**Crashworthiness optimization of NPR structure with Artificial Intelligence**

A Thesis Report

**Submitted in Partial Fulfilment of the Requirements for the Degree**

**Bachelor of Technology**

**in**

**Aerospace Engineering**

**Submitted by**

**Mainak Mallick (Examination Roll No. 511318017)**

**Tarun Kurakula (Examination Roll No. 511318018)**

**Dipayan Parbat (Examination Roll No. 511318019)**

**Under the guidance of**

**Dr. Niloy Khutia**

**Associate Professor**

***Department of Aerospace Engineering and Applied Mechanics, Indian Institute of***

***Engineering Science and Technology, Shibpur***



**Department of Aerospace Engineering and Applied Mechanics**

**Indian Institute of Engineering Science and Technology, Shibpur**

HOWRAH – 711103, INDIA

**May – 2022**

**DECLARATION**

#### We certify that

#### The work contained in this thesis is original and has been done by me under the guidance of my supervisor.

#### The work has not been submitted to any other Institute for any degree or diploma.

#### I have followed the guidelines provided by the Institute in preparing the thesis.

#### 4. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

#### 5. Whenever I have used materials (data, theoretical analysis, figures and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references

**(Mainak Mallick)**

**Examination Roll No. 511318017**

**(Tarun Kurakula)**

**Examination Roll No. 511318018**

**(Dipayan Parbat)**

**Examination Roll No. 511318019**

Date:18/05/2022

Place: IIEST, Shibpur

Department of Aerospace Engineering and Applied Mechanics

Indian Institute of Engineering Science and technology, Shibpur



##### FORWARD

#### I hereby forward the thesis report entitled **“**Crashworthiness optimization of NPR structure with Artificial Intelligence” prepared by Mainak Mallick (Examination Roll No. 511318017), Tarun Kurakula (Examination Roll No. 511318018) and Dipayan Parbat (Examination Roll No. 511318019) under my guidance and supervision in partial fulfilment of the requirements for the award of the degree of “Bachelor of Technology in Aerospace Engineering” in the Department of Aerospace Engineering and Applied mechanics, Indian Institute of Engineering Science and Technology, Shibpur.

#### This report has been completed under my guidance in Department of Aerospace Engineering and Applied Mechanics, Indian Institute of Engineering Science and Technology, Shibpur.

**--------------------------------**

**(Dr. Niloy Khutia)**

Associate Professor,

Dept. of Aerospace Engineering and Applied mechanics, Indian Institute of Engineering and Science and Technology, Shibpur

|  |  |  |
| --- | --- | --- |
| ***Countersigned by***: | |  |
|  | |  |
|  | |  |
| ……………………………………. | | ............................................................ | |
| **(Dr.**  **Sujay Kumar Mukherjea )** | | **(Dr. Sudip Kumar Roy)** |
| ***Professor & Head*** | | ***Dean of Academic Affairs*** |
| Dept. of Aerospace Engineering and Applied Mechanics. Indian Institute of Engineering Science and Technology, Shibpur. | | Indian Institute of Engineering Science and Technology, Shibpur |
|  | |  | |

Department of Aerospace Engineering and Applied Mechanics

Indian Institute of Engineering Science and technology, Shibpur

****

##### CERTIFICATE of APPROVAL

The foregoing progress report is hereby approved as a creditable study of Engineering subject carried out and presented in a satisfactory manner to warrant its acceptance as a prerequisite for the Degree of **‘Bachelor of Technology’** in **Aerospace Engineering** in the Department of Aerospace Engineering and Applied Mechanics, Indian Institute of Engineering Science and Technology, Shibpur for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn there in but approve the report only for the purpose for which it is submitted.

**Board of Examiners**

***..............................................***

***..............................................***

***...............................................***

***..............................................***

##### **ACKNOWLEDGEMENT**

We take the opportunity to express our sincere gratitude and indebtedness to our B.Tech Thesis supervisor, Dr. Niloy Khutia, Associate Professor, Department of Aerospace Engineering and Applied Mechanics, Indian Institute of Engineering Science and Technology, Shibpur, Howrah – 711103 for his resourceful guidance, constant encouragement and enlightening discussions during the course of this research work without which it would not have been possible for us to reach to this point

We would like to extend our gratefulness to all the faculty members and staff of Department of Aerospace Engineering and Applied Mechanics, Indian Institute of Engineering Science and Technology, Shibpur ,to carry out the present study.

Finally, we would like to express our gratitude towards parents, for their unconditional support, love and affection. Their encouragement and never ending kindness made everything easier to achieve.

**CONTENT**

[**1.** **INTRODUCTION** 6](#_Toc103724567)

[**2.** **LITERATURE REVIEW** 7](#_Toc103724568)

[**3.** **PROBLEM STATEMENT & SIMULATION WORKFLOW** 10](#_Toc103724569)

[3.1. The problem Statement: 10](#_Toc103724570)

[3.2. NPR Structure: 10](#_Toc103724571)

[3.3. Properties of GFRP: 11](#_Toc103724572)

[3.4. Python Scripting for Unit Cell: 13](#_Toc103724573)

[3.5. The Assembly Setup: 14](#_Toc103724574)

[3.6. The Overall Workflow: 16](#_Toc103724575)

[**4.** **FE MODEL VALIDATION** 17](#_Toc103724576)

[4.1. FE Model validation using Representative Volume Element: 17](#_Toc103724577)

[4.2. FE Model validation for the 4X4 cell assembly: 18](#_Toc103724578)

[**5.** **DATA POST-PROCESSING** 20](#_Toc103724579)

[**6.** **NEURAL NETWORKS AND PREDICTIONS** 23](#_Toc103724580)

[6.1. Basic Mechanism – 23](#_Toc103724581)

[6.2. Single Layer and Multilayer Perceptron- 24](#_Toc103724582)

[6.3. Neural Net Optimization Algorithms – 26](#_Toc103724583)

[6.4. Implementation of Latin Hypercube Sampling – 31](#_Toc103724584)

[6.5. Configuration of deployed neural net – 33](#_Toc103724585)

[**7.** **PARTICLE SWARM ALGORITHM AND OVERALL WORKFLOW** 34](#_Toc103724586)

[7.1. Particle Swarm Optimization Algorithm 35](#_Toc103724587)

[**8.** **RESULTS AND DISCUSSIONS** 37](#_Toc103724588)

[**9.** **CONCLUSION** 43](#_Toc103724589)

[**10.** **REFERENCES** 44](#_Toc103724590)

# **INTRODUCTION**

This project is based on the optimization of a novel Negative Poisson’s Ratio (NPR) cell assembly structure generally used in the sub-cargo floor area of an aircraft. While making the structure we must consider two constraints. First one is weight as it affects the expenditure and the second one peak reaction force at it affects passenger safety. Here, we have tried to optimize the design of a NPR cell assembly which will absorb maximum energy per unit weight while keeping the peak crushing force as a constraint.

For modelling and simulation ABAQUS (FE modelling and simulation software) was used along with PYTHON scripting, for changing the design parameters easily and automatically. For predicting the data of other design combinations from a set of extracted simulation results, we have used neural networks based out of Tensor Flow module, written in PYTHON programming code.

Then, the trained neural network was fed into conventional Non-Sorting Genetic algorithm (NSGA) and the optimum results were found. Though in the present study only NSGA and two objective optimization were applied, we can easily apply the same technique with more objective and constraint functions along with more advanced algorithms like Grey Wolf or Particle Swarm for more accurate and quick convergence.

# **LITERATURE REVIEW**

Crashworthiness is the ability of an aircraft structure and its internal systems to protect occupants from injury in an event of crash. Specifically, it means that the integrity of the passenger cabin should be maintained, so that the passengers can survive in case of a crash and fires are prevented. To prevent or limit the injury to the body, crash test simulations are required.

Dynamic Response Index (DRI) is calculated using relative motion equations. The DRI is used to evaluate the potential for spinal injury during aircraft crashes and also in the operation of high-speed marine craft.

A typical section of the fuselage for a transport aircraft includes frames, stringers, skin, the passenger floor, the cargo floor and struts or stanchions. During a crash landing, the kinetic energy should be dissipated in a controlled way by the crushing of the structure below the passenger floor. Energy is dissipated by stanchions connecting the frames and beams on the cargo floor, by longitudinal sine wave beams below the subfloor beams or sine wave beams below cargo floor beams[1].

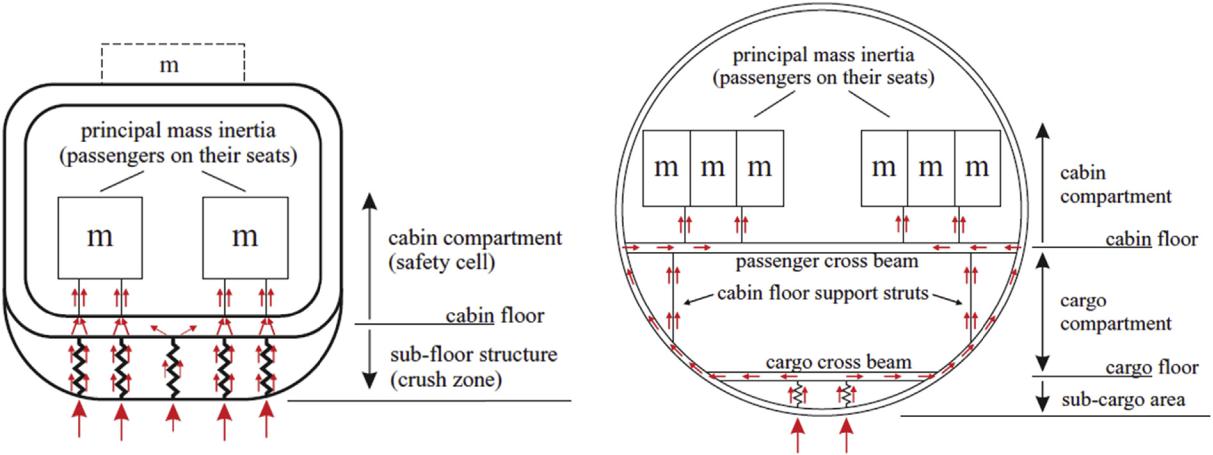


Fig. 1 - Effect of size on structural and load transfer for helicopters and small aircrafts (left) and transport aircrafts (right)[1]

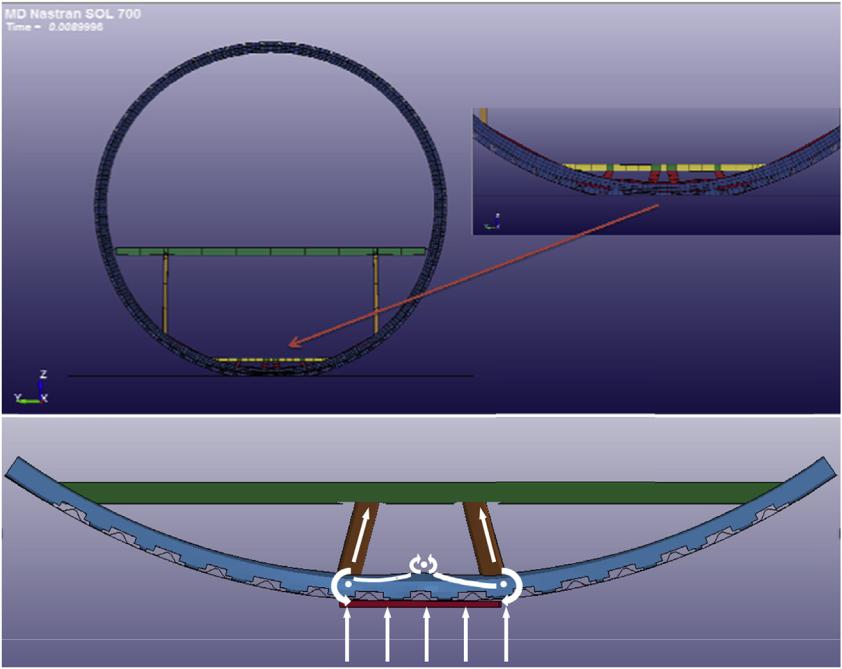
Airplane crashworthiness is dominated by the impact response characteristics of the fuselage. No specific rule addresses compliance for the impact response characteristics which could be considered as survivable crash conditions[2].

