GENERATIVE DESIGN OF CATALAN VAULTS FOR MULTI-STOREY SEISMIC CONSTRUCTION

BUILDING TECHNOLOGY GRADUATION STUDIO AR3B025

FINAL REPORT

Mentors
Charalampos Andriotis (1st mentor)
Simona Bianchi (2nd mentor)
Robin Oval (3rd mentor)
Prateek Bhustali (AiDAPT research buddy)

Delegate of the Board of Examiners
Olindo Caso

Tahir Zahid Ishrat |5698928

ABSTRACT

This paper explores the scope of using a deep learning framework for shape optimization of Catalan vaults for medium seismic areas. Catalan vaults are thin tile vaults that optimize the material usage of a floor slab without form-work and thus, additional material and labour. These structures can be constructed from tiles made from locally sourced earth which can provide an alternate to steel, timber, and concrete for areas with poor access to such materials, bringing down transportation, material, and carbon costs, providing opportunity to accommodate the consequences of rapid population growth.

Seismic optimization of these vaults usually requires topology optimization and shape optimization tools. However, conventionally, these are computationally expensive and time-consuming - making them unsuitable for initial design explorations where a vast array of designs need to be quickly explored. As an alternative, a deep learning framework is explored as a design generation and optimization tool. This uses a Variational Autoencoder (VAE) trained on a dataset of 10,000 samples to extract novel meshes whose seismic performance is then predicted with the help of fully-connected dense Neural Network (NN) surrogate models trained on the results of a Linear Dynamic analysis in Karamba (in Grasshopper). An optimization loop is set-up through Gradient Descent Optimization where the gradient of the predicted score is minimized with respect to the latent space of the VAE - for single and multi-objective optimization. Conditioning the latent space of the VAE is further explored (Conditional VAE) so that the user is able to extract samples from the latent space with particular desirable characteristics such as a desirable height of the vault. This opens up opportunities to gain better control of the latent space and generate meaningful new samples that are able to incorporate user specifications. The geometry of the Catalan vault is represented in terms of polyedge force-densities that allow a 99.91% reduction in dimensionality and thus, faster convergence, as compared to other data structuring techniques explored in the literature as half-adjacency matrices.

ACKNOWLEDGMENTS

I would like to thank my mentors for their support in helping steer this project - Charalampos Andriotis, Simona Bianchi, and Robin Oval. The research dealt with three vast and complex fields easy to overwhelmed and get lost in. Without their guidance, this would not have been possible. Prateek Bhustali and Anish Diwan were able to provide valuable insights in deep learning that I may have not otherwise come across. I would also like to thank Olindo Caso, the delegate of the Board of Examiners to coordinate my P2,P4, and P5.

It’s been a privilege to have had been able to pursue this path. It would not have been possible without the support from my family - and especially my mother who has always been a beacon throughout my academic life. From sitting outside my childhood classrooms making notes for me on account of me missing school - to sending prayers from the other side of the world when I am now in college. Words can never be enough.

This journey would not have been the same without people to share it with - and it was a delight to have had it with friends. And lastly, my gratitude goes whoever invented frozen pizzas - an unsung hero for millions of students!

ABBREVIATIONS

ANN - Artificial Neural Network

BGD – Batch Gradient Descent

CNN - Convolutional Neural Network

DEM – Discrete Element Method

FDM - Force Density Method

FD – Force Density

FEA – Finite Element Analysis

GAN – Generative Adversarial Network

GCN – Graph Convolutional Network

GBF - Gaussian Basis Network

GNN - Graph Neural Network

LSP – Linear Static Procedure

MARS - Multivariate Adaptive Regression Spline Models

NMRSE - Normalized Root Mean Squared Error

NN - Neural Network

NSP – Non-linear Static Procedure

OANN - Optimized Artificial Neural Network

PSS - Particle Spring System analysis

PGA - Peak Ground Acceleration

RMSE - Root Mean Squared Error

RSA - Response Spectrum Analysis

RF - Random Forest

RL - Reinforcement Learning

ReLU - Rectified Linear Unit

SLS - Serviceability Limit State

SPGD – Stochastic Gradient Descent

SPRP - Shape-Preserving Response Prediction

SSS - Stability, Stifness, Strength

TNA - Thrust Network Analysis

ULS - Ultimate Limit State

VAE – Variational Autoencoder

WF – Workflow

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

01 / INTRODUCTION

1.1 PROBLEM STATEMENT

1.1.1 Defining the Problem

By 2050, it is projected that the world population will increase to 8.8-10 billion (Cleland, 2013). A shortage of usable land comes as a consequence of the rising population. This has resulted in a need to go vertical. In cities, this need is satisfied by the concrete, steel, and timber – materials necessary to construct the floor slab. However, in rural and peri-urban areas in the developing world, lacking easy access to these construction materials, constructing multi-storey buildings can pose challenges. In many cases in such areas, the need for multi-storey residential buildings is fulfilled by the use of reinforced concrete which comes at a high carbon, material, and transportation cost (Papanikolaou & Taucer, 2004). While reinforced concrete structures offer strong materials, inadequate construction methods can compromise safety. This problem is exacerbated in seismic regions where the heavy structure further needs to resist lateral loads. The scale of the problem in the concerned regions can be qualitatively extrapolated from the fact that there are approximately 1000 large cities in the developing world at earthquake risk (Wyss & Rosset, 2013), and due to inferior strength, rural homes are much more vulnerable to risk than urban.

To contextualize the problem, one may consider the remote valleys of the Chitral and Gilgit-Baltistan districts in the North of Pakistan. These are seismic areas, where due to risk of natural hazards from seismic activity such as rockfall, flash floods, and landslides, much of the land is not safe to be inhabited, as shown in the Figure 04. The roads that lead to many such valleys located here including Shimshal Valley are unmetalled and present a hindrance for large trucks to carry large construction materials from the city where concrete and steel are available (Butz & Cook, 2011). Freight and transport costs become exceedingly high as the material has to be supplied by smaller vehicles through numerous trips; which further increases the carbon cost as a result. Concerning the supply of wood for construction, that too presents issues. Excessive deforestation has resulted in a shortage of tress so there has been a strict enforcement of permits for cutting down trees for construction (T.Z.Ishrat, personal communication, March 2022). Furthermore, the market supply for construction timber is largely unaffordable for vast majority of the public.

To summarize, this has resulted in the need for a more affordable, resilient, and sustainable alternative utilizing local resources for floor-slab construction for rural/ peri-urban seismic areas of the developing world.

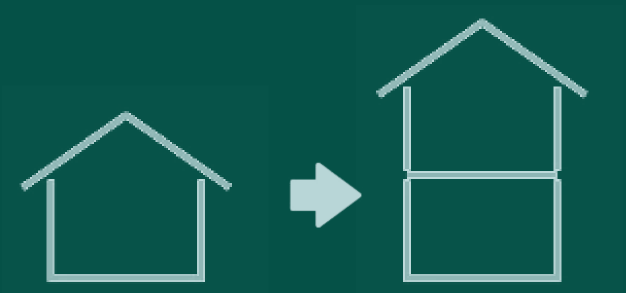


FIGURE 01: Multi-storey house. (Own Work)

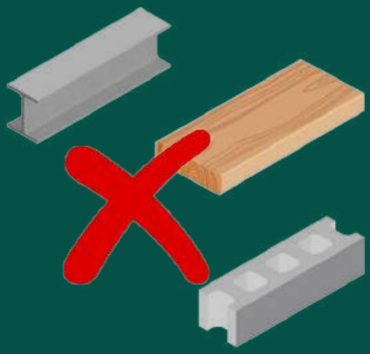


FIGURE 02: Absence of steel, concrete, timber. (Own Work)

FIGURE 03: Carbon cost and transport cost. (Own Work)

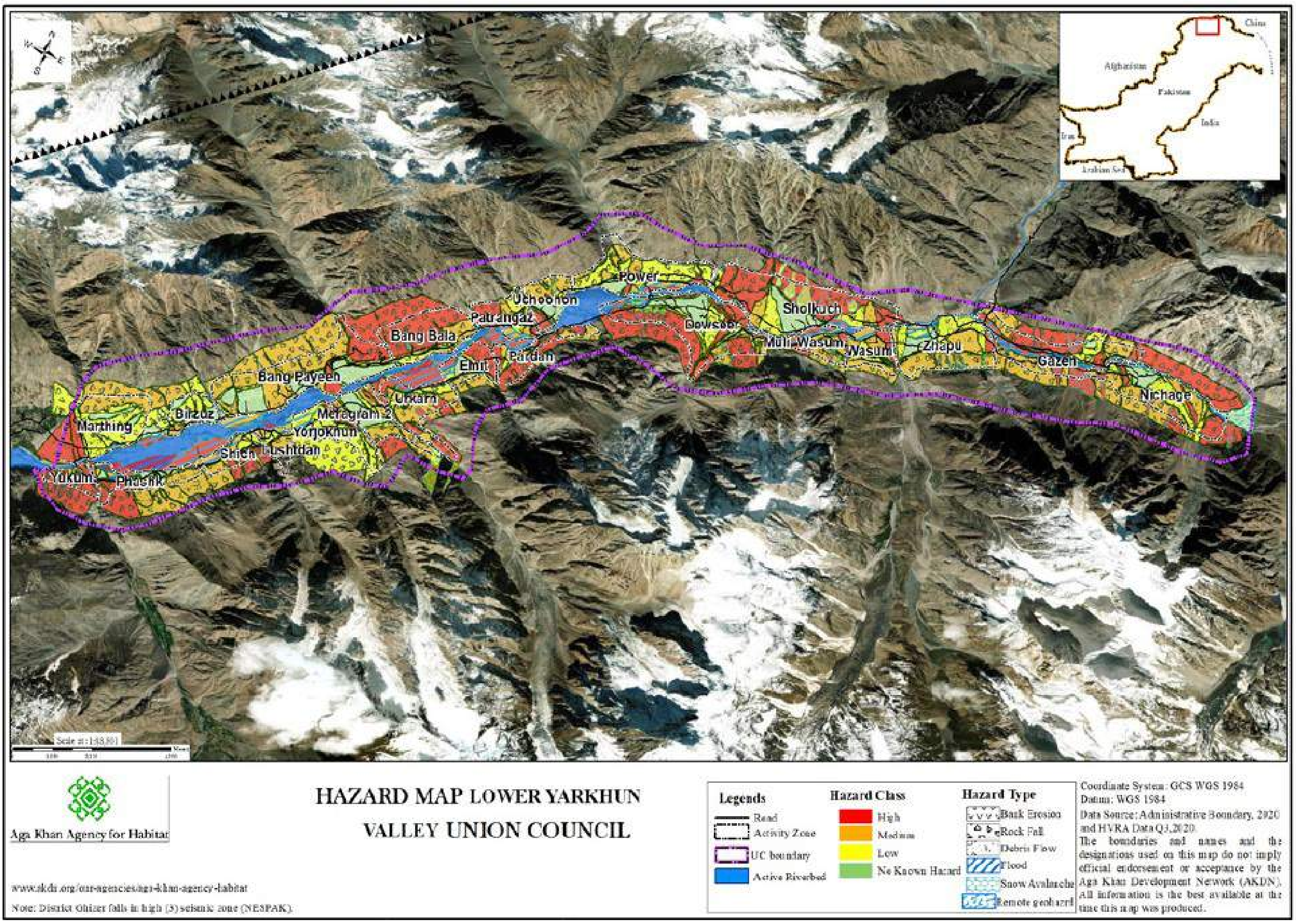

China
Afghanistan
Pakistan
India
Iran
Arabian Sea
Power
Uchoohon
Patrangaz
Bang Bala
Pardan
Emit
Bang Payeen
Uxxari
Marthing
Birzuz
Meragram 2
Yorjokhun
Shich
Lushtdan
Yukum
Phasnk
Dowsoor
Sholkuch
Muli Wasum
Wasum
Zhapu
Gazen
Nichage
Scale at: 1:48,801
HAZARD MAP LOWER YARKHUN
VALLEY UNION COUNCIL
Aga Khan Agency for Habitat
www.akdn.org/our-agencies/aga-khan-agency-habitat
Note: District Ghizer falls in high (3) seismic zone (NESPAK).
Legends
Road
Activity Zone
UC boundary
Active Riverbed
Hazard Class
High
Medium
Low
No Known Hazard
Hazard Type
Bank Erosion
Rock Fall
Debris Flow
Flood
Snow Avalanche
Remote geohazrd
Coordinate System: GCS WGS 1984
Datum: WGS 1984
Data Source: Administrative Boundary, 2020 and HVRA Data Q3,2020.
The boundaries and names and the designations used on this map do not imply official endorsement or acceptance by the Aga Khan Development Network (AKDN). All information is the best available at the time this map was produced.

FIGURE 04: Hazard Vulnerability map of Lower Yarkhun Valley, Chitral.- showing high risk areas in red, medium risk in orange, low risk in yellow, and safe in green. Ishrat & Baig (2022). AKAH Model Home Report: Country Analysis. [unpublished NGO Report]

1.1.2 A Solution?

By leveraging local soil, the lower compressive strength can be offset by optimizing the slab's shape for effective load distribution. An effective topology that makes use of this is the shell; it is more structurally efficient due to which it can become more lightweight and reduce the material demand (Nanayakkara, 2019). However, to offer support to construction – formwork is often required which can become complex, expensive, material intensive, and labour intensive. Whilst there are different types of shells, the Catalan vault offers a particularly promising solution. A minimally reinforced floor constructed as a thin-tiled Catalan vault eliminates the need for form-work, reducing costs. However, whilst the thin compression-only nature of the Catalan vault gives it efficiency under static gravity loads, it becomes vulnerable under seismic loads which introduce bending stresses that may lead to structural failure.

1.1.3 Proposal

To address these concerns, the project aimed to develop a computational framework that optimizes the design of Catalan vault slabs to withstand seismic loads. Multiple simulations may be necessary during the initial design stages, where varying inputs are common to allow for greater design flexibility. This can become time-consuming and computationally expensive. Moreover, conducting shape optimization simulations can be additionally resource-intensive. This is especially true for detailed simulation models such as non-linear (FEA) micro-models or discrete element models (DEM). Implementing an AI-powered generative model could potentially reduce computational time whilst enhancing design freedom in generating diverse design options. Generative Adversarial Networks (GAN) and Variational Autoencoders (VAE) are both types of generative models. The scope of the project is limited to the VAE as GANs would merit a research paper of their own due to their complexity and issues such as instability in training, as mentioned by many sources including Salimans et al. and Arjovsky & Bottou, according to Regenwetter et al. (2022).

1.1.4 Applicability and Scope

In the broader context, the chosen case aims to serve as a test to determine whether the AI framework can generate reliable outputs. If the VAE learns to generate optimized solutions from a simpler dataset, it may be extrapolated that this means that there is also high potential for it to generate optimized solutions for more complex micro-models if the training dataset had been trained on that performance evaluation model as well. This would inform whether the case can serve as an application for the use of a VAE in shape optimization tasks which would otherwise be far too computationally expensive to perform. The project also aims to determine whether user-defined specifications can be considered while sampling as this would present immense potential in the future for controlling the types of samples that are desirable such as of a vaults of a particular height in this case.

1.2 RESEARCH QUESTIONS

1.2.1 Main Research Question

- Can an AI based framework generate new Catalan vaults for optimized seismic performance for use as a floor slab?

1.2.2 Sub Questions

Deep generative + Vault

- Can a user be able to tune the latent space in order to generate novel samples with user-defined desirable characteristics?
- Can individual dimensions of the VAE's latent space be interpretable?
- Can having a reduced sample dimension still justify the use of a Deep Neural Network

Deep generative + Seismic + Vault

- Can the vaults be optimization for multiple performance metrics?

Seismic + Vault

- What effect does varying the force densities have on overall seismic performance?
- Is there any favourable pattern in terms of force densities for seismic performance?

1.3 DESIGN ASSIGNMENT

The goal was to create a deep generative model based on the problem earlier highlighted in the last section. It would consist of a Variational Autoencoder that is able to generate novel designs of Catalan vaults optimized for seismic loads in order for them to be used as floor-slabs. Baked earthen tiles were to be used as the material for the masonry vault as a low-cost alternative to more material-intensive, expensive, and carbon-intensive materials that are not easily accessible.

The project can be divided into largely a two-tiered approach. The first phase involved the literature review which guided the formation of a workflow for the problem. In the second phase, the workflow was applied in the formation and testing of a generative model. The workflow is highlighted in the Research Methodology Chapter.

1.4 RESEARCH WORKFLOW

A deep generative model was to be constructed for shape optimization. The main workflow was categorized into 4 main parts: Geometry Generation, Performance Evaluation, Data Structuring, and Variational Autoencoder, as shown in Figure 05. This section briefly introduces the workflow which will be elaborated later in the Research Methodology chapter. The methodology for the workflow was inspired by the work of Sterrenberg (2023) and Pavlidou (2022).

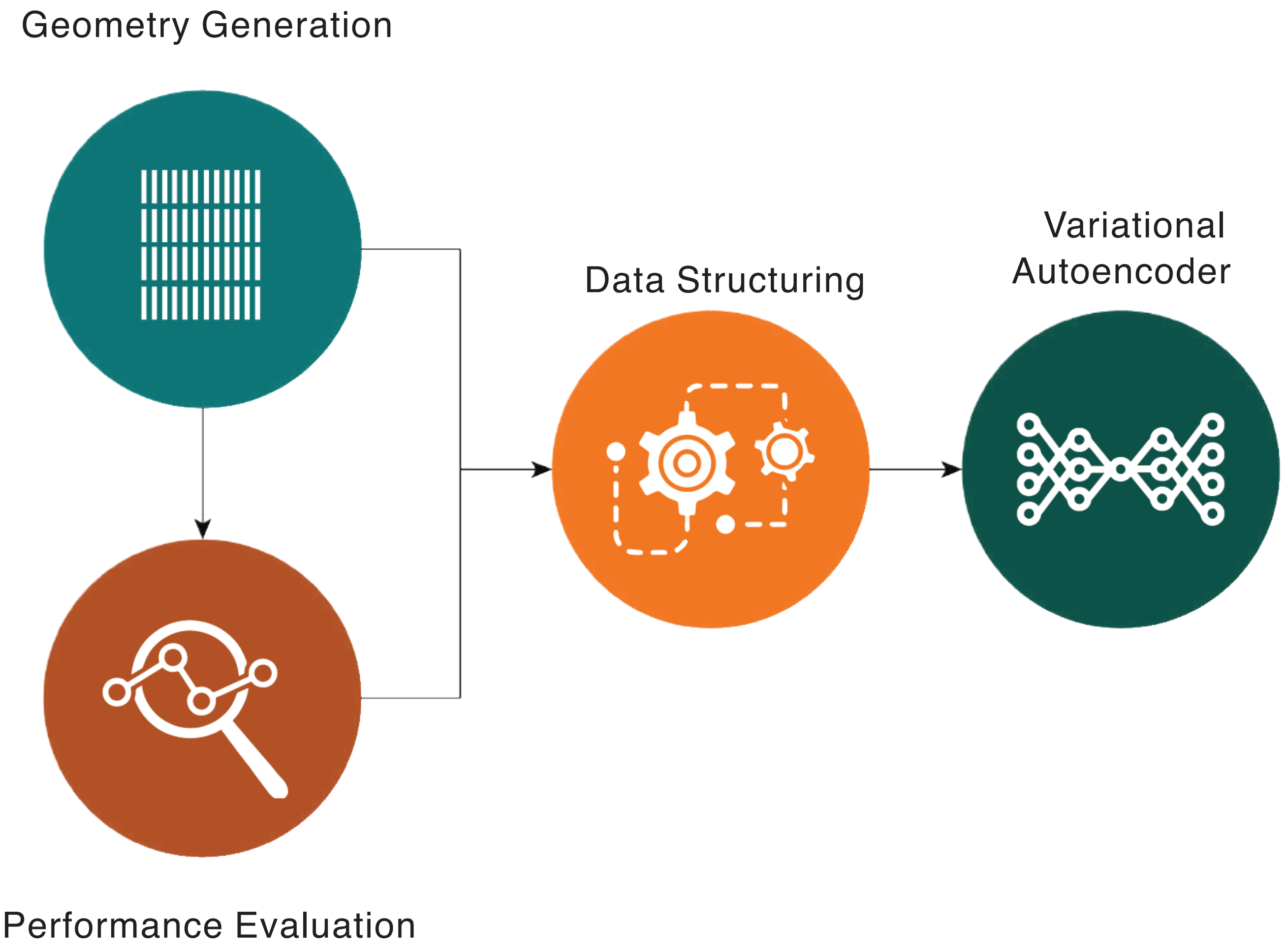


FIGURE 05: Overall Workflow. (Own Work)

Please refer to Figure 06.

Data Generation

An input shape was defined in the initial step. This would form the perimeter of the floor slab. In the initial workflow, for simplification, the footprint was to be kept as a rectangle with constant dimensions x and y. Form-finding would next done be on the input footprint inside Grasshopper where multiple design variations would be produced through the use of Force Densities through COMPAS (See Section 3.1.2 for detail).

Performance Evaluation

The geometry was then evaluated seismically through a Linear Dynamic Procedure in the software Karamba. Since this was also integrated inside Grasshopper, it helped connect the Geometry generation in Grasshopper. ABAQUS was used for validation purposes for modal analysis.

Data Structuring

After this, the data was restructured in a form readable for the neural network.

VAE

The structured dataset was then used to train a Variational Autoencoder. Initially, Graph convolutions (GCN) were to be used inside the encoder layers to learn distinct features and for permutation invariance. However, another way of expressing the geometry into a smaller dimensions became evident which did not require graphs. This is why GCNs were ultimately not used. A surrogate model was then used to evaluate the performance of the reconstructed output from the decoder using back-propagation which would then be used for gradient descent optimization.

CVAE

Once the VAE had been trained and tested successfully generating new designs, the latent space would be conditioned.

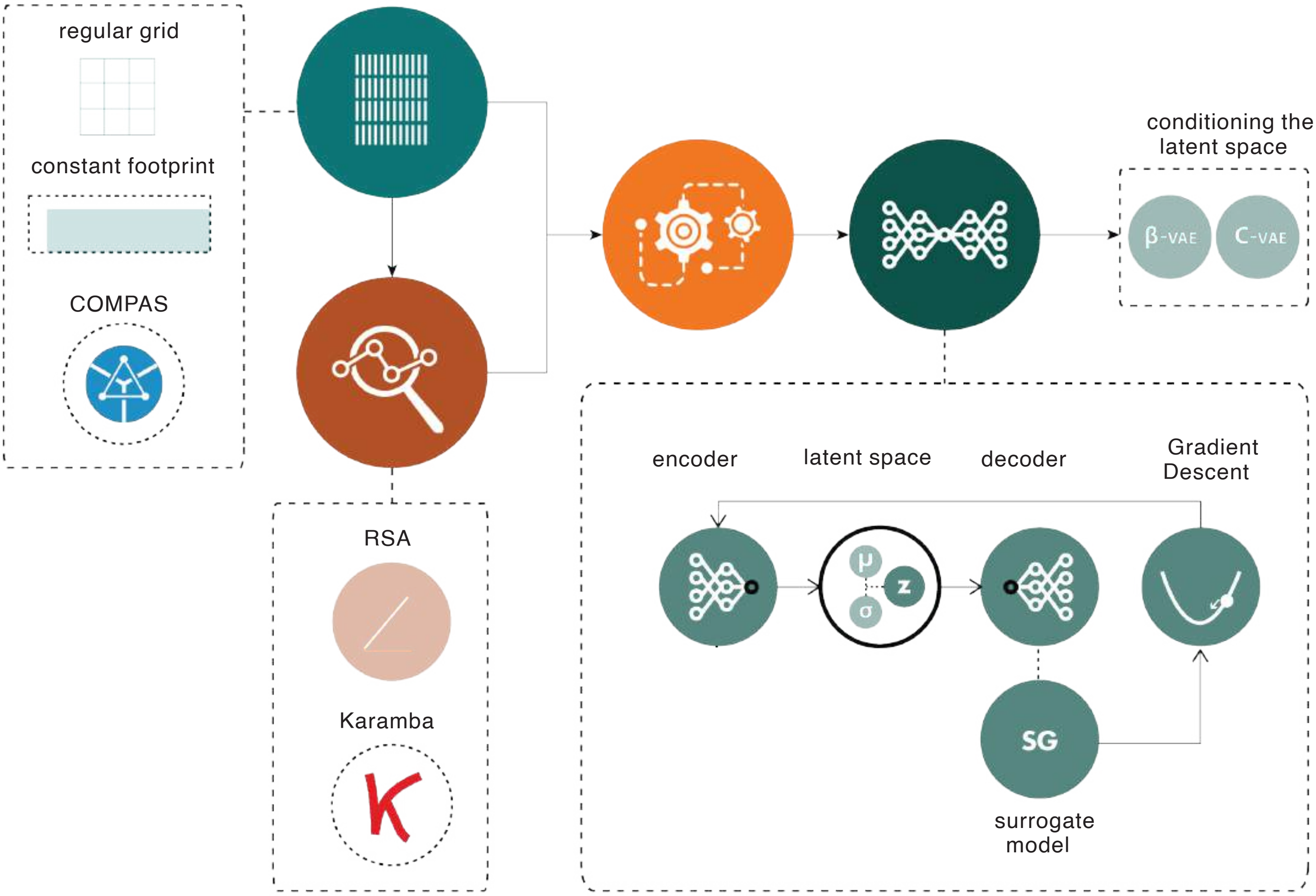


FIGURE 06: Overall Workflow breakdown. (Own Work)

The research consists of tools, software, and concepts that are novel for the author of this paper. Therefore, for practical purposes considering time limitations and computational power, the approach had been structured in the form of a basic structure with incremental layers of complexity introduced as the project develops. This aimed to firstly establish a basic yet holistic framework, then work towards developing more complexity through prioritization of goals and realistic choices under the time-frame. This aimed to keep alternative workflows available in case of possible bottlenecks caused by unforeseen delays instead of reformulation of the workflow altogether.

The other auxiliary workflows were ultimately not carried out due to time constraints. Nevertheless, they have been kept in this document to add context to how the project may be developed in the future. For context, the workflow highlighted earlier is Workflow 1, which would be carried out in all cases to set the foundational structure. Refer to Appendices Section 10.2 for further details.

02/ LITERATURE REVIEW: CATALAN VAULT

This section deals with the application of the generative model – the Catalan Vault.

2.1 BACKGROUND

Shells and vaulted structures are not a modern invention. They have been around for thousands of years. They provide one of the most efficient ways of spanning spaces by making use of the compression-only properties of the funicular geometric form. The earliest vaulted structures date back to 3000BC in Mesopotamia (Nanayakkara, 2019).

Though there are multiple types of vaults. For this paper, we shall only focus on Catalan vaults also known as Guastavino vaults, timbrel vaults, tile vaults, or thin-tile vaults. As mentioned earlier, these can be constructed with little to no formwork. As the alternate names suggest, these are made by 2,3, or more layers of thin tiles. Whilst the Nubian vault also eliminates the use of form-work – the vault itself is heavy and material intensive. For this reason, it was not considered.

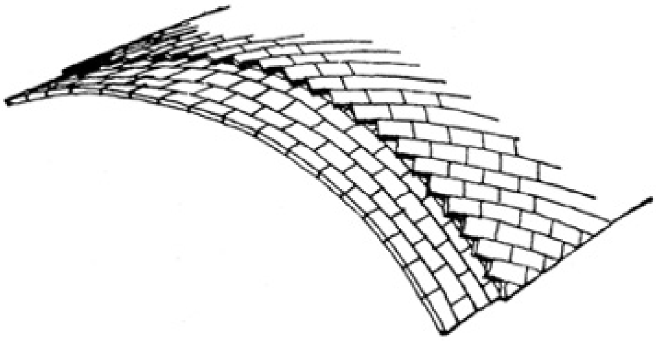


FIGURE 07: Catalan vault made from layers of thin tiles. The second layer is angled at 45 degrees to avoid continuous seams causing instability. Image retrieved from Moya, L. (1957). Archweb. https://www.archweb.com/en/design/page/catalan-vaulting/

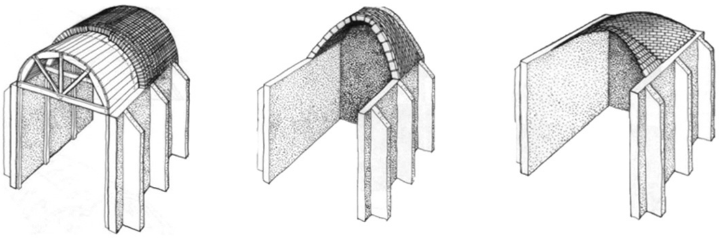


FIGURE 08: Types of vaults: a) Roman (semi-circular section built using formwork),b) Nubian (catenary section, no formwork), c) Catalan Vaulting (catenary section, can also be shallow, no formwork). Image retrieved from Chichester: John Wiley and Sons. Form and Forces. (2009). Archweb. https://www.archweb.com/en/design/page/catalan-vaulting/

2.2 CONSTRUCTION METHOD

The first layer is constructed 'in space' because of which a quick setting mortar is needed which in most cases is gypsum. This allows the inter-tile bond to set in seconds without the need for additional support. The tiles are placed flat and edge to edge, usually of 15-25mm thickness with a mortar thickness of about 10mm (Nanayakkara, 2019). Once the first layer of tiles is complete, it acts like formwork for the second layer. Gypsum is susceptible to environmental conditions so the bonding agent for the second layer of tiles is usually cement mortar. This sets slowly. If a third layer is also required which is constructed the same way as the second layer. In some cases, a third layer can be avoided by increasing the thickness of the second layer.

Once the vault is complete, it can act as permanent formwork for an infill layer. López López et al. suggest that the addition of a top layer of concrete can help reduce time and labour (2019). This would mean a higher weight for the same thickness. However, it is pertinent to mention that the bonding between the two different materials of the composite structure introducing complexity in the structural behaviour (López López et al., 2019). To avoid continuous seams that may lead to failure, the tiling pattern for the second layer is rotated at 45°, as shown in Figure 07.

During construction of the Catalan vault, guide work is required. An example of guide work can be drawn from SUDU, an urban housing project in Ethiopia, where steel frames are positioned on opposite ends of the longitudinal axis of the Catalan vault, and strings are threaded between them to serve as guides. The spacing of these guidelines relies on the abilities of the masons; a proficient mason, with a keen spatial sense, can work effectively even with coarser guide work (Nanayakkara, 2019). SUDU is also an example where the Catalan vault was created as a floor slab for economic in a low cost project (Figure 09).

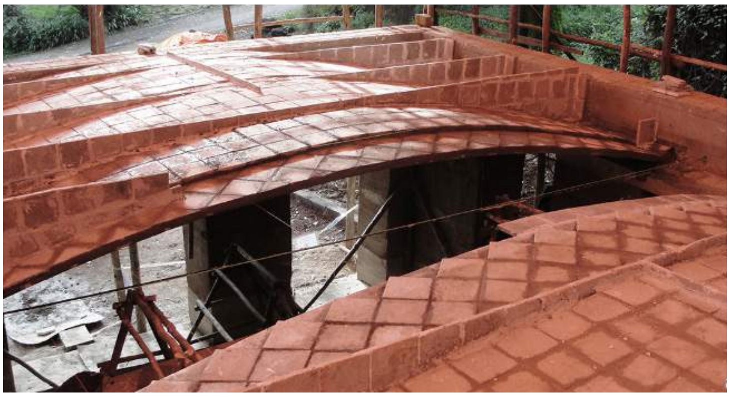
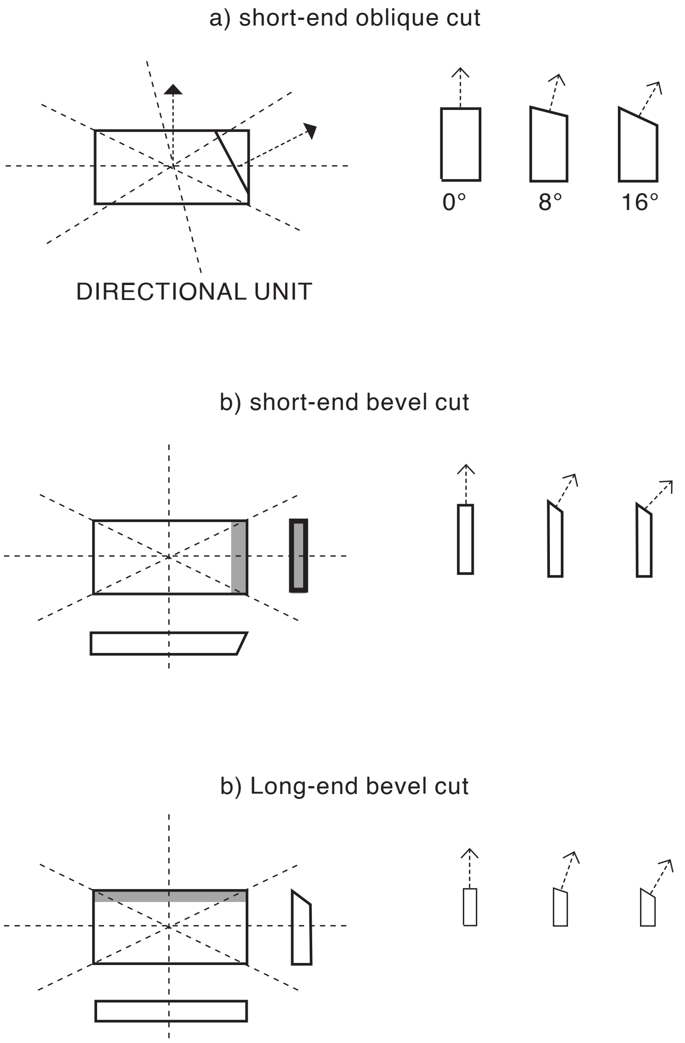


FIGURE 09: Catalan vault floor slab at SUDU project, Ethiopia with lightweight stiffening walls. Image retrieved from López López, D., Van Mele, T., & Block, P. (2016). Tile vaulting in the 21st century. Informes de La Construcción, 68(544), 162. https://doi.org/10.3989/ic.15.169.m15

For achieving high double curvature in vaults with orthogonal masonry units, an effective strategy is custom cutting bricks. This prevents compensation with a high amount of mortar in joints. Davis et al. identify two main methods – one cut systems and two cut systems (2012). Two cut systems are more effective in achieving greater curvature. One cut systems are simpler allowing for high degrees of curvature in one axis. Both systems use three brick primitives (Davis et al., 2012):

- short-end oblique cut
- short-end bevel cut
- long-end bevel cut


a) short-end oblique cut

b) short-end bevel cut

b) Long-end bevel cut

FIGURE 10: Brick primitives for the one-cut and two cut systems. Redrawn by author. Original retrieved from Davis, L., Rippmann, M., & Pawlofsky, T. (2012). Innovative funicular tile vaulting: A prototype vault in Switzerland.

2.3 STRUCTURAL PERFORMANCE

In the design of masonry structures, there are three main structural performance metrics – strength, stability, and stiffness (Heyman, 1966). The structure must be strong enough to carry its own weight in addition to imposed loads, it must be stiff enough not to undergo large deflections, and the structural forces may be contained within the arch preventing (four-bar chain) collapse (Heyman, 1966).

Failure in unreinforced masonry structures is usually due to instability instead of lack of compressive strength (Panozzo et al., 2013). The concept is that the connectivity should represent the flow of forces (Panozzo et al., 2013). The forces follow the shortest path so the risk of sliding can be reduced if the pattern follows the stress flow. The analysis of masonry vaults (applicable to Catalan vaults) has roots in Heyman's Safe Theorem where the equilibrium approach is used. Heyman's Safe Theorem states (Nanayakkara, 2019):

"If a set of internal forces in a masonry structure can be found that equilibrate the external loads, and which lie everywhere within the masonry, then the structure is safe – safe in the sense that it cannot collapse under those loads."

The limitations of linear analysis concerning shell structures have been highlighted by Block at al. (2006). There is a potential for unsafe and deceptive results, especially for thinner structures. Linear elastic FEA analysis may not accurately predict the stability or collapse of the structure, as it assumes the material is capable of resisting tension without considering the actual collapse mechanisms. This is especially true for thinner arches. Additionally, linear elastic analysis may not provide insights into the stability or collapse of the structure based on its geometry and equilibrium conditions, which can be crucial for understanding the behaviour of vaulted masonry buildings. Block et al. highlight that even for 2D problems, it is difficult to draw conclusions from the Linear analysis (2006). It can be understood that the problem may become greater when done in 3 dimensions for double curvature.

From Heyman's theory, it is clear that thin tile vaults are more vulnerable to collapse because the thrust line may not lie inside the masonry vault (Nanayakkara, 2019), and within the middle third to avoid tension (Block et al., 2006).

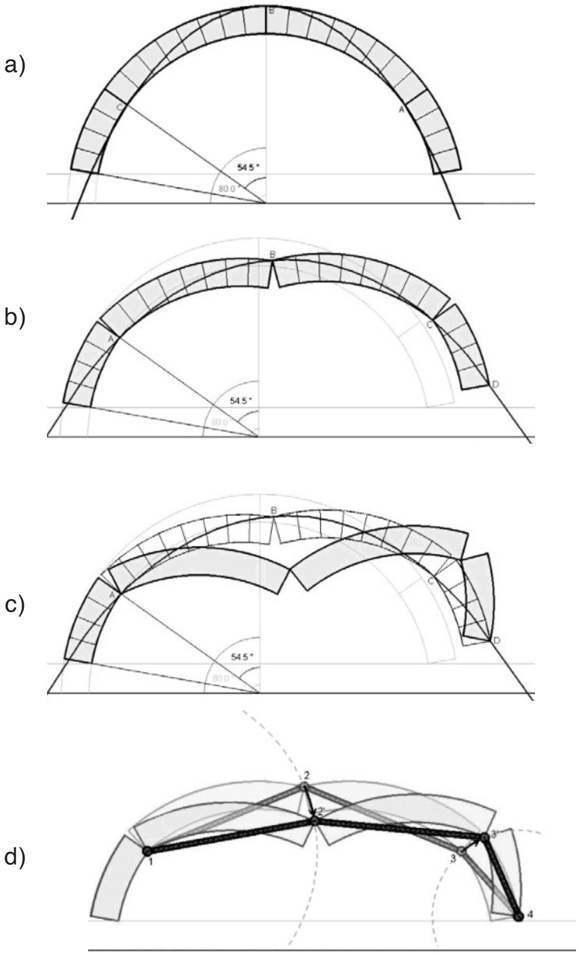


FIGURE 11: An arch which fails according to Heyman's Safe Theorem. Possible limit of deformation when the arch in (a) becomes (b) unstable, and (c) a snapshot of animation during collapse. (d) Four hinges define a three bar mechanism. Image retrieved from Block, P., Ciblac, T., & Ochsendorf, J. (2006). Real-time limit analysis of vaulted masonry buildings. Computers & Structures, 84(29–30), 1841–1852. https://doi.org/10.1016/j.compstruc.2006.08.002

2.3.1 Variable load

While the thin shell makes it lightweight, the Catalan slab is vulnerable to variable loads. Vertical stiffeners can be used to for increasing the stiffness against this as shown in Figure 13. Stiffness can also be increased by introducing double curvature as it provides multiple load paths to carry asymmetric loads; this was used widely by Rafael Guastavino in his designs (Nanayakkara, 2019). See Figure 12 for reference, where the double curvature of the vaults support variable loading from vehicular traffic on the bridge above.

2.3.2 Lateral thrust

To carry the lateral thrusts, modern precedents have used steel tie rods as reinforcement for Catalan vault such as in the Armadillo Vault by Block Research Group. Antoni Gaudi used inclined columns for this purpose in his work Park Güell (Nanayakkara, 2019). Traditionally buttresses would be used for this purpose.

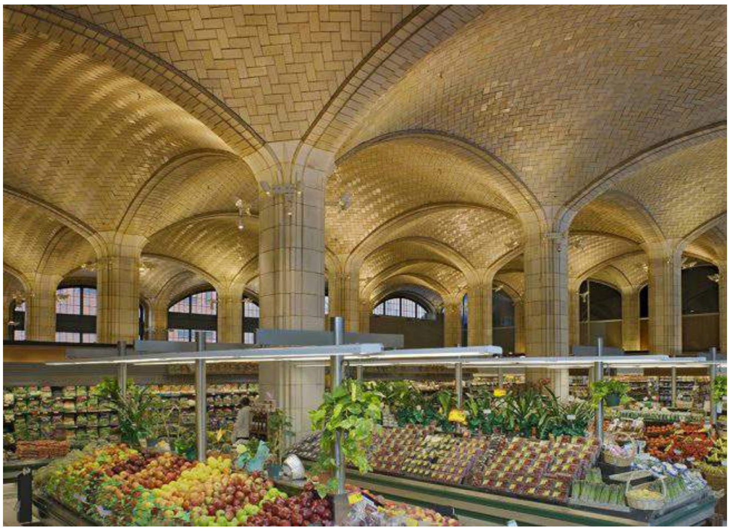


FIGURE 12: Queensboro Bridgemarket, in New York City, USA, has doube curvature supports vehicular traffic on top. Image retrieved from Michael Freeman, n.d., Urbanomnibus.net. https://urbanomnibus.net/2014/08/palaces-for-the-people-guastavino-and-the-art-of-structural-tile/

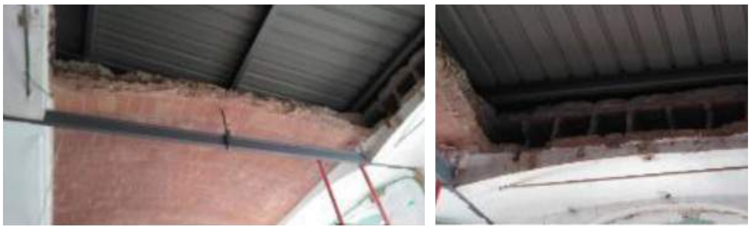


FIGURE 13: Partly demolished part of a Catalan slab showing vertical stiffeners in the hollow slab. Image retrieved from Nanayakkara, K. I. (2019). Shell Structures from Catalan to Mapungubwe Lessons from Structural Efficiency for Sustainable Construction in Developing Countries. https://doi.org/10.13140/RG.2.2.30878.89922

2.3.3 Seismic loads

For stability against asymmetric loading and seismic loads, reinforcement can be added in between the layers of the Catalan vault in the intermediate mortar joints. Additional tensile and bending capacity is provided this way. This also allows for a reduction of thickness. Though steel rods provide such reinforcement in single-curved vaults, they cannot be used for double-curvature – which is a property that is important to the vault system that is to be developed for this project.

Moreover, the steel exposed to environmental conditions rusts causing structural concerns. Instead, a geo-grid reinforcement is proposed that is appropriate for complex double curvature as well as being corrosion resistant and weatherproof (Surat, 2017) . Polymeric grids and glass-fibre meshes are used widely nowadays (López et al., 2019). A recent case study of such reinforcement used in thin tile vaults is the work of Michael Ramage and Matthew DeJong where they applied a geo-grid in between the layers of bricks in the Bowls Project in San Francisco as illustrated in the Figure 14, (Ramage & Dejong, n.d.) which is a low to moderate earthquake risk area. The geo-grid increased the ductility of the structure and improved its bending capacity. This is relevant to the economic nature of the project as it too is an inexpensive solution.

Surat tested 3 techniques of reinforcement of thin tile vaults with basalt geo-grid - laying it over the base and mortared, anchoring it and epoxied, and pre-stressing it. The last two methods were found to be the most effective with observed collapse accelerations being over 60% higher than that of the same unreinforced vault. It was concluded that the anchoring solution is the most effective as pre-stressing requires considerable technical expertise (Surat, 2017). This is shown in Figure 15.

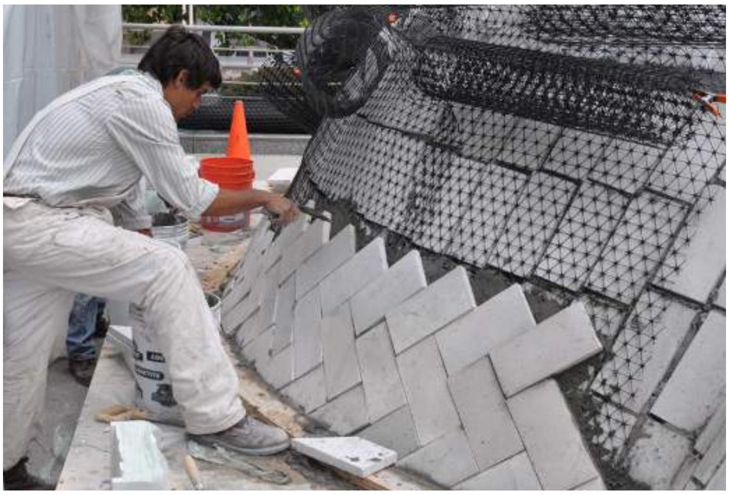


FIGURE 14: Geogrid embedded in between mortar layers and tiles, for reinforcement against seismic loads, in the Bowls Centre, Yerba Buena Centre for the Arts, in San Francisco, USA. Image retrieved from Ramage, M. H., & Dejong, M. J. (n.d.). Design and Construction of Geogrid-reinforced Thin-shell Masonry.

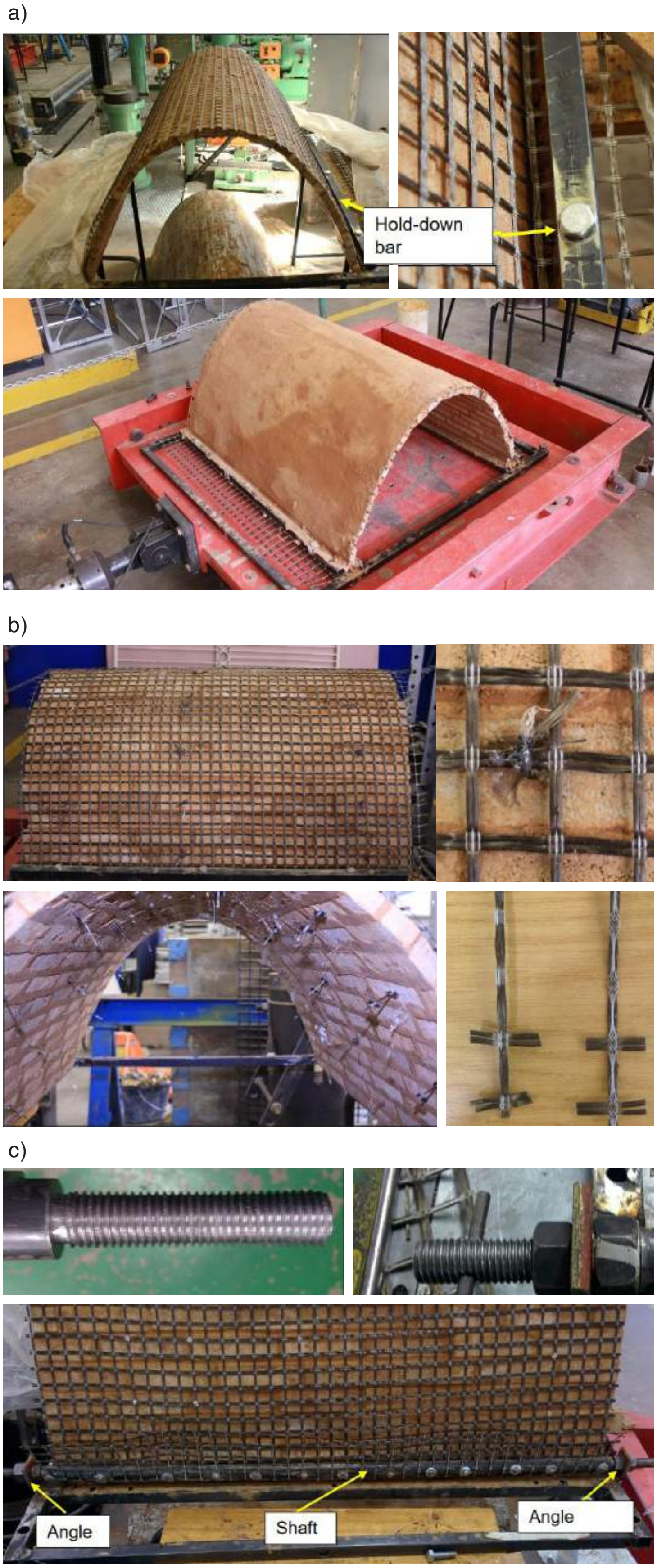


FIGURE 15: Reinforcement strategies for thin tile vaults. a) laying geogrid over the base, b) anchorage of geogrid, c) pre-stressing the geogrid. Image retrieved from Surat, D. (2017). Seismic Analysis of Thin Shell Catenary Vaults [Master Thesis]. University of Witwatersrand.

The tiling pattern is an important consideration. Cross-herringbone patterns have been used to increase the stability by prevention of sliding of tiles and overturning. Used famously by Brunelleschi, the pattern makes a plate-bande resistant system enables equilibrium states of self-supporting shells (Paris et al., 2020). The pattern is defined by a network of double loxodromic curves, one right-handed and one left-handed loxodromic. This is shown in Figure 17.

2.3.4 Strengthening

To strengthen the vault, Auguste Choisy (1873, as cited in López et al., 2019) echoed Fray Lorenzo (1633, as cited in López et al., 2019) as he mentioned the typical practice of adding filling material from the supports of the tile vault, extending it to about one-third of its length. This was also carried out by Guastavino in his design where he used concrete as an infill to stiffen the vault and level it to be used as a flat floor. The same strategy of using an infill was also used for the SUDU project, as mentioned earlier, by ETH Zurich's Block Research Group, for the Catenary floor slab, as shown in Figure 16.

2.4 CHALLENGES AHEAD

Superior structural efficiency may not be enough for the adoption of such a technology. According to Nanayakkara, it wasn't structural performance which led to the adoption of the Guastavino vault in USA (2019). It was its superior fire resistance as compared to the norm at the time which was timber construction. There are social implications of construction material especially in the developing world which present an obstacle to its adoption. Earth as is viewed as an 'inferior' building material to steel and concrete as the latter materials are associated with progress and modernity. This is true in the districts of Chitral and Gilgit-Baltistan where the poor insulating properties of concrete as compared to earth are widely known; however, the homeowners make a conscious decision to opt for concrete construction for new homes because of the status it signifies (T.Z.Ishrat, personal communication, March 2022). The stigma, however, comes from the poor finishing and constant need for maintenance that the construction presents. This also means that there is potential of removing that stigma if the finishing can be improved upon. That, however, is outside the scope of this paper. The same stigma also exists in India but exceptions like Auroville exist where earth construction is well respected (Nanayakkara, 2019).

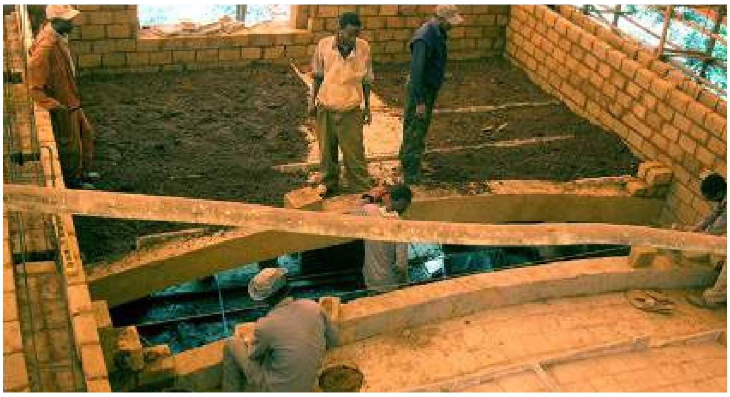


FIGURE 16: Catalan vault floor slab at SUDU project, Ethiopia, with lightweight stiffening walls and compacted fill. Image retrieved from López López, D., Van Mele, T., & Block, P. (2016). Tile vaulting in the 21st century. Informes de La Construcción, 68(544), 162. https://doi.org/10.3989/ic.15.169.m15

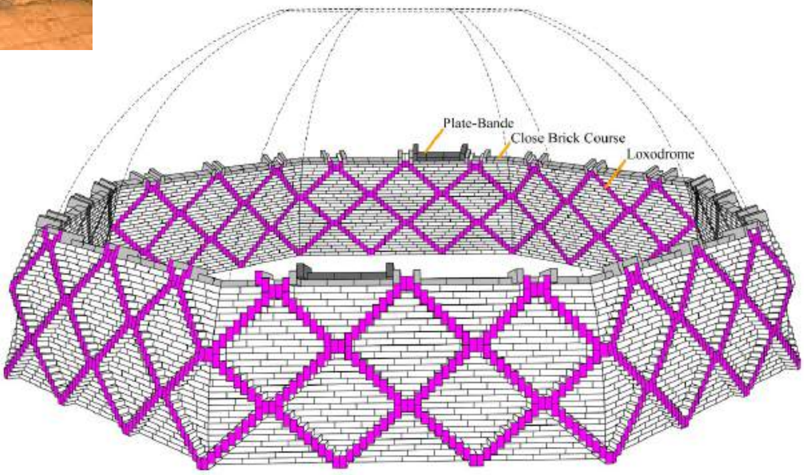


FIGURE 17: The view of a dome under construction, showcasing a plate-bande (emphasized in dark gray), a closed brick course (highlighted in light gray shading), and loxodromic trajectories on both sides (highlighted in purple). Image retreived from Paris, V., Pizzigoni, A., & Adriaenssens, S. (2020). Statics of self-balancing masonry domes constructed with a cross-herringbone spiraling pattern. Engineering Structures, 215, 110440. https://doi.org/10.1016/j.engstruct.2020.110440

03/ LITERATURE REVIEW: GENERATIVE MODEL

This section deals with the literature related to the generative model. It has been structured in the same way as the overall Research Workflow in its constituent sections. See Figure 05 for reference.

3.1 GEOMETRY GENERATION

The first step of the project was to generate the dataset. This was to be composed of the geometry that would be structurally analyzed in the next step. The target sample size of the dataset from the literature had found to been at least 10,000 samples. (Sterrenberg, 2023) and 7,338 (Pavlidou, 2022).

3.1.1 Input Footprint

An input shape was defined in the initial step. This would form the perimeter of the floor slab. For simplification the footprint was a rectangle with constant dimensions 10mx15m. This size was chosen as the footprint of a residence.

3.1.2 Form-finding

Form-finding was done by relaxing the input shape into a funicular mesh. It was a crucial part of the workflow. This is because it was the tool to generate the large variation of geometries that was needed for the generative model. The workflow for this follows the following steps (Oval & Rippmann, 2017):

- 1 - defining the boundaries
- 2 - designing a planar mesh
- 3 - setting constraints
- 4 - form finding

Different methods were considered for form-finding – Particle Spring System analysis (PSS) (used by Kangaroo, the Grasshopper plugin) and Thrust Network Analysis (TNA) (used by RhinoVAULT). It was found, through an FEA in Karamba, that TNA obtained a more structurally efficient design than PSS based on the quantity of masonry used, variations in geometry, differences in curvature and stress distributions, and deformations under two load combinations(Contestabile et al., n.d.). The method also allows greater control over the geometry produced than PSS as it is especially tailored for funicular forms, which is significant for this project. By changing the force densities of the edges that make up the structure of the vault, the force that the edge carries changes, due to which pattern changes could be achieved. This was especially important in introduction of creases and vault segmentation for a large richer geometrical design space. For this reason, TNA was selected.

Thrust Network Analysis

TNA was developed by Phillipe Block and evolved by his research group, Block Research Group. It relies on the principles of graphic statics applied to three-dimensional structures. Similar to graphic statics, the approach revolves around two interrelated diagrams— the form diagram and force diagram that correspond to equilibrium states represented by force polygons (Contestabile et al., n.d.). It uses the concept of force densities from the Force Density Method introduced by Linkwitz and Scheck (1971, as cited in Aboul-Nasr & Mourad, 2015) and Scheck (1974, as cited in Aboul-Nasr & Mourad, 2015). By using FDM, any state of equilibrium of a funicular structure can be obtained by the solution of one system of linear equations. This system is constructed using the force-length ratios or "force densities" in the branches as parameters that describe the network ("degrees of freedom"). Simply put, we specify a singular quantity, namely the force density, for each branch. By solving one system of linear equations, we obtain a unique result—the equilibrium structure with the designated force-length ratio in each branch. (Schek, 1973). TNA is able to manipulate and compute these force densities this intuitively through manipulation of the force diagram. Based on Heyman's Safe Theorem, for a masonry structure, if any graph is completely contained inside the thickness of the masonry structure, then the structure is in equilibrium. The Thrust Network Analysis allows the user to manipulate the force diagram to visualize these graphs for which the structure is in equilibrium which entails that the equilibrium equations are satisfied for each node.

However, RhinoVAULT, the plugin for TNA works directly in the Rhino interface and not Grasshopper. The Grasshopper environment is necessary for the automation of the dataset generation. Whilst RhinoVAULT was not able to be used, COMPAS, the Python framework that it is based on may be used instead directly as a GH Python script in Grasshopper provided by Robin Oval. A number of different geometries can be found from the same load case by changing force densities of the different edges via the force diagram, as shown in Figure 18.

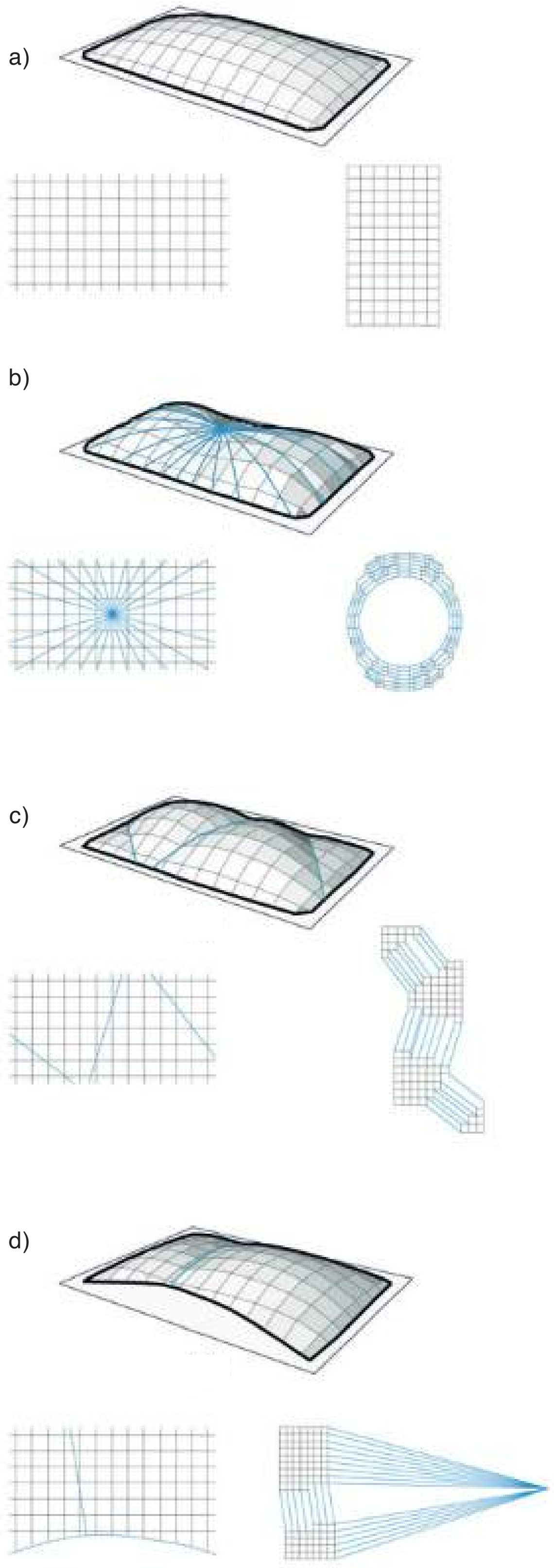


FIGURE 18: For the same UDL, the figure shows different vault geometries formed as a result of indeterminacy of a 4-valent network. a) equal distribution of horizontal forces forms a thrust network with a typical pillow shape, (b-d) higher force densities in different regions result in creases for the equilibrium solutions. Image retrieved from Adriaenssens, S., Block, P., Veenendaal, D., & Williams, C. (Eds.). (2014). Shell structures for architecture: Form finding and optimization. Routledge/ Taylor & Francis Group.

Mesh Generation for Force Diagram

Two main categories of methods were identified for meshing – backward processes and forward processes. (Oval & Rippmann, 2017). Both generate a 2D pattern input as a force diagram for generation of a 3D funicular structure based on the TNA. Backward processes use an input 3D geometry to approximate a self-supporting structure entailing its optimization, rationalization, and post processing. Forward processes, on the other hand, result in a more open-ended design approach which includes mesh generation, exploration, and design.

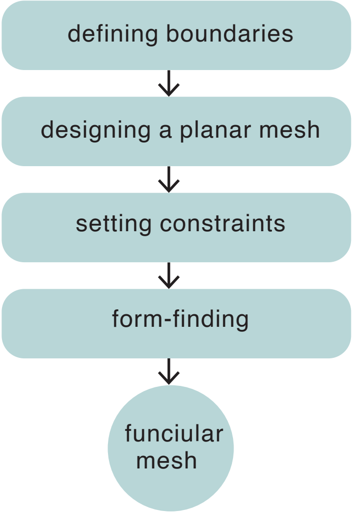


FIGURE 19: Forward Process. (Own Work)

Forward process

The methodology proposed by Oval & Rippmann is a forward process which makes use of a surface that is represented by its boundaries. It is split into a set of topologically simpler patches through a medial-axis based block decomposition process. Each patch is then subdivided after which a quad meshing pattern is used to generate the mesh. Then, through a relaxation technique the mesh is smoothed (Oval & Rippmann, 2017). A quad mesh pattern is generally preferred as it makes manipulation of the funicular geometry more stable and simpler. It also gives allows for directionality useful for tiling patterns.

It is a feature-based topology finding technique where additional features such as curve features and point features. Inclusion of these features change the mesh geometry according to a set of prescribed rules (Oval & Rippmann, 2017). Boundary point features may represent concentrated lateral thrust acting at the corners of the vault. Inner point features may be able to represent nodal supports or point loads. Curve features may represent line loads or creases in the vault.

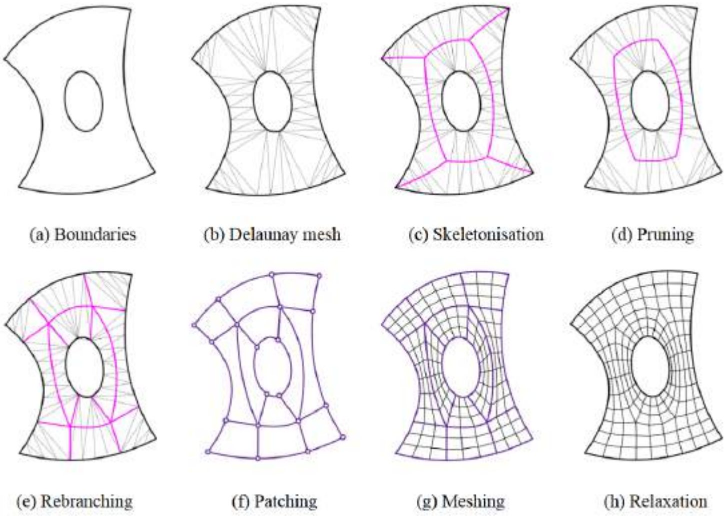


FIGURE 20: Forward Process. Generation of a smooth quad mesh without boundary singularities based on the medial axis. Image retrieved from Oval, R., & Rippmann, M. (2017). Patterns for Masonry Vault Design.

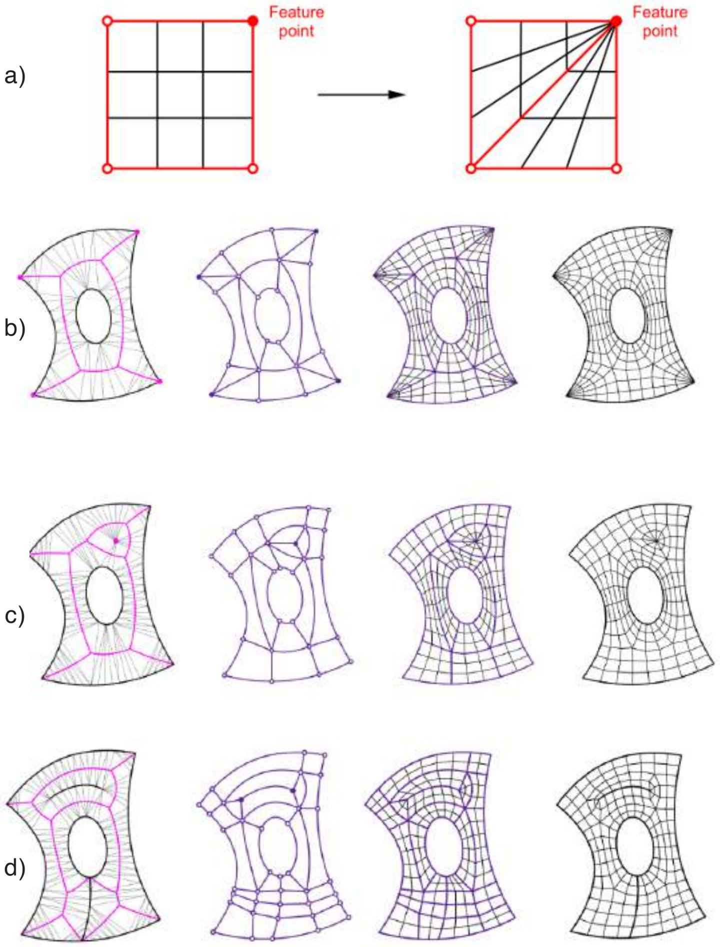


FIGURE 21: Forward Process. a) transformation rule for 4 sided patches for point features, b) mesh pattern from boundary point feature, c) mesh pattern from inner point feature, d) mesh pattern from curve feature. Image retrieved from Oval, R., & Rippmann, M. (2017). Patterns for Masonry Vault Design.

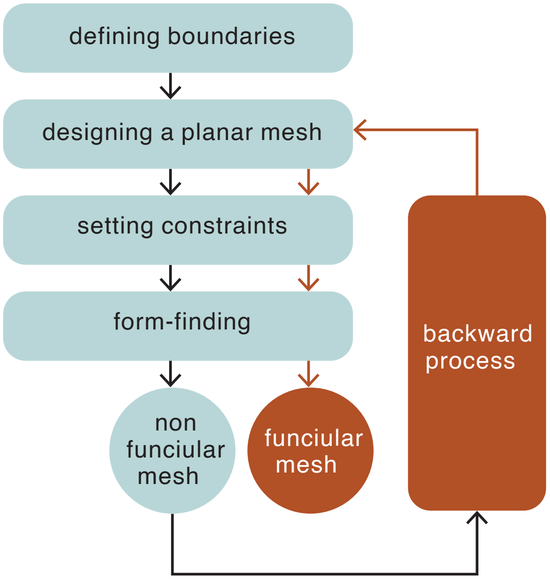


FIGURE 22: Backward Process. (Own Work)

Backward Process

The other category is a backward process where an existing input form is altered to create a structurally-sound mesh.

The forces have to be aligned with the edges. If, for instance, an arbitrary discretization is done and the quad mesh aligns with some features of the surface, the surface will not be properly represented. For instance, the forces in Figure 23a can either go to that feature highlighted in red, or an open edge (orange). In both cases, they do not go to the supports so the feature cannot be represented. If optimization is done through that discretization, then the features of the original surface are lost. However, in Figure 23b it can be seen that from the same point the force goes directly to the support. In this case the features are preserved.

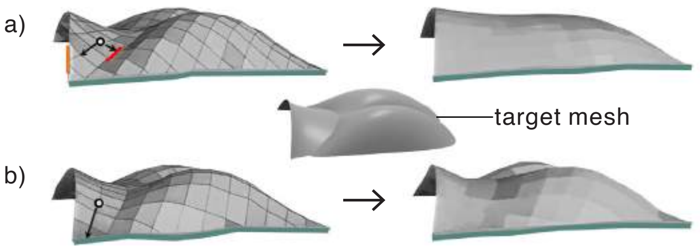


FIGURE 23: a) mesh loses features after becoming self-supporting while b) mesh retains most of the features. Edited image of original retrieved from Panozzo, D., Block, P., & Sorkine-Hornung, O. (2013). Designing unreinforced masonry models. ACM Transactions on Graphics, 32(4), 1–12. https://dl.acm.org/doi/10.1145/2461912.2461958

Negative gaussian curvature is also difficult to be represented. If an arbitrary discretization is done as in Figure 24a, and a point is taken, the forces acting on the point are all downwards leaving no resultant force to cancel out gravity. Figure 24b, however, shows a different discretization where a resultant upward force cancels out gravity. The idea of aligning to the principle curvature direction, however, is limited to local properties and does not take into account global features like open boundaries. For instance, if a hole is made, the curvature doesn't change but the stress distribution has to change as the forces cannot go straight to the hole. Panozzo et al. base their methodology upon both concepts, aligning to the principal curvature direction combined with global optimization. Open edges are first analyzed – as we want the forces to go as fast as possible to the closest support. Then negative curvature and sharp features are identified. These directional form constraints are then interpolated obtaining a cross-field across the whole surface. A quad mesh is then generated on to this aligned with the crosses. The diagram is then projected on to the plane and is input as a form diagram (Panozzo et al., 2013).

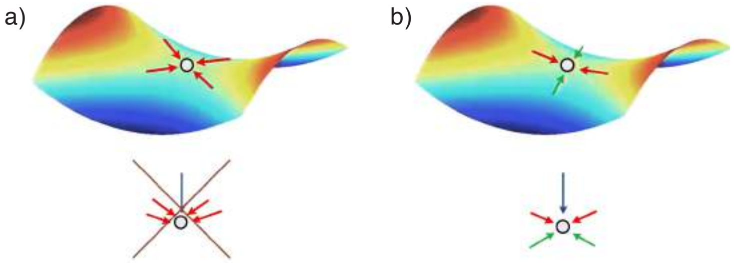


FIGURE 24: In areas with anticlastic curvature (a), an incorrect alignment of the force pattern prevents the representation of equilibrium at a vertex (depicted in red), as all directions of the force flow are directed downward. This issue does not arise in cases where a discretization allows a resultant upward force to cancel out gravity (b). Image retrieved from Panozzo, D., Block, P., & Sorkine-Hornung, O. (2013). Designing unreinforced masonry models. ACM Transactions on Graphics, 32(4), 1–12. https://doi.org/10.1145/2461912.2461958

Pattern to Funicular form

The output from both the approaches is a 2D projection of the shell. Once the pattern generation process has been done, the mesh edges that are fully supported are removed. The output can now serve as the form diagram that is input into the COMPAS framework. The reciprocal force diagram is obtained after which horizontal equilibrium is computed and subsequently, vertical equilibrium is then computed afterwards to achieve the funicular geometry. In the case of the backward process by Panozzo et al., the distance between the generated mesh and the original is minimized through a gradient descent optimization. This approach differs from Vouga et al. where the input form is deformed to make it self-supporting whilst in this case a self-supporting form is deformed to approximate the generated form – which, according to Panozzo, is more robust (Panozzo et al., 2013).

Conclusion

While the backward process presented by Panozzo et al. presented an interesting approach towards approximation of a self-supporting form from a given 3D geometry, it required a target shape adding complexity that was not necessary for this project. As the development of the funicular mesh of the Catalan Vault was meant as a floor slab, it made sense to consider an input 2D footprint rather than a 3D vault that is to be made into a self-supporting structure. Due to this, a forward process was selected going forward. The feature-based topology finding approach has a lot of potential to generate a large variation of designs and incorporating reinforcement.

3.1.3 Tiling Pattern

The Catalan vault is a masonry vault composed of thin tiles. The meshing pattern of its global geometry, as discussed in the section above, is a function of the force pattern used to compute static equilibrium in the form-finding process (Thrust Network Analysis). Therefore, it seems logical to base the tiling pattern on the meshing pattern employed in form finding to maximize structural efficiency. This approach ensures that sliding failure between the voussoirs is averted by aligning the force flow with the interface normals between them (Heyman, 1997, as cited in Oval & Rippmann, 2017). This is echoed by Adiels et al. as they describe how if principal stresses are perpendicular to the head joint and the bed joint of the tiles, there is no possibility of sliding along the bed joints (Adiels et al., 2017). Though earth tiles are used here instead of stone voissoirs, it may be assumed that it would also be applicable in this case.

Up till now we have described how an initial mesh can be generated and manipulated to form the funicular structure that is found through form finding (TNA). If the output mesh were to be extruded and segmented into tiles, there would be issues with fabrication as all tiles may be different. In the context of the case study, it would be highly impractical to suggest this as it would mean production of new moulds for each tile. Therefore, the tiling pattern must be standardized.

Generating the Tiling Curves

Two different approaches have been found by Adiels et al. using geodesic coordinates. Both propose a constant distance between the bed joints. This equal spacing allows constant sized bricks when offset where the tolerance between bricks would be mortar joints, as shown in Figure 25.

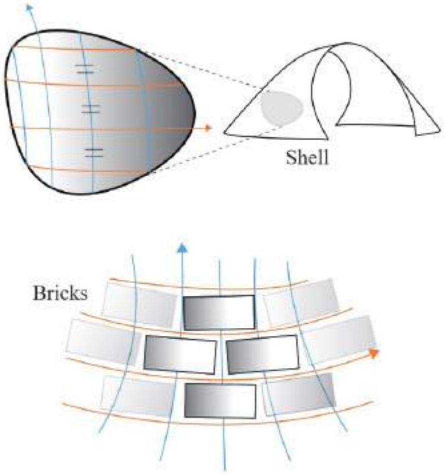


FIGURE 25: Equal sized tiles due to equal spacing of coordinate curves made possible using geodesic coordinates. Image retrieved from Adiels, E., Ander, M., & Williams, C. (2017). Brick patterns on shells using geodesic coordinates

Patterning Approach 1: form & pattern integration

The first approach combines form-finding and pattern generation using dynamic relaxation. This integrates geometric and structural properties into the pattern avoid sliding along the head joints, as discussed above. In the dynamic relaxation process, nodes move opposite to the direction of the out-of-balance force equivalent to a negative mass.

Patterning Approach 2: form and pattern disconnect

The second approach separates form and pattern by offsetting an initial curve, from which geodesics emerge at 90°. A sequence of circles, centered on the surface and situated in the normal plane at its midpoint, intersects the surface, creating a collection of points that define the geodesics. Due to the points lying within normal planes along the geodesics, the geodesic curvature is zero. This means that this can be discretized into tiles. The initial curve is offset to repeat the pattern. It is important that the geodesics do not cross; therefore, for a complex shape, it is better to patch each area separately then generate the geodesic pattern. This is shown in Figure 26.

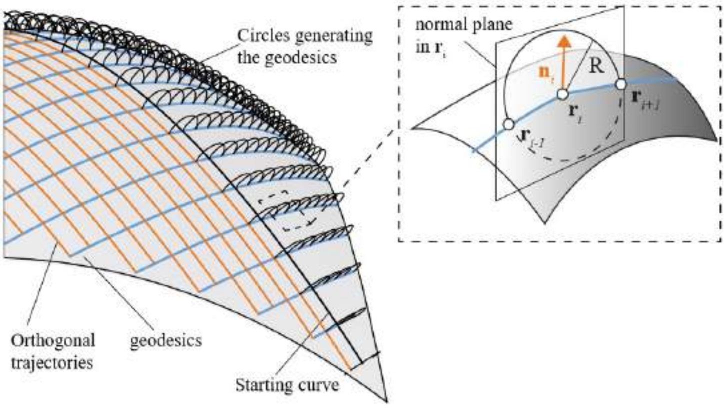


FIGURE 26: Generation of geodesic curves. Image retrieved from Adiels, E., Ander, M., & Williams, C. (2017). Brick patterns on shells using geodesic coordinates

Staggering the Tile

Panozzo et al. describe how the tiles or tessellations can be derived from a quad mesh once the funicular mesh has been generated. Every second edge in the preferential direction (towards the closest boundary) of the mesh is removed resulting in a staggered tiling pattern, as shown in Figure 27. An exception is made if this results in strong voussoir concavities as this may result in stress concentrations and difficulty in positioning correctly. The orientation of the pattern becomes significant to remove the chain of quads that may slide off from the open edges (unsupported arches and openings). It is not important in the regions far from the edges, but nonetheless the pattern is smoothed for aesthetic purposes and ease of constructability (Panozzo et al., 2013).

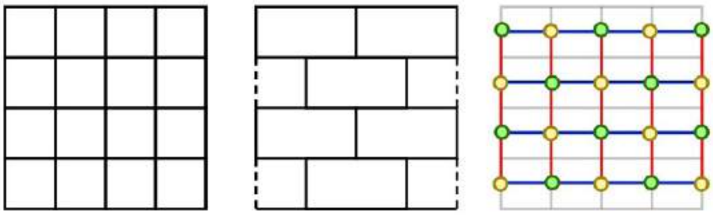


FIGURE 27: a) a quad mesh used to generate a b) Tiling/ brick pattern made by c) removing edges corresponding with yellow nodes. Image retrieved from Panozzo, D., Block, P., & Sorkine-Hornung, O. (2013). Designing unreinforced masonry models. ACM Transactions on Graphics, 32(4), 1–12. https://doi.org/10.1145/2461912.2461958

Conclusion

We can conclude that the Patterning Approach 1 offers the most structurally plausible option as it integrates the pattern formation with form-finding. However, it remains to be seen what design flexibility it offers to generate such a vast array of designs. Patterning Approach 2 offers a more flexible solution which can generate a vast number of design options. Multiple design variations can be made for the dataset by dividing the surface into different patches and varying the initial curve in those patches. The disconnect between form and pattern may allow testing the VAE with poor performing vaults in addition to better performing ones. Moreover, it can be hypothesised that, in this approach, to account for discrepancies between two curve boundaries, it may be useful to generate tween curves in between to smooth the transition of the pattern.

To segment the geodesic curves into a staggered pattern, Panozzo et al.'s approach may be applied not to the initial quad mesh generated from the form-finding but to the standardized geodesic curves mentioned in Patterning Approach 2. These can be extruded to form the tiles.

It would have been interesting to incorporate the cross-herringbone / loxodrome pattern, inside the DEM and see how the absence, and different variations of the pattern may affect structural stability.

Tiling is outside the scope of this paper as they were a part of the initial Workflow 3 which was not implemented (See Appendices Section 10.2). However, these strategies have been left in the paper as they are useful to consider going forward in the future.

3.1.4 Layers

The actual Catalan vault is composed of 2 or more layers. However, for simplicity, the model was based on a single layer. The thickness of the layers was to be defined. From the literature discussed earlier, the tiles thickness varied between 15 and 25mm, and the mortar was 10mm total. Based on this, 3 different thicknesses were used depending upon the number of layers of tiles.

1 layer of variable thickness. Inclusion of the thin layers may allow the VAE to check for failure.

- 35mm (1 masonry layer + 1 mortar layer)
- 60mm (2 masonry layers + 1 mortar layers)
- 95mm (3 masonry layers + 2 mortar layers)

3.2 DATA STRUCTURING

3.2.1 Graphs

While parameters at the vault-level and layer-level would be floats values, a different data structure would be needed to represent the mesh geometry of the vault at the node-level. A way to represent this is through graphs.

A graph is used to represent relations (edges) between different entities (nodes). The graph can store 3 different types of information – node (vertex) embedding, edge (or link) embedding, and global (master node) embedding.

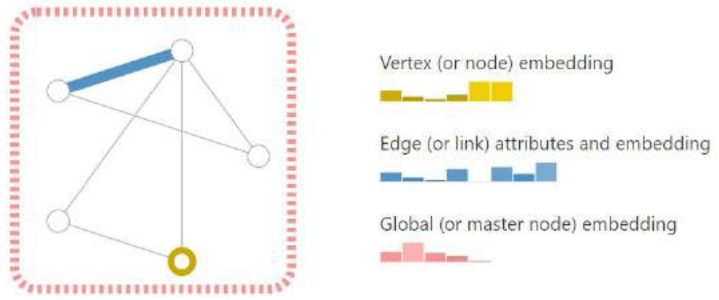

Vertex (or node) embedding
Edge (or link) attributes and embedding
Global (or master node) embedding

FIGURE 28: A graph and its embeddings. Image retrieved from Sanchez-Lengeling, B., Reif, E., Pearce, A., & Wiltschko, A. B. (2021). A Gentle Introduction to Graph Neural Networks. Distill, 6(9), e33. https://doi.org/10.23915/distill.00033

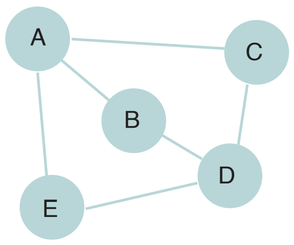
Sterrenberg (2023) and Pavlidou (2022) worked on a similar design problem that also required data structuring of a shell mesh to be used inside a VAE. Different types of graph structures were tested including adjacency matrix, edge-vertex matrix, vertex-vertex mesh, face-vertex mesh, winged-edge mesh, half-edge mesh. Sterrenberg (2022) found that data representations based on the position of vertices provided unrepresentative outputs. This included coordinate data and movement data. However, data representations based on edge/ topology connectivity provided more accurate results. It was noted that (half) adjacency matrices performed the best (Sterrenberg, 2023). This is also echoed by Pavlidou who found adjacency matrices to perform better than data based on vertex coordinates (2022). For this reason, adjacency matrices shall be used as to represent the node-level data structures.

3.2.2 Adjacency Matrix

Adjacency matrices are able to visualize connectivity of a graph. Conventionally, as shown in Figure 29, binary digits represent connectivity. Each row and column represent a node, and the presence of an edge is denoted by the value 1 while the absence of an edge by 0.

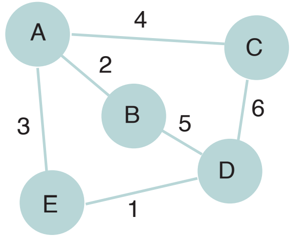
In this project, however, as mentioned earlier, the nodes must represent float values (to represent force densities, height,an direction vector) so the binary representations are not adequate to represent this. For this reason, weighted adjacency matrices using float values could have been used where instead of 0 and 1. Figure 30 shows an example of a weighted adjacency matrix.

