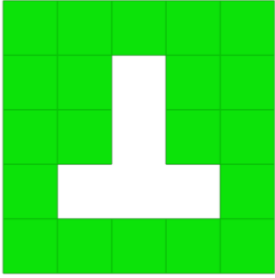
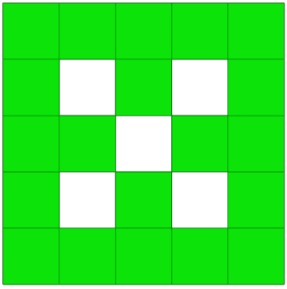
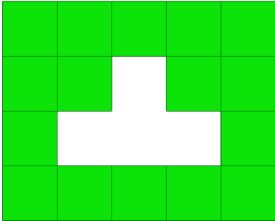
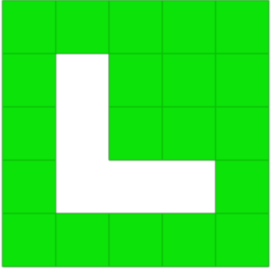
9		10	
11		12	
13			

Table 8 – Bank of topology optimised models (TOM) for test cases

5.2 Automated Post-Processor Basic Functionality and “Sanity Check”

The test cases outlined in sections 5.3-5.11 involve the modification of user-defined variables within the APP such that a series of general trends and suitability of the APP is made. In order to suitably monitor the effects of modifying these user-defined parameters, a series of “sanity checks” were performed on the APP in order to ensure its correct functionality. These included ensuring the APP can create or delete multiple elements for a given model or topologically enclosed voids or floating section, respectively. The voids or sections to delete do not only have to be rectangular sections; they only need an x/y member length smaller than the MHS-X/Y or MSS-X/Y value in that respective axis. Additionally, the APP can extrapolate and create new nodes when creating new elements, if those element’s nodes do not currently exist.

After establishing the basic functionalities of the APP, two parameter rules were introduced in order to exclude “non-logical” scenarios and prevent excessive amounts of duplicate data to be stored throughout the stencil iterations. The two rules are defined as:

- R1:** $SR \leq 0.5AL_i$
R2: $GS-X/Y \leq MHS-X/Y$

5.3 TC1 – Stencil Arm Length vs. Iteration Length vs. Hole Size

TC1 contains 27 models and explores the ability for the APP to create elements within a defined void space subject to several modified parameters. As the APP uses a stencil search approach to locate and fill voids with new elements, it is desirable to identify the limits of its ability to create/not create these elements. This can be tested through the modifying of parameters specific to the stencil search and iteration values.

Using TOMs 1, 2 and 3 (Table 8), Table 9 outlines the variations made to the input APP parameters in order to indicate which ratios are suitable for consistent element creation. It should be noted that for TC1, as well as any subsequent test cases, focus will be taken into modifying lengths in relation to the X-position of the stencil and iteration position. Modification of one axis parameter will allow for the effectiveness of each modified variable to be more apparent.

		GS-X		
		1.0	2.0	4.0
AL _i (X)	1.0	X	X	X
	2.0	X	X	X
	4.0	X	X	X

Table 9 – TC1 parameter variations

5.3.1 TC1 Results – Arm Length Variation

Modifying the x-position stencil arm length, AL_i, in this case study shows little influence on the APP’s ability to identify solids and voids at each iteration point. It is found that if AL_i (X) is larger than the average (X) element size, the stencil may record an iteration point as a void as opposed to a solid if AL_i (X) is equal to the average X element length. However, assuming the solid/void ratio to determine the presence of a solid or void at an iteration space remains the same, the modification of the AL_i parameter may not have a significant influence on the solid-void status determination. Figure 15 is an example of how iteration statuses remain the same for a specific iteration location when different AL_i values are used. Assuming a void to be made solid is determined by where at least three of the five search points register as solid, each

example in Figure 15 will register the iteration point as equal to 0 (void which should become a solid). Whereas this is beneficial under these model setups, problems may occur if the stencil becomes significantly distorted for more complex shapes. It can therefore be considered that inaccuracies in void recording can be made if the AL_i length is significantly larger than an average element size. Inclusions of additional detection rules to identify issues such as “out of bounds” stencil positions can be considered for further updates to the APP.

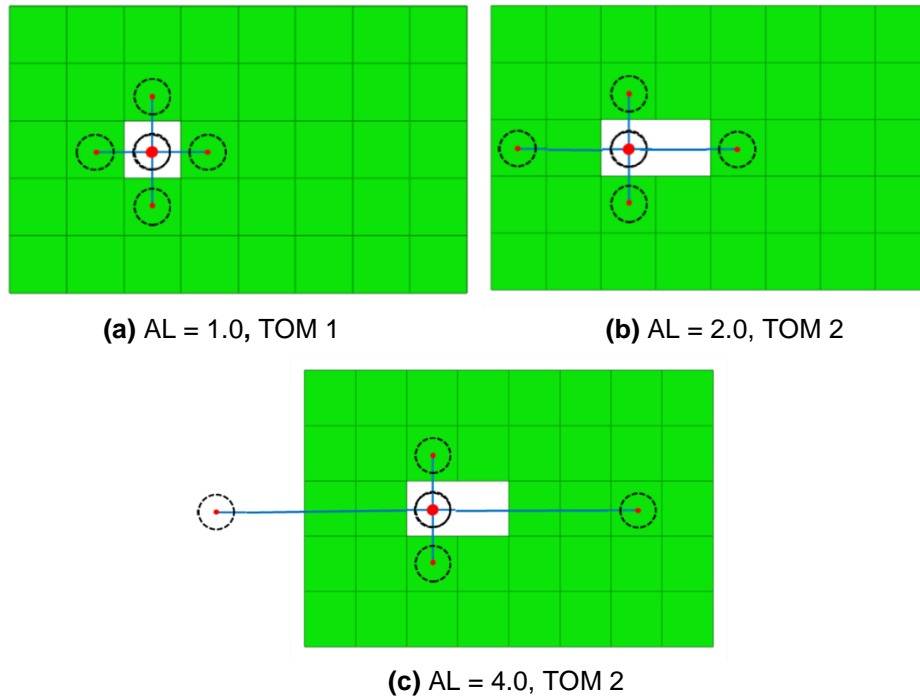


Figure 15 – Variations in AL: despite using different AL lengths, the ratio of solids/voids detected remains equal to or above 3/5, meaning each stencil location will be recorded the same for the void and its surrounding elements

5.3.2 TC1 Results – Grid Spacing Variation

Modifying the x iteration intervals, GS-X, has shown to affect the APP’s ability to correctly fill void spaces with new elements. Most notably, when GS-X is larger than the x AL_i (GS-X = 1.0 and 2.0 in Table 9), no elements were created in the model. An understanding of this can be interpreted in Figure 16, whereby increasing the length of GS-X will also increase the offset starting position’s distance from the structure (see section 4.2.4). This will also offset the positions at which the stencil records data, with some positions missing data that would otherwise be included when GS-X is equal to $AL-X$. By missing some solid/void positions, the element creation process will consequently be affected, showcasing the reliance on recording the stencil data correctly initially. It can be recommended based on these findings that GS-X should be close, or equal to, $AL-X$ to ensure correct element creation. Alternatively, having the user define an offset position for the stencil that is not half of the iteration size may increase the likelihood of elements being created.