Ideally, in case of an emergency landing, landing gear, aircraft structure and occupant seats must all be designed to work together in the attempt to absorb the aircraft kinetic energy, slowing down the occupant to rest without injurious loading.

The first studies on the crashworthiness and the behavior of aero-nautical structures during a collision were introduced in the 50s, when the NACA (National Advisory Committee for Aeronautics) carried out the first tests in order to identify the mechanisms that lead to a post-impact fire; these studies, performed on the military transport aircraft Commando C46, were useful to determine the effectiveness of a fuel system “impact-resistant", in preventing the development of fire after the impact and therefore having serious consequences for the occupants.

Acceptable crashworthiness capabilities under foreseeable survivable impact events may be demonstrated via a combination of tests. Finite Element analysis is a valid alternative to follow.

Fig. 2 - Before and after the collision of fuselage at the ground[1].

The adopted method in the research activity has been reported in [1]. It shows the approach used during the certification process, in order to design, evaluate and optimize the crashworthiness behavior of composite structures, and to develop an evaluation methodology (experimental and numerical) and predictable computational tools following a certification by analysis approach.

After defining the requirements for the structural configuration and identifying the variables, the stages of material characterization and energy absorption mechanism have allowed to increase benchmark data for putting in place constitutive models and failure theory.

The focus work is to describe the development of a lightweight composite crash absorber, which can be used in commercial aircraft to improve the crashworthiness behavior. Component tests and complete crash tests under normal loading conditions provided consistent results within all requirements and showed a very high degree of robustness and reproducibility of the results. Although the absorber in this study was designed and tested for specific load requirements, it can be dimensioned to different crush load levels by an appropriate choice of thickness.

The output results have been used to design, manufacture and test a full-scale lower lobe demonstrator in composite material to install on a commercial aircraft that are traditionally in metallic material. Furthermore, this solution provides an innovative configuration tested to evaluate the energy absorption performance of the sub-cargo concept and permit to aeronautical industrial to go beyond the state of art[3].

These results have allowed to define a detailed FE model which is able to investigate the crash absorbing peculiarities in the attempt to choose the optimum compromise between materials, layups and installation con-figurations, and finally driving to design an innovative portion of the lower lobe fuselage, which was manufactured and subjected to low velocity drop test. In the final step, the last configuration was validated by the numerical simulations aimed to evaluate the part of the structure able to absorb the energy during the impact.

This building block approach helps the definition of a method which will be able to reproduce the actual conditions of the crashworthiness certification and to design a partial tool meeting the emergency landing requirement, starting from the outcomes obtained on the entire scale lower lobe and related to decreased velocity impact. The final stage moves in the direction of a fuselage with crashworthiness overall performance such as to respect the design requirements of a survivable volume and acceptable loads to the occupants[4].

# **PROBLEM STATEMENT & SIMULATION WORKFLOW**

## The Problem Statement:

The problem statement was to optimize the design parameters of the NPR assembly, so that,

The objective functions:

* ***The energy absorption of the structure is maximum.***
* ***The weight of it is minimum.***

And The constraint function:

* ***The peak reaction force stays under a certain limit.***

The variables:

* *The variables are , t, w in a certain bound*

In order to make the simulating environment less complicated the problem setup is simplified as shown in the figure below.

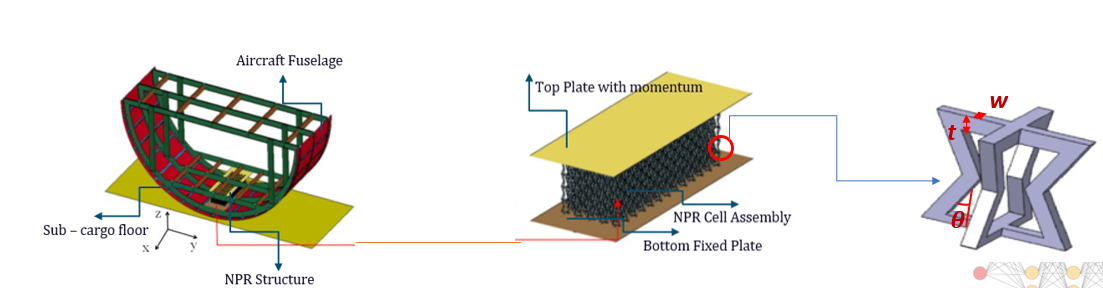


Fig. 3 – Problem Statement Simplification[5]

## NPR Structure:

Materials with a negative Poisson’s ratio, also known as auxetic materials, exhibit unusual and counterintuitive mechanical behaviour—becoming fatter in cross-section when stretched. Negative Poisson ratio solids easily undergo volume changes. By contrast, rubbery materials easily undergo shear deformation but are much stiffer in relation to volume changes. Foams with negative Poisson's ratios were produced from conventional low-density open-cell polymer foams by causing the ribs of each cell to permanently protrude inward, resulting in a re-entrant structure.

Negative Poisson's ratio effects can result from non-affine deformation, from certain chiral microstructures, on an atomic scale, or from structural hierarchy Negative Poisson's ratio materials can exhibit slow decay of stress according to Saint-Venant’s principle. Such materials were later called as anti-rubber, auxetic or dilatational. These materials are an example of external materials. They are also known as meta-materials.

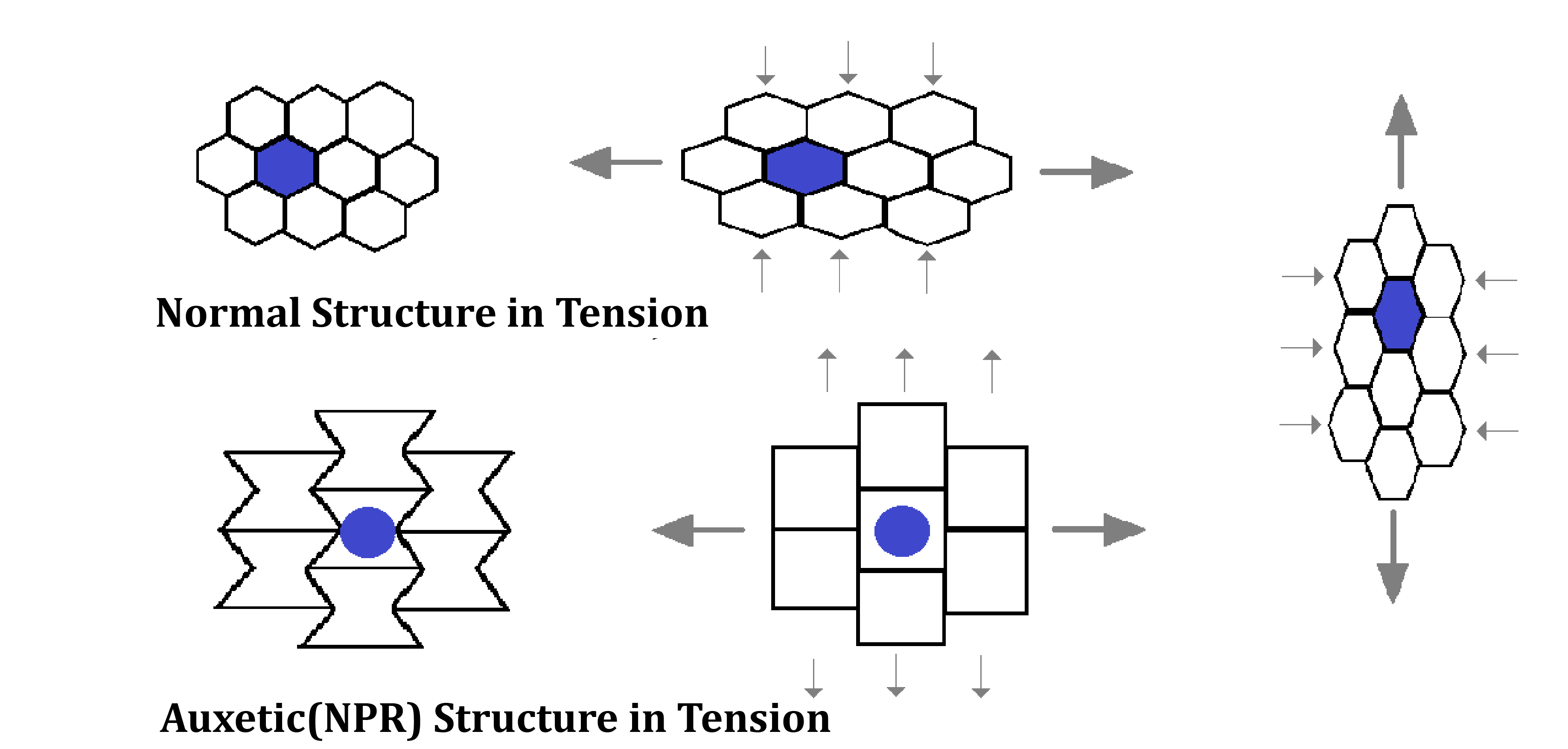


Fig. 4 – Comparison of different structures[6]

## Properties of GFRP:

Our chosen material for the honeycomb structure is Glass-Fibre Reinforced Plastic or GFRP. Glass(fibre) Reinforced Plastic (GFRP) is a composite material that consists of a polymer matrix and glass fibres. GRP becomes a material that resists both compressive and tensile forces very well and due to this reason, it is so strong. Glass fibre is isotropic in nature and high commonly utilized filament. E-Glass, S-Glass, C-Glass and AR-glass are the popular kinds of glass fibre. High strength, well resistant to water and chemicals with low cost are the main characteristics of glass fiber. Relatively low costs compared with other types of FRPs make glass fibre the most generally applied in construction industry. Nevertheless, a comparatively low elastic modulus, low resistant to alkaline with low long-term strength due to stress rupture are the major drawbacks for glass fibre. For the situation that required better resistance to alkaline, the supposed AR-glass could be utilized. The density and the stress vs strain graph of GFRP have been taken in the present analysis.[7]

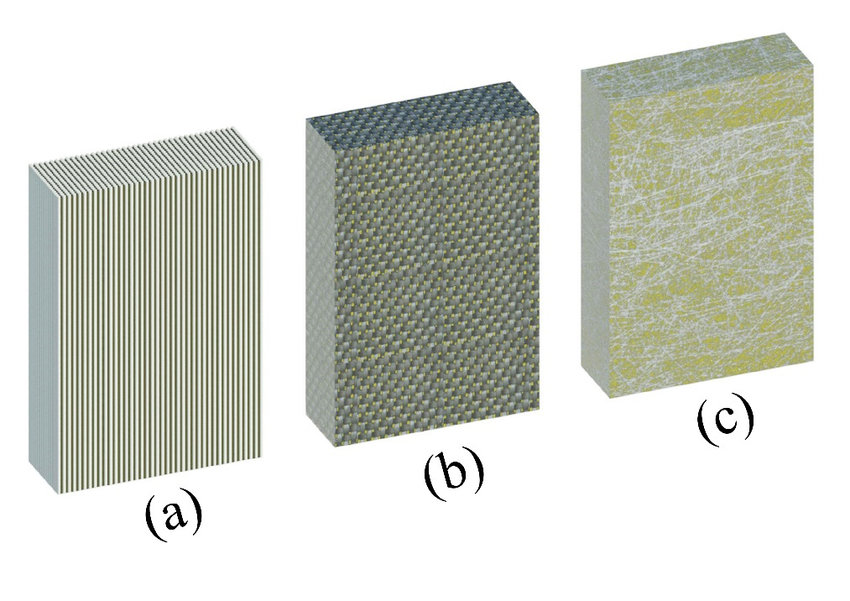


Fig. 5 – Different types of GFRP structure[2]

The extracted stress strain curve from a reference research work in the elasto-plastic domain is given below.