There are 2 main problems associated with adjacency matrices - sparsity and permutation invariance (Sanchez-Lengeling et al., 2021).



	A	B	C	D	E
A	0	1	1	0	1
B	1	0	0	1	0
C	1	0	0	1	0
D	0	1	1	0	1
E	1	0	0	1	0

FIGURE 29: An adjacency matrix. (Own Work)



	A	B	C	D	E
A	0	2	4	0	3
B	2	0	0	5	0
C	4	0	0	6	0
D	0	5	6	0	1
E	3	0	0	1	0

FIGURE 30: A weighted adjacency matrix. (Own Work)

Sparsity

A high number of nodes and few number of edges can cause the adjacency matrix to be very sparse which causes space inefficiency. Therefore, a sparse graph would lead to a sparse adjacency matrix.

Permutation invariant

Another problem with adjacency matrices is that they are not permutation invariant. This means that the model's output should be the same regardless of the order in which the items are presented. Permutation invariance is desirable for deep learning models. This is because there can be a number of adjacency matrices that encode the same connectivity and there's no assurance that the same output would be produced (Sanchez-Lengeling et al., 2021). As shown in Figure 31, for a 4-node graph, 4! (24) adjacency matrices represent the same graph. We can evaluate that for many nodes, as our vault would have, the permutations increase by an order of magnitude.

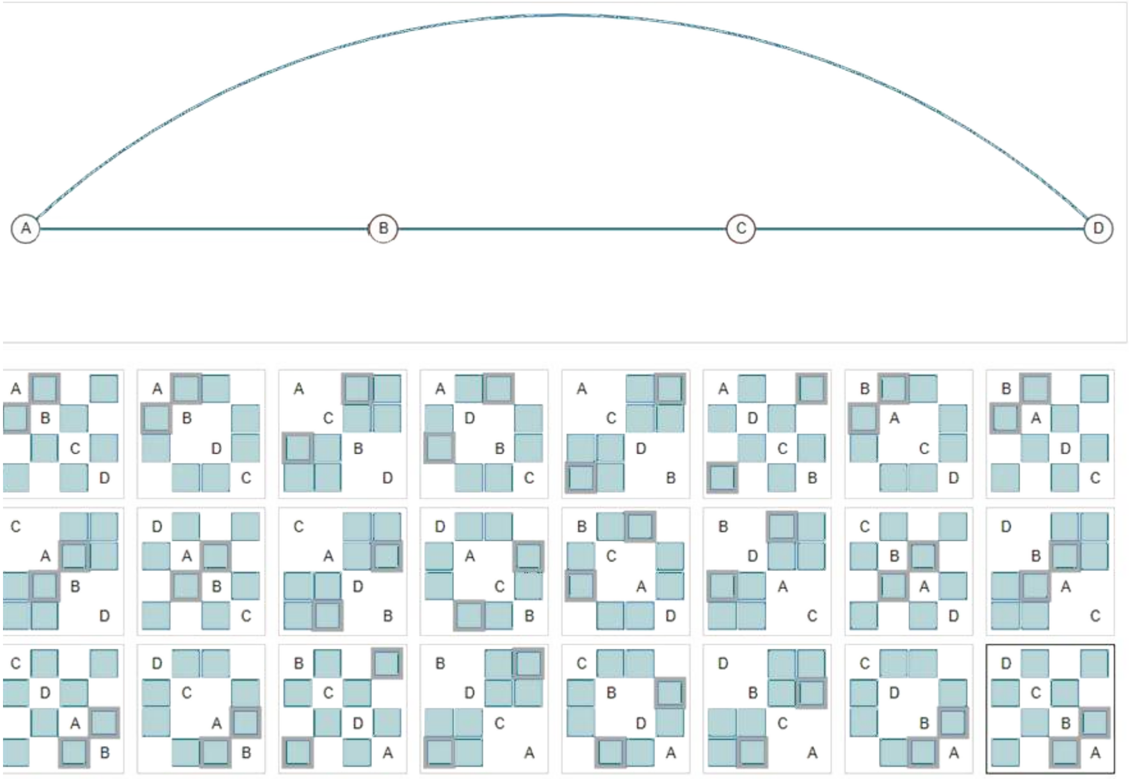
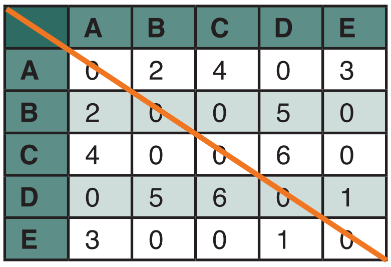


FIGURE 31: Permutation invariance. The same graph is represented by all the adjacency matrices. Image retrieved from Sanchez-Lengeling, B., Reif, E., Pearce, A., & Wiltschko, A. B. (2021). A Gentle Introduction to Graph Neural Networks. Distill, 6(9), e33. https://doi.org/10.23915/distill.00033



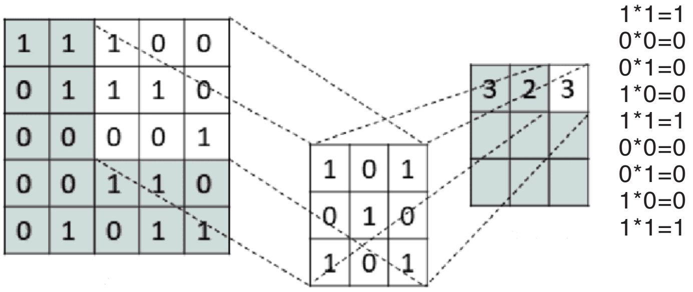
	A	B	C	D	E
A	0	2	4	0	3
B	2	0	0	5	0
C	4	0	0	6	0
D	0	5	6	0	1
E	3	0	0	1	0

FIGURE 32: An adjacency matrix is symmetric (shown by axis of symmetry in orange) so a half adjacency matrix can be used to avoid repeating information to reduce storage. (Own Work)

3.2.3 Graph Neural Networks

A way to solve issues of permutation invariance associated with adjacency matrices for neural networks is to use Graph Neural Networks (GNN). These preserve graph symmetries (permutation invariance) and are optimizable transformations on all attributes (nodes, edges, and global context) of the graph. By employing a 'graph in graph out' architecture, the connectivity of an input graph is preserved while the embeddings are transformed (Sanchez-Lengeling et al., 2021).

Graph Convolutional Networks (GCN) are a type of GNN which use convolutional layers similar to convolutional layers in Convolutional Neural Networks (CNN) to capture patterns inside the data. In a convolutional layer of a CNN, a filter slides over patches of information and aggregates it. It doesn't matter what order the filter slides over it. Hence, it is permutation invariant. Even though GCNs use graphs instead of images, the comparison is still useful to visualize how information is aggregated from neighbour nodes similar to neighbouring pixels in CNNs with the help of filters. This is shown in Figure 33. These filters are learned during the training process and contribute to the ability of the GCN to capture and propagate information through the graph structure. A schematic of a GCN architecture is shown in Figure 34a.



1	1	1	0	0
0	1	1	1	0
0	0	0	0	1
0	0	1	1	0
0	1	0	1	1

1	0	1
0	1	0
1	0	1

3	2	3

1*1=1
0*0=0
0*1=0
1*0=0
1*1=1
0*0=0
0*1=0
1*0=0
1*1=1

FIGURE 33: Image Convolution in CNNs. Edited image. Original retrieved from Dhiman, R., Joshi, G., & Rama Krishna, C. (2021). A deep learning approach for Indian sign language gestures classification with different backgrounds. Journal of Physics: Conference Series, 1950(1), 012020. https://doi.org/10.1088/1742-6596/1950/1/012020

Message passing is used to exchange information from neighbouring embeddings (nodes, edges). This happens in 3 steps. This operation leverages the connectivity of the graph and is permutation invariant as well. This is shown in Figure 34b.

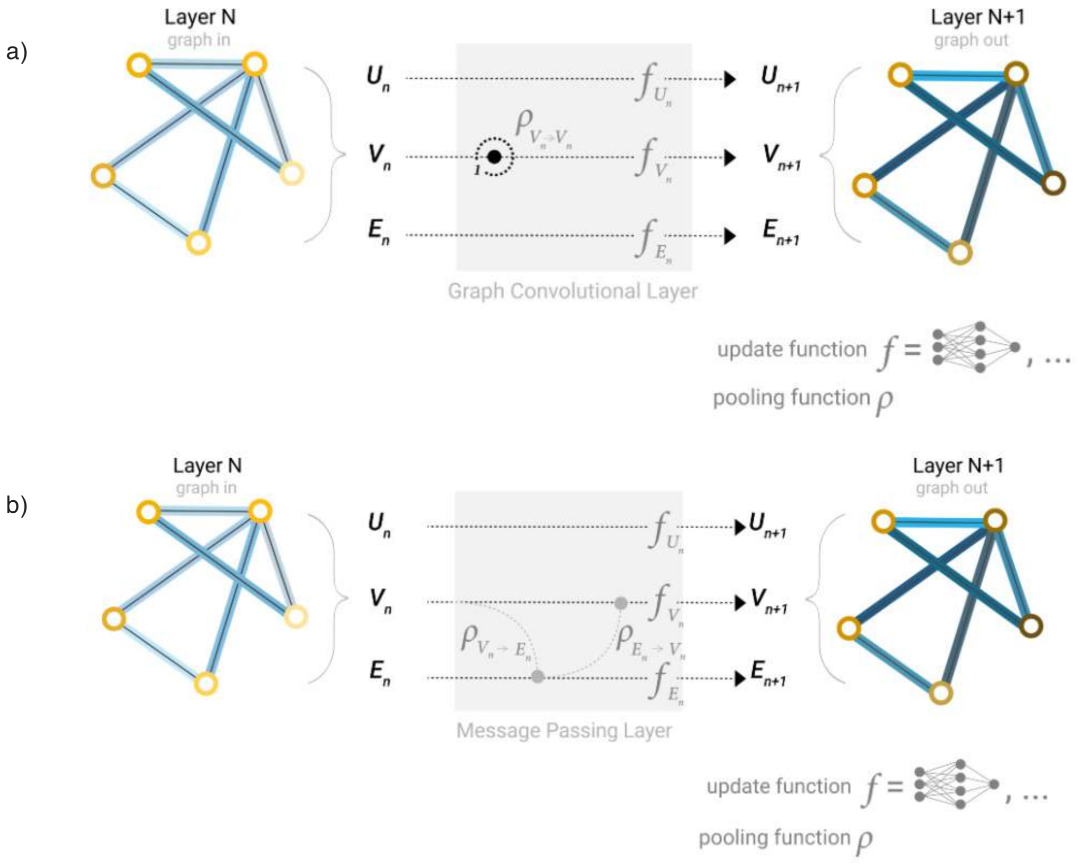


FIGURE 34: a) GCN architecture schematic displaying updates of node representations in a graph through the aggregation of neighboring nodes within a one-degree distance. b) message passes from an edge and its connected nodes to another node. Image retrieved from Sanchez-Lengeling, B., Reif, E., Pearce, A., & Wiltschko, A. B. (2021). A Gentle Introduction to Graph Neural Networks. Distill, 6(9), e33. https://doi.org/10.23915/distill.00033

Message passing happens in 3 steps:

- 1 - Each embedding gathers the neighbouring embeddings/ messages
- 2 - Message aggregation happens by an aggregation function like sum
- 3 - The aggregated messages are updated using an update function that is usually a learned neural network

Information can also be shared between nodes and edges inside the GNN layer.

Through stacking GCN layers, through message passing and convolution, a node can gradually incorporate information from the whole graph. How far a message travels is dependent upon the number of layers. For instance, if there are k layers, a node will capture information from k steps away (Sanchez-Lengeling et al., 2021).

Problem : Embeddings too far apart

An issue arises with nodes being far apart from each other. A solution, called virtual edges, lies in having all nodes pass information to one another. However, this becomes computationally expensive for larger graphs. An alternate solution lies in using the global context vector (or master node) pass information between them building up a richer representation of the graph .

Solution

By conditioning the information of a specific attribute with respect to the others, we can harness them during pooling as all attributes of a graph have learned representations. For a given node, this includes considering information from neighbouring nodes, connected edges, and global information. To condition the new node embedding on these diverse sources of information, one can concatenate them directly. Alternatively, these sources may be mapped to the same space using a linear mapping and combined through addition (Sanchez-Lengeling et al., 2021).

Sanchez-Lengeling et al. note the that if the node and edge don't have information of the same shape or size, linear mapping can be used from the space of one type of embedding to another (2021). Alternatively it can be done by concatenation before the update function (Sanchez-Lengeling et al., 2021). The decision to update which graph attribute first and in which order is an open area of research.

Hyperparameter trends

It is useful to note some trends concerning training data and hyperparameters pertaining to a GNN model (Sanchez-Lengeling et al., 2021)).

Dimensionality

Higher dimensionality of the learned representations for different graph attributes does not necessarily maximize higher bound performance but it does improve lower bound and mean performance.

Layers

The best performing models had two layers but mean performance increased with 4 layers. The lower bound performance tended to decrease after 4 layers due to dilution of node representations form many iterations.

Aggregation function

The sum function performs slightly better than those that used mean or max. There is no one-size-fits-all operation that is universally the best choice for GNN aggregations. 'Mean' proves beneficial when nodes exhibit considerable variability in the number of neighbours or when a normalized view of local neighbourhood features is needed. On the other hand, 'max' is advantageous when emphasizing individual standout features in local neighbourhoods is the objective. The sum operation strikes a balance between mean and max operations. While commonly used in practice, it should be noted that the sum operation is not normalized, which means it can also accentuate outliers.

Attribute Communication

Increasing the communication between graph attributes (nodes, edges, globals), the better the performance

Conclusion

For sparsity, to reduce the space, one approach that could have been taken was to use a half-adjacency matrix instead of a full one to reduce the input size for computational efficiency as shown in Figure 32. This works for undirected adjacency matrices as they are symmetric across the diagonal (Sterrenberg, 2023). An alternate approach is using adjacency lists. Given the number of edges will be significantly fewer than the total entries in an adjacency matrix, computation and storage is skipped for the disconnected segments of the graph (Sanchez-Lengeling et al., 2021). Another approach was to simplify the edge relationships and group them into polyedges. This technique simplifies the data. See Section 5.2 for details.

3.3 GENERATOR

3.3.1 Generative Design

“Generative design varies the parameters of the problem definition while parametric design varies parameters of the geometry directly.” (Matejka et al., 2018 cited in Oh et al., 2019)

The quote above describes how generative design is used as design generator rather than a design parametrizer allowing the designer the generate different boundary conditions for the generation of a variety of optimized designs under different boundary conditions (Oh et al., 2019). There are different types of generative models including Variational Autoencoders, Generative Adversarial Networks, and Reinforcement Learning models.

3.3.2 VAE

An Autoencoder is an unsupervised embedding algorithm which uses a 3 tiered system – an encoder, latent space, and a decoder. An input dataset is given. The encoder compresses this input into a lower dimensional structure only retaining its most prominent features. The decoder then reconstructs that as accurately as possible to give the output. Since the latent space is usually sparse which means sampling a latent vector can cause issues since data is not present there. This shortfall is addressed by Variational Autoencoders (VAE). It regularizes the latent space with a probabilistic distribution with a mean and covariance. To maintain predictability in the latent space, Kullback-Liebler (KL) divergence is added between the distribution of the latent space and a standard Gaussian (Regenwetter et al., 2022).

Given an observed variable x, a vanilla VAE introduces a continuous latent variable z and assumes that x is generated from z. This relationship is expressed as:

$$p_\theta(x, z) = p_\theta(x \mid z) p_\theta(z)$$

Here, θ represents the model parameters. The term p_θ(z) is the prior distribution, commonly a simple Gaussian distribution. The conditional distribution p_θ(x I z) describes the process of generating x from z and is typically modeled using a deep nonlinear neural network (Zhang et al., 2016).

What is The Loss Function in a VAE?

Evidence Lower Bond (ELBO) is the loss function of a VAE. It is a combination of 2 two terms, the reconstruction loss and the regularization term, often called the KL divergence. The ELBO is a lower bound on the log likelihood of the data and is used as the objective function to be maximized during training(Burgess et al., 2018).

$$\text{ELBO}(\varphi) = E_{q\varphi(z|x)}[\log p_\theta(x \mid z)] - D_{KL}(q_\varphi(z \mid x) \mid\mid p(z)).$$

What is the Reparametrization Trick?

The prior p(z)) and the posterior q_θ(z I x) distributions are modeled as Gaussians with diagonal covariance matrices. Typically, the prior is set to an isotropic unit Gaussian, N(0,1) (Burgess et al., 2018).

During training, we need to backpropagate with respect to θ to minimize the ELBO. However, since the ELBO depends on z, which is sampled from the distribution q_θ(z I x), we face a challenge. The Reparameterization Trick (Kingma et al., 2015) addresses this by decomposing z into a deterministic component and a stochastic component, enabling safe backpropagation through the sampling step.

$$z = \mu + \sigma \odot \in$$

where $\in \sim N(0, 1)$, and μ and σ are the mean and the standard deviation of $q_\varphi(z \mid x)$. ∈ is a standard Gaussian variable that plays a role of introducing noise, and ⊙ denotes an element-wise product (Zhang et al., 2016).

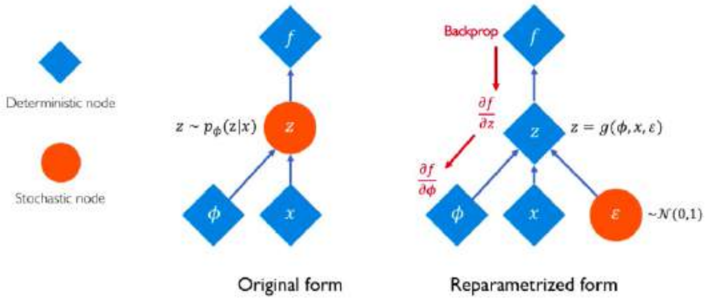


FIGURE 35: Reparametrization trick. Image retrieved from https://www.youtube.com/watch?v=rZufA635dq4&ab_channel=AlexanderAmini

Latent space

After the encoder of the VAE maps input data to the latent space, the distribution of the latent variables is typically modelled as a multivariate Gaussian distribution (as each dimension of the latent space has its own mean and variance).

Ideally, the unique latent dimensions of the VAE should represent unique underlying features that provide variations to the generated data. This unfortunately is not the case with the Vanilla VAE, or simply, VAE. Changes in a single latent dimension often do not result in a single feature variation. This can be seen in Figure 36a. This makes the process of sampling from the latent unpredictable and chaotic so if it is hard to control features that the user requires.

3.3.3 Variations of VAEs

For the issue highlighted above, variations to the VAE can be studied. Disentanglement provides a solution to this. According to Pastrana, this involves a combination of two types of VAEs (2022).

- β-VAE: adding a term on the KL divergence in the ELBO
- C-VAE: conditioning the autoencoder by introducing labels to the data

β-VAE

Beta Variational Autoencoder (β-VAE) allows control over disentanglement. This allows singular latent dimensions to control singular characteristics allowing for greater control over the generated designs. By addition a term β on the regularization term whereby providing a weightage. If β is too large, there is larger disentanglement but the reconstruction loss term becomes smaller due to which the output loses fidelity. If β is too small, there is less disentanglement so the reconstruction loss term is large so the single latent dimensions may not be representative of single characteristics. The lowest value of β is 1 which corresponds to a Vanilla VAE. β needs to be calibrated either through quantitative methods by adding a linear classifier to the trained VAE – or through qualitative methods such as visual heuristics (Pastrana, 2022).

It should be noted, however, that β-VAE is a fairly new concept that was introduced in 2017 (Higgins et al., 2017). According to Fil et al., its brittleness and difficulty in defining disentanglement, and the inconsistency of metrics across datasets and models has been described (Fil et al., 2021). This view, however, is contradicted by Higgins et el., who compares it to GANs and describes how it is very stable to train unlike InfoGAN and DC-IGN, requiring no design decisions or assumptions about the data. They go on to describe how β-VAE consistently and robustly discovers more latent factors and learns cleaner disentangled representations even on challenging datasets such as celebA (Higgins et al., 2017).

Figure 36b shows that the β term is not enough to allow single latent dimensions to learn unique features. It is also necessary to introduce labels into the dataset as done with C-VAEs. On a dataset of 60,000 images of the MNIST database, the methodology produced 3 interpretable visual features of the digits – their tilt, width, and line weight (Pastrana, 2022).

C-VAE

Conditional Autoencoder (C-VAE) allows for the addition of labelled (supervised) data to the input dataset. This allows the latent space to learn the interpretable latent space. While β-VAE is unsupervised learning, CVAE introduces supervised learning. This allows the possibility of generating outputs specific to a particular class label.

By introducing labels during the training process of the variational autoencoder, the latent space can be conditioned to output novel samples specific to the input condition. This means that instead of just mapping the feature x, a label y is also added to the encoder and decoder (Ivasiuk & Misino, 2020).

$$p(x, z, y) = p(x \mid z, y)p(z \mid y)$$

The conditional VAE tries to maximize:

$$\log p\theta(x \mid y) = \int_z \log(p(x \mid z, y)p(z \mid y))dz$$

while the loss function to minimize is:

$$\mathrm{ELBO}(\varphi) = \mathrm{E}_{q\varphi(z|x,y)}[\log p(x \mid z, y)] - \mathrm{D}_{KL}(q_\varphi(z \mid x, y) \mid\mid p(z \mid y)).$$

Conclusion

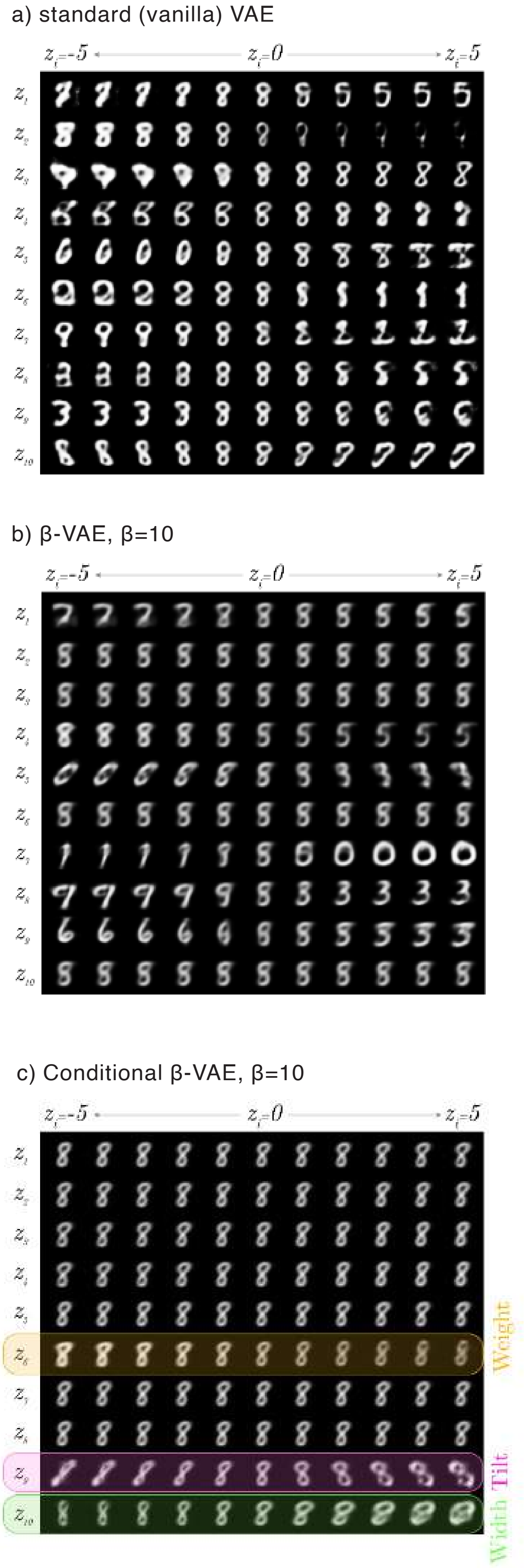
The combination of a C-VAE and β-VAE produces the best results as shown in Figure 36c - single latent dimensions represent unique features.

3.3.4 Generative Adversarial Network (GAN)

Generative Adversarial Network (GAN) is deep generative model that uses a discriminator and a generator that work against each other (maximizing and minimizing the loss function respectively to generate novel designs. Oh et al. (2019) have used GANs for optimization of the design of a 2D wheel based on three criteria: aesthetic quality, diversity, and robustness. Rawat & Shen (2019) have used WGANs for 3D structural topology optimization tasks with empolying CNNs as predictive surrogate models.

3.3.5 Reinforcement Learning (RL)

Reinforcement Learning (RL) is a branch of Machine Learning in which an agent learns decision-making by interacting with an environment (Regenwetter et al, 2022). The agent receives rewards or penalties based on its actions, aiming to maximize cumulative rewards over time through trial and error. Unlike other ML approaches, RL doesn’t rely on labels and is well-suited for tasks where the optimal strategy is uncertain. It finds applications in solving sequential decision-making problems across various domains. Existing research involves optimizing the structure and material distribution of a 2D wheel to meet performance criteria such as compliance minimization and similarity maximization (Jang et al., 2022).



a) standard (vanilla) VAE

b) β-VAE, β=10

c) Conditional β-VAE, β=10

FIGURE 36: A disentangled latent space with distinguishable features is represented only in c) where a conditional β-VAE is used. The features learnt are weight, tilt, and width. Image retrieved from Pastrana, R. (2022). Disentangling Variational Autoencoders. https://doi.org/10.48550/arxiv.2211.07700

3.3.6 Surrogate Model

A surrogate model is a predictive model that approximates a more computationally expensive real model. It requires inputs and outputs through which it generates a trend representative of the structure when the actual relationship between the two is unknown or far too computationally expensive. It may use machine learning but does not necessarily require it. The type of surrogate model that is appropriate for a particular case depends upon the data that is given and the desired model that is required. Its construction entails 3 steps (Williams & Cremaschi, 2019).

- 1 - Sample point selection
- 2 - Training/ optimization of model parameters
- 3 - Evaluation of model

Why is a surrogate model used for VAEs?

In the context of VAEs, a surrogate model can be used to save computational time to perform the Perfomance Evaluation on the generated output from the latent space. Instead of performing an FEA analysis again, the surrogate model may be able to predict the performance. This methodology had been used by Pavlidou (2022) and Sterrenberg (2023). To study where it performed in the architecture of the generator, a surrogate model was connected in one option to the output of the encoder, and in one to the output of the decoder. It was found that when connected to the output of the decoder, it provided promising results.

Selection of a surrogate model

There are different types of surrogate models that can be used depending on the type of relationship between the input and output data, and the desired application for the model. Williams & Cremaschi document their findings on 8 different surrogate models. A diverse set of datasets were generated to model comprehensive evaluation of the 8 models across different scenarios to develop insights about their performance. It was found that Multivariate Adaptive Regression Spline Models (MARS) and single hidden-layer feed-forward Neural Networks (ANN) are well-suited for accurately approximating the design space, while Random Forest (RF) models are particularly effective for guiding optimization efforts towards optimal solutions within the design space (Williams & Cremaschi, 2019). It is important to note, however, that the performance of the surrogate model depends upon the type of dataset and nature of the optimization problem. For instance, a linear model may be used to model linear relationships may easily be interpretable whilst not being suitable to non-linear relationships. Gaussian Processes may be useful in that case to quantify uncertainties.

Since the dataset and nature of optimization dictate the use of surrogate models, it would be natural to draw on conclusions from similar design problems:

Artificial Neural Network (ANN)

Javanmardi & Ahmadi-Nedushan use the Optimized Artificial Neural Network (OANN) for optimizing the structure of a double-layer barrel vault. The OANN is a special type of ANN which is optimized to minimize the risk of over-fitting, which is a significant issue in learning neural networks. It is a suitable surrogate model for solving structural optimization problems efficiently and accurately (Javanmardi & Ahmadi-Nedushan, 2023).

White et al. use a Gaussian Basis Network (GBF) which is a type of ANN in a topology optimization problem. It was trained using the Sobolev norm, which involves training the network with both function data and derivative data. This is valuable for sensitivity analysis and optimization in topology problems where derivative information is important. The model parameters being determined via optimization. This is because if the model has an objective function that is dependent on a variable, The rate of change of the variable (derivative) needs to considered in order to evaluate how the objective function behaves.

Shape-Preserving Response Prediction (SPRP)

Leifsson & Koziel use a physics based surrogate modelling approach called Shape-Preserving Response Prediction (SPRP) to optimize the shape of an aerodynamic component. SPRP is used to find the optimal configuration or parameters for a given system or device where the goal is to maximize or minimize an objective function while satisfying constraints (Leifsson & Koziel, 2016).

Conclusion

To conclude, two types of surrogate models have been identified which can potentially be used. SPRP aligns with the optimization type (shape optimization) as well as being a physics-based approach it may be able to model the complexity of the seismic simulation. ANNs are useful for accurate approximation of the design space whilst OANNs reduce the risk of overfitting.

3.3.7 Gradient Descent (Optimization)

Gradient Descent Optimization is an optimization algorithm which minimizes the loss function. It does this by computing the gradient of the function at a given point and moves opposite to the direction of the slope increase by that computed amount. The process entails passing the training set through the hidden layers of the neural network and then updating the layer parameters by computing gradients derived from the training samples within the training dataset (Patrikar, 2019).

How quickly the model learns determines the learning rate; a balance between efficiency and stability is sought. If the learning rate is too high, the model is unstable and may overshoot the global minima. If the learning rate is too low then it may get stuck in the local minima and be very sensitive to noise. This is shown in Figure 37.

There are 3 different methods of Gradient Descent optimization:

Batch Gradient Descent

In Batch Gradient Descent, all gradients are considered simultaneously, and the sum of errors is calculated to update all weights in a single epoch (Patrikar, 2019; Roy, 2020). While effective for convex curves, this method becomes computationally intensive with inefficient in large batches, such as those encountered in sizable datasets (Patrikar, 2019).

Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) deals with the issue of BGD. With SGD, one example is processed at a time, and the weights are updated after calculating the gradient for each starting point, completing one epoch. Fluctuations allow it to jump to possibly better minima but at the cost of overshooting (Ruder, 2017).

Min-batch Gradient Descent

Mini-batch Gradient Descent offers a balanced approach by incorporating advantages from both BGD and SGD. This method achieves two crucial goals. It ensuring more stable convergence by reducing the variance in parameter updates. It also enhances the efficiency of computing gradients for mini-batches by capitalizing on optimized matrix operations prevalent in advanced deep learning libraries. While typical mini-batch sizes range between 50 and 256, the choice may vary across different applications. Overall, mini-batch gradient descent stands out as the preferred algorithm for training neural networks (Ruder, 2017).

3.3.8 Evolutionary Algorithm (Optimization)

Evolutionary Algorithms are heuristic-based methods used to solve complex problems that resist efficient polynomial-time solutions. These algorithms mimic natural selection, where fitter individuals thrive and less fit ones are

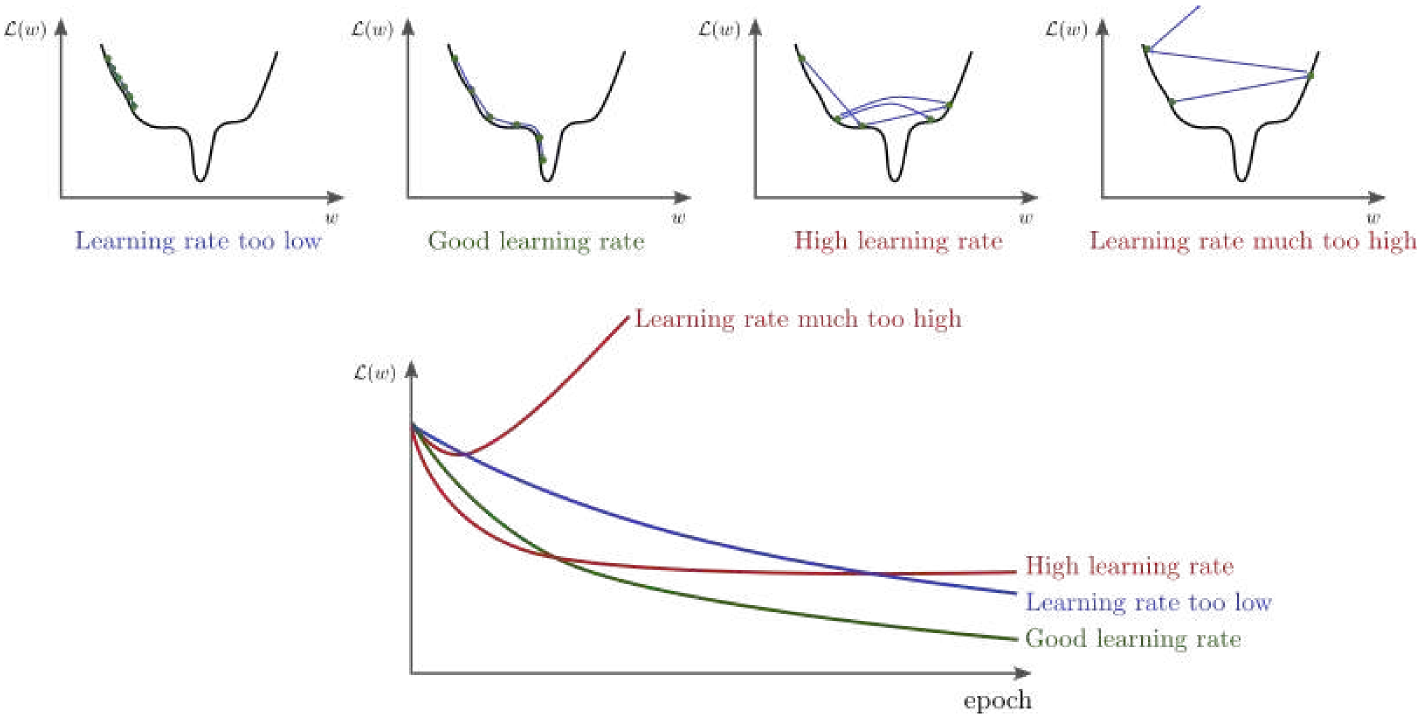


FIGURE 37: Different learning rates. Image retrieved from CS231n Convolutional Neural Networks for Visual Recognition. https://cs231n.github.io/neural-networks-3/

eliminated. The process involves four key steps: initialization, selection, genetic operators, and termination. These steps correspond to facets of natural selection, allowing for modular algorithm design. Evolutionary Algorithms find application in combinatorial problems and can complement other methods by providing optimal starting points for further processing (Soni, 2018).

3.3.9 Encoding Strategies

In this paper, 3 encoding strategies were considered for encoding the labels of the Conditional VAE:

Label Encoding

In label encoding, each category/ label within a categorical variable is assigned a unique integer value (Al-Shehari & Alsowail, 2021).

In label encoding, it is standard to assign integers to categorical variables instead of floats. This approach is used because label encoding converts categorical data into a numerical format suitable for machine learning algorithms. Using floats can create unnecessary complexity and may cause the algorithms to misinterpret the data. Therefore, it is recommended to use integer values for label encoding.

For example, for a categorical variable ‘City’ with the categories ‘New York’, ‘London’, and ‘Paris’, the label encoding might look like this:

‘New York’ : 0

‘London’ : 1

‘Paris’ : 2

One Hot Encoding

One-hot encoding is a technique in machine learning and data processing that converts categorical variables into a numerical format. In this approach, each category is represented by a binary vector with a single “hot” (1) bit and the rest “cold” (0). This method helps machine learning algorithms accurately interpret categorical data by eliminating any false ordinal relationships between categories (Al-Shehari & Alsowail, 2021). It is commonly used to enhance the performance of classification tasks involving categorical variables. Different integer values of Label Encoding may introduce biases. This is avoided by One-hot-encoders due to binary data. The dimensionality of the One-hot encoding depends upon the number of labels that are required. Therefore, for a high number of labels, the encoding becomes high dimensional and sparse which may not be desirable.

For instance, the ‘City’ labels would be represented as follows;

‘New York’ : [1 0 0 0]

‘London’ : [0 1 0 0]

‘Paris’ : [0 0 1 0]

Sinusoidal Positional Encoding

Sinusoidal positional encoding is a technique used in Transformer models to incorporate information about token positions in a sequence. It involves using sine and cosine functions of varying frequencies to create positional embeddings (Vaswani et al., 2023). These embeddings have the same dimensionality as the input embeddings and are added together. By doing so, the model learns to attend to relative positions within the sequence without relying on recurrence or convolution. This approach has contributed to the Transformer’s success in natural language processing tasks (Vaswani et al., 2023).

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

where pos is the position and i is the dimension of the positional encoding corresponding to a sinusoid.

3.3.10 Sensitivity Analysis

The Sensitivity Analysis is used to analyze the effect of hyperparameters on the output of the model. It can be used to study how changes in hyperparameters affect the performance metrics of the model, such as accuracy, precision, recall, or any other relevant metric. This information can guide the hyperparameter optimization process by focusing on tuning the most influential hyperparameters to achieve the desired model performance. This can involve using techniques such as grid search, random search, or Bayesian optimization to systematically explore the hyperparameter space and identify the optimal values that lead to the best model performance.

In VAEs, hyperparameters such as the learning rate, batch size, number of latent dimensions, and the weight of the KL divergence term in the ELBO objective function can have a significant impact on the performance of the model. Optimizing these hyperparameters is crucial for achieving the best performance of the model.

4.1 INDIVIDUAL WORKFLOWS

For the holistic workflow to see all individual parts working together, See Figure and Figure 6 in Section 1.4.

4.1.1 Geometry Generation

The Geometry Generation involved generation of csvs with force densities in python. These were used to make COMPAS meshes that were saved and imported into Grasshopper. The COMPAS meshes were converted into a format for Grasshopper for visualization and Performance Evaluation. The main Geometry Generation workflow was carried out in python rather than Grasshopper to save computational time and provide easier connectivity with other scripts for Data Structuring, Performance Evaluation, and the Generator. A large dataset of meshes (10,000) with variable force densities for a given constant loading (40.0) and a set footprint (15m x 10m) with an equal mesh density (1 division per unit metre).

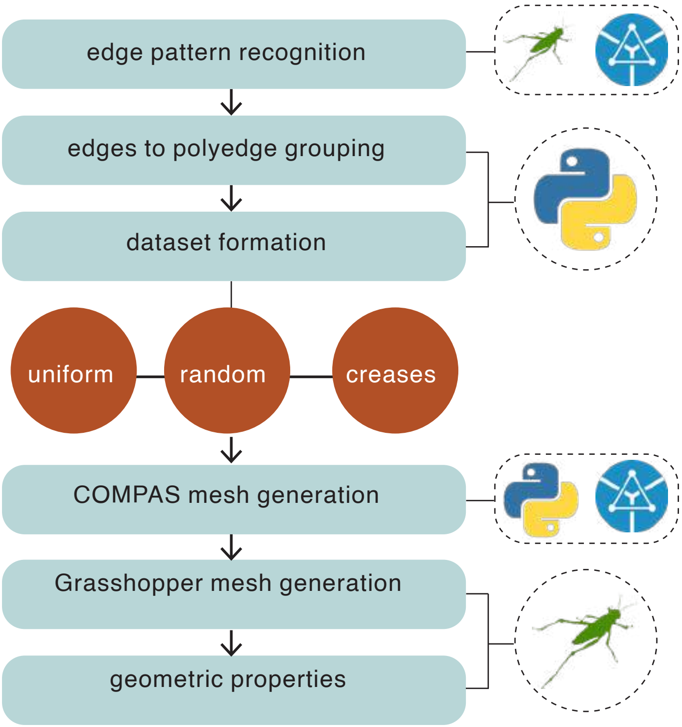


FIGURE 38: Geometry Generation Workflow. (Own Work)

4.1.2 Performance Evaluation

Once the Geometry Generation was complete, the Performance Evaluation was performed using python and Grasshopper. Firstly, the seismic weight of the building was calculated. The modal analysis was performed to check the dominant modes of vibration of the structure. The Time Period from the dominant modes was used for performing Response Spectrum Analysis, a linear dynamic analysis to derive the applied forces for the dominant modes. The seismic combination, Square-Root-of-Sum-of-Squares (SRSS), was used to derive the combined loading for dominant modes followed by an FE analysis conducted in Grasshopper with Karamba. A PGA of 0.2g was used to simulate seismic activity in lateral directions in different simulations. The performance metrics in both directions were compared and the worst performing one was used.

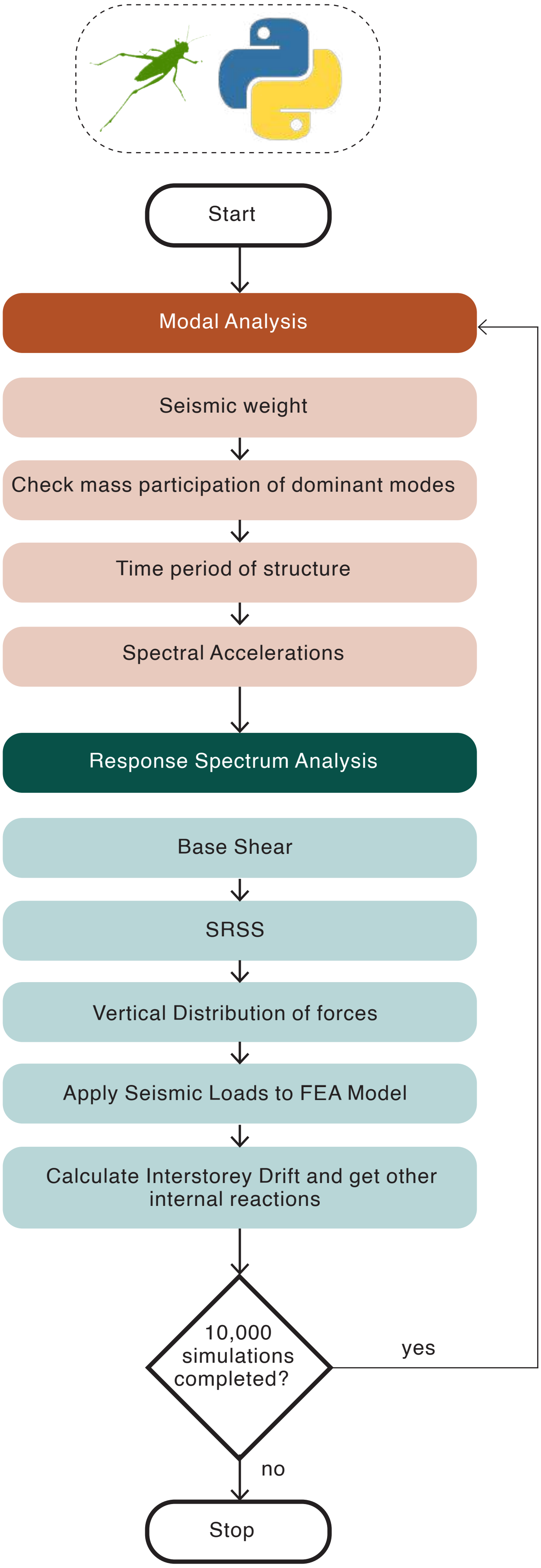

Start
Modal Analysis
Seismic weight
Check mass participation of dominant modes
Time period of structure
Spectral Accelerations
Response Spectrum Analysis
Base Shear
SRSS
Vertical Distribution of forces
Apply Seismic Loads to FEA Model
Calculate Interstorey Drift and get other internal reactions
10,000 simulations completed?
yes
no
Stop

FIGURE 39: Workflow of Performance Evaluation. (Own Work)

4.1.3 Generator

The Generator consisted of a pipeline that used a Conditional Variational Autoencoder to learn the distribution of the dataset of 10,000 samples generated through the Geometry Generation process. The same training features that was used to train the VAE was used for training the surrogate model and the performance metrics from the Performance Evaluation were used as labels. This way it was able to predict the performance of meshes that had not been seen during training. Lastly, optimization was carried out that used the performance score of the new geometries sampled from the VAE generated using the surrogate model in an optimization loop where the latent space of the VAE was explored for better solutions. In the context of the gradient descent, this meant the gradient of the performance score.

From the latent space of the VAE, new meshes could be sampled whose seismic performance was predicted by

Neural Network surrogate models.

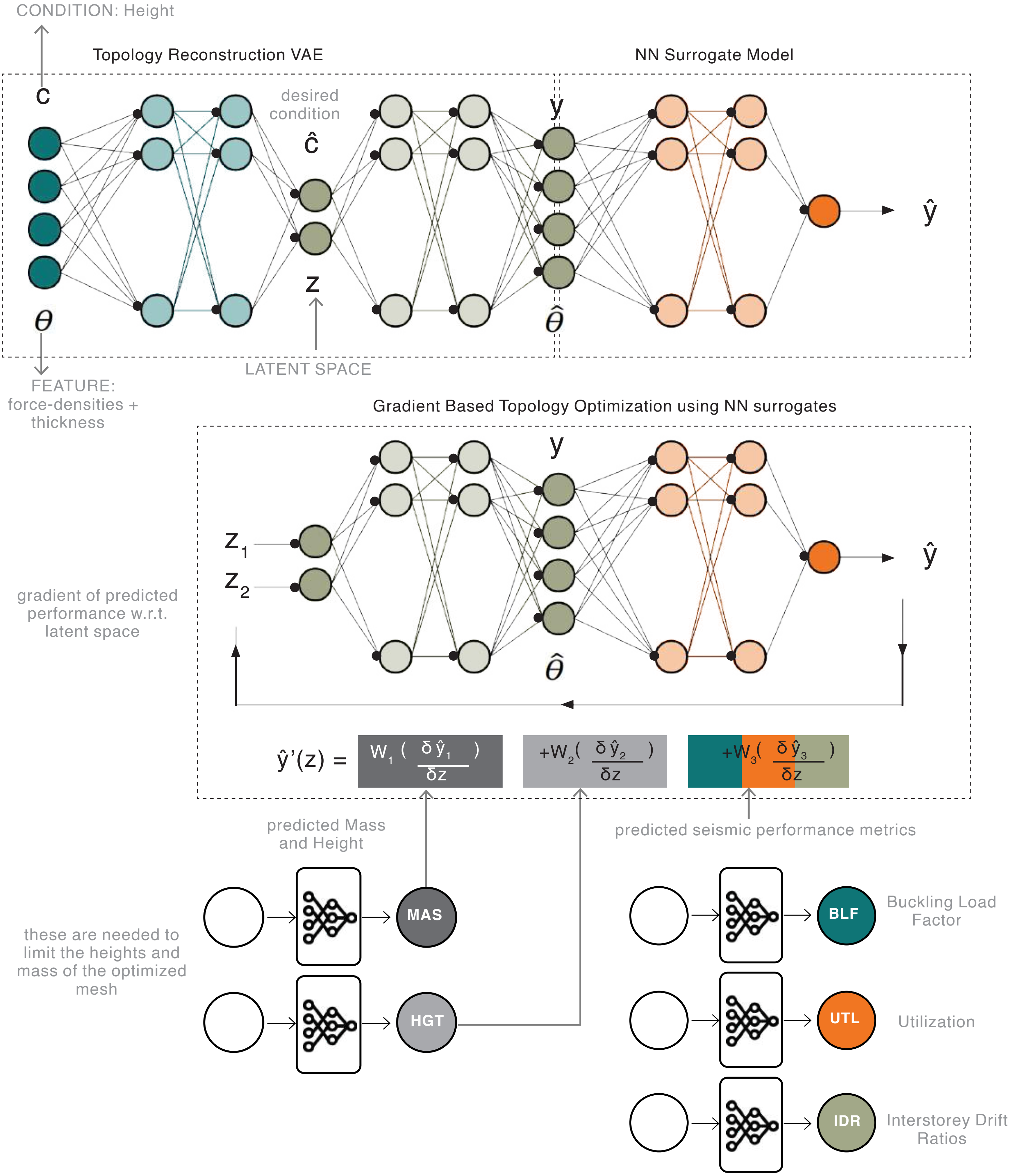

CONDITION: Height
Topology Reconstruction VAE
NN Surrogate Model
c
desired condition
ĉ
y
ŷ
z
θ
θ̂
LATENT SPACE
FEATURE: force-densities + thickness
Gradient Based Topology Optimization using NN surrogates
y
z_1
z_2
ŷ
θ̂
gradient of predicted performance w.r.t. latent space
$\hat{y}'(z) = W_1\left(\frac{\delta \hat{y}_1}{\delta z}\right) + W_2\left(\frac{\delta \hat{y}_2}{\delta z}\right) + W_3\left(\frac{\delta \hat{y}_3}{\delta z}\right)$
predicted Mass and Height
predicted seismic performance metrics
these are needed to limit the heights and mass of the optimized mesh
MAS
HGT
BLF
Buckling Load Factor
UTL
Utilization
IDR
Interstorey Drift Ratios

FIGURE 40: Overall Workflow of the Generator connecting the CVAE to the surrogate model and optimization through Gradient Descent. Inspired by Gladstone, R. J., Nabian, M. A., Keshavarzzadeh, V., & Meidani, H. (2021). Robust Topology Optimization Using Variational Autoencoders (arXiv:2107.10661). arXiv. http://arxiv.org/abs/2107.10661

4.1.4 VAE

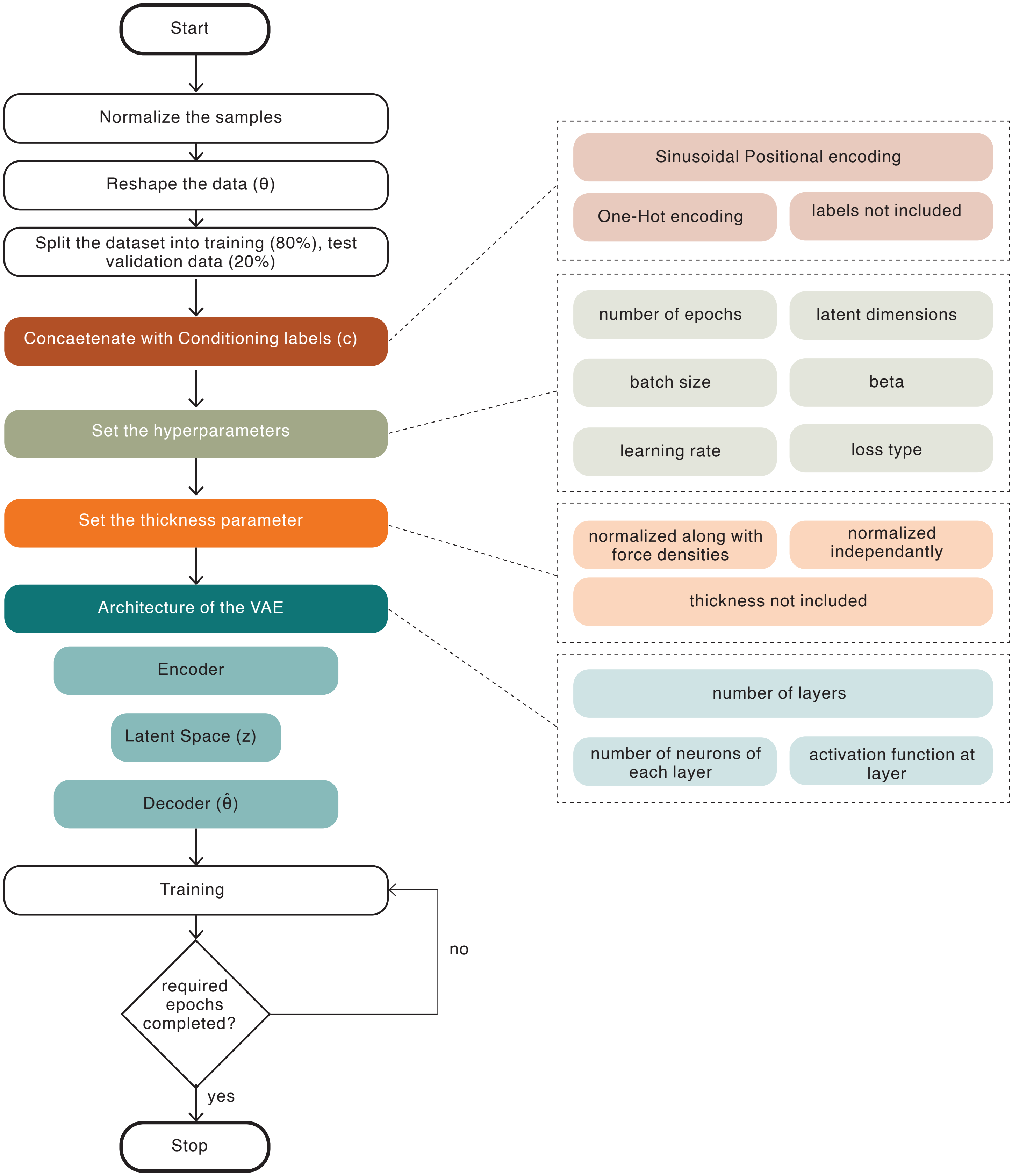


FIGURE 41: Workflow for training the VAE / CVAE. (Own Work)

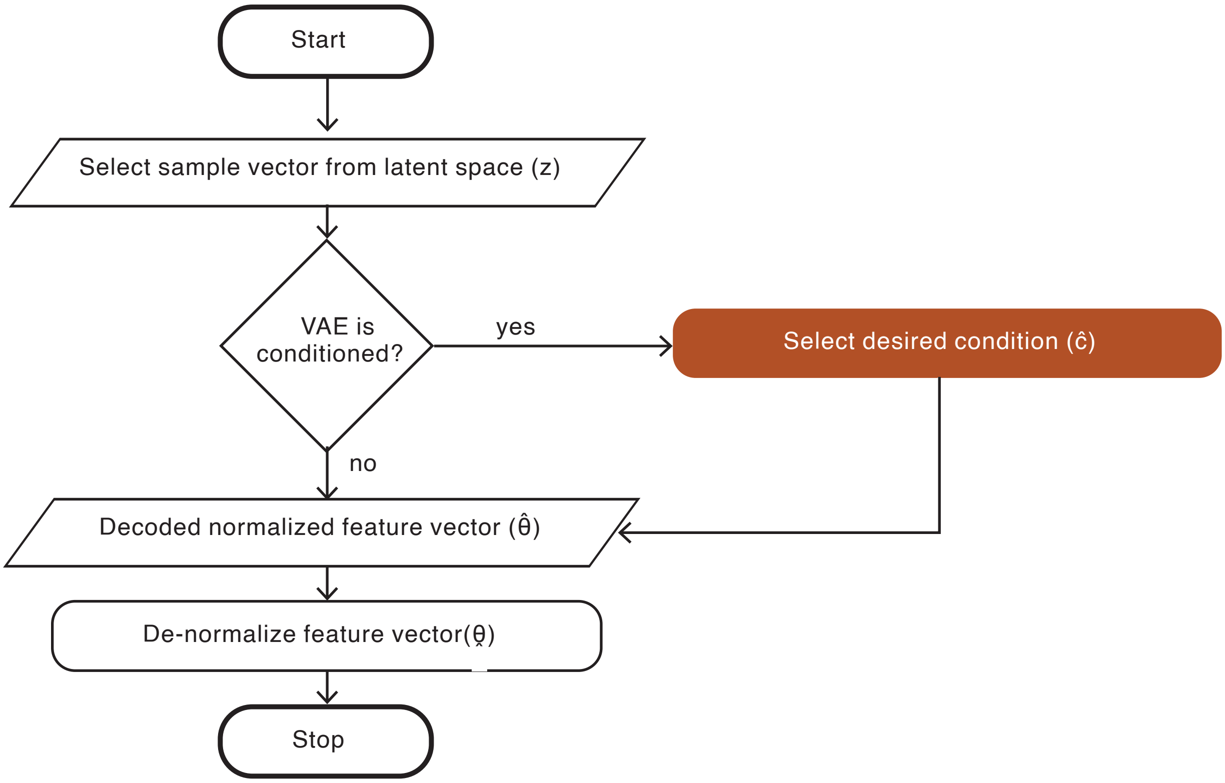

Start
Select sample vector from latent space (z)
VAE is conditioned?
yes
Select desired condition (ĉ)
no
Decoded normalized feature vector (θ̂)
De-normalize feature vector(θ)
Stop

FIGURE 42: Workflow for sampling from the VAE / CVAE. (Own Work)

For the Variational Autoencoder, once the dataset of 10,000 samples had been generated and evaluated in Karamba for seismic performance, the data was restructured into an array of shape (10000, 27) where n is the number of samples, and 27 are the number of force densities of each sample. An additional feature was added to each sample later representing the thickness of the mesh so the shape changed to (10000, 28). The data was normalized so the maximum value was kept 1. Details of normalization of the thickness feature will be covered ahead in the Section 7.1.2. Height labels were later added to turn the VAE in to a Conditional VAE and 2 label encoding strategies were explored - One Hot encoding and Sinusoidal Position encoding where a dimensionality of 58 and 28 were explored respectively concatenated with the 28 dimensional feature vector.

10 random samples were excluded from the training to be used as test data to test the trained model at the end on unseen data. A validation split of 20% was used so 80% was used as training data while the remaining was used for validation. The hyperparameters were tuned over the course of several simulations; these are also shown in Figure 41. These include the number of epochs, batch size, learning rate, the latent dimension, and the beta term. The loss type used in all simulations was Mean Square Error as the features consisted of continuous data rather than discrete, which otherwise may have required Binary Cross Entropy (BCE) instead.

Different architectures of the VAE were tested. After the required number of epochs were completed, the training was stopped. The aim was to end up with the simplest architecture (least number of layers and least number of neurons in each layer) for computational efficiency.

To determine whether the model was sufficiently trained or not, validation loss and training loss were plotted against epochs. If the model was sufficiently trained, there would be convergence. If not, the number of epochs was increased or the training rate was increased.

4.1.5 Surrogate Model

For the overall workflow to function, it was important that the surrogate model used the same feature dataset that was input into the VAE. Similar to the VAE, the feature dataset was normalized and reshaped into (10000, 27) (without thickness) and (10000,28) with thickness. Similar to the VAE, for all surrogate model simulations, to determine whether the model was sufficiently trained, validation and training loss curves were plotted against the epochs to look at convergence. If there was no convergence, the number of epochs was increased or the training rate was increased before an assessment was made on the result. A single surrogate model was trained on the three SSS metrics as well separate surrogate models for each SSS metric in addition to Mass and Height as shown in Figure 43.

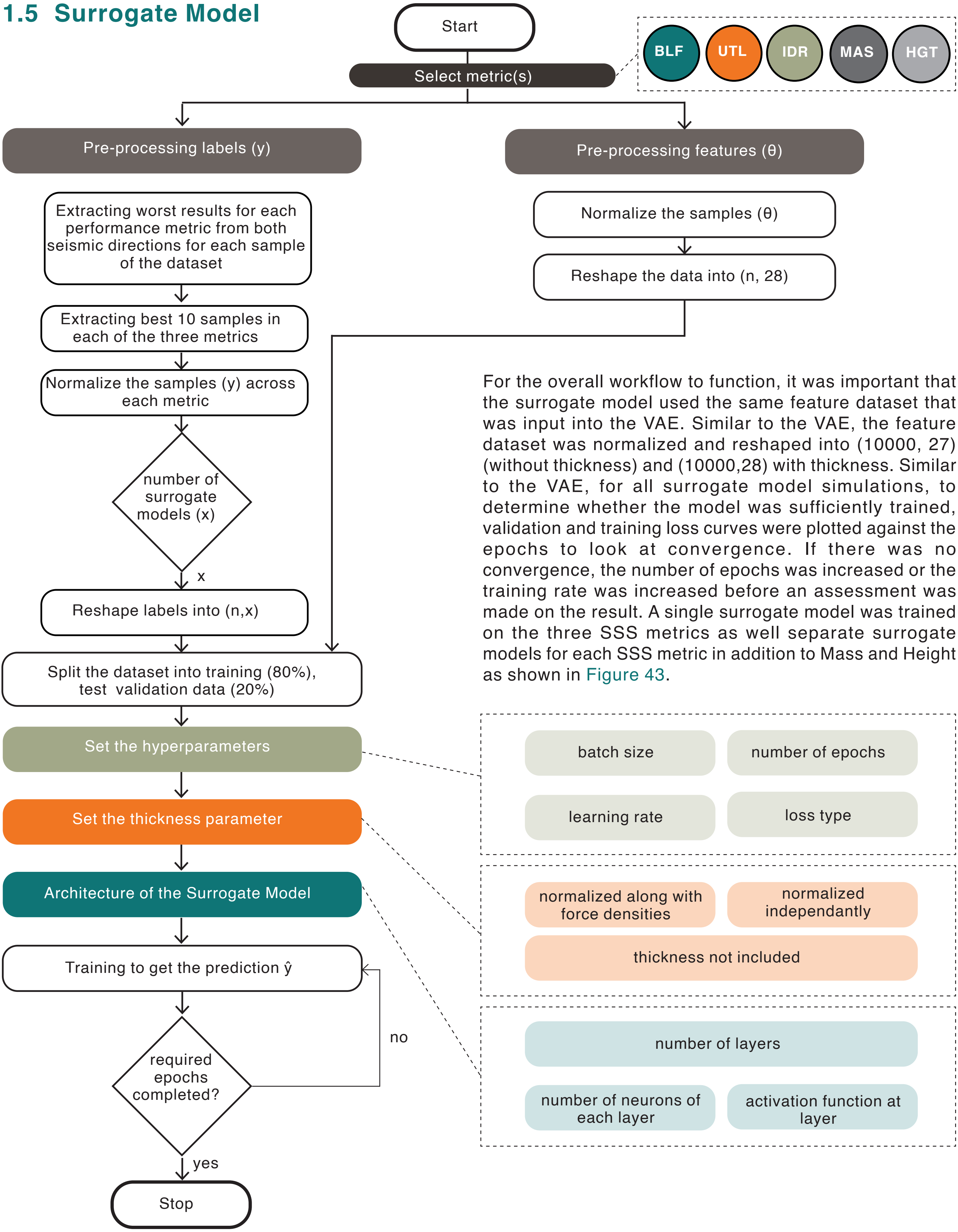

Start
Select metric(s)
BLF
UTL
IDR
MAS
HGT
Pre-processing labels (y)
Extracting worst results for each performance metric from both seismic directions for each sample of the dataset
Extracting best 10 samples in each of the three metrics
Normalize the samples (y) across each metric
number of surrogate models (x)
x
Reshape labels into (n,x)
Pre-processing features (θ)
Normalize the samples (θ)
Reshape the data into (n, 28)
Split the dataset into training (80%), test validation data (20%)
Set the hyperparameters
batch size
number of epochs
learning rate
loss type
Set the thickness parameter
normalized along with force densities
normalized independantly
thickness not included
Architecture of the Surrogate Model
number of layers
number of neurons of each layer
activation function at layer
Training to get the prediction ŷ
required epochs completed?
no
yes
Stop

FIGURE 43: Surrogate Model Workflow. (Own Work)

4.1.6 Gradient Descent Optimization

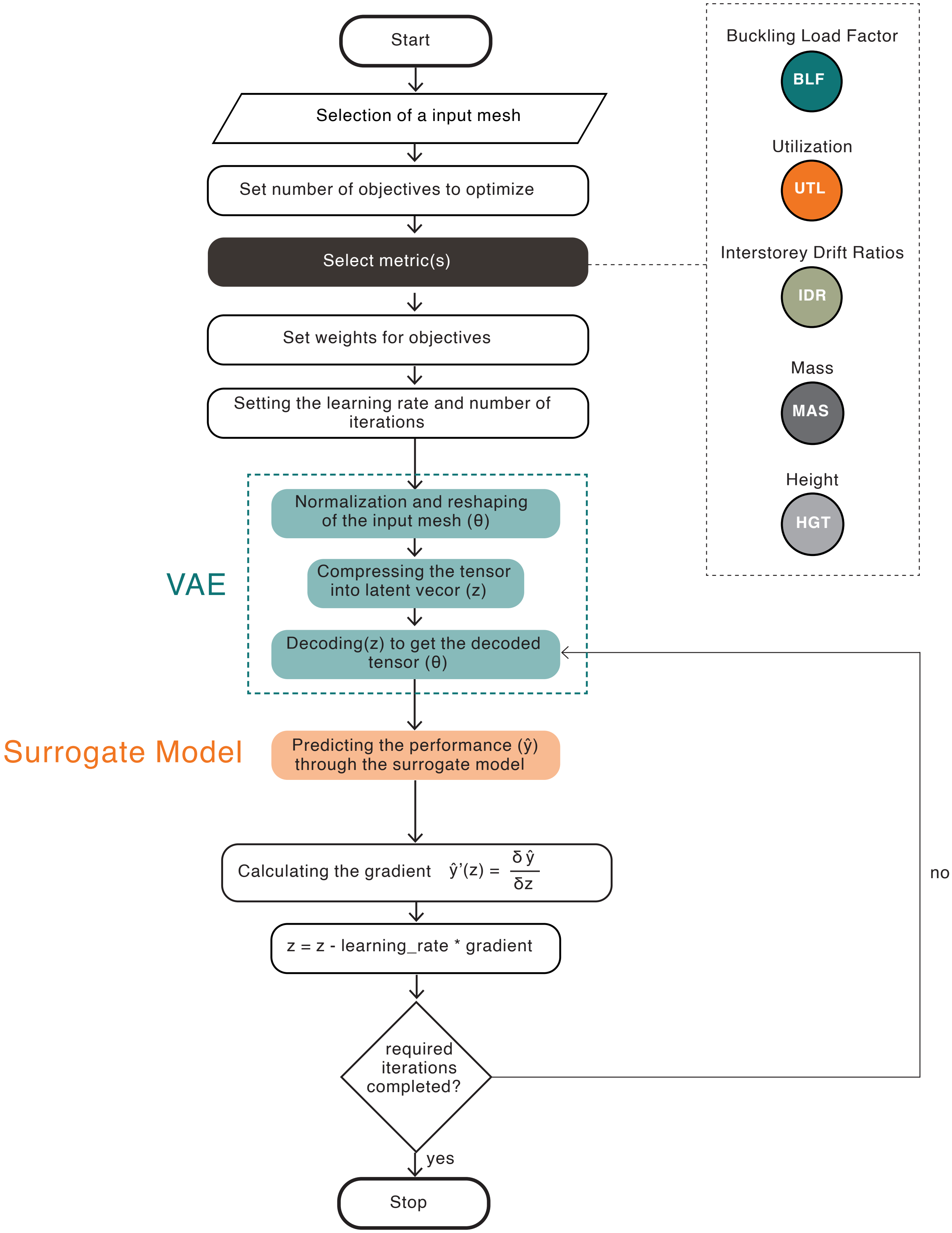

Start
Selection of a input mesh
Set number of objectives to optimize
Select metric(s)
Set weights for objectives
Setting the learning rate and number of iterations
VAE
Normalization and reshaping of the input mesh (θ)
Compressing the tensor into latent vecor (z)
Decoding(z) to get the decoded tensor (θ)
Surrogate Model
Predicting the performance (ŷ) through the surrogate model
Calculating the gradient ŷ'(z) = δŷ/δz
z = z - learning_rate * gradient
required iterations completed?
yes
no
Stop
Buckling Load Factor
BLF
Utilization
UTL
Interstorey Drift Ratios
IDR
Mass
MAS
Height
HGT

FIGURE 44: Gradient Descent Workflow. Inspired by the work of Pavlidou (Own Work)

The workflow for the optimization involved selection of a input mesh. This step depended on the requirements of the user and the scenario. In the literature, Gladstone et al. had recommended finding a good starting point by randomly sampling 100 samples from the latent space and predicting the best performing one (Gladstone et al., 2021). In case the user would like to perform the optimization on a sample of this choice, he would have had an initial mesh to start with, so then that step can be replaced.

The sample was reshaped into (1,28) and normalized before compressed into a latent vector (z) of the VAE. After decoding z, the performance of the output tensor, ŷ, was then predicted. Then the performance tensor (ŷ) could be optimized with respect to the latent vector (z).

In this case, the optimizer performed gradient descent by using a Neural Network. ŷ is calculated according to the following rules depending on what performance metric was being considered.

If ŷ is Buckling Load Factor, then

$$\boxed{\hat{y} = -(1-b)^2}$$

$$\frac{\partial \hat{y}}{\partial z} = -2(1-b) \times \frac{db}{dz} \qquad \text{(Equation I)}$$

If ŷ is Utilization, then

$$\hat{y} = (u-1)^2$$

$$\frac{\partial \hat{y}}{\partial z} = 2(u-1) \times \frac{du}{dz} \qquad \text{(Equation II)}$$

If ŷ is Interstorey Drift Ratio, then

$$\hat{y} = (i-0.010)^2$$

$$\frac{\partial \hat{y}}{\partial z} = 2(i-0.010) \times \frac{di}{dz} \qquad \text{(Equation III)}$$

Failure conditions are taken into account in each performance metric and ŷ is calculated accordingly.

$Buckling\ Load\ Factor < 1$

$Utilization > 1$

$Interstorey\ Drift\ Ratio > 0.010h$ (as specified in Eurocode 8)

(Equation IV)

The gradient of the performance tensor (ŷ) with respect to z is updated according to the following rule:

$$z = z - learning\_rate \times \frac{\partial \hat{y}}{\partial z}$$

The learning rate determines the step size in each iteration. Once the iterations are completed, the mesh has been optimized.

For Utilization and Interstorey Drift Ratios, a global minima is sought to give the optimized result. However, for Buckling Load Factor, a global maxima is sought to give the optimized result. The minus sign in front of the gradient of Buckling Load Factor allows for this change and updates the update rule accordingly as shown below.

$$\boxed{\hat{y} = \ominus(1-b)^2}$$

$$z = z - learning\_rate \times \ominus \frac{\partial \hat{y}}{\partial z}$$

$$z = z \oplus learning\_rate \times \frac{\partial \hat{y}}{\partial z} \qquad \text{(Equation V)}$$

Furthermore, it becomes unnecessary to try to decrease (Utilization, Interstorey Drift Ratio) or increase (Buckling Load Factor) that metric if the metric is already under acceptable limits (no failure). To account for such conditions, the gradient update is configured accordingly by adding additional conditions.

If $(u-1) > 0$, perform the update; otherwise, do nothing.

If $(1-b) > 0$, perform the update; otherwise, do nothing.

If $(i-0.010) > 0$, perform the update; otherwise, do nothing.

(Equation VI)

In order to constrain the optimized meshes to be within a certain height threshold and to minimize material usage by minimizing mass, the gradient function was altered to account for multiple objectives instead of a single objective.

For single objective optimizations, the gradient function was as mentioned below:

$$y'(z) = \frac{\partial \hat{y}}{\partial z}$$

To consider multiple objectives, the different gradients were aggregated to form the overall gradient. Weights were included for each gradient to allow the user to optimize specific metrics over others.

$$y'(z) = W_1\left(\frac{\partial \hat{y}_1}{\partial z}\right) + W_2\left(\frac{\partial \hat{y}_2}{\partial z}\right) + W_3\left(\frac{\partial \hat{y}_3}{\partial z}\right)$$

where,
$y' =$ *aggregated gradient*
$\hat{y}_1 =$ *Height*
$\hat{y}_2 =$ *Mass*
$\hat{y}_3 =$ *Performance metric*
$z =$ *latent space*
$W_1, W_2, W_3 =$ *Weightage for respective gradients*

(Equation VII)

The performance metric ($\hat{y}_3$) may be Buckling Load Factor, Utilization, or Interstorey Drift Ratio. It should be noted that instead of selecting one performance metric, multiple performance metrics may be selected by additional terms in the gradient descent optimization - and the weights can be changed accordingly. For our case, as the likelihood of failure in Utilization was by far the highest, so Utilization was considered as the only performance metric ($\hat{y}_3$) in most simulations. Nonetheless the same workflow can be used to optimize other metrics simultaneously by applying different weights.

4.2 CONDUCTING PERFORMANCE EVALUATION

For conducting the Performance Analysis, due to the geometric complexity of doubly curved vaults, a numerical approach (in Karamba) was employed rather than an analytical one. This was done through Response Spectrum Method (RSA), a linear dynamic analysis. Since this is also integrated inside Grasshopper, it provided an easy transition from the Geometry Generation phase.

The scheme for the computation of the seismic response is visualized in Figure 45. This is based on the model used by Marseglia et al. (2020).

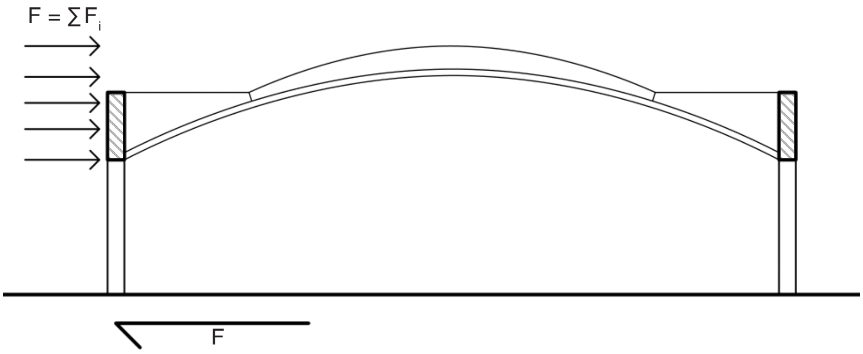


FIGURE 45: Seismic response scheme. (Own Work)

4.2.1 Performance Metrics

All of the output parameters from the Performance Evaluation are shown in Table 01. However, only 3 were chosen for the surrogate model. based on Strength, Stability, and Stiffness (SSS) (Heyman, 1966) indicating Ultimate Limit State (ULS) and Serviceability Limit State (SLS). These metrics from that were used are shown in Table 02.

SLS

This was conducted to check the serviceability limit state. The maximum interstorey drift of the vault was tabulated from this.

ULS

This was conducted to check the ultimate limit state. Utilization and Buckling Load Factor were assessed.

Other metrics were saved in case necessary for future use but were not used in the final surrogate model. These other metrics included maximum bending stress that would indicate failure in bending induced by lateral (seismic) loads when the applied stress exceeds the flexural strength. The maximum principal stress indicated if failure will occur in compression or tension when applied loads exceed the compressive strength or tensile strength. Average bending stress and average principal indicated stresses building up over a long period of time, which may not exceed the ULS but it may cause weakening over time.

	Performance metric
0	Buckling_Load_Factor
1	Utilization
2	Interstorey_Drift_Ratios
3	Avg_Displacement
4	Max_Displacement
5	Avg_Shear_Force
6	Max_Shear_Force
7	Avg_Bending_Moment
8	Max_Bending_Moment
9	Max_Compresive_Stress
10	Max_Tensile_Stress
11	Max_Principal_Stress

TABLE 01: All Output Parameters from Performance Evaluation. (Own Work)

Load case	Property
ULS	Utilization
	Buckling Load Factor
SLS	Interstorey Drift Ratio

TABLE 02: Output Parameters for both SLS and ULS. (Own Work)

4.2.2 Methods And Softwares Used For The Analysis

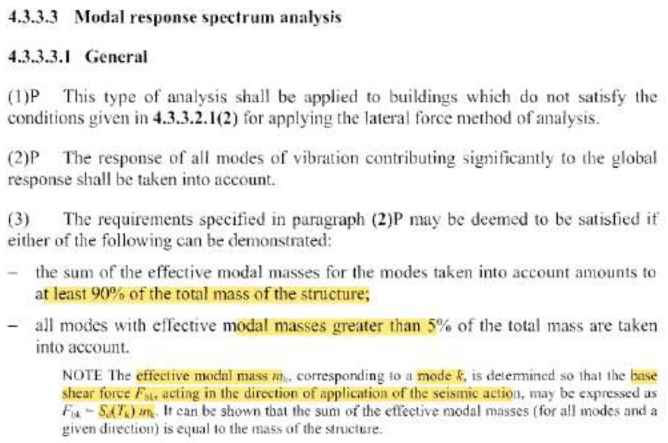
A balance of speed and accuracy was required as 10,000 simulations needed to be run efficiently. As the geometry was produced in Grasshopper, it was preferable to use a plugin for Grasshopper in keeping with the workflow. Nonetheless simulations in ABAQUS were also run, but mostly for validation purposes.

Karamba and Alpaca4d were explored. Alpaca4d is a plugin for Grasshopper developed on top of OpenSees. While it was specifically made for running dynamic analysis, it is still under development, therefore containing a considerable number of software defects. Therefore, its results needed to be first validated before moving on with it.

A simple case of a cantilevered beam was considered and simulations were run in Karamba and Alpaca and verified with hand calculations. Alpaca gave correct results for mass participation and Time periods (as verified by similar results in Karamba and ABAQUS). This was also true for internal reactions for line elements (beams), but it did not give accurate internal reactions (stresses, deflections) for shell elements. As the vaults needed to be modeled as shells, this would not work for the project. Moreover, Alpaca was found to be incompatible with the computer that was going to be used to run the 10,000 Performance Evaluation simulations in TU Delft Bouwkunde's VR Lab. For this reason, Alpaca proved to be useful but only as a validation tool for modal analysis but not for the seismic analysis afterwards.

For the seismic analysis, it was initially considered running non-linear analysis but it was found inappropriate for the project due to the number of simulations that needed to be run as it was far too time-consuming. Moreover, a non-linear analysis would only be possible in ABAQUS where the workflow would have had to be revisited as manual pre-processing was required to convert the imported rhino geometry into a mesh for ABAQUS. These steps would have had to be automated and that was not possible within the limited time-frame of the project.

A Linear Time History Analysis was considered in Alpaca but it took longer than was computationally feasible. In addition, instead of exciting the ground, point loads had to be applied on the building which get accelerated with the earthquake which may not have yielded accurate results and as mentioned earlier, results for shells are also inaccurate in Alpaca. For this reason, a linear dynamic analysis was chosen – the Response Spectrum Analysis. This would take into account the dominant modes - each with a minimum mass participation ratio of 5% combining to give a total mass participation ratio equal to or greater than 90% of the total mass, as prescribed by the Eurocode (image of Eurocode mass participation). This would produce more realistic results than the linear static approach of Equivalent Lateral Force method and still be fast enough for running a large number of simulations – with each simulation taking 3.0-4.6 seconds.



4.3.3.3 Modal response spectrum analysis

4.3.3.3.1 General

(1)P This type of analysis shall be applied to buildings which do not satisfy the conditions given in **4.3.3.2.1(2)** for applying the lateral force method of analysis.

(2)P The response of all modes of vibration contributing significantly to the global response shall be taken into account.

(3) The requirements specified in paragraph **(2)P** may be deemed to be satisfied if either of the following can be demonstrated:

- the sum of the effective modal masses for the modes taken into account amounts to at least 90% of the total mass of the structure;
- all modes with effective modal masses greater than 5% of the total mass are taken into account.

NOTE The effective modal mass m_k, corresponding to a mode k, is determined so that the base shear force F_{bk}, acting in the direction of application of the seismic action, may be expressed as $F_{bk} = S_d(T_k)\,m_k$. It can be shown that the sum of the effective modal masses (for all modes and a given direction) is equal to the mass of the structure.

FIGURE 46: Eurocode 8 formulation on Modal Response Spectrum Analysis. Image taken from International Organization for Standardization. (2004). EN 1998-1:2004 Eurocode 8: Design of structures for earthquake resistance - Part 1: General rules, seismic actions and rules for buildings. https://www.phd.eng.br/wp-content/uploads/2015/02/en.1998.1.2004.pdf

To carry out the analysis on a continuous model as is the case in the Linear Dynamic analysis, the brick and mortar would behave in combination like a single masonry unit. The properties are shown in Table 03.

Properties	Macromodel		
		Units	Unit (mortar + masonry)
Young's Modulus	*E*	GPa	3.2
Possion ratio	*ν*	-	0.15
Density	*ρ*	kg/m3	1219.4
Tension	*Gfl*	N/mm	0.14
	ft	MPa	0.24
Compression	*Gfc*	N/mm	9.44
	fc	MPa	5.9
Flexure	*fflex*	MPa	-
Cohesion	*c*	MPa	-
Friction angle	*φ*	deg	-

TABLE 03: Material property for Macromodel. Material properties for the macromodel are retrieved from López, L., Rodríguez, D., & Fernández, P. (n.d.). Using a Construction Technique to Understand it: Thin-Tile Vaulting.

4.2.3 Site

The location was chosen based on sesimicity and prevalence of earth construction. Pakistan is divided into 5 seismic zones (Zone 1, Zone 2A, Zone 2B, Zone 3, and Zone 4) in increasing order of seismicity level. A medium seismicity location was chosen based on the literature in the previous sections where it was concluded in resources that thin tile vaults reinforced with basalt geo-grids may be used in low to medium seismic regions. According to Figure 48, the Khuzdar District, in the province of Balochistan, in Pakistan, houses predominantly adobe construction. It also lies in a seismic zone where low to high seismicity is found. A location in Zone 2B, was selected corresponding to a Peak Ground Acceleration of 0.2g - a medium seismicity level. The dataset was also generated based on this Peak Ground Acceleration (PGA). The location can be seen marked in Figure 47.

