Development of artificial neural network (ANN) model for modelling turbulent flames

B.Tech. Project Report

by

Anmoldeep Singh

180030002

Faculty Adviser

Dr. Rudra Narayan Roy

School of Mechanical Sciences

Indian Institute of Technology Goa

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources.

I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_____

(Signature)

Anmoldeep Singh

(Name of the student)

180030002

(Roll No.)

Date: 19/12/2021

Abstract

Turbulent combustion is one of the key processes in many energy conversion systems in modern life. Considering the economical importance of hydrocarbon combustion in energy production processes, it is evident that there is a need for an accurate tool with a relatively low computational cost for the prediction of these kinds of reacting flows.

The present work is devoted to the development of an artificial neural network that can be used in place of the tabulated chemistry approach, i.e., flamelet progress variable model. The work is based on the coupling of an in-house chemistry solver to generate flamelets using the principle of flamelet progress variable model. OpenFOAM 6.0 was used to solve RANS, and an in-house flamelet solver was used to generate flamelet libraries. The coupled OpenFOAM-FPV model in the present study has the potential to investigate the various phenomena arising in the turbulent flames. The dependencies between OpenFOAM 6.0 and the in-house solver were later removed in order to get the values of the reactive scalars in terms of Z, Zeta and Yc, where earlier the scalars were calculated as a function of Z and Zeta only. The input for mixture fraction grid, mixture fraction variance grid and progress variable grid were directly taken from the user and the values of reactive scalars were calculated using the flamelet progress variable model.

The drawback of the OpenFOAM-FPV model is the high memory cost. The ANN model is proposed as it reduces memory usage by 99.9981%. The regression comparisons of the predicted ANN model and FPV show linear correlation in both sets of data. The developed ANN model shows huge enhancements as compared to the FPV model and the previously developed ANN model.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Background

Since the time humans evolved enough to understand what fire is, they have been fascinated by it. It has been the main driving force behind the revolutionalisation of the human life. From the primitive uses of cooking food and using fire for light to the most advanced uses of burning coal and fuel to generate heat for using in various applications, fire, and more specifically, combustions has been always around.

In today's world, many areas like industrial furnaces, heaters, power generation, transportation sector etc. rely heavily on combustion. The energy demand is majorly satisfied with the combustion of fossil fuels, even though with the advent of electrical power, combustion systems are not disappearing anytime soon. In the recent times, the pollution caused due to inefficient burning of fuels or the wastage of fuels has become the central concern among various researchers all over the world. To tackle this problem of making the combustion systems, more efficient and more cleaner, various governing bodies throughout the world have put up strict and stringent regulations pertaining to the emissions from any combustion system. A number of novel combustion technologies have been developed or are being developed to control the emissions. But with every new advent, the practical limitations posed on a combustion system are coming nearer, therefore, it is of great interest to ensure safe and profitable operation. All practical scenarios involve combustion taking place in an turbulent environment, and thus, it becomes challenging to model such systems so that better management can be performed.

In order to understand the physical phenomenons taking place during turbulent combustion, it is very important to carry out numerical investigation so that the number of experiments required can be reduced, thus, saving time and resources. But combustion is a complicated phenomenon in itself, involving a number of elementary reactions that involve radical species that are very reactive and unstable. And the combustion in practical scenratios like jet engines, domestic combustors etc. involve turbulent environment as well, which in itself is another challenge to model. In practical scenarios, turbulence and combustion are strongly coupled due to the interaction between the flow and chemical reactions.

Turbulence is identified with a large spectrum of temporal and spatial scales that involve energy

transfer between different levels. In addition to this, combustion is also identified with various time scales where various species are produced and destroyed. It is a big challenge to model turbulent combustion as one has to deal with various temporal and spatial time scales involved in turbulent flow which are coupled with the complex chemical reactions taking place across various chemical time scales.