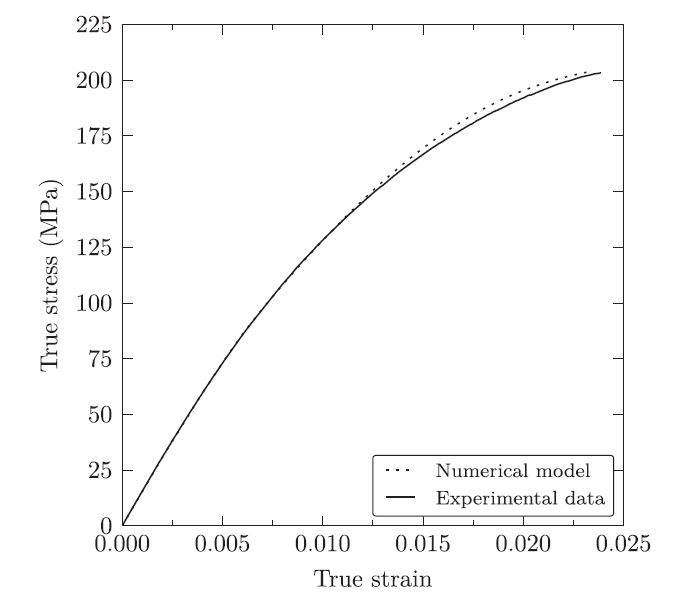


Fig. 6 – Stress strain structure for used GFRP composite[8]

## Python Scripting for Unit Cell:

The initial values of the dimensions of the structures extracted from the research work have been provided in the table below:

|  |  |
| --- | --- |
| Initial variables | Values |
| Length(l) | 10mm |
| Thickness(t) | 3mm |
| Angle(theta) | 30 deg |

The Python script which is implemented for the creation of the single cell is shown below:

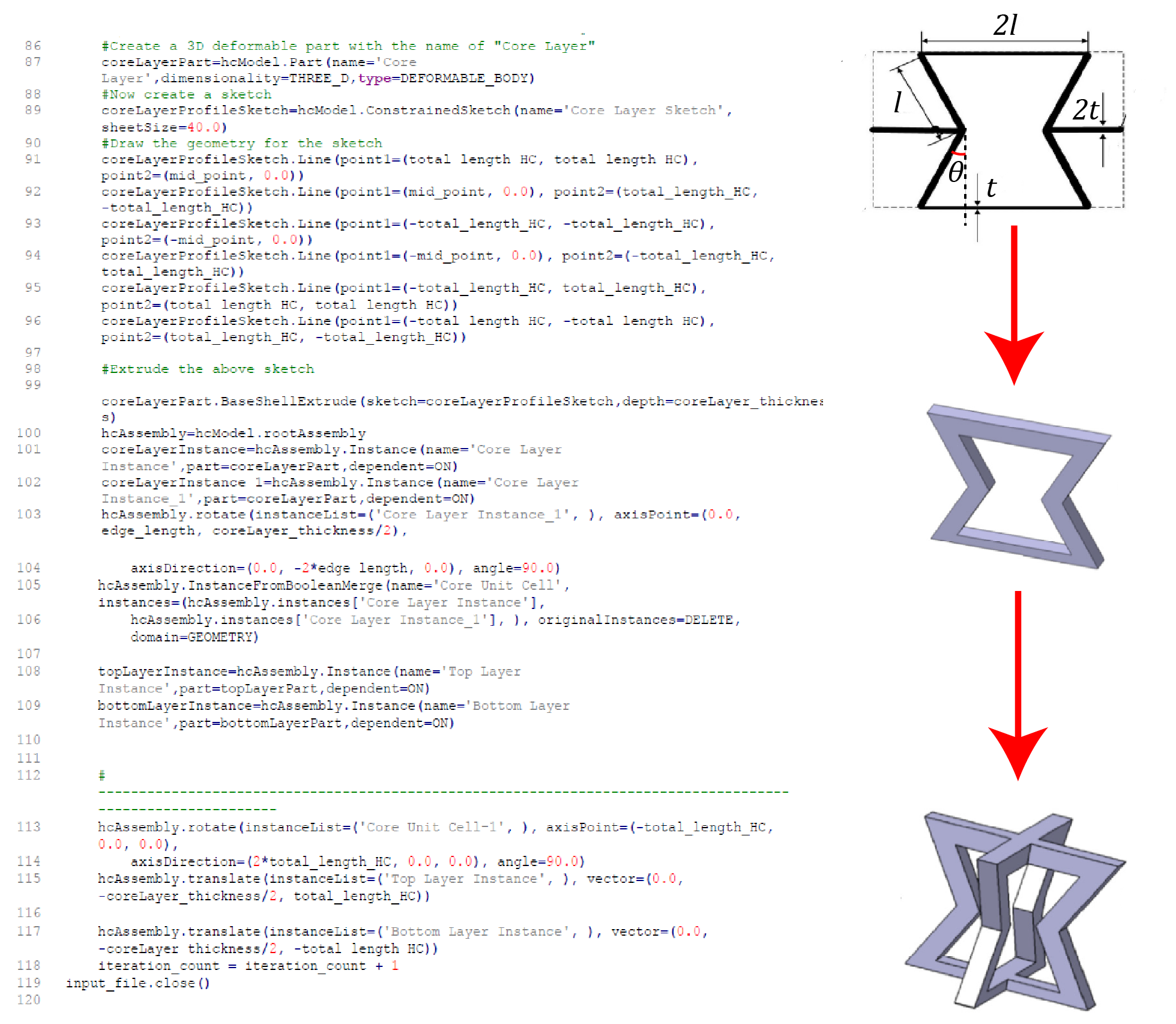


Fig. 7 – Python scripting workflow to create an unit cell

## The Assembly Setup:

Before going to recreate the total assembly in Abaqus, we first tried to create the total NPR assembly in Solid Works and it is provided below for true scale visualization:

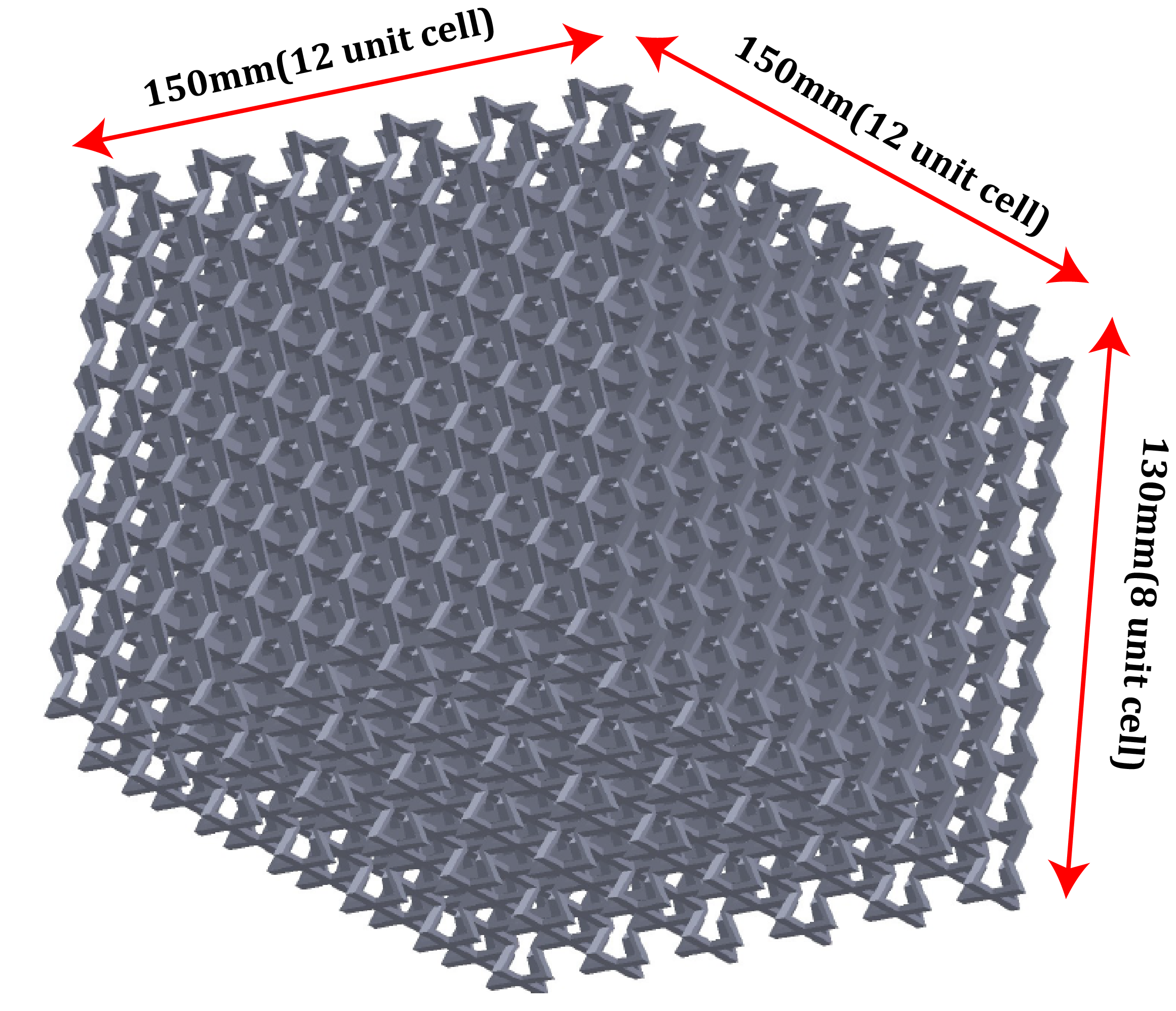


Fig. 8 – Full scale NPR core layer

Now, for the full-scale assembly setup in Abaqus:

we move on to setting up the Abaqus simulations. After running the python script with the appropriate input file, the assembly will be generated in Abaqus assembly module. Then we provide the material properties and assign it to the section of Honeycomb structure. Now the crash test has been set up, where we will give the upper plate a momentum and it will crash towards the lower plate with the honeycomb unit cell in between.

* The total time is set as 0.001 sec.
* The whole process will have a friction coefficient of 0.2 in between all surfaces.
* We will tie the upper and lower plate with the honeycomb structure’s upper and lower surfaces.
* All degrees of freedom of the lower plate will be constrained.
* The upper plate will only be able to move in one direction.
* The value of the velocity given to the upper plate is 11.6m/s .
* The inertial mass given to the top layer is .275tonne.