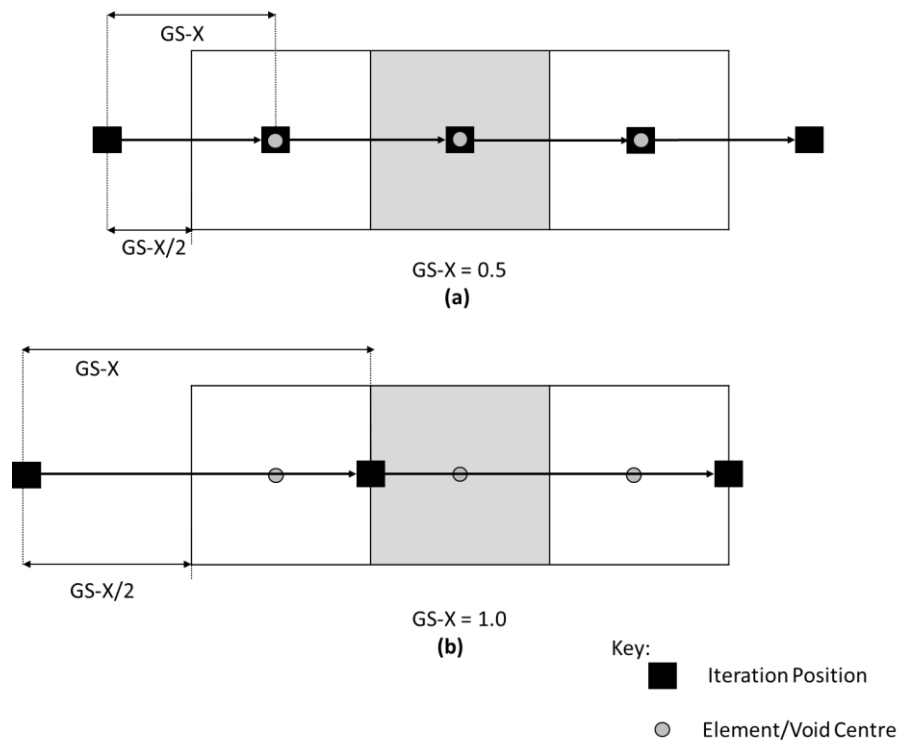


Figure 16 - (a) – $GS-X = AL =$ average element size: each iteration point lands on a predetermined solid/void location, increasing the likelihood of accurately recording locations **(b)** – $GS-X > AL$ or average element size: iteration points are offset such so that solids may not be recorded for neighbouring positions

5.3.3 TC1 Results – Hole Size Variation

Consideration of varying enclosed void size as illustrated in TOM 1, 2 and 3 has shown no variations in result generation when considering the previous variables. This can be contributed to the stencil's ability to locate iteration positions that can potentially be filled with elements (0). Assuming detection of these features shows no dissimilarities, identification of voids in the x-row will be performed in the same manner for all model variations.

5.3.4 TC1 Results – Hole and Grid Size Variations and Conclusions

Combinations of the above tested variables identify that modification to the iteration increment ($GS-X$) is the greatest influencer of whether solids and voids are correctly recorded and stored by the stencil status. It is recommended that $GS-X$ is close in magnitude to the AL value. Recommendation to ensure AL is close to an average element size should also be considered, due to the potential increased risk of recording incorrect stencil status's through missing nearby iteration data. This recommendation may not be the most practical solution, and it should be considered that this offset position can be manually determined for future developments of the APP. Additional parameters can be modified to potentially allow for more data to be recorded for each iteration position even if the position does not lie close to an element or void centre. Examples can be to increase the stencil search radius, SR , or prioritise the creation of elements with a lower solid-void ratio, $SVRTH$ and $STRTH$. These parameters will be addressed in TC2 (Section 5.4) and TC3 (Section 5.5), respectively.

5.4 TC2 – Stencil Search Radius Variation

TC2 contains 205 models and identifies the APPs ability to create elements with varying stencil search point radii, SR. This involves modifying the SR value for all stencil points for the models run in TC1 (TOMS 1-3, with varying ALi (X) and GS-X). The SR value has been modified to be either less than the default ALi, 0.6 (default), equal to it, 1.0, or larger than it, 1.4. It should be noted that the increase in SR must not allow the individual search radii to overlap, as indicated in Section 5.2. In accordance to this, any parameter configurations that violate rules R1 and R2 from Section 5.2 will be omitted, such as if the SR is larger than the ALi of the stencil. Table 10 highlights these modifications made to the TC1 model setups. Table 10 will also highlight the inclusion, or lack thereof, of the IC stencil position within the status determination, as explained in section 4.2.3. Two variations, one where the aggregate of all stencil points is found to determine the iteration status (ICINCL = 1), or the aggregate of all stencil points except the IC (ICINCL = 2).

		SR		
		0.6	1.0	1.4
ICINCL	1 (IC included)	X	X	X
	2 (IC not included)	X	X	X

Table 10 – TC2 Parameters

5.4.1 TC2 Results – Search Radius Variation and Combinations with TC1

For the model setups in TC2, the APP is able to create elements under the same TC1 parameter conditions when increasing the SR to the larger 1.0 and 1.4 values. This can be due to TOMs used including symmetric elements, meaning the solid/void ratios would stay consistently the same even if more data is read by the SR. This ability for the SR to prioritise whether a solid or void should be created is determined through the user-defined ratios, SVRTH and STRTH, which will be explored in TC3.

When excluding the IC from the element status determination (ICINCL = 2) consistent position status and element creation is attained as with when ICINCL is equal to 1, for STRTH = 0.5. This similarity is less consistent when STRTH is equal to 0.6 (the default ratio used for five stencil points), in which several model configurations do not create elements where their ICINCL = 1 counterpart does. This is simply due to the lower number of stencil points being read when ICINCL is 2 making the criteria to determine voids to fill being harsher if kept the same. It is therefore recommended that the STRTH value is updated to accommodate to the number of stencil points included in the status determination. Both the STRTH and SVRTH values will be modified and analysed within TC3 in Section 5.5.

5.5 TC3 – Determining Solid/Void Status

This test case uses 359 models and identifies how modifying the solid/void ratios affects element creation. Determination of the solid/void status is derived from a user-defined ratio of solids and voids detected by the stencil at individual search points, SVRTH, or for the overall stencil, STRTH, which a user will both provide. These values will be modified for the models previously run for TC1 and TC2. Initial threshold values for SVRTH and STRTH of 0.6 (3/5 solid-void ratio for a five point plus-shape stencil) have been provided for TC1 and TC2. The 0.6 ratio aims to provide a slight bias towards the detection of solids without over-constraining and not unintentionally recording all voids as solid. In addition to these two parameters, further consideration is made for what the iteration point status should be if the ratio is exactly equal to the provided STRTH and SVRTH ratios. The default settings treat the position as a solid if STRTH or SVRTH equal the solid-void ratio, with the additional favouring of voids tested using parameters FSSP and FSIC.

In order to identify which ratio is most proficient in suitably detecting and filling in voids with new elements, Table 11 highlights the variations of SVRTH, STRTH, FSSP and FSIC tested within TC3.

		SVRTH		
		0.4	0.6	0.8
STRTH	0.4	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)
	0.6	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)
	0.8	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)	FSSP = 1(solid), 2(void) FSIC = 1(solid), 2(void)

Table 11 – TC3 Parameters

5.5.1 TC3 Results – Local Stencil Solid/Void Ratio Modification

Locally modifying the SR ratios has shown to affect the APP's ability to record individual stencil points as solid or void. Using a lower threshold of 0.4 (40% of all read points in the SR are solid) has shown to pose no significant change to the generated results from TC1 and TC2. This can be due to the equal spacing of the element's centres and voids, in which ratios will stay consistent. If a lower SVRTH value is used, the same void and solid locations will be recorded. Differences from the TC1 and TC2 results are present however when SVRTH is increased to a higher ratio of 0.8. Correct identification of solids and voids is present when using an SR smaller than the AL (0.3) but is less consistent when using SR values equal to or greater than the AL (0.5 or 0.7). This is due to the increased numbers of void points recorded when using the larger SRs, resulting in more voids being recorded and the ratio lowering.

5.5.2 TC3 Results – Global Point Solid/Void Ratio Modification

Modifying the global parameter STRTH has shown to hold a greater effect on the APP’s ability to identify suitable voids to fill with new elements, when compared to locally modifying the ratio. Most notably, when the overall stencil ratio to determine a solid is set higher than the default 0.6, the stencil is less likely to determine the status of an iteration point as solid, even if the centre lands on a solid. This is evident when STRTH = 0.8, where 4/5 points on a plus-shape stencil must be registered as solid. In the instances shown in Figure 17, not all elements will be surrounded by at least three other solids, causing them to be recorded as void. This issue is increasingly evident when using AL lengths larger than GS-X/Y, as seen in Figure 17b.