In the recent decades, various models have come into light that can be used to model turbulent combustion processes, like, RANS and LES based combustion models that involve presumed PDF approaches like flamelet [1], conditional moment closure [2] and transported PDF [3]. These models have been employed to study properties of various turbulent flames like turbulent piloted, bluff-bofy, lifted flames [4][5][6][7][8][9] etc. Among the various models available, the tabulated chemistry based approach, i.e., flamelet model is of most interest as it helps in significant reduction in computational resources, even when using detailed chemical mechanisms. It is of vital importance to employ detailed chemical mechanisms to model turbulent flames as it enables us in observing various combustion phenomenons like flame extinction and blowoff that are central in prediction of emissions and efficiency. These phenomenons are transient, taking place at various scales, and thus, capturing them with a simplified chemical mechanism is not possible as there are various species that are not accounted for in a simple chemical mechanism. The idea of flamelet was put forward by Peters [1]. It was observed that it was not possible to predict extinction and lift-off phenomenon using the steady flamelet model in turbulent combustion, and thus, it led Pierce and Moin [10] to develop a flamelet progress variable (FPV) approach which was able to predict the above phenomenons. The central idea was the introduction of a scalar variable known as progress variable that was used in conjunction with the mixture fraction.

Later on, Ihme et al. [11] studied the ability of FPV model in predicting phenomenons like local extinction and reignition in nonpremixed combustion using the available DNS database. It was later discovered that the FPV model can be used to model unsteady phenomenons accurately if the PDF for progress variable is modeled using beta PDF. Ihme and Pitsch [12] predicted the NO formation, a leading contributor to pollution, using the FPV model and found reasonable agreement with the experimental data. To capture the phenomenon of autoignition in turbulent lifted flames, Ihme and See [13] developed an unsteady FPV model. Further, Renzo et al. [15] performed large eddy simulation (LES) for piloted Sandia flame D using the FPV model. The result were compared with the statistical model known as statistically most likely

distribution (SMLD) which was solved with Reynolds averaged Navier-Stokes (RANS) equations. Both RANS and LES turbulence models predicted the mixing field results accurately but the reactive fields were better predicted by LES.

With the great advancement in computational science field, the use of numerical methods that require huge computational resources have become feasible. One such example is OpenFOAM [16] which is a open source platform that is widely used to investigate fluid behaviours. OpenFOAM is an object oriented, C++ based, computational fluid dynamics tool that provides the capability of adding advanced models and numerical methods that can be employed for turbulent combustion research. The coupling of FPV model with RANS and LES turbulence models is highly beneficial for investigating turbulent flames.

It is well known that fluid simulations are accompanied by massive amounts of data that is generated from experiments, field measurements and large-scale numerical simulations. The massive amounts of data, or big data, as it is called in literature, have been associated with fluid simulations and research [21]. Over the years, many novel techniques have been developed to handle big data like advanced algorithms for data processing and compression, fluid mechanics databases ([22][23]).

The growth of data in today's era is prominent across varios scientific areas and extracting great insights and valuable information from data has taken the center stage in scientific research. We see a vast and ever increasing volumes of data, development of cheaper and efficient computational hardware, better and affordable means of data storage and transfer, complex and efficient algorightms, plethora of open source softwares and standards. All these factors have fueled the application of machine learning in various big data problems, like, computational fluid dynamics. It is becoming a new norm to use data driven problem solving paradigms, instead of only relying on first principles.

Machine learning algorightms like supervised, unsupervised, and semisupervised algorightms are heavily being used in computational fluid dynamics problems. The major advantage that data-driven methods, like ML, provide us is that it is flexible and robust modeling frameworks that can be tailored to cater to specific problems, like, reduced order modeling, turbulence closure modeling etc. Implementation of such data driven methods in fluid simulations holds the key in making complex simulations resource effective.

1.2 Objective

In this project, our major objective is to develop an machine learning model that can be used in place of the tabulated chemistry approach in modeling of turbulent bluff-body flame. The turbulent flame is modeled in OpenFOAM based flow solver and flamelet based combustion model, i.e., flamelet progress variable model. RANS based turbulence model available in OpenFOAM is used to model turbulence flow field. Coupling of FPV solver with OpenFOAM is performed to model the reacting field. An open-source python library. Pytorch is used to train the artificial neural network (ANN) on the data collected from the OpenFOAM-FPV model. Our main aim here is to minimize the storage resources required for modeling turbulent bluff-body flame.