In the following figure we can see the meshed full scaled structure created in Abaqus:

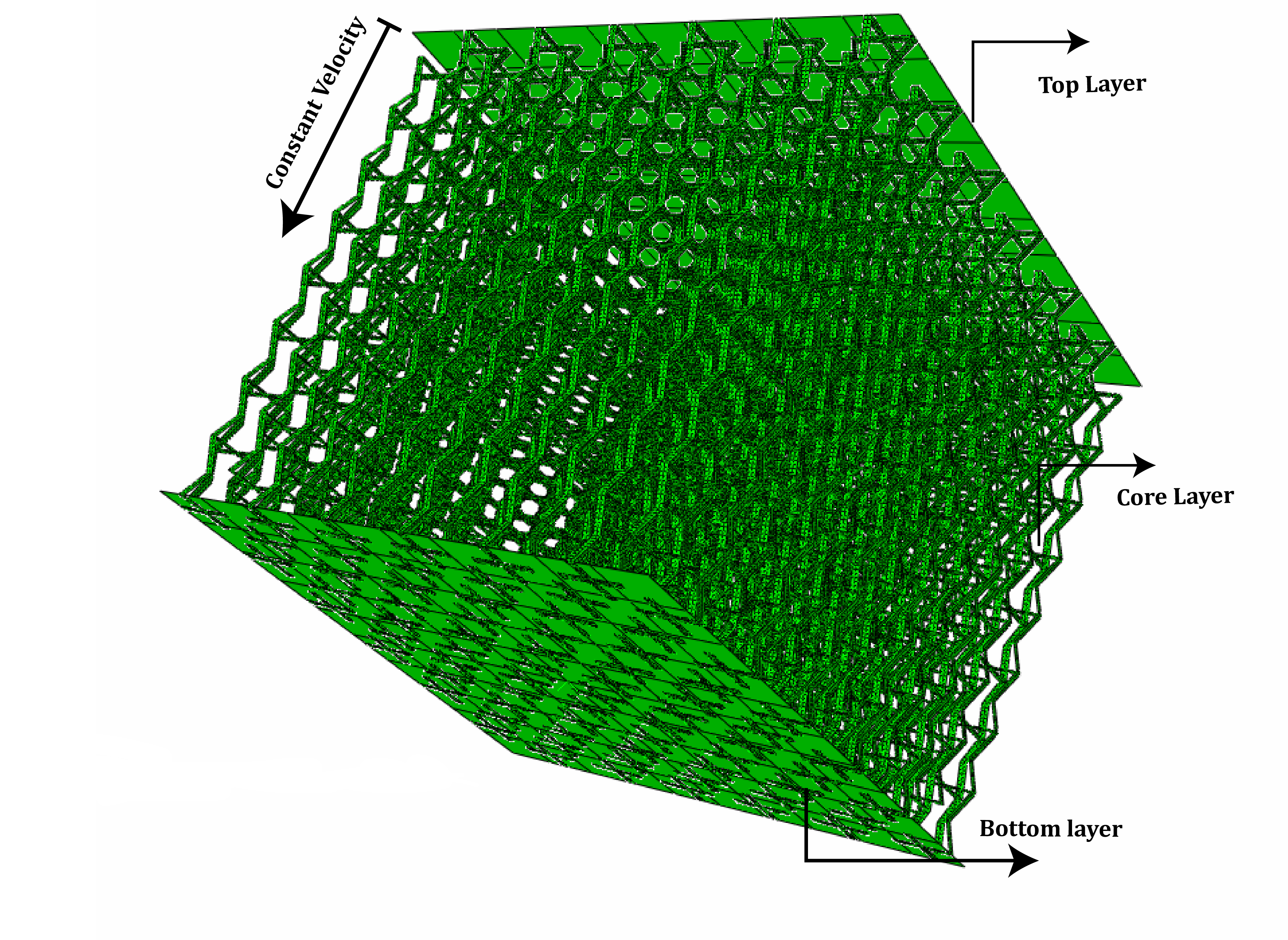


Fig. 9 – Full scale NPR assembly adaptation in ABAQUS

For meshing of our NPR structure, we have used a shell element because the ratio of the thickness to main dimension is < 1/15. We have used the S4R type shell element with an element size of 0.4. S4R is one of the general-purpose, conventional shell elements available in Abaqus. These elements provide robust and accurate solutions to most applications.

* Allows transverse shear deformation.
* Converges to shear flexible theory for thick shells and classical theory for thin shells
* The element has several hourglass modes that may propagate over the mesh.
* Accounts for finite membrane strains and arbitrarily large rotations so they are suitable for large-strain analysis.

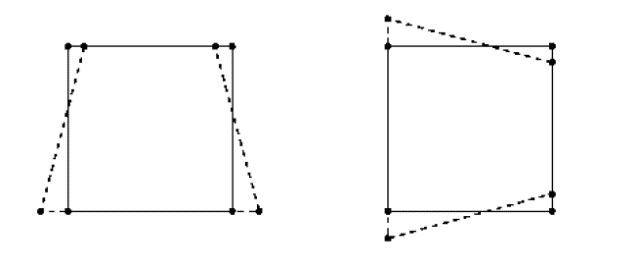


Fig. 10 – S4R element

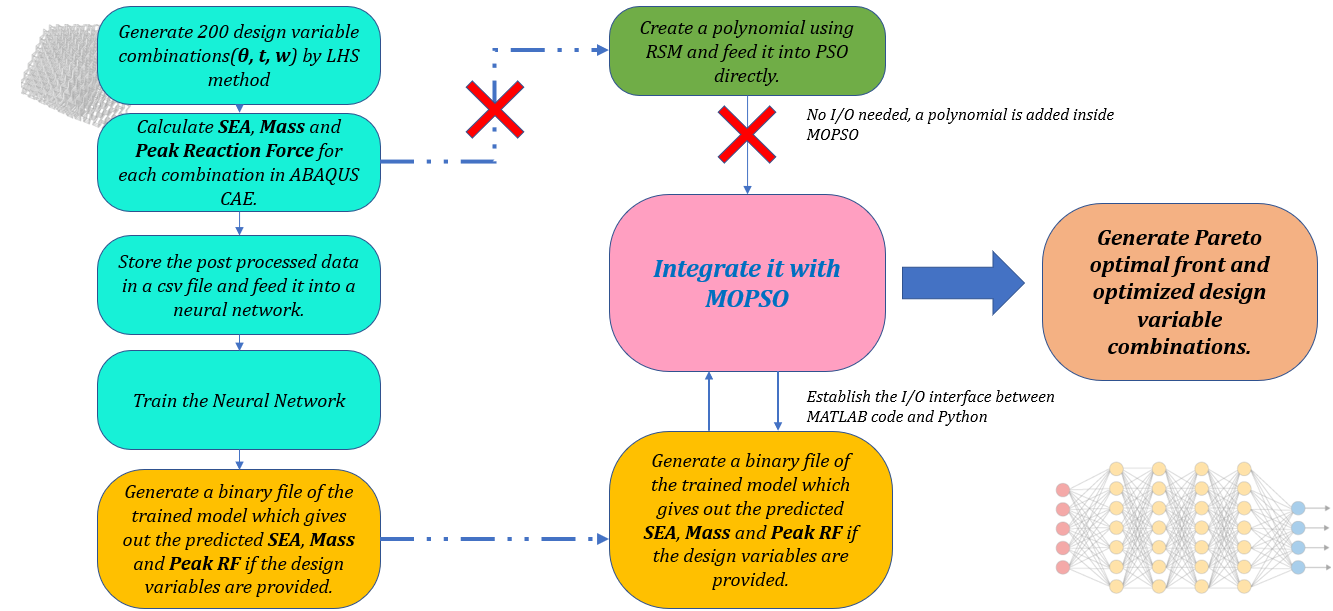
## The Overall Workflow:

Replicate the energy absorbing NPR structure

from reference research work

Validate the RVE FE model mechanical properties

with their experimental counterpart



Extract the SEA and Mass properties

for different theta combinations

Change the material to GFRP

Validate the same for the whole pattern

# **FE MODEL VALIDATION**

## FE Model validation using Representative Volume Element:

For FE model validation we have first checked with a single RVE model to find the effective Elastic Modulus by taking Nylon as material.

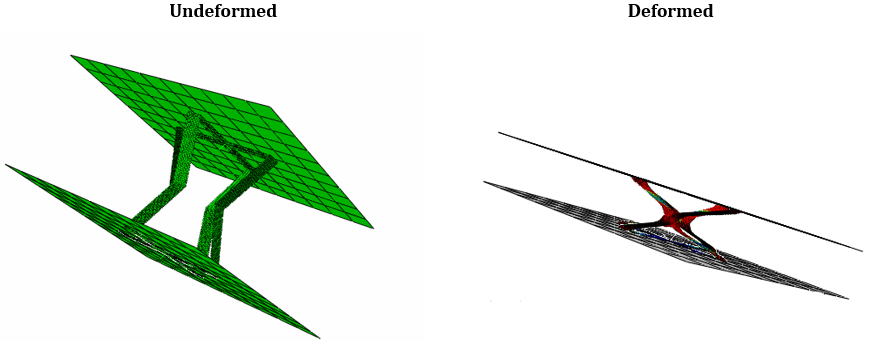


Fig. 11 - FE Model validation for Single cell

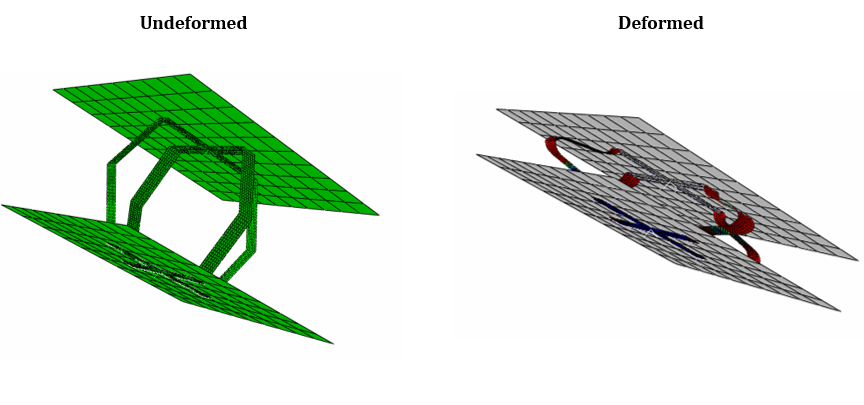


Fig. 12 - FE Model validation for Single cell

To calculate the equivalent elastic modulus, we have divided the average force obtained in the bottom plate to the average area of the plate. The strain is calculated by dividing the deformed length with total length.

The obtained value of the equivalent elastic modulus is 29.83 which gives us 9.2 % error when compared to the theoretical results obtained in the research work.

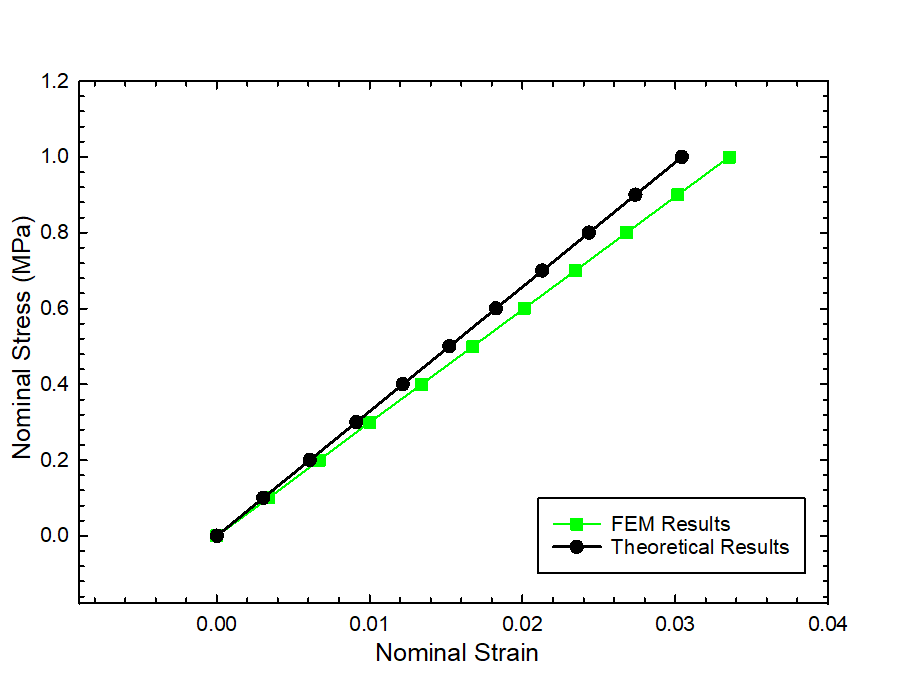


Fig. 13 – Nominal stress vs strain for single cell

## FE Model validation for the 4X4 cell assembly:

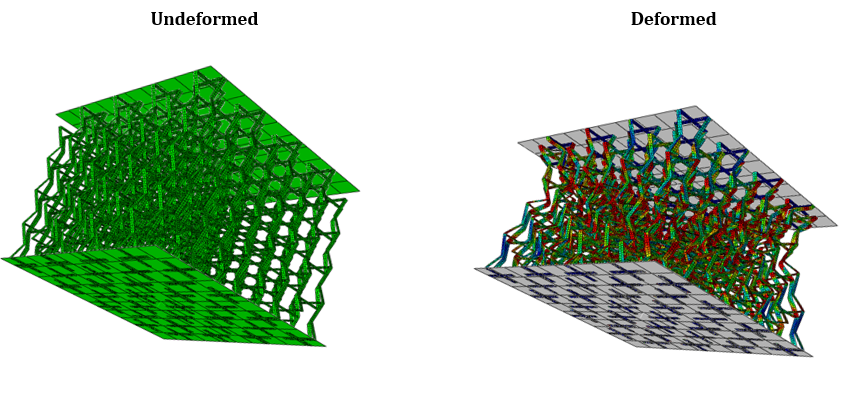


Fig. 14 - FE Model validation for 4X4 cell assembly

The value of the elastic modulus for a 4X4 assembly is also within the tolerable range when compared to the theoretical results.