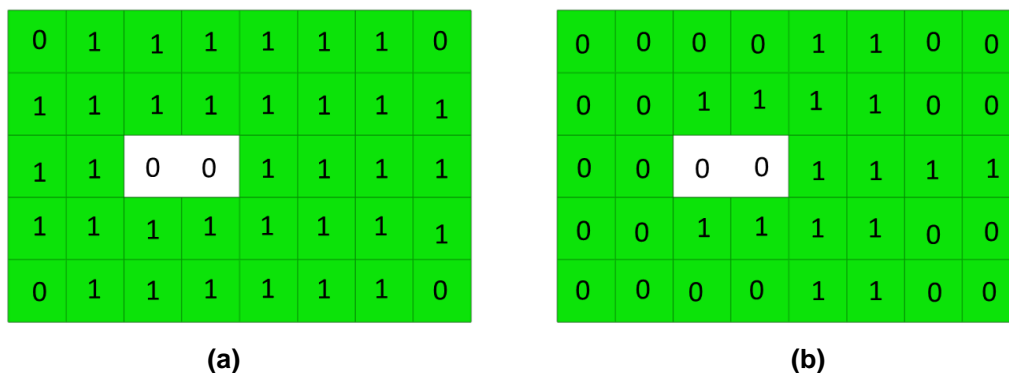


Figure 17 – (a) STRTH = 0.8 with AL = 0.5: corner iteration locations do not record areas as solids
(b) STRTH = 0.8 with AL = 1: even less locations registered as solids due to the increased AL and harsher STRTH recording. Note: 1 (Solid) and 0 (Void)

5.5.3 TC3 Results – Local Solid/Void Threshold Preference

Favouring the status of a void instead of a solid for an individual stencil search point has shown no discernible change in the results generated. This is due to all 0.3 SR results initially recording only one result in its search radius, resulting in always 100% solids or voids being detected. Results also follow the same patterns when SR is increased to 0.5 or 0.7 and multiple solid and void locations are recorded. This is due to each solid/void determination being treated on a local level as opposed to influencing the overall status at the IC. It is expected that differences between the solid/void favouring will be more evident with the global FSIC value.

5.5.4 TC3 Results – Global Solid/Void Threshold Preference

Favouring voids for the overall stencil status has shown to create elements for fewer results than equivalent model setups where FSIC favours elements. Notable inconsistencies appear when ALi is increased above the 0.5 datum value: different data is read for the stencil arms and therefore lower ratios may not be treated as solids. Similar issues arise when GS-X is larger than the 0.5 interval, as shown in Section 5.3.2: here, the misalignment has shown to be less favourable in the creation of new elements and selection of suitable void spaces to create them.

5.5.5 TC3 Results - Overview

It is shown that favouring of solid locations within the element status recordings can create elements, with less restrictions placed on the iteration steps and stencil arm lengths. Inconsistencies can also arise when the SVRTH and STRTH values are at a higher ratio than 0.6 when using a plus-shape stencil: not being able to capture all solid locations may prevent actual void locations from being correctly recorded and filled with new elements. These ratios will be tested with the alternative stencil shape configurations in TC4.

5.6 TC4 - Stencil Shape

TC4 consists of 1890 models and involves using various stencil shape designs to create elements. Models from TC1, TC2 and TC3 will be used and repeated for the two-point horizontal, x-shape and box stencils (see Figure 9, section 4.2.1). These three additional stencil shapes were used to record void and element data, with these results compared to the performance of the plus-shape stencil results from the previous test cases. Table 12 highlights these stencil variations.

	Variation Name	Results Produced
SS	1 (2-Point Horizontal)	X
	2 (Plus Shape)	Completed in TC3
	3 (X-Shape)	X
	4 (Box Shape)	X

Table 12 – TC4 Parameters

5.6.1 TC4 Results – 2-Point Horizontal Stencil

The 2-point horizontal stencil has shown to create less elements per setup for the parameter variations used in TC1, TC2 and TC3. Key inconsistencies occur when increasing the AL to a value larger than the GS-X value or an average element size and when increasing the SR to a value larger than the average element size (SR = 1.4). This can be due to the lack of information read by the two stencil points accounting to several inaccuracies when determining solid-void status of an iteration point. This inaccuracy is additionally more noticeable when increasing the SVRTH ratio value; the higher the value the less number of solids are recorded.

5.6.2 TC4 Results – X-Shape Stencil

Using an X-shape stencil has shown to create elements for mostly the same parameter variations as with the plus-shape stencil. Generation of elements is shown to be less consistent when increasing the SVRTH and STRTH values above 0.6. This can potentially be contributed to the configuration of the X-shaped stencil, in which the solids/voids recorded by the ALi points may not be connected to the IC solid/void along a leading edge.

5.6.3 TC4 Results – Box Shape Stencil

Similarities are present with the models that generate elements for the plus and x shape stencil with those from the plus-shape stencil. Some inconsistencies arise from increasing the ALi length greater than the GS-X iterations as well as increasing the SR value to larger than GS-X, resulting in less elements created. This stencil shape also encountered difficulties to identify and create elements for higher SVRTH and STRTH values, specifically when equal to 0.8. This can be contributed to the increased number of search points resulting in recording a greater number of voids. This increase number of voids will cause a much lower ratio being needed to for SVRTH and STRTH to determine the presence of solids.

5.6.4 TC4 Results – Overview

It is shown that a plus-shaped stencil ($SS = 2$) provides greater flexibility when creating elements for various ALi and GS-X lengths, as well as when modifying individual and overall stencil solid-void ratios. Inaccuracies and misrepresentation of iteration status is more likely with the two-point horizontal stencil ($SS = 1$), causing results to not create as many elements under the same parameters provided when SS equals 2. An x-shape stencil ($SS = 3$) has also provided levels of inaccuracy due to the stencil point positions not being close to horizontal and vertical edges of each element. Finally, it has shown the plus and x stencil ($SS = 4$) can be considered “over-constrained” due to the reduced number of models creating new elements compared to $SS = 2$. It is recommended that a plus shape stencil is used when refining topology solutions over the other shapes. This is due to the lack of increased accuracy of elements created when using more complex SS than the plus shape, as well as the potential increase in computational time needed to compute data recorded by more stencil points. It is required that the least amount of data is recorded to not overcomplicate this run, in which the plus-shape stencil proves to be the most suitable.

5.7 TC5 – Stencil Arm Length vs. Iteration Length vs. Deleted Member Size

TC5, alongside TC6, focus on the deletion of elements using the APP's element deletion feature. As element deletion uses the same stencil methodology to delete elements as to create them, it is expected that result generation would follow a similar pattern to that of TC1-4. Nonetheless this process will be tested by modifying the same parameters used within TC1 and TC2. TC5 includes 26 models and is comparable to TC1 by modifying element member size, GS-X and ALi (X) and aims to determine whether similar trends are established for element deletion and creation. TOMs 4, 5 and 6 are used for this TC, with Table 13 outlining the parameters used for these models.

		GS-X		
		1.0	2.0	4.0
ALi (X)	1.0	X	X	X
	2.0	X	X	X
	4.0	X	X	X

Table 13 – TC5 parameter variations

5.7.1 TC5 Results – Arm Length Variation

Detection of solids and voids when modifying ALi shows strong similarities with result correlation when compared to element creation from TC1. This similarity in the trend of results which delete elements is because the process of detecting solid-void status is the same regardless of whether elements are created or deleted. As concluded from TC1 it is desirable that the ALi value is closer to the average element size and the iteration increments. This is such that data is not either repeated or skipped when the stencil iterates across.

5.7.2 TC5 Results – Grid Spacing Variation

Increasing the GS-X value so that it is greater than the default ALi and average element length has proven to not detect nor delete any elements within the structure. It is expected that this is due to the DMD parameter being considered, in which a suitable element to delete is determined by whether any elements are immediately to the left of the located element within the DMD distance. If this value is left unaltered, the iterations are solely reliant on landing close to the left-most element in the structure or else no elements in that structure will be deleted. It should be noted that the left-most element is detected due to the iteration order (left to right, bottom to top). It is possible that variations of these element deletion results will occur from those produced in TC5 if this iteration order (and DMD detection) is re-defined.

5.7.3 TC5 Results – Deleted member Size Variation

As with the MHS-X variable (which is constant for TC1), assuming MMS-X is larger than the member size to be deleted, the elements will be removed from the model. This is however also subject to the solid-void status determined with the different ALi and GS-X values. If the solid sections are not initially recorded as solids, they will not be removed from the model. This is the case when ALi and GS-X are larger than the average element size.