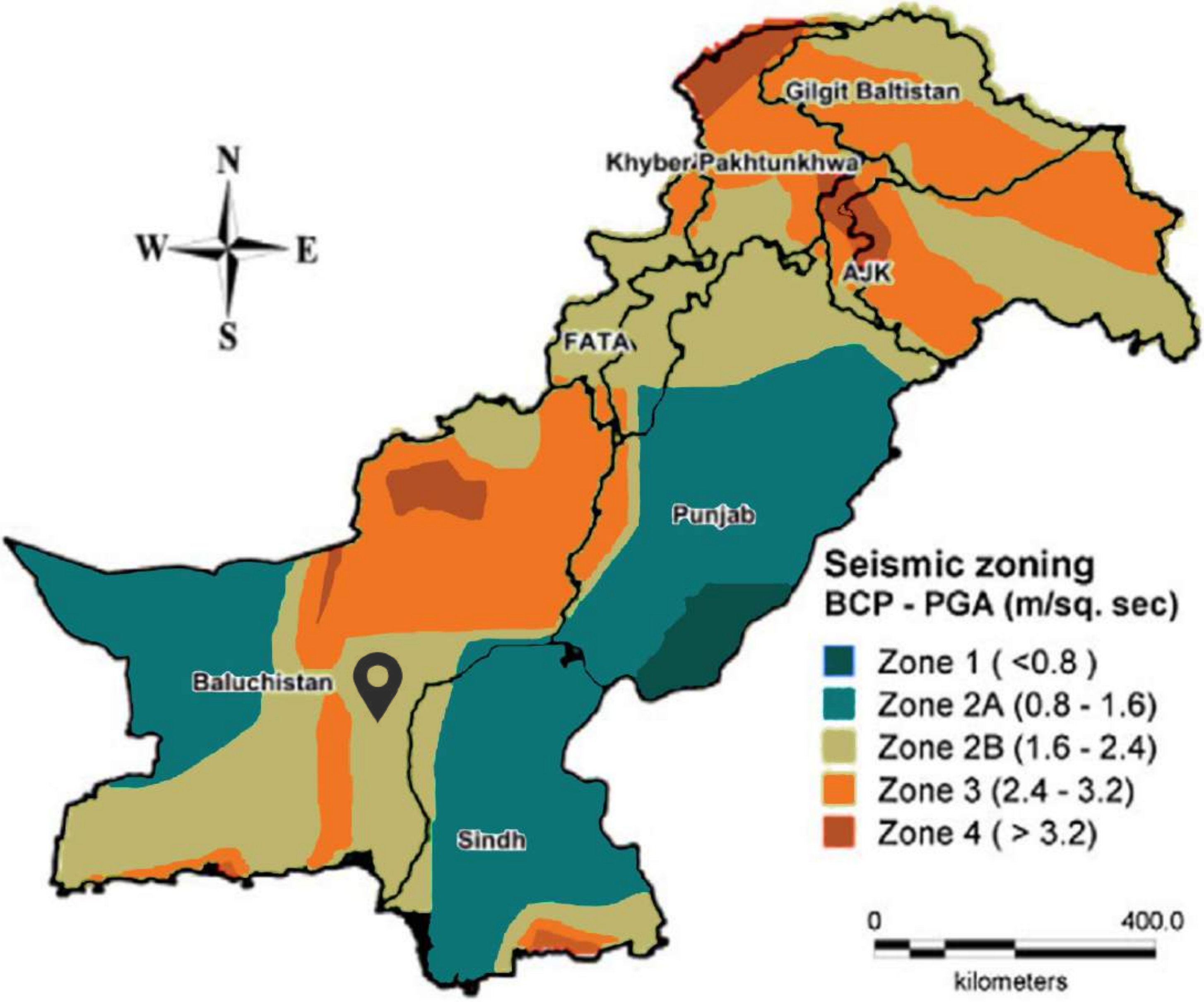
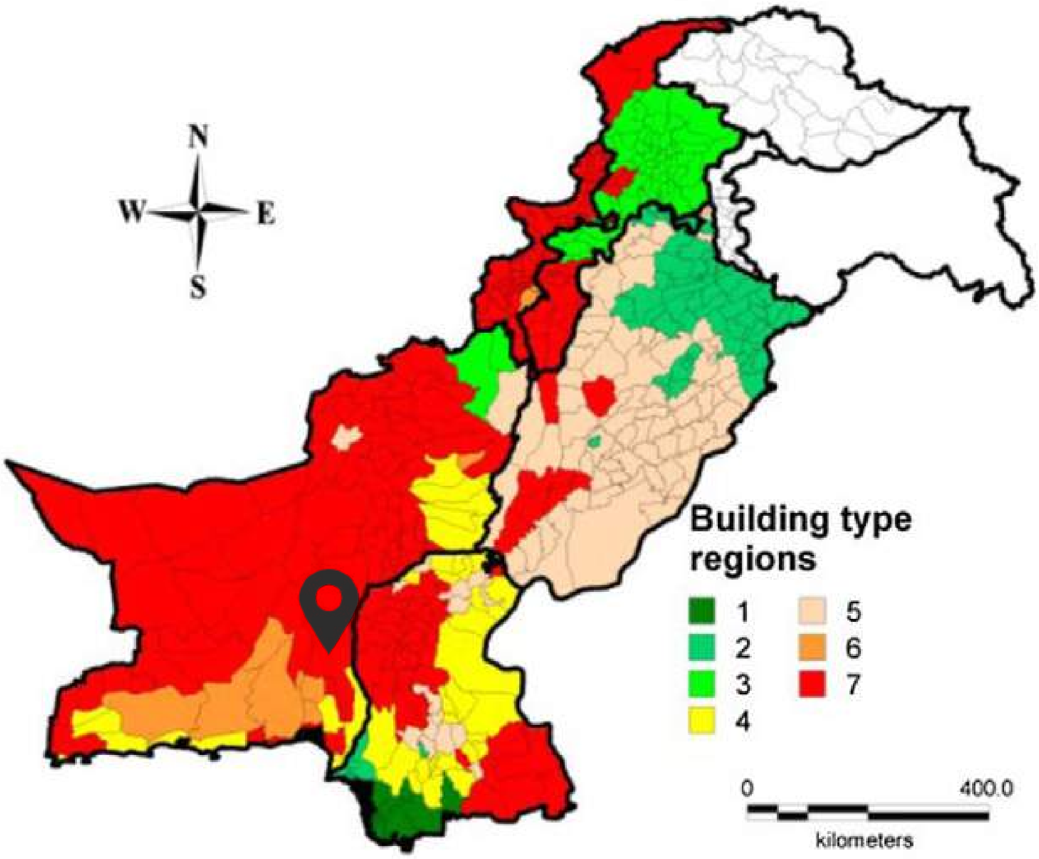
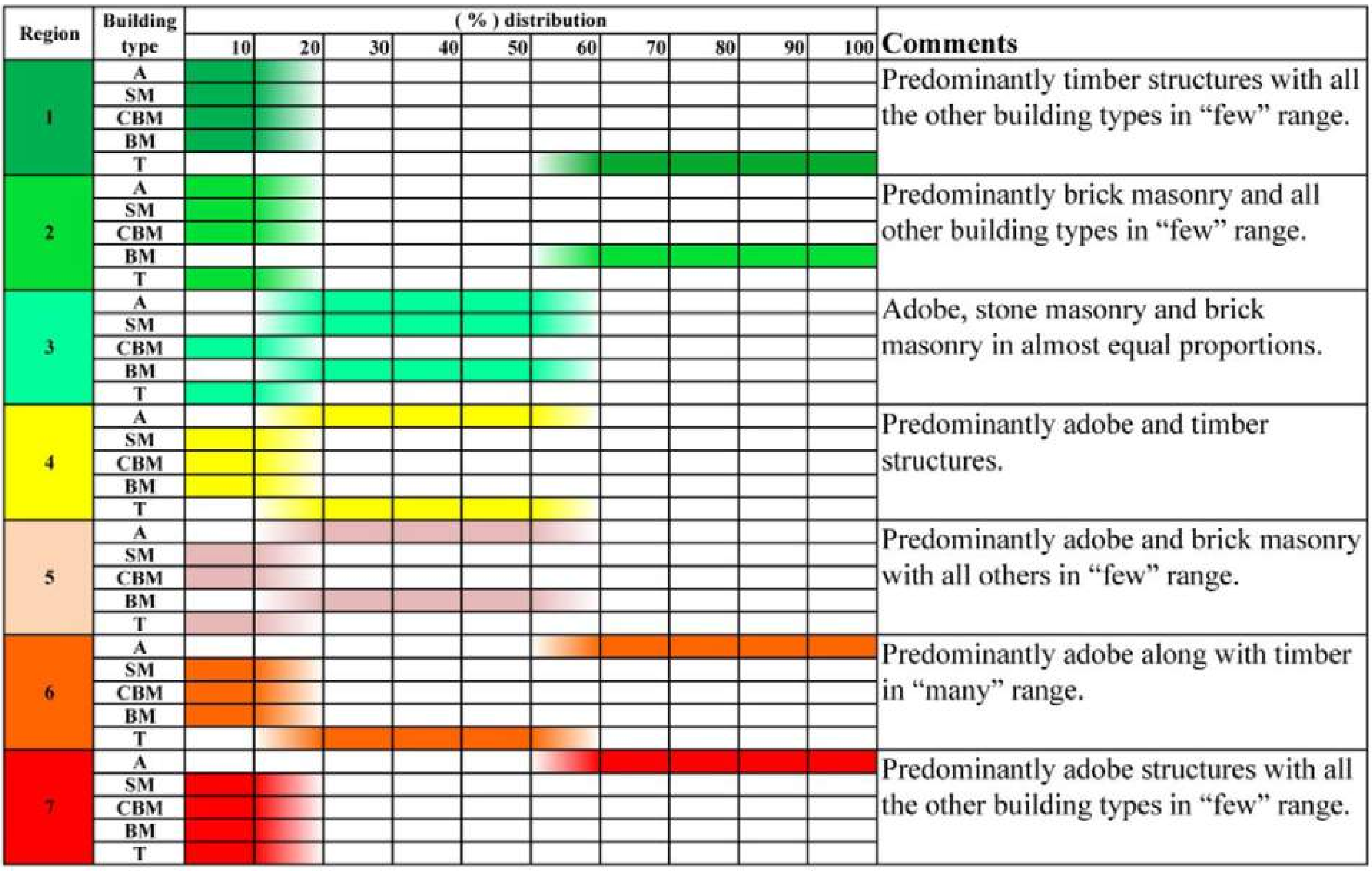


FIGURE 47: Sesimic Zoning map of Pakistan according to Building Code of Pakistan (BCP). Edited by Author. Image Taken from Siddique, M. S., & Schwarz, J. (2015). Elaboration of Multi-Hazard Zoning and Qualitative Risk Maps of Pakistan. Earthquake Spectra, 31(3), 1371–1395. https://doi.org/10.1193/042913EQS114M





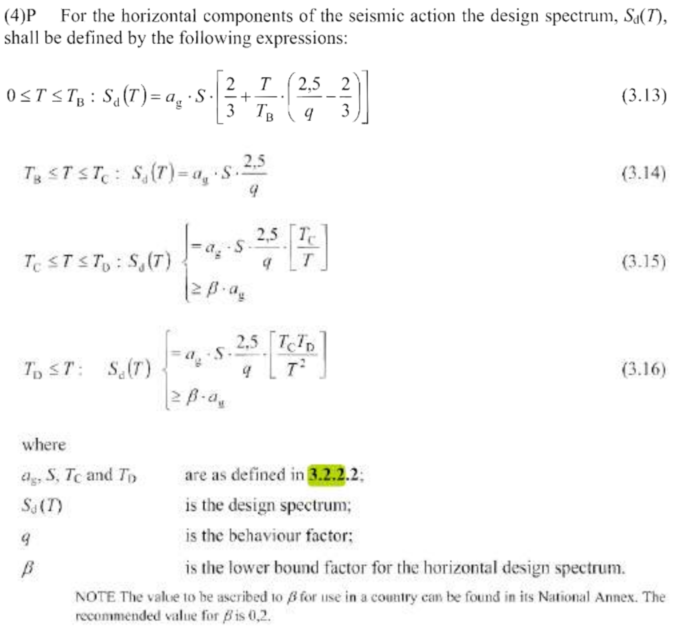
Region	Building type	(%) distribution										Comments
		10	20	30	40	50	60	70	80	90	100	
1	A	■	▒									Predominantly timber structures with all the other building types in "few" range.
	SM	■	▒									
	CBM	■	▒									
	BM	■	▒									
	T						▒	■	■	■	■	
2	A	■	▒									Predominantly brick masonry and all other building types in "few" range.
	SM	■	▒									
	CBM	■	▒									
	BM						▒	■	■	■	■	
	T	■	▒									
3	A		▒	■	■	■	▒					Adobe, stone masonry and brick masonry in almost equal proportions.
	SM		▒	■	■	■	▒					
	CBM	■	▒									
	BM		▒	■	■	■	▒					
	T	■	▒									
4	A		▒	■	■	■	▒					Predominantly adobe and timber structures.
	SM	■	▒									
	CBM	■	▒									
	BM	■	▒									
	T		▒	■	■	■	▒					
5	A		▒	■	■	■	▒					Predominantly adobe and brick masonry with all others in "few" range.
	SM	■	▒									
	CBM	■	▒									
	BM		▒	■	■	■	▒					
	T	■	▒									
6	A						▒	■	■	■	■	Predominantly adobe along with timber in "many" range.
	SM	■	▒									
	CBM	■	▒									
	BM	■	▒									
	T		▒	■	■	■	▒					
7	A						▒	■	■	■	■	Predominantly adobe structures with all the other building types in "few" range.
	SM	■	▒									
	CBM	■	▒									
	BM	■	▒									
	T	■	▒									

FIGURE 48: Building type region map of Pakistan. Image taken from Siddique, M. S., & Schwarz, J. (2015). Elaboration of Multi-Hazard Zoning and Qualitative Risk Maps of Pakistan. Earthquake Spectra, 31(3), 1371–1395. https://doi.org/10.1193/042913EQS114M

4.2.4 Response Spectrum Analysis

After the Time Periods were taken out for the dominant modes, the following was done for all modes

Design Spectrum.



(4)P For the horizontal components of the seismic action the design spectrum, $S_d(T)$, shall be defined by the following expressions:

$$0 \le T \le T_B : S_d(T) = a_g \cdot S \cdot \left[\frac{2}{3} + \frac{T}{T_B} \cdot \left(\frac{2,5}{q} - \frac{2}{3}\right)\right] \quad (3.13)$$

$$T_B \le T \le T_C : S_d(T) = a_g \cdot S \cdot \frac{2,5}{q} \quad (3.14)$$

$$T_C \le T \le T_D : S_d(T) \begin{cases} = a_g \cdot S \cdot \frac{2,5}{q} \cdot \left[\frac{T_C}{T}\right] \\ \ge \beta \cdot a_g \end{cases} \quad (3.15)$$

$$T_D \le T : \quad S_d(T) \begin{cases} = a_g \cdot S \cdot \frac{2,5}{q} \cdot \left[\frac{T_C T_D}{T^2}\right] \\ \ge \beta \cdot a_g \end{cases} \quad (3.16)$$

where

a_g, S, T_C and T_D	are as defined in 3.2.2.2;
$S_d(T)$	is the design spectrum;
q	is the behaviour factor;
β	is the lower bound factor for the horizontal design spectrum.

NOTE The value to be ascribed to β for use in a country can be found in its National Annex. The recommended value for β is 0,2.

FIGURE 49: Eurocode 8 formulation on Design Spectrum. Image taken from International Organization for Standardization. (2004). EN 1998-1:2004 Eurocode 8: Design of structures for earthquake resistance - Part 1: General rules, seismic actions and rules for buildings. https://www.phd.eng.br/wp-content/uploads/2015/02/en.1998.1.2004.pdf

Constants to calculate spectral accelerations from the Design Spectrum	
Characteristics	
Ground Type (macro)	Type 1
Ground Type (micro)	B
importance factor (γ_1)	1
Behaviour factor (q)	1.5
Number of storeys	1
Peak Ground Accelration (PGA)	2.0 g

TABLE 04: Constants to calculate spectral accelerations from the Design Spectrum. (Own Work)

Residences belong to an Importance factor, (γ_1), of 1.0 corresponding to Importance Class II. The behaviour factor, q, was taken as 1.5, assumed for unreinforced masonry according to the Eurocode formulation given below.

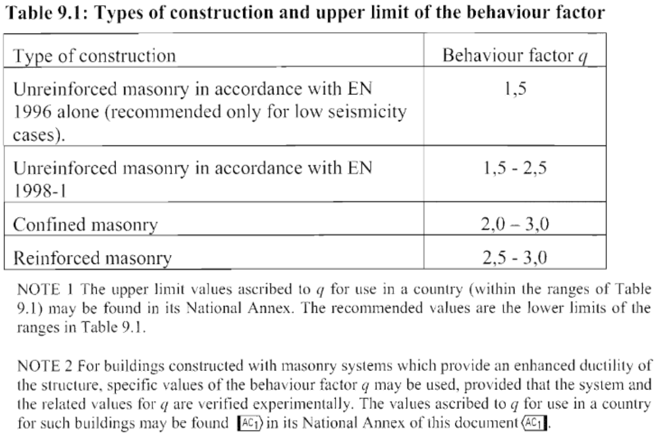


Table 9.1: Types of construction and upper limit of the behaviour factor

Type of construction	Behaviour factor q
Unreinforced masonry in accordance with EN 1996 alone (recommended only for low seismicity cases).	1,5
Unreinforced masonry in accordance with EN 1998-1	1,5 - 2,5
Confined masonry	2,0 – 3,0
Reinforced masonry	2,5 - 3,0

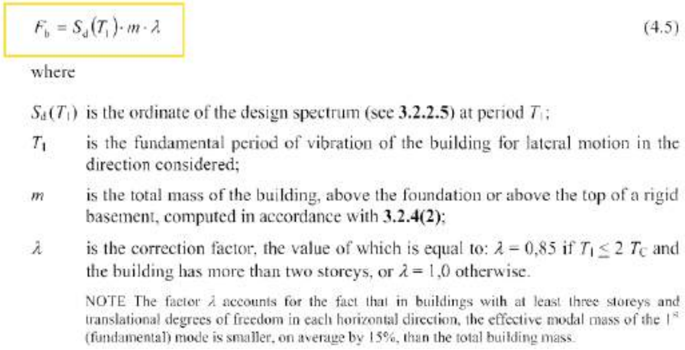
NOTE 1 The upper limit values ascribed to q for use in a country (within the ranges of Table 9.1) may be found in its National Annex. The recommended values are the lower limits of the ranges in Table 9.1.

NOTE 2 For buildings constructed with masonry systems which provide an enhanced ductility of the structure, specific values of the behaviour factor q may be used, provided that the system and the related values for q are verified experimentally. The values ascribed to q for use in a country for such buildings may be found AC1 in its National Annex of this document AC1.

FIGURE 50: Behaviour factor for unreinforced masonry. Table taken from En 1990: Eurocode - basis of structural design. (n.d.). https://www.phd.eng.br/wp-content/uploads/2015/12/en.1990.2002.pdf

Base Shear

Base shear was calculated according to the expression.



$$F_b = S_d(T_1) \cdot m \cdot \lambda \qquad (4.5)$$

where

$S_d(T_1)$ is the ordinate of the design spectrum (see **3.2.2.5**) at period T_1;

T_1 is the fundamental period of vibration of the building for lateral motion in the direction considered;

m is the total mass of the building, above the foundation or above the top of a rigid basement, computed in accordance with **3.2.4(2)**;

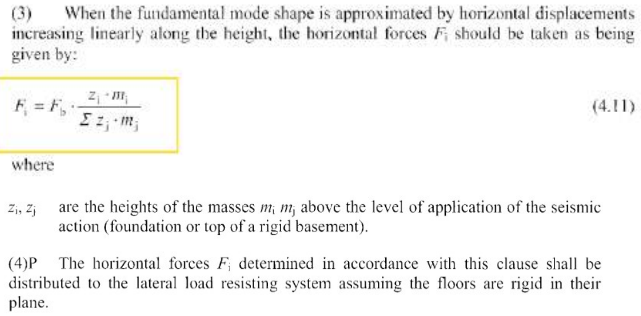
λ is the correction factor, the value of which is equal to: $\lambda = 0{,}85$ if $T_1 \leq 2\ T_C$ and the building has more than two storeys, or $\lambda = 1{,}0$ otherwise.

NOTE The factor λ accounts for the fact that in buildings with at least three storeys and translational degrees of freedom in each horizontal direction, the effective modal mass of the 1st (fundamental) mode is smaller, on average by 15%, than the total building mass.

FIGURE 51: Eurocode 8 formulation of Base Shear. Image taken from International Organization for Standardization. (2004). EN 1998-1:2004 Eurocode 8: Design of structures for earthquake resistance - Part 1: General rules, seismic actions and rules for buildings. https://www.phd.eng.br/wp-content/uploads/2015/02/en.1998.1.2004.pdf

Vertical Distribution of Forces

As the main dataset consisted of a single storey structure, the base shear was distributed into the first storey base of the vault and the rest of the vault. It was done using the following expression:



(3) When the fundamental mode shape is approximated by horizontal displacements increasing linearly along the height, the horizontal forces F_i should be taken as being given by:

$$F_i = F_b \cdot \frac{z_i \cdot m_i}{\Sigma\, z_j \cdot m_j} \qquad (4.11)$$

where

z_i, z_j are the heights of the masses m_i m_j above the level of application of the seismic action (foundation or top of a rigid basement).

(4)P The horizontal forces F_i determined in accordance with this clause shall be distributed to the lateral load resisting system assuming the floors are rigid in their plane.

FIGURE 52: Eurocode 8 formulation for the vertical distribution of force. Image taken from International Organization for Standardization. (2004). EN 1998-1:2004 Eurocode 8: Design of structures for earthquake resistance - Part 1: General rules, seismic actions and rules for buildings. https://www.phd.eng.br/wp-content/uploads/2015/02/en.1998.1.2004.pdf

Seismic Combinations

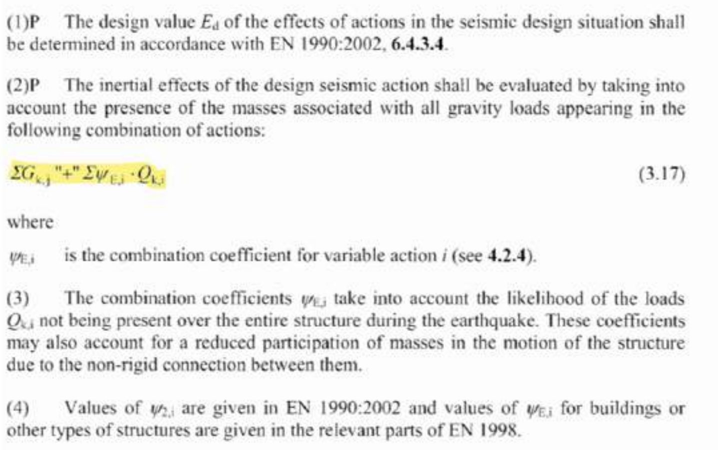
The forces were combined for the different modes using the Square Root of the Sum of Squares (SRSS) method.

$$\text{SRSS} = \sqrt{\sum_{i=1}^{n} F_i^2}$$

where F_i is the force for a specific mode and n is the number of different modes.

Other constants

The seismic weight of the structure was calculated using factors as relayed in the Eurocode.



(1)P The design value E_d of the effects of actions in the seismic design situation shall be determined in accordance with EN 1990:2002, **6.4.3.4**.

(2)P The inertial effects of the design seismic action shall be evaluated by taking into account the presence of the masses associated with all gravity loads appearing in the following combination of actions:

$$\Sigma G_{k,j} \text{ "+" } \Sigma \psi_{E,i} \cdot Q_{k,i} \qquad (3.17)$$

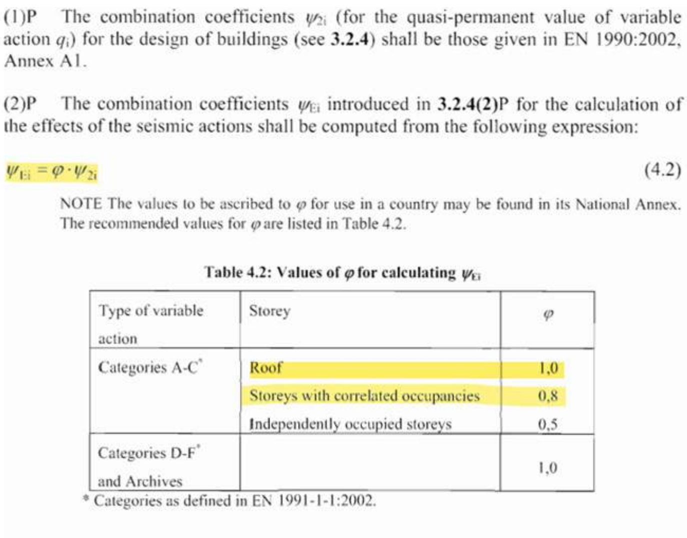
where

$\psi_{E,i}$ is the combination coefficient for variable action i (see **4.2.4**).

(3) The combination coefficients $\psi_{E,i}$ take into account the likelihood of the loads $Q_{k,i}$ not being present over the entire structure during the earthquake. These coefficients may also account for a reduced participation of masses in the motion of the structure due to the non-rigid connection between them.

(4) Values of $\psi_{2,i}$ are given in EN 1990:2002 and values of $\psi_{E,i}$ for buildings or other types of structures are given in the relevant parts of EN 1998.

The calculation of combination coefficent for variable, ψEi, action is further given by



(1)P The combination coefficients ψ_{2i} (for the quasi-permanent value of variable action q_i) for the design of buildings (see **3.2.4**) shall be those given in EN 1990:2002, Annex A1.

(2)P The combination coefficients ψ_{Ei} introduced in **3.2.4(2)**P for the calculation of the effects of the seismic actions shall be computed from the following expression:

$$\psi_{Ei} = \varphi \cdot \psi_{2i} \qquad (4.2)$$

NOTE The values to be ascribed to φ for use in a country may be found in its National Annex. The recommended values for φ are listed in Table 4.2.

Table 4.2: Values of φ for calculating ψ_{Ei}

Type of variable action	Storey	φ
Categories A-C*	Roof	1,0
	Storeys with correlated occupancies	0,8
	Independently occupied storeys	0,5
Categories D-F* and Archives		1,0

* Categories as defined in EN 1991-1-1:2002.

FIGURE 53: Combination coefficients. Image taken from En 1990: Eurocode - basis of structural design. (n.d.). https://www.phd.eng.br/wp-content/uploads/2015/12/en.1990.2002.pdf

As mentioned in Figure X, the value of ɸ = 1.0 is considered as it is a single storey structure.

The value of ψ2 was considered to be 0.3 according to the values for residential areas as shown in Figure 54.

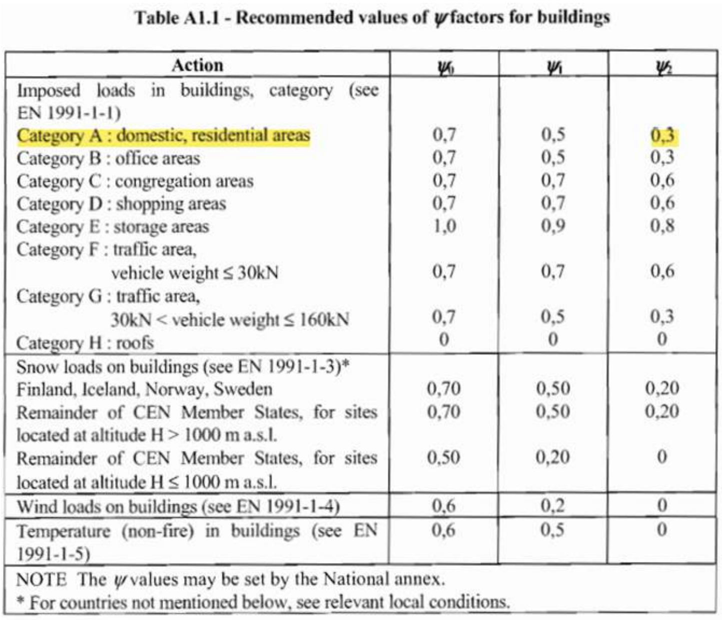


Table A1.1 - Recommended values of ψ factors for buildings

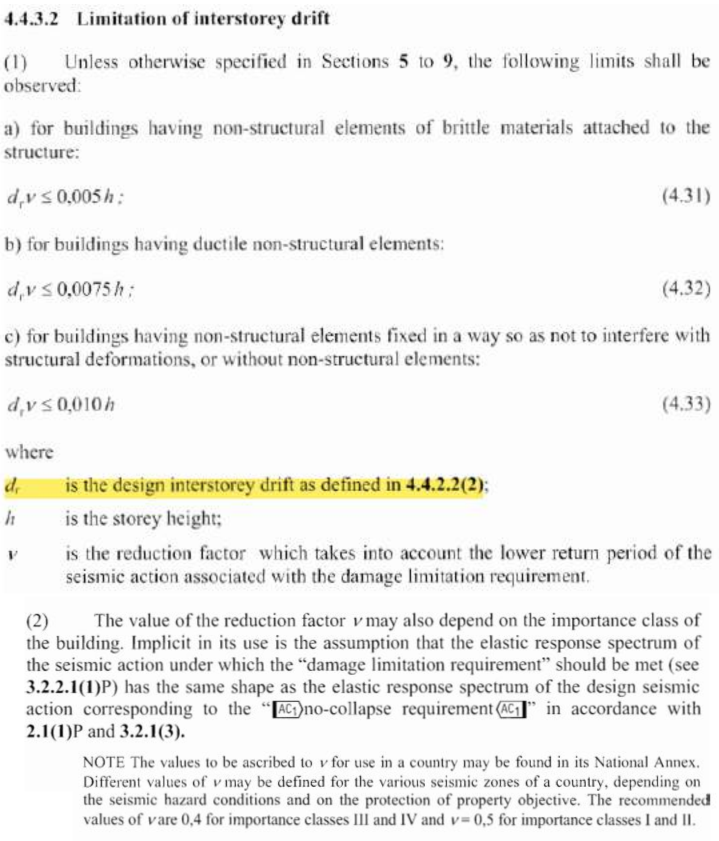
Action	ψ_0	ψ_1	ψ_2
Imposed loads in buildings, category (see EN 1991-1-1)			
Category A : domestic, residential areas	0,7	0,5	0,3
Category B : office areas	0,7	0,5	0,3
Category C : congregation areas	0,7	0,7	0,6
Category D : shopping areas	0,7	0,7	0,6
Category E : storage areas	1,0	0,9	0,8
Category F : traffic area, vehicle weight ≤ 30kN	0,7	0,7	0,6
Category G : traffic area, 30kN < vehicle weight ≤ 160kN	0,7	0,5	0,3
Category H : roofs	0	0	0
Snow loads on buildings (see EN 1991-1-3)*			
Finland, Iceland, Norway, Sweden	0,70	0,50	0,20
Remainder of CEN Member States, for sites located at altitude H > 1000 m a.s.l.	0,70	0,50	0,20
Remainder of CEN Member States, for sites located at altitude H ≤ 1000 m a.s.l.	0,50	0,20	0
Wind loads on buildings (see EN 1991-1-4)	0,6	0,2	0
Temperature (non-fire) in buildings (see EN 1991-1-5)	0,6	0,5	0

NOTE The ψ values may be set by the National annex.
* For countries not mentioned below, see relevant local conditions.

FIGURE 54: Recommended values for ψ2 factor. Image taken from En 1990: Eurocode - basis of structural design. (n.d.). https://www.phd.eng.br/wp-content/uploads/2015/12/en.1990.2002.pdf

Interstorey Drift

Whilst the other two performance metrics used (Buckling Load Factor and Utilization) were calculated directly by the FEA model in Karamba, calculations for interstorey drift depended on Eurocode formulations. It should be less than 0.010h, where h is the storey height, which was 3 meters in for all samples.



4.4.3.2 Limitation of interstorey drift

(1) Unless otherwise specified in Sections **5** to **9**, the following limits shall be observed:

a) for buildings having non-structural elements of brittle materials attached to the structure:

$$d_r \nu \le 0{,}005\,h\,; \qquad (4.31)$$

b) for buildings having ductile non-structural elements:

$$d_r \nu \le 0{,}0075\,h\,; \qquad (4.32)$$

c) for buildings having non-structural elements fixed in a way so as not to interfere with structural deformations, or without non-structural elements:

$$d_r \nu \le 0{,}010\,h \qquad (4.33)$$

where

d_r is the design interstorey drift as defined in **4.4.2.2(2)**;

h is the storey height;

ν is the reduction factor which takes into account the lower return period of the seismic action associated with the damage limitation requirement.

(2) The value of the reduction factor ν may also depend on the importance class of the building. Implicit in its use is the assumption that the elastic response spectrum of the seismic action under which the "damage limitation requirement" should be met (see **3.2.2.1(1)**P) has the same shape as the elastic response spectrum of the design seismic action corresponding to the "[AC1)no-collapse requirement(AC1]" in accordance with **2.1(1)**P and **3.2.1(3)**.

NOTE The values to be ascribed to ν for use in a country may be found in its National Annex. Different values of ν may be defined for the various seismic zones of a country, depending on the seismic hazard conditions and on the protection of property objective. The recommended values of ν are 0,4 for importance classes III and IV and ν = 0,5 for importance classes I and II.

FIGURE 55: Interstorey Drift. Image taken from International Organization for Standardization. (2004). EN 1998-1:2004 Eurocode 8: Design of structures for earthquake resistance - Part 1: General rules, seismic actions and rules for buildings. https://www.phd.eng.br/wp-content/uploads/2015/02/en.1998.1.2004.pdf

Since the importance class was II, the value for the reduction factor, v , was taken as 0.5. The design interstorey drift, d_r, was evaluated as the difference of the average lateral displacements, d_s , at the top and bottom of the storey and calculated using the following expression as specified in Section 4.3.4 of the Eurocode:

$d_s = q_d . d_c$

where,

d_s is the displacement of a point of the structural system induced by the design seismic action

q_d is the displacement behaviour factor assumed to be q unless otherwise specified

d_c is the displacement of the same point of the structural system, as determined by a linear analysis based on the design response spectrum.

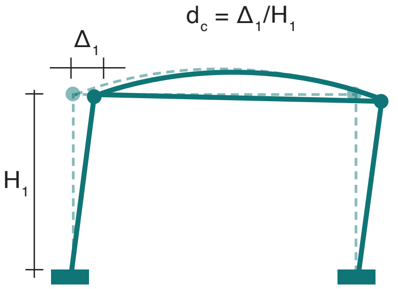


FIGURE 56: Interstorey Drift visualization. (Own Work)

The results of the Performance Evaluation are shown later in Section 06.

05/ RESULTS : GEOMETRY GENERATION

5.1 FORCE DENSITIES

See Section 4.1.1 for the workflow.

As mentioned earlier in the paper, the method that was used to generate geometries of the vault was by varying the force densities. The flat 2D projection of the vault was composed of a flat mesh. Each edge of the mesh was assigned a force density denoting how much force per unit length is carried. Negative force densities correspond to tensile structures while positive force densities correspond to compression structures. This is shown in Figure 57. The scope of this paper is limited to compression structures so only positive force densities will be discussed.

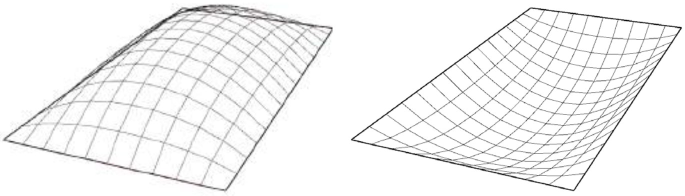


FIGURE 57: Positive Force Densities (left) and Negative Force Densities. (Own Work)

Since variation in force densities served as the input variable that was used to generate a variations in geometry to produce the large dataset for the generator, it is important to understand the effect of varying force densities on the geometry. Figure 58 shows that for a mesh with uniform force densities for all edges, the effect on the height of the vault of increasing force densities diminishes rapidly. The simulation was carried out with a constant load (10.0). It should be noted that varying the load and force densities serve the same purpose as they produce inverse geometric effects. Throughout the paper, for simulations over a dataset, the load is kept constant so that the effect of varying force densities can be seen.

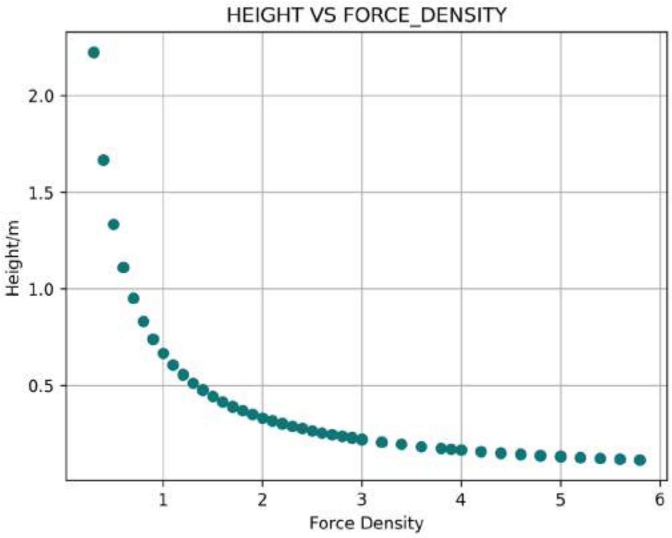
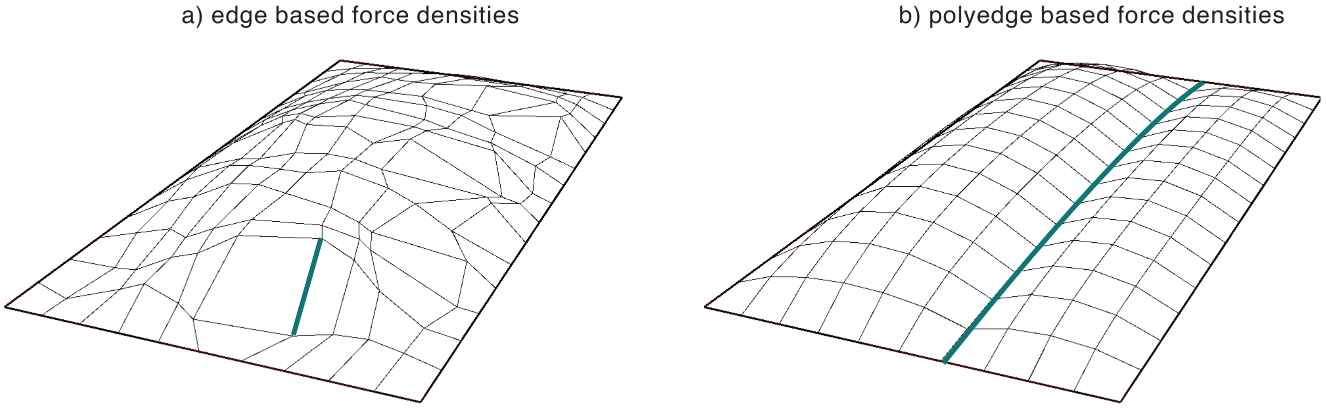


FIGURE 58: Relationship of Height of vault with force density. (Own Work)

5.2 POLYEDGES VS SINGLE EDGES

The greater the force density of the vault, the shallower the part of the vault becomes. Since we are dealing with tile vaults, the degree in variability of force densities is restricted to polyedges rather than single edges. Figure 59 illustrates the difference. Such forms as Figure 59a would not be possible to construct for practical purposes out of thin tile vaults. The term polyedges here represents the series of continuous edges connected. This is further highlighted in Figure 59b.


a) edge based force densities

b) polyedge based force densities

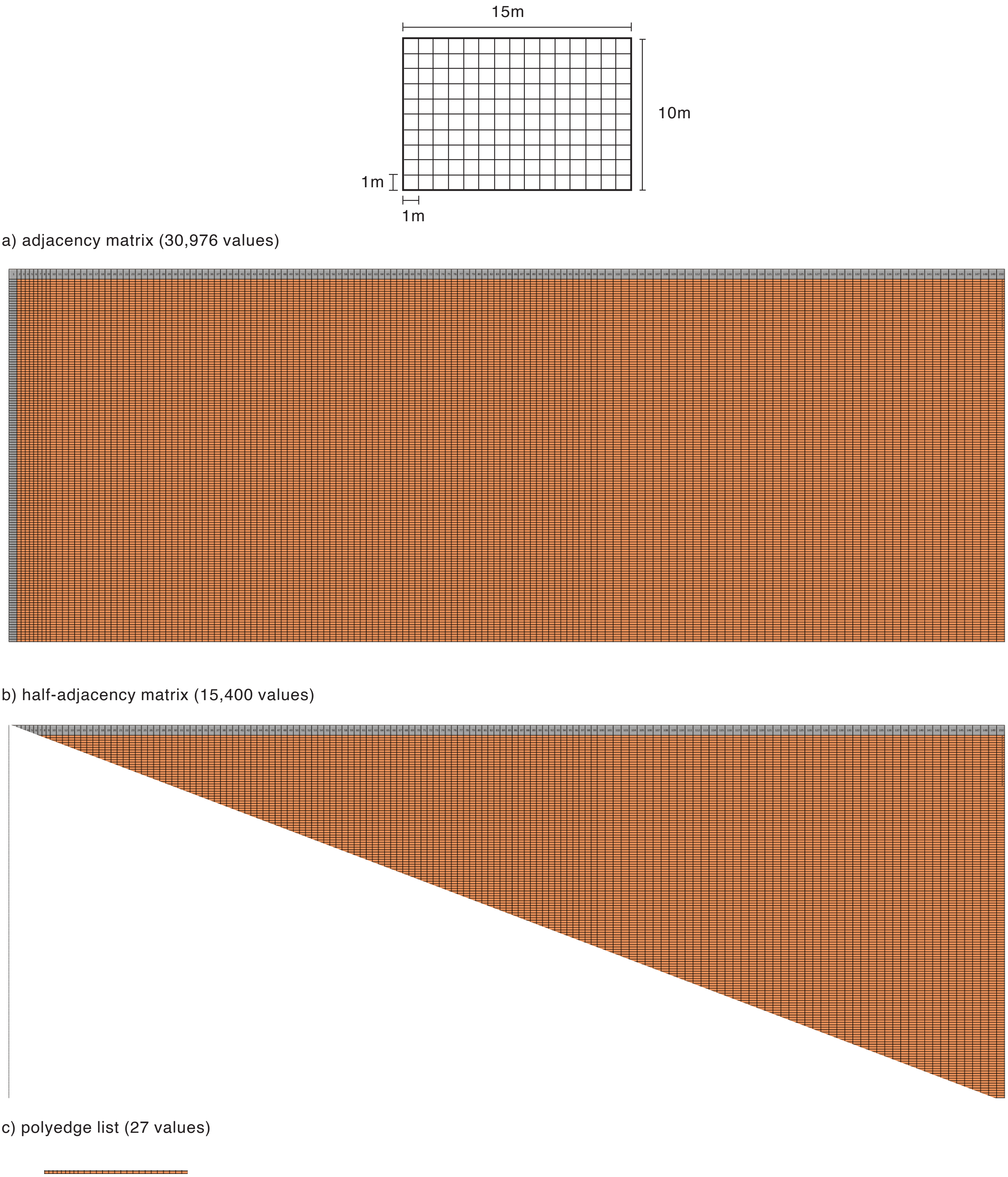
FIGURE 59: Force density change in single edges (left) and polyedges (right). (Own Work)

By assigning force densities to polyedges, we also reduce the size of the representative dataset considerably. For instance, a mesh of 15x10 divisions would be represented by an adjacency matrix of shape 176x176 which flattens to form a list of 30,976 values, where each value represents the force density of each single edge connection. Since adjacency matrices are symmetrical, if we only consider half of the adjacency matrix, we get 176x175/2 values which flattens to form 15,400 values.

In contrast to this, if we only consider polyedges, there is no need to look at connectivity or edge-based relationship representations such as adjacency matrices or any other type of graphs. This is because, for a quad mesh, all the edge-based relationships are the same, only the value of the force density for each polyedge matters. Therefore, each mesh can be represented as a list with a size equal to the number of polyedges present. For the same mesh density of 15x10 divisions, a nested list would have a total of 27 flattened values – 16 values in one direction, while 11 values in the perpendicular one.

To conclude, by keeping the following data-structure, the size of each sample has been reduced by 99.91% (relative to adjacency matrix) and 99.82% (relative to half-adjacency matrix). This may be visualized in Figure 59.

The size of the mesh is kept constant throughout all samples of the dataset. The dimensions are 15mx10m. The size was chosen to represent the case of footprint of a 150m² house. The mesh density was kept constant at 1 division per metre.



a) adjacency matrix (30,976 values)

b) half-adjacency matrix (15,400 values)

c) polyedge list (27 values)

FIGURE 60: Reduction of the dataset size. (Own Work)

An example is shown below of a sample taken from the ‘crease’ dataset. It highlights how multiple edges are grouped into a single polyedge that may be either a row or a column. This is represented in the form of a nested list where,

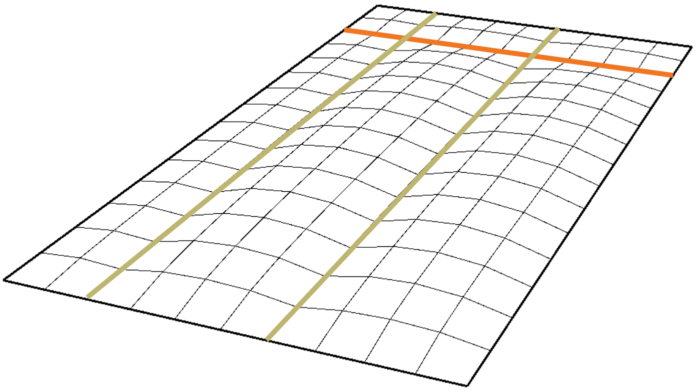
number of polyedges = $n_{x,y}+1$

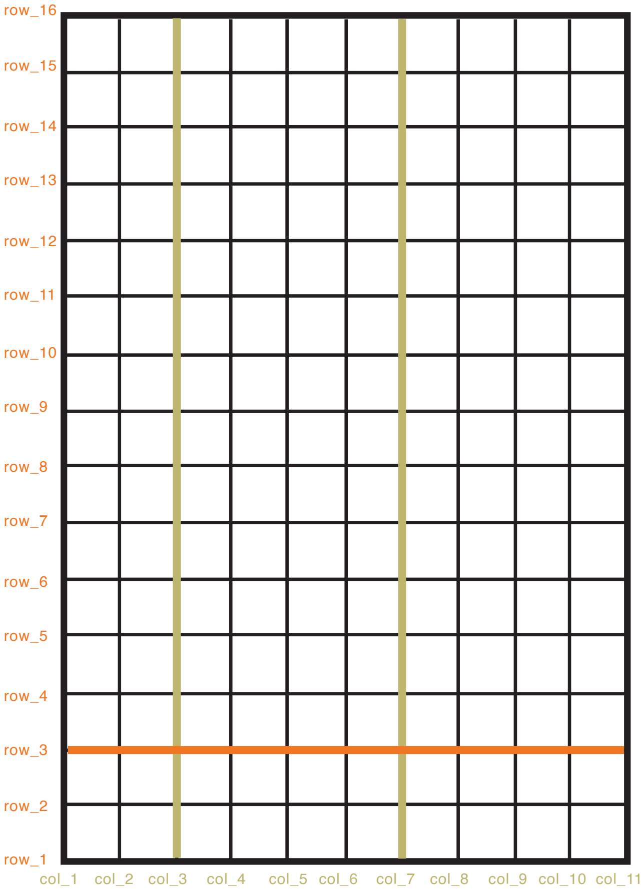
For the dataset, x=15, y=10, so $n_x = 11$, $n_y = 16$. Each value in the nested list corresponds to the force density, q, of the polyedge.

polyedge = column

polyedge = row

$$[\ [q_1,\ q_2,\ q_3,q_4,q_5,q_6,q_7,q_8,\ q_9,\ q_{10},\ q_{11}]\ ,\ [q_{12},q_{13},q_{14},\ q_{15},\ q_{16},q_{17},q_{18},q_{19},q_{20},\ q_{21},\ q_{22},\ q_{23},\ q_{24},\ q_{25},\ q_{26},\ q_{27}\ ]\ ]$$





COLUMN Individual Edge number	1	2	3	4	5	6	7	8	9	10	11
	1	31	62	93	124	155	186	217	248	279	310
	3	33	64	95	126	157	188	219	250	281	311
	5	35	66	97	128	159	190	221	252	283	312
	7	37	68	99	130	161	192	223	254	285	313
	9	39	70	101	132	163	194	225	256	287	314
	11	41	72	103	134	165	196	227	258	289	315
	13	43	74	105	136	167	198	229	260	291	316
	15	45	76	107	138	169	200	231	262	293	317
	17	47	78	109	140	171	202	233	264	295	318
	19	49	80	111	142	173	204	235	266	297	319
	21	51	82	113	144	175	206	237	268	299	320
	23	53	84	115	146	177	208	239	270	301	321
	25	55	86	117	148	179	210	241	272	303	322
	27	57	88	119	150	181	212	243	274	305	323
	29	59	90	121	152	183	214	245	276	307	324

ROW Individual Edge number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
	32	34	36	38	40	42	44	46	48	50	52	54	56	58	60	61
	63	65	67	69	71	73	75	77	79	81	83	85	87	89	91	92
	94	96	98	100	102	104	106	108	110	112	114	116	118	120	122	123
	125	127	129	131	133	135	137	139	141	143	145	147	149	151	153	154
	156	158	160	162	164	166	168	170	172	174	176	178	180	182	184	185
	187	189	191	193	195	197	199	201	203	205	207	209	211	213	215	216
	218	220	222	224	226	228	230	232	234	236	238	240	242	244	246	247
	249	251	253	255	257	259	261	263	265	267	269	271	273	275	277	278

FIGURE 61: Representation of polyedges and their relationship to force densities in a creased dataset. (Own Work)

5.3 MAKING THE POLYEDGES

The COMPAS library was used to generate mesh geometries using python and grasshopper. A ghpython script (python script in Grasshopper) was provided by Robin Oval which generated a mesh for constant force densities for all edges. The script was modified to include grouping of edges into polyedges and assigning force densities to these polyedges separately. Another script was provided by Oval which included already grouped polyedges but due to software compatibility issues, it was not used.

In order to group the edges into polyedges, the output of the compas script was analyzed from Grasshopper. This numbering of edges was studied for different mesh densities to derive patterns for numbering the mesh edges. It was found that 7 basic numbering sequences existed. The sequence repeated as the mesh-density in the y-axis was increased. This is visualized in Figure 62. Therefore, the same numbering pattern could be used for different mesh densities in the y-axis. As shown in Figure X, the same numbering sequence happens when y=27 as well as when y=13. The first polyedge is shown in the figure with which the numbering begins.

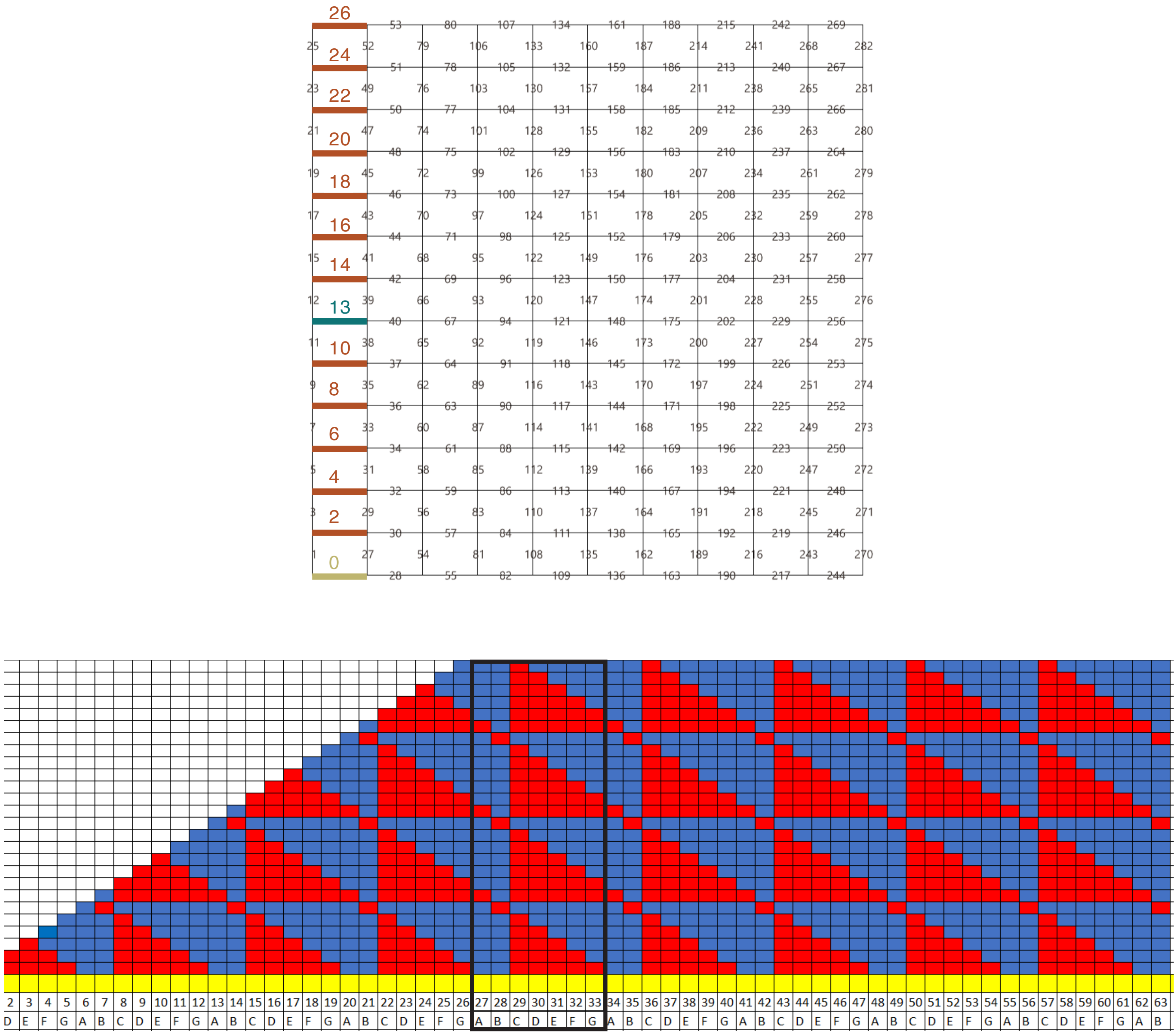


FIGURE 62: Pattern recognition of the first polyedge of a mesh of y dimension from COMPAS meshes in Grasshopper. The 7 repeating patterns are highlighted (Own Work)

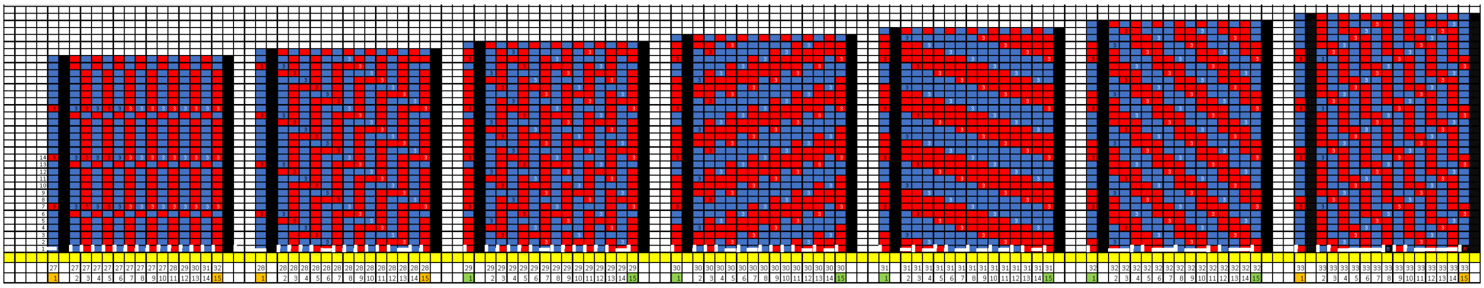


FIGURE 63: The 7 patterns of how different dimensioned COMPAS meshes are ordered in Grasshopper. Pattern recognition of all edges for a specific y-axis dimension from COMPAS meshes in Grasshopper. (Own Work)

Figure 63 shows the 7 different patterns highlighted earlier in Figure 62 as A,B,C,D,E,F,and G. The pattern represents the numbering sequence for the rest of the polyedges.

However, it was found that the sequence changed if the COMPAS mesh was generated in python rather than in the Grasshopper environment. These conflicts were resolved and the sequence was updated accordingly. An example of the conflicts is shown in Figure 64.

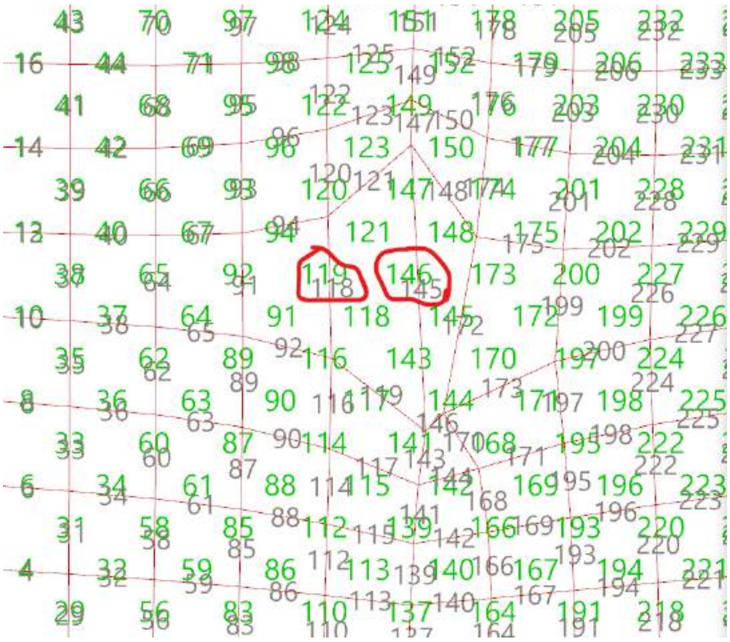


FIGURE 64: Discrepancy in edge numbering patterns in meshes formed in Grasshopper and directly in python in COMPAS. (Own Work)

5.4 DATASETS

Different types of datasets were created to study the effect of different parameters to see what was the effect on performance. 10,000 samples were to be generated for the final dataset to be used for the Generator.

- Uniform force densities
- Creases
- Randomized force densities

Creases allowed controlled variability while increasing the design space keeping randomness limited to specific polyedges for a given sample. Randomized force densities, however, introduced complete randomness (within a probability distribution) for all polyedges for a given sample.

5.4.1 Uniform force densities

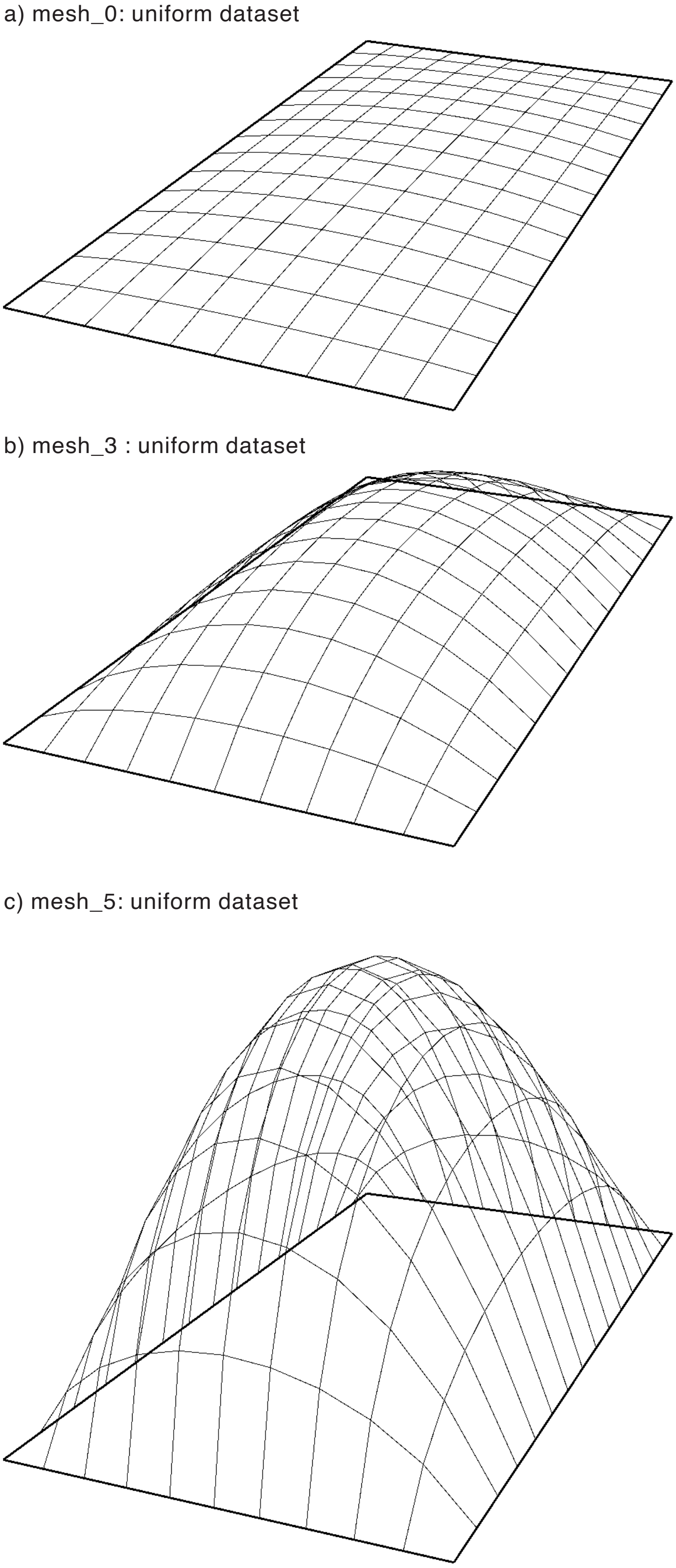
The script generated vaults where force densities of all polyedges was the same. This is termed as uniform mesh density throughout the rest of the paper.

A high value of force density corresponds to a greater force per unit length for edge. This creates shallower vaults for higher force densities. The change in seismic performance of varying the force densities for uniform vaults will be discussed in the next chapter.

Table 05 shows examples of uniform mesh densities and the corresponding meshes are visualized in Figure 65.

DATASET: uniform force densities						
Edge number	mesh number					
	0	1	2	3	4	5
column_1	5	3	2	1	0.6	0.3
column_2	5	3	2	1	0.6	0.3
column_4	5	3	2	1	0.6	0.3
column_5	5	3	2	1	0.6	0.3
column_6	5	3	2	1	0.6	0.3
column_7	5	3	2	1	0.6	0.3
column_8	5	3	2	1	0.6	0.3
column_9	5	3	2	1	0.6	0.3
column_10	5	3	2	1	0.6	0.3
column_11	5	3	2	1	0.6	0.3
row_1	5	3	2	1	0.6	0.3
row_2	5	3	2	1	0.6	0.3
row_3	5	3	2	1	0.6	0.3
row_4	5	3	2	1	0.6	0.3
row_5	5	3	2	1	0.6	0.3
row_6	5	3	2	1	0.6	0.3
row_7	5	3	2	1	0.6	0.3
row_8	5	3	2	1	0.6	0.3
row_9	5	3	2	1	0.6	0.3
row_10	5	3	2	1	0.6	0.3
row_11	5	3	2	1	0.6	0.3
row_12	5	3	2	1	0.6	0.3
row_13	5	3	2	1	0.6	0.3
row_14	5	3	2	1	0.6	0.3
row_15	5	3	2	1	0.6	0.3
row_16	5	3	2	1	0.6	0.3
row_17	5	3	2	1	0.6	0.3

TABLE 05: Uniform Force Densities Dataset. (Own Work)


a) mesh_0: uniform dataset

b) mesh_3 : uniform dataset

c) mesh_5: uniform dataset

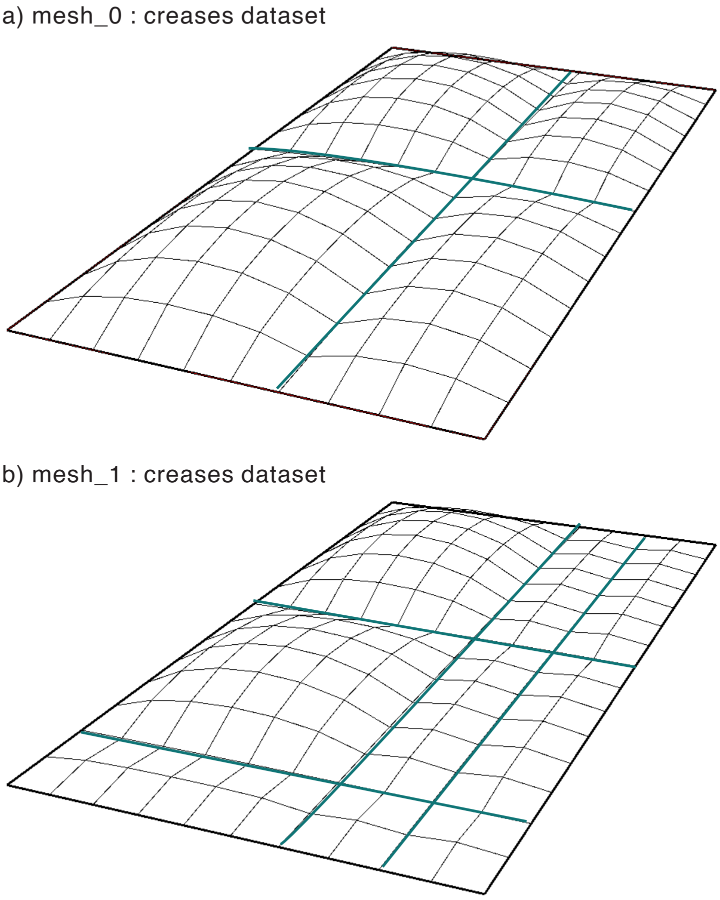
FIGURE 65: Meshes from the Uniform Force Densities Dataset. (Own Work)

5.4.2 Creases

Different creases were formed by increasing the force density of specific polyedges while keeping force-densities of the rest constant. In this case, the force densities of the rest of the polyedges were kept 1 while the force densities of the creases were randomly sampled from a uniform distribution of mean = 150 and standard deviation = 30. Figure 66 visualizes the creases corresponding to a sample from the dataset.

Constraints were coded inside the script for crease formation such that a minimum distance between consecutive parallel creases would be maintained. This distance would also be maintained between the edge and the first crease. This was done to avoid flattening of meshes due to the presence of consecutive high force densities spaced apart at short distances. Flat meshes perform poorly (as later highlighted in Section 6.2.2). Another reason for the introduction of these constraints were practical considerations such as constructability in making densely creased vaults.

To generate the dataset, all possible crease combinations were extracted using itertools in python, where each crease was assigned a force density sampled randomly from the uniform distribution.


a) mesh_0 : creases dataset

b) mesh_1 : creases dataset

FIGURE 66: Meshes from the Creased Force Densties Dataset. (Own Work)

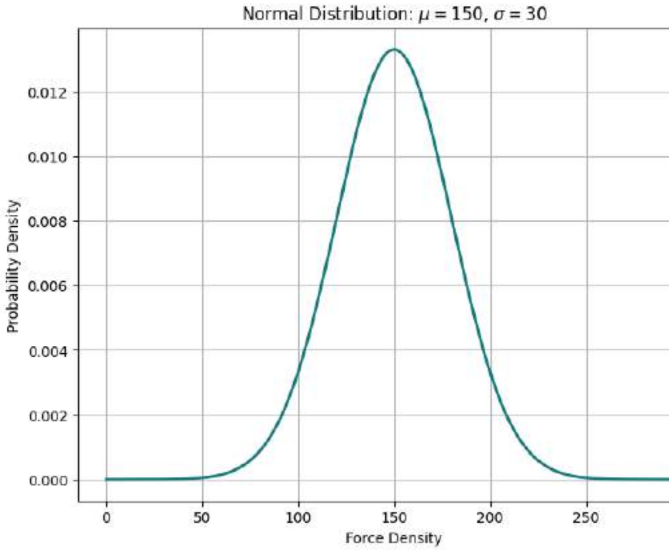


FIGURE 67: Normal Distribution force densities at creases. (Own Work)

DATASET: creases						
Edge number	mesh number					
	0	1	2	3	4	5
column_1	1	1	1	1	1	1
column_2	1	1	1	1	1	1
column_4	1	1	1	137	137	170
column_5	1	1	1	1	1	1
column_6	1	1	94	1	1	1
column_7	118	100	1	1	1	1
column_8	1	1	1	127	1	1
column_9	1	120	1	1	147	1
column_10	1	1	1	1	1	140
column_11	1	1	1	1	1	1
row_1	1	1	1	1	1	1
row_2	1	1	1	1	1	1
row_3	1	1	1	1	1	1
row_4	1	1	93	1	198	93
row_5	1	1	1	1	1	1
row_6	1	1	1	1	1	1
row_7	1	1	1	1	1	1
row_8	95	131	1	1	1	1
row_9	1	1	1	1	1	1
row_10	1	1	1	1	1	1
row_11	1	1	1	1	1	1
row_12	1	1	1	217	1	1
row_13	1	1	1	1	1	1
row_14	1	96	1	1	1	1
row_15	1	1	1	1	1	1
row_16	1	1	1	1	1	1

TABLE 06: Creased Dataset. (Own Work)

The introduction of constraints and combinations allowed for greater control over the crease dataset. However, this also meant greater inclusion of user bias. This would also be reflected in sampling from the VAE which would only produce samples similar to the dataset; this is later covered in Section 7.1.1.

5.4.3 Randomized force densities

In this dataset, bias was removed from the Geometry Generation process. Instead of assigning randomized force densities to specific polyedges, all polyedges were assigned randomized values. While the normal distribution used for creases only gave very high force densities, a different distribution was required which would result in a high probability of values and a low probability of high ones. Force densities were randomly sampled from values normally distributed in a logarithmic scale. This was done to increase the probability of sampling lower force densities than higher ones as a mesh containing high values would result in a flattened mesh as shown in Figure 69b.

This method reduced control over the generated dataset. This allowed for generation of less desirable samples in terms of seismic performance. At the same time, it also allowed for generation of more novel samples that may not have been considered intuitively of instead of ones pre-conceived by the architect/engineer.

For different datasets, the effects of different parameters on performance will be elaborated in Section 6.2.5. This section deals only with generation of the geometry for these datasets.

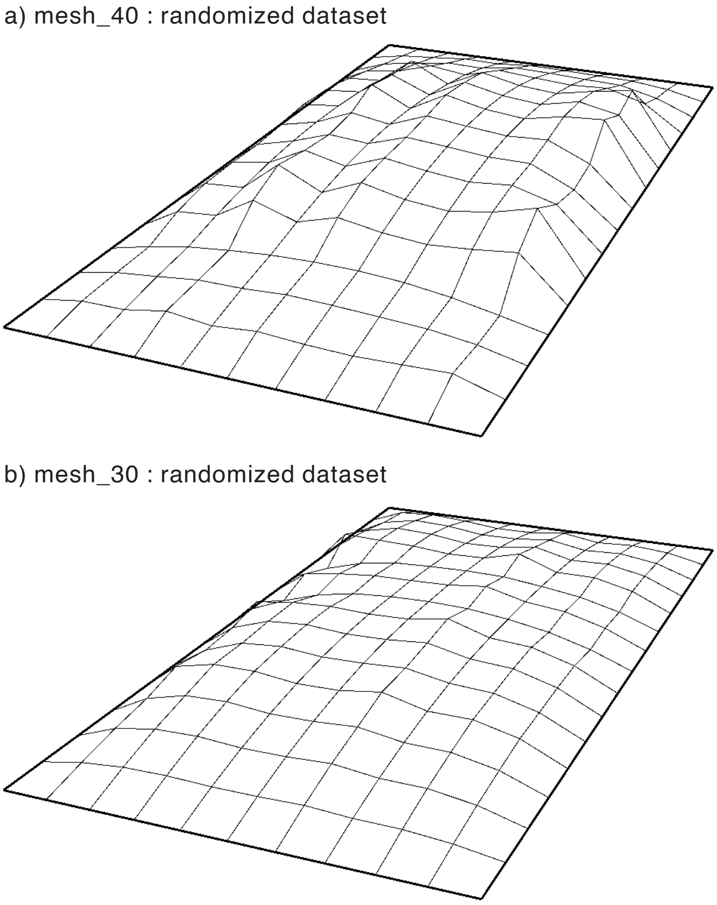
```
log_min = math.log(0.2)

log_max = math.log(100)

mean = math.log(1)

standard_deviation = (log_max - log_min) /4
```

FIGURE 68: Python code for generating distribution for randomized force densities. (Own Work)


a) mesh_40 : randomized dataset

b) mesh_30 : randomized dataset

FIGURE 69: Meshes from Randomized Force Densities Dataset. (Own Work)

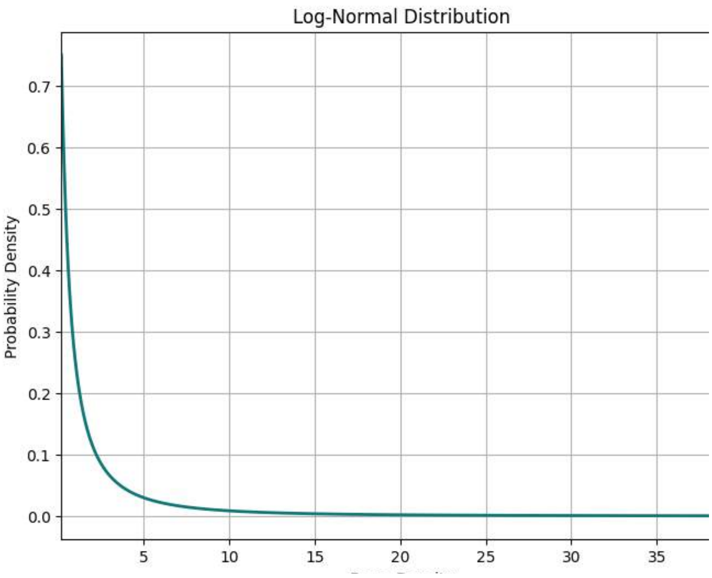
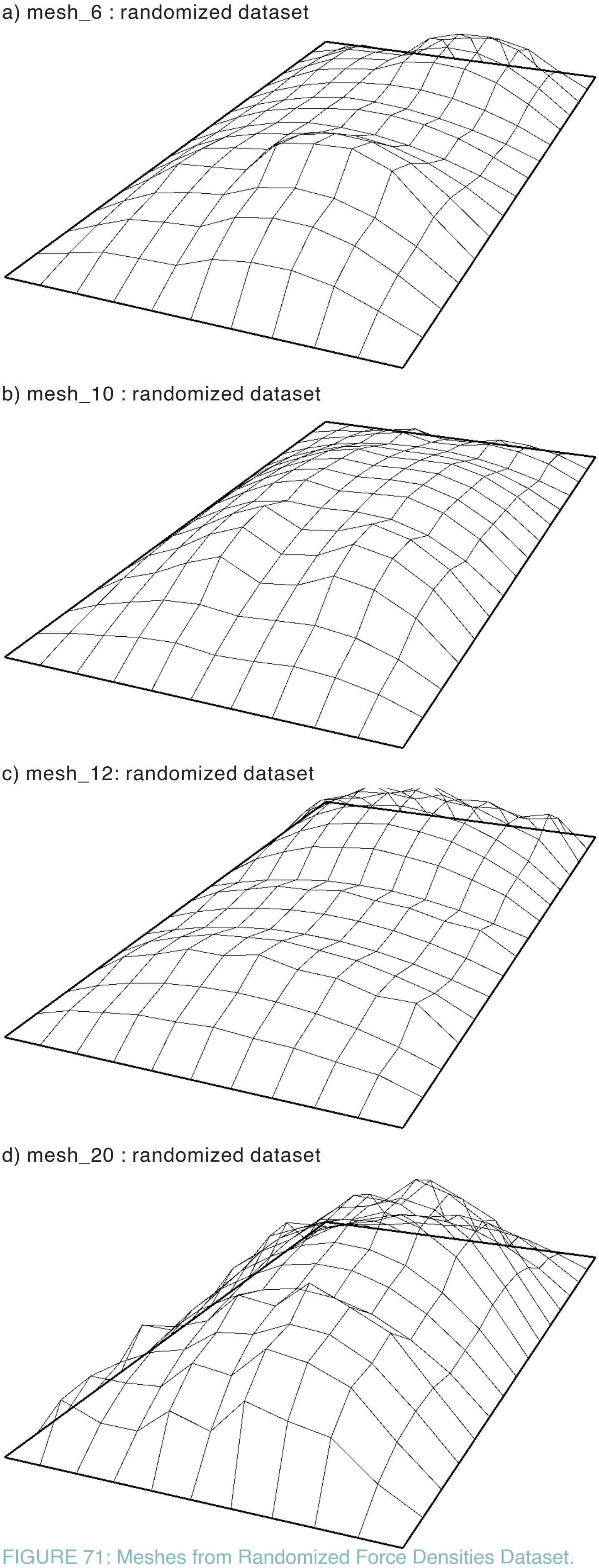


FIGURE 70: Log Normal Distribution of randomized force densities. (Own Work)

DATASET: randomized						
Edge number	mesh number					
	6	10	12	20	30	40
column_1	0.1	6.3	2.6	2.1	0.5	0.2
column_2	1.1	0.8	0.1	1.3	2	0.1
column_4	0.2	0.6	1.1	0.2	1.7	1.4
column_5	0.2	0.5	0.5	0.3	9.9	5.5
column_6	0.3	3.1	1	0.1	0.5	4.4
column_7	0.2	6	0.3	1.5	7	1
column_8	4.6	0.4	1	0.2	8	10
column_9	1.9	1.3	0.4	2.9	2.2	0.3
column_10	0.6	4.1	1	1.4	0.9	8.5
column_11	4.3	3.7	0.6	0.1	0.2	1.4
row_1	10	1.7	12.3	3.9	0.8	6.6
row_2	1.9	0.9	3	0.5	3	1.7
row_3	0.3	0.3	0.2	1.4	0.7	2.3
row_4	0.4	0.4	0.1	0.6	1	1.6
row_5	1	6.5	0.4	3.3	0.7	0.8
row_6	0.8	1.6	1.2	0.1	0.1	0.1
row_7	3.2	0.6	1.1	0.2	4.9	0.3
row_8	5.3	1	0.8	0.5	0.3	0.1
row_9	1.4	0.9	12.1	0.3	5.2	0.6
row_10	4.7	0.4	0.4	1.3	0.2	0.8
row_11	1.6	2.2	4.2	0.9	0.9	0.4
row_12	0.6	0.1	7.1	2.1	1.2	0.1
row_13	0.1	0.2	1.2	0.1	0.4	0.2
row_14	1.6	0.1	0.2	0.6	1.2	14.5
row_15	0.8	2	3.6	0.2	1.4	2.5
row_16	0.8	1	1	0.1	2.8	0.6
row_17	0.1	2.2	0.7	1.5	1.1	0.9

TABLE 07: Randomized Force Densities Dataset. (Own Work)



a) mesh_6 : randomized dataset

b) mesh_10 : randomized dataset

c) mesh_12: randomized dataset

d) mesh_20 : randomized dataset

FIGURE 71: Meshes from Randomized Force Densities Dataset. (Own Work)

See Section 4.1.2 for the workflow.

The design spectrum was calculated. Figure 72 shows the spectrum. All 10,000 samples had Time Periods in the range between 0.08s and 0.13s.

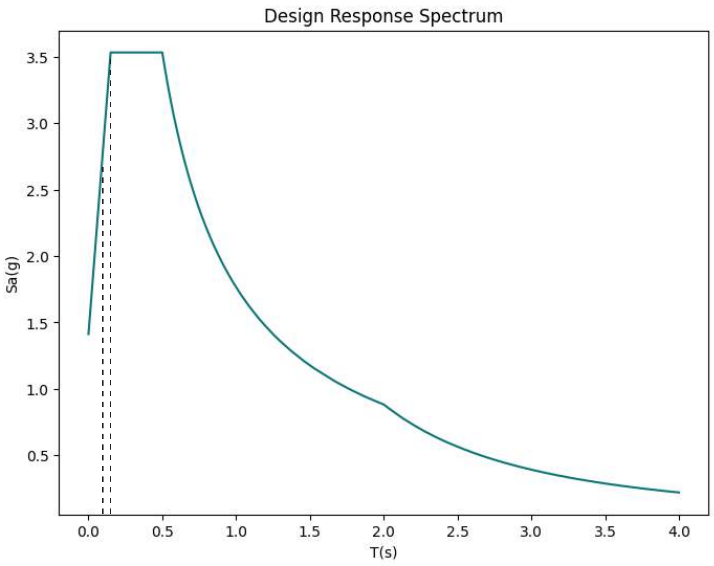


FIGURE 72: Response Spectrum of Model. (Own Work)

6.1 BOUNDARY CONDITIONS

The vault was modeled with different boundary conditions. Initially, the vault was modeled as directly supported on the floor for simplicity. Later it was supported on the structure representative of the actual house. Adding a complexity incrementally allowed analysis of the vault as an isolated element initially and then analyzing it in the broader context with the rest of the structure.

To check whether a vault may behave similar in different storey structures, the force densities were kept constant but the loading was varied. As mentioned earlier, this would equally affect all of the polyedge force densities uniformly. Figure 73 shows the performance of the same basic vault structure with increasing loading for a 3-storey structure, 1-storey structure, and one that is directly supported on the floor. It was interesting to note that there were behavioural changes noted in the different cases. The best seismically performing vault in terms of lowest utilization, highest buckling load factor, and lowest displacement, was not the same in all cases (1-storey, 3-storey, isolated vault).

Therefore, it can be concluded that seismic performance of a vault is unique for a particular structure in relation to the number of storeys that structure has. This is makes sense as a taller/shorter structure would influence the Time Period of its natural vibration and therefore would be affect the base shear.