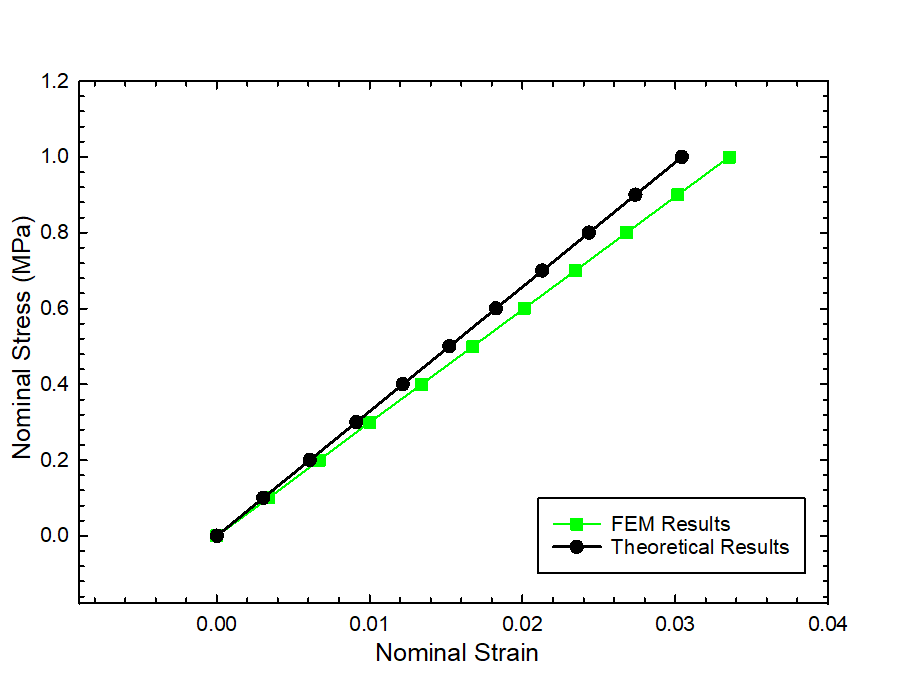


Fig. 15 – Nominal stress vs strain for 4X4 cell assembly

# **DATA POST-PROCESSING**

After the Abaqus simulations were done, we moved to the data collection and post processing part. In the problem statement we have stated that we are interested in mainly two sets of data for optimization purposes namely, Total Energy Absorbed and Reaction Force. In the Visualization module we have plotted and gathered these two sets of time varying data from History Output Requests. The plots are provided below.

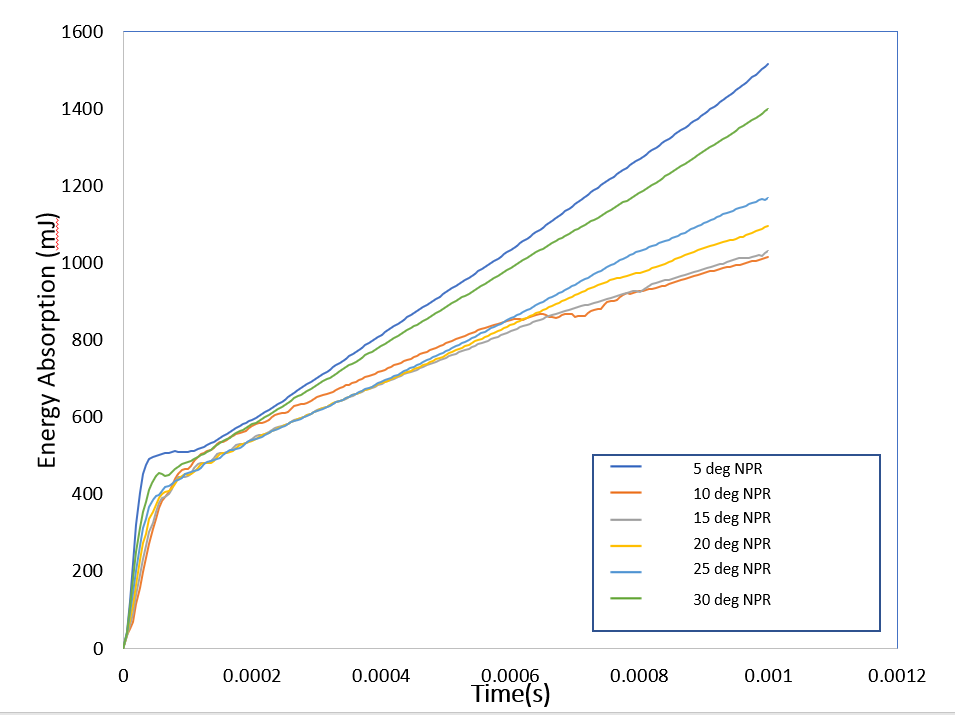


Fig. 16– Energy Absorption vs time for NPR cell

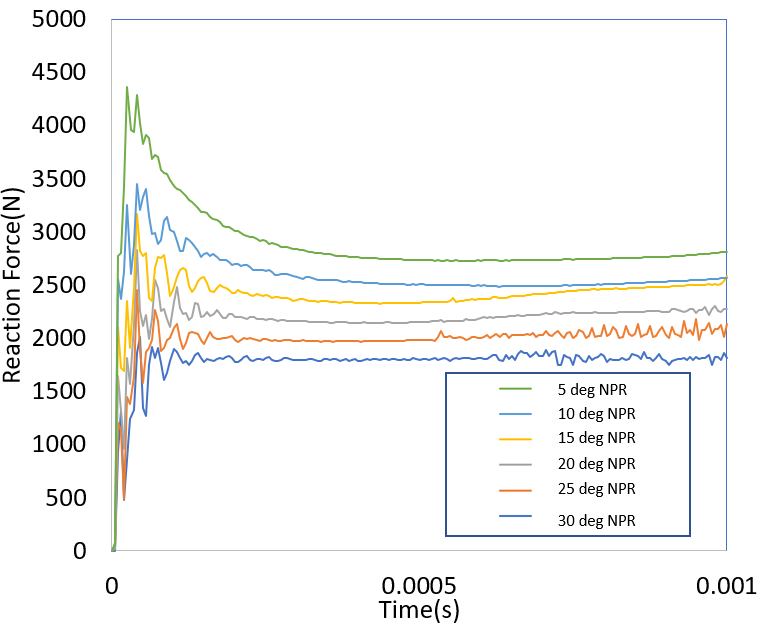


Fig. 17 – Reaction Force vs time for NPR cell

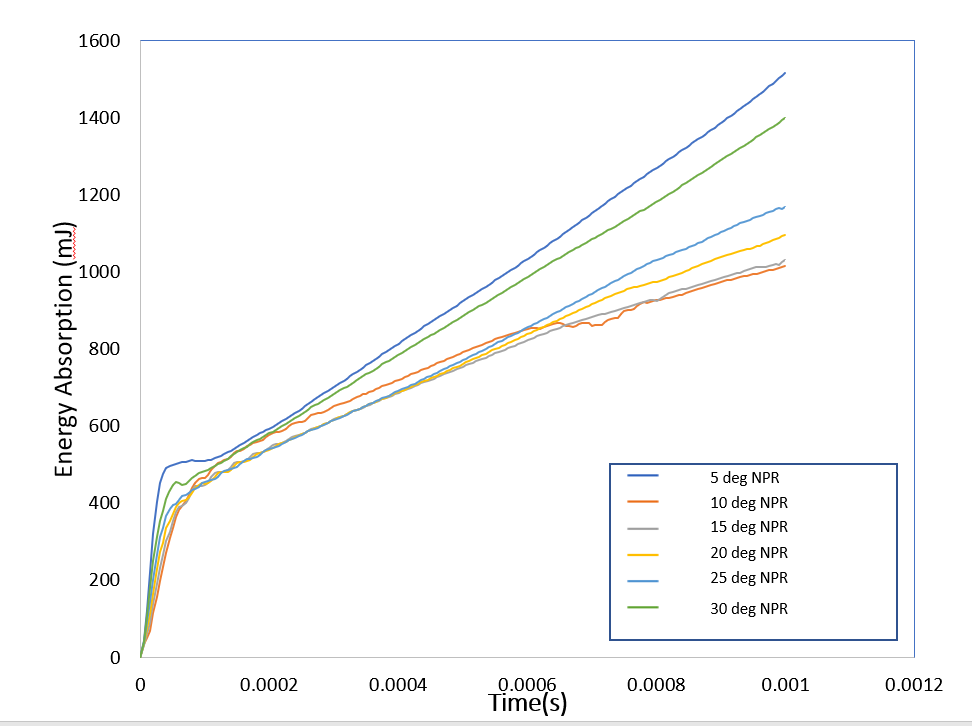


Fig. 18– Energy Absorption vs time for normal cell

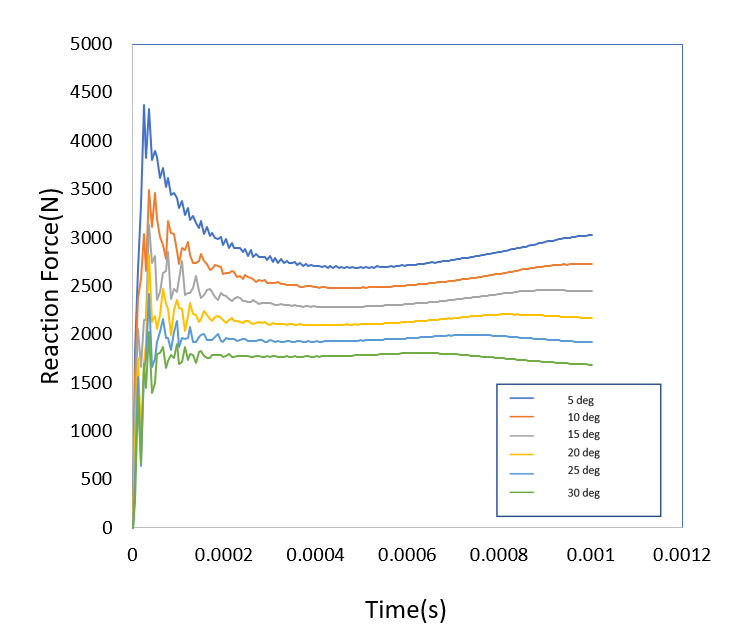


Fig. 19 – Reaction Force vs time for normal cell

# **NEURAL NETWORKS AND PREDICTIONS**

## Basic Mechanism –

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behaviour of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. It distinguishes itself from classical machine learning by the type of data that it works with and the methods in which it learns.