5.7.4 TC5 Results - Multiple Variations and Conclusions

As with TC1, GS-X is shown to influence the deletion of elements the most: when it is larger than the average element size no elements are correctly deleted (i.e. under the parameters where the dimensions of the model are smaller than MSS-X and MSS-Y). If GS-X stays at its default value of 1.0 however, elements are correctly deleted for all member sizes and ALi lengths. This is due to the stencil correctly landing close to element centres and determining its desired solid-void status from this. Additionally, the solid and void data recorded by the stencil is below the STRTH threshold. As the models used are relatively simple, this ratio is easily achieved, with more complex test cases required to ensure this stays consistent. Further work can also be done to counter any incorrect ratios by modifying each individual stencil arm length as opposed to only X and Y lengths. These are two considerations that will be addressed in future work.

5.8 TC6 – SR and DMD Variation for Element Deletion

Similar to the element creation parameters used in TC2, TC6 consists of 188 models and will focus on identifying what affect changing the SR value has on the detection of solids and voids and, consequently, whether elements will be correctly deleted. In addition to this, the user-defined parameter for element deletion, DMD, will be modified. This relates to the distance a single detected solid should be from another solid, such that it does not incorrectly delete elements attached to larger structures. Table 14 presents these parameter variations and associated model numbers.

		SR		
		0.6	1.0	1.4
ICINCL	1 (IC included)	DMD = 0.5, 1.0, 2.0	DMD = 0.5, 1.0, 2.0	DMD = 0.5, 1.0, 2.0
	2 (IC not included)	DMD = 0.5, 1.0, 2.0	DMD = 0.5, 1.0, 2.0	DMD = 0.5, 1.0, 2.0

Table 14 – TC6 Parameters

5.8.1 TC6 Results – Search Radius Variation

Determination of solid-void status through SR variation follows the same trend established in TC2. Solid-void status generally stays consistent even when SR is significantly larger than the average element size or GS-X iteration spacing. In the case of element deletion, if an IC position lands on an element centre but is registered incorrectly as a void, the element status will determine that this area should be considered void. It will therefore flag this iteration position to ensure no elements will be present after the run. Overall, elements will be deleted regardless of their initial stencil status as they are all below or equal to the minimum member size, MMS-X, value.

5.8.2 TC6 Results – Deleted Member Distance Variation

Increasing or decreasing the distance a solid should be from the main structure has provided little change with the overall result correlation but may be a feature to be kept close attention to. Whereas a similar pattern is provided to those elements deleted from TC5 i.e. GS-X ~ ALi, several inconsistencies are present in how these elements are deleted when DMD is smaller than GS-X or an average element length. For instance, when DMD is less than GS-X (e.g. 0.25) the element deletion program will not detect any nearby elements to the immediate left (due to iteration process – see Section 5.7) of the initial solid located. This can cause elements that are connected to other elements to their immediate left to be selected for deletion, additionally causing these elements to be recorded multiple times at different iteration positions. This issue is illustrated in Figure 18. It can be suggested that a guideline should be established where DMD must be larger than or equal to GS-X to prevent repeat data or too many solids from being selected and deleted, respectively.

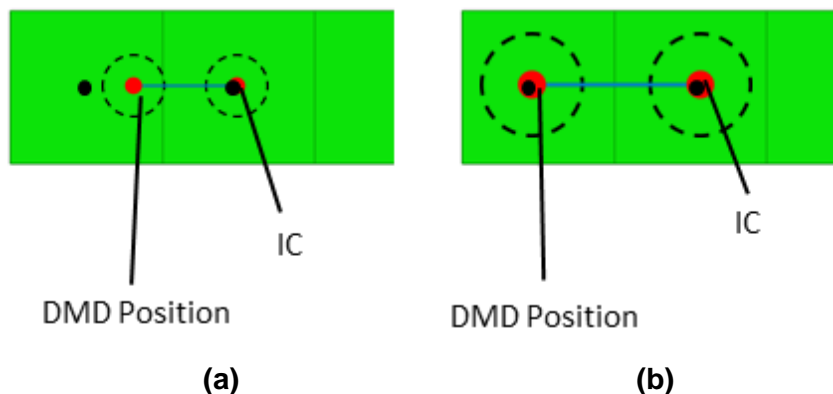


Figure 18 – (a) If DMD is smaller than GS-X, there is an increased likelihood that no solid will be detected, therefore allowing the element at IC to be incorrectly deleted (b) Both the IC and DMD positions record elements as DMD is close in size to GS-X, therefore no element is incorrectly deleted

5.8.3 TC6 – Parameter Combinations and Conclusions

Whereas the results for the modified parameters mirror those produced by TC2 it is important to consider the additional distance of an element from a structure, DMD. If this value is too small multiple elements that would otherwise be disregarded will be deleted, removing too much of the topology. This in combination with the use of GS-X spacing close to the average element size should be considered when using the APP on more complex topologies.

5.9 TC7 - File Write Switch

TC7 consists of 268 models with identify the effect the element write order has on the creation of elements. As described in Table 5, Section 4.2.5, the APP has two file write switches that update the model either after all iterations have passed (Post-Iteration - PI) or between each iteration (Mid-Iteration - MI). With PI used for the previous test cases, MI will now be introduced to the setups previously outlined in TC1-TC4. Identification of whether changing between MI or PI will affect the topology final solution will be addressed, specifically whether the same number of elements are created for a single model. Table 15 details the parameters used for TC7.

	Variation Name	Results Produced
MI/PI	MI	X
	PI	Completed in TC1-4

Table 15 – TC7 Parameters

5.9.1 TC7 Results – Stencil and Iteration Modification

Modification to the stencil parameters outlined in in TC1 (GS-X, ALi (x)) show no discernible deviation of results when filling varying enclosed void space sizes with new elements. Elements are still not able to be created when GS-X is much greater than ALi (x) or the average x-element length (see Section 5.3). It is understood that elements are able to be created one per grid point as the void size decreases when a new element is added to a void. This means that the void will become smaller over the iteration steps and not violate the MMS-X value (see Figure 19).

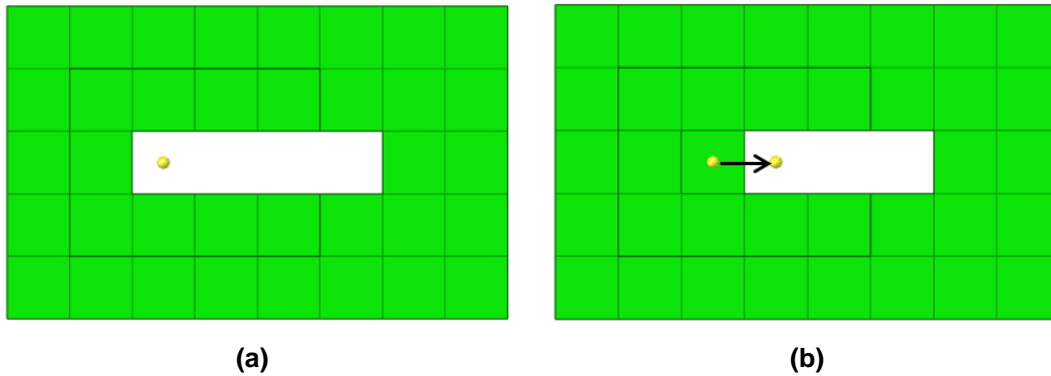


Figure 19 – (a) void size of 4.0x1.0 specified is suitable to be filled with elements (b) after moving to the next iteration point, the void size is reduced to 3.0x1.0, which will also be filled

5.9.2 TC7 Results – Solid–Void Ratio

Filling in voids by modifying the solid-void ratio parameters (SVRTH, STRTH, FSSP, and FSSIC) shows no change when compared to the PI results from TC3. This can be expected as the general detection process of suitable voids to fill does not differ between PI and MI variants. Assuming this consistency, the creation of elements follows the same trends identified within TC3 (Section 5.5).

5.9.3 TC7 Results – Stencil Shape Modification

When using the two-point, x-shape, and box-shape stencils, no discrepancies were present between the previous PI runs from TC4 and those using the PI element creation. This again is expected due to the detection parameters for each stencil not differing depending on whether MI or PI is used. Additional studies may need to be considered for the creation of multiple elements in the y-axis to ensure this trend is consistent. For the purposes of filling an x-axis series of voids, no difference between MI and PI is present.