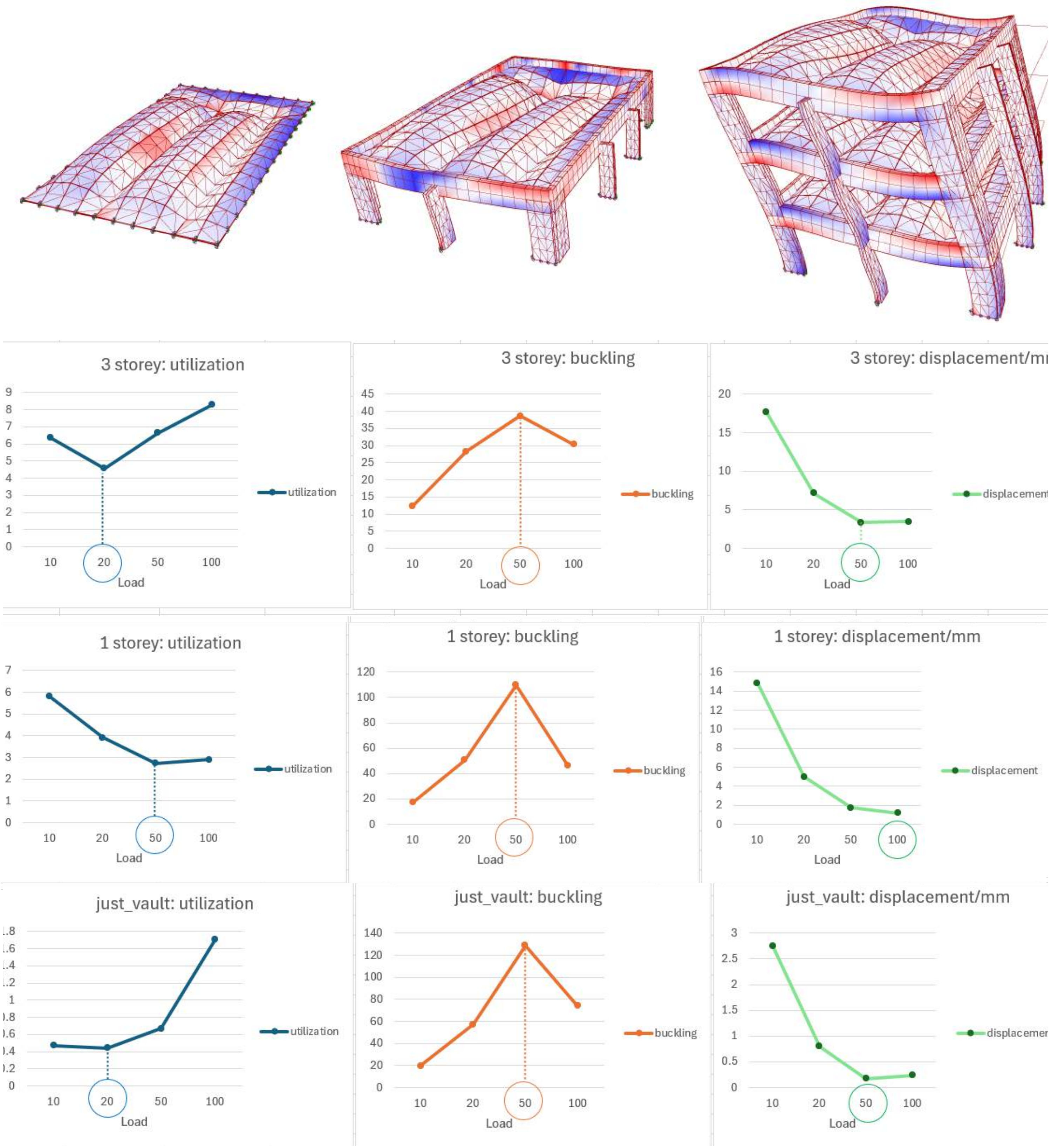


FIGURE 73: Performance for single storey, 3-storey and vault supported on ground. (Own Work)

6.1.1 Vault

From the results, the following were known for a vault (supported directly on the ground):

- Translational mass participation in lateral directions x and y is negligible. There is a high mass participation in the z axis (axis parallel with gravity)

- Rotational mass participation in x and y is high. There is negligible mass participation in the rotational z axis

For seismic analysis, considering that conventionally the dominant mode is taken in the direction of the lateral load (translation x or y) but it is negligible in these cases, the model without supporting walls may not represent the actual dominant mode as one with supporting walls. Moreover, the seismic weight would also be significantly less than one with the rest of the structure. This would give a seismic force much smaller than one in reality. Due to these reasons, it would not be correct that seismic analysis done on an isolated vault supported on the ground could be used for a structure where it is used as a roof slab/floor slab of the upper storey.

Once results from the Karamba model were validated with those from the ABAQUS and Alpaca Model, the rest of the structure was modeled adding more complexity in the next step.

Modal Analysis validation		
Mode number	Natural Frequency / Hz	
	Karamba	Abaqus
Mode 1	3.64	3.74
Mode 2	6.42	6.35
Mode 3	6.81	6.71
Mode 4	8.94	8.95
Mode 5	9.47	10.31
Mode 6	11.84	11.21

TABLE 08: Comparison of Natural Frequency of the same mesh through validation in Abaqus of Karamba model results. (Own Work)

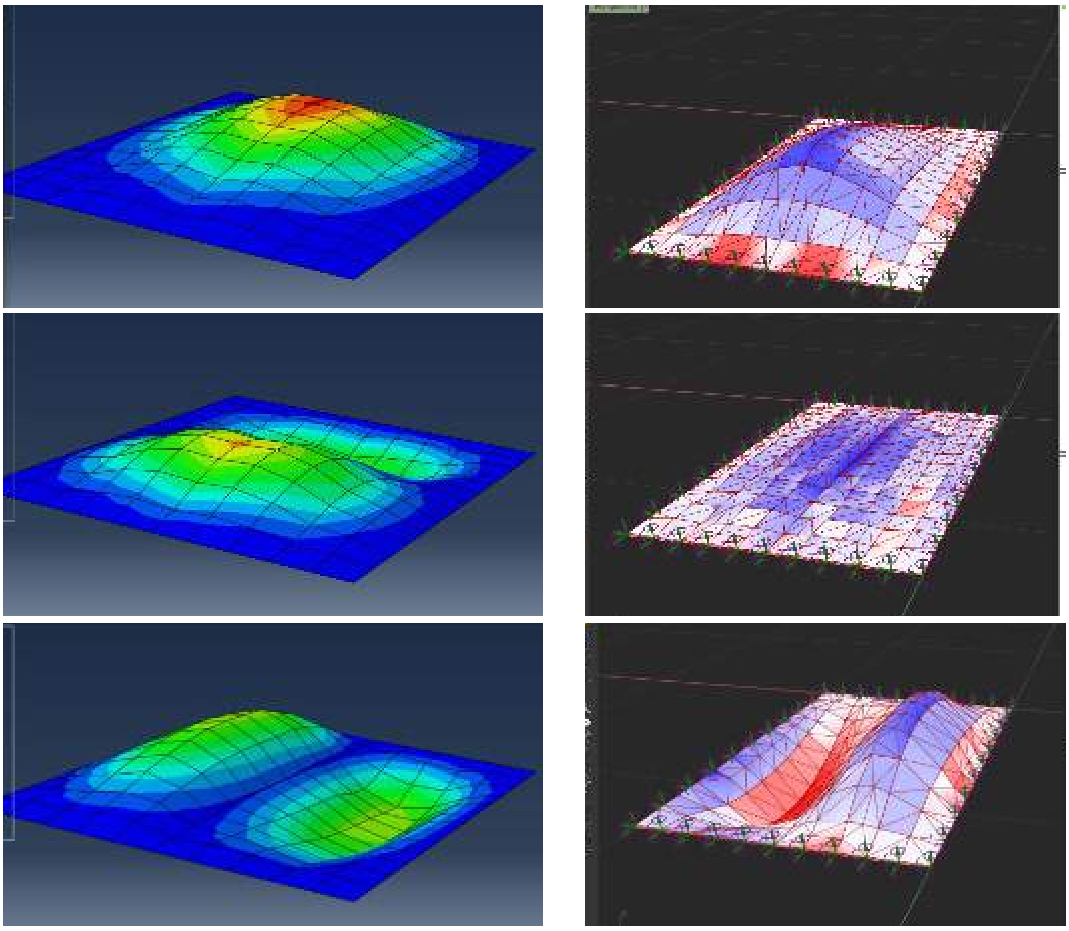


FIGURE 74: Modal Analysis in Abaqus (left) and in Karamba (right) showing first 3 modes. (Own Work)

6.1.2 Vault + rest of the structure

The structure was assumed as a one-storey building where the vault was the roof slab.

It was considered that load-bearing walls would support the vault and the walls would be made from earth having the same material properties as the earth material in macro-model (See Table 03 in Section 4.2.2 for reference).

Walls

This was initially modelled as walls in Abaqus. As shown in Figure 75, the results from the modal analysis show atypical behaviour in the different modal states. These elements were found to be very stiff and did not represent masonry walls realistically – especially because there are no openings for doors and windows. Similar results were also found in Karamba.

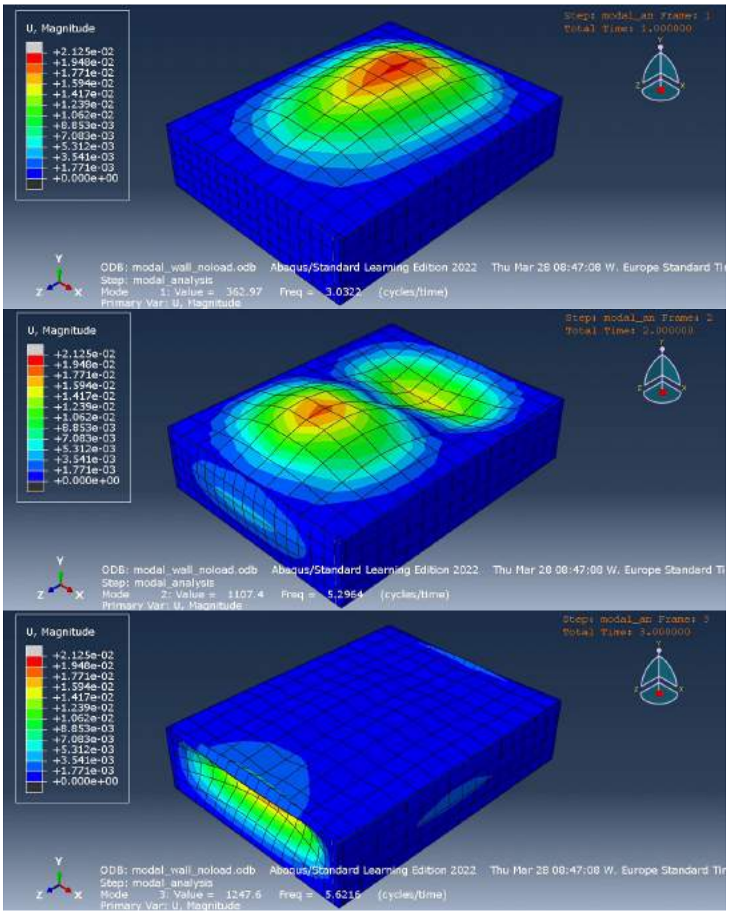


FIGURE 75: Modal Analysis showing first 3 modes of 1 storey vault with walls modeled as solids in Abaqus. (Own Work)

Columns

Next, instead of the walls, bernoulli elements were used to model columns with equivalent bending stiffness as the walls. Two different cross-sections were calculated – one for the corner columns and one for the other columns. This approach was in line with estimated behaviour in the modal analysis. This is shown in Figure 79

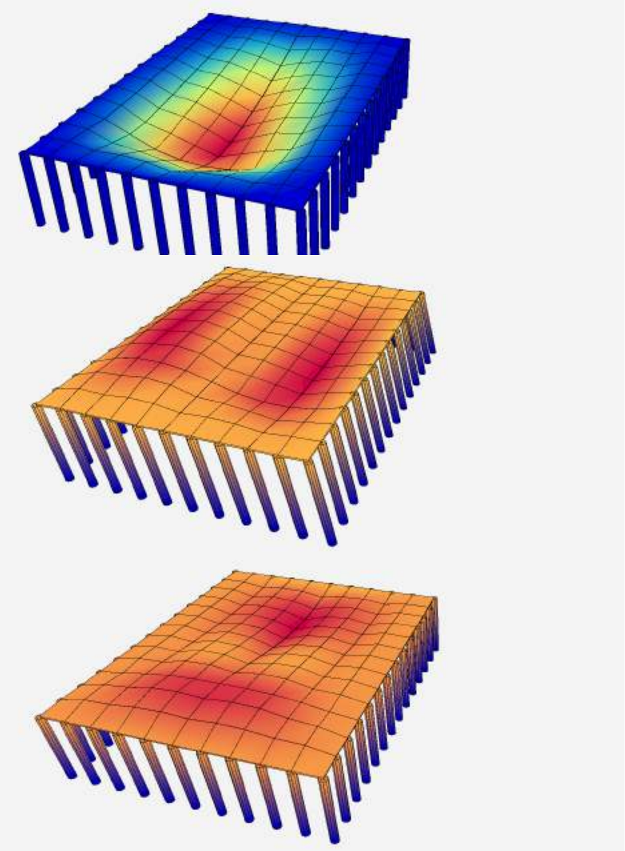


FIGURE 76: Modal Analysis showing first 3 modes of 1 storey vault with columns in Alpaca4d. (Own Work)

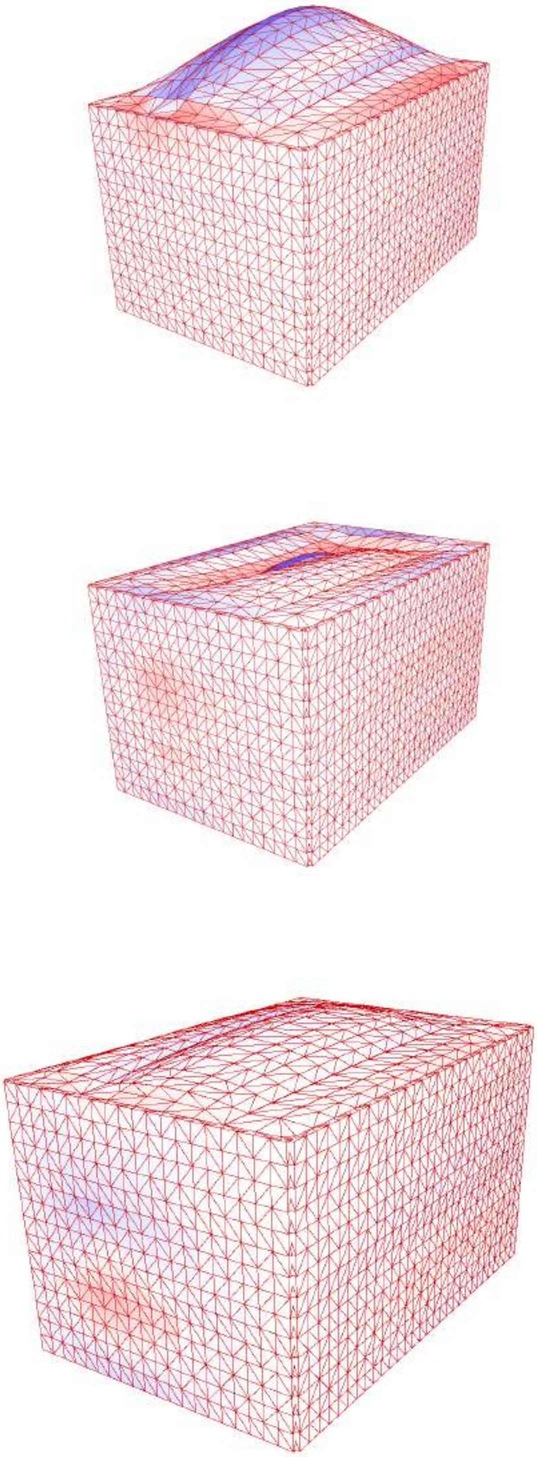


FIGURE 77: Modal Analysis showing first 3 modes of 3 storey vault with walls modeled as shells in Karamba. (Own Work)

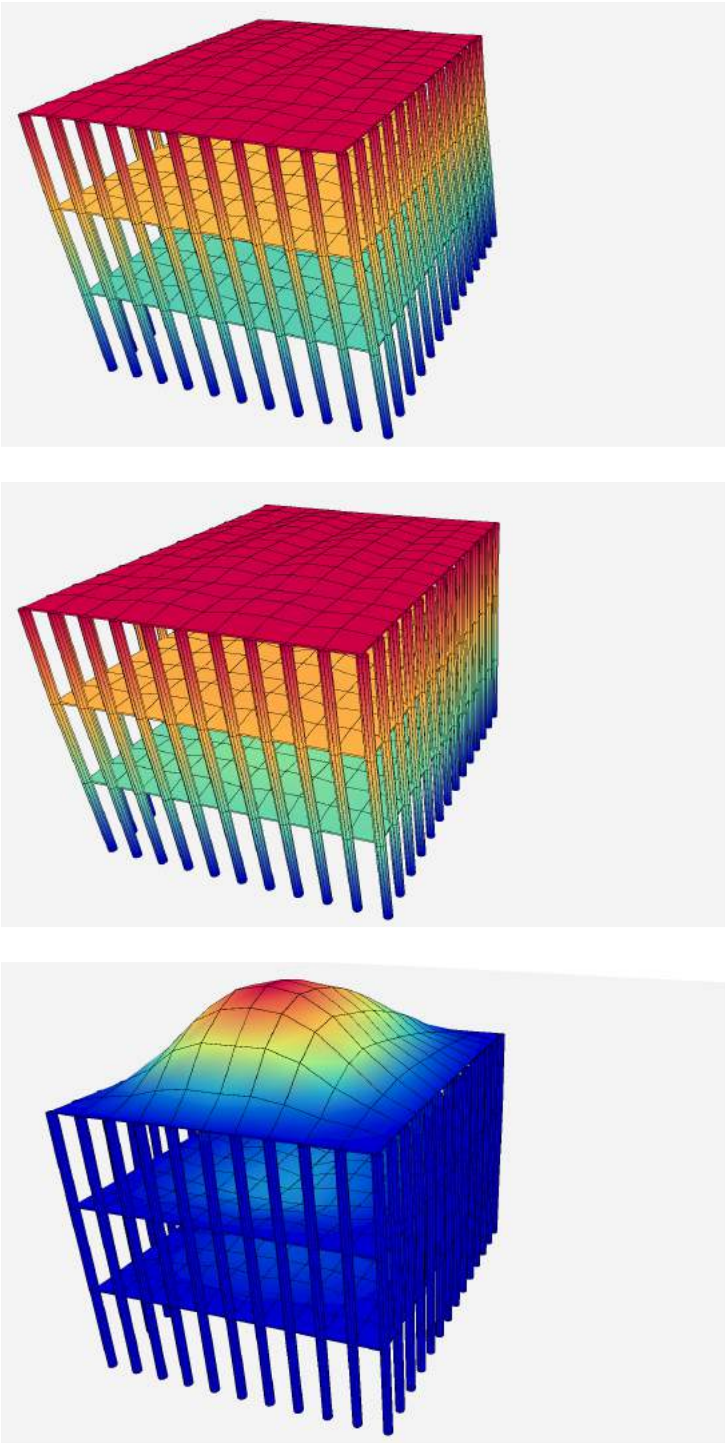
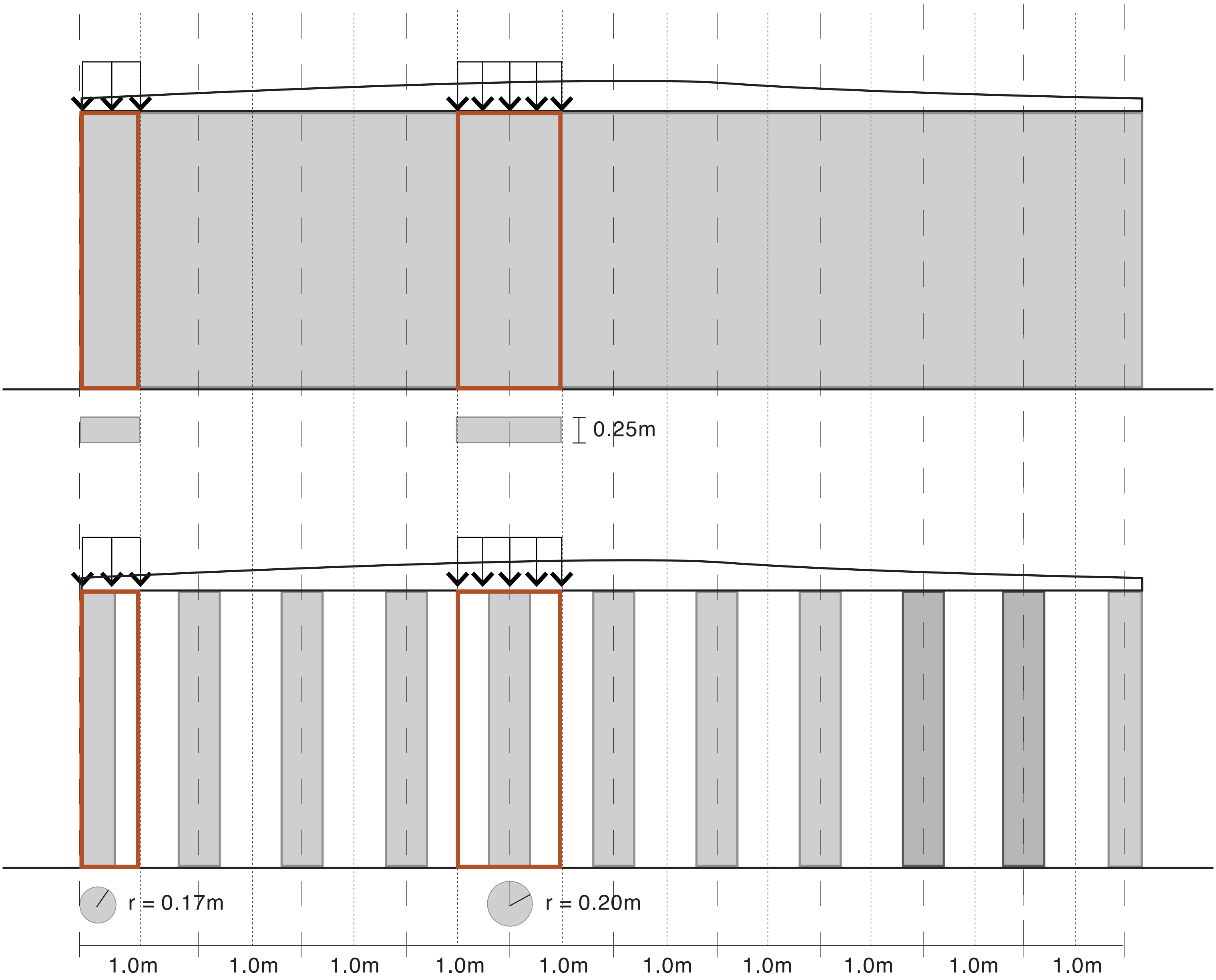


FIGURE 78: Modal Analysis showing first 3 modes of 3 storey vault with columns in Alpaca4d. (Own Work)

Figure 77 and Figure 78 show how different the modes of vibrations are with columns as compared to walls

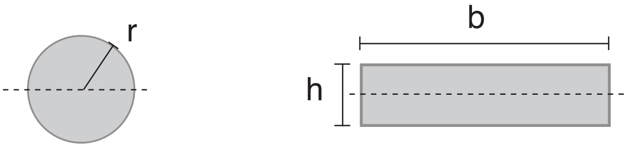


stiffness of column section = stiffness of equivalent wall section

$$EI_{wall} = EI_{column}$$

$$I_{wall} = I_{column}$$

$$bh^3/12 = \pi .r.4/ 4$$



However, this approach did not capture the shear stiffness correctly. An amendment was suggested to include appropriate bracings between consecutive columns for this that would account for the loss in shear stiffness. This would have also been a viable option but the structure was revisited afterwards which removed the application of this model.

FIGURE 79: Equivalent Bending Stiffness Column stiffness to that of walls. (Own Work)

Shear Walls

The overall structure of the building was revisited and it was reconsidered whether a building with earthen masonry walls would be appropriate against seismic activity. It was found in the literature that such structures with unreinforced masonry are highly susceptible to collapse in seismic areas due to the brittle nature of the material and its high seismic weight. Appropriate strategies for reinforcement were identified suitable for the type of construction that the project entails; as the building typology was for low-income homes in remote areas, construction guidelines for non-engineered structures in seismic regions were studied. These included local guidelines (AKAH Seismic Housing Report BACIP) and international guidelines (Auroville Earth Institute).

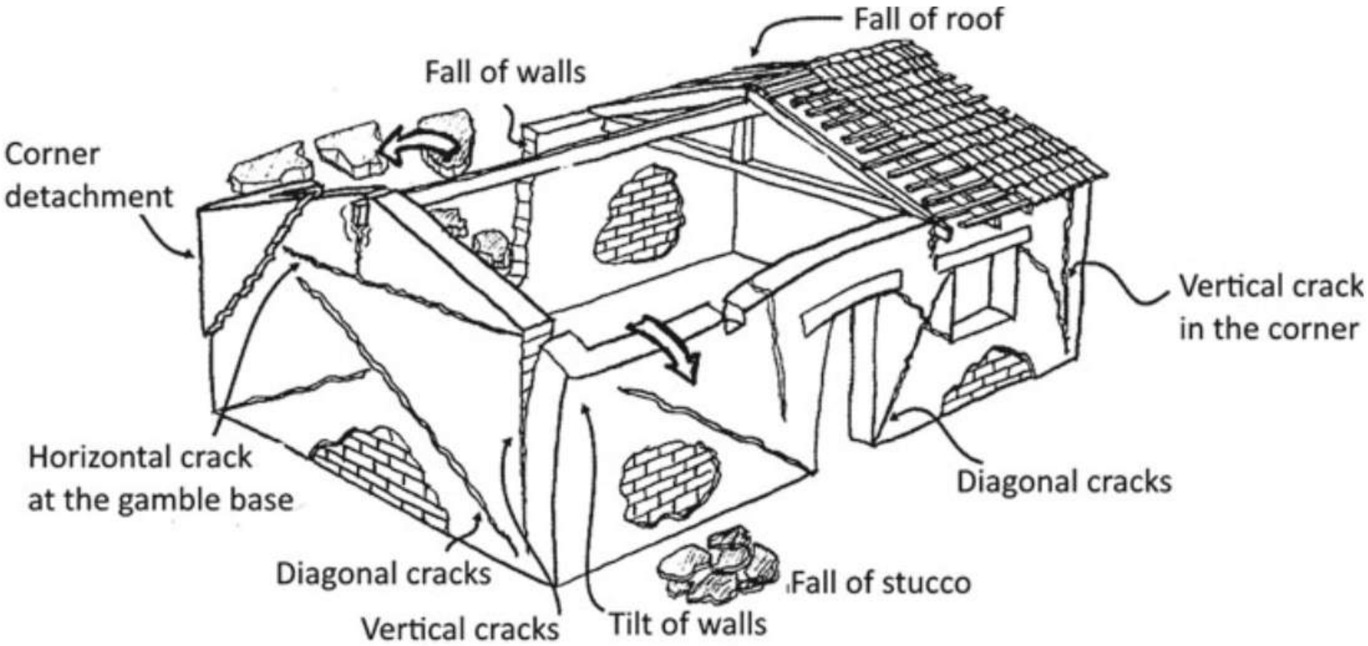


FIGURE 80: Vulnerability of Adobe Structures in Seismic events. Tarque, N., Sayın, E., Rafi, M.M., Tolles, E.L. (2021). Behaviour of Adobe Construction in Recent Earthquakes. In: Varum, H., Parisi, F., Tarque, N., Silveira, D. (eds) Structural Characterization and Seismic Retrofitting of Adobe Constructions. Building Pathology and Rehabilitation, vol 20. Springer, Cham. https://doi.org/10.1007/978-3-030-74737-4_2

The structure was composed of a ring beam that supported the vault-slab which was supported, in turn, by shear walls. Unreinforced walls were not modelled. The shear walls were placed at corners of the structure which are the most vulnerable places for tensile cracks to form. Long unreinforced walls greater than 4.5m were supported by external buttresses with depth 0.75m. This is equivalent to 3 times the thickness of the walls (0.25m) as required as shown in Figure 81. This was the final structure that was used for running the Performance Evaluation on the dataset (See Figure 82).

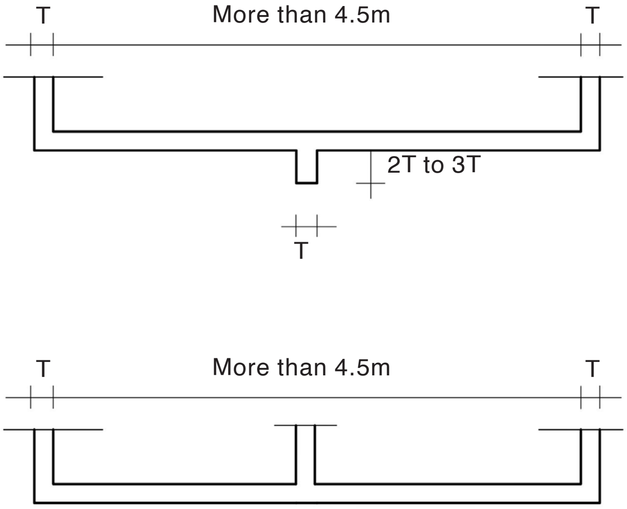


FIGURE 81: Butressing / Crosswalls needed for unreinforced masonyry walls greater than 4.5m. https://dev.earth-auroville.com/

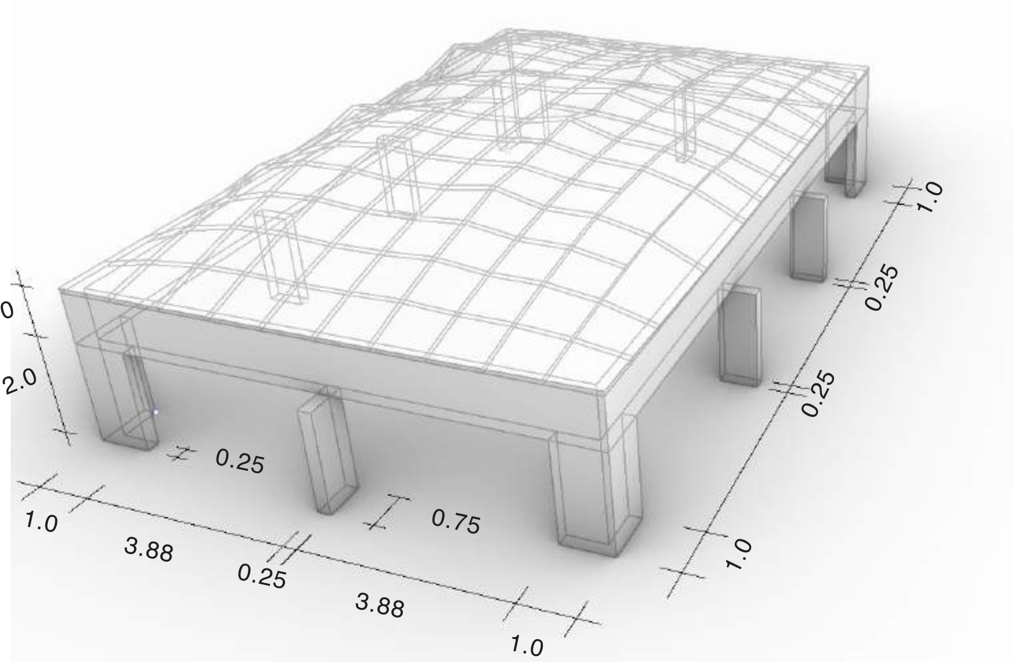


FIGURE 82: Final structure with shear walls and ring beams. (Own Work)

Appropriate literature could not be found to estimate the material properties for reinforced masonry so material was assumed as unreinforced masonry as that of the macro-model of the vault; an assumption was made that high tensile zones would be identified, and tensile cracks would be accommodated by adding reinforcement in those places at a later stage in the design process. The same assumption was also valid for the vault itself where basalt geo-grid masonry would be applied at a later stage. For details into the basalt geo-grid reinforcement, please refer to Section 2.3.4.

Table 09 shows the performance indicators that were taken out from the Performance Evaluation. However, for the surrogate model, only the Buckling Load Factor, Utilization, and Interstorey Drift Ratios were used to account for failure in Stability, Strength, and Stiffness respectively (See Table 10). Table 12 also shows the properties of the modal analysis for the selected samples shown. This was done to check the mass participation for each sample and their Time Periods, and whether it met the requirements for Eurocode.

Simulations were carried out with the applied force in x direction and y direction separately. For every sample, the lower performance in x and y was taken (minimum buckling loading factor, maximum utilization, maximum interstorey drift).

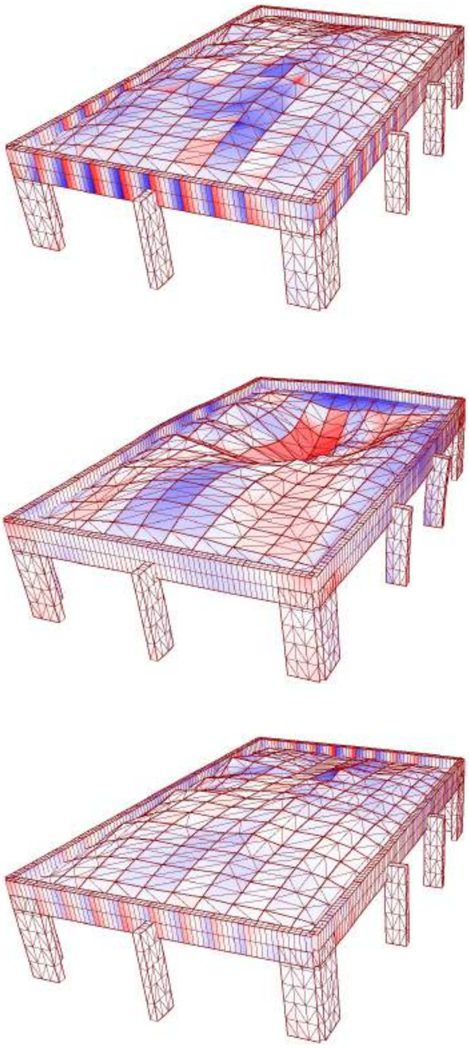


FIGURE 83: Modal Analysis of final structure in Karamba. First 3 modes are shown. (Own Work)

DATASET: randomized					
Performance metric	Mesh number				
	mesh_1	mesh_2	mesh_3	mesh_4	mesh_5
Buckling_Load_Factor	20.384	2.468	20.807	17.458	21.273
Utilization	12.469	11.034	5.908	10.041	9.830
Interstorey_Drift_Ratios	0.0033542h	0.00841904h	0.00197305h	0.00251683h	0.00287915h
Avg_Displacement/mm	10.051	21.939	6.192	9.438	8.500
Max_Displacement/mm	30.015	76.820	17.778	26.988	26.866
Avg_Shear_Force/KN	1.233	0.370	1.249	1.568	1.368
Max_Shear_Force/KN	12.666	3.052	9.481	13.098	16.628
Avg_Bending_Moment/KNm	0.614	0.092	0.469	0.722	0.559
Max_Bending_Moment/KNm	5.055	0.580	2.278	3.965	4.088
Max_Compresive_Stress/MPa	-0.996	-1.551	-0.315	-0.804	-0.802
Max_Tensile_Stress/MPa	3.314	3.222	1.492	2.719	2.615
Max_Principal_Stress/MPa	0.022	0.035	0.017	0.019	0.020

TABLE 09: All Performance Metrics of Randomized Force Densities Dataset. (Own Work)

DATASET: randomized					
Performance metric	Mesh number				
	mesh_1	mesh_2	mesh_3	mesh_4	mesh_5
Buckling_Load_Factor	20.384	2.468	20.807	17.458	21.273
Stability_Failure?	Safe	Safe	Safe	Safe	Safe
Utilization	12.469	11.034	5.908	10.041	9.830
Strength_Failure?	Failure	Failure	Failure	Failure	Failure
Interstorey_Drift_Ratios	0.0033542h	0.00841904h	0.00197305h	0.00251683h	0.00287915h
Stiffness_Failure?	Safe	Safe	Safe	Safe	Safe

TABLE 10: Performance Metrics to be used as labels for surrgoate model of Randomized Force Densities Dataset, and Failure state. (Own Work)

Though only 5 samples are shown in Table 10, they show a significant trend of failure in strength (utilization) of the vaults but no failure in buckling.

DATASET: randomized			
Mesh	Physical Characteristics		
	Mass / kg	Height of vault / m	Thickness / m
mesh_1	39552.84929	1.474194	0.095
mesh_2	28502.54607	1.562264	0.035
mesh_3	39842.25422	1.505192	0.095
mesh_4	41598.92123	2.361411	0.095
mesh_5	39663.53523	1.665554	0.095
mesh_6	29194.68304	2.461413	0.035
mesh_7	42119.57844	2.606465	0.095
mesh_8	32719.5475	1.643492	0.06
mesh_9	32567.36816	1.255298	0.06
mesh_10	40975.36613	2.105192	0.095
mesh_11	30495.84377	3.316291	0.035

TABLE 11: Physical characteristics of Randomized Force Densities Dataset. (Own Work)

DATASET: randomized							
Mesh number	Modal Analysis Metrics						
	Mass Participation total cumulative (x,y,z) / %	Time Period / s	Mass Participation total / %	Mode	Mass Participation for dominant modes / %	Base shear / KN	Eurocode satsisfied?
mesh_1	[82.75, 115.82, 48.92]	[0.11]	100.91	[3]	[100.91]	130.37	yes
mesh_2	[55.04, 48.52, 19.79]	[0.09]	36.6	[9]	[16.52]	85	no
mesh_3	[78.79, 67.25, 47.51]	[0.11]	54.34	[3]	[27.07]	131.33	no
mesh_4	[119.89, 106.52, 59.73]	[0.11]	92.07	[5]	[60.81]	137.12	yes
mesh_5	[111.76, 68.48, 50.2]	[0.11]	54.5	[3]	[27.85]	130.74	no
mesh_6	[69.42, 66.21, 21.63]	[0.1, 0.08]	48.3	[8, 11]	[19.52, 11.76]	123.3	no
mesh_7	[112.85, 118.38, 60.8]	[0.11]	90.89	[4]	[90.89]	138.83	yes
mesh_8	[131.91, 106.32, 37.63]	[0.1]	95.22	[7]	[47.68]	102.71	yes
mesh_9	[90.84, 98.73, 39.72]	[0.1]	74.18	[8]	[66.36]	102.24	no
mesh_10	[120.31, 116.9, 58.6]	[0.11]	98.3	[4]	[37.57]	135.06	yes
mesh_11	[98.56, 75.71, 23.59]	[0.1, 0.08]	48.09	[7, 11]	[21.09, 13.04]	128.79	no

TABLE 12: Modal analysis results of Randomized Force Densities Dataset. (Own Work)

6.2 PERFORMANCE COMPARISON

6.2.1 Variation in Seismic Zone

20 samples each from 3 different types of datasets were tested in high, medium, and low seismicity and analyzed. The 3 three different seismicity levels could be tested with ground accelerations inside the same province, Balochistan. In Zone 2A (0.12 g), the location was the in the Chaghi District Zone. A location in Zone 2B (0.18g) was in the Khuzdar District while a location in Zone 3 (0.28g) was in the Awaran District. These are shown in Figure 84.

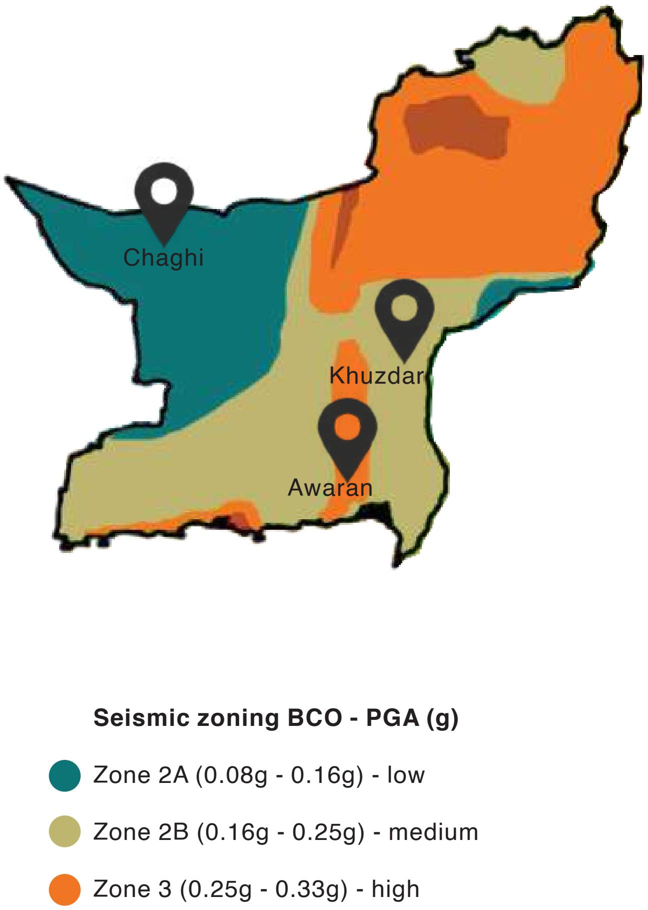

Chaghi
Khuzdar
Awaran
Seismic zoning BCO - PGA (g)
● Zone 2A (0.08g - 0.16g) - low
● Zone 2B (0.16g - 0.25g) - medium
● Zone 3 (0.25g - 0.33g) - high

FIGURE 84: Seismic Zoning map of Balochistan, Pakistan according to Building Code of Pakistan (BCP). Edited by Author. Image Taken from Siddique, M. S., & Schwarz, J. (2015). Elaboration of Multi-Hazard Zoning and Qualitative Risk Maps of Pakistan. Earthquake Spectra, 31(3), 1371–1395. https://doi.org/10.1193/042913EQS114M

The three different levels of PGA were applied to the same datasets. Different conclusions were drawn for the effect of increasing seismicity for different types of datasets - uniform force densities, creases, and randomized force densities.

a) uniform force densities dataset

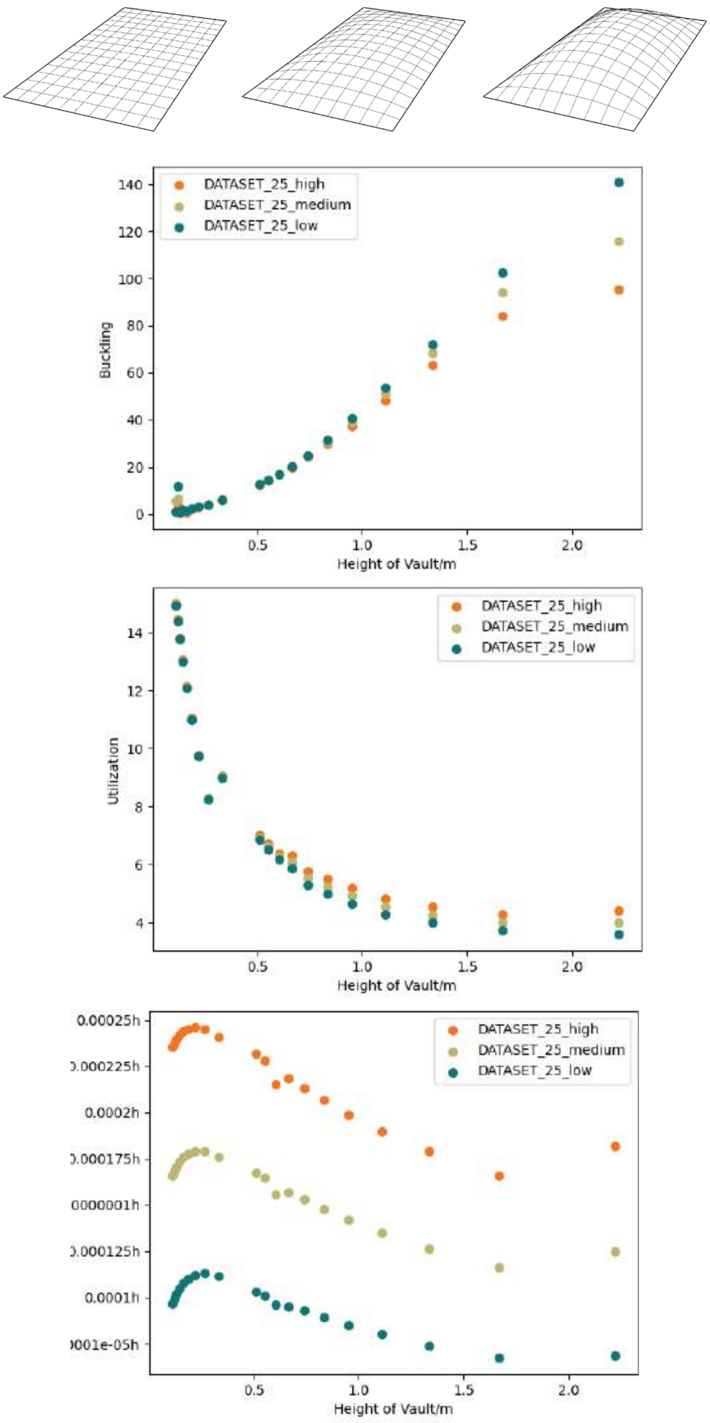


FIGURE 85: Correlation of performance and variation in seismicity: uniform dataset. (Own Work)

For the dataset where only uniform densities were present for each vault, according to the data, there is a large increase in interstorey drift ratios in for higher seismicity. The interstorey drift ratio increased with increasing seismicity by 39.5% for low to medium and 65.2% for medium to high. The reduction in the buckling loading factor and increase in utilization only becomes more significant for taller vaults. The reduction in performance with a decrease in height of the vault is also notable for all three performance metrics caused by an increase in force density.

b) creased dataset

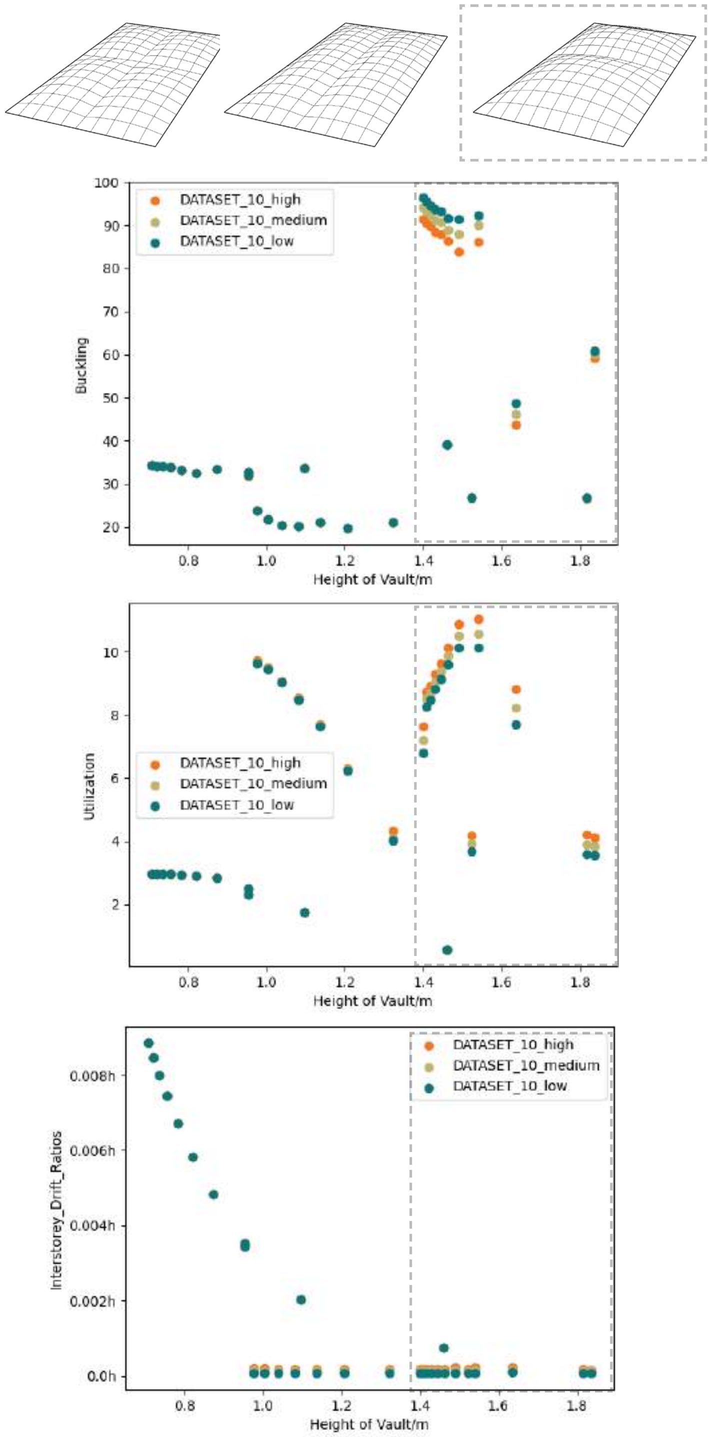


FIGURE 86: Correlation of performance and variation in seismicity: creased dataset. (Own Work)

c) random force densities dataset

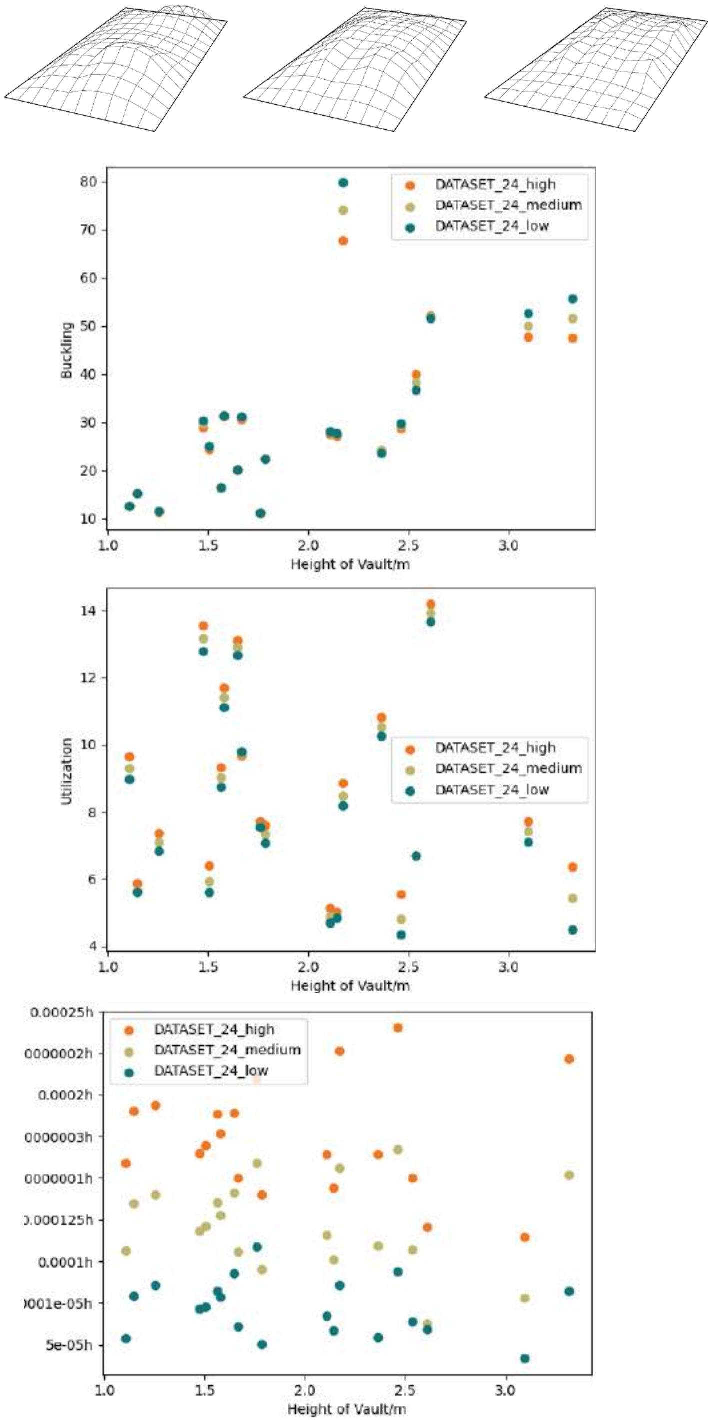


FIGURE 87: Correlation of performance and variation in seismicity: randomized dataset. (Own Work)

For a dataset with creases, a similar trend was present as with the uniform force densities. However, what is notable is the where a larger effect of increasing seismicity is noted in utilization and buckling where the crease was in the direction parallel to the seismic load. The interstorey drift ratios for especially for vaults with perpendicular creases did not get affected by seismic loading.

For the dataset with randomized force densities, similar to the uniform densities dataset, there was only a noticeable increase in the Buckling load factor for taller vaults. It was interesting to note that utilization was reduced uniformly regardless of height (by 3.4% per 1g)- different than the other two datasets. The interstorey drift ratio also increased with increasing seismicity by 68.6% for low to medium and 42.9% for medium to high.

6.2.2 Variation in Force Densities

The effect of increasing force densities was analyzed in the uniform force densities dataset, and the creased dataset to determine how performance was affected. There was no pattern to consider in the randomized dataset because of the nature of its randomness.

a) uniform densities dataset

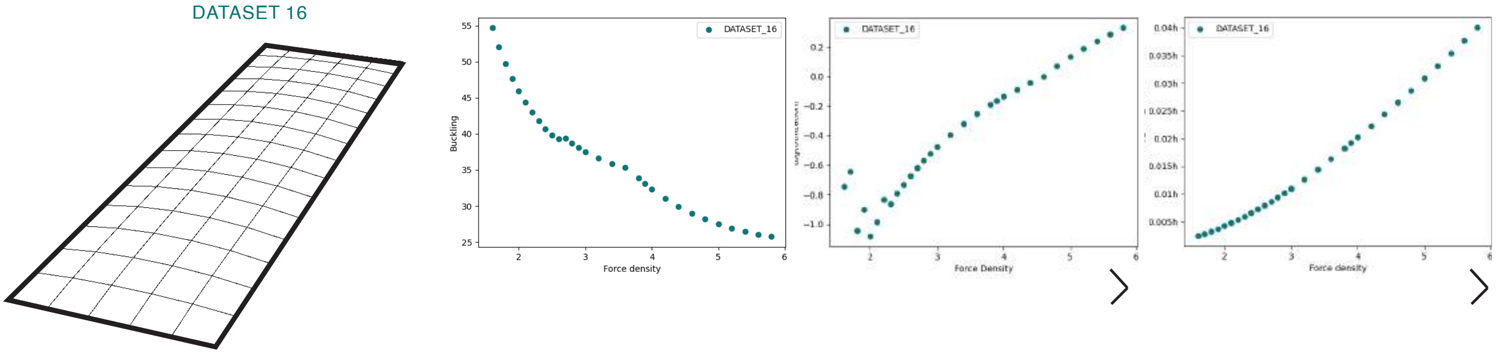


FIGURE 88: Correlation of performance and variation in force densities: uniform dataset. (Own Work)

If each sample has polyedges with all the same force densities (uniform force densities), an increase in force densities causes a reduction in height of the vault. This means that the buckling load factor is reduced, and utilization and displacement increases.

b) creased dataset

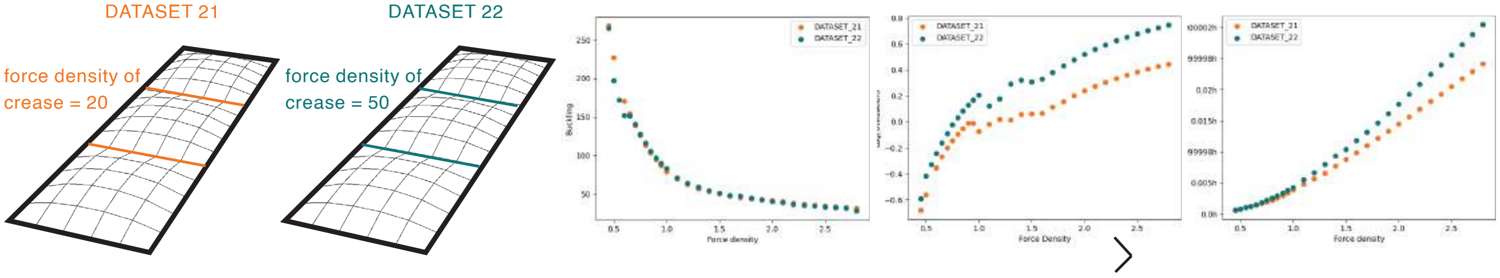


FIGURE 89: Correlation of performance and variation in force densities: creased dataset. (Own Work)

Figure 89 shows that if the force density of the crease polyedge is increased from 20 to 50, there is a 5.5% increase in buckling load factor increasing the stability but also a 29.2% increase in utilization and 11.3% increase in displacement. However, it should be noted that because the rest of the force densities of the dataset were identical, a higher force density in the crease produced shallower vaults.

6.2.3 Variation in Number of Layers of Tiles

As mentioned in the earlier chapters, the dataset was composed of three different thickness for vaults (0.035m, 0.06m, and 0.095m) representing single, double, and triple layers of tiles.

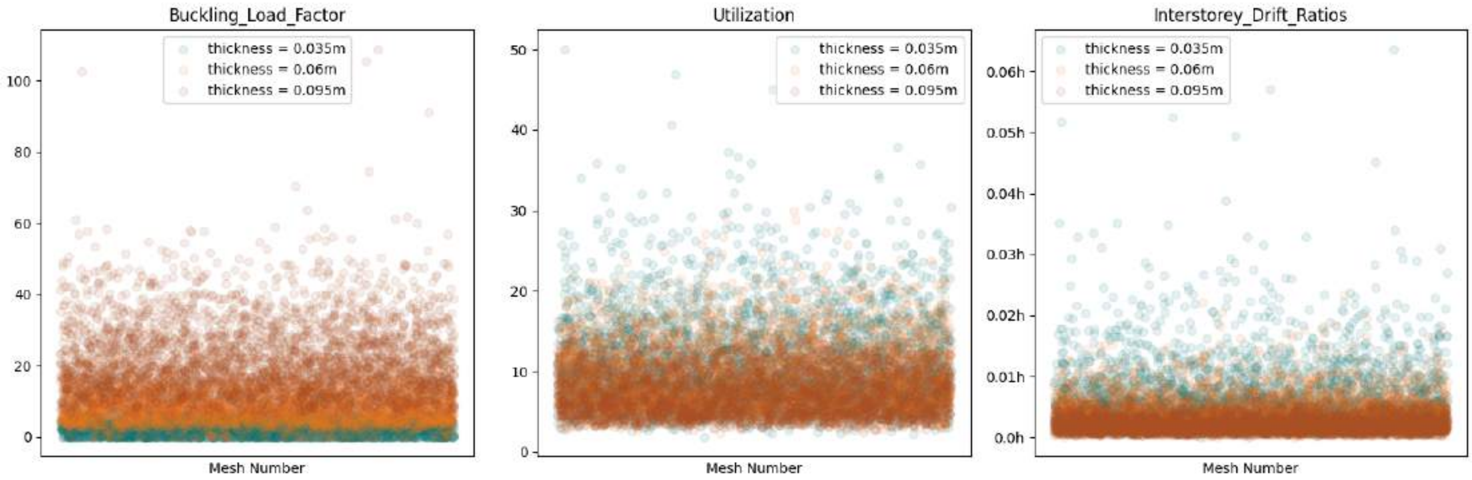


FIGURE 90: Correlation of Performance and variation in thickness. (Own Work)

Figure 90 shows that there is a prominent trend in how the stability, stiffness, and strength increase with thickness for despite randomized force densities.

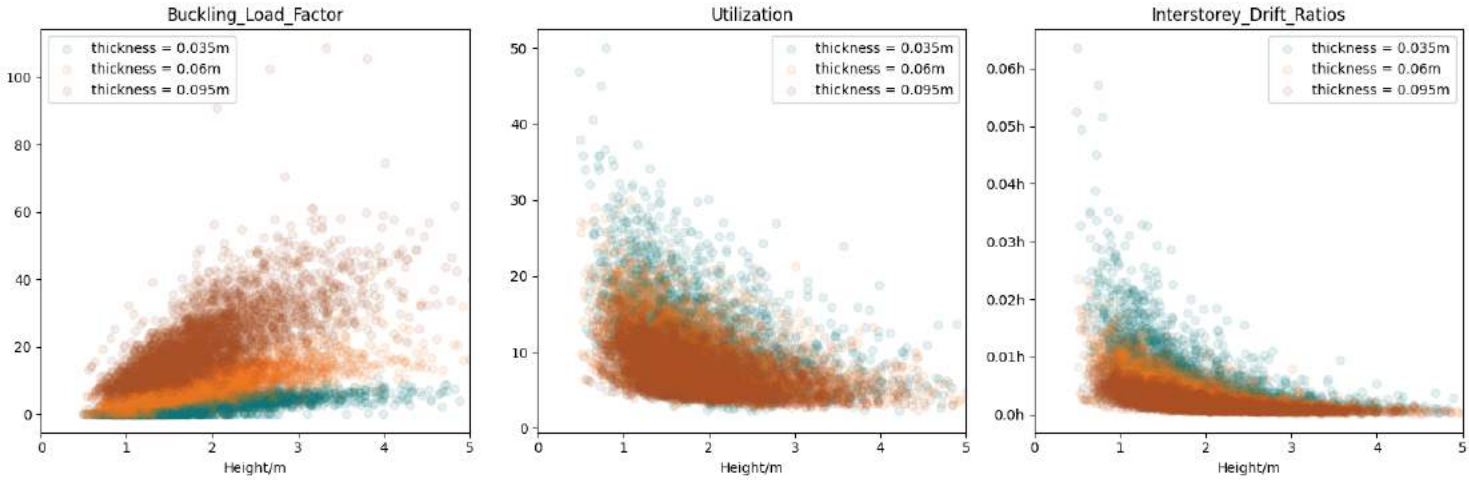


FIGURE 91: Correlation of Performance, thickness, and height. (Own Work)

The correlation of height with performance is an important consideration for performance as mentioned earlier. Therefore, it is important to ascertain how important thickness is for shallower or deeper vaults in terms of performance. As the height increases, a divergent spread between different thicknesses is noted for vaults of the same height. This is shown in Figure 91. Contrary to this, the spread of utilization and interstorey drift converges to a minimum with an increase in height. In conclusion, the effect of thickness on performance in stiffness and strength reduces with the increase in height whereas the effect of thickness on performance in stability increases with height. Thin vaults become stiffer with increasing height comparable to thicker (double/triple layered) vaults but the this is for a 4-5m high vault which cannot be used as a floor slab for a house. The same cannot be said for thin vaults in terms of buckling and strength.

From the results of the seismic analysis, there is a notable difference especially in the buckling load factor when thickness (number of tile layers) is changed. The average reduction/ increase in performance is noted in Table 13 for change in thickness from single layers to 2 layers and 2 layers to 3 layers across the entire 10,000 samples. The buckling load factor increases with a 213.4% increase from a transition from a single to a double layer and a 59.1% increase further to a triple layer vault. The strength also increases with a 16.3% and 26.6% reduction in utilization respectively for the same change in thickness. The stiffness also increases with a 38.5% and 62.3% reduction as well.

DATASET: randomized		
Performance metric	Number of layers of tiles	
	single layer to double layer tiles	double layer tiles to triple layer tiles
Buckling_Load_Factor	213.4% increase	59.1% increase
Utilization	16.3% reduction	26.6% reduction
Interstorey_Drift_Ratios	38.5% reduction	62.3% reduction

TABLE 13: Performance change with number of layer of tiles/ thickness. (Own Work)

6.2.4 Variation in Support Conditions

To determine appropriate support conditions for the vault, simulations were carried out for fixed supports and pinned supports separately.

Figures 92, 93, and 94 show the performance results for different datasets. Table 14 shows the percentage difference in performance for each dataset between the supporting conditions for the 20 samples.

Percentage difference in support conditions			
Performance metric	DATASET: uniform force densities	DATASET: creases	DATASET: randomized force densites
Buckling Load Factor	26.0%	53.4%	27.2%
Utilization	26.2%	6.0%	2.8%
Interstorey Drift Ratios	90.7%	22.6%	9.1%

TABLE 14: Percentage difference in support condition. (Own Work)

a) uniform force densities dataset

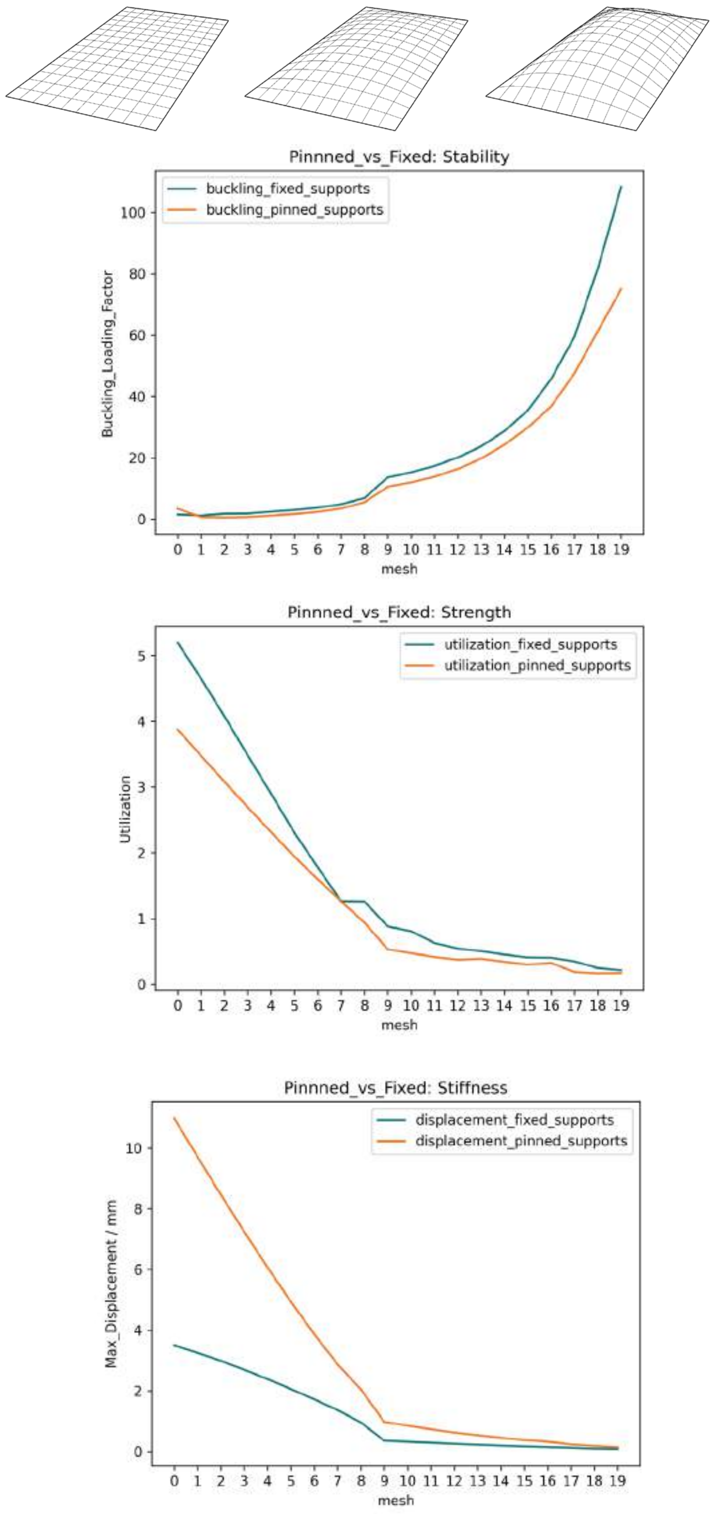


FIGURE 92: Pinned vs Fixed supports: uniform dataset. (Own Work)

For the uniform force densities dataset, there was better performance against buckling, in utilization, and in stiffness for fixed supports as indicated in the graph. As the uniform force density reduces, the height of the vault increases and the difference in performance in utilization and displacement for fixed and pinned supports reduces. The opposite happens in buckling; reduction in force densities increases the buckling load factor.

b) creased dataset

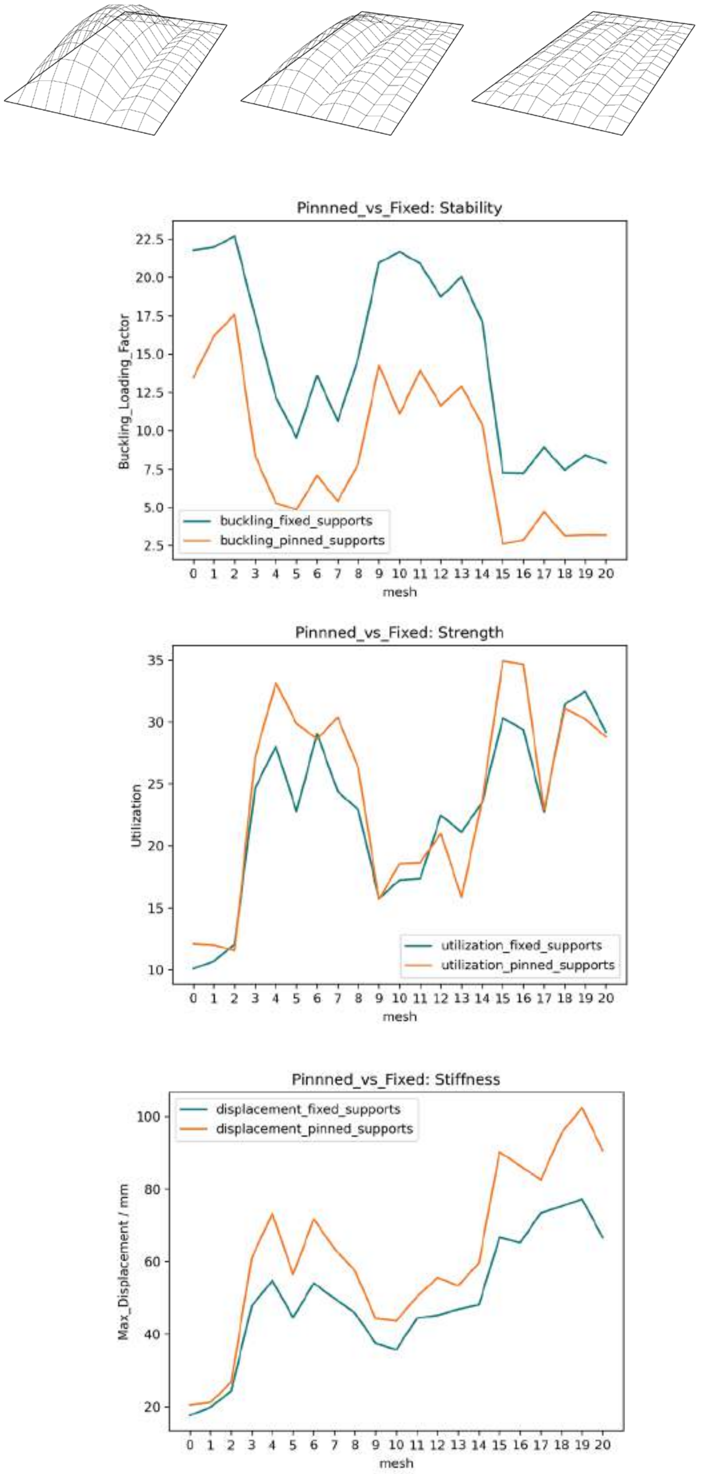


FIGURE 93: Pinned vs Fixed supports: creased dataset. (Own Work)

For different creases, the performance with fixed supports was better than that with pinned supports in terms of buckling and displacement. However, in some vaults, there was better performance in utilization for pinned supports.

c) random force densities dataset

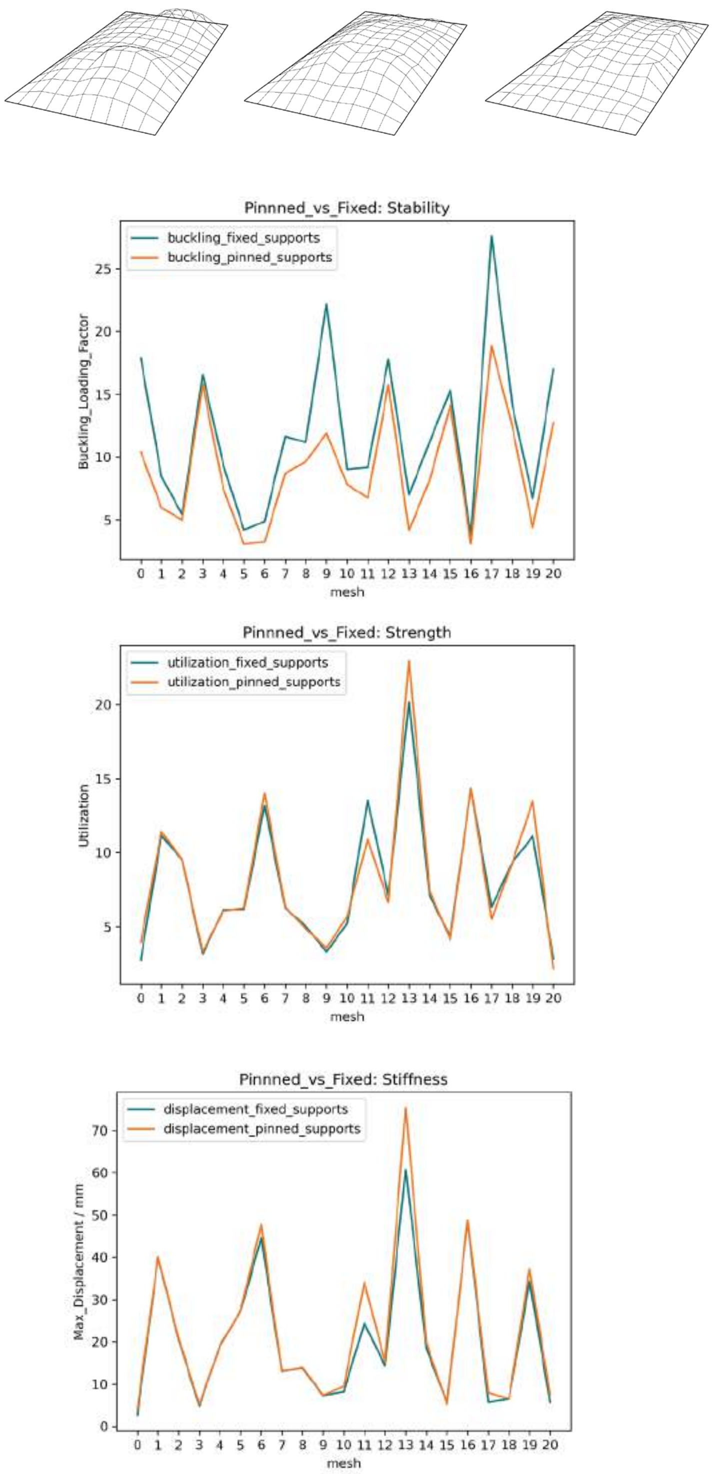


FIGURE 94: Pinned vs Fixed supports: random dataset. (Own Work)

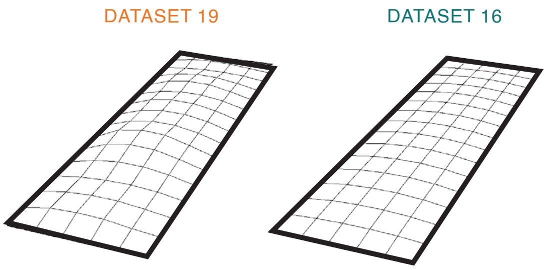
For randomized force densities, there was no clear pattern in utilization. There was better performance for fixed supports overall in buckling. Some samples performed better in displacement with fixed supports.

In conclusion, the fixed support was chosen for overall better performance for the main dataset as it also composed of meshes with uniform force densities.

6.2.5 Variation in Sample Datasets

To find the which dataset strategy would give comparably better performing solutions, different sample datasets were compared made from different input strategies.

a) random force densities vs uniform force densities



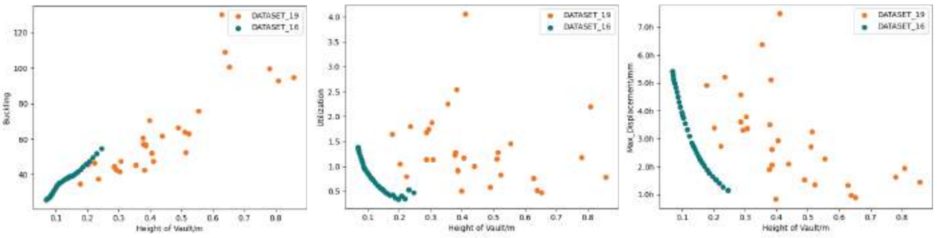
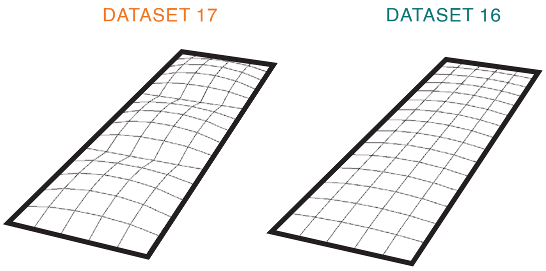


FIGURE 95: random force densities vs uniform force densities. (Own Work)

For the same height, the performance of the uniform force densities dataset under buckling, utilization, and displacement was better than the randomized dataset. However, it should be noted that this was just 30 samples.

b) creased vaults (into thirds of length) vs uniform force densities



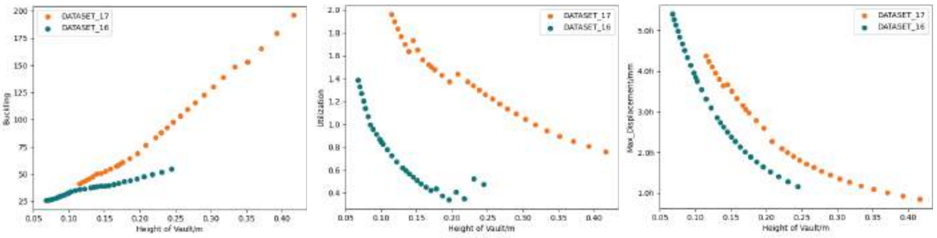
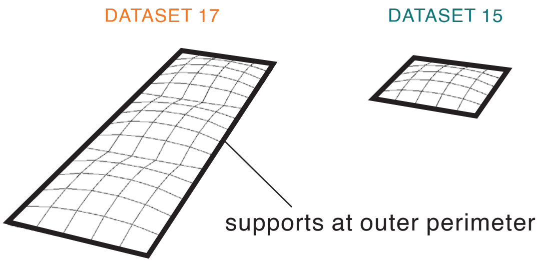


FIGURE 96: creased vaults (into thirds of length) vs uniform force densities. (Own Work)

The performance of the uniform force densities was also better in all three performance metrics than the creased dataset. As the force densities of the creases became closer to the rest of the vault, the performance increased - so sharp creases performed qualitatively worse than smoother creases. This can be seen in Figure 96.

c) creased vaults (into thirds of length) vs small vaults (third of original length)



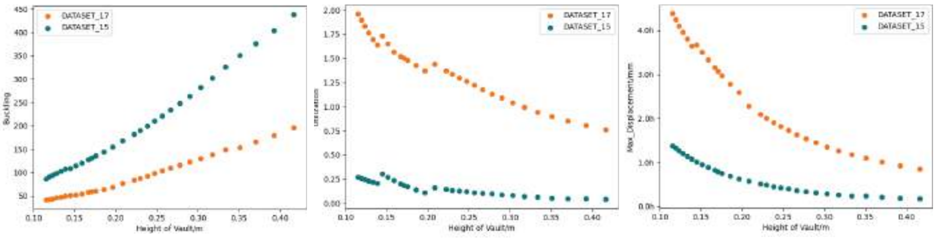


FIGURE 97: creased vaults (into thirds of length) vs small vaults (third of original length). (Own Work)

The creased dataset had creases at every third along the longer dimension of the vault. This was compared to a smaller vault measuring the third of the longer dimension. It was found that for the same height, the performance of the smaller vault was significantly better than the creased vault. It should be noted that no supports were present at the creases. There were only supports at the outer perimeter.

d) segmented vaults (third of original length) supported separately vs small vaults (third of original length)

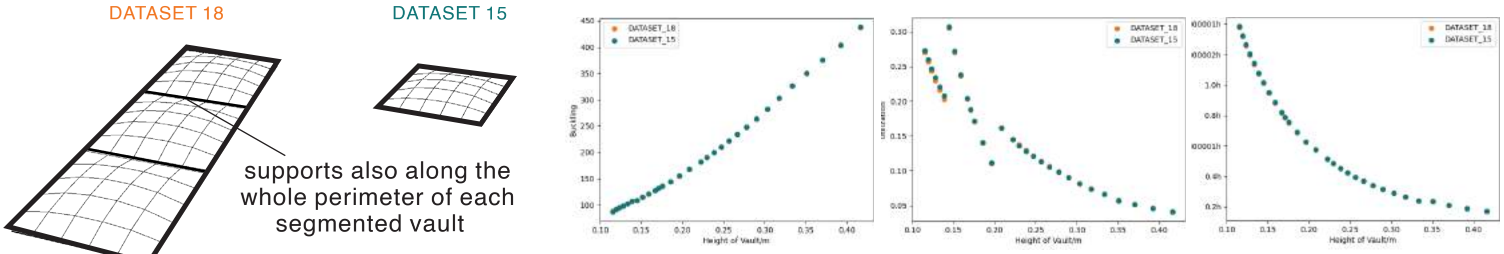


FIGURE 98: segmented vaults (third of original length) supported separately vs small vaults (third of original length). (Own Work)

In theory, the smaller vault should behave the same as a vault segmented in the same dimension if each of the segments had with fixed supports. The hypothesis was correct which meant that performance could be increased by segmentation.

e) segmented vaults (third of original length) supported separately vs uniform force densities

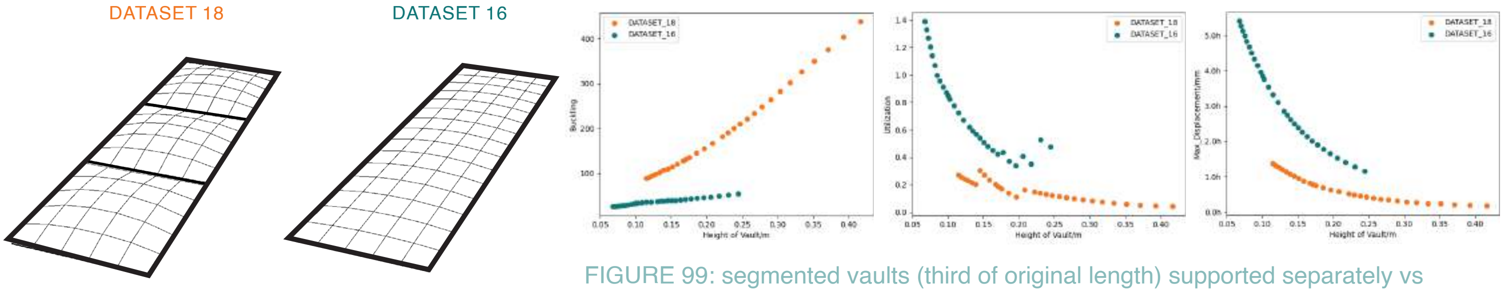


FIGURE 99: segmented vaults (third of original length) supported separately vs uniform force densities. (Own Work)

The uniform force densities dataset had performed better than all other previous datasets. However, it was noted that the performance of the segmented vault was significantly better than that of uniform force densities.