Machine learning algorithms leverage structured, labelled data to make predictions—meaning that specific features are defined from the input data for the model and organized into tables. Deep learning eliminates some of data pre-processing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction, removing some of the dependency on human experts. Then, through the processes of gradient descent and backpropagation, the deep learning algorithm adjusts and fits itself for accuracy, allowing it to make predictions about a new photo of an animal with increased precision.

Machine learning and deep learning models are capable of different types of learning as well, which are usually categorized as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilizes labeled datasets to categorize or make predictions; this requires some kind of human intervention to label input data correctly. In contrast, unsupervised learning doesn’t require labeled datasets, and instead, it detects patterns in the data, clustering them by any distinguishing characteristics. Reinforcement learning is a process in which a model learns to become more accurate for performing an action in an environment based on feedback in order to maximize the reward.

## Single Layer and Multilayer Perceptron-

* A perceptron is a neural network unit (an artificial neuron) that does certain computations to detect features or business intelligence in the input data.
* Perceptron was introduced by Frank Rosenblatt in 1957. He proposed a Perceptron learning rule based on the original MCP neuron.
* A Perceptron is an algorithm for the supervised learning of binary classifiers. This algorithm enables neurons to learn and processes elements in the training set one at a time.

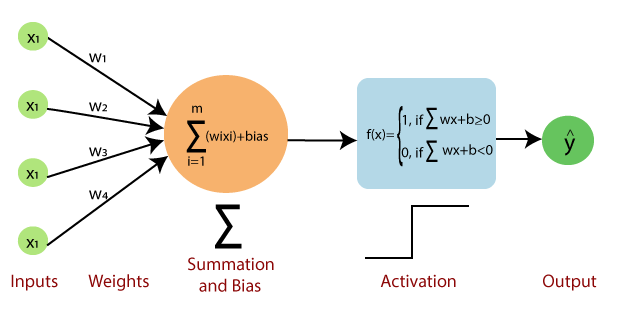
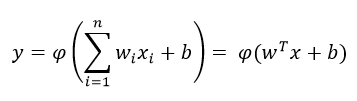


Fig. 20 – Single Layer Perceptron Model



(1)

Single-layer Perceptron’s can learn only linearly separable patterns. For classification we as Activation function as a threshold to predict class. And for Regression, we need not need the Activation function (Thresholding) or we can use a linear function to predict continuous value. Where **w** denotes the vector of weights, **x** is the vector of inputs, **b** is the bias and **φ** is the non-linear activation function.

Single-layer Perceptron is not able to figure out the nonlinearity or complexity of the data. So, researchers developed the Multilayer perceptron using the idea of the single-layer perceptron. Multiple Hidden layers are used to find the nonlinearity of the data. This instruction is also called a feed-forward network.

Multilayer perceptron’s train on a set of input-output pairs and learn to model the correlation (or dependencies) between those inputs and outputs. Training involves adjusting the parameters, or the weights and biases, of the model to minimize error. Backpropagation is used to make those weigh and bias adjustments relative to the error.

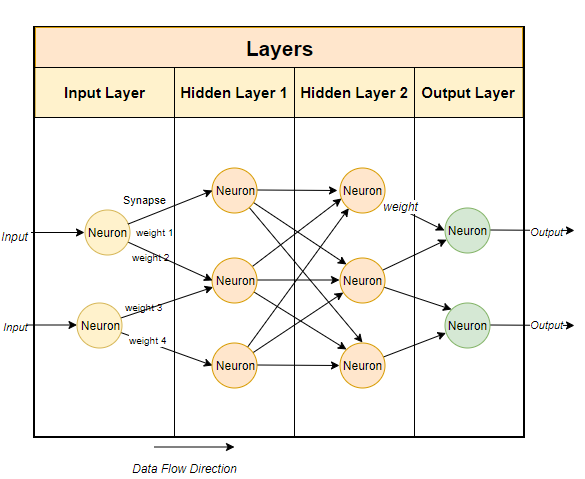
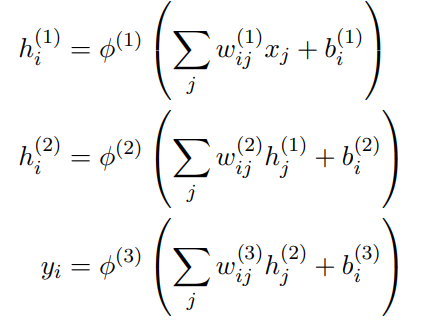


Fig. 21 – Multi Layer Perceptron Model



(2)

## Neural Net Optimization Algorithms –

* **Stochastic Gradient Descent**
* **Stochastic Gradient descent with momentum**
* **Mini-Batch Gradient Descent**
* **Adagrad**
* **RMSProp**
* **AdaDelta**
* **Adam**

Before proceeding there are few terms that we should be familiar with,

**Epoch** – The number of times the algorithm runs on the whole training dataset.

**Sample** – A single row of a dataset.

**Batch** – It denotes the number of samples to be taken to for updating the model parameters.

**Learning rate** – It is a parameter that provides the model a scale of how much model weights should be updated.

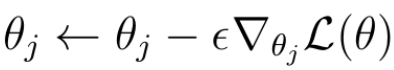
**Cost Function/Loss Function** – A cost function is used to calculate the cost that is the difference between the predicted value and the actual value.

**Weights/ Bias** – The learnable parameters in a model that controls the signal between two neurons.

**Gradient Descent Deep Learning Optimizer: -**

Gradient Descent, a popular optimization algorithm, uses calculus to modify the values consistently and to achieve the local minimum.

In simple terms, let’s consider one is holding a ball resting at the top of a bowl. When one loses the ball, it goes along the steepest direction and eventually settles at the bottom of the bowl. A Gradient provides the ball in the steepest direction to reach the local minimum that is the bottom of the bowl.



(3)

In this technique, we must calculate the gradient of the loss function **L** with respect to the weights (or parameters **θ**) that we want to improve. Subsequently, the weights/parameters are updated in the direction of the negative direction of the gradient.

**Stochastic Gradient Descent with Momentum: -**

Below is the description of stochastic gradient descent with momentum.



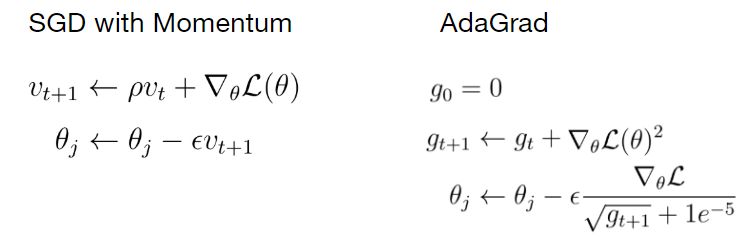
(4)

On the left side in Eq. 2, the equation for the weight updates according to the regular stochastic gradient descent has been provided. The equation on the right shows the rule for the weight updates according to the SGD with momentum. The momentum appears as an additional term **ρ**times **v**that is added to the regular update rule.

By, adding this momentum term we let our gradient to build up a kind of velocity **v**during training. The velocity is the running sum of gradients weighted by **ρ**. **ρ**can be considered as friction that slows down the velocity a little bit. In general, it can be seen that the velocity builds up over time. By using the momentum term saddle points and local minima become less dangerous for the gradient. Because step sizes towards the global minimum now don’t depend only on the gradient of the loss function at the current point, but also on the velocity that has built up over time.

**AdaGrad: -**

Another optimization strategy is called AdaGrad. The idea is that you keep the running sum of squared gradients during optimization. In this case, we have no momentum term, but an expression **g** that is the sum of the squared gradients.



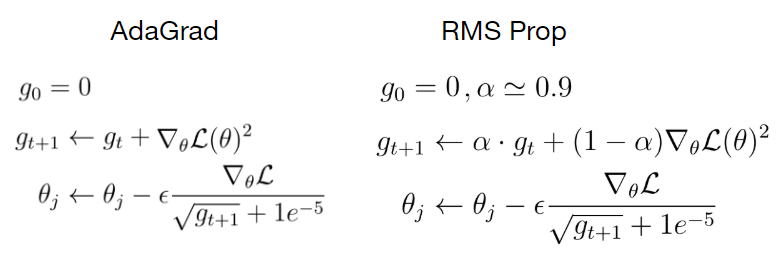
**(5)**

When we update a weight parameter, we divide the current gradient by the root of that term **g**. To explain the intuition behind AdaGrad, let’s imagine a loss function in a two-dimensional space in which the gradient of the loss function in one direction is very small and very high in the other direction.

Summing up the gradients along the axis where the gradients are small causes the squared sum of these gradients to become even smaller. If during the update step, we divide the current gradient by a very small sum of squared gradients **g**, the result of that division becomes very high and vice versa for the other axis with high gradient values.

**RMSProp: -**

There is a slight variation of AdaGrad called RMSProp that addresses the problem that AdaGrad has. With RMSProp we keep the running sum of squared gradients but instead of letting that sum grow continuously over the period of training we let that sum actually decay.



**(6)**

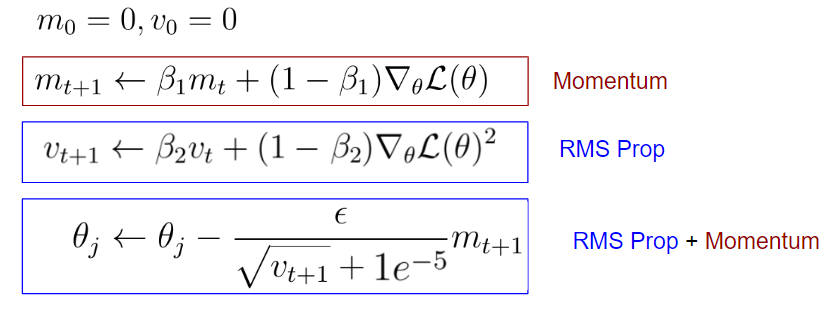
In RMSProp we multiply the sum of squared gradients by a decay rate **α**and add the current gradient weighted by (1- **α)**. The update step in the case of RMSProp looks exactly the same as in AdaGrad where we divide the current gradient by the sum of squared gradients to have this nice property of accelerating the movement along the one dimension and slowing down the movement along the other dimension.

**Adam Optimizer**

So far, we have used the moment term to build up the velocity of the gradient to update the weight parameter towards the direction of that velocity. In the case of AdaGrad and RMSProp, we used the sum of the squared gradients to scale the current gradient, so we could do weight updates with the same ratio in each dimension.

We can just take the best of both worlds and combine these ideas into a single algorithm? This is the exact concept behind the final optimization algorithm called Adam.

The main part of the algorithm consists of the following three equations. These equations have some familiarity with previous optimization algorithms.



**(7)**

The first equation looks a bit like the SGD with momentum. In the case, the term would be the velocity and the friction term. In the case of Adam, we call the first momentum and is just a hyperparameter.

The difference to SGD with momentum, however, is the factor (1- **β1**), which is multiplied with the current gradient.