Chapter 2: Numerical Modeling

2.1 RANS Turbulence model

RANS equations are the time-averaged equation of motion that describe the fluid flow. It is constructed from the Reynolds decomposition of the Navier Stokes equation [19]. For stationary turbulence, i.e., for a turbulent flow that does not vary with time, on an average, we can use time averaging. For such a flow, we express an instantaneous flow variable as *f(x, t)*. The time averaged form of *f(x, t)* is $F_T(x)$, which is defined by [18],

$$F_T(x) = \overline{f(x,t)} = \lim_{n\to\infty} \frac{1}{\tau} \int_{1+\tau}^{1+\tau} f(x,t)dt$$

Reynolds decomposition is a mathematical modeling technique that is employed in order to segregate the mean value of a time varying quantity from its fluctuations [19],

$$\varphi(x,t) = \phi(x) + \varphi'(x,t)$$

For the sake of understanding, we will limit our analysis to incompressible, constant property flow. The Reynolds averaged Navier stokes equation for such a flow is [18]:

$$\rho\left(\frac{\partial \bar{u}}{\partial t} + \nabla(\bar{u}.\bar{u})\right) = -\nabla\bar{P} + \mu\nabla^2(\bar{u}) - \nabla(\rho\overline{u'.v'})$$

Here $\overline{u'v'}$ is the Reynolds stress tensor which is in unclosed form, and this term is the main problem faced while solving the Navier-Stokes equation. For the closure of the Reynolds shear stress tensor many mathematical techniques are introduced, first closure problem was performed by Boussinesq with the introduction of the eddy viscosity concept [4]. It was experimentally observed that turbulence decays over time and was found to increase as the mean rate of deformation increases. Boussinesq proposed that Reynolds stresses could be linked to the mean rate of deformation [18],

$$\tau_{ij} = -\rho\overline{u'.v'} = \mu_t\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \frac{2}{3}\rho k\delta_{ij}$$

Here μ_t is turbulent viscosity, *k* is turbulent kinetic energy, and δ_{ij} is Kronecker delta.

2.2 Mixture Fraction Theory

In non-premixed combustion, if the chemical time scale is very short compared to the mixing time scale, the turbulent flame can be approximated using the fast chemistry assumption, which states that the chemistry is not affected by the flow. Thus, temperature and species concentrations can be treated as functions of a conserved scalar quantity, generally called the

mixture fraction. Thus, the chemistry and turbulence are decoupled by this conserved scalar quantity. Mixture fraction can be defined, at any location in the system, as the local ratio of mass flux originating from the fuel side to the sum of both mass fluxes [1] and is given as

$$Z = \frac{\dot{m}_F}{\dot{m}_F + \dot{m}_{Ox}}$$

where, m_F and m_{Ox} denote the mass flow rates of fuel and oxidizer stream respectively. Assuming that the species diffusivities are all equal to *D*, the mixture fraction transport equation is written as [1]

$$\frac{\partial(\rho Z)}{\partial t} + \frac{\partial(\rho u_i Z)}{\partial x_i} = \frac{\partial}{\partial x_i}\left(\rho D \frac{\partial Z}{\partial x_i}\right)$$

The LHS terms represent the local rate of change and convection of *Z*. The term of the RHS accounts for *Z* transport by means of molecular diffusion. The Favre averaged governing equation is written as [1]

$$\frac{\partial(\bar{\rho}\tilde{Z})}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u_\iota}\tilde{Z})}{\partial x_i} = \frac{\partial}{\partial x_i}\left(\overline{\rho D \frac{\partial Z}{\partial x_\iota}} - \bar{\rho}\widetilde{u_i'' Z''}\right)$$

The second term in RHS is modelled using gradient-diffusion hypothesis [18] and is given as.