5.9.4 TC7 Results – Overview

Modification to MI and PI for element creation shows no inconsistencies between parameter variations. This is true for the creation of elements in the x-axis but may require further validation when using more complex topologies.

5.10 TC8 - Varying x/y Hole Size

TC8 consists of 209 models relating to the creation of elements for varying size enclosed voids. In addition to the consideration of writing elements pre- or post-iterations (TC7), identification of how these write parameters influence the creation of elements for varying void sizes is to be looked at. Consideration as to how the APP determines whether an element is created for varying x and y length voids is to be conducted for TC8, in which the void sizes shown in TOMs 7-13 in Table 8, Section 5.1, will be tested under the parameters displayed in Table 16. The combinations of these parameters that create suitable elements will be noted, with the key consideration of what constitutes a hole to be filled to be discussed.

		MHS-X			
		1.0	2.0	3.0	4.0
MHS-Y	1.0	MI and PI	MI and PI	MI and PI	MI and PI
	2.0	MI and PI	MI and PI	MI and PI	MI and PI
	3.0	MI and PI	MI and PI	MI and PI	MI and PI
	4.0	MI and PI	MI and PI	MI and PI	MI and PI

Table 16 – TC8 Parameters

5.10.1 TC8 Results – Post-Iteration Element Creation

The PI variant of the APP element creation can create elements to fill voids for TOMs 7-13 when using suitable MHS-X/Y values (i.e. larger than the void member length). These suitable values can be identified as being equal to or greater than the unit void size in the respective x or y axis. An example of this can be when filling in the 1.0x1.0 unit size void in TOM 7 in Table 8 (Section 5.1): if the MHS-X and MHS-Y values are equal to or greater than 1.0, elements will be created for this void shape. If either MHS-X/Y is smaller than the void size but the other MHS-X/Y value is equal to or greater, no elements will be created, with elements only being created if both values correspond to the suitable element length, as there is no direct link between MHS-X and Y, i.e. they are not influenced by each other's values. This is also the case when looking into more complex shapes such as the T-Shaped void (TOM 10) and L-shaped void (TOM 13), in which elements will only be created if the MHS-X/Y values are equal to or greater than the longest leading edge of the void in that respective axis (for TOM 13, MHS-X = MHS-Y = 1.5 (3 element lengths)). Material checkerboarding is also able to be completely removed from a structure (TOM 11) with elements created to fill all enclosed void spaces when MHS-X/Y is greater than or equal to the void's length.

Whereas addressing the issue of removing checkerboarding is highly beneficial, questions can be drawn as to whether the filling in of voids that are not entirely symmetric is correct or not. For instance, the L-shaped void may be referred to as a fillet for certain manufacturable

designs and might not be intended to be a feature to remove or fill with elements. It may also be required that the L-shape is only partially filled in and that the APP should be adapted to account for these shape configurations. It can therefore be suggested that further testing is needed to identify what features the APP should consider when creating elements, specifically what constitutes a hole and what is a non-removable feature. This will ideally be explored in further research on more complex test cases and practical manufacturable topological designs.

5.10.2 TC8 Results – Mid-Iteration Element Creation

Filling voids of varying shapes using MI element creation generally follows the same pattern as that of the previously mentioned PI procedure. Elements are only created when the MHS-X and MHS-Y values are above or equal to the longest length of the void to be filled. If either value is lower, no element will be created for this void shape. A single exception occurs when dealing with voids that act by design as two intersecting thinner sections, such as the T-Shape identified in TOM 10. Due to the MI process, the stencil data is only stored for the iteration position the stencil is currently on, as opposed to saving this data into database to be read after the iteration process (PI). Because of this, elements are created based on the status of the current iteration position (solid/void) and whether any solid/void statuses are changes for the previous iterations. This can cause more complex shapes to not include some positions as elements to be created despite them being smaller or equal to the maximum void size. Figure 20 illustrates this discrepancy. This can then bring the question as to what constitutes a void that should be filled with an element. Should, for instance, the T-shape structure in Figure 20 be considered a void feature that should be filled with elements or should element creation be limited to only more rectangular or circular shapes? Furthermore, should the irregular circular shape illustrated in Figure 3, section 2, be considered a void to fill even if it is generally open and circular but has varying edge angles? Additionally, it may be worth considering how these features are filled in more complex test cases and establishing a series of rules as to what should or should not be filled in. These rules can be defined through a series of pre-existing data already found and recorded in the APP, such as the number of nodes that constitute a leading edge, various combinations of stencil shapes or placing further restrictions on the solid/void ratios. These questions will be addressed in future development of the APP.

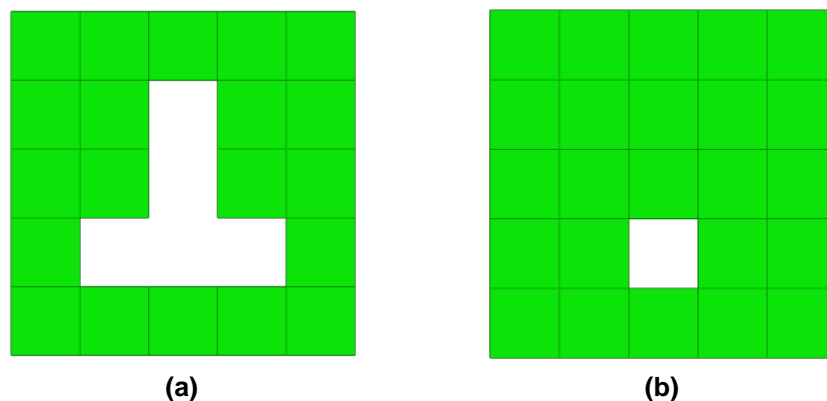


Figure 20 – (a) T-Shape topology design (b) APP element creation with MI and MHS-X/Y = 1.5 (3 element lengths)

5.11 Application and Validation of Recommended Settings

After the identification of recommended parameters to remove unwanted topological features, the ability to perform these refinement techniques on larger, more complex structures is considered. This section will overview the results correlated from the previous sub-sections and determine which of the recommended parameters can be used when adding or deleting elements for more complex designs. Specifically, the inclusion of both adding and deleting elements will be considered within one component, with multiple features such as material checkerboarding, various hole sizes and disconnected elements being present in a single model.

From the results identified in sections 5.3 - 5.10, identification of several user-defined parameters can be made. These recommended parameters should be considered when attempting to create a refined topology solution that successfully removes all small holes and disconnected elements. A summary of these parameters is outlined in Table 17.

Variable name	Recommended Value	Suggested Values for Half Bridge
SS	Plus-shape	Plus-shape
ST	Fixed ALi length	Fixed ALi length
AL _i	AL1 = AL2 = AL3 = AL4 ~ average element length	AL1 = AL2 = AL3 = AL4 = 5
AL _i _max	N/A	N/A
SR	SR1 = SR2 = SR3 = SR4 ≤ AL _i	SR1 = SR2 = SR3 = SR4 = 3
SPSR	Equations (2), (3), (4) and (5) – No other variations available	Equations (2), (3), (4) and (5)
ICSR	Equations (2) and (3) – No other variations available	Equations (2) and (3)
GS-X/Y	~ average element length in respective X/Y coordinates	X = 5, Y = 5
MI / PI	PI	PI
MHS-X/Y	≥ X and Y lengths of the longest edge of an enclosed void	X = 15, Y = 15
MSS-X/Y	≥ X and Y lengths of the longest edge of a solid to delete	X = 5, Y = 5
DMD	~ average X element length	5
SVRTH	0.6 – for Plus-shaped stencil	0.6 – for Plus-shaped stencil
STRTH	0.6 – for Plus-shaped stencil	0.6 – for Plus-shaped stencil
FSSP	Favour solids	Favour solids
FSIC	Favour solids	Favour solids
ICINCL	IC included in ratio	IC included

Table 17 – Recommended variable settings for element creation and deletion using the APP

The recommended settings derived from the results of TC1 – TC10 can be used as suggested inputs for larger, more complex models. As an example, Figure 21 illustrates the setup of a loaded half bridge structure consisting of symmetric 5mm quadrilateral elements, designed to undergo topology optimisation. Several optimisation parameters have been modified in an attempt to produce a suitable refined TO structure. The objective of the optimisation is to

minimise the compliance (therefore increase material stiffness) subject to a volume constraint of 30% of the original structure. The model is run in Optistruct and therefore uses a VDM-SIMP method, creating a result as described in Figure 1b, Section 2. Table 18 highlights several variations of these optimisation parameters. The modified features include changing the values MINDIM (eliminates members smaller than a specified size), CHECKER (controls checkerboard-like element distributions, with the effect of adding more intermediate densities) and MMCHECK (provides a checkerboard-free solution with the unwanted effect of adding many intermediate density elements). For all TO model variations shown in Table 18 the DISCRETE value of 3 is used, such that the solutions created remove as many intermediate material densities as possible, enabling a more manufacturable and discrete design. It should be noted that the different topologies produced may alleviate certain unwanted features but do not necessarily rule out the need to post-process the TO result after solving.