Conclusion

In conclusion, the best performing strategy was found to be segmentation of the vault into smaller vaults that are supported along each of their own perimeter. Each segment behaves like a uniform force densities vault with uniform stress distribution avoiding build-up of stress concentrations. It can be seen in Figure 100 how introduction of creases introduces areas of stress concentrations subsequently increasing risk of failure especially forming high tensile zones the bottom of the vault. These would induce cracking. This also happens in the randomized force densities where variation in force densities causes unequal load distribution causing stress concentrations.

Even though they are the best performing strategy for seismic optimization of vaults, segmented meshes were not used as they would require a different data-structure altogether as the support conditions were not taken into consideration by the current data-structure of a nested list of force densities. Moreover, it was not clear whether a large enough dataset for the VAE could be generated from just this strategy alone. The final dataset was made from randomized force densities to remove user bias and have a large variation is generated data for novel samples. 50 meshes of uniform force densities were also included in the dataset as this strategy gave the second best performances (for the limited sample sizes that were tested).

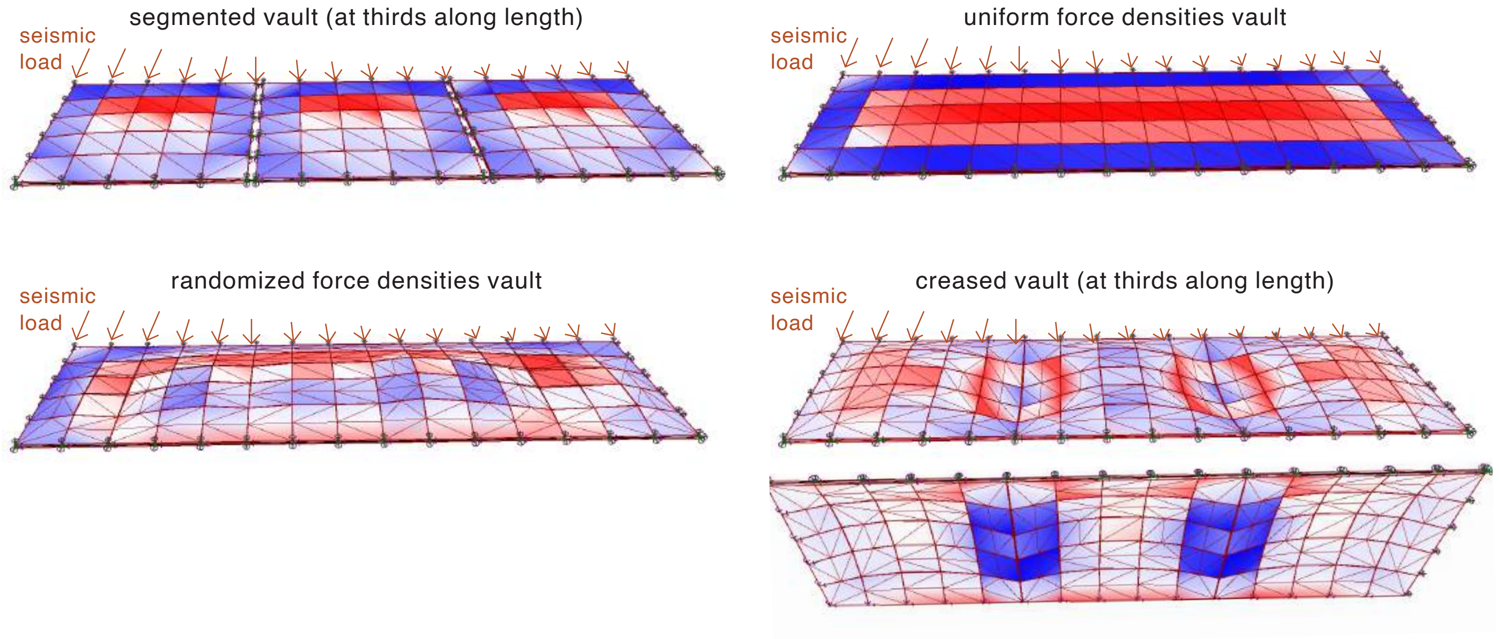

segmented vault (at thirds along length)
seismic load
uniform force densities vault
seismic load
randomized force densities vault
seismic load
creased vault (at thirds along length)
seismic load

FIGURE 100: Stresses on different types of vaults. (Own Work)

6.2.6 Best Performing Samples in the Main Dataset

The main dataset composed of 9950 samples of randomly generated force densities and 50 samples of uniform force densities. The 50 best performing samples were chosen from each performance metric and plotted.

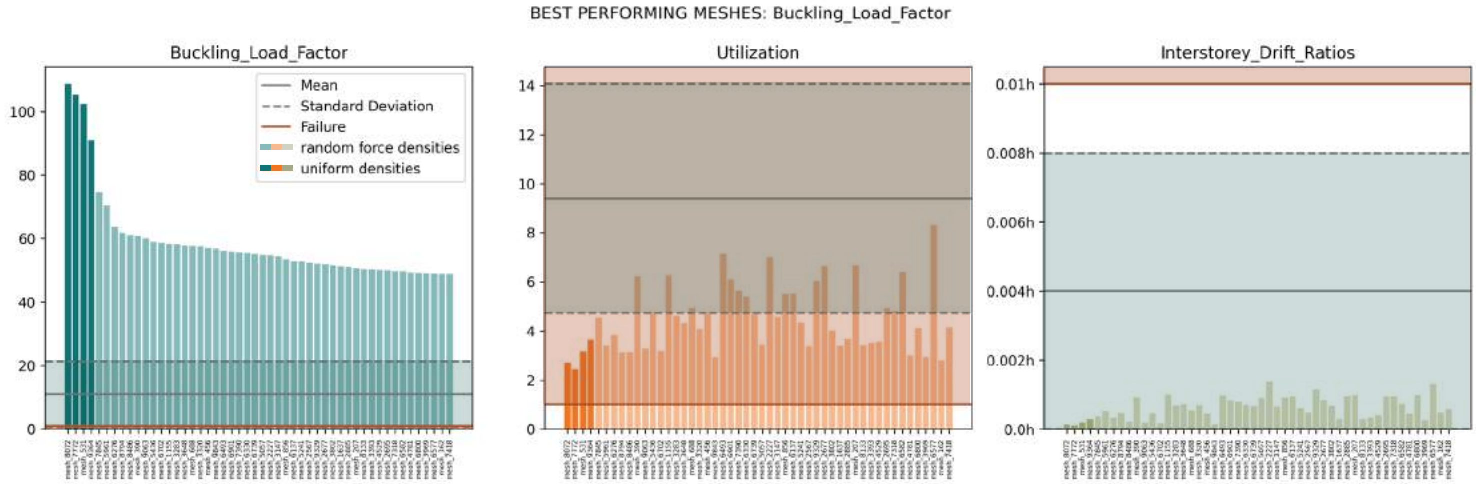


FIGURE 101: Top 50 best performing meshes in Buckling Load Factor from the main (randomized) dataset. (Own Work)

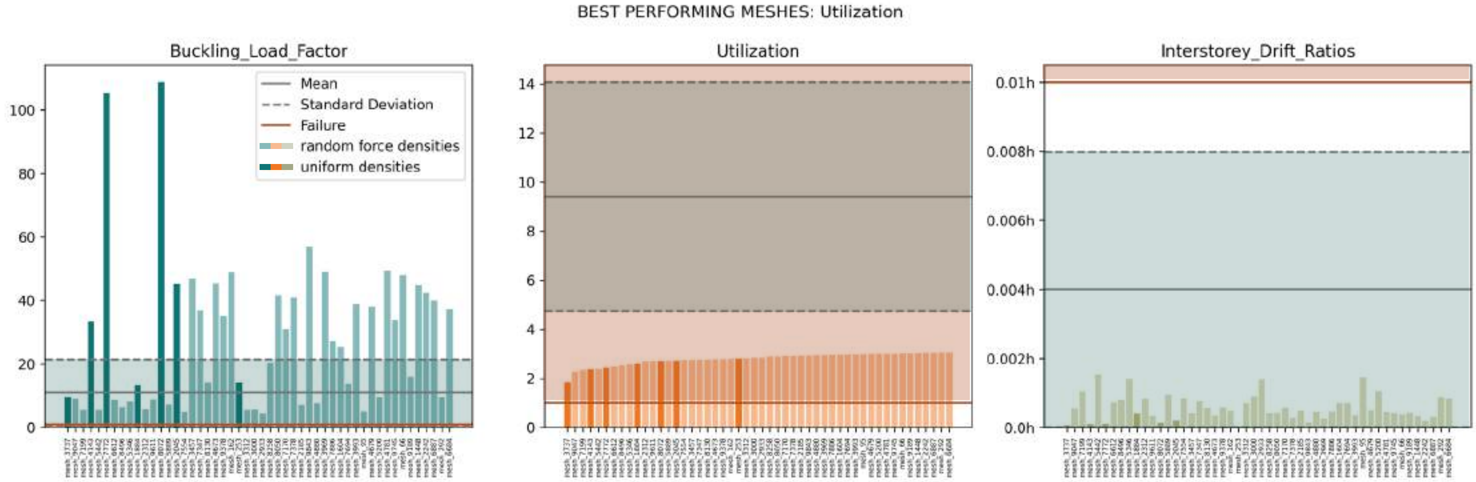


FIGURE 102: Top 50 best performing meshes in Utilization from the main (randomized) dataset. (Own Work)

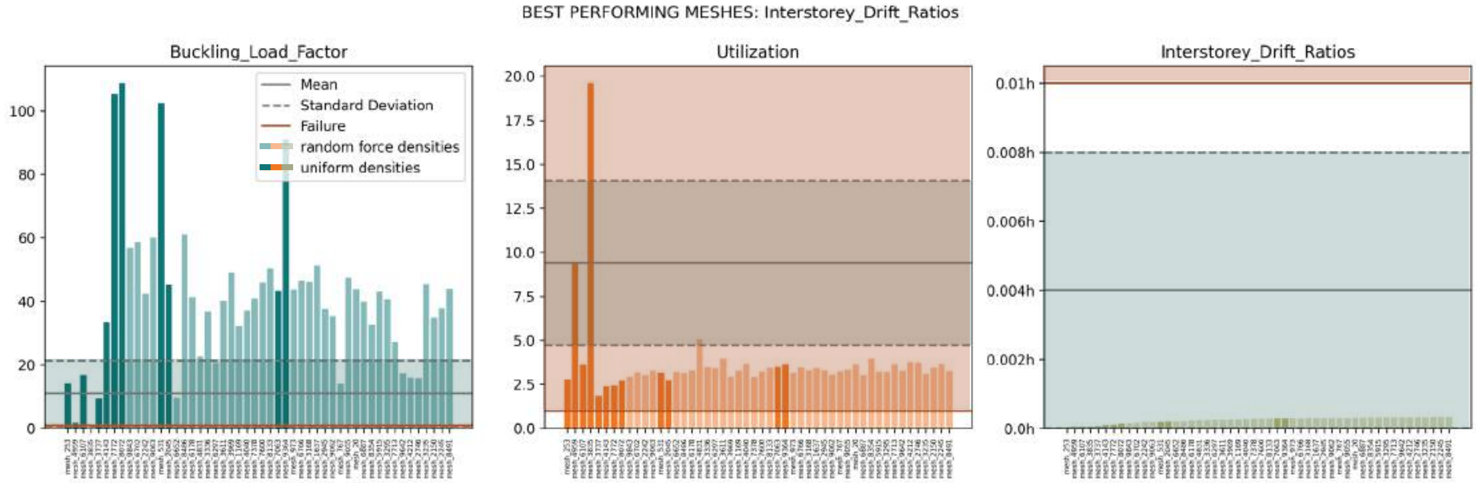


FIGURE 103: Top 50 best performing meshes in Interstorey Drift Ratios from the main (randomized) dataset. (Own Work)

From Figures 100, 101, and 102, we can conclude the following:

- Stability: For the best performing samples, there is no failure in buckling except for one sample that performs the 4th best under interstorey drift (mesh_3835).
- Strength: For all meshes, even the best performing ones in utilization, it can be seen that all have failure in utilization. This can be attributed due to failure due to tensile stresses resulting in cracking.
- Stiffness: There is no failure in interstorey drifts for any mesh within 1 standard deviation from the mean.

It can be concluded that primarily failure will happen in Utilization first. All samples failed in Utilization.

6.2.7 Performance of Uniform Force Densities in the main Dataset

From the comparative results of samples from randomized force densities against those generated from uniform force densities at the initial stage, it seemed likely that the highest of the best performing meshes would be those of uniform force densities. It was found the number of uniform meshes that were also one the 50 best performing meshes out of the main dataset of 10,000 samples in buckling, utilization, and interstorey drift ratios were 4,7, and 12 respectively. However, it should be noted that another parameter that governed performance was the thickness of the vault which was also randomly assigned to each sample. Therefore, a fair estimate would be to check whether a thick mesh with uniform force densities failed to make it to the top 50 best meshes. For this reason Figure 104, has been plotted.

For each performance metric, we can see the force densities of the uniform meshes.

Meshes with maximum thickness (95mm) were found to be perform best in buckling - as confirmed by the analysis earlier on correlating significant increase in buckling load factor with increased thickness. These belong to meshes with force densities between 0.7 to 1.3 - lower thickness meshes in between with FD (force density) =0.9, 0.11,0.12 did not perform as well. This is not the case with utilization where high thickness meshes like FD=1.0 (95mm), and FD=1.1 (60mm) performed poorer in utilization than FD=1.2 with lower thickness (35mm). There was range of force densities that performed well under buckling (FD = 0.7-1.3), as well as utilization (FD = 0.3-1.2). However, all meshes except one (with 35mm low thickness) performed well for interstorey drifts below FD = 1.3

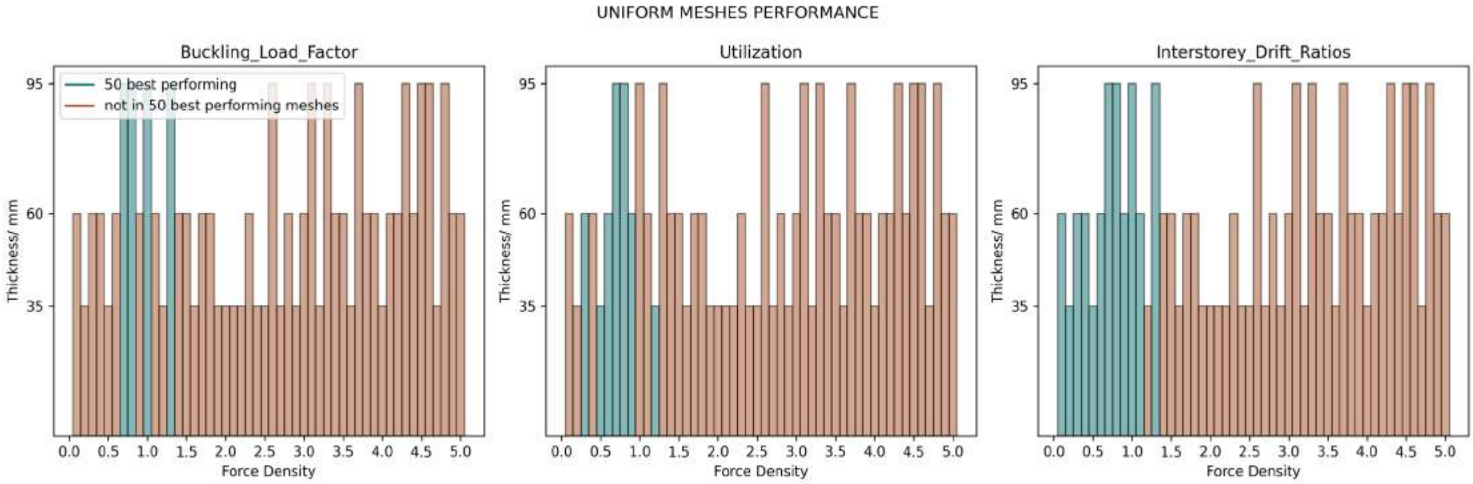


FIGURE 104: Relationship of the force densities and thickness of Uniform meshes and whether they are in the top 50 best meshes for each category. (Own Work)

07/ RESULTS: GENERATOR

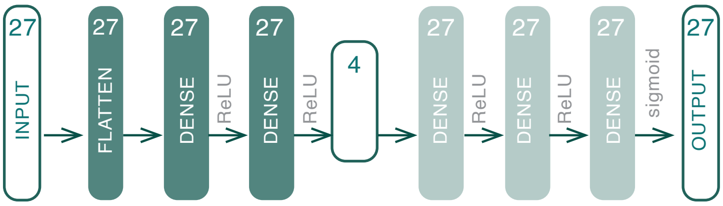
See Section 4.1.3 for the overall workflow of the overall Generator which includes the VAE, Surrogate model and Gradient Descent optimizer. All the modeling for this chapter was done in Python using the Tensorflow library.

7.1 VARIATIONAL AUTOENCODER

See Section 4.1.4 for the workflow of the VAE.

7.1.1 Latent Distribution of the Creased Dataset vs Random Dataset

Two different types of datasets were used. Some notable differences were found in the distribution. Although the same set of hyperparameters are unique to a particular dataset, the set of hyperparameters and the architecture used for comparison will be kept constant to reduce the number of variables for the comparison for the sensitivity analysis.



HYPERPARAMETERS: latent_dimension = 4, **beta** = 0.2, **epochs** = 600, **batch_size** = 64, **learning_rate** = 1E-03

FIGURE 105: VAE architecture + hyperparameters - creased vs random dataset. (Own Work)

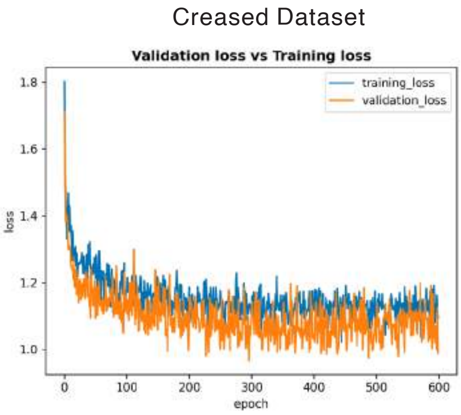


FIGURE 106: Training loss vs Validation Loss - creased dataset. (Own Work)

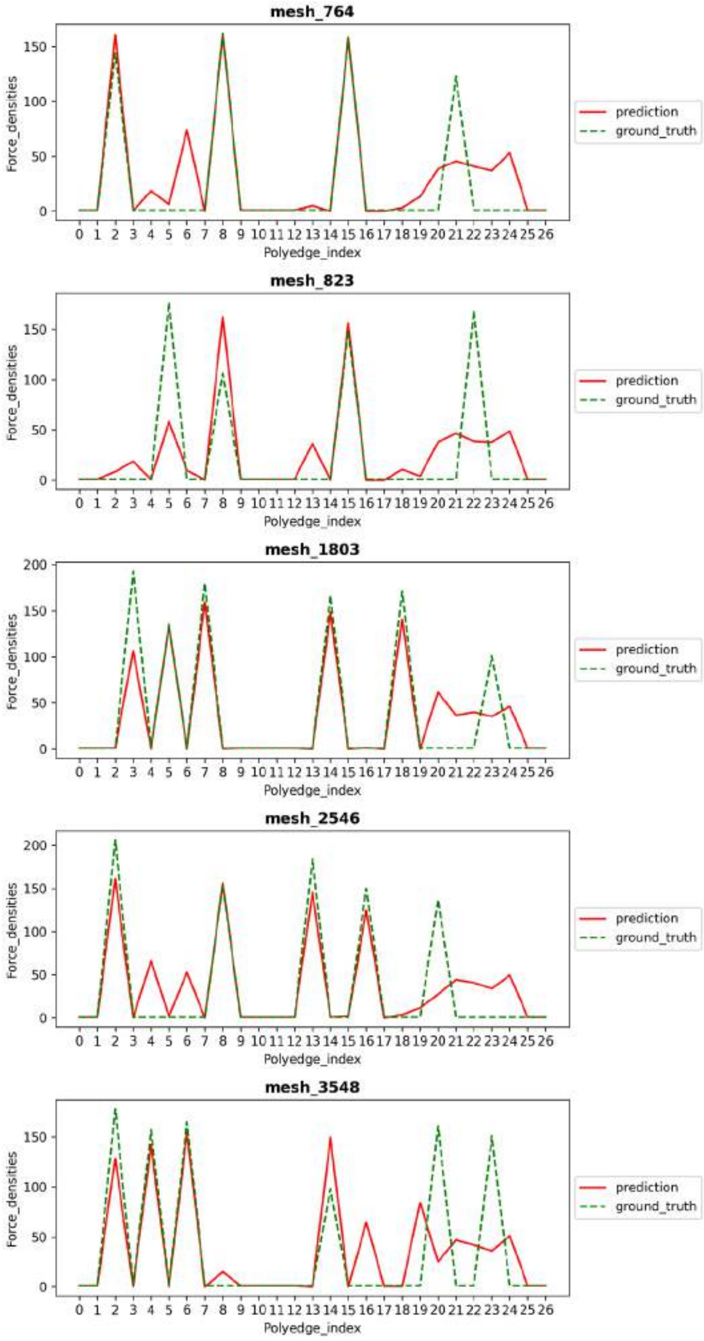


FIGURE 107: Predictions vs Ground Truth - creased dataset. (Own Work)

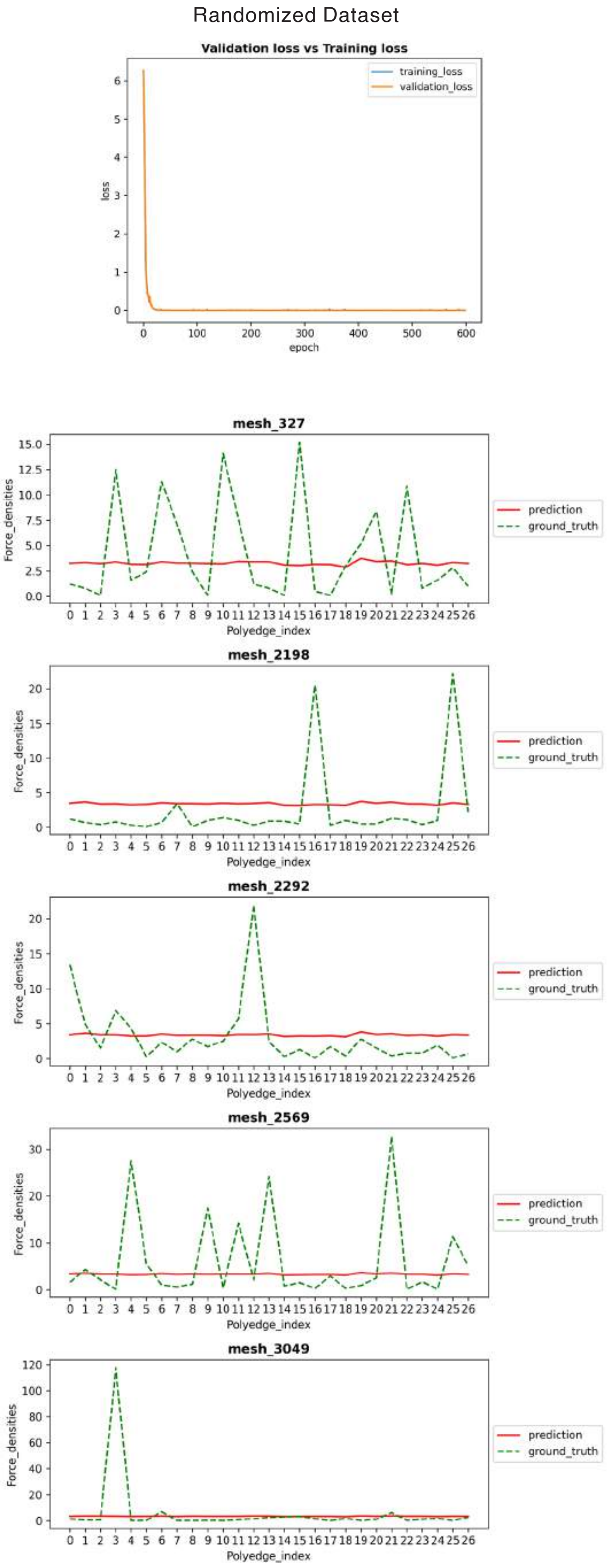


FIGURE 108: Training loss vs Validation Loss (top). Predictions vs Ground Truth (bottom) - randomized dataset. (Own Work)

Figure 107 and Figure 108 show in how each case the test data is predicted. Each graph represents one mesh with its 27 force densities. In each case, the predictions differ significantly. For the creased dataset, the VAE seems to make good predictions based on the closeness of the red and green lines. This may be due to the fact that is able to recognize the distribution of the dataset and generate good predictions based on that dataset. However, for the randomized dataset, this is not the case, because even though samples from the ground truth contain highly variable force densities, the predictions are mostly uniform along a mean. This may be because the creased dataset was made from samples generated using a pattern as discussed in the earlier sections so it is able to detect the distribution.

However, in the randomized dataset, for every mesh, as each force density from the 27 was generated randomly from a logarithmic distribution of 10,000 values, there isn't a pattern to be captured in the randomness makes up the dataset.

In the workflow of this project, the purpose of the VAE, however, involves just producing novel samples rather than predicting the test samples - and the distribution of the latent space represents how well novel samples may perform. As illustrated earlier in Section 6.2.5, large variations in force densities are likely to produce poor performing meshes. A latent space that samples more uniform force densities like the one from the randomized dataset has a greater likelihood to produce better performing meshes.

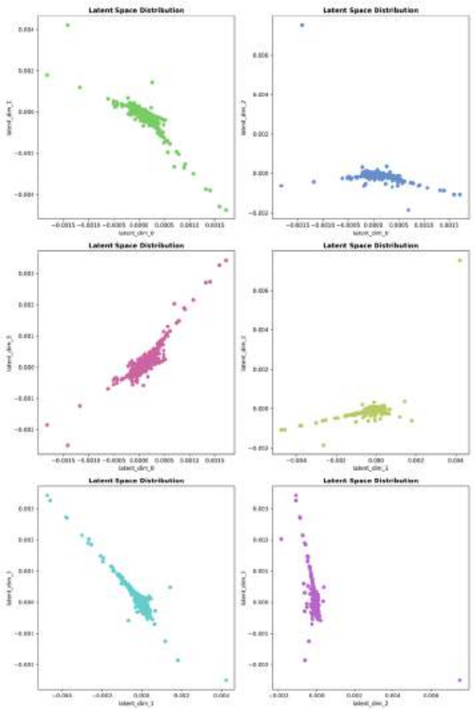


FIGURE 109: Latent space representation - creased dataset. (Own Work)

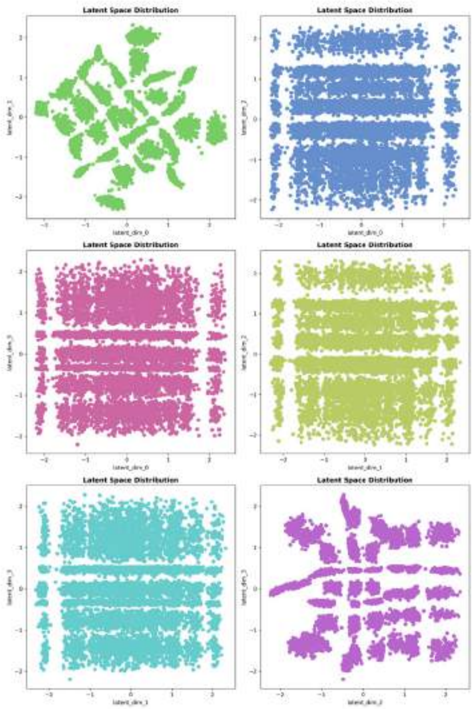


FIGURE 110: Latent space representation - randomized dataset. (Own Work)

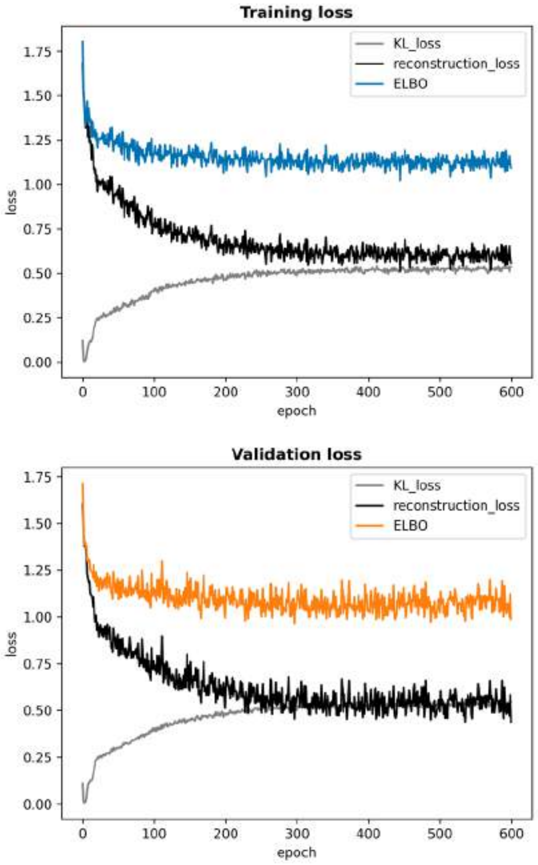


FIGURE 111: KL loss and Reconstruction Loss of Training Loss (top) and Validation Loss (bottom) - creased dataset. (Own Work)

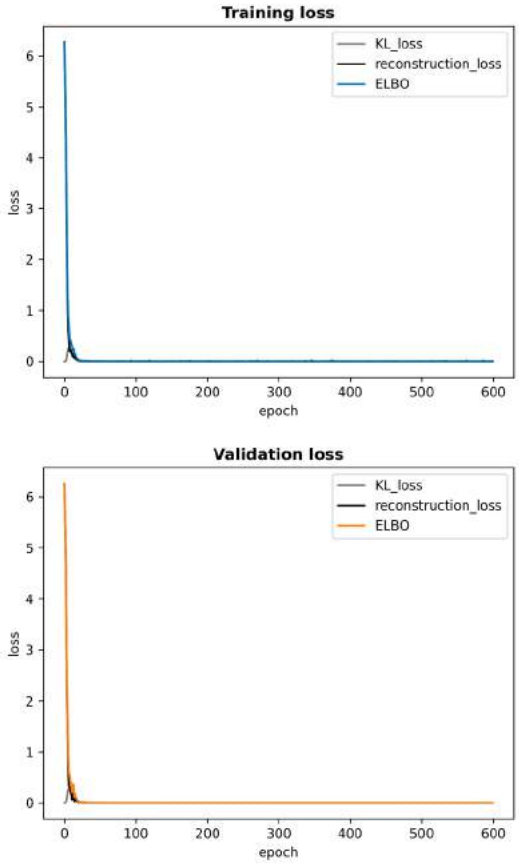


FIGURE 112: KL loss and Reconstruction Loss of Training Loss (top) and Validation Loss (bottom) - randomized dataset. (Own Work)

The number of neurons in each layer were increased (from 27 to 32, then 64) to note changes in the distribution of the randomized dataset but it showed similar results. The hyperparameters were kept the same. The same trend follows other changes in architecture (using a single dense layer each in the encoder and decoder) and change in hyperparameters as well like varying batch sizes, learning rates, beta, number of latent dimensions.

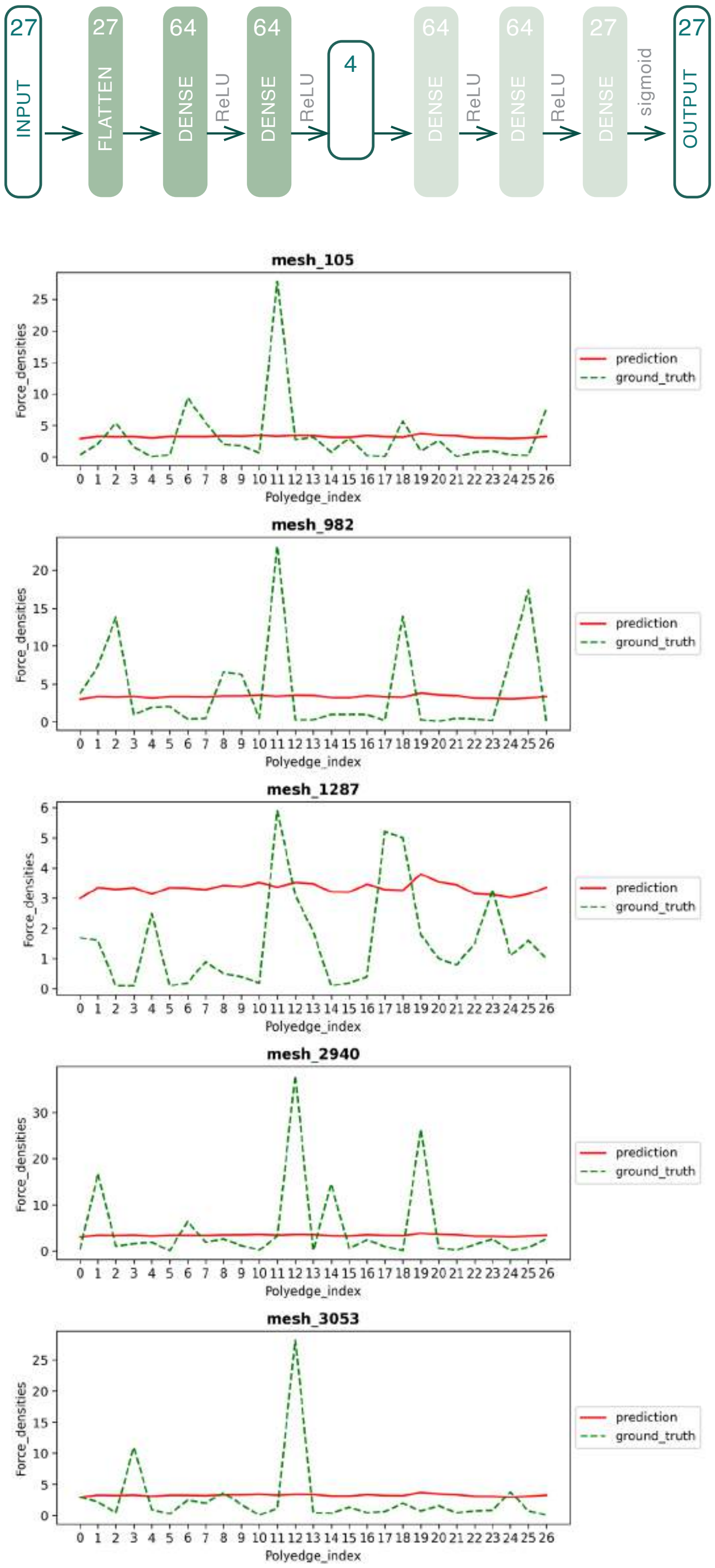


FIGURE 113: VAE of architecture 27-64-64-4-64-64-27 (top) and its ground truth vs predictions (bottom). (Own Work)

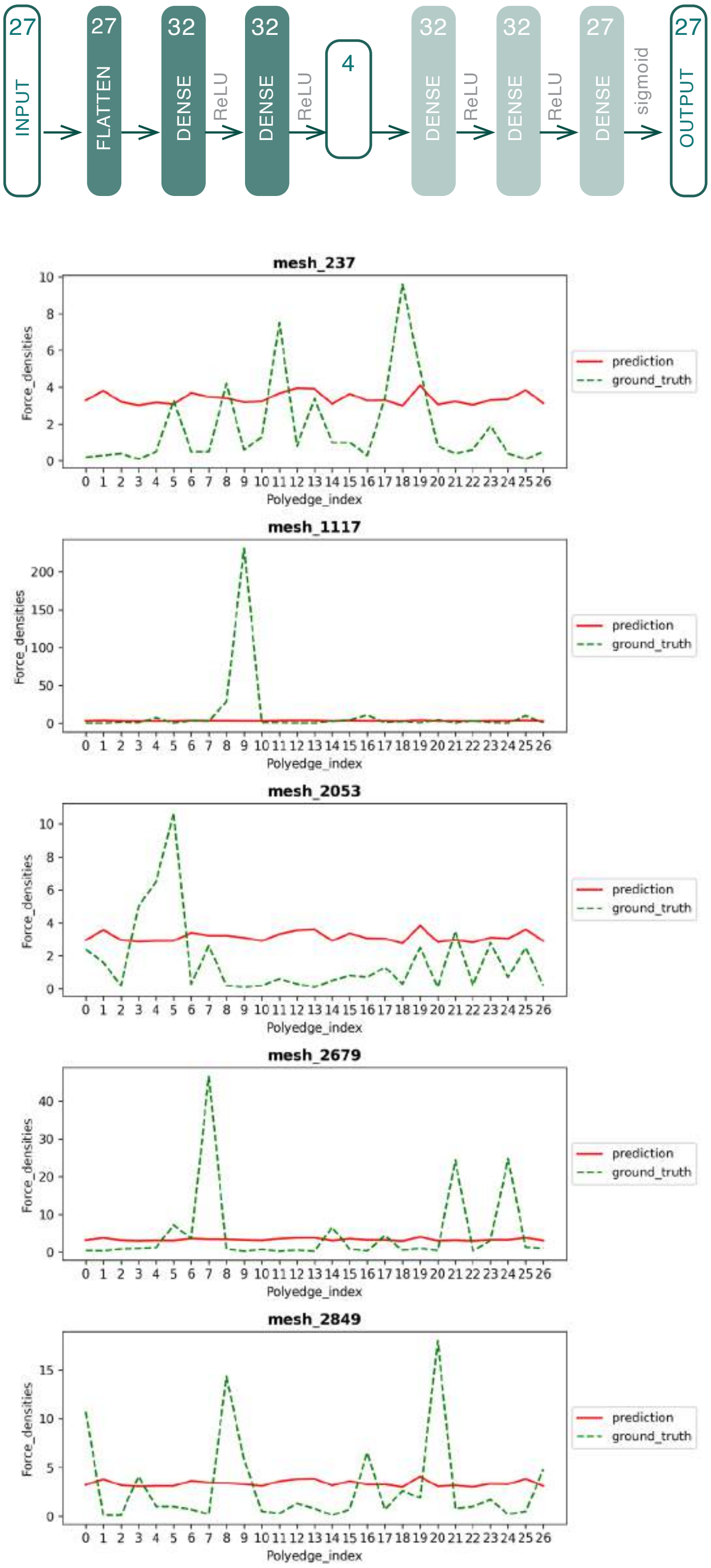


FIGURE 114: VAE of architecture 27-32-32-4-32-32-27 (top) and its ground truth vs predictions (bottom). (Own Work)

7.1.2 Inclusion of the Thickness

The VAEs that had been shown until now only used force densities as the features for training. However, as thicknesses were also varied they were to be included as well. They were included as features. The figure below represents how this is represented in the randomized dataset.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	
[	0.7	9.3	0.1	9.3	0.1	0.6	0.1	0.6	1.1	7.3	6.3	0.4	2.6	0.1	3.5	3.4	3.6	0.3	0.3	1.2	2.1	0.1	2.2	1.7	0.9	0.1	0.1	0.095	]
[	0.6	7.6	3.7	3.9	0.5	0.1	1.1	0.7	1.2	0.6	0.8	1.4	0.1	1.7	0.4	1.0	0.9	0.4	0.5	11.4	1.3	1.8	0.8	11.0	0.1	13.8	10.6	0.060	]
[	5.5	3.4	0.2	0.2	26.8	7.6	0.2	309.7	4.9	0.4	0.6	0.1	1.1	10.2	6.7	2.6	7.3	0.6	0.4	0.6	0.2	1.0	2.0	7.0	3.4	0.3	0.1	0.095	]
[	0.1	0.1	0.6	1.4	2.5	0.9	0.2	1.9	1.0	19.7	0.5	1.1	0.5	1.5	0.4	0.3	1.2	7.2	0.1	6.5	0.3	0.9	0.4	3.8	10.5	1.0	1.5	0.035	]
[	1.9	0.5	6.6	0.5	5.0	0.7	0.3	0.7	18.1	0.5	3.1	0.3	1.0	6.9	1.1	0.2	0.3	0.2	0.8	10.6	0.1	2.3	1.9	0.2	11.0	10.2	0.9	0.095	]
[	3.3	1.0	20.9	1.6	0.1	2.5	0.2	0.3	0.1	7.2	6.0	0.4	0.3	11.1	8.4	0.9	0.5	16.0	2.2	1.4	1.5	15.7	3.1	0.6	0.8	0.1	0.1	0.035	]
[	1.4	1.1	0.8	0.3	0.1	0.2	4.6	3.1	1.8	1.1	0.4	0.2	1.0	4.4	4.0	0.7	1.0	1.0	0.6	0.8	0.2	16.8	0.2	0.3	2.0	0.1	0.4	0.095	]
[	2.9	0.6	1.6	1.0	0.1	0.6	1.9	0.1	1.2	0.1	1.3	0.2	0.4	0.3	0.2	0.8	0.6	0.5	2.6	1.1	2.9	0.6	24.4	1.8	1.6	0.1	0.7	0.060	]

FIGURE 115: thickness inclusion in feature dataset. (Own Work)

The data-structure of the first 8 meshes are shown in the figure. Each mesh is represented a single row consisting of 27 numbers equal to the number of force densities per polyedge in that mesh. The 28th value represents the thickness of the mesh. It can be either a single layer of tile (0.035m), two layers (0.060m), or three layers (0.095m).

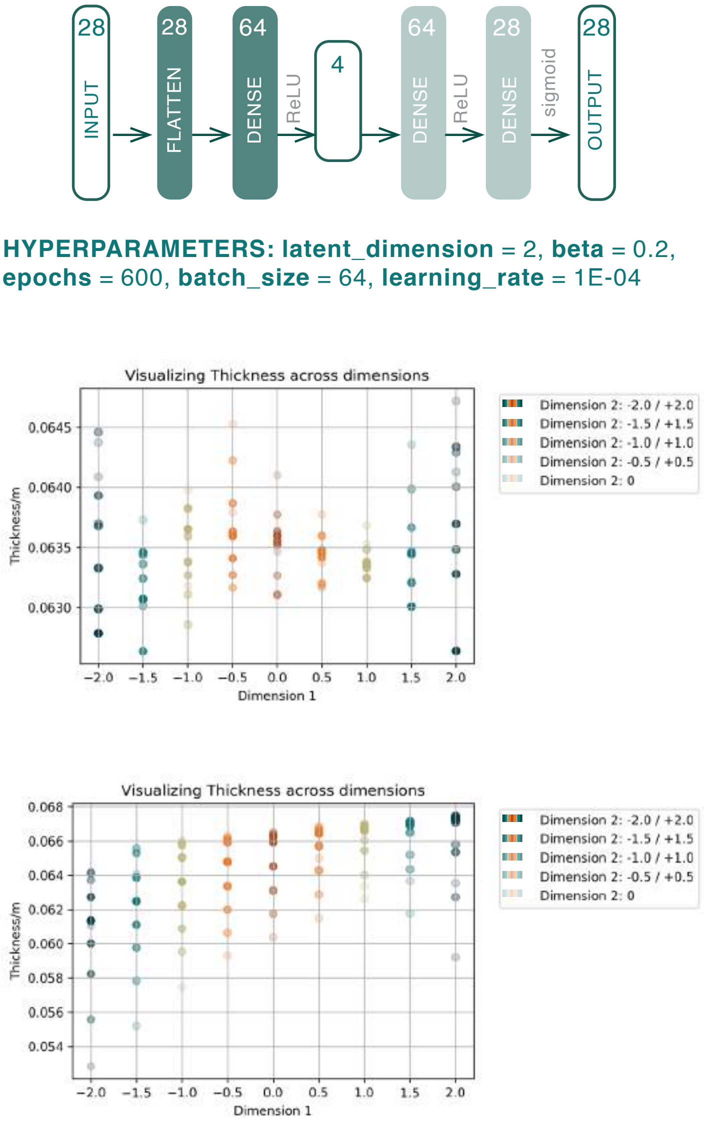
It was important to consider how the thickness would be normalized in relation to the rest of the dataset. Two approaches were carried out whilst keeping the architecture and the hyperparameters constant:

- Normalization with the rest of the force densities
- Normalization independent of the force densities.

When thickness was normalized along with the rest of the features, it gave a wider distribution along the mean of the thickness dataset (μ=0.06349m). However, if it was normalized independent of the force densities, the distribution along the mean was narrower.

The effect of the normalization technique considered is more apparent on the output of the surrogate model when thickness is input as a feature. This will be discussed in the next section on the Surrogate Model.

It is notable to mention that the inclusion of thickness as a label instead of a feature was also considered but not carried out. This would be the subject of a further study into Conditional Variational Autoencoders.



HYPERPARAMETERS: latent_dimension = 2, **beta** = 0.2, **epochs** = 600, **batch_size** = 64, **learning_rate** = 1E-04

FIGURE 116: Architecture of VAE (top). Visualization of thickness across dimensions of the latent space - independent normalization (middle) thickness normalization along with force densities (bottom). (Own Work)

7.1.3 Sampling across the Latent Dimensions

Some observations were noted in sampling while changing the architecture of the VAE. The randomness of the latent space distribution is highly dependent upon the activation functions used - ReLU and sigmoid.

Only ReLU

When only Rectified Linear Unit (ReLU) was used as the activation function, mostly zero values were produced by the decoder as shown below. This was the case in variations of other architectures and hyperparameters using ReLU as the only activation function. Figure 117a shows 50 samples that were predicted from unseen data. All 50 samples produced zero values for force densities while 10 of those only produced non-zero values. Figure 117b shows 5 new samples generated from the latent space. Peak values were sampled up to a force density of 60. After several training models, as shown in Table 15, using ReLU alone was abandoned.

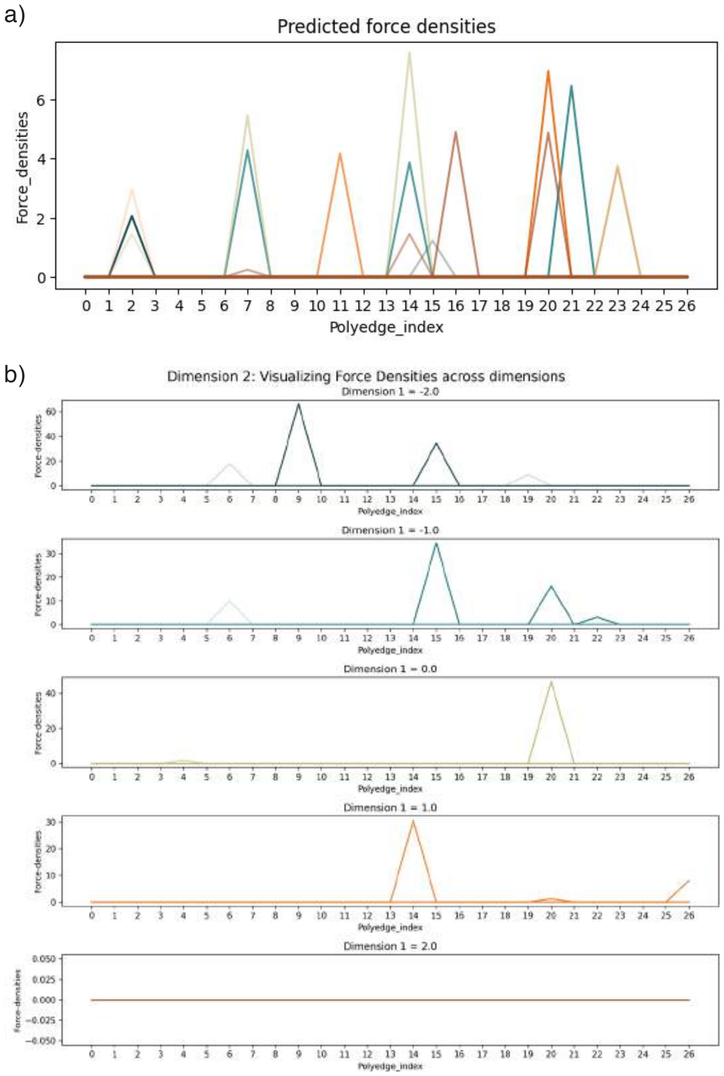


FIGURE 117: Force densties of 50 predicted samples (top). force densities of 5 new. (Own Work)

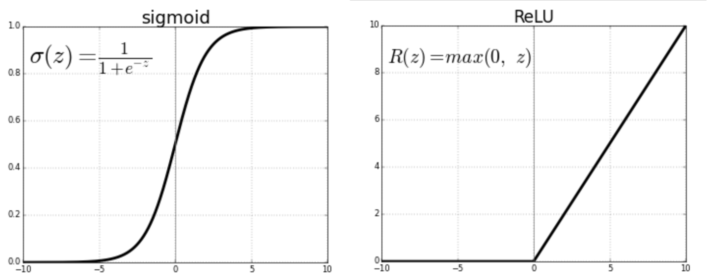


FIGURE 118: sigmoid (left) and ReLU activation function. Image taken from https://www.researchgate.net/publication/352419028_Automatic_decision_making_system_with_environmental_and_traffic_data/figures?lo=1

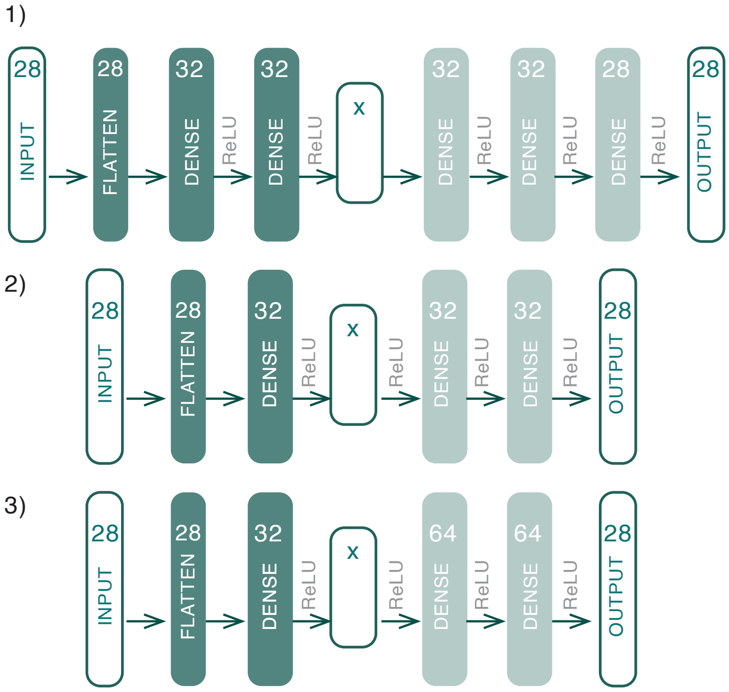


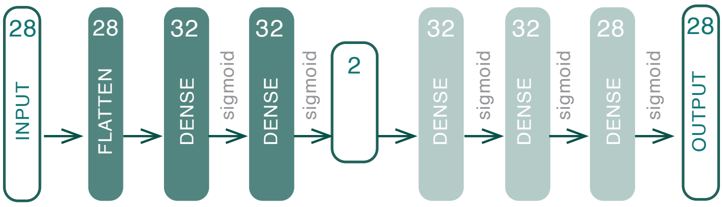
FIGURE 119: VAE architectures tested only using ReLU. The corresponding hyperparameters are shown in Table 15. (Own Work)

DATASET: randomized				
#	Hyperparameters - in models where the only activation function that used is ReLU			
	Latent Dimension (x)	Beta	Batch size	Learning Rate
1a	2	1	128	1E-05
2a	3	1	128	1E-05
3a	3	0.2	128	1E-05
4a	4	0.2	128	1E-05
5a	4	0.2	128	1E-03
1b	2	0.5	256	1E-05
2b	2	0.1	512	1E-05
1c	2	1	128	1E-05
2c	2	0.2	256	1E-05

TABLE 15: VAE hyperparameters tested only using ReLU. The corresponding architectures are shown in Figure 119 (Own Work)

Only Sigmoid

When sigmoid was used in combination with ReLU or with sigmoid alone, the decoder of the latent space produced non-zero values with significantly smaller peaks. These are further highlighted using by visualizing the latent space and sampling new designs from across its dimensions. In the simulations the latent dimensions were kept 2 which is



HYPERPARAMETERS: latent_dimension = 2, **beta** = 0.2, **epochs** = 600, **batch_size** = 128, **learning_rate** = 1E-05

FIGURE 120: VAE architecture - only sigmoid (Own Work)

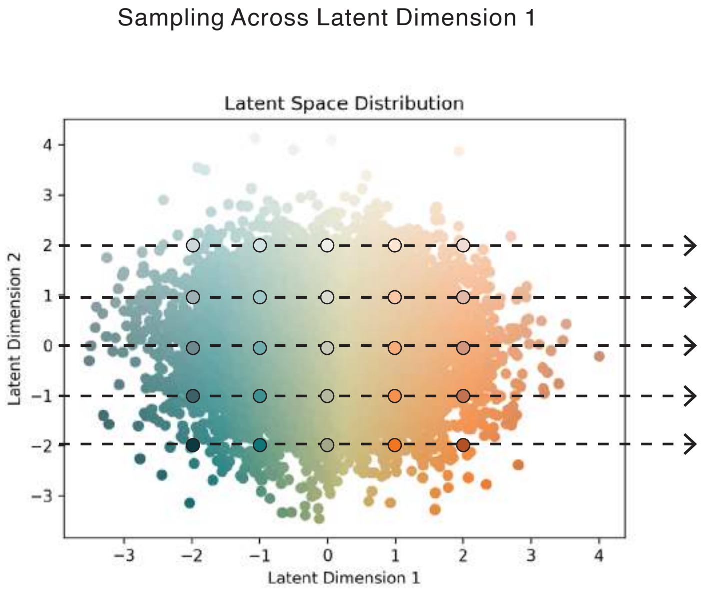


FIGURE 121: Latent space distribution across dimension 1 - only ReLU. (Own Work)

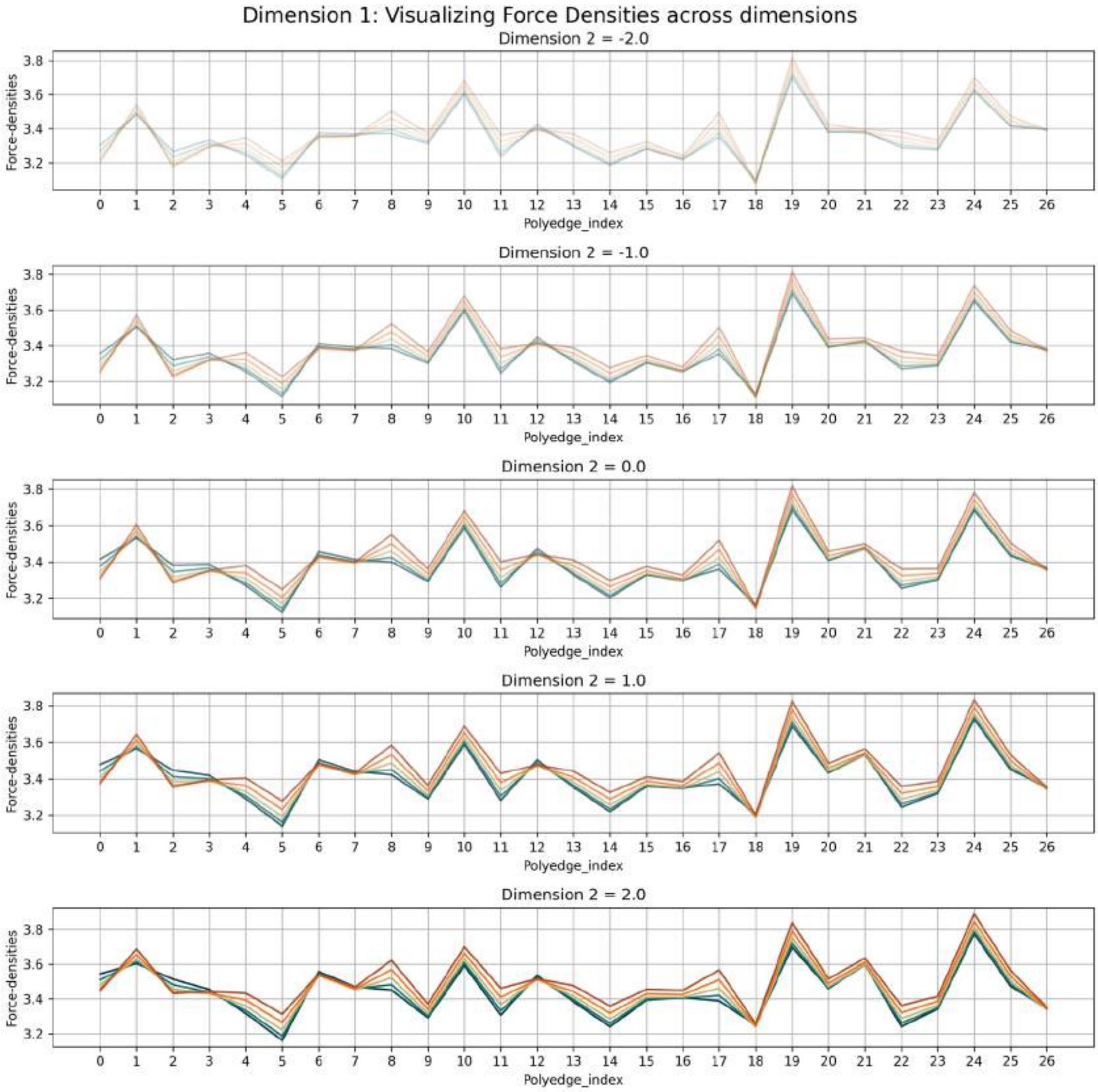


FIGURE 122: Visualizing Force densities of samples across Latent dimension 1 - only ReLU. (Own Work)

easier to visualize along the x (Latent Dimension 1) and y axis (Latent Dimension 2).

Figure 122 and Figure 123 show the newly sampled force meshes and their force densities across each latent dimension. Although samples were taken from across a large area of the latent space as visualized in Figure 124, the samples that were produced were very similar. The values of force densities range between 3.1 and 3.9 despite the large variation in input data. The same peaks were seen in all samples, with deviation from the mean across different samples. When we move across Latent Dimension 2 in the positive axis, the value of the force densities reduce (marked in the figures as dark to light) except for some polyedges whose force densities remain nearly constant with minimal variation like 9,10, and 19. Movement across Latent Dimension 2 produced a similar result in the fact that there is little deviation from the standard pattern.

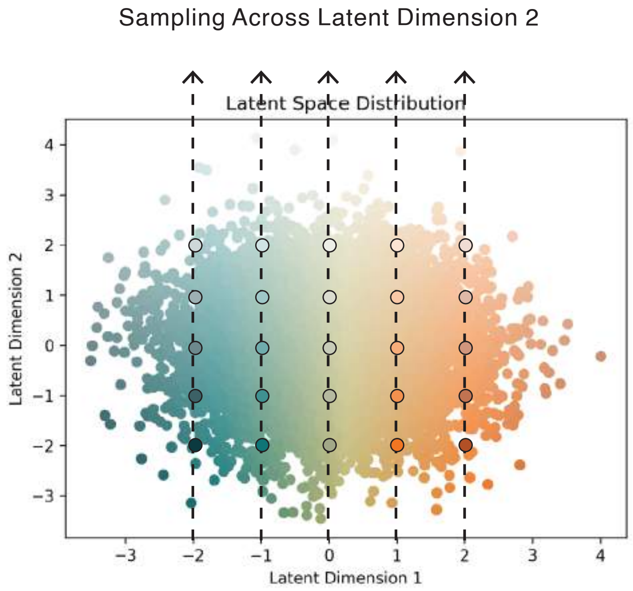


FIGURE 124: Latent space distribution across dimension 2 - only ReLU

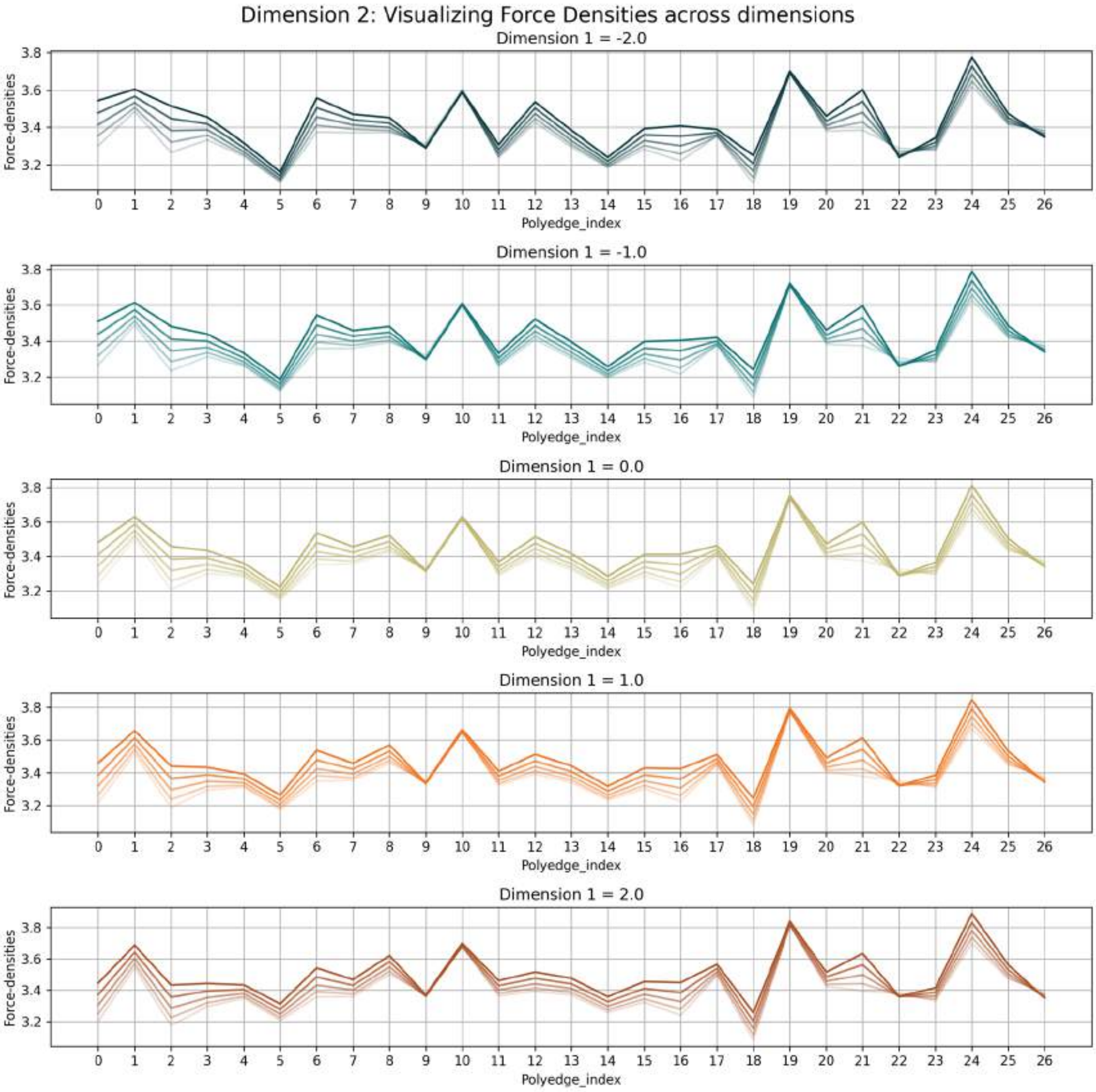


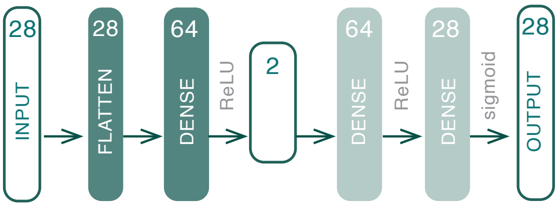
FIGURE 123: Visualizing Force densities of samples across Latent dimension 2 - only ReLU. (Own Work)

However, the pattern changes differently as movement in the positive direction sometimes causes an increase (Polyedge 0, 2, 3, 6,and 7), while sometimes there is a decrease (Polyedge 13-25) and in some cases there is minimal change (6, 7, 18, and 26). It is important to note that the same patterns were followed at different positions in the latent space.

There is a clear linear trend in thickness as it reduces with movement in the positive direction of the Latent space in either Latent Dimension. There is minimal variation overall with a change of just a millimeter (0.0631m-0.0641m).

Relu + Sigmoid

It could be concluded that using either ReLU or sigmoid alone for the VAE would not be adequate. Using ReLU as the only activation function produced many zero force densities as that results in invalid meshes. It produced higher peak values than using only the sigmoid function - which smoothens the distribution along a mean. Using the sigmoid function alone smoothens the distribution and it results in producing very similar samples. For this reason, a models were tested which used a combination of the two to produce valid meshes that were not too similar.



HYPERPARAMETERS: latent_dimension = 2, **beta** = 0.2, **epochs** = 600, **batch_size** = 128, **learning_rate** = 1E-04

FIGURE 125: Architecture of VAE - sigmoid + ReLU. (Own Work)

Similar to the model with sigmoid alone, the force densities were sampled from a low range but there was a higher degree of randomness in sampling each sampling each polydege's force density. Nonetheless, both dimensions showed similar patterns in how force densities change across the dimensions. There is an oscillating trend where force densities increase unlike the linear trends earlier shown by ReLu alone. Across Dimension 1, there is a general trend of values moving towards a maximum in the centre (maroon) region while minimum values are seen towards the peripheries of the latent space (blue). Another trend can be seen where the periphery minimas remain (blue) but central values (maroon, orange) are maximum at the peripheries of the latent space but they decrease towards the centre. This is seen especially evident in comparison sets such as where Latent Dimension 2 = 0.0 and 0.5 where the maximas shift.

In Dimension 2, the same trend can be seen as dimension one where the force densities of the samples towards the

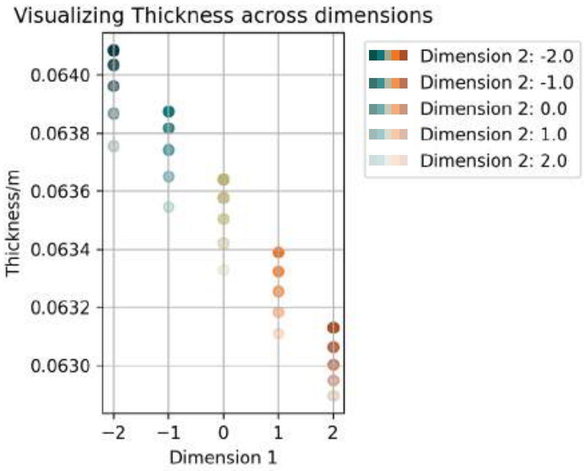


FIGURE 126: Visualizing thickness across dimensions - only sigmoid. (Own Work)

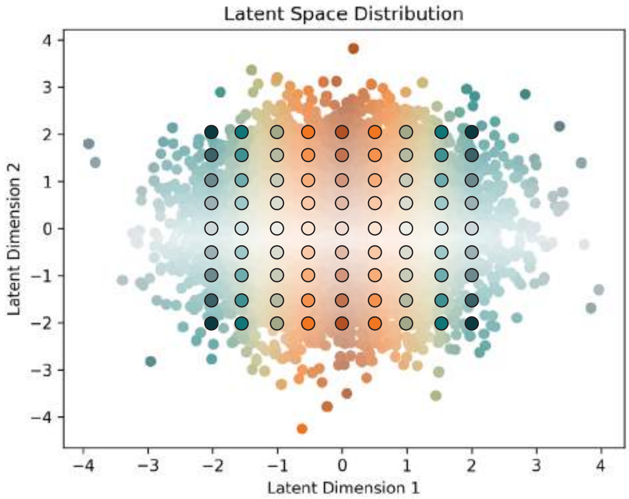


FIGURE 127: Latent space distribution across dimensions - sigmoid + ReLU. (Own Work)

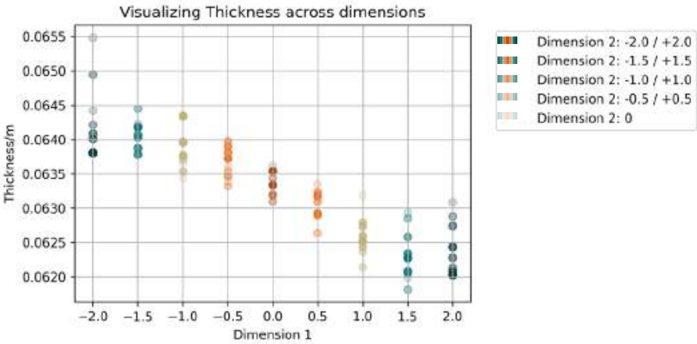


FIGURE 128: Visualizing thickness across dimensions - sigmoid + ReLU. (Own Work)

centre (dark) decrease but the minimas (light values) stay constant.

It is interesting to note is that the thickness does not follow the same trend as the force densities even though it is just another value in the same array in the training set. An oscillating trend can be seen where thickness reduces towards the positive direction of Dimension 1. No clear pattern can be seen for Dimension 2 in this case.

There is still a minimal range over which the thickness is distributed (0.0618m to 0.0655m) over the sampling space considered. However, a better way of gauging the maximum thickness represented by the latent space of the VAE would be by using the gradient descent for maximizing the Buckling Load Factor. As buckling is highly dependent upon thickness, the maximum thickness was be sampled this way. This was found to be 0.0932m for this architecture - very close to the maximum thickness that was present in the dataset - 0.0950m. It should also be noted that although discrete values were input representing thickness of layers of tiles - the output from the latent space is of a continuous nature. This is expected as the distribution of the input data is mapped as a Gaussian distribution in the latent space.

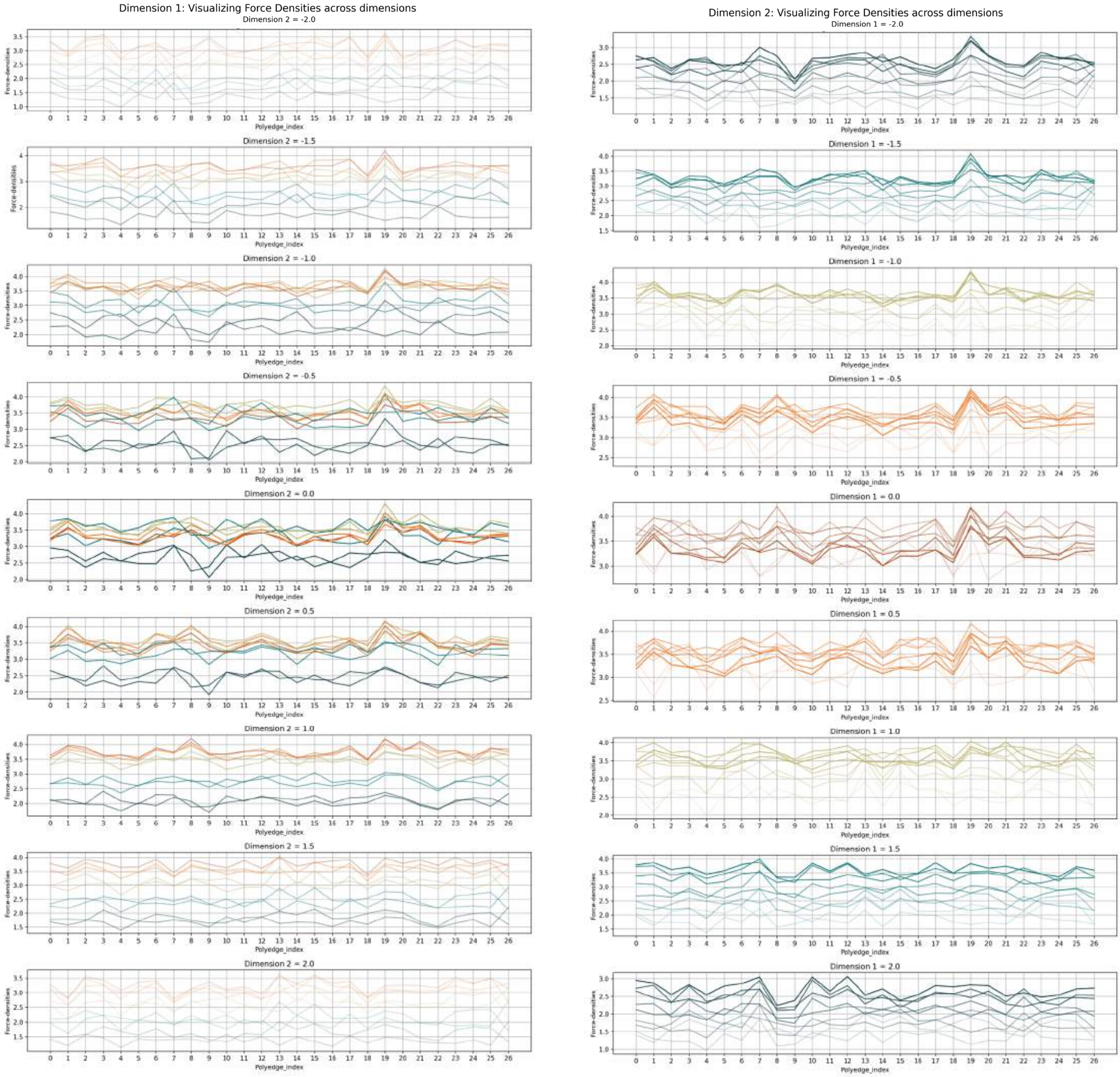


FIGURE 129: Visualizing Force densities of samples across Latent dimension 1 (left) and dimension 2 (right) - sigmoid + ReLU. (Own Work)

7.1.4 Conditioning the VAE

The next step was to extract samples from the VAE with specific to desired characteristics. The characteristic, in this case was the height of the vault, because for a floor slab certain heights are preferred over others. This was achieved through conditioning the features through supervised data (See Section 3.3.3 for details on derivation). Therefore, in addition to θ (features), c (labels) were also added to the encoder and decoder.

The heights labels, c, were rounded off to the nearest 0.1m. This gave a total of 58, and 28 different labels respectively for One-Hot encoding and Sinusoidal Positional Encoding ranging from 0.5m to 26.7m. All of the labels are shown in Table 16 against the frequency of occurrence in the main dataset in terms of percentage. This is also plotted below in the Figure 130. Different strategies were tried to represent the label c.

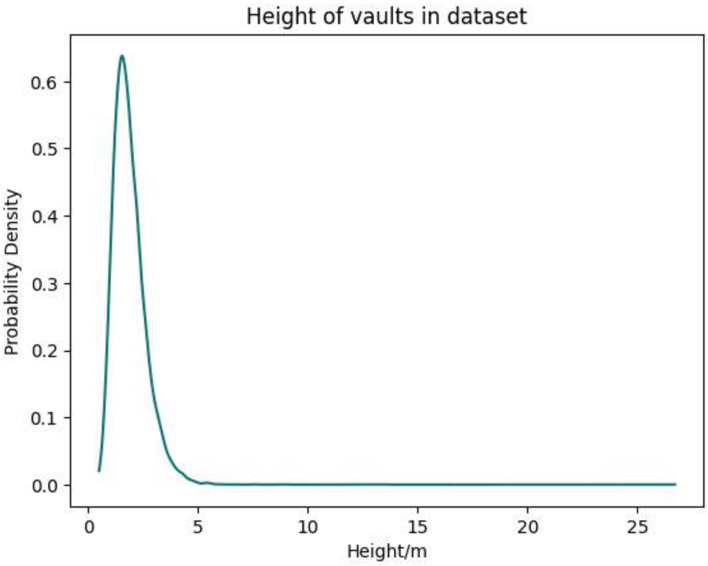


FIGURE 130: Probability Density of Heights of randomized Dataset. (Own Work)

- Label Encoding

  example: 0.5m is represented as an integer like 3. See Section 3.3.9 for detail on encoding strategies.

- One-Hot-Encoding

  example: 0.5m is represented as an array of shape (58,) where only the position of the one-hot-encoded label is represented as 1 while the rest are 0.

  [1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

- Sinusoidal Positional Encoding

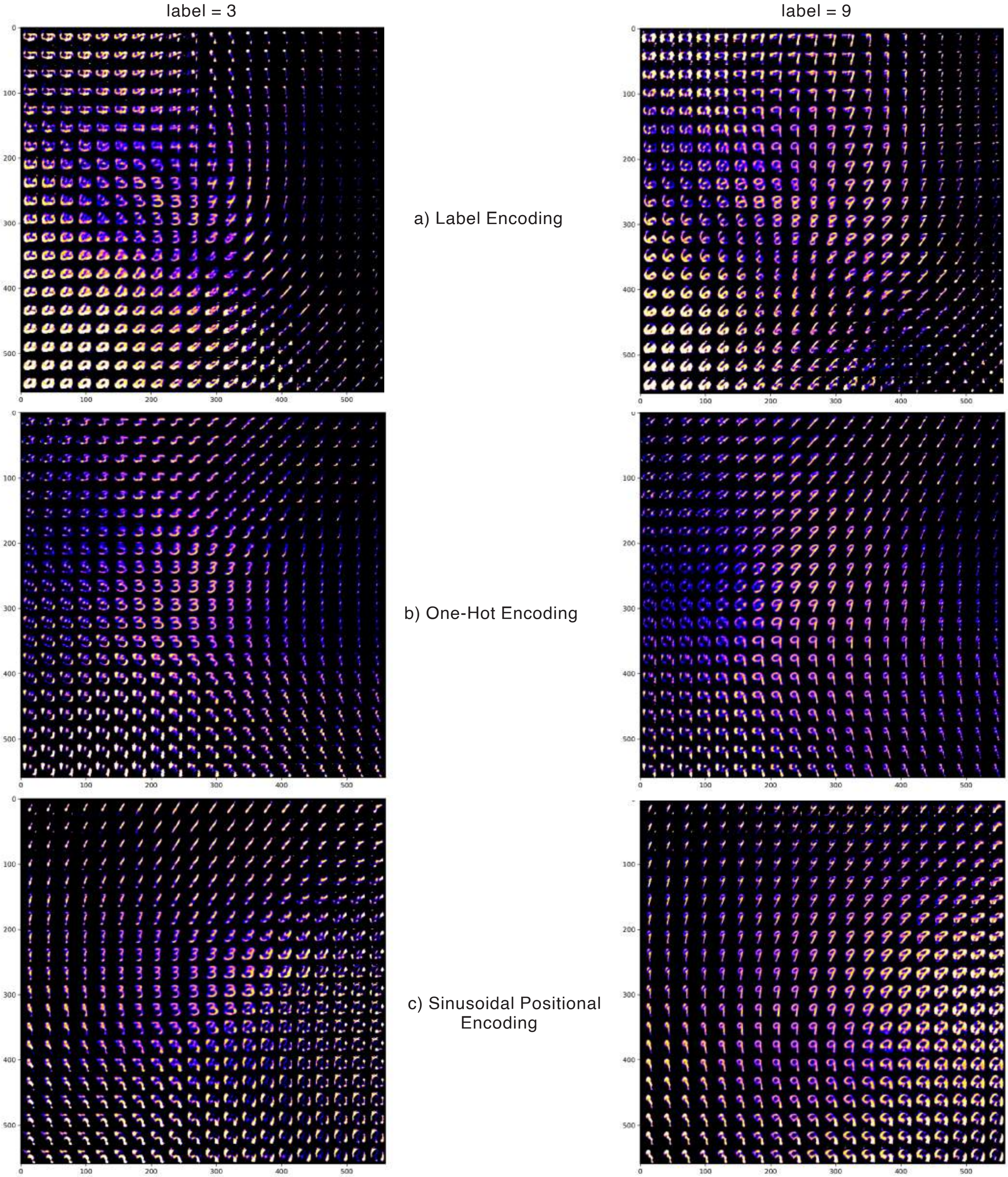
  example: 0.5m is represented as a tensor of a user-defined dimension. Since the feature shape is (28,), this is also the same shape of the encoding.