$$\widetilde{u_i'' Z''} = -D_t \frac{\partial Z}{\partial x_i}$$

Here D_t is turbulent diffusivity. Since the magnitude of the molecular diffusion term is small compared to that of the turbulent diffusion term and therefore it is neglected as is written as

$$\frac{\partial(\bar{\rho}\tilde{Z})}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u_\iota}\tilde{Z})}{\partial x_i} = \frac{\partial}{\partial x_i}\left(-D_t \frac{\partial \tilde{Z}}{\partial x_i}\right)$$

In the context of modelling nonpremixed turbulent combustion we need to solve an equation for mixture fraction variance $\widetilde{Z''^2}$ in addition to mean mixture fraction equation [1]. It is written as

$$\bar{\rho}\left(\frac{\partial Z''^2}{\partial t} + \bar{\rho}\tilde{v}.\nabla Z''^2\right) = -\nabla.(\bar{\rho}\, v'' Z''^2) + 2\bar{\rho} D_t (\nabla \tilde{Z})^2 - \bar{\rho}\tilde{\chi}$$

2.3 Flamelet Model

Peters [1] derived the flamelet concept for non-premixed flames. A flamelet is defined as a thin reactive-diffusive layer embedded within a non-reactive turbulent flow field. According to this concept, we can deduce that the chemical time scale is very small compared to that of the convection and diffusion time scales. Thus, the effect of turbulence on the local structure of the flame is negligible and the flame is also treated to be locally laminar [1]. The flamelet equation

is given as follows:

$$-\rho\chi\left(\frac{\partial^2\Psi_\alpha}{\partial Z^2}\right)=\dot{\omega}_\alpha$$

where, Ψ_α is any reactive scalar variable, $\dot{\omega}_\alpha$ is the reaction source term and χ is scalar dissipation rate, which is given by [1],

$$\chi=2D\left(\frac{\partial Z}{\partial x_i}\right)^2$$

Scalar dissipation rate plays an essential role in non-premixed combustion. It depicts the inverse of the diffusion time scale and diffusion in the mixture fraction space. A two-dimensional library is constructed to store the solution of the steady state laminar flamelet equations. The entries of the library consist of temperature and species mass fractions that are functions of Z and χ . If the shape of the PDF of Z and χ is known beforehand, then we can calculate the mean value of reactive scalar quantities as:

$$\Psi_\alpha=\int_0^1\int_0^\infty\Psi_\alpha(Z,\chi)\tilde{P}(Z,\chi)\,d\chi dZ$$

2.4 Flamelet Progress Variable Model

The flamelet progress variable model introduces a new flamelet parameter, λ , which is based on a reactive scalar. The state of the flame is then parameterized in terms of the flamelet parameter λ, instead of the scalar dissipation rate, χ, which uniquely describes each single flame state along the S-shaped curve, including the unstable branch. Flamelets experiencing a transition from the burning to the extinguished flame state, or those which are likely to reignite, are then projected vertically onto the S-shaped curve (mirrored 'S') as shown here [10].

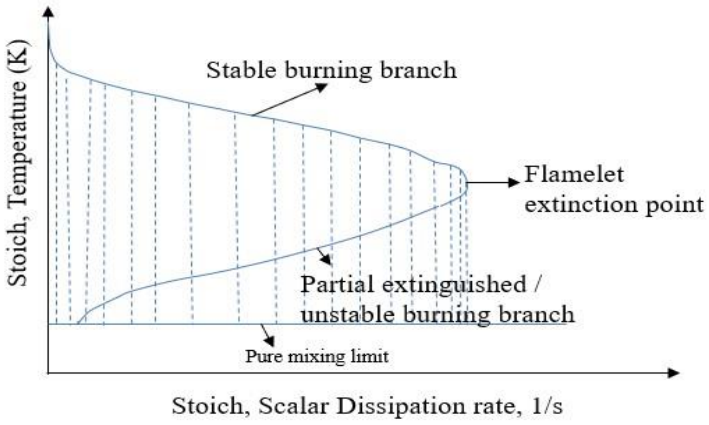


Figure 2.1: Unsteady flamelet solution

The flamelet parameter λ is defined in terms of a reactive scalar *C* (Progress Variable), that was introduced and defined by Pierce and Moin [10] as a linear combination of mass fraction of major reaction products. In this study CO_2, H_2O, and CO were used as the major reaction products. The value of *C* is estimated by solving the following transport equation [10].