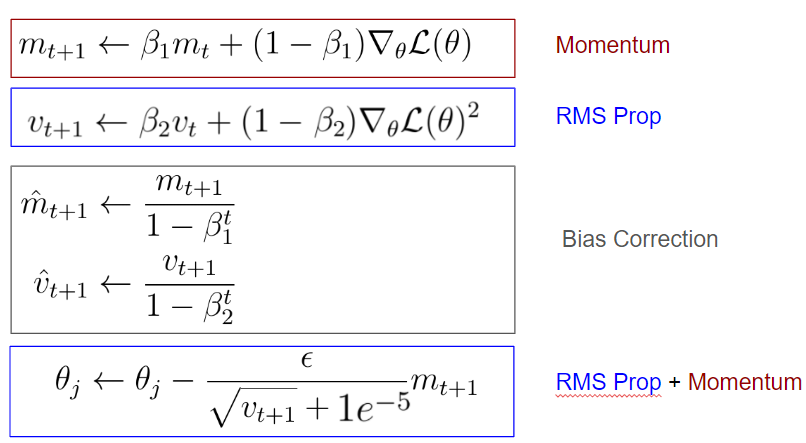
The second part of the equations, on the other hand, can be regarded as RMSProp, in which we are keeping the running sum of squared gradients. Also, in this case, there is the factor (1-**β2**) which is multiplied with the squared gradient.

The term in the equation is called the second momentum and is also just a hyperparameter. The final update equation can be seen as a combination of RMSProp and SGD with Momentum.

So far, Adam has integrated the nice features of the two previous optimization algorithms, but here’s a little problem, and that’s the question of what happens in the beginning.

At the very first time step, the first and second momentum terms are initialized to zero. After the first update of the second momentum, this term is still very close to zero. When we update the weight parameters in the last equation, we divide by a very small second momentum term **v**, resulting in a very large first update step.

This first very large update step is not the result of the geometry of the problem, but it is an artifact of the fact that we have initialized the first and second momentum to zero. To solve the problems of large first update steps, Adam includes a correction clause:



**(8)**

It can be seen that after the first update of the first and second momentum and we make an unbiased estimate of these momentums by taking into account the current time step. These correction terms make the values of the first and second momentum to be higher in the beginning than in the case without the bias correction.

As a result, the first update step of the neural network parameters does not get that large and we don’t mess up our training in the beginning. The additional bias corrections give us the full form of Adam Optimizer.

## Implementation of Latin Hypercube Sampling –

Latin hypercube sampling (LHS) is a statistical method for generating a near-random sample of parameter values from a multidimensional distribution. The sampling method is often used to construct computer experiments or for Monte Carlo integration.

In the context of statistical sampling, a square grid containing sample positions is a Latin square if (and only if) there is only one sample in each row and each column. A Latin hypercube is the generalization of this concept to an arbitrary number of dimensions, whereby each sample is the only one in each axis-aligned hyperplane containing it.

When sampling a function of {\displaystyle N}variables, the range of each variable is divided into {\displaystyle M} equally probable intervals. {\displaystyle M}Sample points are then placed to satisfy the Latin hypercube requirements. This forces the number of divisions, {\displaystyle M} to be equal for each variable. This sampling scheme does not require more samples for more dimensions (variables); this independence is one of the main advantages of this sampling scheme. Another advantage is that random samples can be taken one at a time, remembering which samples were taken so far.

[Calendar

Description automatically generated with medium confidence](https://en.wikipedia.org/wiki/File:LHSsampling.png)

Fig. 22 – Probability distribution of points generated via LHS

In two dimensions the difference between random sampling, Latin hypercube sampling, and orthogonal sampling can be explained as follows:

In **random sampling** new sample points are generated without taking into account the previously generated sample points. One does not necessarily need to know beforehand how many sample points are needed.

In **Latin hypercube sampling** one must first decide how many sample points to use and for each sample point remember in which row and column the sample point was taken. Such configuration is similar to having N rooks on a chess board without threatening each other.

In **orthogonal sampling**, the sample space is divided into equally probable subspaces. All sample points are then chosen simultaneously making sure that the total set of sample points is a Latin hypercube sample, and that each subspace is sampled with the same density.

Thus, orthogonal sampling ensures that the set of random numbers is a very good representative of the real variability, LHS ensures that the set of random numbers is representative of the real variability whereas traditional random sampling (sometimes called brute force) is just a set of random numbers without any guarantees.

We have taken 200 sample combinations generated via LHS to cater maximum variations while training the neural network.

## Configuration of deployed neural net –

* **Deep learning package used – TensorFlow, Keras.**
* **Number of sample combinations generated through LHS for training- 200**
* **Specs of learning model used – 1 input node, 1 output node and 2 hidden layer with 64 node in each layer.**
* **Total epochs – 1200**

# **PARTICLE SWARM ALGORITHM AND OVERALL WORKFLOW**

PSO is a stochastic optimization technique based on the movement and intelligence of swarms. In PSO, the concept of social interaction is used for solving a problem. It uses several particles (agents) that constitute a swarm moving around in the search space, looking for the best solution. Each particle in the swarm looks for its positional coordinates in the solution space, which are associated with the best solution that has been achieved so far by that particle. It is known as pbest or personal best. Another best value known as gbest or global best which is tracked by the PSO. This is the best possible value obtained so far by any particle in the neighborhood of that particle.

**Group optimization and Ensemble Learning**

It has been seen that all optimization algorithms perform equally well when averaged across all potential problems. To train a model, we must define a loss function to measure the difference between our model predictions. Our objective is to minimize or optimize this loss function so that it will be closer to 0. Ensemble Learning speaks about learning in a group or crowd. It is like training a model with the help of multiple algorithms. A single base learner is a weak learner. But, when all these vulnerable learners are combined, they become strong learners. They become strong learners because their predictive power, accuracy and precision become high. And the error rate becomes less. We call this type of combined model ‘Meta-learning’. It refers to learning algorithms that can learn from other learning algorithms. It decreases variance, decreases bias, and improves prediction.

The concept of swarm intelligence inspired the POS model. Here the goal is to find an optimal solution in a high-dimensional solution space by maximizing or minimizing a function. A function can only have one global maximum and one global minimum. If the function is very complex, then finding the global maximum becomes a very daunting task. PSO tries to capture the global maximum or minimum. Even though it cannot capture the exact global maximum/minimum, it goes very close to it. It is the reason we called PSO a heuristic model.

**An Intuition of Particle Swarm Optimization**

The movement towards a promising area to get the global optimum.

* Each particle adjusts its traveling velocity dynamically, according to the flying experiences it has and its colleagues in the group.
* Each particle tries to keep track of:

1. It’s best result for him/her, known as personal best or pbest.
2. The best value of any particle is the global best or gbest.

* Each particle modifies its position according to:

1. its current position
2. its current velocity
3. the distance between its current position and pbest.
4. The distance between its current position and gbest.

## Particle Swarm Optimization Algorithm -

Let us assume a few parameters first.

f: Objective function, Vi: Velocity of the particle or agent, A: Population of agents, W: Inertia weight, C1: cognitive constant, U1, U2: random numbers, C2: social constant, Xi: Position of the particle or agent, Pb: Personal Best, gb: Global Best

The algorithm is shown below:

1. Create a ‘population’ of agents (particles) which is uniformly distributed over X.

2. Evaluate each particle’s position considering the objective function.

3. If a particle’s present position is better than its previous best position, update it.

4. Find the best particle (according to the particle’s last best places).

5. Update particles’ velocities.

Particle Swarm Optimization - velocities

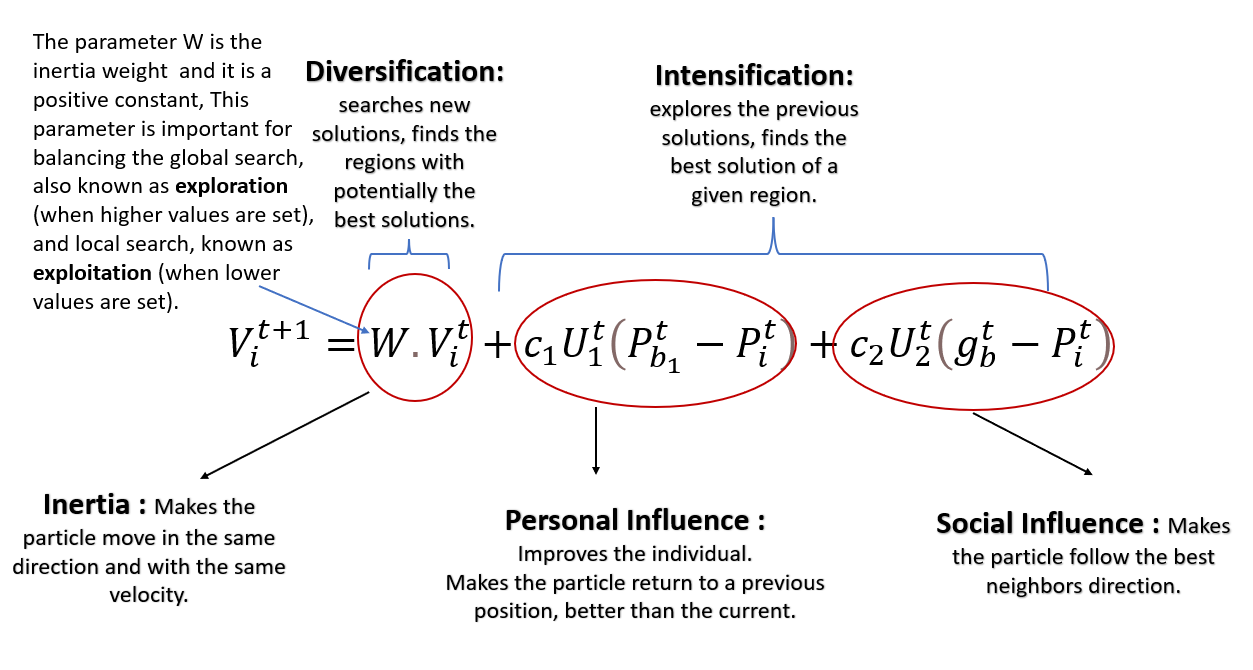
(9)

6. Move particles to their new positions.

Particle Swarm Optimization moves

(10)

7. Go to step 2 until the stopping criteria are satisfied.



(11)

# **RESULTS AND DISCUSSIONS**

The following table shows some of the random parameter combinations taken for each single cell simulation run and the corresponding result with it. 150 combinations were taken in total and the results were fed in the neural network.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No** | **Angle (deg)** | **Width(mm)** | **Thickness(mm)** | **Mass(kg)** | **Energy(J)** | **Rf(N)** |
| 1 | -24.0472 | 4.372838 | 4.255684 | 0.00403 | 7.89 | 11794.3 |
| 2 | -26.1284 | 4.580203 | 2.697816 | 0.00268 | 1.621 | 2877.45 |
| 3 | -13.908 | 4.850663 | 2.526085 | 0.00257 | 1.768 | 5372.74 |
| 4 | -29.1368 | 3.72006 | 2.302312 | 0.00191 | 0.835 | 2415.57 |
| 5 | -27.3489 | 3.039169 | 2.406886 | 0.00163 | 0.821 | 1995.44 |
| 6 | 29.37046 | 3.585936 | 4.743985 | 0.00379 | 8.772 | 12420.8 |
| 7 | 27.07856 | 4.748824 | 2.882886 | 0.00298 | 2.608 | 5432.66 |
| 8 | 18.17504 | 4.280136 | 4.02353 | 0.00368 | 7.958 | 12365.1 |
| 9 | 1.772228 | 4.340276 | 2.477442 | 0.00223 | 1.927 | 8033.82 |
| 10 | 7.142554 | 3.16829 | 4.123198 | 0.00277 | 9.527 | 11408.5 |
| 11 | 11.07861 | 4.080396 | 3.622133 | 0.00312 | 5.827 | 9386.94 |
| 12 | -12.062 | 4.449875 | 3.720693 | 0.00348 | 6.88 | 11074.2 |
| 13 | 6.087464 | 2.970135 | 2.189879 | 0.00138 | 0.906 | 3598.01 |
| 14 | 15.58294 | 2.153994 | 4.9996 | 0.00235 | 9.575 | 10685.3 |
| 15 | -19.1257 | 3.222894 | 4.510578 | 0.00316 | 9.003 | 11272.5 |
| 16 | -7.59724 | 2.755561 | 3.330106 | 0.00196 | 3.151 | 5549.06 |
| 17 | 3.184507 | 2.530012 | 3.969415 | 0.00215 | 7.504 | 8729.69 |
| 18 | -3.1729 | 2.460638 | 2.964826 | 0.00156 | 2.034 | 5111.48 |
| 19 | -10.9613 | 4.801529 | 2.589658 | 0.00259 | 1.97 | 6004.86 |

Plots provided below show the loss and accuracy of the predicted results obtained from the trained Neural Network. After 1200 epochs we can see that the loss has gone below 2% and the has accuracy has attained a value of above 80% in all the cases.