DATASET: randomized			
Label Height /m	%	Label Height /m	%
0.5	0.12	3.4	0.66
0.6	0.32	3.5	0.58
0.7	0.74	3.6	0.45
0.8	1.38	3.7	0.38
0.9	2.32	3.8	0.37
1	3.22	3.9	0.24
1.1	4.28	4	0.3
1.2	5.49	4.1	0.12
1.3	5.52	4.2	0.24
1.4	6.37	4.3	0.17
1.5	6.54	4.4	0.13
1.6	6.58	4.5	0.09
1.7	6.1	4.6	0.06
1.8	5.9	4.7	0.08
1.9	5.7	4.8	0.06
2	4.7	4.9	0.04
2.1	4.33	5	0.02
2.2	4.64	5.1	0.02
2.3	3.71	5.3	0.03
2.4	2.9	5.4	0.04
2.5	2.91	5.5	0.02
2.6	2.45	5.6	0.02
2.7	2.11	5.8	0.01
2.8	1.73	6	0.01
2.9	1.49	6.7	0.01
3	1.19	7.6	0.01
3.1	1.2	8.9	0.01
3.2	0.91	13.3	0.01
3.3	0.96	26.7	0.01

TABLE 16: Height labels and their distribution in the randomized dataset. (Own Work)

Label: One-Hot-Encoded Vector vs Label Encoding vs Sinusoidal Positional Encoding

To determine the appropriate strategy, initial tests were carried out for checking the script whether the unique samples were being extracted as desired by the condition. The MNIST dataset was used at this stage only for visualization purposes. 400 samples were plotted across the latent dimensions in a 20x20 grid. The same architecture was used for all encoding strategies (794-32-32-2-32-32-794).



a) Label Encoding

b) One-Hot Encoding

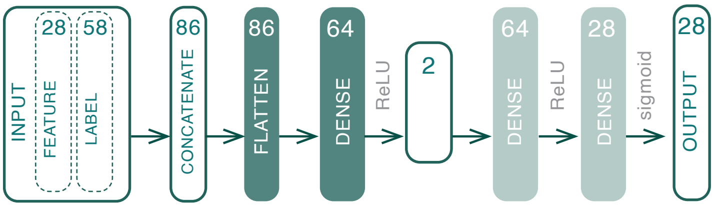
c) Sinusoidal Positional Encoding

FIGURE 131: Comparison of label conditioning strategies with MNIST dataset. a) Label Encoding, b) One-hot Encoding, c) Sinusoidal Positional Encoding. Samples across the latent space visualized from Conditional VAE (Own Work)

As shown in Figure 131, the distinguishable samples produced by One-Hot-Encoding and Sinusoidal Positional Encoding were constrained to the desired label (e.g. when label = 9, only handwritten digits of 9 are produced) but distinguishable samples that were produced from Label Encoding were across a range of labels (e.g. when label = 9, handwritten digits of 6,7,8,9 are produced and when label = 3, handwritten digits of 3,4 are produced). Therefore, it was evaluated that One-Hot-Encoding and Sinusoidal Positional Encoding produce better results so these were used for the Randomized Force densities dataset.

One Hot Encoding

Similar to the MNIST dataset, samples from the randomized force densities dataset were sampled from across the latent space for different labels. For 5 different height labels (0.5m, 0.8m, 1.0m, 1.5m, 1.8m), 1000 samples each were randomly taken from the area under the graph represented by the rectangle in Figure 133a. A surrogate model was used for predicting their heights and these were plotted in the form of a probability density graph in Figure 133b. This represents the probability of sampling heights when a desired height is input as the label/ condition.



HYPERPARAMETERS: latent_dimension = 2, **beta** = 0.2, **epochs** = 600, **batch_size** = 64, **learning_rate** = 1E-04

FIGURE 132: Conditional VAE architecture and hyperparameters where label is a One-Hot-encoded vector. (Own Work)

Height of the Conditioned Samples

From the results, we can see that conditioning with a height does not produce the exact same height but a range of heights on either side of the label. Even though a One-Hot-Encoded Vector was used, the results are similar as produced previously in the MNIST dataset when Label Encoding was used.

Furthermore, we can see that from 0.5m to 1.0m, samples are produced close to the conditioned value. However, when the height is further increased, very few samples close to that desired label are found. It is hypothesized that because the distribution of the heights in the training set is concentrated towards the lower heights, perhaps that is why sampling from lower heights produces higher accuracy. It must be noted, however, that the heights that are sampled fairly better do not have the highest frequency of occurrence in the main dataset. The distribution is skewed to the left as shown in Figure 133c. It should be noted that 5m onwards, labels are not shown in the figure.

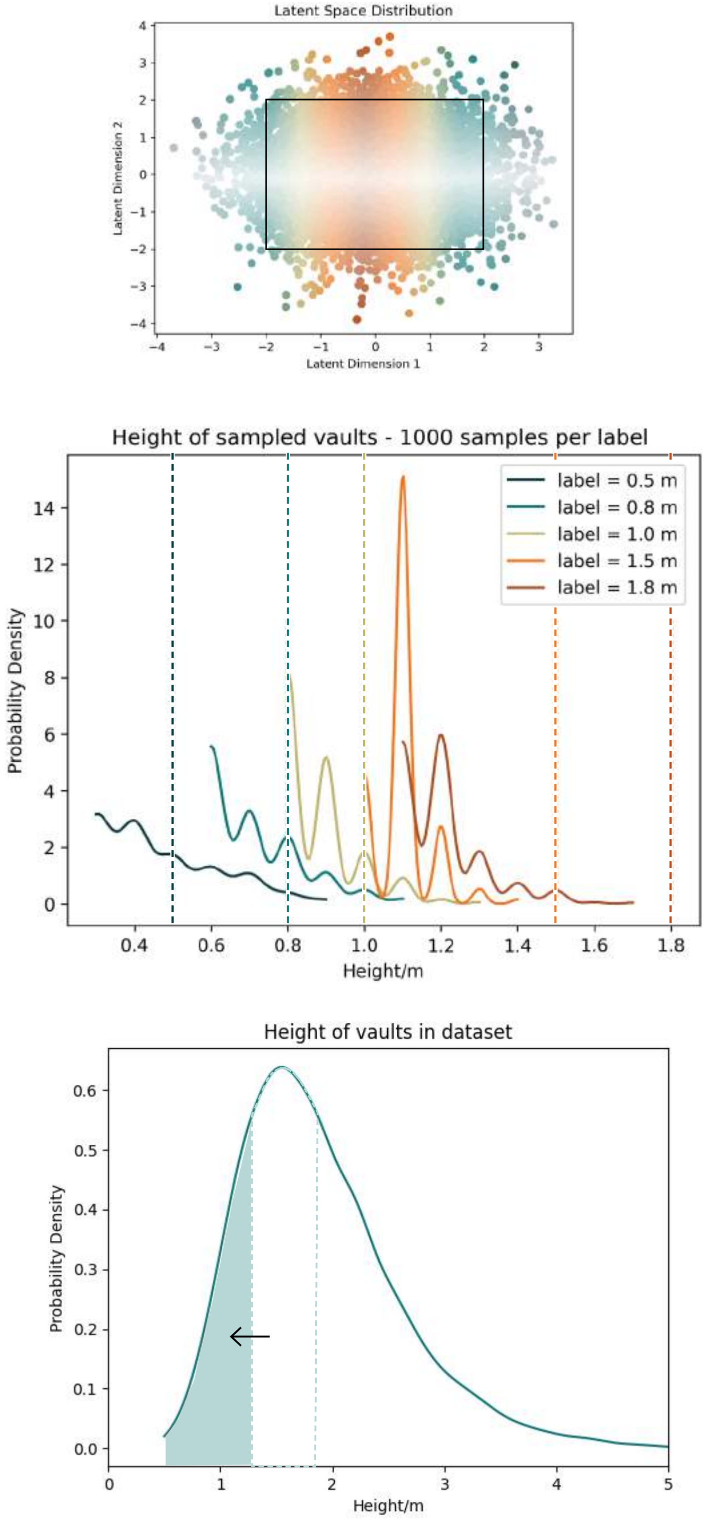


FIGURE 133: a) CVAE latent space , b) height - probably density relationship for 1000 samples taken from the latent space (x = -2 to 2, y= -2 to 2) for different labels of the CVAE, c) Probability Density of Heights of the randomized dataset. Label is One Hot Encoded Vector. (Own Work)

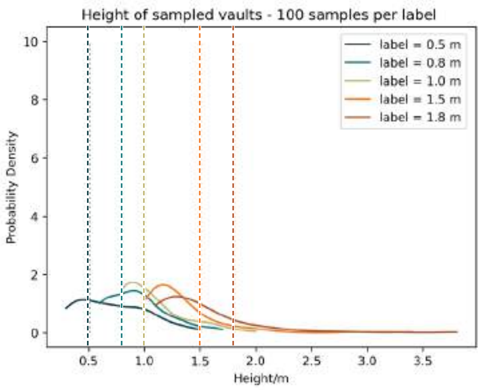


FIGURE 134: height - probably density relationship for 100 samples taken from the latent space (x = -3 to 3, y= -3 to 3) for different labels of the CVAE, where label is One Hot Encoded Vector. (Own Work)

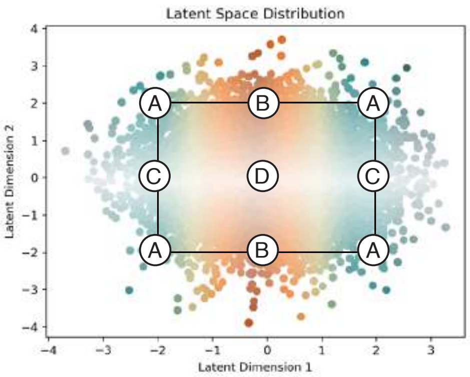
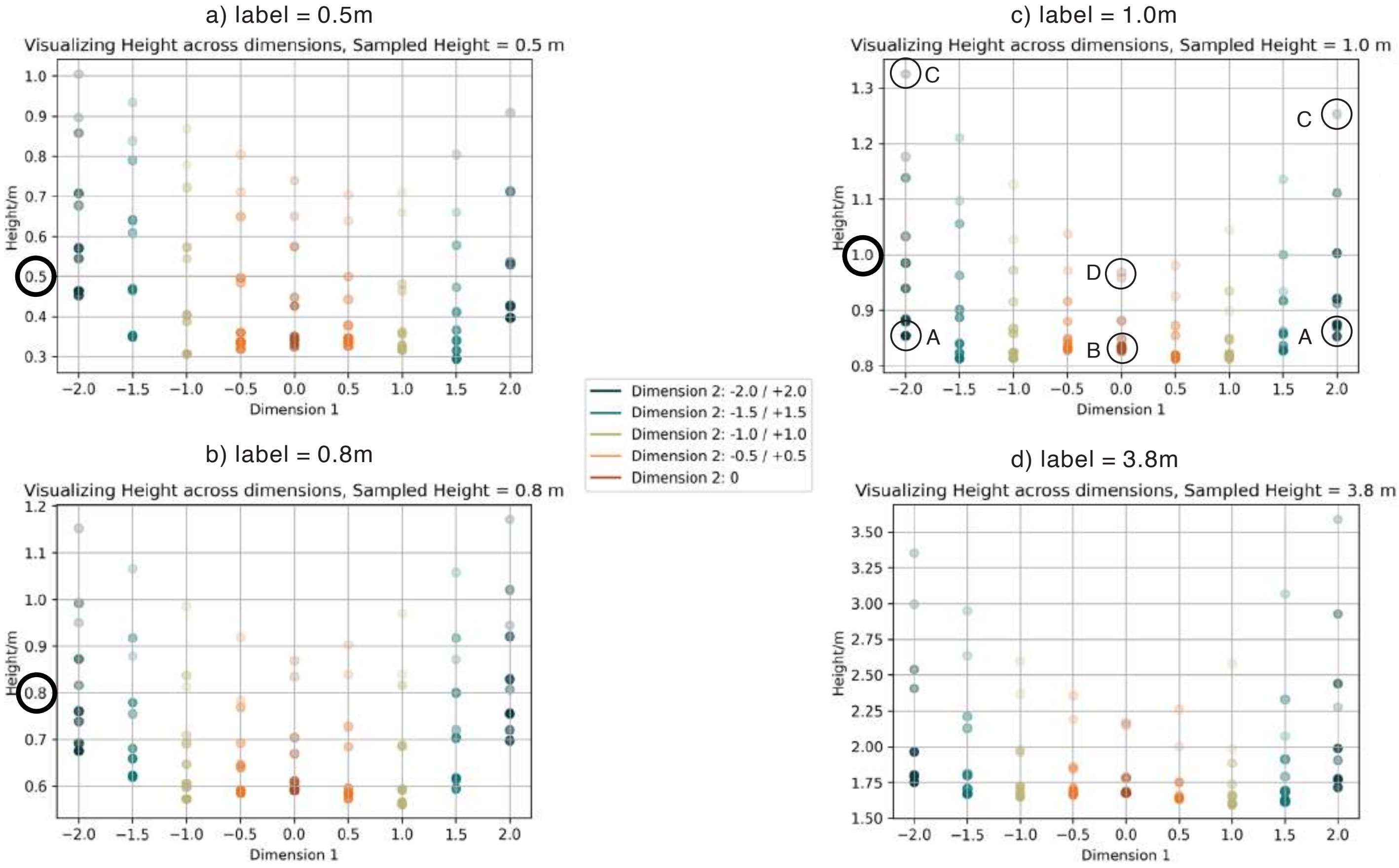


FIGURE 135: CVAE latent space where label is One Hot Encoded Vector. (Own Work)


a) label = 0.5m

b) label = 0.8m

c) label = 1.0m

d) label = 3.8m

FIGURE 136: Visualizing height across dimensions of the CVAE for different labels where label is One Hot Encoded Vector. (Own Work)

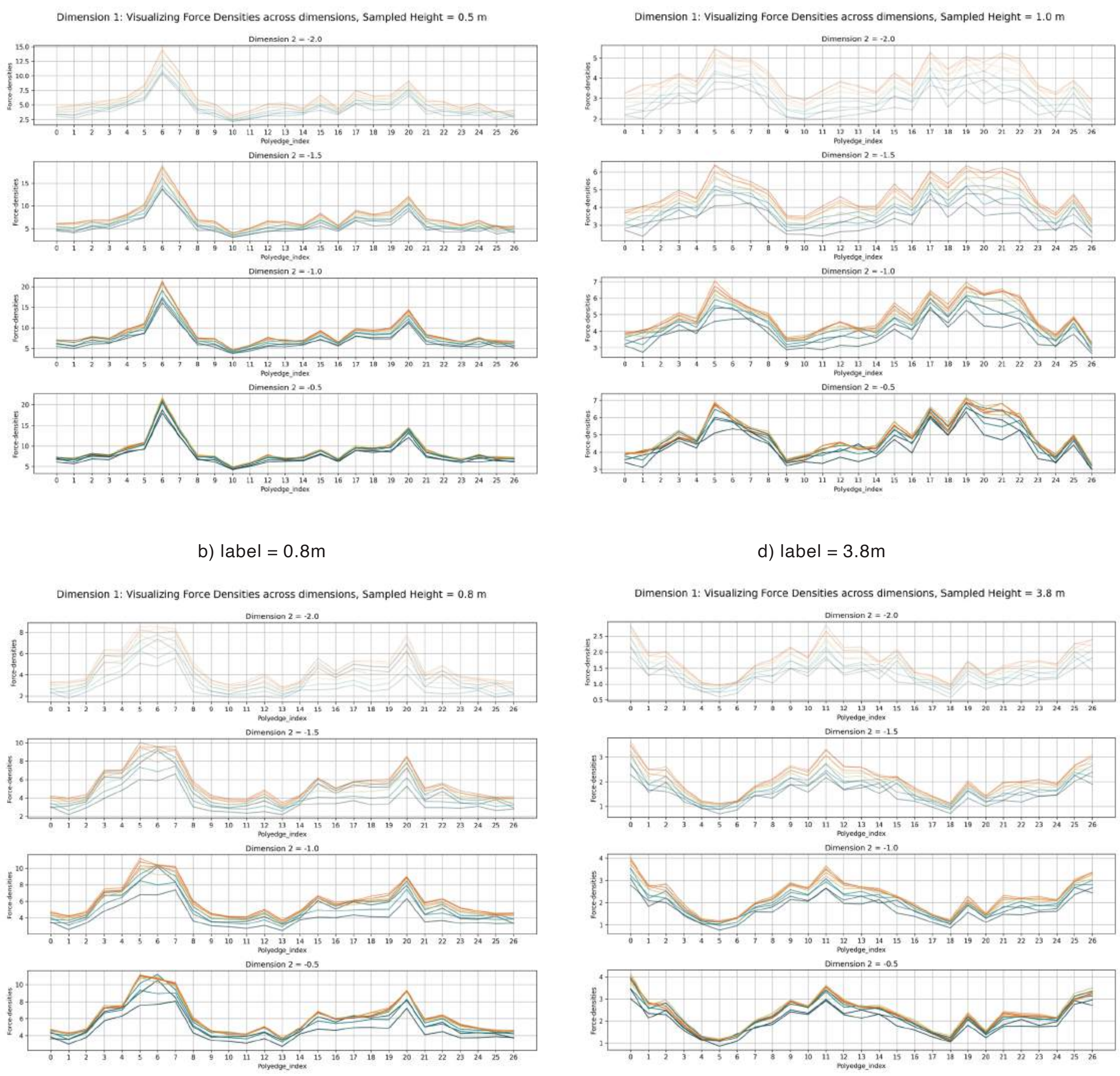
It should also be noted that meshes with extreme values of heights can be sampled if the area of the rectangle in Figure 133b. However, this would also introduce samples out of the distribution. This is shown in Figure 134.

Patterns in Sampling Heights across the Latent Dimensions

9x9 samples were taken from the area under the graph represented by the rectangle in Figure 135. From the graph it becomes apparent how the pattern of heights changes from the periphery towards the centre of the latent space.

In Figure 136, at the centre of Latent Dimension 2 (Latent Dimension 2 = 0), as we move across Dimension 1 from its periphery (C) towards its centre (D), the height of the mesh decreases under all label conditions. This change in height decreases as the we move from the centre towards the periphery of Latent Dimension 2 (from a CD trajectory to an AB trajectory). This is marked in Figure 136c.

On the contrary, at the centre of Latent Dimension 1 (Latent Dimension 1 = 0) moving across Latent Dimension 2 from its periphery (B) to its centre (D) causes an increase in height. Similarly, this change in height increases as we move from the centre towards the periphery of Latent Dimension 1 (from a BD trajectory to an AC trajectory). See Figure 136c.


a) label = 0.5m

c) label = 1.0m

b) label = 0.8m

d) label = 3.8m

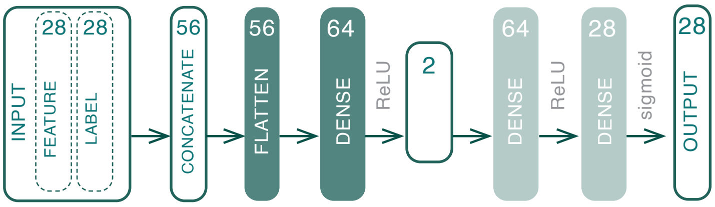
FIGURE 137: Visualizing Force densities of samples across Latent Dimension 1 of the CVAE for different labels. Label is One Hot Encoded Vector (Own Work)

Patterns in Sampling Force Densities across the Latent Dimensions

For each label, a similar pattern follows when we scroll across Latent Dimension 1 which has been discussed in Section 6.2.4. What is notable is that for each label, the meshes seem to have a different force density pattern. Scrolling across Latent Dimension 1 only produces fluctuations in that pattern but it largely remains the same as seen in earlier cases

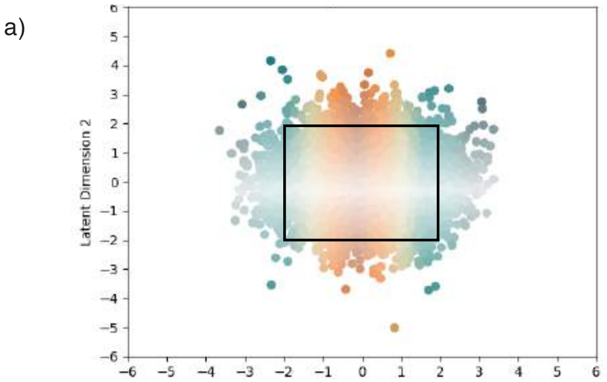
Sinusoidal Positional Encoding

Besides the One-Hot-Encoding, Sinusoidal Positional Encoding was also explored for encoding the labels. The results were very different. The number of dimensions is not restricted to the number of different labels unlike One-Hot-encoded vectors so Sinusoidal Encoded provided greater flexibility. The dimensions of the features were matched so the shape was (n,28). Figure 139 and Figure 140 show the distribution of heights of 100 meshes each for each label (0.5m, 0.8m, 1.0m, 1,8m). The rectangles in Figure 139a and Figure 140a show where in the latent space the meshes were sampled from.



HYPERPARAMETERS: latent_dimension = 2, **beta** = 0.2, **epochs** = 600, **batch_size** = 64, **learning_rate** = 1E-04

FIGURE 138: Conditional VAE architecture and hyperparameters where label is sinusoidally positional encoded. (Own Work)



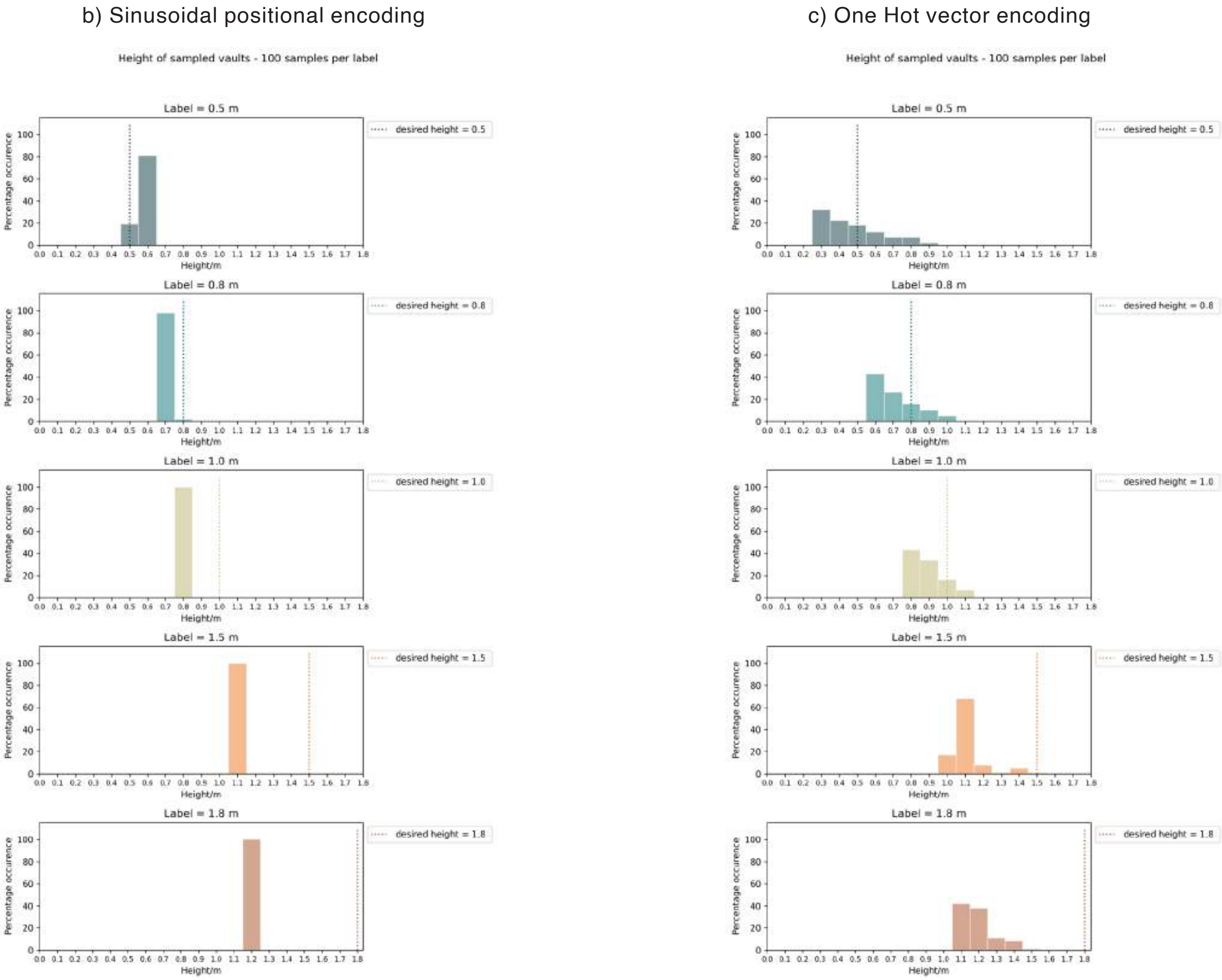
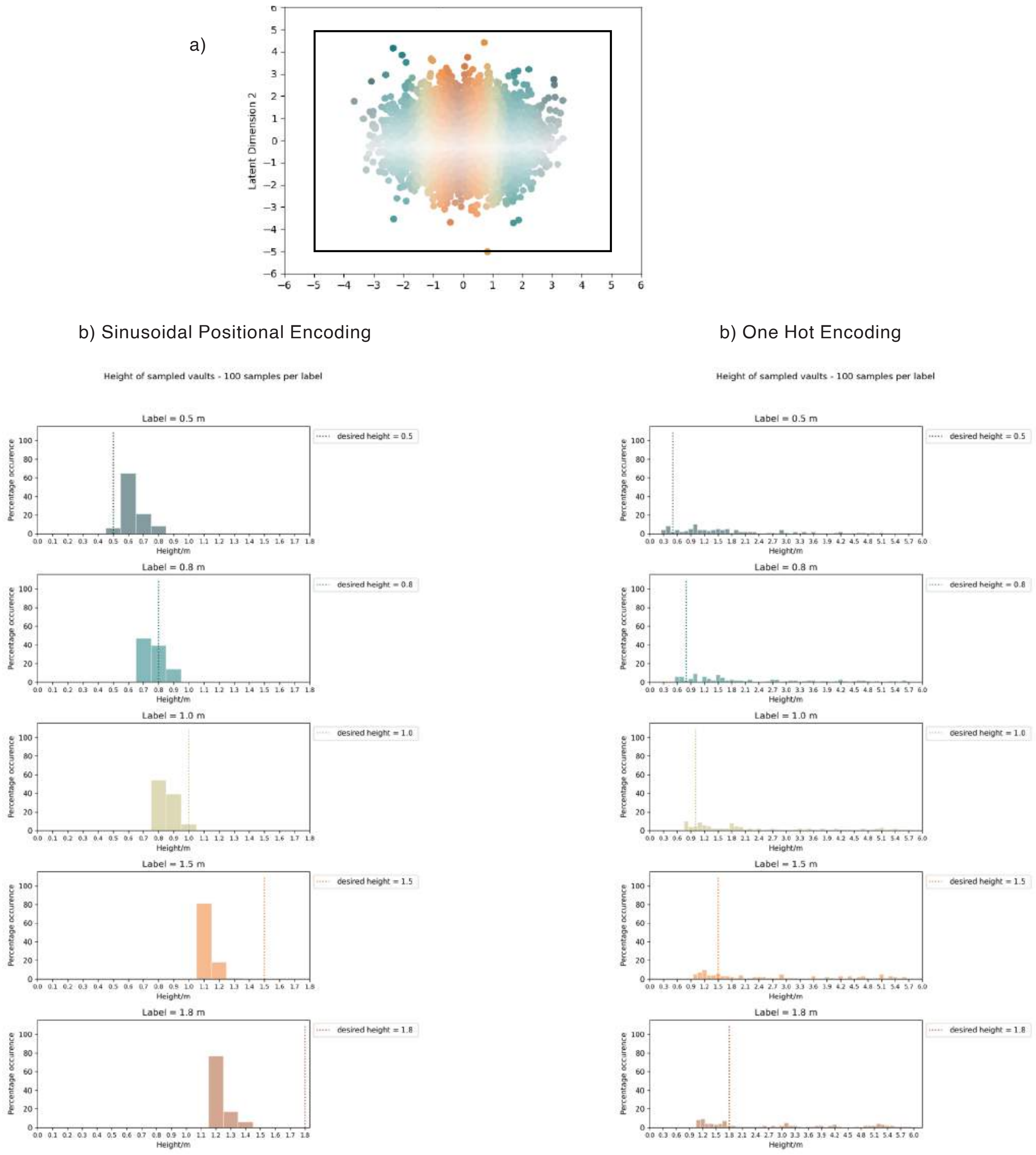


FIGURE 139: a) a small dense area of the CVAE latent space shown by the rectangle to be sampled from, b) and c) Percentage of Occurrence of heights of samples for each label where samples are randomly taken from a dense latent space in a). (Own Work)


a)

b) Sinusoidal Positional Encoding

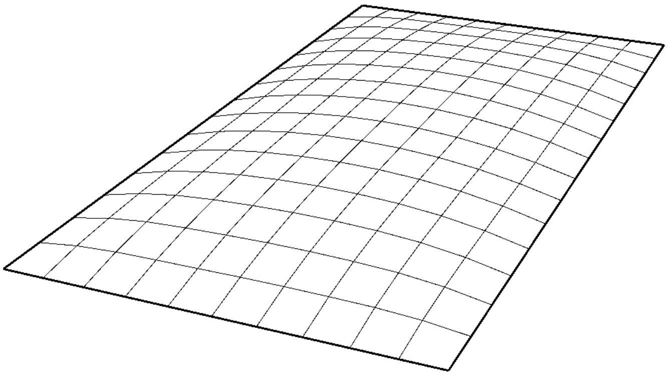
b) One Hot Encoding

FIGURE 140: a) a large sparse area of the CVAE latent space shown by the rectangle to be sampled from, b) and c) Percentage of Occurrence of heights of samples for each label where samples are randomly taken from a large sparse latent space in a). (Own Work)

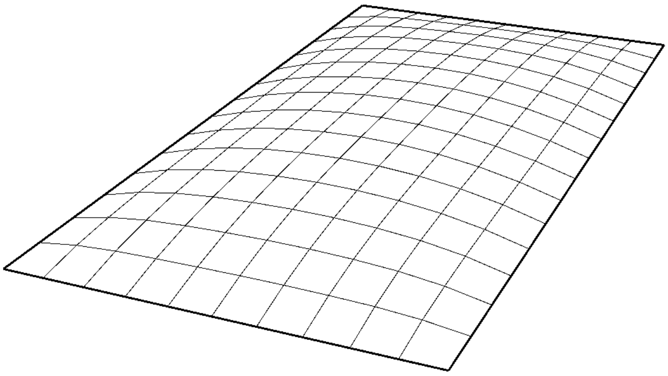
The most prominent difference can be seen that the Sinusoidal Positional Encoding constrained the sample heights to a much larger degree as compared to the One-Hot-encoded vector. Therefore, there was much less variation in the spread of sample heights generated. See Appendices for other results for the positional encoding.

The meshes sampled from different height labels for the Sinusoidal Positional Encoding are shown below. For correction in scaling, the meshes have been scaled by twice the total load of the original dataset.

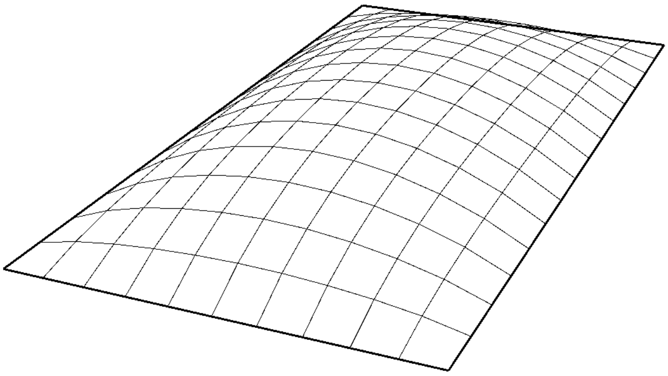
a) label = 0.5m, actual height = 0.7m



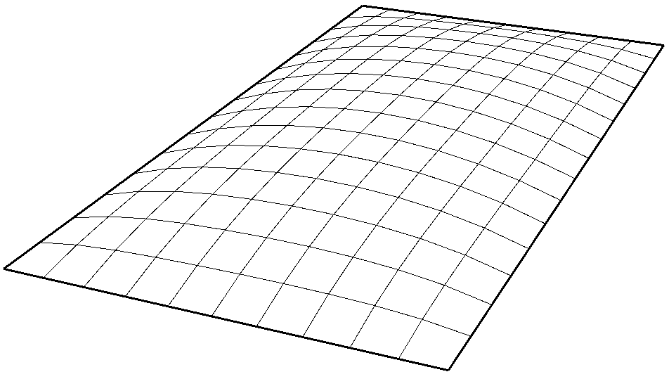
b) label = 0.8m, actual height = 0.8m



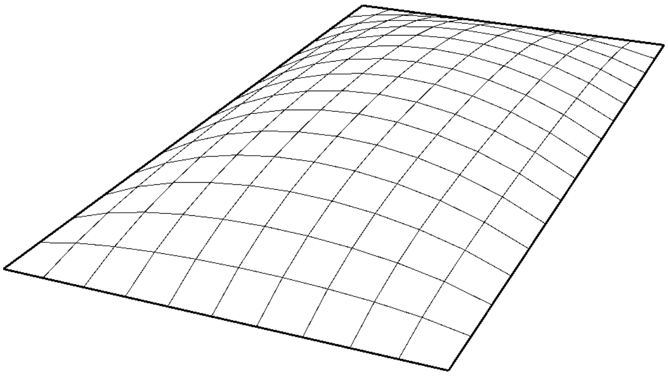
d) label =1.8m, actual height = 1.8m



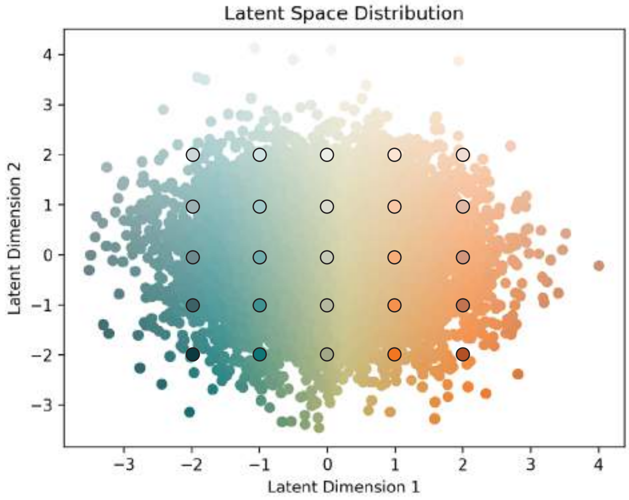
c) label = 1.0m, actual height = 0.9m

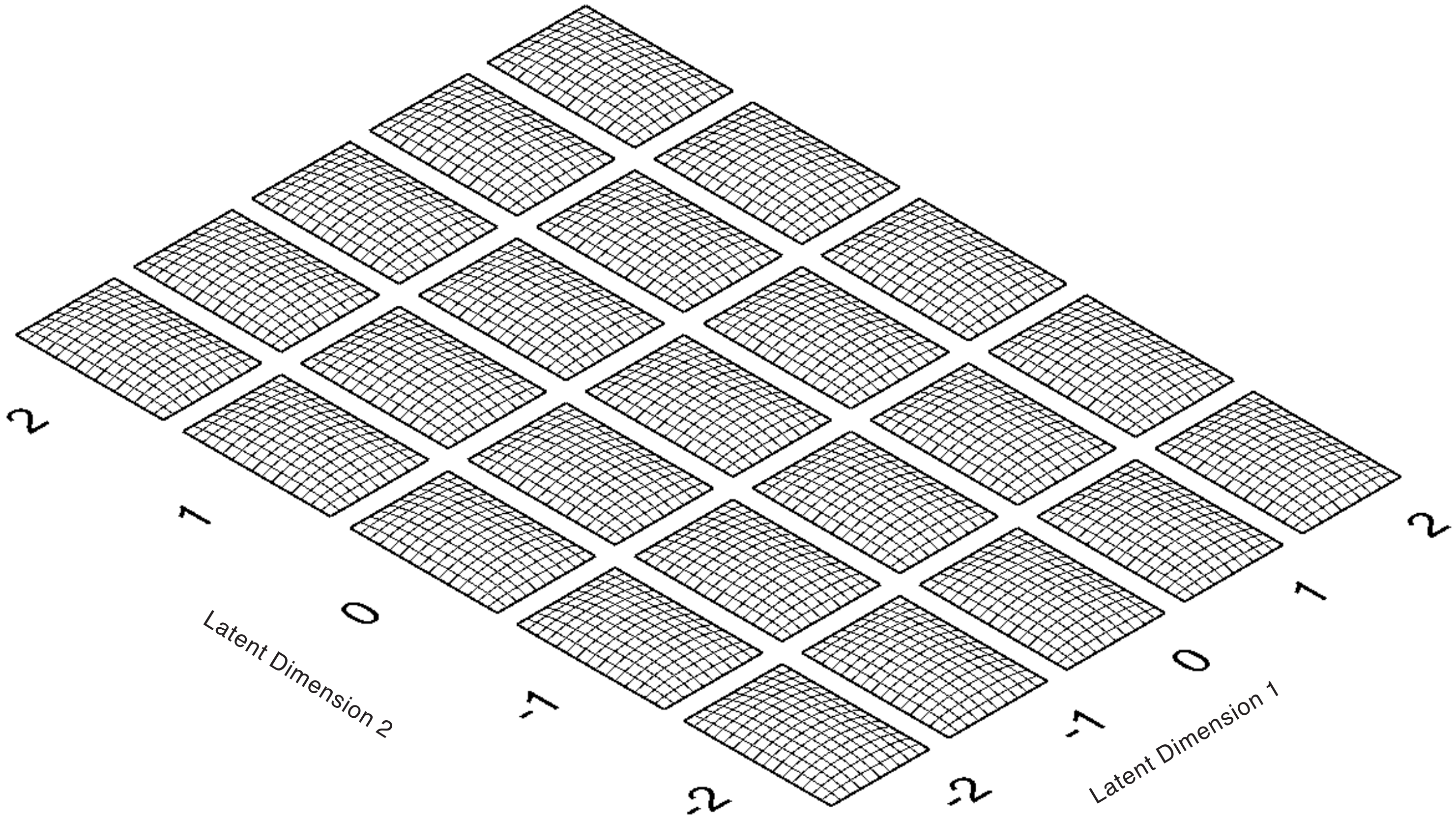


d) label = 1.5m, actual height = 1.4m



This is a visualization of the meshes sampled across dimensions for label of height = 1.8m, where the encoding is Sinusoidal Positional Encoding.





7.2 SURROGATE MODEL

See Section 4.1.5 for the workflow of the Surrogate Model.

For the label dataset, the SSS (Stiffness, Stability, Strength) performance metrics extracted. As the performance data for each mesh involved two simulations - one for seismic loads in x direction and the other in y direction - a combination strategy had to be implemented. SRSS was considered initially but would have yielded overestimations of Buckling Load Factor and underestimations of capacity in strength and SLS. As an alternate, the worst performance value was chosen (highest Utilization and Interstorey Drift Ratio, lowest Buckling Load Factor). The labels were then normalized with respect to each metric separately.

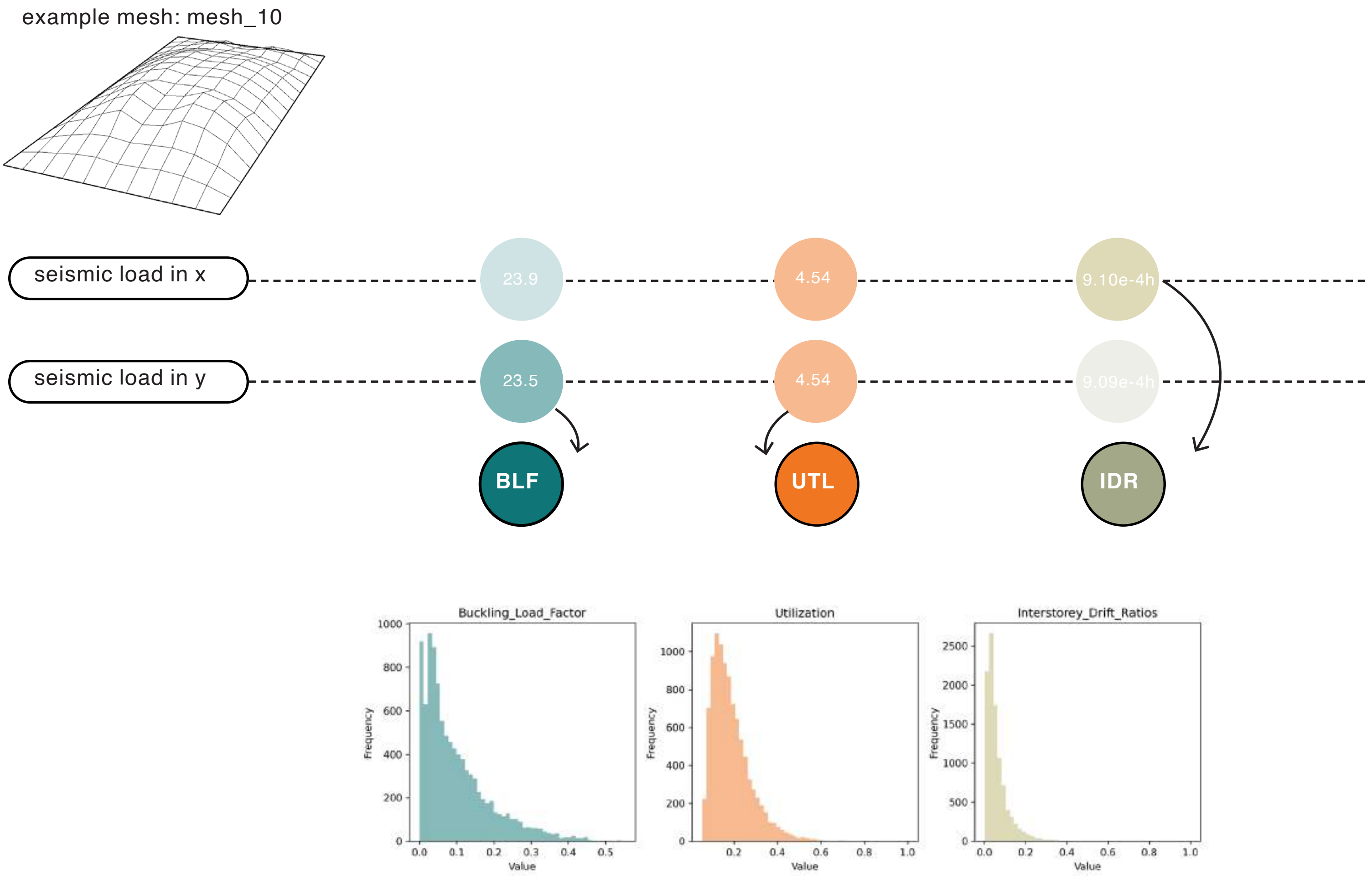


FIGURE 141: Selection of metrics for seismic load in both directions based on choosing the metric more likely to cause failure. (Own Work)

The best 10 samples of each metric were extracted in addition to 50 random test samples. This was done to compare with how the surrogate performed on samples not seen by during training.

7.2.1 Seismic Performance: 1 combined Surrogate Model

At the very first stages it was considered combining the three metrics into a single performance score. However, this approach would not give any indication of the type of failure whether it is in strength, stability, or stiffness. Moreover, the single score would be an oversimplification based on a biased formulation. Initially, one surrogate model was created which gave three metrics each as the output so the shape of the labels was (n,3). Although auxiliary inputs generally improve the training of the model, it was found that some metrics were performing better than others.

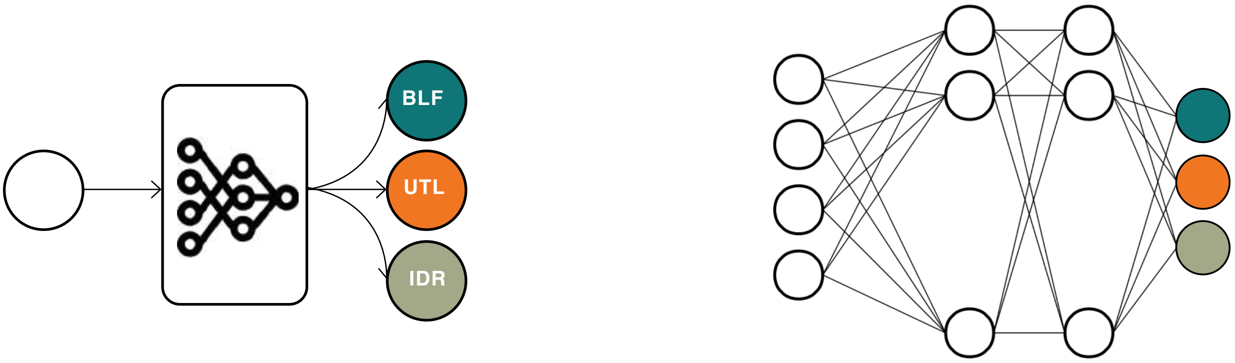


FIGURE 142: One surrogate model. inspired by the work of Sterrenberg (2023). (Own Work)

A likely possibility for this could be discrepancy in the weight of the loss function as the Mean Square Error of each loss is combined to give the overall loss.

$$loss_1 = \frac{1}{n} \sum_{i=1}^{n} \left(y_{1,i} - y_{1,true,i}\right)^2$$

$$loss_2 = \frac{1}{n} \sum_{i=1}^{n} \left(y_{2,i} - y_{2,true,i}\right)^2$$

$$loss_3 = \frac{1}{n} \sum_{i=1}^{n} \left(y_{3,i} - y_{3,true,i}\right)^2$$

where,

loss$_1$ = Loss of the Buckling Load Factor

loss$_2$ = Loss of the Utilization

loss$_2$ = Loss of the Interstorey Drift Ratio

The individual losses are then aggregated to form the overall loss

$$loss_{total} = w_1 . loss_{\ 1} + w_2 . loss_2 + w_3 . loss$$

Tensorflow assigns the weights (w$_1$, w$_2$, w$_3$) equally. This means that a loss function of an individual metric may have a larger effect than others.

Thickness not Considered

As mentioned earlier in the VAE section, early simulations did not include thickness so the shape of the feature dataset was (n,27). Figures 144 shows how well the model learnt to predict each metric for each test mesh sample that was not seen during training. It should be noted that the performance values on the y axis are normalized and do not indicate actual performance values.

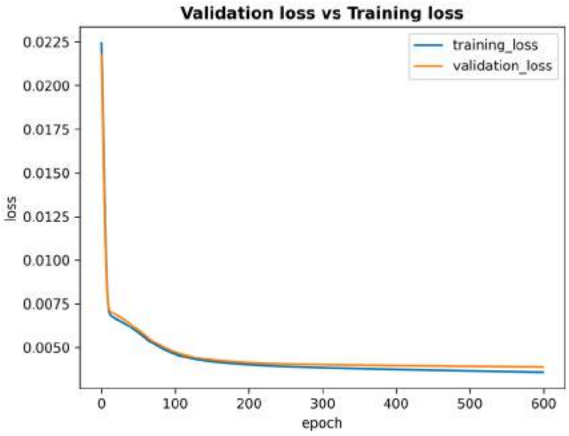
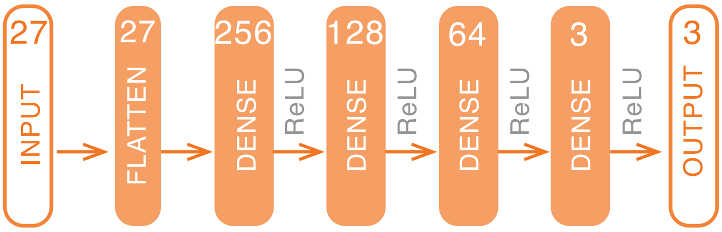


FIGURE 143: Validation vs Training loss - thickness not considered. (Own Work)



HYPERPARAMETERS: epochs = 600, **batch_size** = 128, **learning_rate** = 5E-06

FIGURE 145: SG (surrogate model) architecture - 4 dense layers thickness not considered. (Own Work)

This particular architecture performed well for Utilization and Interstorey Drift Ratios. Most models were able to predict Interstorey Drift Ratios while only some were able to predict Utilization. No model was able to learn how to predict Buckling Load Factor. This corresponds to the earlier findings mentioned in the previous chapter on the dependence of the Buckling Load Factor on the thickness of the vault. Reducing the learning rate from 5E-06 to 1E-07 required increasing the epochs from 600 to 40000 but that yielded nearly the same results.

Reducing the number of layers by removing the 256 neuron layer to make the neural network simpler did not help. The figure below shows the results. The architecture of the NN is shown in Figure 147.

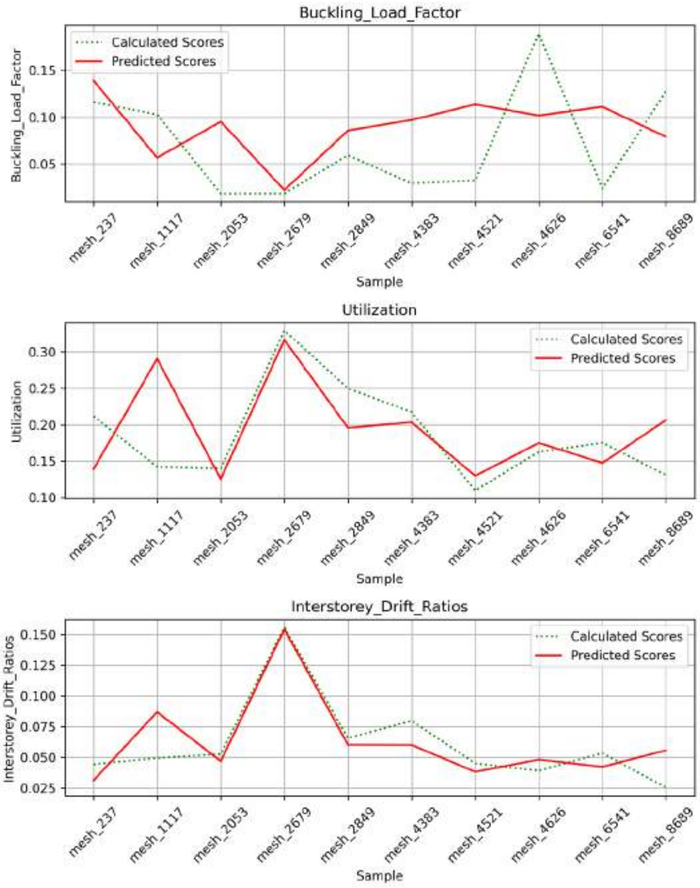


FIGURE 144: Ground Truth- Prediction 4 dense layers. (Own Work)

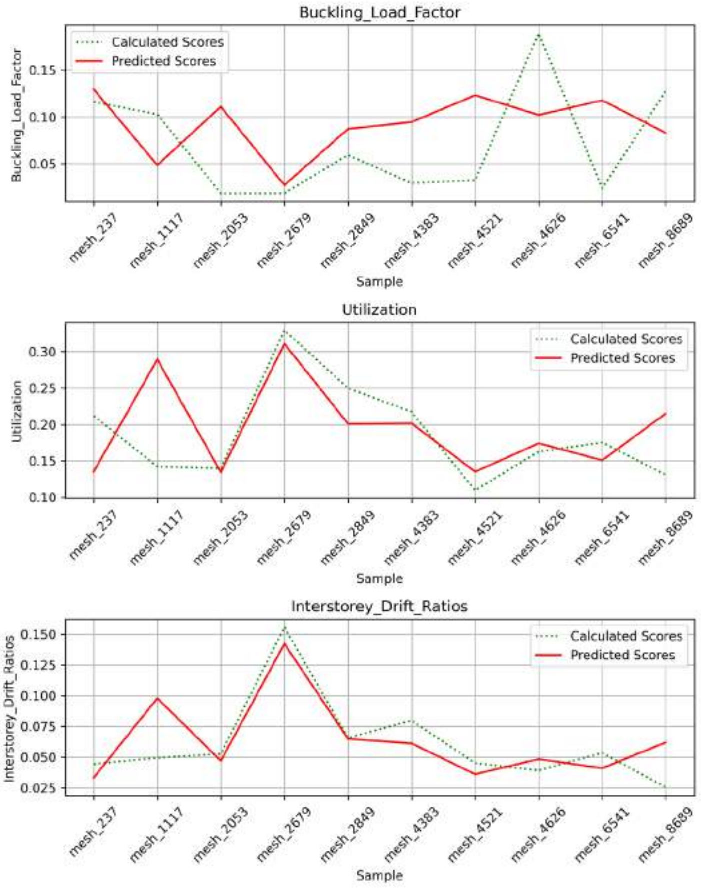
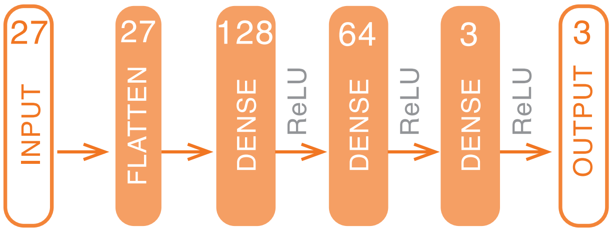


FIGURE 146: Ground Truth- Prediction 3 dense layers. (Own Work)



HYPERPARAMETERS: epochs = 4000, **batch_size** = 64, **learning_rate** = 1E-07

FIGURE 147: SG architecture - 3 dense layers thickness not considered. (Own Work)

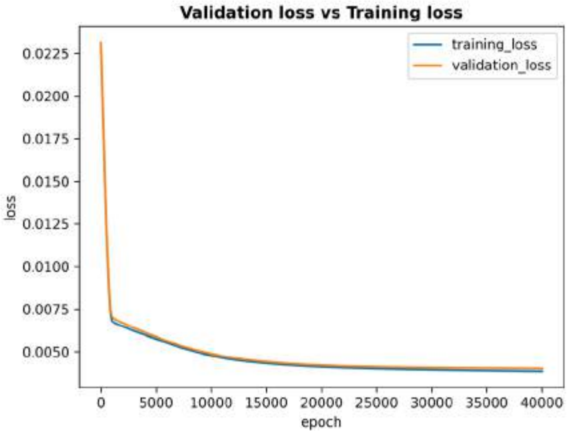


FIGURE 148: Validation vs Training loss - 3 dense layers -thickness not considered. (Own Work)

Reducing the layers further proved to be counterproductive and worsened the results.

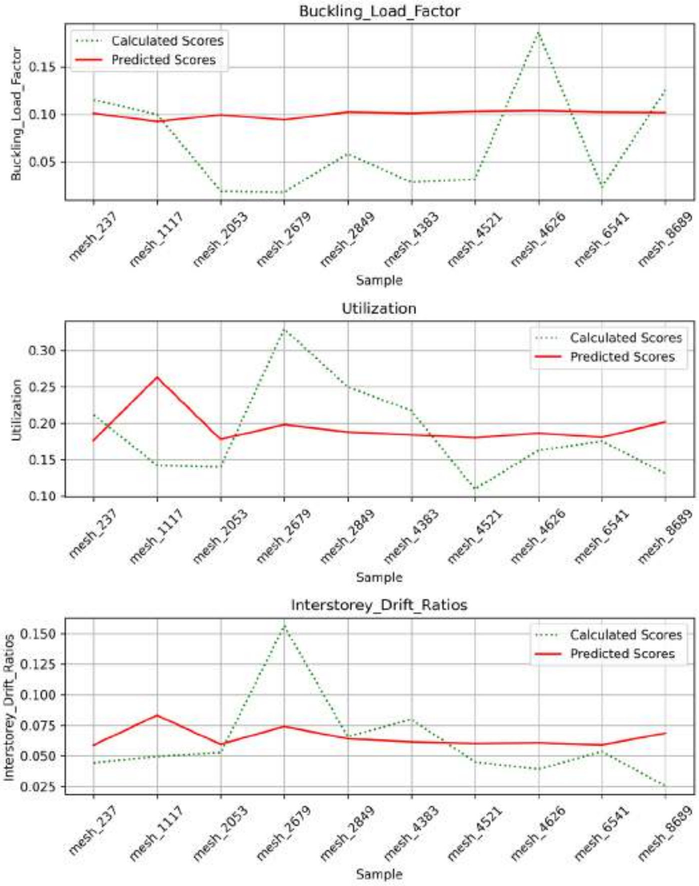
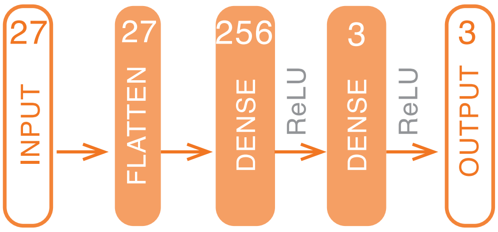


FIGURE 149: Ground Truth- Prediction 2 dense layers. (Own Work)



HYPERPARAMETERS: epochs = 4000, **batch_size** = 64, **learning_rate** = 1E-07

FIGURE 150: SG architecture - 2 dense layers thickness not considered. (Own Work)

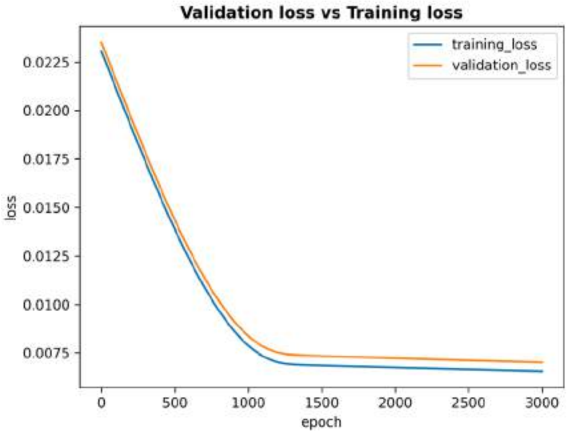
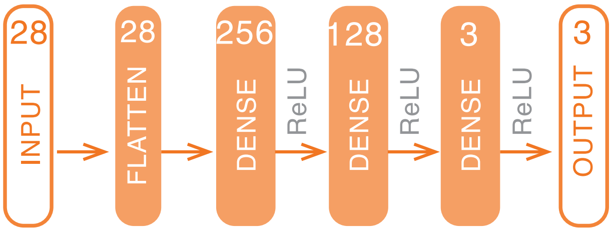


FIGURE 151: Validation vs Training loss - 2 dense layers -thickness not considered. (Own Work)

It can be seen that training has finished as convergence has been found shown in Figure 151. However, decreasing the layers further reduces the predictive capabilities by a great degree as shown in Figure 149.

Thickness Included: Normalization of Thickness along with the rest of the Force Densities

As seen in all cases, the Buckling Load Factor was not able to be predicted with appropriate accuracy without considering the thickness. By including this, the shape of the input changed from (n,27) to (n,28). Including the thickness as a feature presented different options of normalization.



HYPERPARAMETERS: epochs = 3000, **batch_size** = 128, **learning_rate** = 1E-06

FIGURE 152: SG architecture - 3 dense layers thickness included. (Own Work)

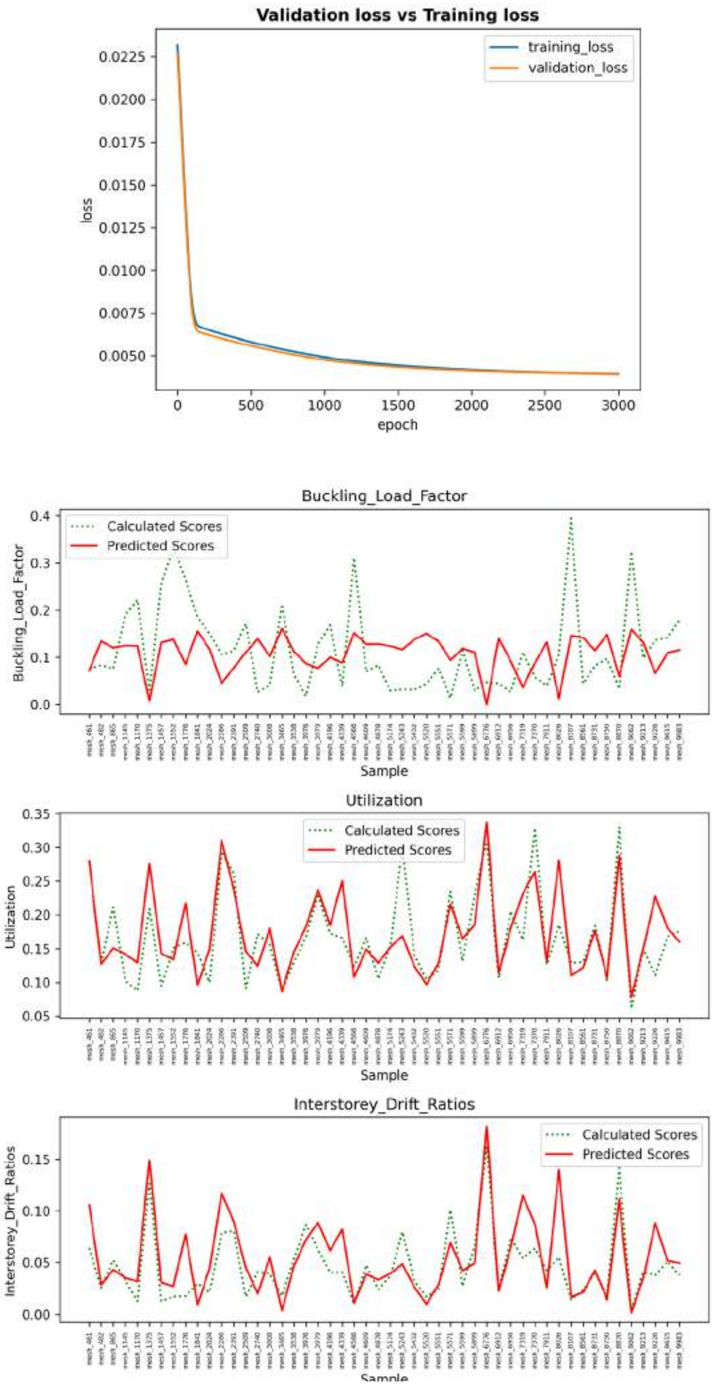


FIGURE 153: Validation vs Training loss (top) - Prediction vs Ground Truth (bottom). 3 dense layers. Thickness normalization along with force densities. (Own Work)

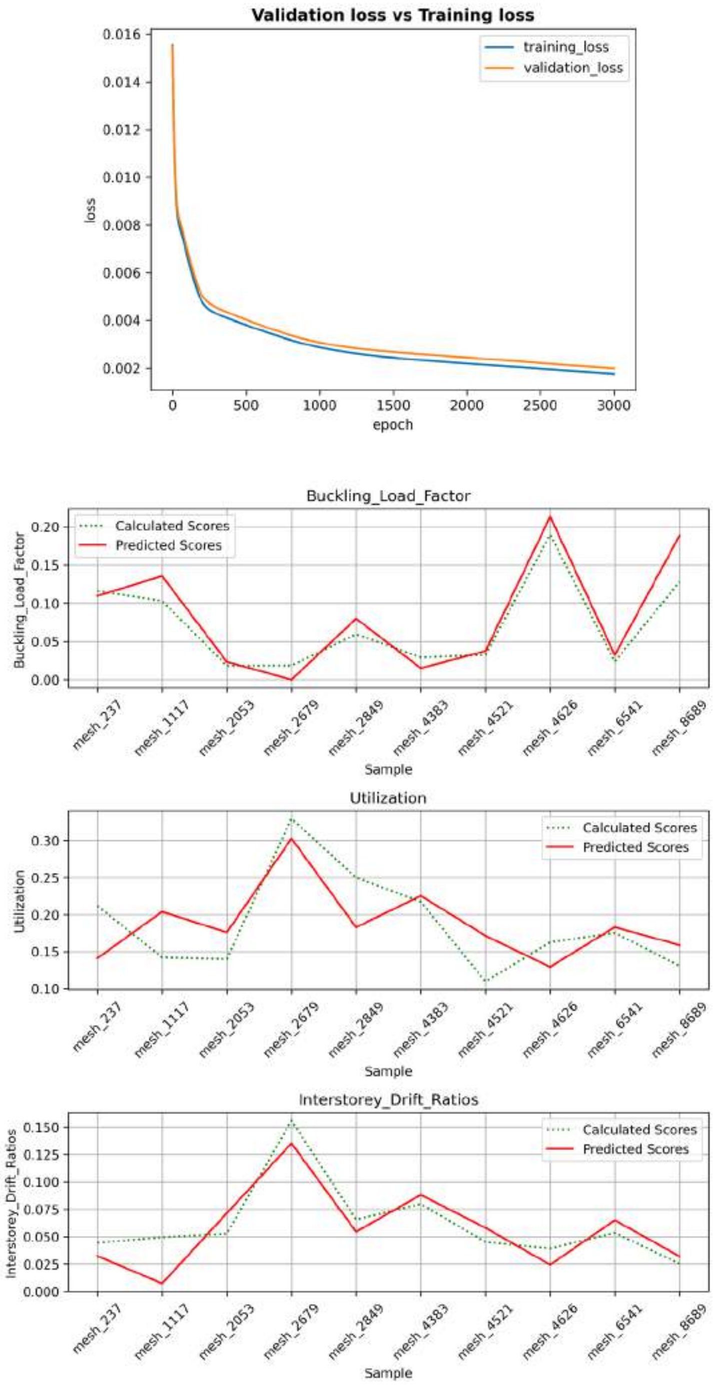


FIGURE 154: Validation vs Training loss (top) - Prediction vs Ground Truth (bottom). 3 dense layers. Thickness normalization independently. (Own Work)

The same architecture and hyperparameters were used to test the effect on normalization. Figure 153 shows the when normalization of thickness is done along with the rest of the parameters. The effect on prediction capabilities is very similar to when the same architecture was used without the inclusion of thickness in Figure 144 and 146 - the model fails to predict the Buckling Load Factor with any success but seems to give better results for Interstorey Drift Ratios and Utilization. However, when thickness was normalized independently, the model was able to learn how to predict the Buckling Load Factor. This was because the thicknesses have a maximum value of 0.095 whereas the maximum value of the rest of the dataset (the force densities) is 1038.8 whilst the minimum is 0.1. If the thickness is normalized along with the rest of the other force densities it's weight reduces by the factor of 10934. This makes all values of thickness smaller than the minimum value of force densities. Therefore, there is little to no effect on the loss function. In contrast when the thickness is normalized separately, it assigns proportionately much higher value to the value of thickness which is why there is such a remarkable increase in buckling accuracy. This can be assessed by comparing the mean and standard deviation of the force densities against that of the thickness as shown in Table 17. After normalization, the mean of thickness values is 2.08e4% greater that than that of the mean of force densities.

DATASET: randomized		
Statistical quantities	Thickness	Force Densities
Before normalization		
Maximum	0.0950	1038.8000
Minimum	0.0350	0.1000
Mean (μ)	0.0635	3.3485
Standard deviation (σ)	0.0245	10.1136
After independant normalization		
Maximum	1.0000	1.0000
Minimum	0.3684	9.63E-5
Mean (μ)	0.6683	0.0032
Standard deviation (σ)	0.2583	0.0097

TABLE 17: Statistical quantities of Dataset features for SG. (Own Work)

7.2.2 Seismic Performance: 3 Separate Surrogate Models

After testing different architectures, a different approach was tried where each of the performance metric divided into the a single output tensor from a separate surrogate model. This method aimed at eliminating the effect of variable weights of each of the three losses on the overall loss - which may allow the model to learn to predict some metrics and not predict others. This would be done as because the overall loss would not need to aggregated.

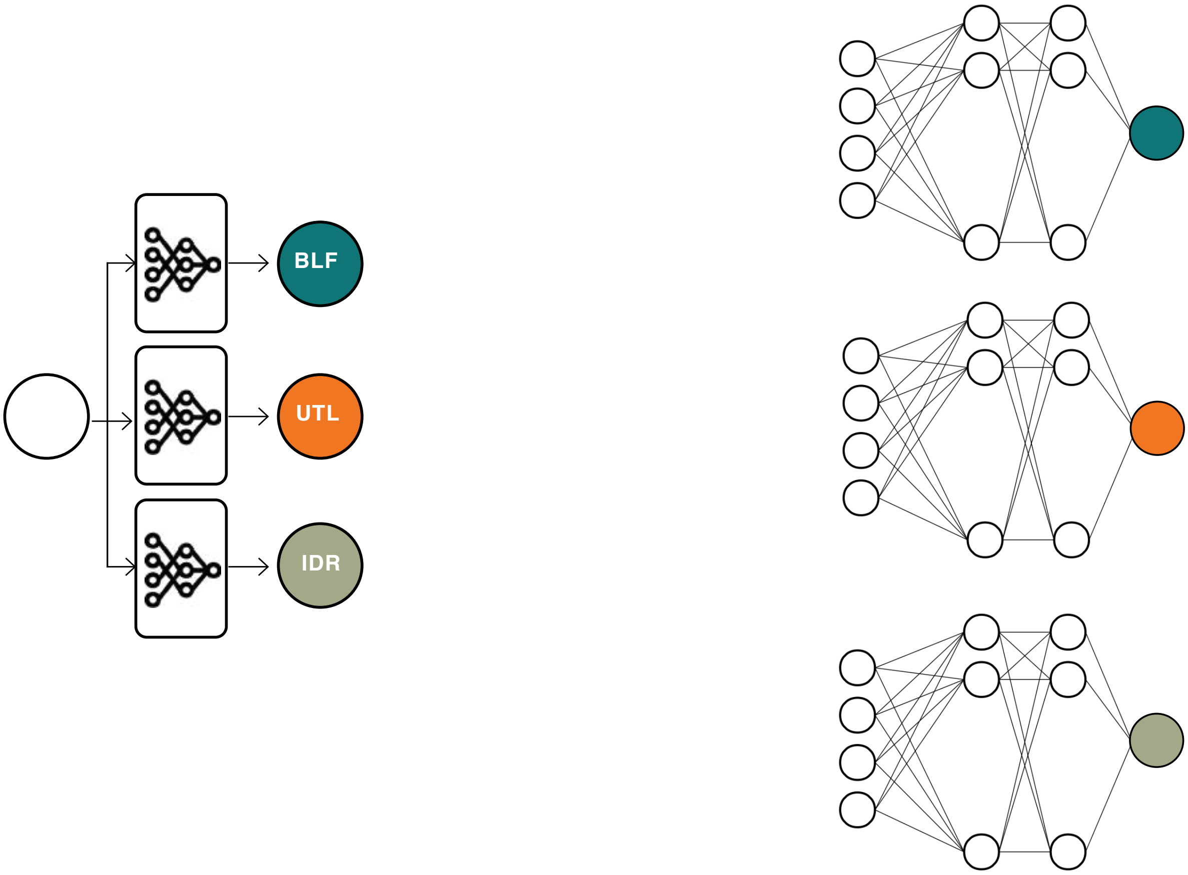
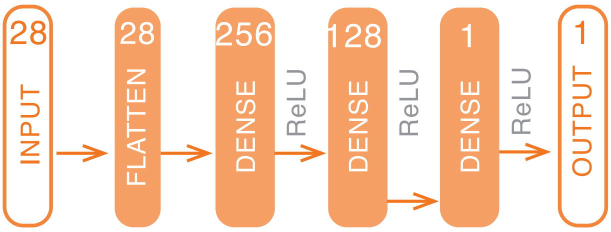


FIGURE 155: Three surrogate model. inspired by the work of Sterrenberg (2023). (Own Work)

As thickness was a parameter that had an effect on performance metrics, all surrogate models took it into account as a feature - so the shape of all feature datasets was (n, 28). As a single tensor was output, the shape of all label datasets was (n,1).

Model 1: Buckling Load Factor

Several simulations were carried out. The best performing one is highlighted below.



HYPERPARAMETERS: epochs = 2000, **batch_size** = 256, **learning_rate** = 5E-06

FIGURE 156: best SG architecture for Buckling Load Factor.- thickness normalized independently. (Own Work)

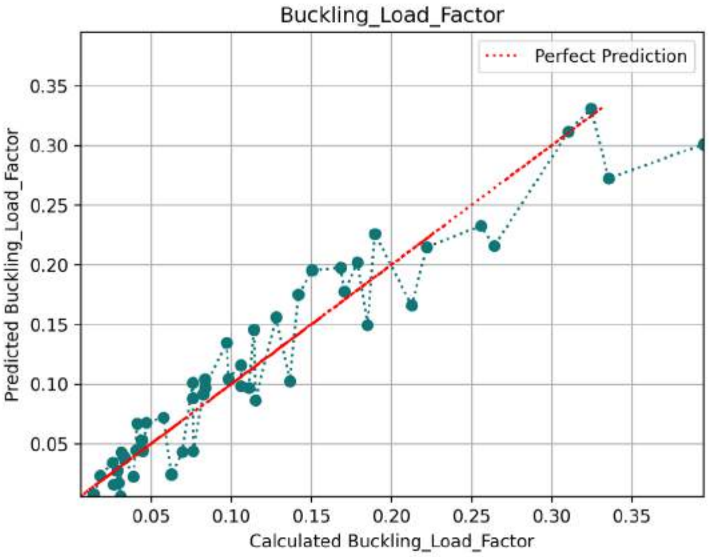


FIGURE 157: Prediction pattern on test data. (Own Work)

$$loss_1 = \frac{1}{n} \sum_{i=1}^{n} \left(y_{1,i} - y_{1,true,i}\right)^2$$

$$loss_2 = \frac{1}{n} \sum_{i=1}^{n} \left(y_{2,i} - y_{2,true,i}\right)^2$$

$$loss_3 = \frac{1}{n} \sum_{i=1}^{n} \left(y_{3,i} - y_{3,true,i}\right)^2$$

where,

$loss_1$ = Total Loss of the Buckling Load Factor

$loss_2$ = Total Loss of the Utilization

$loss_2$ = Tota Loss of the Interstorey Drift Ratio

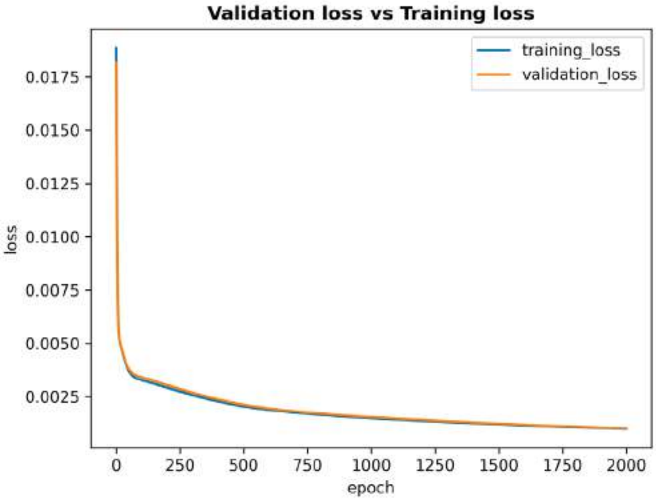


FIGURE 158: Validation vs Training loss. (Own Work)

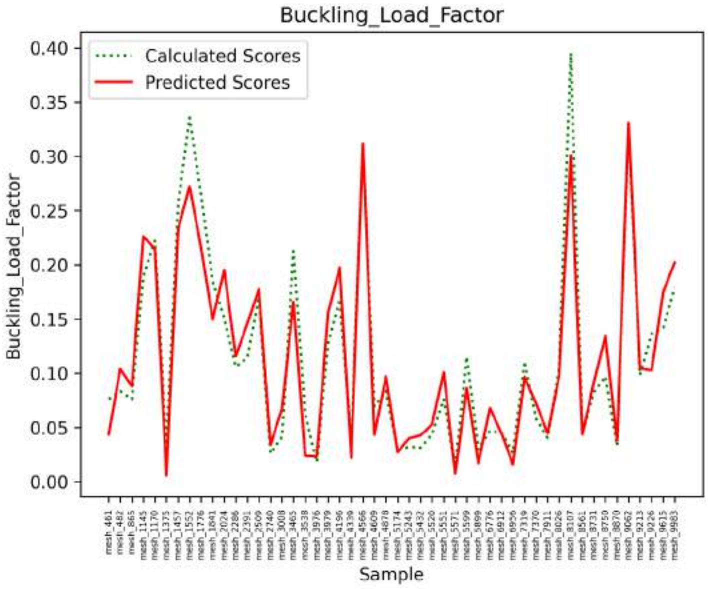


FIGURE 159: Prediction vs Ground Truth. (Own Work)

We can see from the validation and training loss curves, there is convergence. The relationship between the perfect model and the actual predictive model is expressed in Figure 157. We can qualitatively assess that the line of perfect performance passes through the actual predictive capability.

To make quantitative assessments of the performance, Normalized Root Mean Squared Error (NRMSE was used. While Root Mean Square Error (RMSE) is popular for assessing performance accuracy in similar regression tasks as a metric for comparing different predictive models, the absolute value of the RMSE is highly relative to the type of data and the dataset itself so it does not serve well as an indicator of how 'good' the model is to predict the ground truth.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_{,i} - \widehat{y_{,i}})^2}$$

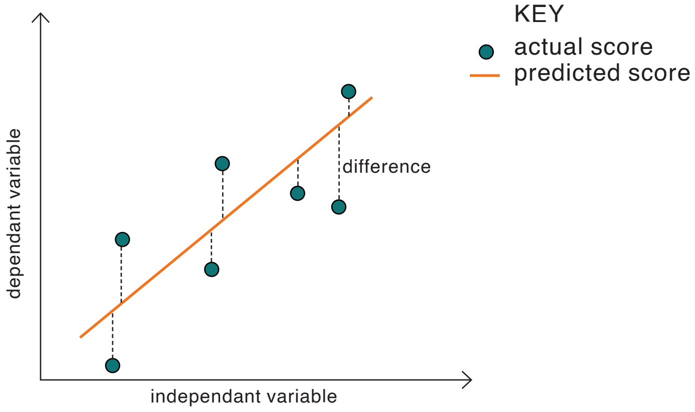


FIGURE 160: Root Mean Squared Error. (Own Work)

NRMSE is expressed as a fraction of the RMSE divided by the range of target variable. While in some instances the denominator is the mean instead, it is relevant to scenarios where the accuracy is to be calculated relative to the average value of the target variable. In our case, since we required the accuracy relative to the spread of values, the range was used. NMRSE is measured in percentages, with percentages close to 0 having high accuracy.

$$NRMSE = \frac{\sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_{,i} - \widehat{y_{,i}})^2}}{y_{max} - y_{min}}$$

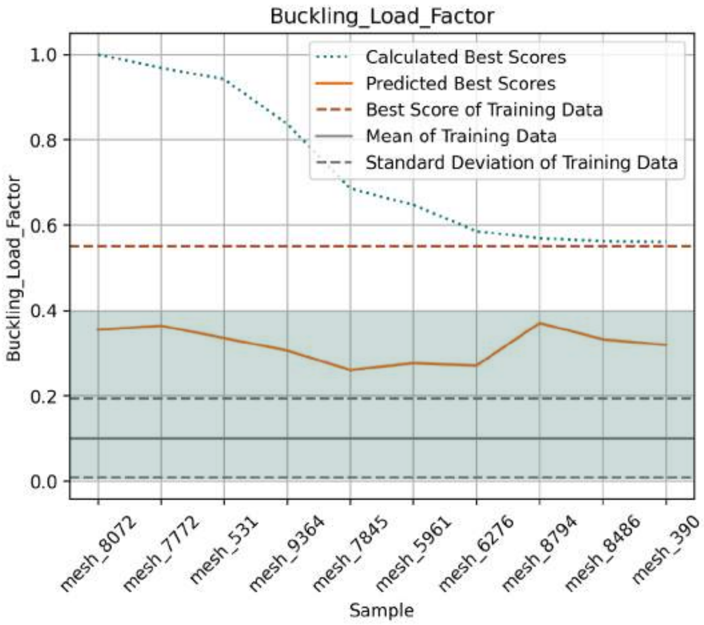
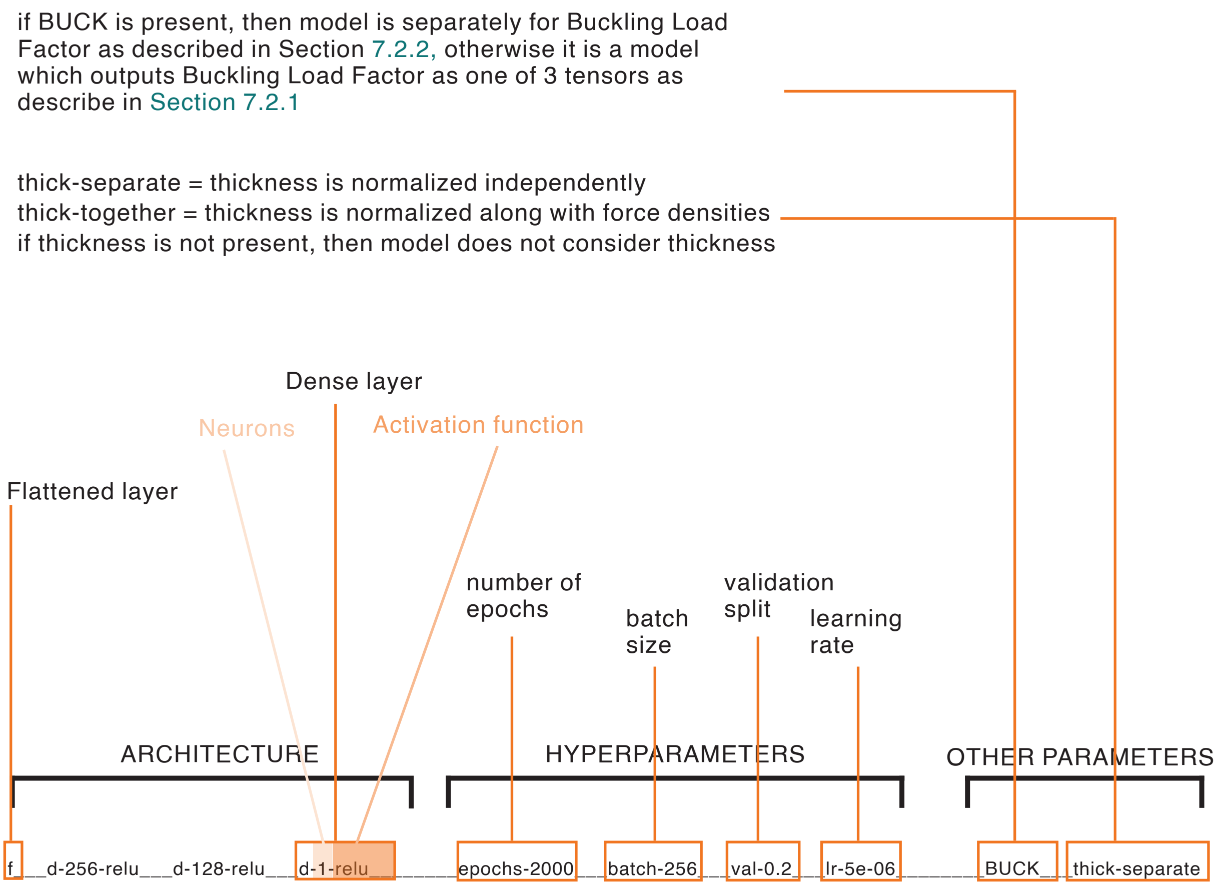


FIGURE 161: Prediction on 10 best samples of Buckling Load Factor. (Own Work)

The best performing model for Buckling Load Factor was found to have an NRMSE of 7.27% (RMSE = 3.01). It was found that in all hyperparameter combinations, the model with thickness normalized independently performed better than the one where the it was normalized with the rest of the force densities. The best performing model with thickness normalized along with force densities had a 14.25% absolute increase in NRMSE than the best performing one with thickness normalized independently. The results of the different types of models are shown in the summary on the next page.