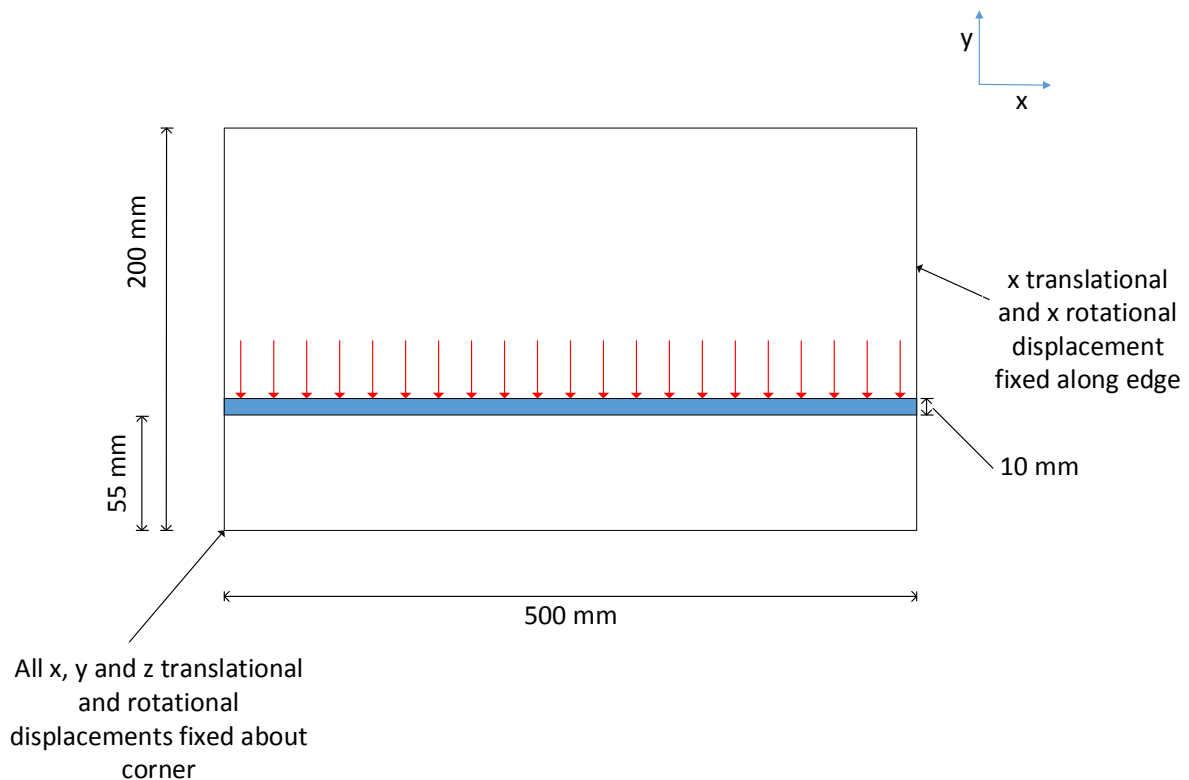


Figure 21 – Half bridge model setup prior to TO with non-design space (blue middle segment), constraints on right hand edge and left bottom corner and a uniformly distributed load applied downward along the middle segment

Model No.	Density Threshold	MINDIM	CHECKER	MMCHECK	TO Result
TO-1	0.6	N/A			
TO-2	0.6	3x elem size			
TO-3	0.6	6x elem size			
TO-4	0.6	N/A	✓		
TO-5	0.6	3x elem size	✓		
TO-6	0.6	6x elem size	✓		
TO-7	0.6	N/A		✓	
TO-8	0.6	3x elem size		✓	

TO-9	0.6	6x elem size		✓	
------	-----	--------------	--	---	--

Table 18 – TO result variations for Half Bridge model

The TO generated solutions from Table 18 suggest significant differences in the optimum structure when applying different optimisation parameters. For example, TO-1 highlights several features that can be removed using the APP, such as many examples of material checkerboarding and disconnected members. TO-3 and TO-6 indicate TO results that represent more ideal structures that will require less work from the APP, due to the lack of variable densities, small size holes, material checkerboarding or floating elements. The reader should be made aware that the APP will be able to smooth the edges of a TO solution by creating elements along jagged edges in future developments (see Section 6). There are some TO model variations that still present the need to use the APP however, such as when using a MINDIM of 3 times the element size as shown in TO-2 and TO-5. As highlighted in the circled regions labelled “A” in Table 18, several small holes are present within TO-2 and TO-5 which would be filled when run through the APP. Additionally, use of the MMCHECK parameter has shown to generate more intermediate densities in the TO solution, as shown in solutions TO-1 and TO-7. It is shown from the results in Table 18 that even when applying the variations of optimisation parameters there is still the chance to create small holes which may eventually need filling. In an attempt to avoid generating small holes, inclusion of parameters such as MMCHECK can be used to create a more fully connected structure. This may however inadvertently create more intermediate densities within the TO solution, which would in turn increase the difficulty of deciding on a density threshold i.e. which elements should be kept and which should be removed. These potential issues can be corrected using the APP. Overall the TO results from Table 18 identify some solutions that may not need to use the element creation or deletion (step 1-2) of the APP, although the APP’s modular approach to consider additional features such as discretising the solution (step 1-1) and potentially smoothing edges (Section 6) can highlight an overall necessity for the APP.

The results from Table 18 outline several variations of TO solutions for the half bridge setup defined in Figure 21. TO-1 has shown to include several individual topology features that the APP should be able to refine, including checkerboarding, small holes and disconnected elements. This model will therefore be run through the APP in order to identify its capabilities, with the model shown in Figure 22.

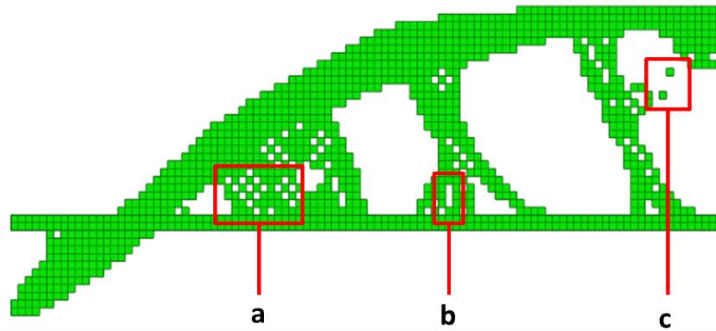


Figure 22 – Half bridge model tested which includes: (a) material checkerboarding, (b) small holes and (c) disconnected elements

This model will be subsequently run through the APP using a set of parameters based on the recommendations in Table 17. Assuming the half bridge uses a uniform mesh with symmetric elements of 5mm x and y lengths, the following suggested values for each variable in Table 17 can be chosen, as highlighted under the column “Suggested Values for Half Bridge”.

As discussed in Section 5.5.5 it is recommended that both the local and global solid/void ratios should be set to 0.6 or lower in order to capture the correct locations where elements are created or deleted. Due to the complexity of certain hole shape configurations it is desired that this value is modified close to the 0.6 ratio recommendation to ensure for its consistency in this more complex topology. Assuming features such as MHS-X/Y, MSS-X/Y and GS-X/Y remain consistent due to the equal element spacing, three variations of the parameters tested in Table 17 were run, with variations only present for the SVRTH and STRTH parameters. Table 19 outlines these parameter changes and overviews the refined designs.