Graphical user interface

Description automatically generated with medium confidence

Fig. 23 – Loss and accuracy plot for mass prediction

Graphical user interface

Description automatically generated

Fig. 24 – Loss and accuracy plot for SEA prediction

Graphical user interface

Description automatically generated with medium confidence

Fig. 25 – Loss and accuracy plot for RF prediction

From the neural network a set of optimized parameters have been obtained for our NPR cell structure. Now, different type of assembly simulations will be constructed to compare the energy absorption and reaction force. At first, three simulation models have been made with 6\*6 assembly under pressure loading. One with the initial parameters for our seventh semester work, one with the optimized parameters for our NPR cell structure and one with the optimized parameters but for a normal cell.

From the following graphs we can see the results of this comparison study. The energy absorption is better for our optimized NPR cell structure than the initial structure and normal cell structure. Although the peak reaction force is greater than the original structure for our optimized structure, the sustained values are mostly same. And the reaction force for the NPR cell structure is much less than the normal cell structure. Comparing these points, it can be said that optimization of NPR cell structure is successful.

Chart, histogram

Description automatically generated

Fig. 26 – Comparison of crushing length vs reaction force

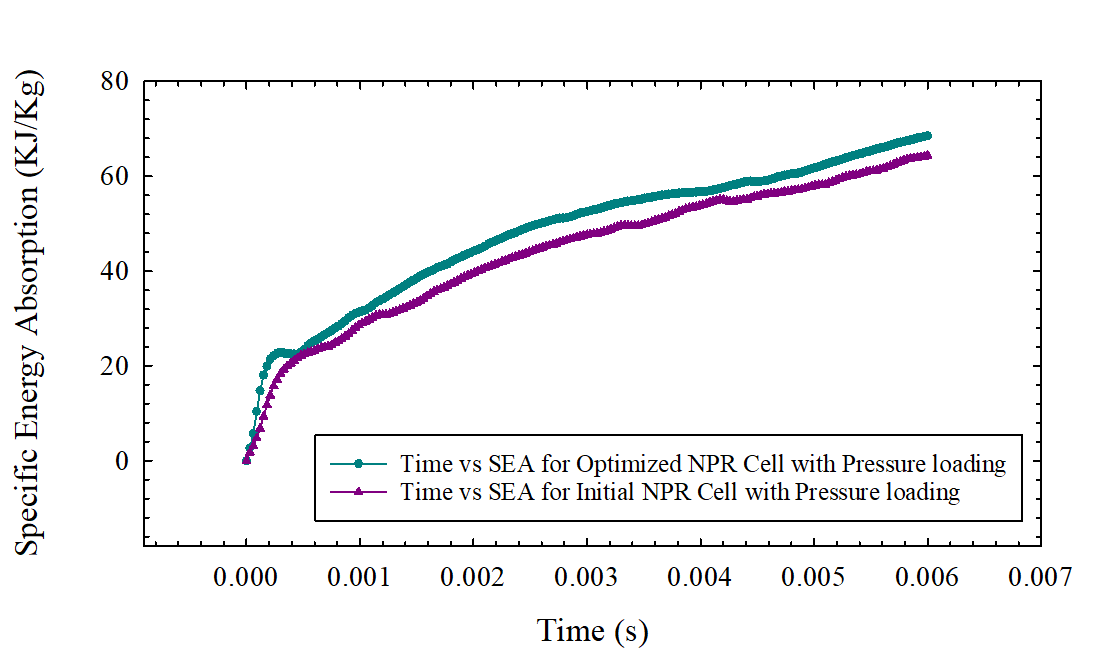


Fig. 27 – Comparison of time vs SEA

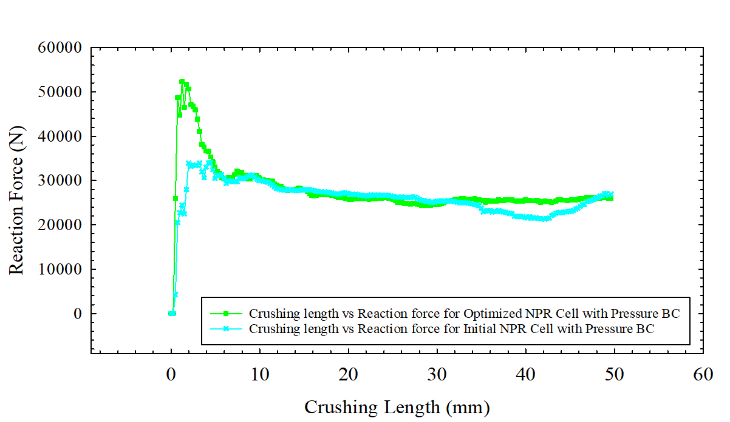


Fig. 28 – Comparison of crushing length vs Reaction force

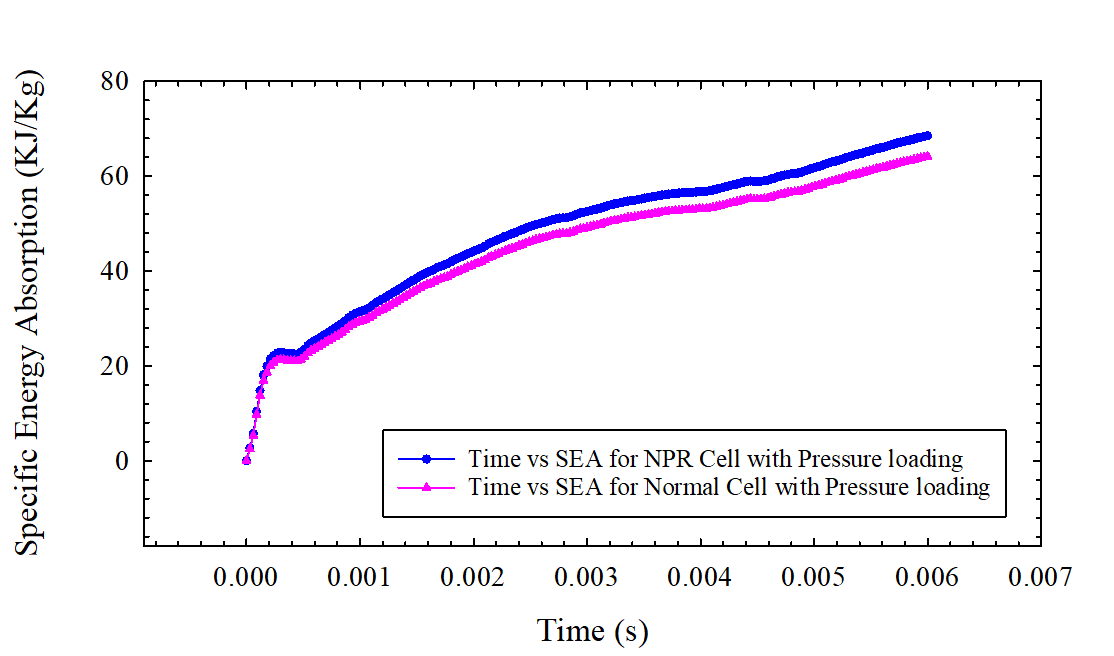


Fig. 29 – Comparison of time vs SEA

Then, further assembly simulations have been constructed with our optimized parameters varying the unit cell numbers while keeping the total volume constant. We have taken three cases, 2\*2 assembly, 4\*4 assembly and the original 6\*6 assembly. The following graph shows that the Specific Energy Absorption (KJ/Kg) and Reaction Force is highest for the 6\*6 assembly and lowest for the 2\*2 assembly. So, it can be concluded that a greater number of cells is better for Specific Energy Absorption.

Chart, line chart

Description automatically generated

Fig. 30 - Comparison of time vs SEA for 2,4,6 cell assembly

Chart

Description automatically generated

Fig. 31 - Comparison of time vs SEA for 2,4,6 cell assembly

**Table 1: Summery of the Improvements gained via optimization procedure**

|  |  |  |  |
| --- | --- | --- | --- |
| **Design Combinations** | **Loading Conditions** | **Specific Energy Absorption** | **Improvements of optimized 6 cell assembly when compared to other combination** |
| **Initial NPR 6 cell design(30, 3, 2.28)** | **Pressure Loading**  **(27.5 kg, 11.6m/s)** | **61.135 (KJ/Kg)** | **12.69%** |
| **Optimized NPR 6 cell design**  **(10, 2 , 3.14)** | **Pressure Loading**  **(27.5 kg, 11.6m/s)** | **68.455 (KJ/Kg)** | **-** |
| **2 cell Optimized NPR assembly** | **Pressure Loading**  **(27.5 kg, 11.6m/s)** | **59.45 (KJ/Kg)** | 23.23% |
| **4 cell Optimized NPR assembly** | **Pressure Loading**  **(27.5 kg, 11.6m/s)** | **52.92 (KJ/Kg)** | **13%** |

# **CONCLUSION**

* **An unique structural constrained design optimization workflow is proposed incorporating Artificial Neural Network Prediction and Particle Swarm Algorithm.**
* **A maximum improvement of SEA 12.69% was obtained when compared to the initial design.**
* **A comparison of NPR and Normal cell Reaction force reveals that the NPR cell has 34.68% lower sustained pick RF when compared to its normal counter-part.**
* **The number of optimized design cells size were varied while keeping the volume constant, the general trend revealed by the analysis is – the more number of cells the more amount of energy it will absorb but the pick reaction force will also increase simultaneously.**

# **REFERENCES**

[1] M. Guida, F. Marulo, and S. Abrate, “Advances in crash dynamics for aircraft safety,” *Prog. Aerosp. Sci.*, vol. 98, no. December 2017, pp. 106–123, 2018.

[2] “Energy Absorption in composite stiffner.pdf.” .

[3] J. J. Carruthers, A. P. Kettle, and A. M. Robinson, “Energy absorption capability and crashworthiness of composite material structures: A review,” *Appl. Mech. Rev.*, vol. 51, no. 10, pp. 635–649, 1998.

[4] P. Feraboli, “Development of a corrugated test specimen for composite materials energy absorption,” *J. Compos. Mater.*, vol. 42, no. 3, pp. 229–256, 2008.

[5] T. Wang, J. An, H. He, X. Wen, and X. Xi, “A novel 3D impact energy absorption structure with negative Poisson’s ratio and its application in aircraft crashworthiness,” *Compos. Struct.*, vol. 262, no. January, p. 113663, 2021.

[6] L. Cabras and M. Brun, “Auxetic two-dimensional lattices with Poisson’s ratio arbitrarily close to −1,” *Proc. R. Soc. A Math. Phys. Eng. Sci.*, vol. 470, no. 2172, p. 20140538, Dec. 2014.

[7] I. S. Abbood, S. A. Odaa, K. F. Hasan, and M. A. Jasim, “Properties evaluation of fiber reinforced polymers and their constituent materials used in structures – A review,” *Mater. Today Proc.*, vol. 43, pp. 1003–1008, Jan. 2021.

[8] J. Paz, J. Díaz, L. Romera, and M. Costas, “Size and shape optimization of aluminum tubes with GFRP honeycomb reinforcements for crashworthy aircraft structures,” *Compos. Struct.*, vol. 133, pp. 499–507, 2015.