As mentioned earlier, the 10 best performing meshes from each performance metric (including Buckling Load Factor) were excluded from the training dataset to monitor how the model predicted their results. It can be seen that the predicted scores lie below the values of the actual scores. This is expected as the training set had not seen any values that had performed as well so the mode could only generate a lower performance. The predicted performances are between 1 and 2 standard deviations above the mean of the training dataset.

This is a summary of the various models that had been made to calculate the Buckling Load Factor and their RMSE and NRMSE for comparison. The list is in the order of decreasing performance. Models which used 3 tensor outputs instead of 1 specifically for Buckling Load Factor are also included in terms of NRMSE performance for Buckling Load Factor.



if BUCK is present, then model is separately for Buckling Load Factor as described in Section 7.2.2, otherwise it is a model which outputs Buckling Load Factor as one of 3 tensors as describe in Section 7.2.1

thick-separate = thickness is normalized independently
thick-together = thickness is normalized along with force densities
if thickness is not present, then model does not consider thickness

1 f___d-256-relu___d-128-relu___d-1-relu________epochs-2000___batch-256___val-0.2___lr-5e-06________BUCK___thick-separate

NRMSE: 7.27 %

RMSE: 3.0113935776828527

----

2 f___d-256-relu___d-128-relu___d-3-relu________epochs-4000___batch-128___val-0.2___lr-1e-06________thick-separate

NRMSE: 13.4 %

RMSE: 2.4979500939334214

----

3 f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________BUCK___thick-separate

NRMSE: 14.06 %

RMSE: 2.6200409686651596

----

4 f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________thick-separate

NRMSE: 14.74 %

RMSE: 2.7471964734982413

----

5 f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06___thick-norm_with-only-one-loss

NRMSE: 15.73 %

RMSE: 2.930791694228252

----

6 f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-5e-06________BUCK___thick-separate

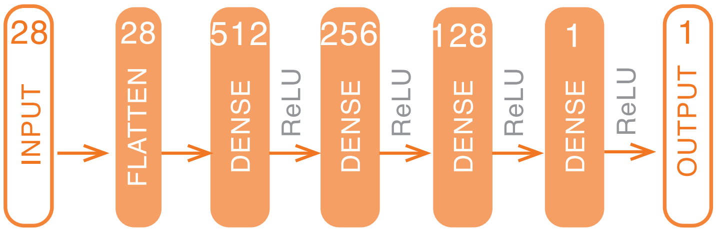
NRMSE: 17.41 %

RMSE: 3.2445617003847795

```
    ----------------------------------------------------------
7   f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-256___val-0.2___lr-5e-06________BUCK___thick-separate
    NRMSE: 17.75 %
    RMSE: 3.308732461290613
    ----------------------------------------------------------
8   f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________thick-separate
    NRMSE: 17.79 %
    RMSE: 3.3156644940893405
    ----------------------------------------------------------
9   f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-5e-06________BUCK___thick-together
    NRMSE: 21.51 %
    RMSE: 8.914716884731384
    ----------------------------------------------------------
10  f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-5e-06________BUCK___thick-together
    NRMSE: 34.19 %
    RMSE: 6.371296332688003
    ----------------------------------------------------------
11  f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________thick-together
    NRMSE: 36.6 %
    RMSE: 6.821276430654844
    ----------------------------------------------------------
12  f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________BUCK___thick-together
    NRMSE: 37.02 %
    RMSE: 6.899975407392418
    ----------------------------------------------------------
13  f___d-256-relu___d-128-relu___d-1-relu________epochs-1500___batch-256___val-0.2___lr-1e-06________BUCK___thick-together
    NRMSE: 37.02 %
    RMSE: 6.899975407392418
    ----------------------------------------------------------
14  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-together
    NRMSE: 37.02 %
    RMSE: 6.900158783783083
    ----------------------------------------------------------
15  f___d-256-relu___d-128-relu___d-3-relu________epochs-4000___batch-128___val-0.2___lr-1e-06________thick-together
    NRMSE: 37.33 %
    RMSE: 6.956856323841029
    ----------------------------------------------------------
16  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________thick-together
    NRMSE: 37.7 %
    RMSE: 7.026840288131792
    ----------------------------------------------------------
17  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-separate
    NRMSE: 53.22 %
    RMSE: 9.917899103284437
```

Model 2: Utilization

Several simulations were also carried out for the surrogate model predicting Utilization. The best one is highlighted below



HYPERPARAMETERS: epochs = 2000, **batch_size** = 256, **learning_rate** = 5E-06

FIGURE 162: best SG architecture for Utilization - thickness normalized independantly. (Own Work)

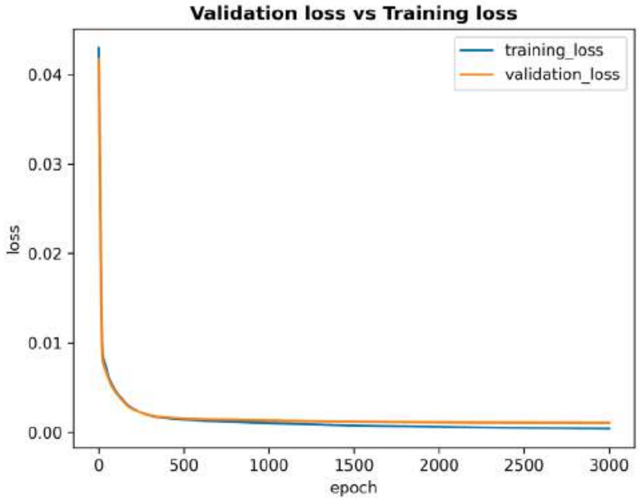


FIGURE 164: Validation vs Training loss. (Own Work)

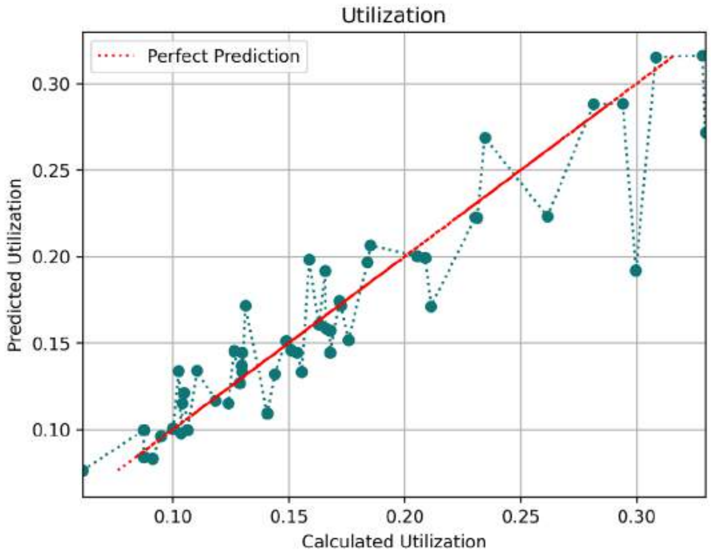


FIGURE 163: Prediction pattern on test data. (Own Work)

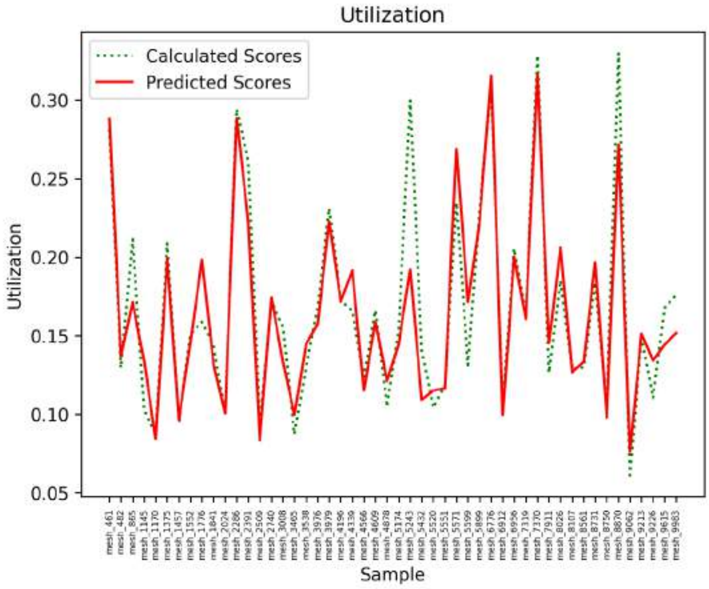


FIGURE 165: Prediction vs Ground Truth. (Own Work)

The training losses and validation losses indicate convergence. The model gives an NRMSE of 9.18% - a value that is higher than the error received during buckling. The model performs well on the best 10 samples of Utilization. The values are distributed below 1 standard deviation below the mean close to the best performing sample of the training set.

As there are 4 dense layers, simpler models were tested with lesser number of dense layers. The NRMSE increased 1.41% when the 64 neuron layer was dropped. However, there was significant reduction in performance for the 10 best performing samples. The values are distributed below the mean within half a standard deviation. One sample was detected (mesh 7772) with the best accuracy detecting the normalized value of 0.0610 with a predicted value 0.0487. This is close to the best score of the training data.

Besides architectures with 4 layers and above, it is interesting to note that models that predicted the best performing meshes did not necessary give the best predictions on the 50 unseen samples. This can be visualized in the difference between the predicted best scores (orange line) in Figure 167 and Figure 168. The architecture for Figure 168 shows better prediction for the 50 best samples but worse predictions for the rest of the samples.

Increase in the number of neurons in each layer showed reduction in the NRMSE. This can be seen when moving from 256-128-1 to 512-256-1. The absolute reduction is 1.05%. A further 1.18% absolute reduction is seen when the number of layers are increased from 3 dense layers to 4 dense layers of 512-256-1 to 512-256-128-1. However, this was only the case when normalization of thickness was done independently. When thickness was normalized along with force densities, there was minimal change - a 0.06% absolute increase with increasing the number of layers.

In terms of activation functions, ReLU produced better results than sigmoid for the same hyperparameters and architecture. There is 1.08% absolute reduction for the same architecture and hyperparameters.

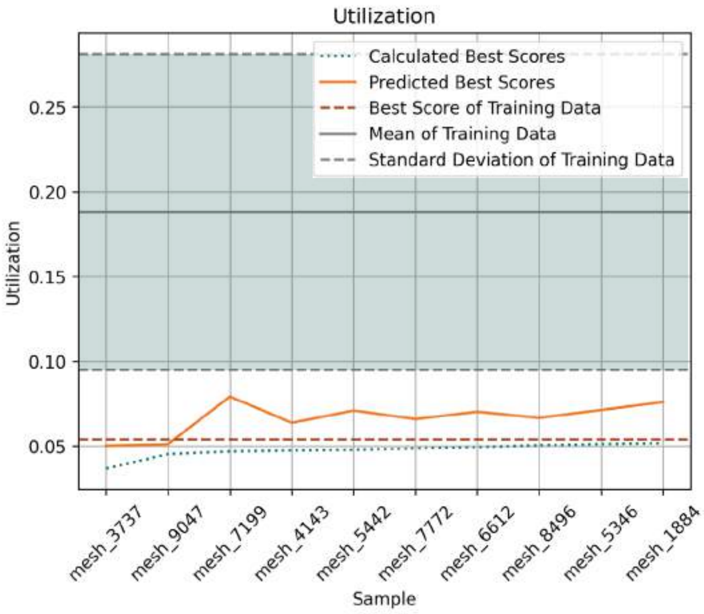


FIGURE 166: Prediction on 10 best samples of Utilization for best performing SG for utilization. (Own Work)

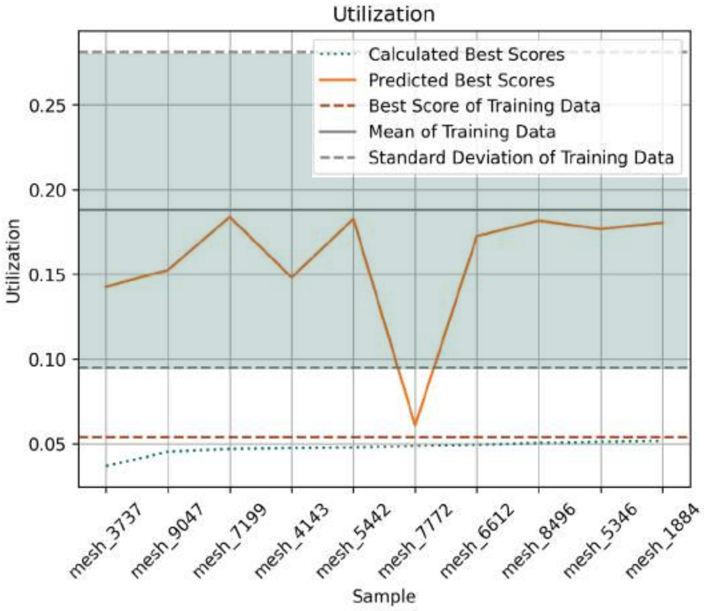


FIGURE 167: Prediction on 10 best samples of Utilization for 3 dense layers - thickness normalized independently. (Own Work)

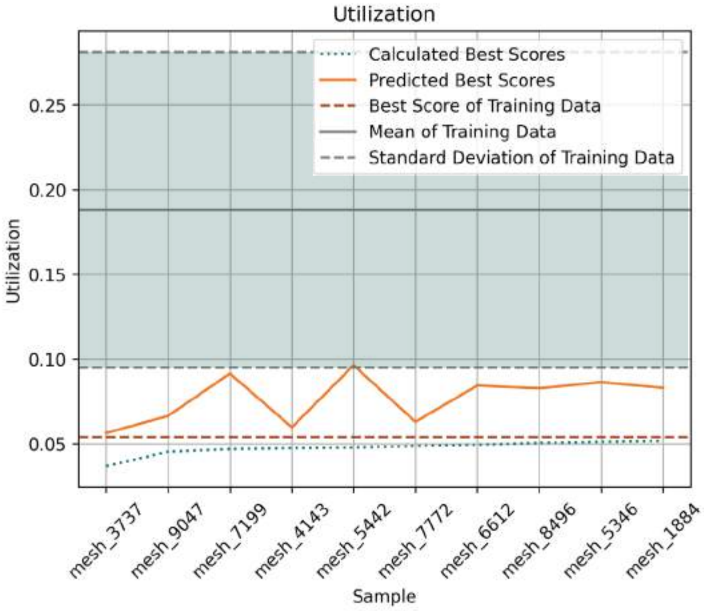
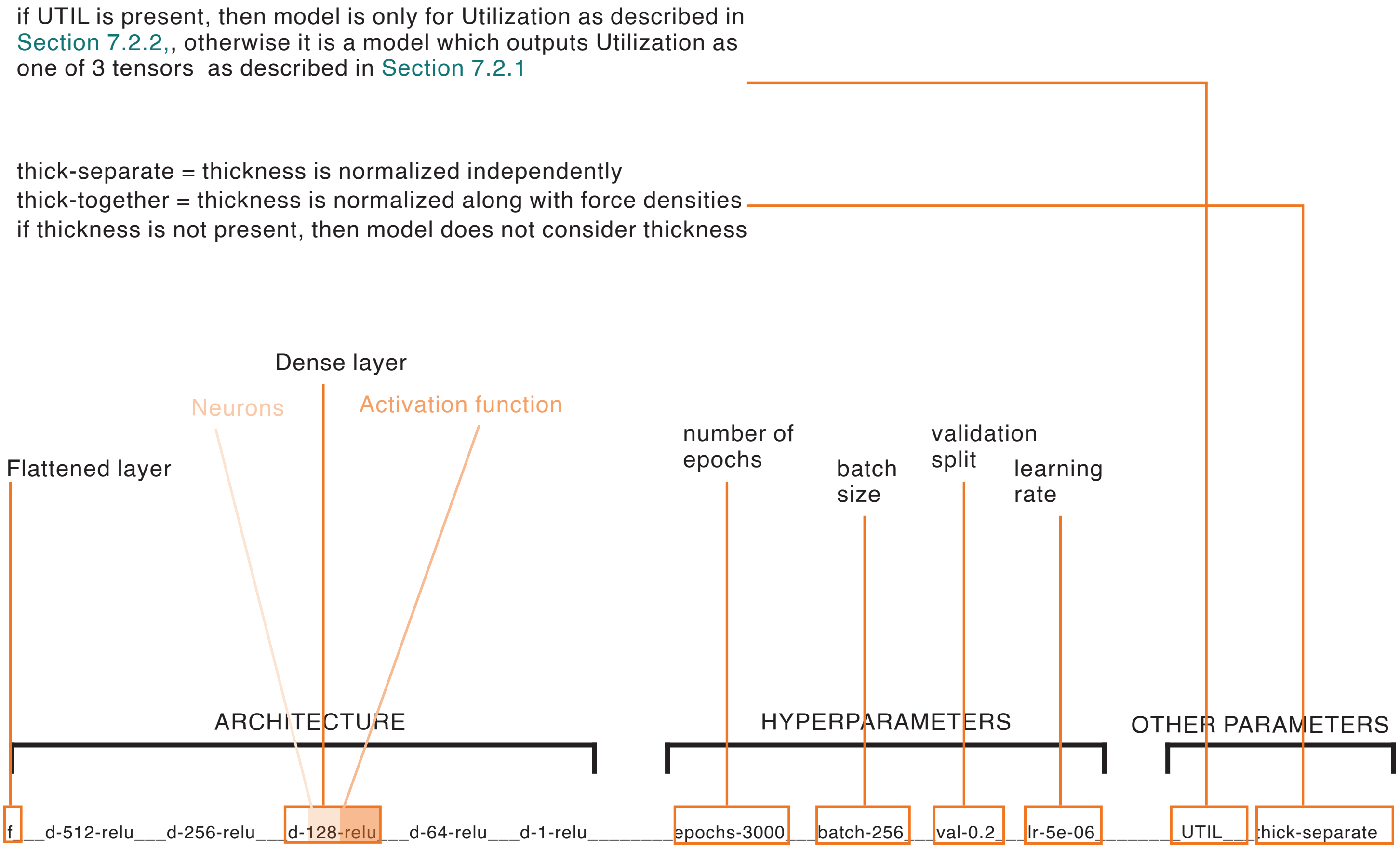


FIGURE 168: Prediction on 10 best samples of Utilization for 3 dense layers - thickness normalized along with force densities. (Own Work)

This is a summary of the various models that had been made to calculate Utilization and their RMSE and NRMSE for comparison. The list is in the order of decreasing performance. Models which used 3 tensor outputs instead of 1 specifically for Utilization are also included in terms of NRMSE performance for Utilization.



if UTIL is present, then model is only for Utilization as described in Section 7.2.2,, otherwise it is a model which outputs Utilization as one of 3 tensors as described in Section 7.2.1

thick-separate = thickness is normalized independently
thick-together = thickness is normalized along with force densities
if thickness is not present, then model does not consider thickness

1 f___d-512-relu___d-256-relu___d-128-relu___d-64-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate

NRMSE: 9.18 %

RMSE: 1.234909477260432

----

2 f___d-1024-relu___d-512-relu___d-256-relu___d-128-relu___d-64-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate

NRMSE: 9.22 %

RMSE: 1.2398733225002645

----

3 f___d-512-relu___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate

NRMSE: 10.59 %

RMSE: 1.4234490793780434

----

4 f___d-512-relu___d-256-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate

NRMSE: 11.77 %

RMSE: 1.5830211050191585

----

5 f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate

NRMSE: 12.82 %

RMSE: 1.7233268529980592

----

6 f___d-256-relu___d-128-relu___d-1-sigmoid________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate

NRMSE: 13.9 %

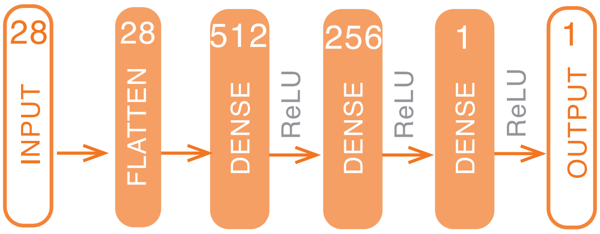
RMSE: 1.868948555884073

```
    ---------------------------------------------------------
7   f___d-512-relu___d-256-relu___d-128-relu___d-64-relu___d-1-relu________epochs-600___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate
    NRMSE: 14.31 %
    RMSE: 1.9239409368081997
    ---------------------------------------------------------
8   f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-together
    NRMSE: 14.37 %
    RMSE: 1.9325439599071121
    ---------------------------------------------------------
9   f___d-512-relu___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-together
    NRMSE: 14.43 %
    RMSE: 1.9403529014796959
    ---------------------------------------------------------
10  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-separate
    NRMSE: 15.06 %
    RMSE: 1.6570501291747386
    ---------------------------------------------------------
11  f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-5e-06________UTIL___thick-separate
    NRMSE: 15.92 %
    RMSE: 1.7511587561866122
    ---------------------------------------------------------
12  f___d-256-relu___d-128-sigmoid___d-1-sigmoid________epochs-3000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate
    NRMSE: 17.02 %
    RMSE: 2.288209422814802
    ---------------------------------------------------------
13  f___d-256-relu___d-128-relu___d-3-relu________epochs-4000___batch-128___val-0.2___lr-1e-06________thick-separate
    NRMSE: 19.26 %
    RMSE: 2.119297175430908
    ---------------------------------------------------------
14  f___d-256-relu___d-1-relu________epochs-1000___batch-256___val-0.2___lr-5e-06________UTIL___thick-separate
    NRMSE: 19.37 %
    RMSE: 2.604651135792025
    ---------------------------------------------------------
15  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________thick-separate
    NRMSE: 20.92 %
    RMSE: 2.301628917187026
    ---------------------------------------------------------
16  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-together
    NRMSE: 23.83 %
    RMSE: 2.622125577645853
    ---------------------------------------------------------
17  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________thick-together
    NRMSE: 24.81 %
    RMSE: 2.7295861049831456
    ---------------------------------------------------------
18  f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-5e-06________UTIL___thick-together
    NRMSE: 25.79 %
    RMSE: 2.8372870707384292
```

```
   ---------------------------------------------------------
19 f___d-256-relu___d-128-relu___d-3-relu________epochs-4000___batch-128___val-0.2___lr-1e-06________thick-together
   NRMSE: 26.62 %
   RMSE: 2.9293060658312426
   ---------------------------------------------------------
20 f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________thick-separate
   NRMSE: 27.6 %
   RMSE: 3.037264517987134
   ---------------------------------------------------------
21 f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-1e-06___UTIL___thick-norm
   NRMSE: 28.4 %
   RMSE: 3.1244642731656778
   ---------------------------------------------------------
22 f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________thick-together
   NRMSE: 28.54 %
   RMSE: 3.1400811607284513
   ---------------------------------------------------------
23 f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06___thick-norm_with-only-one-loss
   NRMSE: 30.36 %
   RMSE: 3.3402097374194626
   ---------------------------------------------------------
24 f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-1e-06___UTIL
   NRMSE: 30.61 %
   RMSE: 3.367366692057121
```

Model 3: Interstorey Drift Ratios

Several simulations were also carried out for the surrogate model predicting Interstorey Drift Ratios. The best result is shown below.



HYPERPARAMETERS: epochs = 3000, **batch_size** = 256, **learning_rate** = 5E-06

FIGURE 169: best SG architecture for Interstorey Drift Ratios - thickness normalized independently. (Own Work)

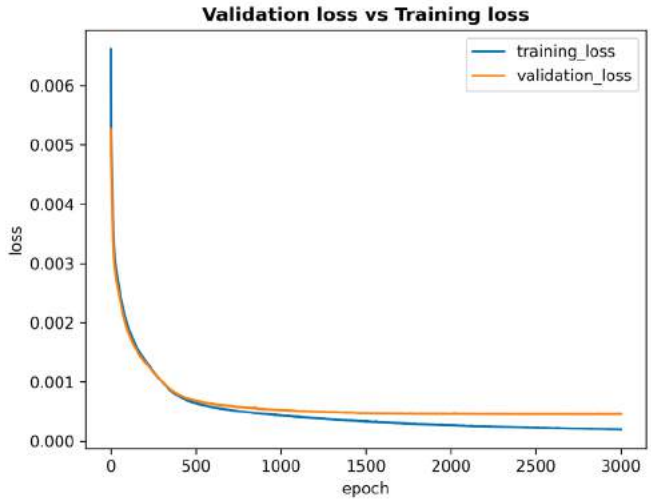


FIGURE 171: Validation vs Training loss. (Own Work)

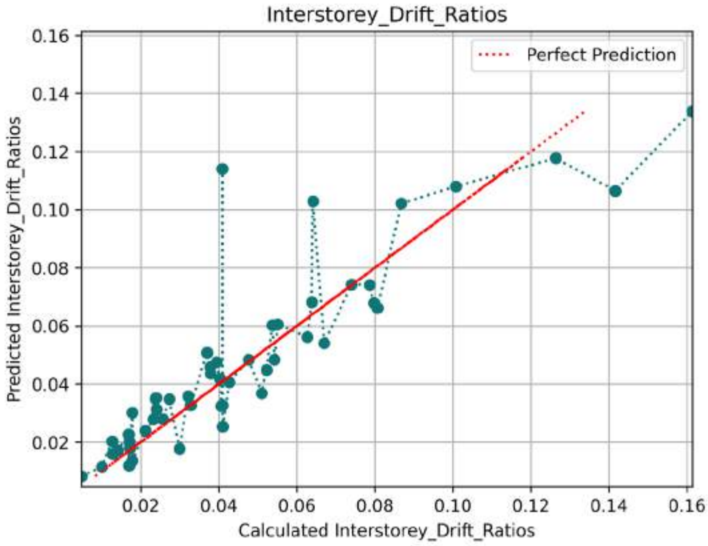


FIGURE 170: Prediction pattern on test data. (Own Work)

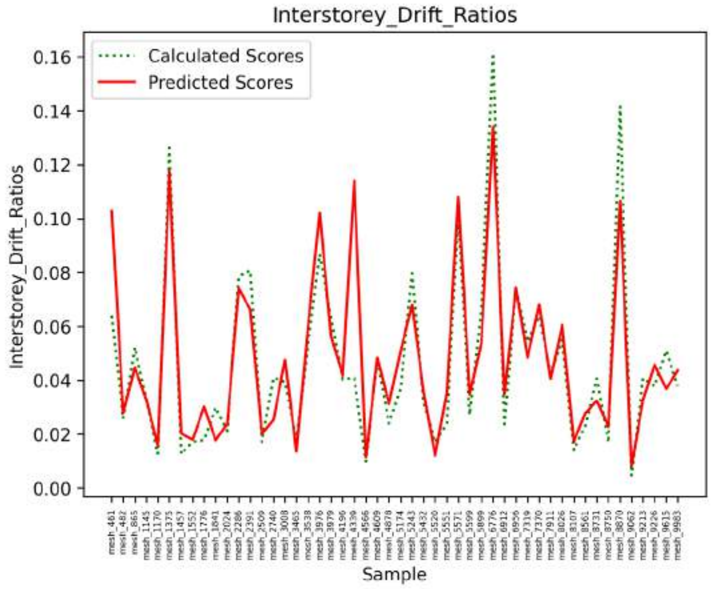


FIGURE 172: Prediction vs Ground Truth. (Own Work)

It can be seen Figure X how the predicted values lie close to the perfect prediction line besides some outliers. A total NRMSE was found to be 9.81% in this case. By increasing the number of neurons from 256-128-1 to 512-256-1, the NRMSE reduced by an absolute percentage of 2.1%. Predicting the 10 best samples in Interstorey Drift not seen during training was challenging as the 10 best samples including the best sample during training had actual values close to zero. It was unexpected to see 60% of the predicted values fall below the best performing sample

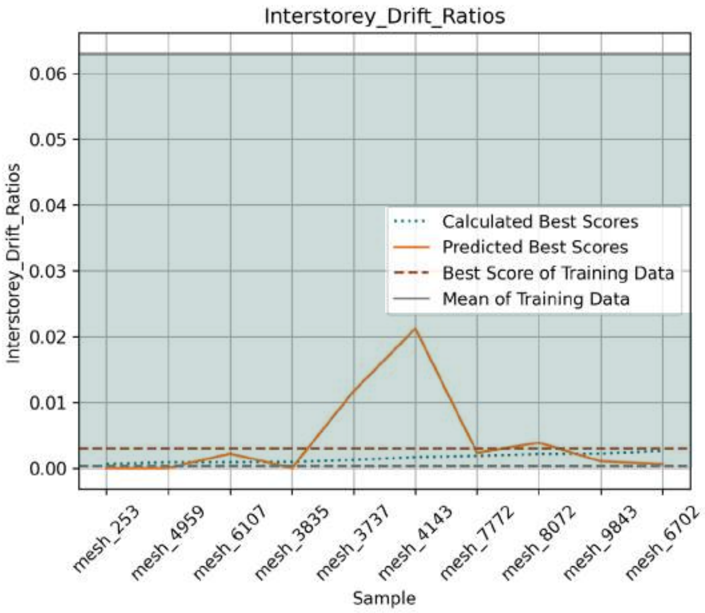


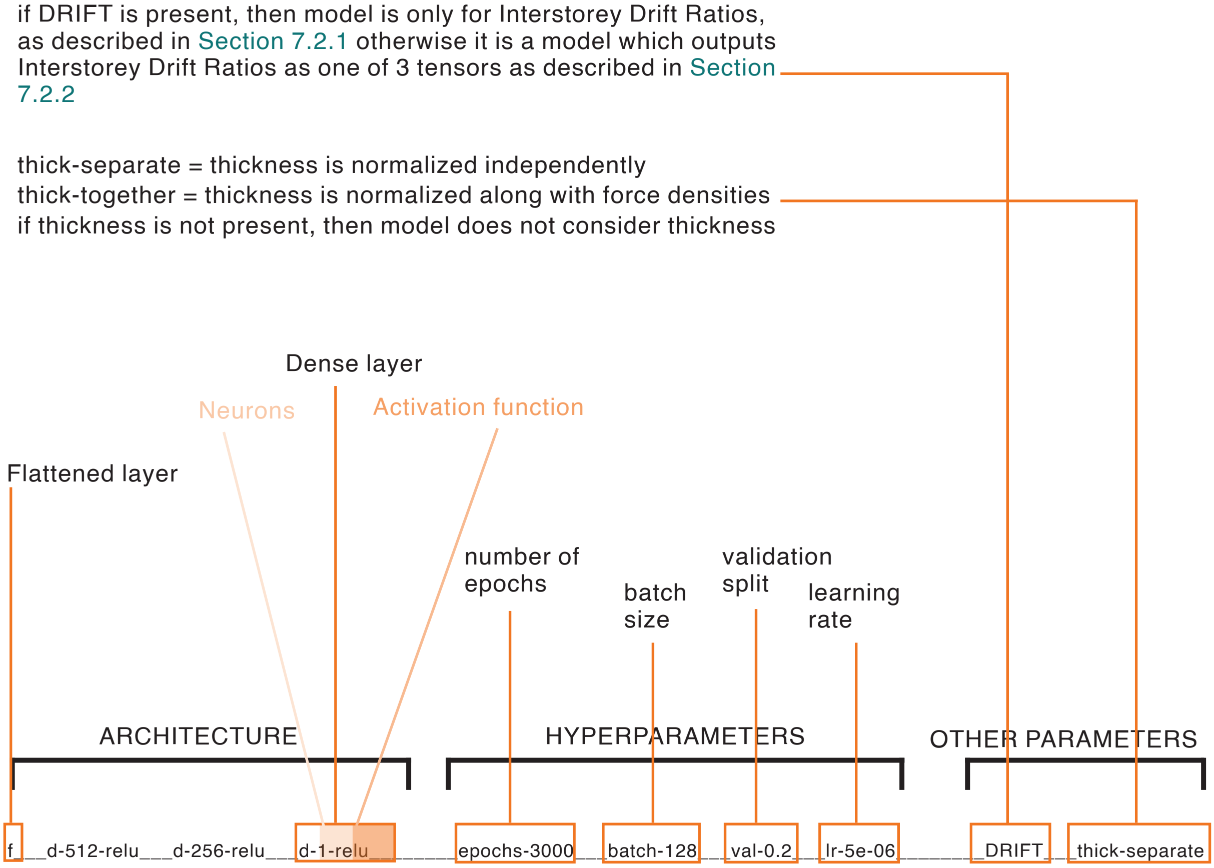
FIGURE 173: Prediction on 10 best samples of Utilization for best performing SG for Interstorey Drift Ratios. (Own Work)

during training even though the model had no values below that. 1 out of those 6 values was non zero. Therefore, it would not be fair to assess the model as performing poorly on the best values as the best values themselves were close to zero.

As was the case with Buckling Load Factor and Utilization, the best performing models were those with thickness normalized independent of the force densities. Using just ReLU as the activation functions for all dense layers had a better effect on the performance as using sigmoid on the model increased the absolute NRMSE by 1.57%.

It was also interesting to note that increasing the number of dense layers from 3 to 4 reduced performance by an absolute percentage of 1.19%. Moreover, reducing the number of layers from 3 to 2 (512-256-1 to 512-1) caused a 1.76% absolute reduction. It was notable that the 512-1 model performed better than 88.46% of all models including the 256-128-1. This shows that increasing the number of neurons had a much stronger effect than increasing the number of layers on the outcome. However, there is a threshold beyond which performance starts to flatten out and then decrease so the increase in the size of the NN is not worth the computational cost. For instance the two layered NN 512-1 performed better than 1024-1.

This is a summary of the various models that had been made to calculate Interstorey Drift Ratios and their RMSE and NRMSE for comparison. The list is in the order of decreasing performance. Models which used 3 tensor outputs instead of 1 specifically for Interstorey Drift Ratios are also included in terms of NRMSE performance for Interstorey Drift Ratios.


if DRIFT is present, then model is only for Interstorey Drift Ratios, as described in Section 7.2.1 otherwise it is a model which outputs Interstorey Drift Ratios as one of 3 tensors as described in Section 7.2.2

thick-separate = thickness is normalized independently
thick-together = thickness is normalized along with force densities
if thickness is not present, then model does not consider thickness

1 f___d-512-relu___d-256-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-separate

NRMSE: 9.81 %

RMSE: 0.0009780667564782254

----

2 f___d-512-relu___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-separate

NRMSE: 11.0 %

RMSE: 0.0010974877328150067

----

3 f___d-512-relu___d-256-relu___d-1-sigmoid________epochs-3000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-separate

NRMSE: 11.38 %

RMSE: 0.0011351569237790884

----

4 f___d-512-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-separate

NRMSE: 11.57 %

RMSE: 0.0011536630079410665

----

5 f___d-1024-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-separate

NRMSE: 11.89 %

RMSE: 0.0011859609732852864

----

6 f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________DRIFT___thick-separate

NRMSE: 11.93 %

RMSE: 0.0011894115681266308

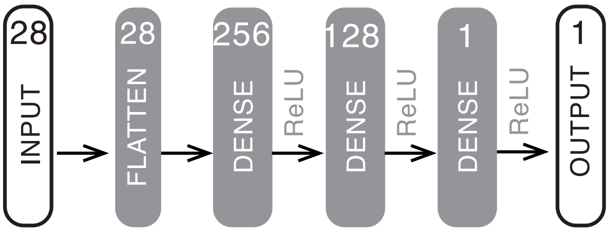
```
    ---------------------------------------------------------
7   f___d-256-relu___d-128-relu___d-3-relu________epochs-4000___batch-128___val-0.2___lr-1e-06________thick-together
    NRMSE: 12.03 %
    RMSE: 0.0009996250650747952
    ---------------------------------------------------------
8   f___d-256-relu___d-128-relu___d-1-relu________epochs-6000___batch-128___val-0.2___lr-1e-06________DRIFT___thick-separate
    NRMSE: 12.47 %
    RMSE: 0.0012435723003128211
    ---------------------------------------------------------
9   f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-together
    NRMSE: 12.9 %
    RMSE: 0.001071398708132039
    ---------------------------------------------------------
10  f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________DRIFT___thick-separate
    NRMSE: 13.26 %
    RMSE: 0.001322514377068918
    ---------------------------------------------------------
11  f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-separate
    NRMSE: 13.62 %
    RMSE: 0.001131338065472033
    ---------------------------------------------------------
12  f___d-256-relu___d-128-relu___d-1-relu________epochs-1000___batch-256___val-0.2___lr-5e-06________DRIFT___thick-separate
    NRMSE: 13.74 %
    RMSE: 0.0011412395446222328
    ---------------------------------------------------------
13  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________thick-separate
    NRMSE: 14.26 %
    RMSE: 0.0011848487975821562
    ---------------------------------------------------------
14  f___d-512-relu___d-256-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-together
    NRMSE: 14.89 %
    RMSE: 0.0014854920337636825
    ---------------------------------------------------------
15  f___d-512-relu___d-256-relu___d-3-relu________epochs-3000___batch-512___val-0.2___lr-1e-06________thick-together
    NRMSE: 15.43 %
    RMSE: 0.0015391005433508082
    ---------------------------------------------------------
16  f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________DRIFT___thick-together
    NRMSE: 15.7 %
    RMSE: 0.0015660532326641982
    ---------------------------------------------------------
17  f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-together
    NRMSE: 15.72 %
    RMSE: 0.001567525906330497
    ---------------------------------------------------------
18  d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________thick-together
    NRMSE: 15.89 %
    RMSE: 0.0015846452463676509
```

```
   ---------------------------------------------------------
19 f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________DRIFT___thick-together
   NRMSE: 16.19 %
   RMSE: 0.001614825566051438
   ---------------------------------------------------------
20 f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-128___val-0.2___lr-1e-06________thick-together
   NRMSE: 16.31 %
   RMSE: 0.0016267079440805666
   ---------------------------------------------------------
21 f___d-256-relu___d-128-relu___d-1-relu________epochs-3000___batch-128___val-0.2___lr-5e-06________DRIFT___thick-together
   NRMSE: 16.35 %
   RMSE: 0.0016307541455895754
   ---------------------------------------------------------
22 f___d-512-relu___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-together
   NRMSE: 17.28 %
   RMSE: 0.001723631356911795
   ---------------------------------------------------------
23 f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________thick-together
   NRMSE: 20.06 %
   RMSE: 0.001666860115000885
   ---------------------------------------------------------
24 f___d-256-relu___d-128-relu___d-3-relu________epochs-1000___batch-128___val-0.2___lr-1e-06________thick-separate
   NRMSE: 21.13 %
   RMSE: 0.0017555871659895369
   ---------------------------------------------------------
25 f___d-256-relu___d-128-relu___d-3-relu________epochs-3000___batch-256___val-0.2___lr-5e-06________thick-separate
   NRMSE: 53.77 %
   RMSE: 0.004467771333400297
   ---------------------------------------------------------
26 f___d-256-relu___d-128-relu___d-3-relu________epochs-4000___batch-128___val-0.2___lr-1e-06________thick-separate
   NRMSE: 53.77 %
   RMSE: 0.004467771333400297
```

7.2.3 Vault Properties: Separate Surrogate Models

Height

As mentioned earlier, in order to retrieve samples of a particular height, the VAE was conditioned with heights. However, since the output of the VAE was feature data comprising of force densities and thicknesses, another prediction model was needed to validate whether the new samples were indeed of the desired height as specified in the condition. For this reason, a surrogate model was created to predict the height. This would also be used for regulating the height during gradient descent in case a particular height of the final optimized vault was required.



HYPERPARAMETERS: epochs = 6000, **batch_size** = 128, **learning_rate** = 5E-06

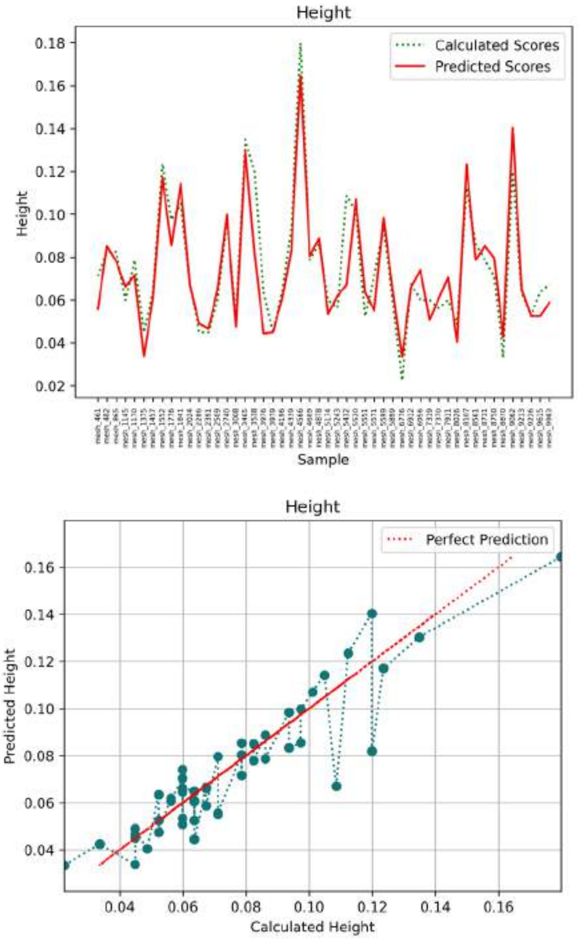
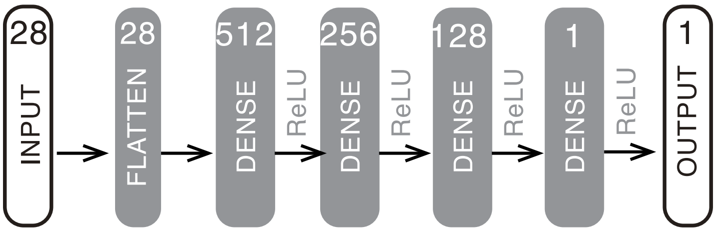


FIGURE 174: Best Height surrogate model a) architecture, b) Prediction vs Ground Truth, c) Prediction pattern on test data. (Own Work)

Mass

The goal of optimization was to improve performance (Buckling Load Factor / Utilization / Interstorey Drift Ratio) while having the lowest amount of material possible. The condition for minimizing material was included by minimizing mass. In order to predict the mass of each sample, a surrogate model was created. If the condition of mass was not included, the samples would result in minimizing the force densities creating very high and thus, very massive vaults. Therefore, a condition minimizing mass would provide regulation.



HYPERPARAMETERS: epochs = 3000, **batch_size** = 256, **learning_rate** = 5E-06

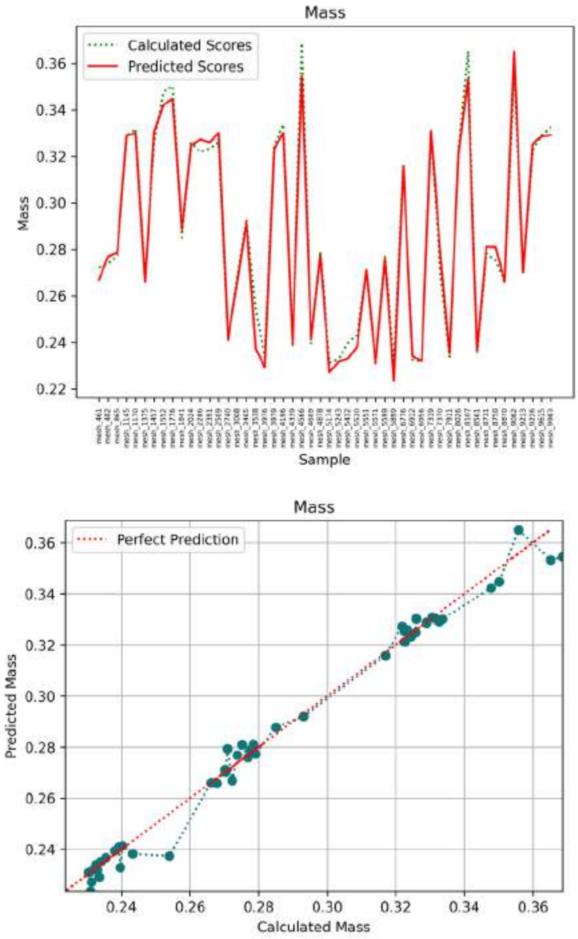


FIGURE 175: Best Height surrogate model a) architecture, b) Prediction vs Ground Truth, c) Prediction pattern on test data. (Own Work)

7.2.4 Conclusion

To summarize, the strategy to divide the 1 surrogate model with 3 tensor outputs into 3 separate surrogate models worked for all three performance metrics. Though larger neural networks with 5 and 6 dense layers performed slightly better, one with 4 dense layers was chosen (performing the 3rd highest in all Utilization models).

The best separate models gave these NRMSE (Figure 177):

- Buckling Load Factor: 7.27%
- Utilization: 9.18%
- Interstorey Drift Ratio: 9.81%

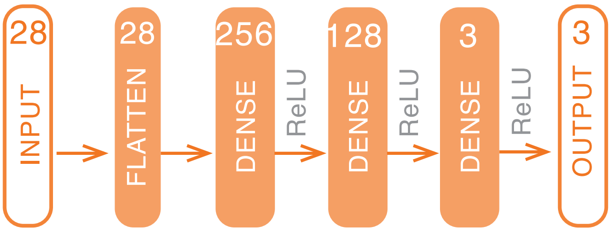
In contrast, the best model which had 3 tensor outputs gave the following result. It is shown in Figure 176:

- Buckling Load Factor: 14.74%
- Utilization: 20.92%
- Interstorey Drift Ratio: 14.26%

By having separate models, the performance metrics decreased by 50.7%, 56.1%, and 31.2% respectively. Thus, there is a significant increase in performance for all three metrics.

Moreover, normalization of thickness independent to the force densities and use of ReLU as the activation function also produced the best results in all three cases. The following summarizes the architectures an hyperparameters of the chosen surrogate models to be used in the optimization. Although each metric has a different architecture for its own optimal performance, similarities arose during hyperparameter tuning. Though other learning rates were tested such as 1E-05, 1E-06 and 1E-07, the most optimal one found was 5E-06 in all three cases with a batch size of 256.

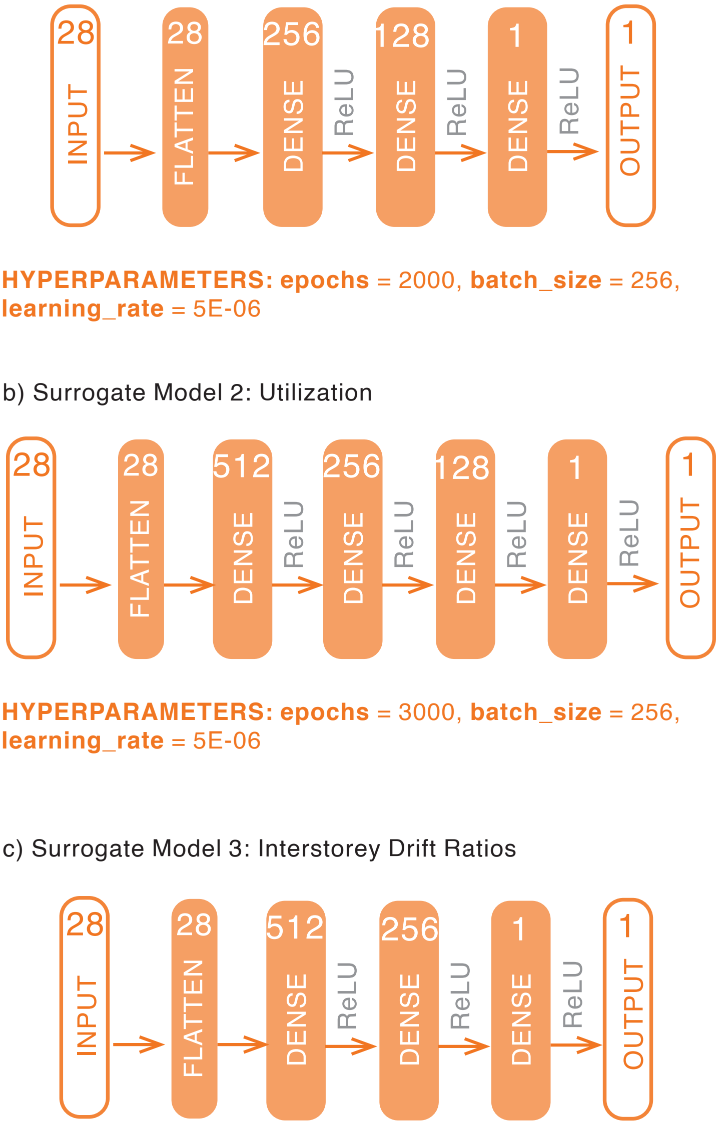
Single Surrogate Model



HYPERPARAMETERS: epochs = 3000, **batch_size** = 128, **learning_rate** = 1E-06

FIGURE 176: Best single surrogate model architecture - thickness normalized independently. (Own Work)

a) Surrogate Model 1: Buckling Load Factor



HYPERPARAMETERS: epochs = 2000, **batch_size** = 256, **learning_rate** = 5E-06

b) Surrogate Model 2: Utilization

HYPERPARAMETERS: epochs = 3000, **batch_size** = 256, **learning_rate** = 5E-06

c) Surrogate Model 3: Interstorey Drift Ratios

HYPERPARAMETERS: epochs = 3000, **batch_size** = 128, **learning_rate** = 5E-06

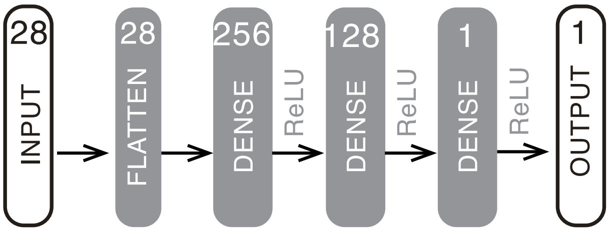
FIGURE 177: Best separate surrogate model architectures - thickness normalized independently. top (a) Buckling Load Factor, (b) Utilization, (c) Interstorey Drift Ratios (Own Work)

Besides predicting the performance metrics, two additional surrogate models were created to predict features of samples - the mass and the height. This was done in order to minimize material in the optimization process and selecting specific new samples having specific user-defined characteristics such as a desired height.

The best models gave the following NRMSE:

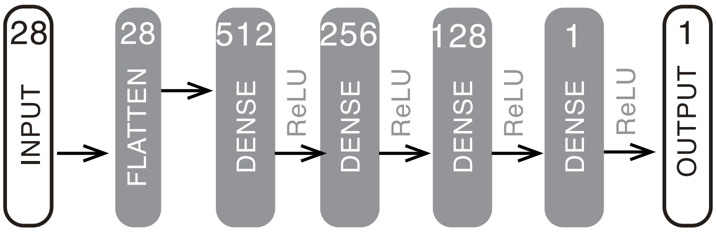
- Height: 6.95%
- Mass: 3.58%

a) Surrogate Model 4: Height



HYPERPARAMETERS: epochs = 10000, **batch_size** = 128, **learning_rate** = 5E-06

b) Surrogate Model 5: Mass



HYPERPARAMETERS: epochs = 3000, **batch_size** = 256, **learning_rate** = 5E-06

FIGURE 178: Best feature predicting surrogate model architectures - thickness normalized independently. top (a) Height, (b) Mass. (Own Work)

7.3 GRADIENT DESCENT OPTIMIZATION

7.3.1 Single Objective Optimizations

As each surrogate model predicted a separate performance metric, that particular metric could be optimized accordingly. This was useful for cases where it is known what the likely failure mode is. This was also true for our case where we know that failure is likely to occur in Utilization of the vault. Nonetheless, gradient descent on other performance metrics was explored to develop conclusions for future explorations for different datasets and geometries. It should be noted the Equations I - Equations VI in Section 4.1.6 were not considered for this.

VAE

Tests were initially conducted on the VAE. A sample from the test data was selected. Different learning rates were tested and the percentage change in performance was noted.

DATASET: randomized	
Sample for Optimization: Mesh_237	
Metric	**Values before Optimization**
Buckling Load Factor	12.6452
Utilization	10.6016
Interstorey Drift Ratio	2.83E-03h
Thickness	0.060m

TABLE 18: Metrics of mesh 237 - before optimization. (Own Work)

The learning rates were 0.01, 0.1, 1, 5 , 10, 15, 20 and 30. 1000 iterations were run for each simulation. In each case, the tables are organized according to the best performing optimizations at the top. The best learning rates for each metric has been highlighted in Figure 180.

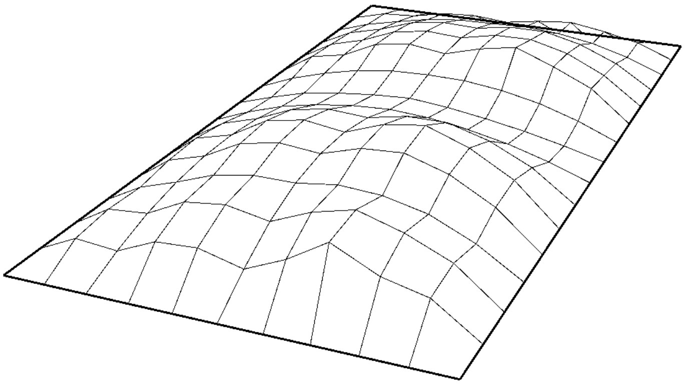


FIGURE 179: Mesh to be optimized: mesh 237. (Own Work)

For Buckling Load Factor, a learning rate of 20 was found to give the maximum optimization increasing the Buckling Load Factor from 12.6542 to 45.0239 by a percentage of 265.05%. The Gradient Descent algorithm optimized the thickness from 0.060m to 0.094m. This is expected as Where thickness could not be increased, such as in learning rate 1,5,10, and 30, the Buckling Load Factor instead reduced.

For Utilization, the best learning rate found was 15 which reduced the Utilization from 10.6016 to 2.0123 - a change of 81.02%. The thickness increased from 0.060m to 0.0688.

Even though the predicted performances of the surrogate models indicate better performances, the predicted force densities for Utilization and Buckling are too small to generate valid meshes. Though it may seem that the

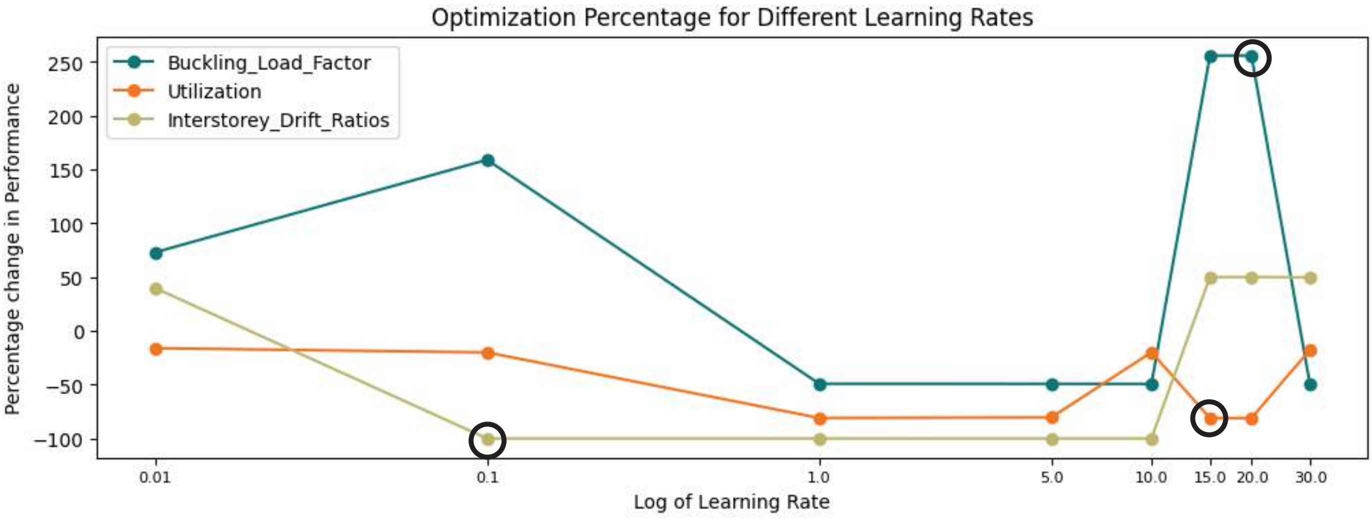


FIGURE 180: Log of learning rate vs percentage change in performance - gradient descent optimization. (Own Work)

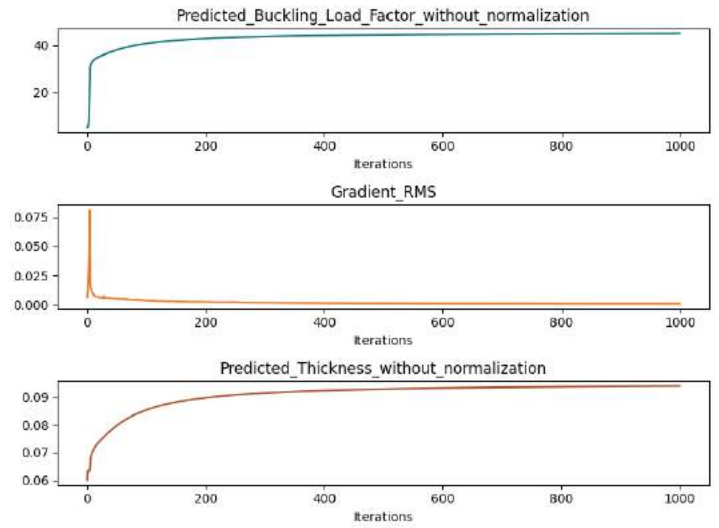


FIGURE 181: Predicted Buckling Load Factor without normalization, Gradient, and thickness vs iterations. learning rate = 20 (Own Work)

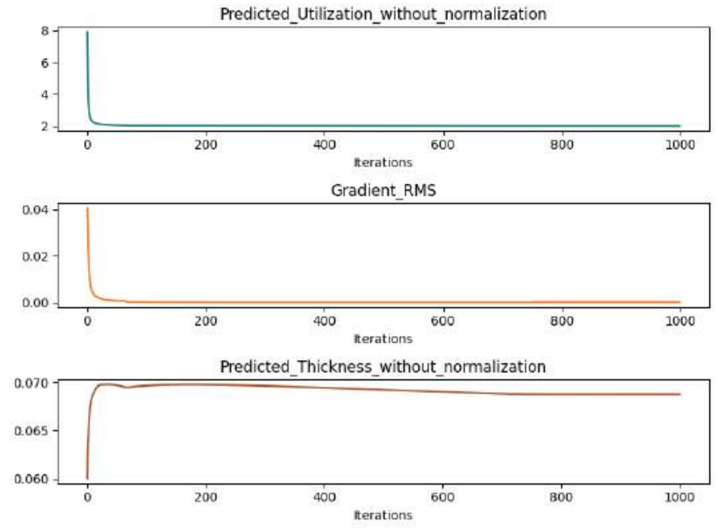


FIGURE 182: Predicted Utilization without normalization, Gradient, and thickness vs iterations. learning rate = 15 (Own Work)

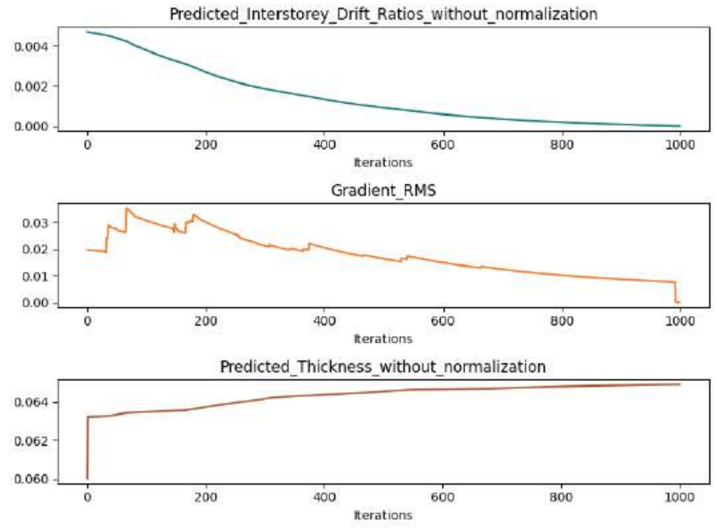


FIGURE 183: Predicted Interstorey Drift without normalization, Gradient, and thickness vs iterations. learning rate = 0.1 (Own Work)

DATASET: randomized - Mesh 37				
#	Gradient Descent: Buckling Load Factor			
	Learning Rate	Percentage change	Final Thickness / m	Final Performance score
1	20	256.05%	0.094	45.0239
2	15	255.97%	0.094	45.0139
3	0.1	159.13%	0.0702	32.768
4	0.01	72.99%	0.0646	21.8748
5	1	-49.14%	0.0637	6.4318
6	5	-49.19%	0.0637	6.4251
7	10	-49.27%	0.0637	6.4149
8	30	-49.61%	0.0637	6.3714

TABLE 19: Gradient descent on Buckling Load Factor. (Own Work)

DATASET: randomized - Mesh 37				
#	Gradient Descent: Utilization			
	Learning Rate	Percentage change	Final Thickness / m	Final Performance score
1	15	-81.02%	0.0688	2.0123
2	20	-81.02%	0.0688	2.0123
3	1	-80.95%	0.068	2.0197
4	5	-80.43%	0.0718	2.0750
5	0.1	-19.99%	0.0637	8.4822
6	10	-19.85%	0.0637	8.4973
7	30	-17.53%	0.0634	8.7430
8	0.01	-16.04%	0.0637	8.9007

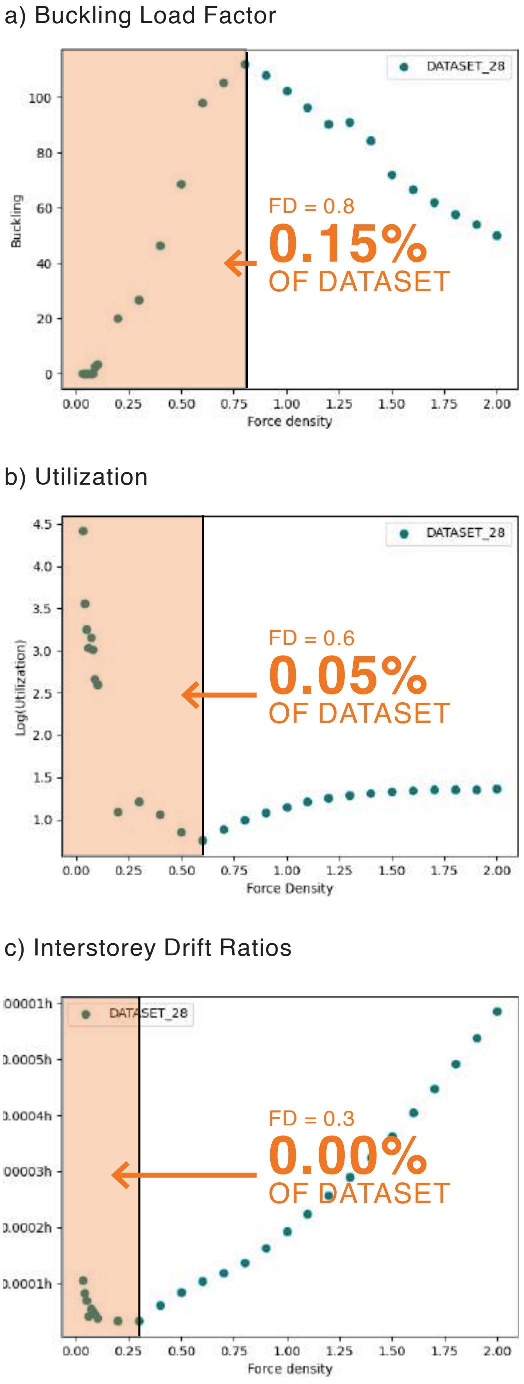
TABLE 20: Gradient descent on Utilization. (Own Work)

DATASET: randomized - Mesh 37				
#	Gradient Descent: Interstorey Drift Ratios			
	Learning Rate	Percentage change	Final Thickness / m	Final Performance score
1	0.1	-100%	0.0649	0.00E+00h
2	10	-100%	0.0668	0.00E+00h
3	1	-100%	0.0654	0.00E+00h
4	5	-100%	0.0654	0.00E+00h
5	0.01	39.99%	0.0636	3.96E-03h
6	30	49.89%	0.0636	4.24E-03h
7	15	50.03%	0.0636	4.24E-03h
8	20	50.12%	0.0636	4.24E-03h

TABLE 21: Gradient descent on Interstorey Ratios. (Own Work)

DATASET: randomized				
		Optimized mesh		
Edge no.	Initial Mesh	Buckling Load Factor	Utilization	Drift Ratio
col_1	0.20	3.75E-16	3.01E-02	0.60
col_2	0.30	1.22E-18	1.36E-02	0.34
col_4	0.40	9.66E-19	9.68E-03	0.35
col_5	0.10	9.29E-16	1.69E-02	0.38
col_6	0.50	1.40E-17	1.26E-02	0.43
col_7	3.30	6.74E-17	2.29E-02	0.60
col_8	0.50	1.50E-18	9.51E-03	0.27
col_9	0.50	2.52E-17	1.60E-02	0.48
col_10	4.20	4.75E-17	1.42E-02	0.44
col_11	0.60	1.51E-17	1.57E-02	0.45
row_1	1.30	1.03E-16	1.31E-02	0.34
row_2	7.50	3.06E-18	2.18E-02	0.46
row_3	0.80	4.78E-17	1.25E-02	0.39
row_4	3.40	5.69E-17	1.27E-02	0.33
row_5	1.00	3.95E-17	1.73E-02	0.48
row_6	1.00	2.63E-18	1.25E-02	0.43
row_7	0.30	5.50E-18	1.27E-02	0.33
row_8	3.40	5.12E-17	2.17E-02	0.48
row_9	9.60	6.38E-18	9.30E-03	0.29
row_10	4.90	1.70E-18	1.04E-02	0.45
row_11	0.80	1.13E-17	2.06E-02	0.38
row_12	0.40	5.89E-17	1.46E-02	0.40
row_13	0.60	1.34E-16	2.42E-02	0.48
row_14	1.90	2.72E-17	2.16E-02	0.47
row_15	0.40	6.59E-17	1.19E-02	0.42
row_16	0.10	2.47E-17	1.25E-02	0.36
row_17	0.50	9.19E-18	1.34E-02	0.43

TABLE 22: Force densities of best samples from each optimization. (Own Work)


a) Buckling Load Factor

b) Utilization

c) Interstorey Drift Ratios

FIGURE 184: Performance vs Force Density pattern - smaller force densities below inversion threshold highlighted in orange and larger force densities above inversion threshold. (Own Work)

optimization has gone wrong, it is evaluated that the reason for such numbers is that lower force densities show increased performance in all three metrics. The non-highlighted section of Figure 184 shows the pattern that that is representative of the data in the training set. It can be seen a decrease in force density is synonymous with an increase in performance. However, as the force densities fall below a certain threshold, performance decreases with a further decrease in the mean force density. This relationship (highlighted in orange in Figure 184) is not represented in the randomly generated dataset adequately enough for the pattern to be recognized by the surrogate models. This is quantitatively highlighted ahead.

A mean force density less than 0.8 is required for this performance inversion in Buckling Load Factor but only 0.15% of samples in the dataset represent this. A mean force density less than 0.6 is required for a performance inversion in Utilization but only 0.05% of the samples represent this. A mean force density of 0.3 is required for a performance inversion in Interstorey Drift Ratios but 0.00% of the samples represent this.

It is probable that the reason why the force densities for Interstorey Drift Ratios is near real values and not near zero is that Interstorey Drift Ratios reduce significantly for lower uniform force densities, as seen earlier. Before such low values reached for force densities, the minima is already reached. This also explains the lower learning rate (0.1) as compared to that for the other two metrics. Higher learning jump over the minima and instead of reducing the gradient, it increases. This can be seen by the fluctuating Gradient RMS in Figure 185 for the learning rate of 30 that instead causes an increase in the Interstorey Drift Ratio by 49.89%.

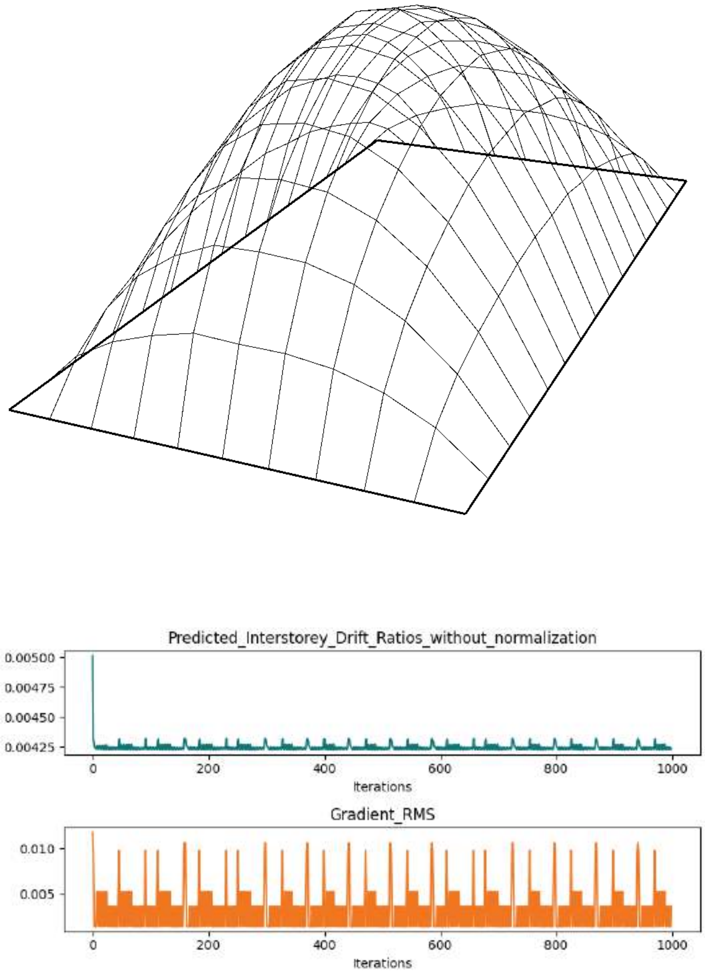


FIGURE 185: (top) Optimized mesh for Interstorey Drift Ratios - learning rate = 0.1, (bottom) Gradient RMS and predicted Interstorey Drift Ratios vs iterations for learning rate = 30

Conditional VAE

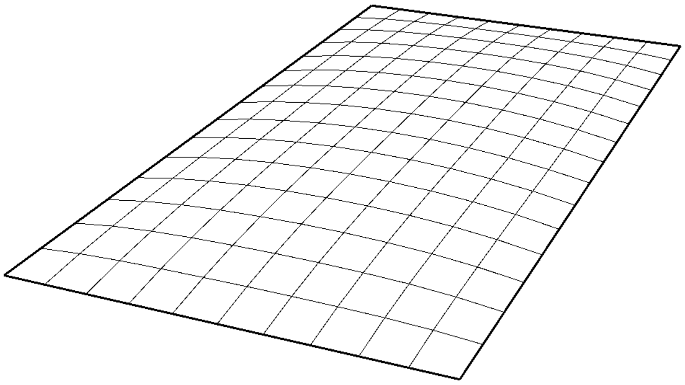
After the VAE, tests were conducted on a Conditional VAE where the initial sample was not taken from the dataset Instead a sample of desired height was extracted from the latent space and then optimized.

When one-hot-encoding was used, similar results resulted for all optimizations in Buckling Load Factor, Interstorey Drift Ratios, and Utilization. Invalid meshes formed due to force densities falling near zero.

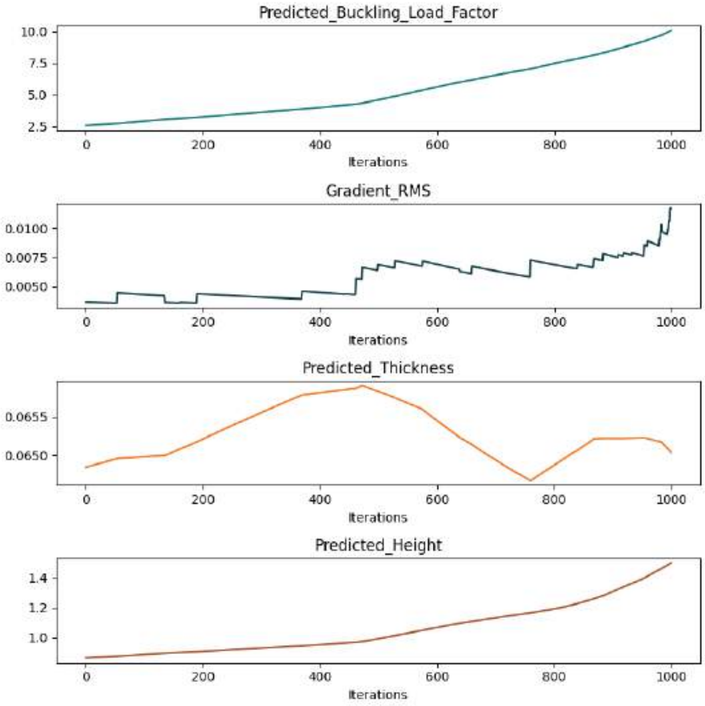
However, when Sinusoidal positional encoding was used, the latent space was better constrained to the desired label (height). Hence, this problem did not occur.

DATASET: randomized	
Sample for Optimization	
Metric	**Values before Optimization**
Buckling Load Factor	2.56
Utilization	14.41
Interstorey Drift Ratio	0.009612
Thickness	0.065m
Mass	32591 kg
Height	0.86m

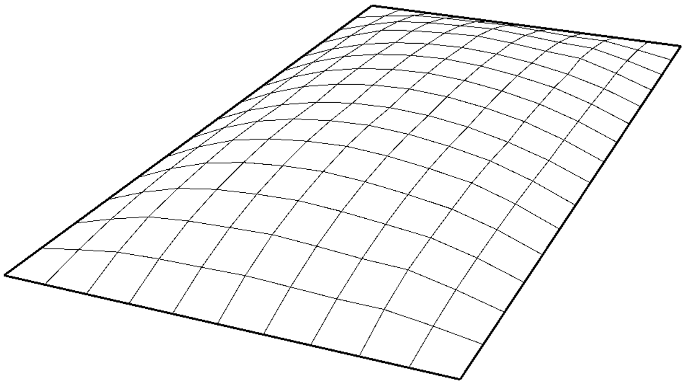
Initial mesh before optimization



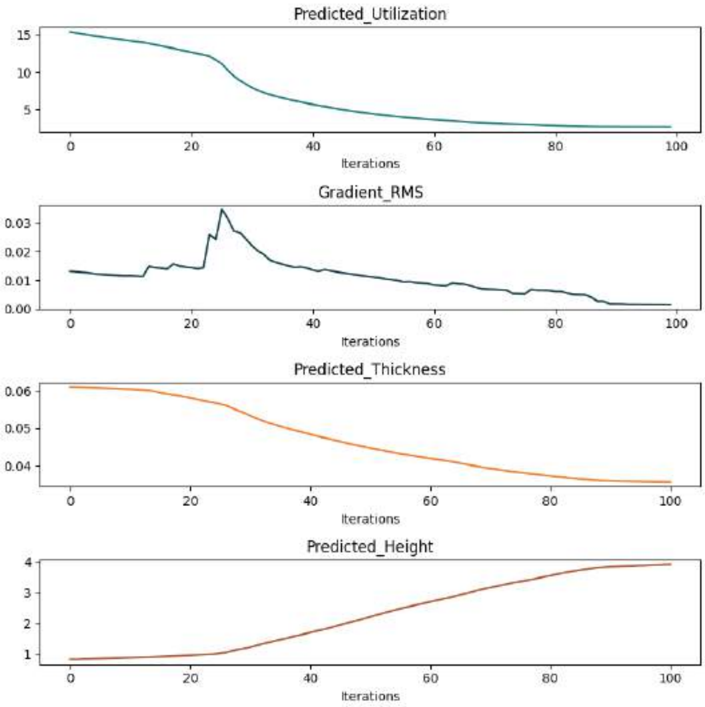
Learning rate = 1. iterations =1000



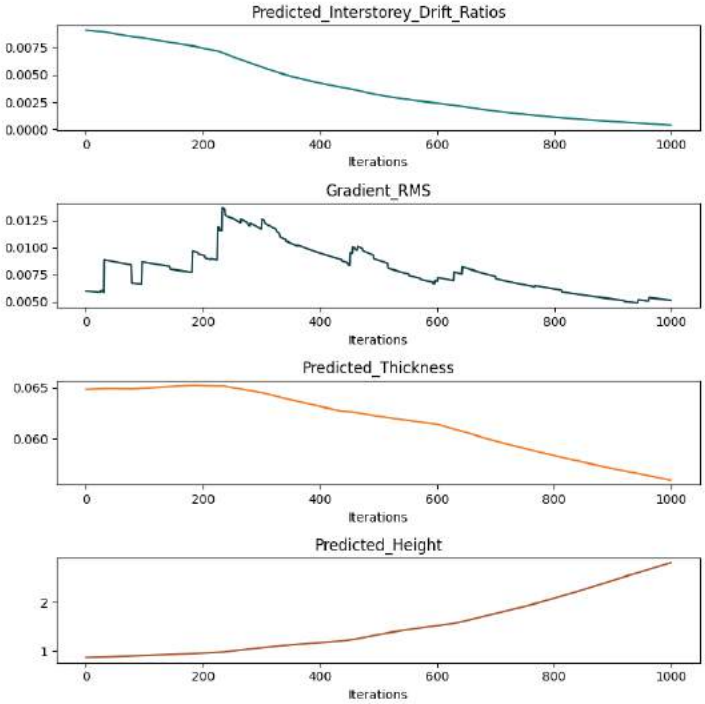
Mesh after optimization of Buckling Load Factor



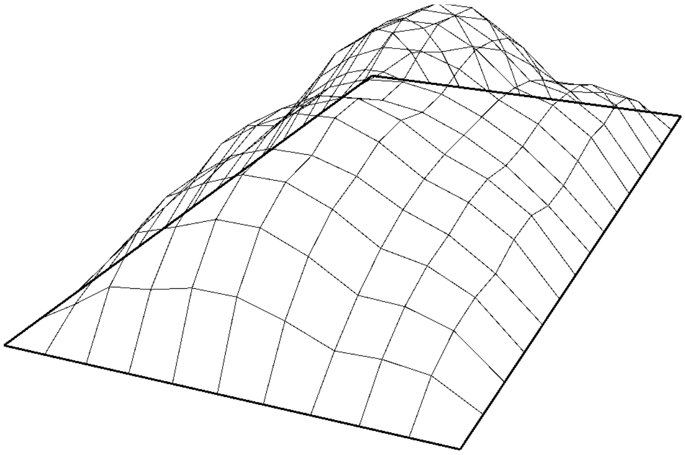
Learning rate = 8. iterations =100



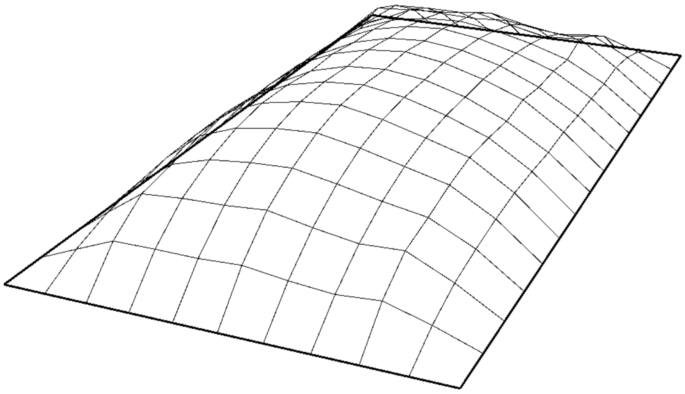
Learning rate = 1. iterations =1000



Mesh after optimization of Utilization



Mesh after optimization of Interstorey Drift Ratios



The Buckling Load Factor was optimized by 293% as it changed from 2.56 to 10.07. Interstorey Drift Ratios reduced by 95.8% as it changed from 0.009 to 0.0003. For Utilization, a higher number of iterations such as 10 caused force densities to drop near zero so 100 iterations were kept. This caused a reduction of 82.2% changing from 15.41 to 2.64

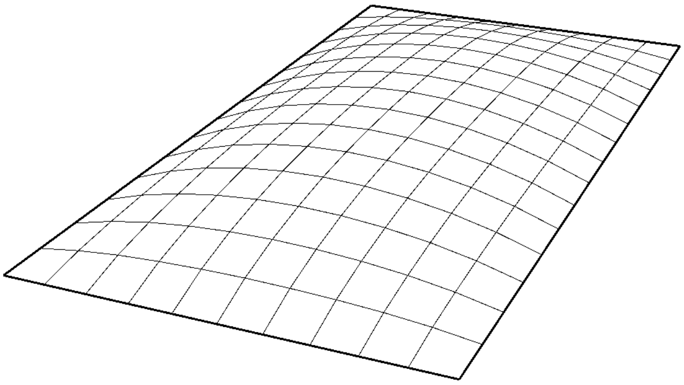
7.3.2 Multi-objective Optimization

All optimizations using Sinusoidal Positional Encoding caused an increase in height and thus, an increase in mass. Multi-objective optimizations were explored where the objective was of improving seismic performance metrics (SSS) in addition of decreasing material usage by reduction of mass. Experiments introducing the mass objective directly were unsuccessful but when the height objective was minimized, this worked (as mass reduces as a result of reduction of height). See Equations I - Equations VI in Section 4.1.6 which were considered for this.