Test Name	Variable	Model	Result
HB1	SVRTH = 0.4 STRTH = 0.4	Half Bridge	All holes below threshold filled. All floating elements removed
HB2	SVRTH = 0.6 STRTH = 0.6	Half Bridge	Some holes below threshold still present. All floating elements removed
HB3	SVRTH = 0.8 STRTH = 0.8	Half Bridge	Several holes below threshold not filled. Some floating elements still present

Table 19 – Variable setup and result summary for half bridge model

Conclusions determined from the results in Table 19 indicate several unrefined features are present (i.e. leftover material checkerboarding) in tests HB2 and HB3 when using the recommended (or higher) SVRTH and STRTH thresholds from Table 17. It was found that by lowering the SVRTH and STRTH thresholds to those in test HB1, the half bridge structure is able to have all small hole features and disconnected elements removed, as requested by the user (see Figure 23). Unlike the previously determined threshold of 0.6, the complexity and variety of the hole features require a lower threshold to ensure certain sections are filled with new elements. This claim is also justified through the increased number of inconsistencies present when increasing the SVRTH and STRTH values to 0.8. Therefore, along with the recommended settings from Table 17, alongside appropriate adjustments to SVRTH and

STRTH values, it is clear the APP has the capability to refine a topological structure with multiple features to remove (checkerboarding, small holes and disconnected elements) for a single model.

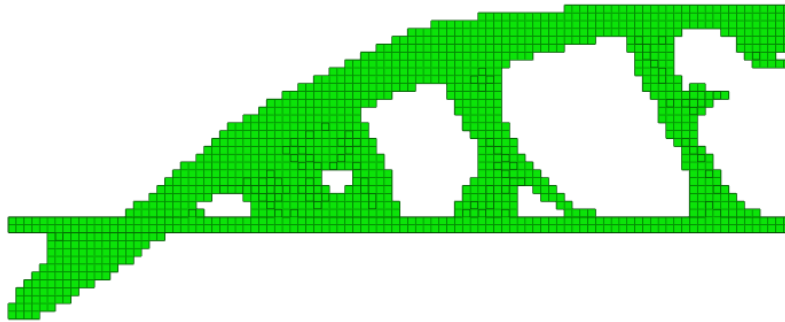


Figure 23 – HB1 setup with SVRT and STRTH = 0.4. Note all material checkerboarding, small holes and disconnected elements are removed

When relating the HB1 solution to the TO solutions highlighted in Table 18, several issues which are present in several of the TO models such as material checkerboarding, disconnected elements and small member sizes have been removed in HB1. This also rivals designs such as TO-6 in Table 18, which used a different set of optimisation parameters to the baseline model of HB1, TO-1. The ability of the APP to produce a suitable refined solution similar to the more suitable model produced from Table 18 shows potential for the APP to potentially “guide” TO solutions to a more refined design. Additionally, existing software such as Altair HyperWorks have features such as OSSmooth which can take an optimisation solution (such as TO-1) and import the structure in a more refined manner than using manual approaches. These include the use of features such as “connection detect” which is designed to take ambiguous locations such as disconnected members and create connections wherever the software deems necessary. This process could perhaps create more issues such as increase checkerboarding by connecting disconnected elements into thin, non-manufacturable members. It also returns back to the question of what should warrant the need to delete elements and what is classified as a disconnected member or a hole. Figure 24 illustrates the effect of the connection detect feature in Optistruct when applied to TO-1 from Table 18. Here, it is evident that checkerboarding is still present, although the top right hand section is shown to be connected with a new member. This new connected member is not present in the APP solution in Figure 23 and has created a member which would be too thin to have been kept in when using the APP under the provided parameters in Table 17. It should be considered within further work that the APP is compared to methods such as OSSmooth in order to ensure which process is most suitable and whether the APP can be adapted to produce more structurally suitable designs. It should be noted that the APP is able to not only create elements but also delete members all subject to a series of user-defined parameters. This provides greater user “freedom” over the control of what should be kept and added within the TO structure, as opposed the OSSmooth only adding elements to prevent disconnections. One focus of further experimentation should be to compare these two approaches to see which refined solutions are more suitable.

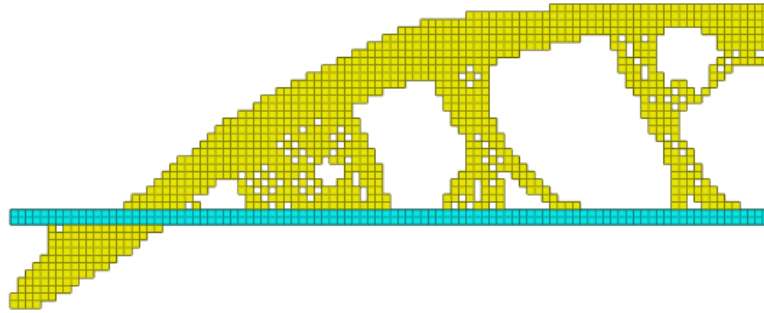


Figure 24 – OSSmooth variation of TO-1 solution with new member created in top right corner

6 Conclusions & Further Development

Following the results discussed in Section 5, it can be concluded that the inclusion of a mesh-independent semi-automated post-processor has allowed for the creation of refined structures that are tailored to user input specifications. The ability to consistently and systematically generate a refined solution that closer resembles a manufacturable design has allowed for a reduction in manual steps and therefore an increased level of automation for the post-processing of topology optimised designs. This step also provides a methodology to an otherwise unruly process and aims to reduce time spent by a user in determining a suitable manufacturable design. Step 1-1 of the APP is able to manipulate an input variable density solution and create a binary variant of this, generating a discretised design which is easier to interpret. Step 1-2 of the APP then uses a mesh-independent process to refine this binary solution by either adding or removing material (elements), in which the user has considerable control over the level of addition/removal of material through the selection of pre-set parameters. The customisability of the search stencil allows for the user to provide their own desired level of refinement. Modification to features such as the desired ratio of solids/voids for a specific region, as well as the sizes of voids to fill and members to delete, has aided to this user freedom. From the results obtained in the case study outlined in Section 5, several guidelines can be provided to an end user for the appropriate parameters to use when refining a topology optimised design. It is recommended that a plus-shaped stencil is used in order to more accurately record solid/void status for each iteration position, as well as ensuring that a near 1:1 ratio is established for the stencil arm (AL_i) length and the iteration spacing ($GS-X/Y$), if possible. This will ensure all solid and void data is recorded and a more accurate visualisation of the structure is recorded by the APP. Assuming the provided topology has mostly symmetric elements, a recommended solid/void ratio to determine the solid/void stencil status is 0.6, i.e. three out of five stencil search points must be registered as solid for the location to be considered a solid. When considering more complex structures that involve the filling in of various hole shapes, a lower threshold, such as 0.4, might be better suited for element creation (see section 5.11). It is also recommended to use a PI element creation method as the APP will read solid/void data on a global topology design as opposed to only locally searching when using mid-iteration methodology. A list of these recommended settings is provided in Table 17 (section 5.11).

Further Development

The ability for the APP to successfully fill in void shapes that are not square or symmetric shows its potential to be used for the refinement of more complex, industry-standard topology optimisation models. Whereas the APP provides a unique approach to post-processing topology optimisation results, some additional developments may be implemented to improve the its refinement features. Some of these considerations can include the following:

- Application of an adaptable stencil with AL_i values that dynamically update during a single iteration step and comparing this to the existing APP methodology (see Section 4)
- Edge smoothing
- Decreasing CPU run time of the APP
- Establishing a series of rules for the APP in determining what constitutes a hole and which enclosed void geometry shapes require elements to be created
- Test case generation on more complex topologies similar to industry-standard designs, with the inclusion of multiple void shapes and structures to delete, as well as significant element distortions, within a single model
- Application of APP for 2D models in 3D geometric space or PP of full 3D models
- Application of APP to non-meshed TO solutions
- Generation of a CAD result using NURBS curves

It is recommended that the considerations made in Table 17 are applied to the APP to ensure for a relatively seamless process of inputting an optimised solution and generating a manufacturable component design. The test cases outlined in section 5 have identified several correlations between individual parameters, most notably the establishing of a near 1:1 ratio between average element lengths and the $GS-X$ and AL_i parameters. The solid/void ratio provided must be in relation to the type of stencil used (i.e. $3/5$ for a plus-shaped stencil or $1/2$ for the two-shape horizontal). Additionally, the APP can create elements for structures of varying X and Y dimensions for an enclosed set of voids. Element generation is more consistent when using suitable $MHS-X/Y$ and $MSS-X/Y$ parameters when using a post-iterative (PI) creation rule, as opposed to mid-iteration (MI).