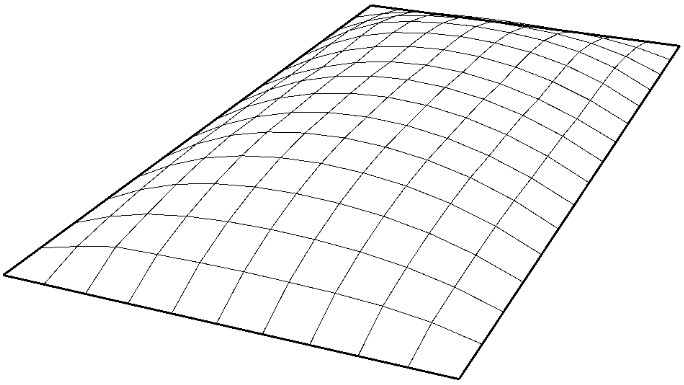
Multi-objective optimizations were not successful at optimizing conflicting objectives such as reducing material (by reducing height) and reducing Utilization. Manipulating the weights of objectives only allowed optimization of one objective and not the other.

DATASET: randomized	
Sample for Optimization	
Metric	Values before Optimization
Buckling Load Factor	8.63
Utilization	6.79
Interstorey Drift Ratio	0.00265
Thickness	0.061m
Mass	32701 kg
Actual Height	1.08 m
Predicted Height	1.42m
Desired Height	2.0m

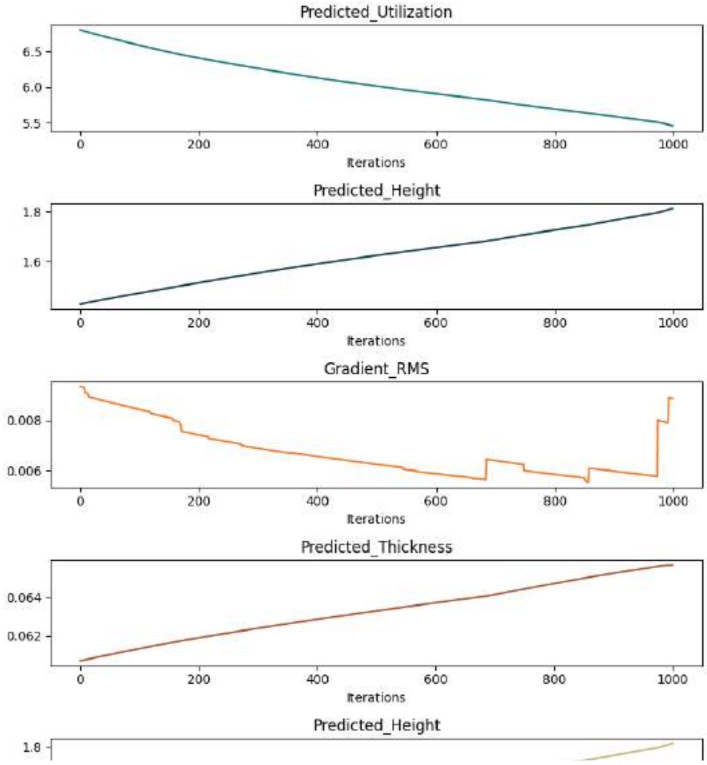
Initial mesh sampled from latent space.
Actual Height = 1.08m



Mesh after optimization of Utilization and material.
Predicted Height = 1.8m, Actual Height = 1.4m



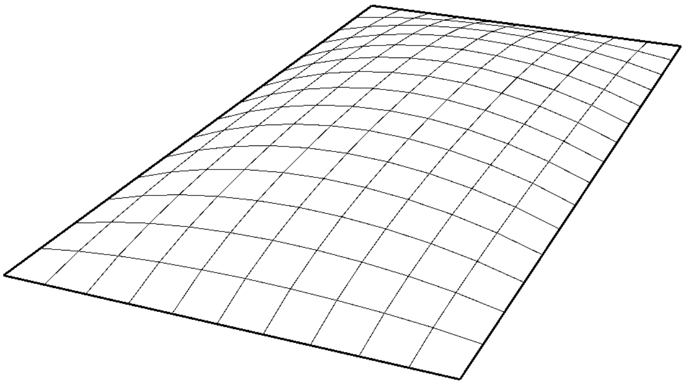
Learning rate = 0.5. iterations =1000



The weights were 1.0 for Utilization and 0.5 for Height. See Section 4.1.6 for the equation. Utilization reduced by 19.69% from 6.79 to 5.45 but Height (and thus, material) increased.

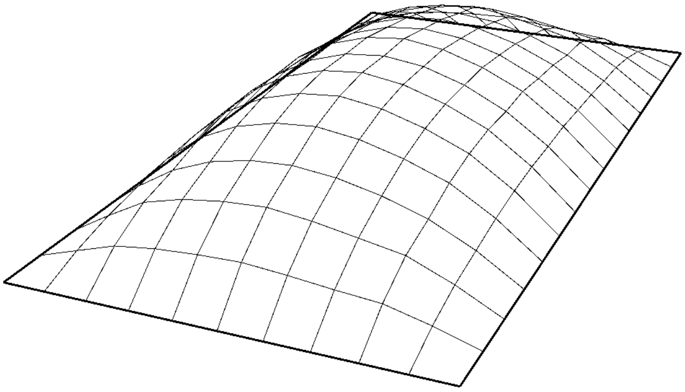
DATASET: randomized	
Sample for Optimization	
Metric	Values before Optimization
Buckling Load Factor	8.89
Utilization	6.99
Interstorey Drift Ratio	0.00292
Thickness	0.064m
Mass	33383 kg
Actual Height	1.06m
Predicted Height	1.38m
Desired Height	2.0m

Initial mesh sampled from latent space.
Actual Height = 1.08m

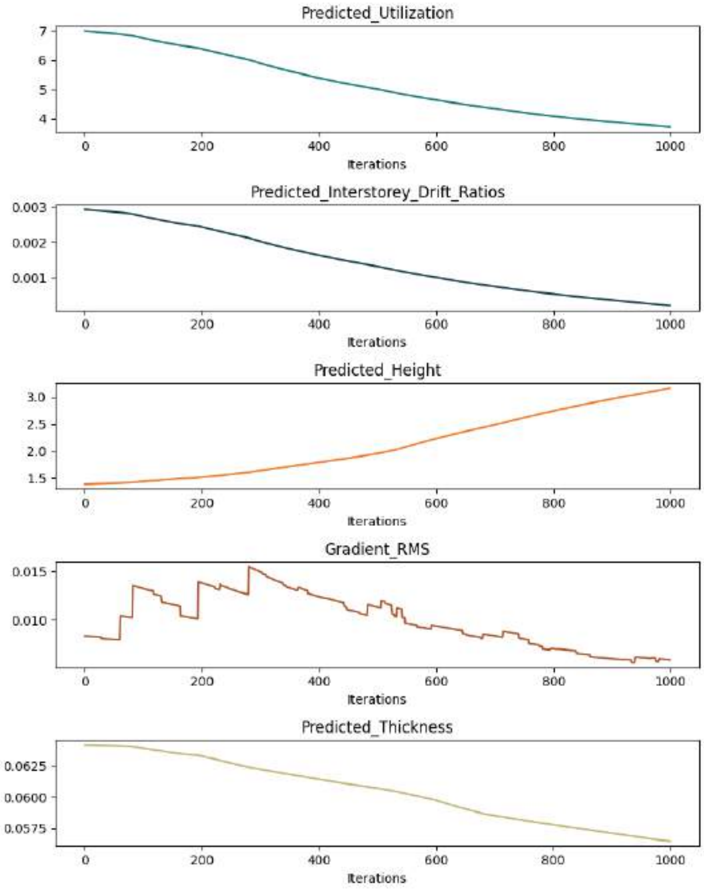


The weights were 1.0 for Utilization, 0.1 for Interstorey Drift Ratios and 0.5 for Height. There was a reduction in Utilization of 46.74% reducing from 6.99 to 3.72 and Interstorey Drift Ratio reduction of 92.42% from 0.00292 to 0.000222. The gradient was stabilized by the objective reducing height so near zero force densities did not form.

Mesh after optimization of Utilization and material.
Predicted height = 3.17m, Actual Height = 2.68m



Learning rate = 0.5. iterations =1000



08/ CONCLUSIONS AND REFLECTION

8.1 CONCLUSIONS

The main research question and sub-questions that the research paper aimed to answer will be answered below.

8.1.1 Main Research Question

Q) Can an AI based framework generate new Catalan vaults for optimized seismic performance for use as a floor slab?

Yes. The results have shown that the preliminary design of a vault may be optimized using the presented pipeline where optimization happens with respect to the latent space of a Conditional Variational Autoencoder using the predictive capabilities of surrogate models.

The framework uses a Conditional VAE trained on features (force densities and thickness) and labels (heights of each vault) to generate new vaults. Separate surrogate models are used to predict seismic performance metrics (Buckling Load Factor, Utilization, Interstorey Drift Ratios) as well as other metrics (height, mass) that are used in a multi-objective optimization that minimizes the aggregated gradient with respect to the latent space for gradient descent optimization.

Without including a threshold to minimize mass and limit the height for use as a floor slab, single-objective optimizations for Buckling, and Utilization generated invalid meshes with force densities close to zero. Interstorey Drift Ratio optimizations generated valid meshes only because the minima was reached much quicker.

However, once the thresholds were added to the gradient descent algorithm, optimized valid meshes were able to be generated through multi-objective optimization. Each objective has a weight which can be changed according to the requirement of the user. In this project, as failure was happening in Utilization, the weights for Buckling Load Factor and Interstorey Drift Ratios were kept zero while Utilization, Height, and Mass were non-zero. This way, separate metrics can be considered. Once the gradient falls below the threshold where failure in either metrics in ULS or SLS no longer happens, the gradient descent is stopped for that metric.

8.1.2 Subquestions

Q) Can the vaults be optimized for multiple performance metrics?

It is important to consider the nature of the optimization problem. Although non-conflicting objectives were able to be optimized together, it should be noted that conflicting objectives were not able to be optimized simultaneously during the multi-objective optimization such as reducing both, material usage and Utilization. A decrease in one caused an increase in the other. The reason for this may be that the latent space is restricted in generating a wide enough variety of samples that may satisfy the conflicting objectives; this is because even though the dataset is randomized, for each label condition, new samples only fluctuate around the same force density pattern so there is only a prominent change in height whilst force density distribution across polyedges largely remains similar.

Q) Can a user be able to tune the latent space in order to generate novel samples with user-defined desirable characteristics?

Yes, when the characteristic was encoded as a label, this was able to be done by conditioning the VAE. When the characteristic was encoded as a feature, this would be able to be done as long as the appropriate activation function was chosen in the hyperparameters.

Characteristic is a Feature

Thickness of the vault and force densities were encoded as features during training. Two different thickness normalization strategies were implemented. It was found that thickness normalization independent from the force densities provided better results for surrogate model predictions in all cases especially for the Buckling Load Factor.

Thickness was seen to change across both latent dimensions in all cases except ReLU being used as the only activation function. When Sigmoid was used as the only activation function, this change in thickness was linear across Dimension 1. This means that it can be used as a scrollable feature as well if a particular thickness needs to be sampled. However, very similar samples were generated this way which would be undesirable for sample generation.

Characteristic is a Label

Height was encoded as a label during training. Conditioning the VAE allowed for better control of the latent space. It has been proven that the latent space can be conditioned using labels to extract samples close to the desired label - height of the vault (in this project). For this, Sinusoidal Positional Encoding provided more accurate results than One-Hot Encoding and restricted the spread of sampled heights to a much larger degree. However, it should be noted that new samples were not exactly of the desired height but were instead 'close to' the desired heights. The deviation from the desired height continued to increase as the desired height itself/ label was increased (above 1.5m). There is potential to increase the accuracy by hyper-parameter tuning of the Conditional VAE. Another recommendation is to filter out outliers from the main dataset as this may affect the learning process; the heights ranged from 0.5m to 26.7m but only 6.06% of the total dataset had heights above 3.0m. If the total spread of heights is reduced, accuracy in sampling the desired height may be increased.

Q) Can individual dimensions of the VAE's latent space be interpretable?

Yes, this is true when considering characteristics that are encoded as features (thickness) or labels (height). Dimension 1 of the latent space was able to show an interpretable pattern in the change in thickness and height as we scrolled across that latent dimension.

It was seen that interpretability of Latent Dimension 1 improved and became almost linear when only Sigmoid was used as the activation function. This approach, however, did not work for the CVAE as all meshes sampled from the latent space were of the same height around 1.1m.

In conclusion, while individual dimensions may be interpretable to some extent, if the purpose is to tune it like a control knob to extract samples with a particular characteristic, it is recommended to include that as a label to condition the VAE with rather than use it as a scrollable feature of a latent dimension.

It would be interesting to see additional labels being encoded inside the Conditional VAE including thickness.

Q) What effect does varying the force densities have on overall seismic performance?

Decreasing the force densities increases the height of the vault which causes a improvement in seismic performance. This is due to an increase in the Buckling Load Factor, a decrease in Utilization and a decrease in Interstorey Drift Ratio. However, this improvement happens only till a threshold after which performance reduces if force densities decrease any further. For the specific vault properties used in this paper (dimensions = 10m x 15m, load = 40.0, PGA = 0.2g), the threshold for Buckling Load Factor is FD (force density) = 0.8, for Utilization FD = 0.6, and for Interstorey Drift Ratio FD = 0.3. These values are for vaults of uniform force densities.

Q) Is there any favourable pattern in terms of force densities for seismic performance?

In terms of force density patterns,the best performing meshes are those with uniform force densities for overall structural performance including seismic performance. Variations in force densities of the vault cause stress concentrations which reduces performance such when they are randomly generated or in the presence of creases.

Q) Can having a reduced sample dimension still justify the use of a Deep Neural Network?

The use of polyedges allowed the dataset to be reduced from 30,976 values to 27 values (for force densities) causing a reduction of 99.91%. If the VAE is seen in isolation, then no, using a Neural Network (NN) makes no sense as a randomizer can also generate force densities and thicknesses (features of the VAE). However, in the conditional VAE where a vault with a specific label (height) can be generated this becomes much more useful in extracting samples with hidden characteristics that cannot be explicitly stated during the geometry generation process - characteristics that they are not necessarily inputs to the process but are outputs. In the larger framework where the CVAE is connected to surrogate models used for gradient descent optimization, the approach becomes much more useful. The decrease in size allows it to be highly efficient as the final gradient descent optimization takes under 30 seconds for 1000 iterations (changing force densities in each iteration). This is incomparable to running a genetic algorithm in Grasshopper where each iteration in geometry takes 3.0-4.6 seconds (on the same computer) so the equivalent number of iterations would take 1 hr and 6 minutes.

8.2 CONSTRAINTS AND SHORTCOMINGS

8.2.1 ‘Garbage in Garbage Out’

A predictive optimization model is only as good as its dataset. In this case, since the dataset was composed of randomized force densities, the probability of having a vast array of ‘bad performing designs’ was great - which gives little room for optimization of already good performing meshes. This is because we have learnt from the Performance Evaluation chapter that fluctuations in force densities cause stress concentrations leading to a higher risk of failure - showing that uniform force densities show greatest seismic performance. The same is true for the accuracy of the Conditional VAE.

A large variation in heights due to the randomized nature of the dataset generation process had an adverse effect.

In essence, if the initial sample to be optimized is taken from the latent space or is any other non-uniform vault from the dataset, that, by the nature of its variable geometry performs poorly, the current model is able to optimize the gradient to a lower minima as it is trained on a dataset of randomized force densities. This however, does not mean that a good performing uniform vault from the dataset (made from uniform force densities) can be optimized using the same dataset.

8.2.2 Practical Application

From the literature it became clear that the dominant construction materials (wood, steel, concrete) could not be avoided completely especially for earth buildings due to their high mass and low stability in seismic events which is why ring beams and shear walls were included in the final structure designed using seismic guidelines for non-engineered structures. There is scope, however, to minimize the use of these materials by the strategies highlighted in the research - through the use of earthen vaults as floor slabs reinforced with basalt geo-grids. This has also been showcased in projects highlighted in the paper such as the SUDU housing in low seismic zones.

To conclude, there is some success in the optimization workflow by use of a VAE, surrogate model, and optimizer for initial stage design. However, practical considerations for such a workflow would not be feasible for use in a low-income housing project currently. This is we now know that because simpler is better in terms of force density distribution for structural stability of a vault.

The starting point of a design is integral to the outcome of the optimization. This would be useful as an optimizer for optimizing sculptural vaults which would be more appropriate as museum pieces rather than floors for a low-cost homes. By nature, the engineer/ architect would start from a simple vault (of uniform force densities). While it does not fit the intended purpose, the workflow shows scope for complex optimization tasks.

A significant absence in complete earthen construction lies in the absence of a slab system. Catalan vaulting highlights how shells can serve as the structural element for an earthen slab system. However, considering that Catalan vaulting might not be feasible everywhere, it’s crucial to explore and experiment with other local material technologies. It can be concluded that seismic performance of a vault is unique for a particular structure in relation to the number of storeys that structure. Therefore, if this construction technology is to be scaled for mid-rise buildings, each floor slab would need to need to designed uniquely.

8.2.3 Time Constraints

Due to time constraints in the project, it was not possible to conduct a more accurate non-linear analysis such as Non-linear Time History Analysis (NLTHA) for the Performance Evaluation as intended at the very initial stages. It became clear that such a technique was not feasible for a large dataset of 10,000 samples.

8.3 LOOKING AHEAD

8.3.1 Future Potential

Zooming out

In this research, for optimization, a linear dynamic seismic analysis was performed. The NRMSE for the surrogate models for the seismic performance metrics were 7.27%, 9.18% and 9.81% respectively, and it is highly probable that the error can be further reduced if a more structured dataset is used which limits the spread of data, such as the one recommended in Section 8.3.2. From this, it may be extrapolated that other more computationally expensive FEA approaches such as Non-linear Static Procedures (NSPs) and Discrete Element Modeling (DEM) may also be used in this way which provide greater accuracy. These could be run on supercomputers for the initial dataset generation process and then the same accuracy of these otherwise complex time consuming techniques could be harnessed through trained surrogate models. This may not be feasible for stand-alone niche design problems but for those problems where the dataset gives a good generalization of usage, it could be highly effective especially for companies which run the same type of optimization tasks. It may be a Utopian dream that surrogate models like such are trained and made open-source so that the average freelancer may be able to harness the power of complex macro-models and micro-models but the pace at which the AI is growing is somewhat of a dream itself I would think.

Though sinusoidal positional encoding already constrains the latent space well, gradient descent forces the z vector to be sampled from extreme ends of the latent space - which may results in large deviations from the desired condition. It would be interesting to see how the conditioned latent space during the gradient descent can be further restricted so that values outside the desired label are not sampled.

Zooming in

The dataset used in this project was based on a uniform footprint. It would be interesting to see how a variable footprint can be accommodated which opens up the potential for wider applications for floor sizes of variable dimensions. This can be challenging because the data-structure would need to be rethought as this would entail a different number of polyedges across different vaults (for the same polyedge density). A possibility lies in padding the 'missing' polyedges for smaller vaults. Moreover, in the future, tiling patterns may also be explored due and how they may contribute to increased seismic performance of the vault through patterns such as loxodrome. Representing these patterns into data for a neural network may pose a significant challenge. A possible approach for this is highlighted in the Appendices Section 10.2.4 and Section 3.1.3.

The polyedge density was kept constant throughout this project (1 polyedge per metre); as an increase in density would result in a smoother vault (for a vault with uniform force densities), it would be interesting to the effect of optimizing the polyedge density to evaluate the trade off between calculation time and structural performance and see how the graph converges. The 99.82% reduction in dimensionality of the sample size did not require the use of graphs due to the grouping of edges into polyedges. This was necessary for thin tile vault geometry. However, if other shell geometries are used that require individual edges to be expressed, then there is potential to explore Graph Convolution Networks for permutation invariance (See Section 3.2 for further detail).

8.3.2 Improvements for Future Reference

All samples failed in utilization . In terms of designing thin shell vaults for seismic loads, alternate form-finding strategies may be explored in the future for better performance. It would be very useful if appropriate reinforcement techniques such as geo-grids can be represented into data usable for neural network implementation.

Shell Corrugation

There is high potential to explore shell corrugation to allow the compression load path to carry lateral loads for earthquake resilience. Michiels et al. have highlighted a workflow that derives the geometries through a series of funicular polygons that are obtained by employing graphics statics for combined gravity and seismic loads (2019). This has resulted in a lateral capacity that is 79% greater than

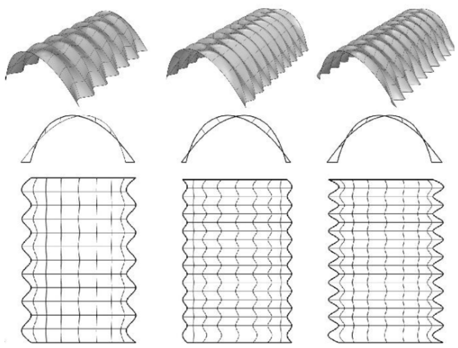


FIGURE 186: Shell corrugation. Image retrieved from Michiels, T., Adriaenssens, S., & Dejong, M. (2019). Form finding of corrugated shell structures for seismic design and validation using non-linear pushover analysis. Engineering Structures, 181, 362–373. https://doi.org/10.1016/j.engstruct.2018.12.043

non-form-found shapes for the same material usage (Michiels et al., 2019) outperforming Eladio Dieste's shell of a similar shape. Michiels et al. use a non-linear pushover analysis to determine change in capacity where it is found that corrugation depth has a high influence on the capacity whereas the corrugation period has a minor effect. It is yet unclear whether such an approach may be suitable to generate a large enough dataset required for a deep generative model as used in this paper. Nevertheless, it opens up potential for seismic design of thin tile masonry vaults.

Vault Segmentation

Another option is using vault segmentation as we've found that this improves the seismic performance of a uniform vault further, where each segment is supported independent to the rest of the vault. However, the data-structure for this may need to be rethought as the current data-structure does not account for variability in boundary supports as the current scheme assumes supports at the periphery. A solution may lie in using the creased dataset as an inspiration but with binary digits. (0) could represent polyedges where all force densities are the same while (1) could represent the polyedge that is supported. However, this would only work if all force densities for all samples are the same. To give a unique value of a force density to each sample, an additional digit may be added to represent the value of that force density It would be interesting to see if the binary values remain binary and the force density remains non-zero.

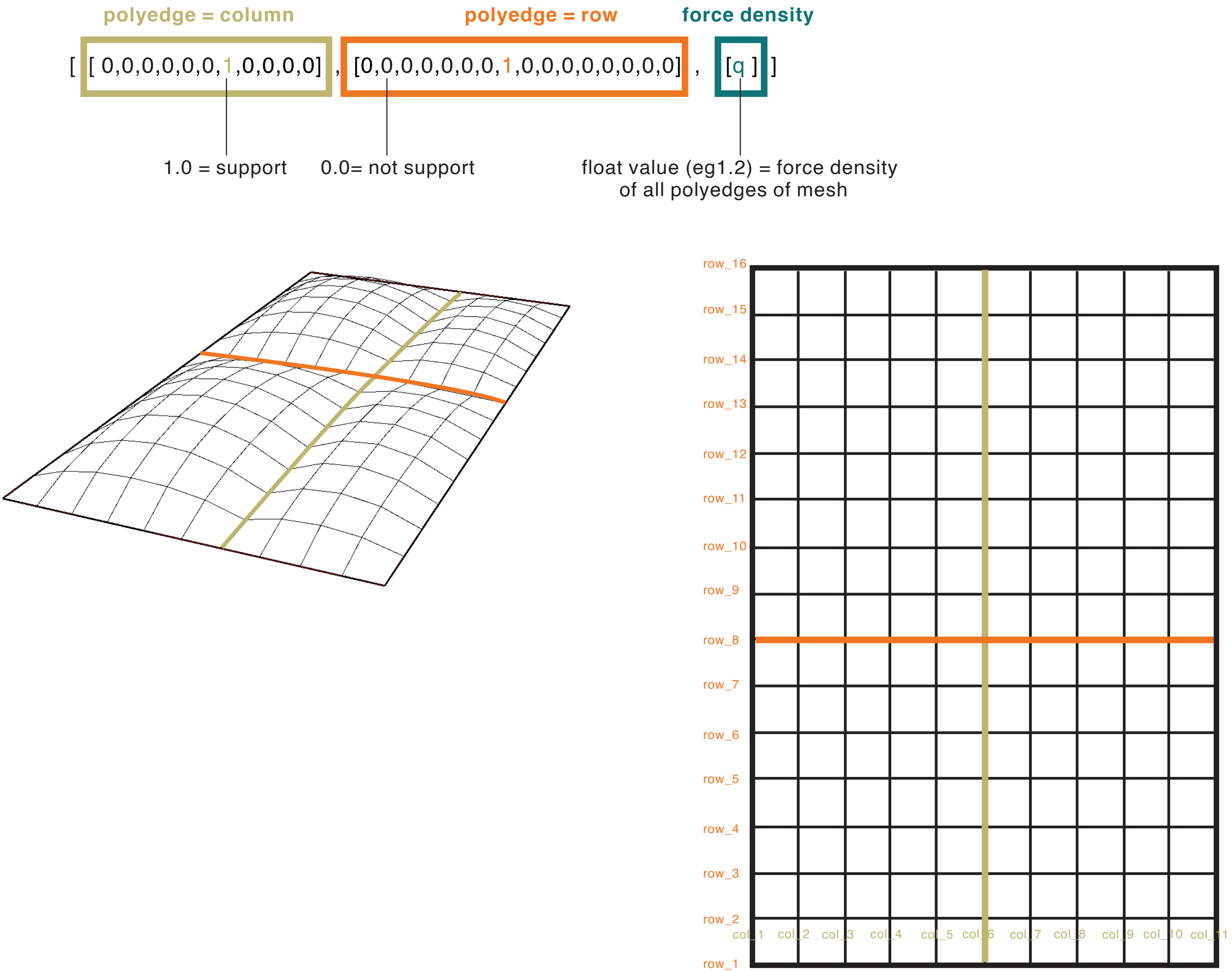


FIGURE 187: Possible data-structure visualization for vault segmentation. (Own Work)

A practical drawback of using segmentation, however, is the increase in the number of beams that are used - and the avoidance of a large number of beams (due to the absence of wood, steel, concrete) was the initial motivation behind the research. This would present a question: at what point does it make sense to even consider using a vault instead of a floor slab if there is already a large network of beams that is needed from the standpoint of seismic performance, material usage and cost effectiveness.

Feature-based Topology Finding

In feature-based topology finding techniques (Section 3.1.2), additional features such as curve features and point features were introduced. This could be of high significance if it could be able to represent vertical stiffeners like ribs for strengthening of vaults. Using force densities that have smoother transitions across polyedges instead may prevent stress concentrations.

Thickness Variation

In this research, a constant thickness of the vault was kept throughout all the samples. Though this is simpler in terms of construction, it is not very efficient in terms of material usage because of non-uniform force distribution. A strategy of non-uniform thickening may be adopted. This can be done by increase the thickness of the vault in areas of higher forces. To maintain continuity in the shell structure, an additional layer of tile can be sandwiched between the intrados and extrados such as in Los Manantiales Restaurant at Xochimilco, Mexico City by Felix Candela (Davis et al., 2012). Areas of higher force densities correspond to higher forces experienced – corresponding to areas where the thickness of the cross-section is to be increased.

8.3.3 Comparison With Other Optimization Techniques

This project only utilized the Gradient Descent algorithm for optimization. It would be interesting to see comparisons with other optimization algorithms in terms of performance and time.

When comparing the current optimization pipeline with conventional evolutionary algorithms built inside Grasshopper such as Galapagos (for single objective optimization) and Wallacei (for multi-objective optimization, we can make educated guesses based on two different criteria - time and performance. In terms of performance, the results lead to the assumption that a greater variety of options may be explored in Grasshopper-based evolutionary algorithms when compared to the current dataset. This is because Grasshopper based algorithms will not be constrained to the latent space of a VAE and thus would be able to generate a large variety of solutions. Therefore, the optimized performance may likely be better for inefficient latent spaces. However, it is imperative to consider the time it takes for Grasshopper to calculates a single iteration (ranges between 3.0 to 4.6 seconds) as compared to the current pipeline (0.015 seconds) which makes convergence in the Grasshopper-based optimizations196-400 times slower than the proposed framework.

A comparison that could be more interesting in the future would be to replace the Gradient Descent Optimization with Genetic Algorithms within the proposed pipeline. This would harness the computational efficiency of the fast structural predictions on instantly generated geometry in the latent space.

8.3.4 Comparison with other Generative models

The scope of the project was limited to exploration of a VAE. There is reason to believe there are other deep generative models that can be used as an alternate instead for topology optimization and shape optimization tasks. GANs are notoriously unstable and difficult to train but WGANs present potential in improving the stability of the environment and have been used for 3D structural topology optimization (Rawat & Shen, 2019) with a similar pipeline using CNN based surrogate models.

Reinforcement learning may also be explored alternatively as topology optimization problems have been solved with this (Jang et al., 2022).

8.4 THE TOPIC IN CONTEXT

8.4.1 In Context of the Studio

The topic, 'Generative Design of Catalan Vaults for Multi-storey Seismic Design', combines two fields of the Building Technology Master Track – Design Informatics (DI) and Structural Design (SD). Within the two fields, it specifically delves into the field of Deep Generative Design due to which it is relevant to the AiDAPT Lab as well.

The topic builds up from the foundations laid by previous courses completed in the Masters Program. The field of Artificial Intelligence (AI) was introduced in the Computational Intelligence for Integrated Design (AR0202), Sustainable Architectural whilst CORE (AR30B12) established a background in python programming and provided an introduction to seismic design. Materials and Structures (AR1B023) and Technoledge Structural Design (AR0133) developed a foundation for understanding of structural mechanics.

This thesis dealt with three complex fields with massive learning curves. The initial goals were kept quite ambitious and some of them were not achieved. However, this allowed the overall potential of the project to be explored further. The complexity of three different fields produced constraints on achievability of the initial targets and a greater in-depth study into either fields could not be carried out. Nonetheless, looking back, it proved to be a great learning experience where different interests were juxtaposed into a singular topic.

8.4.2 In Context of a Larger Social/ Technological Framework

In the broader context, the project aimed to serve as a test to determine whether the AI framework could generate reliable outputs - aiming to add to the state of the art. Learning to generate optimized solutions from a simpler dataset may be indicative of high potential for the model to generate optimized solutions for more complex micro-models if the training dataset had been trained on that performance evaluation model as well. This way, the proposed framework can serve as an application for the use of a VAE in shape optimization tasks which would otherwise be far too computationally expensive to perform.

Generative design is a developing field and has great potential in several disciplines including engineering and architectural design. While there is some precedent research that deals with generative design of skeletal shells, there was a research gap in generative design for thin tiled shells. Moreover, there is also a gap in research in the development of a generative framework for seismic design as well. This research had aimed to deal with these research gaps in the larger scientific framework.

The project targets a problem that is a growing concern in the present – the need to go vertical in an absence of conventional building materials. Due to a need for multistorey construction arising from rapid population growth, it may pose a much greater problem in the future

09/ BIBLIGORAPHY

Adiels, E., Ander, M., & Williams, C. (2017). Brick patterns on shells using geodesic coordinates.

Al-Shehari, T., & Alsowail, R. A. (2021). An Insider Data Leakage Detection Using One-Hot Encoding, Synthetic Minority Oversampling and Machine Learning Techniques. Entropy, 23(10), 1258. https://doi.org/10.3390/e23101258

Block, P., Ciblac, T., & Ochsendorf, J. (2006). Real-time limit analysis of vaulted masonry buildings. Computers & Structures, 84(29–30), 1841–1852. https://doi.org/10.1016/j.compstruc.2006.08.002

Butz, D. A., & Cook, N. E. (2011). Accessibility interrupted: The Shimshal road, Gilgit-Baltistan, Pakistan. Canadian Geographies / Géographies Canadiennes, 55(3), 354–364. https://doi.org/10.1111/j.1541-0064.2011.00365.x

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., & Lerchner, A. (2018). Understanding disentangling in β-VAE (arXiv:1804.03599). arXiv. http://arxiv.org/abs/1804.03599

Cleland, J. (2013). World Population Growth; Past, Present and Future. Environmental and Resource Economics, 55(4), 543–554. https://doi.org/10.1007/s10640-013-9675-6

Contestabile, M., Iuorio, O., & Garrity, S. W. (n.d.). An Investigation of Alternative Form-Finding Methods for Free-Form Masonry Shells.

Davis, L., Rippmann, M., & Pawlofsky, T. (2012). Innovative funicular tile vaulting: A prototype vault in Switzerland.

EN 1990: Eurocode - basis of structural design. (n.d.). https://www.phd.eng.br/wp-content/uploads/2015/12/en.1990.2002.pdf

Fil, M., Mesinovic, M., Morris, M., & Wildberger, J. (2021). Beta-VAE Reproducibility: Challenges and Extensions (arXiv:2112.14278). arXiv. http://arxiv.org/abs/2112.14278

Gladstone, R. J., Nabian, M. A., Keshavarzzadeh, V., & Meidani, H. (2021). Robust Topology Optimization Using Variational Autoencoders (arXiv:2107.10661). arXiv. http://arxiv.org/abs/2107.10661

Heyman, J. (1966). The stone skeleton. International Journal of Solids and Structures, 2(2), 249–279. https://doi.org/10.1016/0020-7683(66)90018-7

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., & Lerchner, A. (2017). β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework.

International Organization for Standardization. (2004). EN 1998-1:2004 Eurocode 8: Design of structures for earthquake resistance - Part 1: General rules, seismic actions and rules for buildings. https://www.phd.eng.br/wp-content/uploads/2015/02/en.1998.1.2004.pdf

Ivasiuk, B., & Misino, E. (n.d.). Conditional Image Generation.

Jang, S., Yoo, S., & Kang, N. (2022). Generative Design by Reinforcement Learning: Enhancing the Diversity of Topology Optimization Designs. Computer-Aided Design, 146, 103225. https://doi.org/10.1016/j.cad.2022.103225

Javanmardi, R., & Ahmadi-Nedushan, B. (2023). Optimal design of double-layer barrel vaults using genetic and pattern search algorithms and optimized neural network as surrogate model. Frontiers of Structural and Civil Engineering, 17(3), 378–395. https://doi.org/10.1007/s11709-022-0899-9

Kingma, D. P., Salimans, T., & Welling, M. (2015). Variational Dropout and the Local Reparameterization Trick (arXiv:1506.02557). arXiv. http://arxiv.org/abs/1506.02557

Leifsson, L., & Koziel, S. (2016). Surrogate modelling and optimization using shape-preserving response prediction: A review. Engineering Optimization, 48(3), 476–496. https://doi.org/10.1080/0305215X.2015.1016509

López López, D., Van Mele, T., & Block, P. (2019). The combination of tile vaults with reinforcement and concrete. International Journal of Architectural Heritage, 13(6), 782–798. https://doi.org/10.1080/15583058.2018.1476606

Marseglia, P. S., Micelli, F., & Aiello, M. A. (2020). Analysis of Equivalent Diaphragm Vault Structures in Masonry Construction under Horizontal Forces. Heritage, 3(3), 989–1017. https://doi.org/10.3390/heritage3030054

Michiels, T., Adriaenssens, S., & Dejong, M. (2019). Form finding of corrugated shell structures for seismic design and validation using non-linear pushover analysis. Engineering Structures, 181, 362–373. https://doi.org/10.1016/j.engstruct.2018.12.043

Nanayakkara, K. I. (2019). Shell Structures from Catalan to Mapungubwe Lessons from Structural Efficiency for Sustainable Construction in Developing Countries. https://doi.org/10.13140/RG.2.2.30878.89922

Oh, S., Jung, Y., Kim, S., Lee, I., & Kang, N. (2019). Deep Generative Design: Integration of Topology Optimization and Generative Models. Journal of Mechanical Design, 141(11), 111405. https://doi.org/10.1115/1.4044229

Oval, R., & Rippmann, M. (2017). Patterns for Masonry Vault Design.

Panozzo, D., Block, P., & Sorkine-Hornung, O. (2013). Designing unreinforced masonry models. ACM Transactions on Graphics, 32(4), 1–12. https://doi.org/10.1145/2461912.2461958

Paris, V., Pizzigoni, A., & Adriaenssens, S. (2020). Statics of self-balancing masonry domes constructed with a cross-herringbone spiraling pattern. Engineering Structures, 215, 110440. https://doi.org/10.1016/j.engstruct.2020.110440

Pastrana, R. (2022). Disentangling Variational Autoencoders. https://doi.org/10.48550/arxiv.2211.07700

Patrikar, S. (2019, October 1). Batch, Mini Batch & Stochastic Gradient Descent. Medium. https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a

Pavlidou, S. (2022). A Deep Learning Framework for Optimized Shell Structures [Master Thesis]. TU Delft.

Rawat, S., & Shen, M. (2019). Application of Adversarial Networks for 3D Structural Topology Optimization. SAE Technical Paper 2019-01-0829. https://doi.org/10.4271/2019-01-0829.

Ramage, M. H., & Dejong, M. J. (n.d.). Design and Construction of Geogrid-reinforced Thin-shell Masonry.

Regenwetter, L., Nobari, A. H., & Ahmed, F. (2022). Deep Generative Models in Engineering Design: A Review. Journal of Mechanical Design, 144(7), 071704. https://doi.org/10.1115/1.4053859

Ruder, S. (2017). An overview of gradient descent optimization algorithms (arXiv:1609.04747). arXiv. http://arxiv.org/abs/1609.04747

Sanchez-Lengeling, B., Reif, E., Pearce, A., & Wiltschko, A. B. (2021). A Gentle Introduction to Graph Neural Networks. Distill, 6(9), e33. https://doi.org/10.23915/distill.00033

Schek, H. J. (1973). The Force Density Method for Form-Finding and Computation of General Networks. North-Holland Publishing Company.

Siddique, M. S., & Schwarz, J. (2015). Elaboration of Multi-Hazard Zoning and Qualitative Risk Maps of Pakistan. Earthquake Spectra, 31(3), 1371–1395. https://doi.org/10.1193/042913EQS114M

Soni, A. (2018). Introduction to evolutionary algorithms. Towards Data Science.

Sterrenberg, A. (2023). A Deep Learning Framework for Optimized Spatial Truss Structures with Stock Constraints [Master Thesis]. TU Delft.

Surat, D. (2017). Seismic Analysis of Thin Shell Catenary Vaults [Master Thesis]. University of Witwatersrand.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). Attention Is All You Need (arXiv:1706.03762). arXiv. http://arxiv.org/abs/1706.03762

Webmaster, A. E. I. (n.d.). http://www.earth-auroville.com/design_and_masonry_en.php

Williams, B. A., & Cremaschi, S. (2019). Surrogate Model Selection for Design Space Approximation And Surrogatebased Optimization. In Computer Aided Chemical Engineering (Vol. 47, pp. 353–358). Elsevier. https://doi.org/10.1016/B978-0-12-818597-1.50056-4

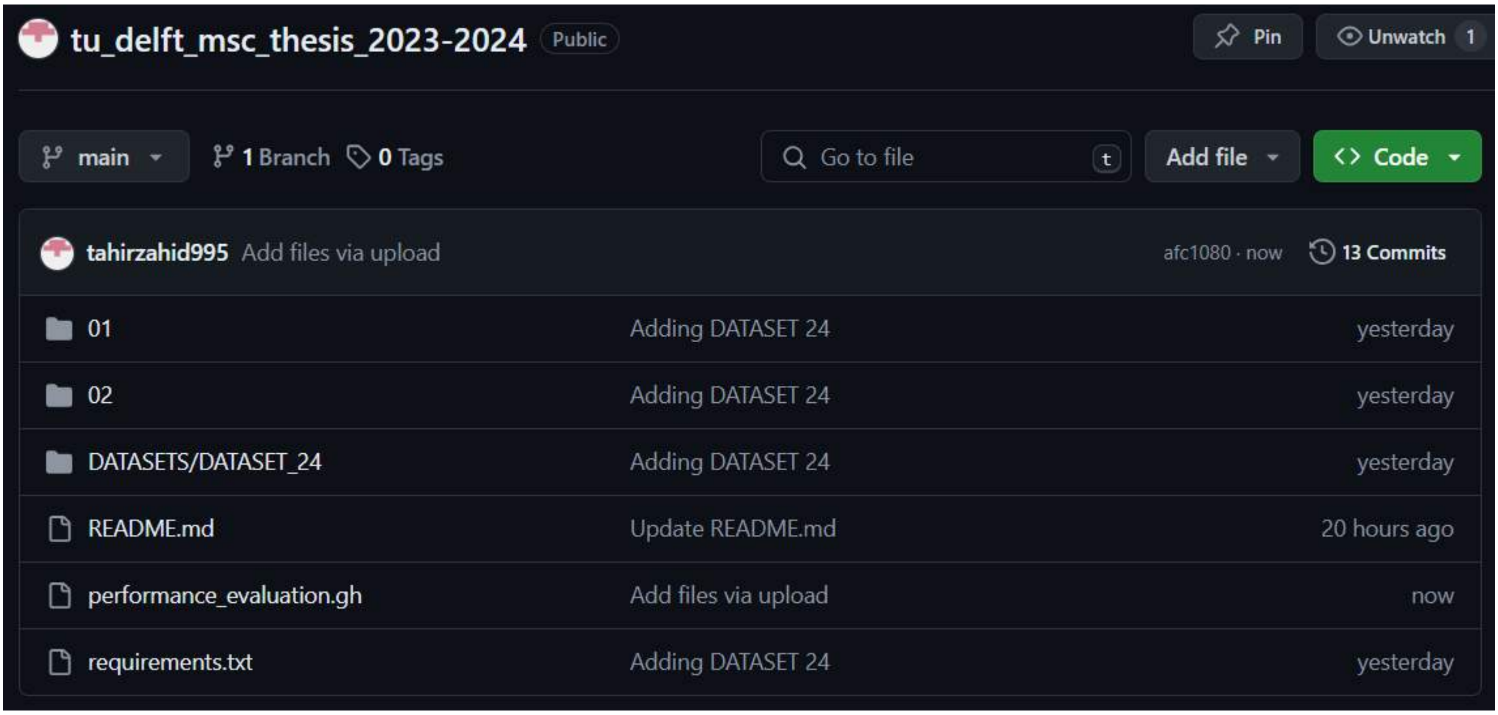
Wyss, M., & Rosset, P. (2013). Mapping seismic risk: The current crisis. Natural Hazards, 68(1), 49–52. https://doi.org/10.1007/s11069-012-0256-8

Zhang, B., Xiong, D., Su, J., Duan, H., & Zhang, M. (2016). Variational Neural Machine Translation. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 521–530. https://doi.org/10.18653/v1/D16-1050

10 / APPENDICES

The code for this project can be found in the Git Hub repository:

https://github.com/tahirzahid995/tu_delft_msc_thesis_2023-2024.git

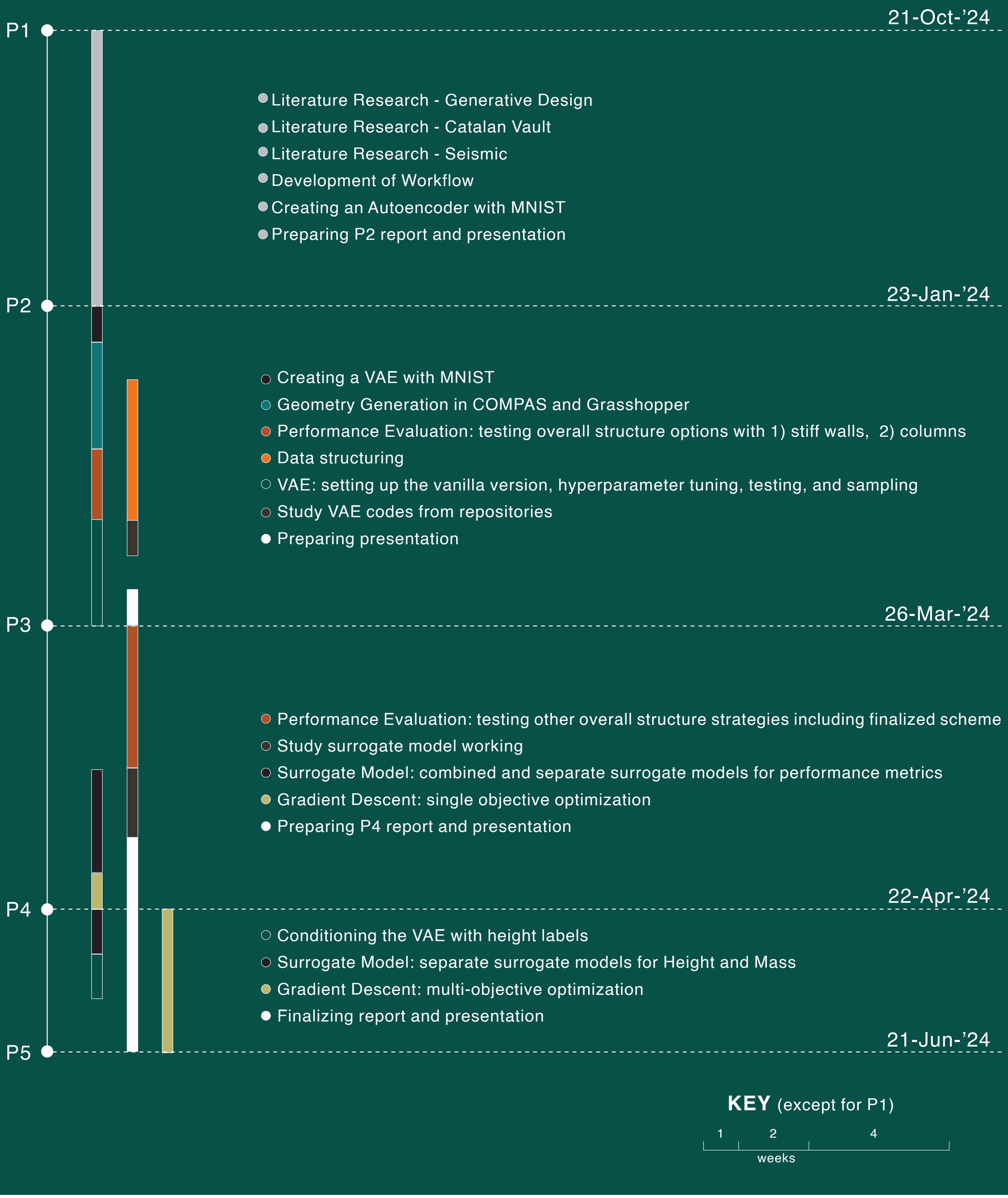


tahirzahid995 Add files via upload afc1080 · now 13 Commits

01	Adding DATASET 24	yesterday
02	Adding DATASET 24	yesterday
DATASETS/DATASET_24	Adding DATASET 24	yesterday
README.md	Update README.md	20 hours ago
performance_evaluation.gh	Add files via upload	now
requirements.txt	Adding DATASET 24	yesterday

The grasshopper script ‘performance_evaluation.gh’ is the script for the seismic analysis while 01 and 02 are directories for the python code. ‘01’ is relevant to the Geometry Generation while ‘02’ is relevant to the Generator containing the VAE/CVAE, surrogate models, and Gradient Descent. DATASET 24 is the randomized dataset used but directory ‘01’ can be used for generating a new dataset while the grasshopper script can be used to perform the Performance Evaluation on it.

10.1 GRADUATION TIMELINE



P1 21-Oct-'24

- ● Literature Research - Generative Design
- ● Literature Research - Catalan Vault
- ● Literature Research - Seismic
- ● Development of Workflow
- ● Creating an Autoencoder with MNIST
- ● Preparing P2 report and presentation

P2 23-Jan-'24

- ● Creating a VAE with MNIST
- ● Geometry Generation in COMPAS and Grasshopper
- ● Performance Evaluation: testing overall structure options with 1) stiff walls, 2) columns
- ● Data structuring
- ○ VAE: setting up the vanilla version, hyperparameter tuning, testing, and sampling
- ● Study VAE codes from repositories
- ● Preparing presentation

P3 26-Mar-'24

- ● Performance Evaluation: testing other overall structure strategies including finalized scheme
- ● Study surrogate model working
- ● Surrogate Model: combined and separate surrogate models for performance metrics
- ● Gradient Descent: single objective optimization
- ● Preparing P4 report and presentation

P4 22-Apr-'24

- ○ Conditioning the VAE with height labels
- ● Surrogate Model: separate surrogate models for Height and Mass
- ● Gradient Descent: multi-objective optimization
- ● Finalizing report and presentation

P5 21-Jun-'24

KEY (except for P1)

10.2 INITIAL ALTERNATIVE WORKFLOWS

After the initial workflow shown in Figure 05 in Section 1.4, the regular footprint was to be made variable and non-linearity introduced in the analysis. Refer to Section 3.2.2 where the limitations of LSP are discussed for masonry vaults – which is the reason it was considered initially. For all the alternative workflows (WF2A, WF2B, WF3), the Performance Evaluation was to be done using NSP. This applied to Workflow 3 as well where tiling was to be done. This is because although the Discrete Element Method (DEM) simulates a discretized, hence tiled surface, it would be far too computationally intensive to be done for a dataset of 10,000 simulations. Therefore, a continuous model was to be assumed for the Performance Evaluation phase, and if time permitted, a DEM analysis could have been carried out on 1 sample be carried out later as Validation. It was found, however, that non-linear analysis techniques were far too time consuming to be carried out within the project time-frame. This could be carried out in the future when there is greater time and computational power.

Either Workflow 2A (WF2A) would be carried out or Workflow 2B (WF2B). The main difference between the two was the form finding approach - Workflow 2B had much greater complexity in generating a different topology patterns through the feature-based topology finding approach. Workflow 3 would build upon Workflow 2A introducing tiling patterns. This is shown in Figure 07.

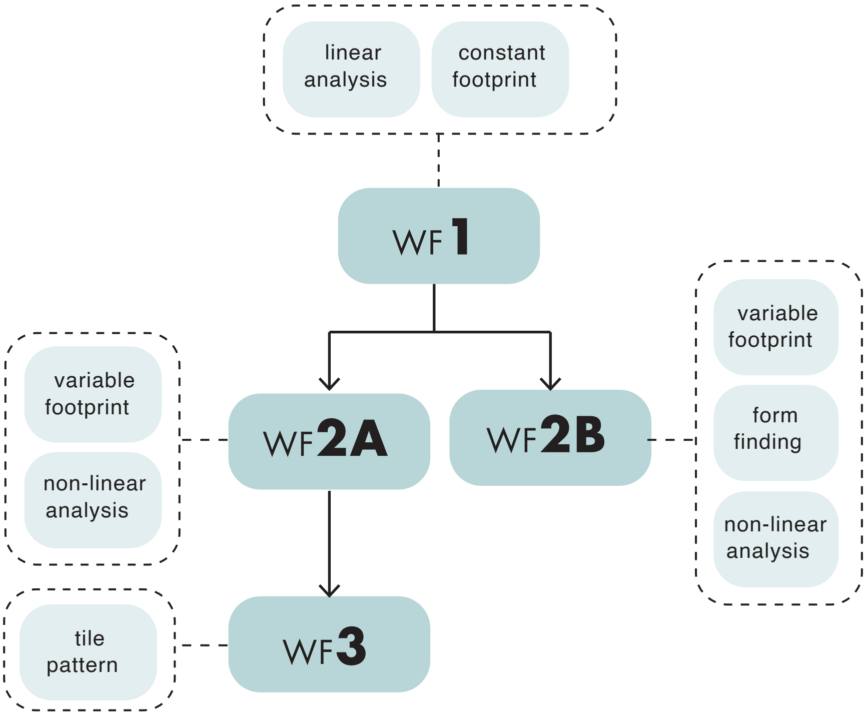


FIGURE 188: Initial Workflow logic. (Own Work)

10.2.1 Workflow 2A (WF2A) and 2B (WF2B)

Please refer to Figure 08 for reference.

Data Generation

A variable input rectangular footprint would generated of x and y dimensions. Form-finding would be next done like in WF1. In WF2A, a regular grid like the one in WF1 would be used; in WF2B, feature-based topology finding would be done instead.

Performance Evaluation

For both workflows, the continuous geometry would be then evaluated seismically through a Non-linear Static Procedure (NSP) in ABAQUS. For material properties, please refer to the Section 3.2.

Data Structuring

For WF2A, it is the same as WF1. For WF2B, graphs may not be used to represent nodal data.

VAE 2

For WF2A, the structured data would be used as an input dataset to VAE 2. In case of WF2B, GCNs may not be used due to absence of graph data.

10.2.2 Workflow 3 (WF3)

Please refer to Figure 09 for reference. This is only done if Workflow 2A is chosen in the previous step.

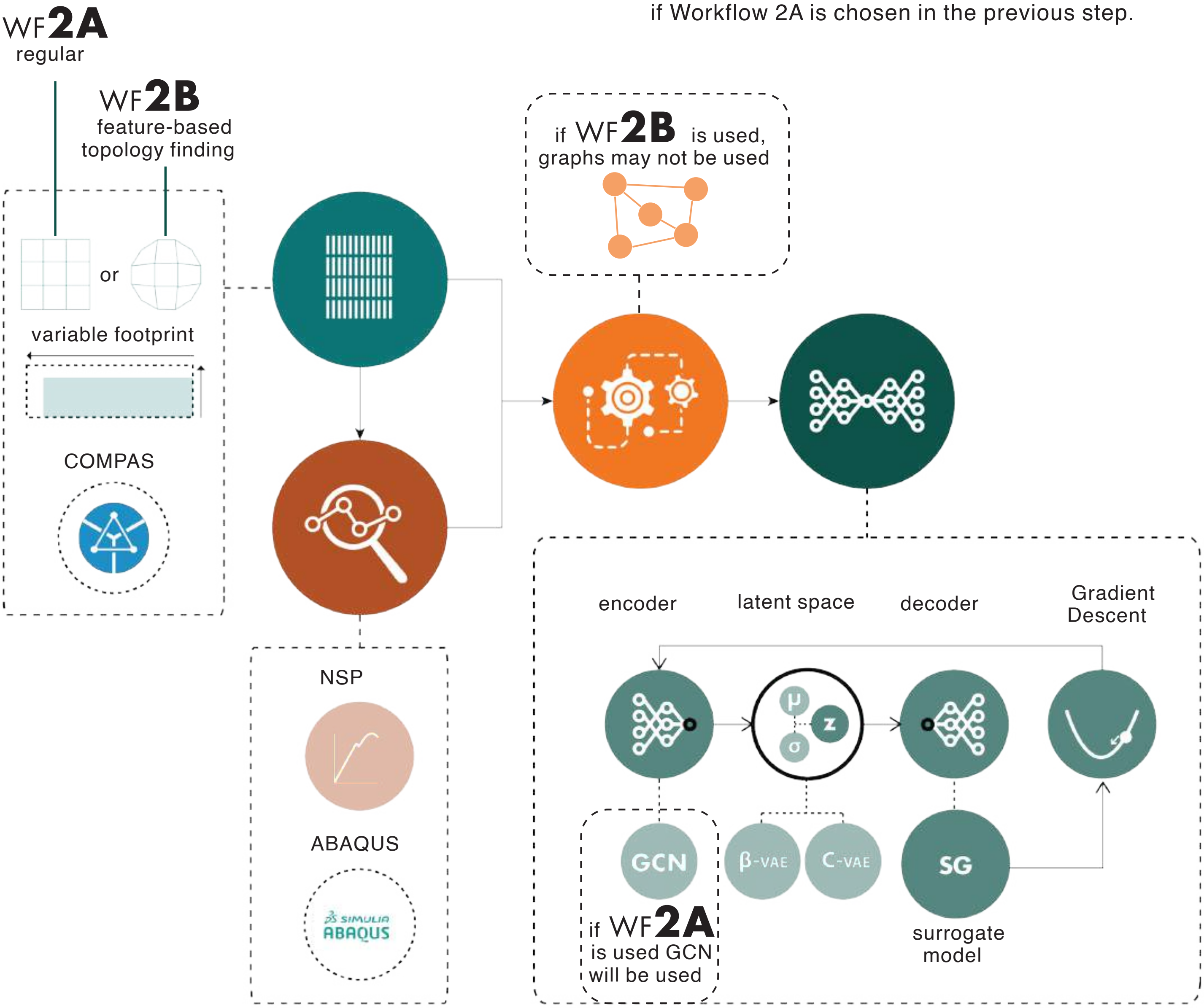


FIGURE 189: Workflow 2A and 2B. (Own Work)

Data Generation

The same Data Generation process from Workflow 2A would also be done here. However, after that, tiling curves would be mapped onto it; Patterning Approach 1 would be used for this purpose due to integration of patterning with structural stability (See Section 3.1.3 for more information on Patterning Approaches). Then, Patterning Approach 2 will be pursued. It should be noted that mesh is not discretized, it is a continuous one - the tiling only exists as curves.

Performance Evaluation

Same as Workflow 2A.

Data Structuring

The data would then be restructured similar to Workflow 2A. However, this time, the tile geometry would need to be restructured as well. It would become far too computationally expensive to represent every tile separately as a node. Hence, it is hypothesized that this may have been done by grouping similar tiles and representing them by a single node. It would have been logical that the tiling pattern could have been based on the meshing pattern so nodes are based on aggregations of similar tiles according to position and direction (as indicated in Patterning Approach 1). This is shown in Figure 31. The tiling direction was to be represented by a vector in radians.

VAE 2

Same as Workflow 2A.

Validation through DEM (Optional)

This part was optional. The tiles were to be extruded. Then two simulations of different tiling patterns were to be run to see how the tiling pattern affects seismic performance through Discrete Element Method. It is too computationally expensive to be included in the dataset formation.

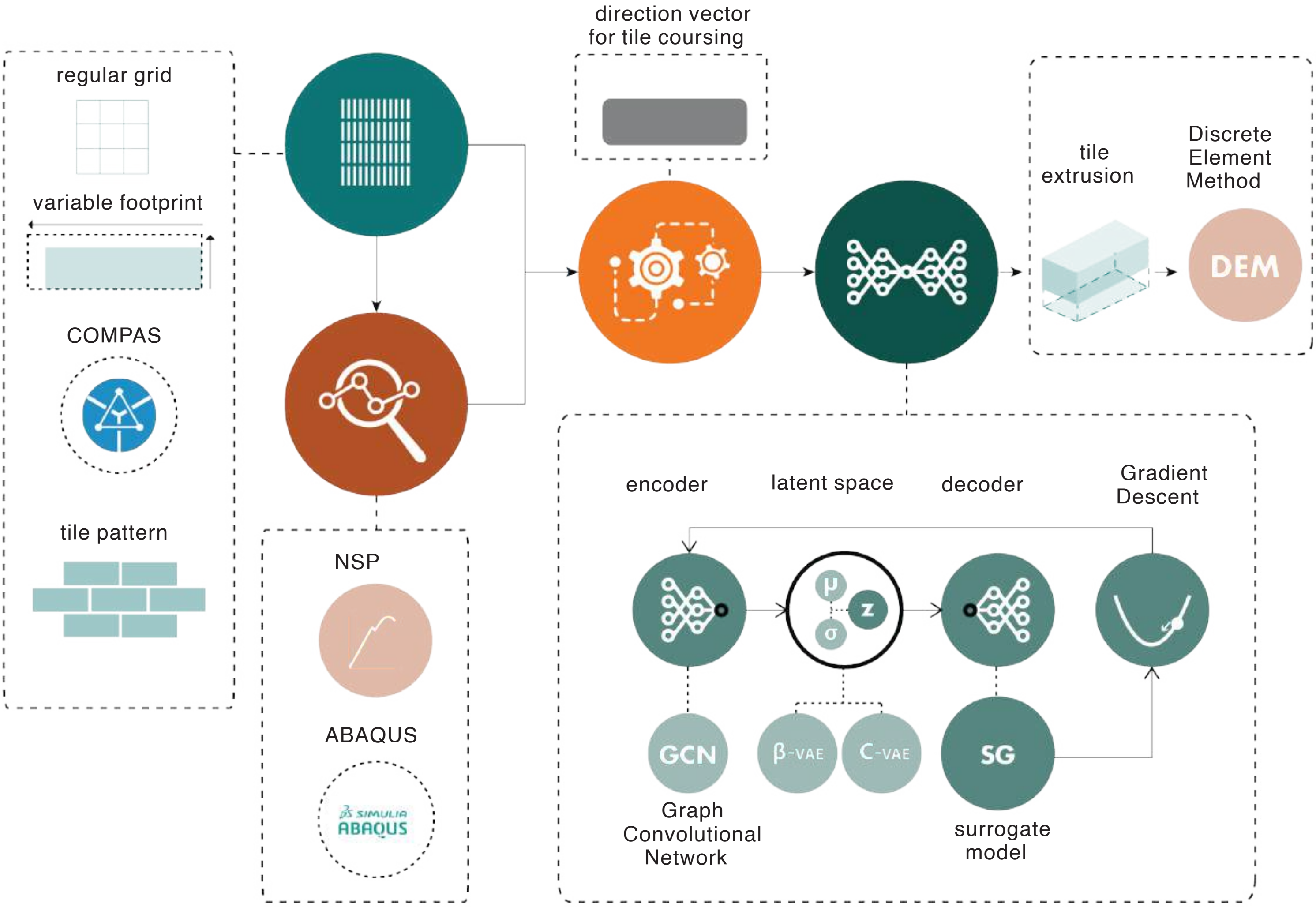


FIGURE 190: Workflow 3. (Own Work)

10.2.3 Material Properties of the DEM model for WF3

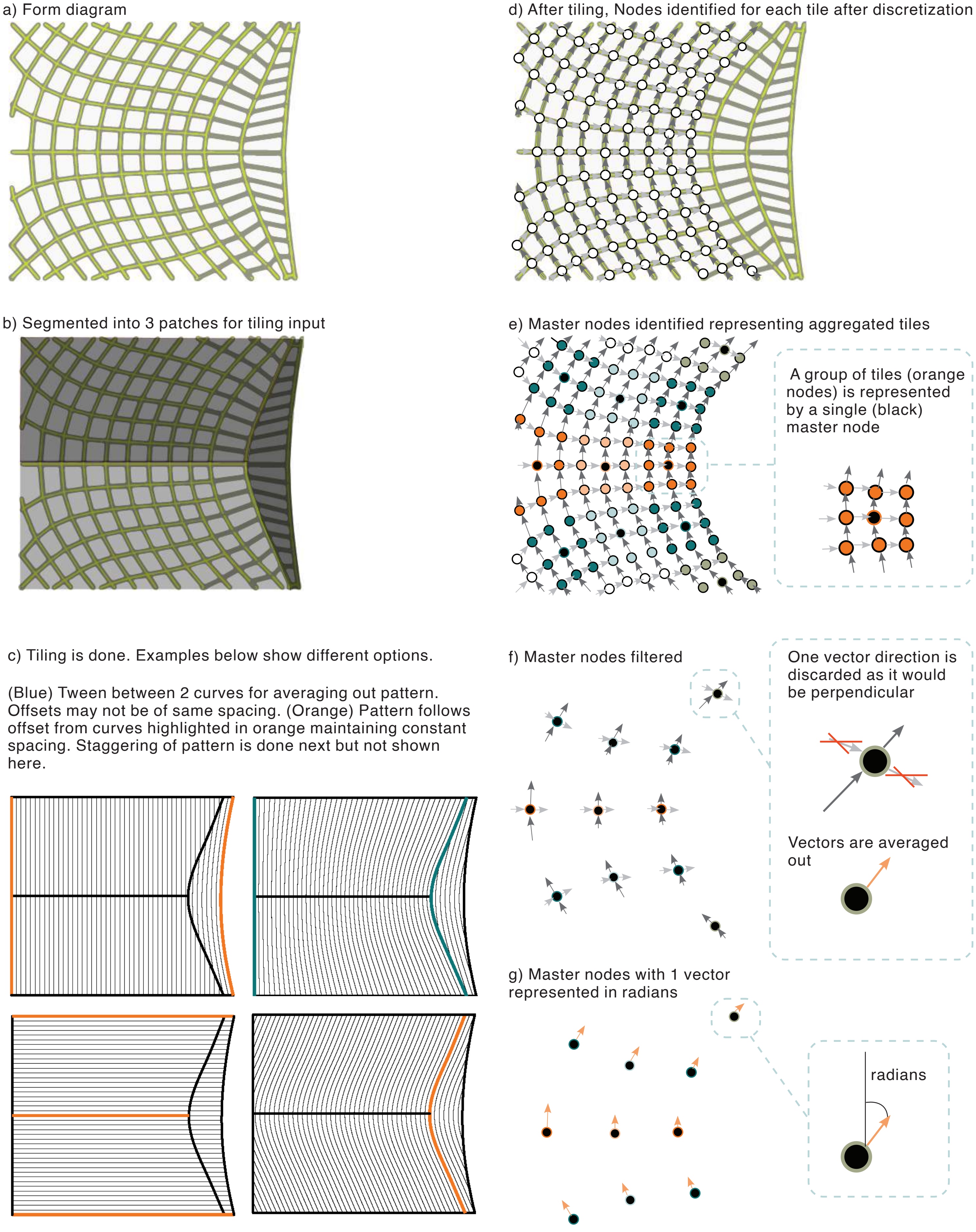
To carry out the analysis on a discrete model (DEM), the brick and mortar will behave independently. The properties for that have been tabulated below in Table 2.

			Discrete model		
		Units	Masonry	Mortar	Interface / joints
Young’s Modulus	E	GPa	6.2	0.370	-
Possion ratio	v	-	0.25	-	-
Density	ρ	kg/ m3	1800	-	-
Tension	Gfl	N/ mm	-	-	-
	ft	MPa	-	-	0.3
Compression	Gfc	N/ mm	-	-	-
	fc	MPa	-	-	9.1
Flexure	$fflex$	MPa	-	-	-
Cohesion	c	MPa	-	-	0.3
Friction angle	φ	deg	-	-	38

TABLE 23: Material property for Discrete model. Material properties of the discrete model are retrieved from Oktiovan, Y. P., Davis, L., Wilson, R., Dell’Endice, A., Mehrotra, A., Pulatsu, B., & Malomo, D. (2023). Simplified Micro-Modeling of a Masonry Cross-Vault for Seismic Assessment Using the Distinct Element Method. International Journal of Architectural Heritage, 1–34. https://doi.org/10.1080/15583058.2023.2277328

10.2.4 Data Structuring Strategy for WF3

For vectors representing tile direction, it would become far too computationally expensive to represent every tile separately as a node. Therefore, similar tiles may be grouped and represented as a single node. It would be logical that the tiling pattern can be based on the meshing pattern so nodes are based on aggregations of similar tiles according to position and direction (as indicated in Patterning Approach 1). This is shown in Figure 31. The tiling direction is to be represented by a vector in radians.



a) Form diagram

b) Segmented into 3 patches for tiling input

c) Tiling is done. Examples below show different options.

(Blue) Tween between 2 curves for averaging out pattern. Offsets may not be of same spacing. (Orange) Pattern follows offset from curves highlighted in orange maintaining constant spacing. Staggering of pattern is done next but not shown here.

d) After tiling, Nodes identified for each tile after discretization

e) Master nodes identified representing aggregated tiles

A group of tiles (orange nodes) is represented by a single (black) master node

f) Master nodes filtered

One vector direction is discarded as it would be perpendicular

Vectors are averaged out

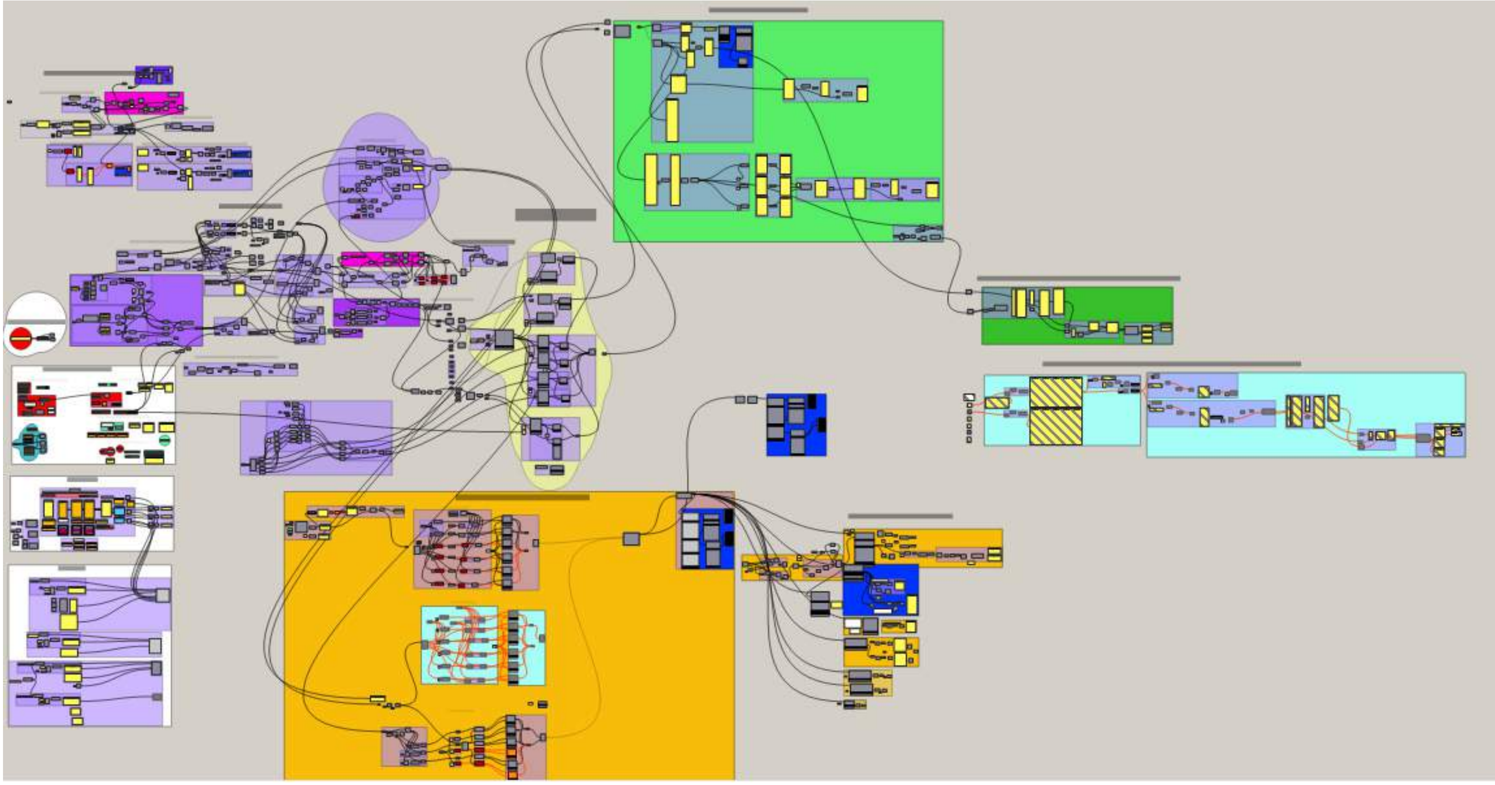
g) Master nodes with 1 vector represented in radians

radians

FIGURE 191: Representation of tiles for Data Structuring. (Own Work)

10.3 GRASSHOPPER SCRIPT (PERFORMANCE EVALUATION)

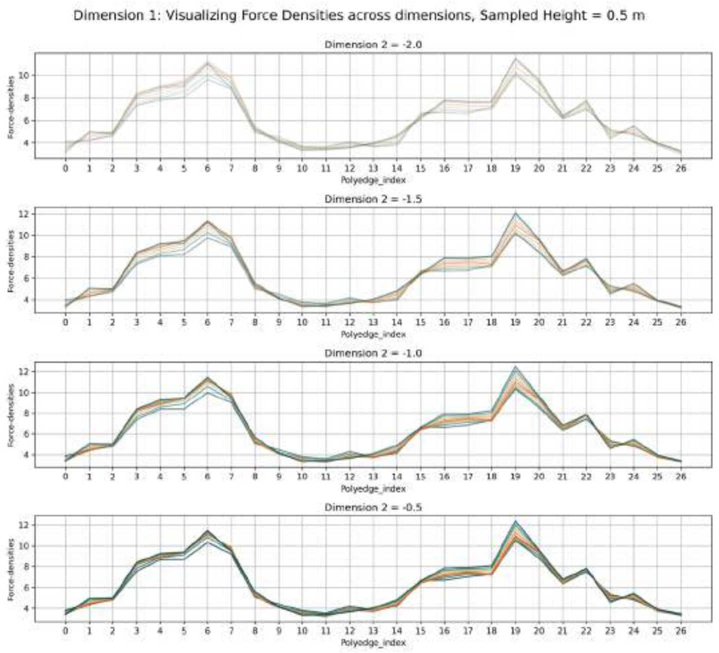
For the script, See the repository link.



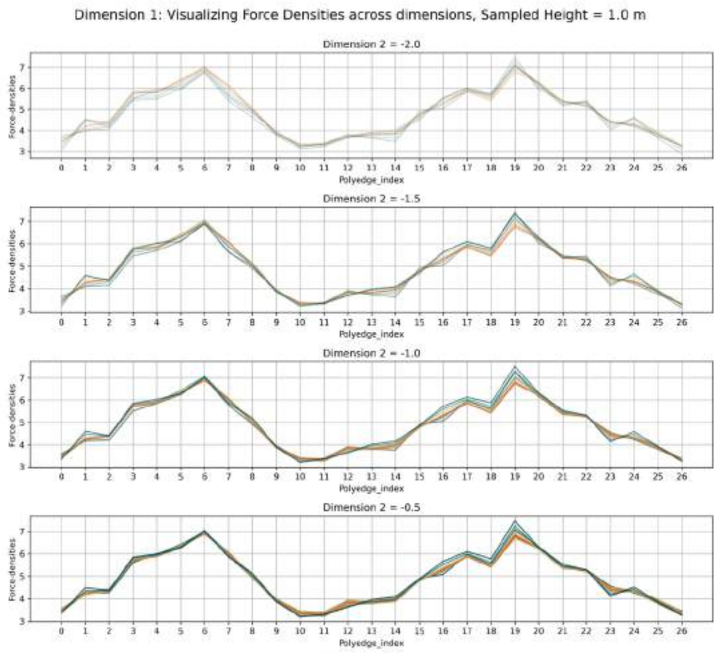
10.4 RESULTS: CONDITIONAL VAE

This is for Sinusoidal Positional Encoding.

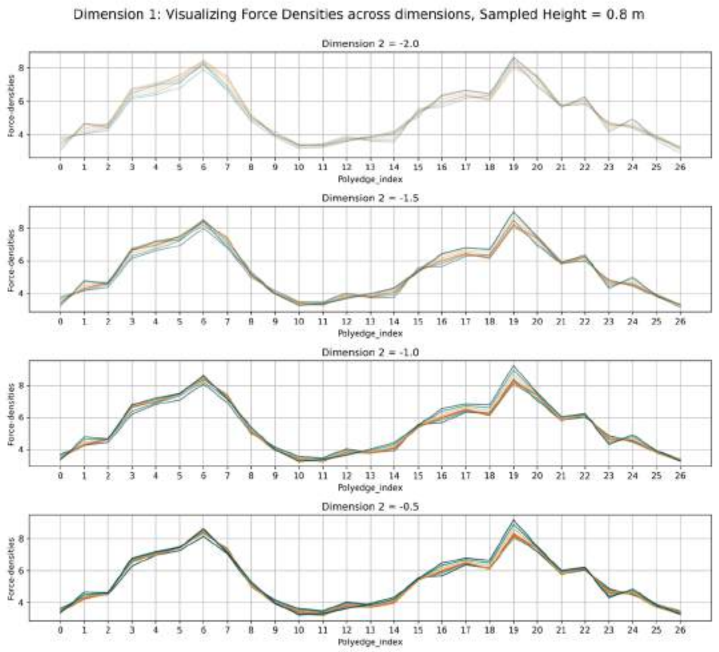
a) label = 0.5m



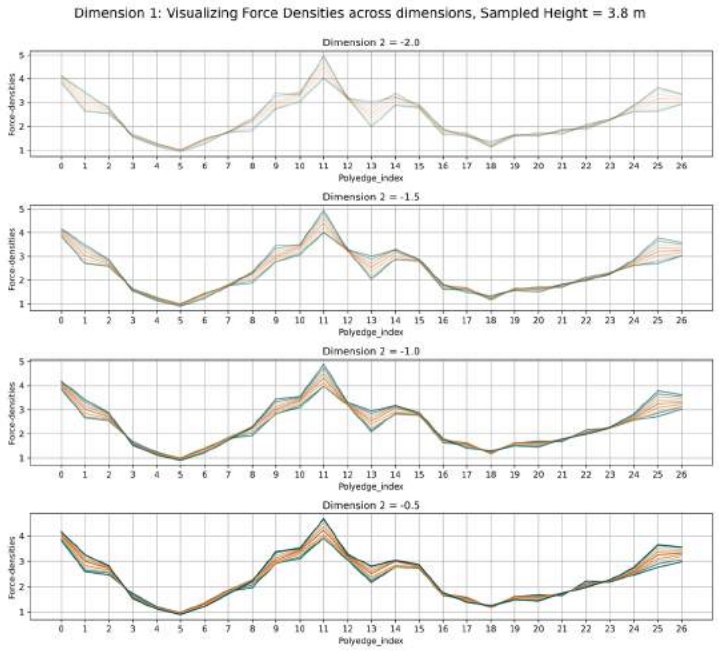
c) label = 1.0m

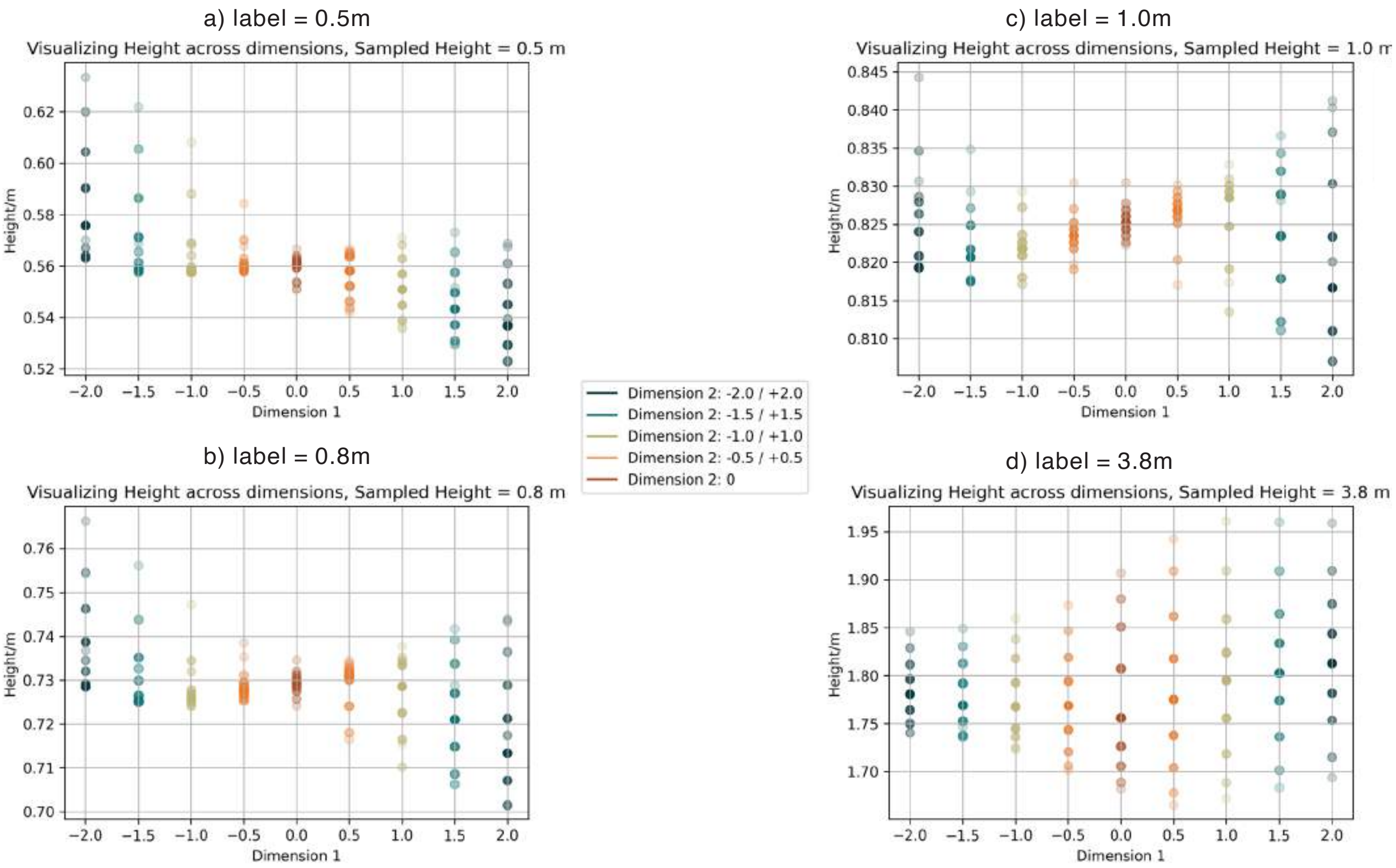


b) label = 0.8m



d) label = 3.8m





a) label = 0.5m

c) label = 1.0m

b) label = 0.8m

d) label = 3.8m