As outlined in Section 5.11, the APP is able to refine a TO solution which reduces checkerboarding and disconnected elements to a similar level when using varying optimisation parameters in existing solvers. Both results however do not create structures with fully smooth edges, a feature which can be applied to the APP in future updates. Here, the APP can create tria elements, for example to fill right angle edges, thus reducing stress concentrations at these locations.

Computation time is relatively fast (several seconds) when running these test case models but is expected to increase with model complexity. Currently, PI is more favourable in terms of CPU time due to the smaller number of element write steps performed during this process. Desired following procedures will therefore involve testing the APP on more industry-standard models and considering its cooperation alongside various commercial optimisation solvers. This includes improving the performance of the APP such that it can confidently refine irregular meshes and element distortions. Further testing will be made on industry-standard designs with more severe element distortions than those shown in the previously tested half bridge and c-clip models. If needed, the APP methodology will be updated in order to allow for these model variations.

Modification into the grid spacing (GS-X/Y) to include varying iteration distances may also be considered to accompany these irregular grid shapes. This can be in the form of an extendible stencil, in which the individual AL_i lengths of the stencil dynamically change for each iteration position. These updated parameters can also aid in the understanding of what should be classified as a hole to fill by the APP and potentially avoid element creation issues such as those found in the T-shape hole (Section 5.10.2).

Consideration should also be made to ensure CPU time is lower than that of a comparable manual refinement, in which additional focus groups may be required to trail the APP's performance. It is desired that this further work will be presented with more complex test cases established for these new scenarios. Additional consideration would also be made to the types of models in which the APP can refine. These include not limiting the APP methodology to only solve 2D TO solution designs and allow this process to be applied to 2D TO solutions in a 3D environment, as well as refining full 3D components with 3D meshes. It is desired that the APP stencil will need adapting for 3D applications, typically by applying additional stencil arms and search points as well as adapting the iteration process to move in a 3D environment.

The methodology described by the APP is intended to be mesh-independent, in which it should not be limited in its method to refine only discretised mesh solution files. It is expected in future updates that the APP will be able to identify features and create or delete material for continuous CAD surfaces using the same principles. It is expected that there will be some potential challenges when integrating the APP to non-meshed solutions in that different sets of data will need to be recorded in order to determine the solid/void status at a specific iteration. The general methodology should however follow the same process outlined in the APP so general integration to non-meshed models should provide little challenge. Furthermore, as well as being able to refine CAD solutions it is ideal that the APP can generate a CAD solution from the refined TO PP solution. This will directly address steps 2 and 3 of the APP methodology and create a fully seamless and automated approach to generating manufacturable designs from discrete TO solution files.

7 References

- [1] Altairhyperworks.com. (2019). Simulation Software for Stamping and Tooling Industries| Altair HyperForm. [online] Available at: <https://altairhyperworks.com/product/HyperForm> [Accessed 23 Jan. 2019].
- [2] Challis VJ (2009) A discrete level-set topology optimization code written in Matlab. *Struct Multidiscip Optim* 41(3): pp.453–464
- [3] Hsu, M.-H. and Hsu, Y.-L. (2005), Interpreting three-dimensional structural topology optimization results. *Computers & Structures*, Vol. 83 No. 4, pp. 327-337.
- [4] Hughes TJR, Cottrell JA, BazilevsY (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comput Methods Appl Mech Eng* 194(39–41): pp.4135–4195
- [5] Jiang, L. and Chen, S. (2017). Parametric structural shape & topology optimization with a variational distance-regularized level set method. *Comput Methods Appl Mech Eng* , 321, pp.316-336.
- [6] Kang P, Youn S-K (2016) Isogeometric topology optimization of shell structures using trimmed NURBS surfaces. *Finite Elem Anal Des* 120: pp.18–40
- [7] Kang, Z. and Wang, Y. (2013). Integrated topology optimization with embedded movable holes based on combined description by material density and level sets. *Comput Methods Appl Mech Eng* , 255, pp.1-13.
- [8] Koguchi, A. Kikuchi, N. (2006), A surface reconstruction algorithm for topology optimization. *Engineering with Computers*, Vol. 22 No. 1, pp. 1-10.
- [9] Larsen, S. and Jensen, C.G. (2009), Converting Topology Optimization Results into Parametric CAD Models. *Computer-Aided Design and Applications*, Vol. 6, pp. 407-418.
- [10] Lee, S., Jeong, M., Kim, B., Lee, S. and Lee, S. (2012). Die shape design of tube drawing process using FE analysis and optimization method. *The International Journal of Advanced Manufacturing Technology*, 66(1-4), pp.381-392.
- [11] Lin, C. and Chao, L. (2000). Automated image interpretation for integrated topology and shape optimization. *Struct and Multidiscip Optim*, 20(2), pp.125-137.
- [12] Lin, H., Maekawa, T. and Deng, C. (2018). Survey on geometric iterative methods and their applications. *Computer-Aided Design*, 95, pp.40-51.
- [13] Liu J, Ma Y-S (2015) 3D level-set topology optimization: a machining feature-based approach. *Struct Multidiscip Optim* 52(3): pp.563–582

- [14] Lovadina, C., Reali, A. and Sangalli, G. (2014). What is Isogeometric Analysis?. [online] Sintef.no. Available at: <https://www.sintef.no/globalassets/upload/ikt/9011/geometri/terrific/wmf/part1/2-lovadina.pdf> [Accessed 28 Mar. 2019].
- [15] MachineMfg.com. (2019). *Mechanical Properties of Sheet Metal Materials (Complete List) MachineMfg.com*. [online] Available at: <https://www.machinemfg.com/sheet-metal-mechanical-properties/> [Accessed 5 Nov. 2019].
- [16] Mathworks.com. (2019). Image Recognition. [online] Available at: <https://www.mathworks.com/discovery/image-recognition.html> [Accessed 12 Feb. 2019].
- [17] Mathworld.wolfram.com. (2019). Circle -- from Wolfram MathWorld. [online] Available at: <http://mathworld.wolfram.com/Circle.html> [Accessed 4 Apr. 2019].
- [18] Nee, A., Ong, S., Chryssolouris, G. and Mourtzis, D. (2012). Augmented reality applications in design and manufacturing. *CIRP Annals*, 61(2), pp.657-679.
- [19] Osher S, Sethian JA (1988) Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J Comput Phys* 79(1): pp.12–49
- [20] Parvzian, J., Düster, A. and Rank, E. (2012), Topology optimization using the finite cell method, *Optim Eng Vol. 13*: pp. 57-78
- [21] Sehmi, M., Christensen, J., Bastien, C. and Kanarachos, S. (2018). Review of topology optimisation refinement processes for sheet metal manufacturing in the automotive industry. *Struct Multidiscip Optim*, 58(1), pp. 305-330
- [22] Tang, P.-S. and Chang, K.-H. (2001), Integration of topology and shape optimization for design of structural components. *Struct Multidiscip Optim*, Vol. 22 No. 1, pp. 65-82.
- [23] van Dijk, NP., Maute, M., Langelaar, M., van Keulen, F. (2013) Level-set methods for structural topology optimization: a review. *Struct Multidiscip Optim* 48(3): pp.437–472
- [24] Yew, A., Ong, S. and Nee, A. (2016). Towards a griddable distributed manufacturing system with augmented reality interfaces. *Robotics and Computer-Integrated Manufacturing*, 39, pp.43-55.
- [25] Yi, G. and Kim, N. (2016). Identifying boundaries of topology optimization results using basic parametric features. *Struct Multidiscip Optim*, 55(5), pp.1641-1654.
- [26] Zhang, S., Norato, J., Gain, A. and Lyu, N. (2016). A geometry projection method for the topology optimization of plate structures. *Struct and Multidiscip Optim*, 54(5), pp.1173-1190.