$$\frac{\partial(\bar{\rho}\tilde{C})}{\partial t} + \nabla.(\tilde{u}\rho\tilde{C}) = \nabla.(\bar{\rho}D\nabla\tilde{C}) + \dot{\omega}_c$$

λ is assumed to be independent of mixture fraction and progress variable for a given flamelet. We then express any reactive scalar in terms of [10],

$$\Psi = \Psi(Z, C)$$

Mean values of the scalars can be determined by using the presumed PDF approach, which is given by the below equation [10],

$$\widetilde{\Psi}_i = \int_0^1 \int_0^1 \Psi_i(Z, C)\tilde{P}(Z, C)\, dZdC$$

Here $\tilde{P}(Z, \mathrm{C})$ is the Favre joint PDF of *Z* and *C*. Using Bayes' theorem, the joint PDF can be rewritten in terms of the conditional PDF [10],

$$\tilde{P}(Z, C) = \tilde{P}(C|Z)\tilde{P}(Z)$$

Here $\tilde{P}(Z)$ is classically assumed to be a beta PDF, and the conditional $\tilde{P}(C|Z)$ is taken from a single flamelet relation [10],

$$\tilde{P}(C|Z) = \delta(C - C(Z, \chi_{st}))$$

Here δ-PDF is the Dirac-delta function and $C(Z, \chi_{st})$ is the progress variable as a function of *Z* for a particular flamelet at a scalar dissipation rate of χ_{st}. The scalar dissipation rate χ_{st} is chosen such that is satisfies the following equation [9],

$$\tilde{C} = \int_0^1 C(Z, \chi_{st})\tilde{P}(Z)dZ$$

A two-dimensional library is constructed and the solution of the laminar flamelet equations is stored in that library. The entries of this library consist of temperature and species mass fractions which are a function of mixture fraction and progress variable. The values of *Z* and *C* is calculated from the transport equation, and by solving the below equation tables are formed

$$\widetilde{\Psi} = \int_0^1 \int_0^1 \Psi(Z, C)\beta(Z)\delta(C - \bar{C})\, dCdZ$$

2.5 Assumed PDF Approach

In this approach a suitable two-parameter probability density function is "presumed" in advance, so that the functional form of the PDF is fixed. It is achieved by relating the two parameters in terms of the known values of the mean mixture fraction and its variance at each point of the flow field. Since in a two-feed system the mixture fraction *Z* varies from *Z = 0* to *Z = 1*, the beta function PDF is widely used for the Favre PDF approach in non-premixed turbulent combustion [1]. The beta function PDF is defined as,

$$\tilde{P}(Z) = \frac{\tilde{Z}^{\alpha-1}\left(1-\tilde{Z}\right)^{\beta-1}}{\Gamma(\alpha)+\Gamma(\beta)}\Gamma(\alpha+\beta)$$

Where Γ is the generalized factorial function, or the gamma function. The two parameters α and β are related to the Favre mean mixture fraction (Z) and its variance (Z''^2) by following equations,

$$\gamma = \frac{Z(1-Z)}{Z''^2} - 1$$

$$\beta = (1-Z)\gamma$$

$$\Gamma(t) = \int_0^\infty x^{t-1}e^{-x}dx$$

2.6 Redefining the earlier method

The data being generated from the chemical mechanism and OpenFOAM, any reactive scalar, ϕ, is a function of $\tilde{\phi} = \phi(\tilde{Z}, \tilde{C})$ only but the interpolation was performed using $\tilde{Z}, \tilde{C}$ and $\widetilde{Z''}$. So to encounter this problem, the dependency of the data on the OpenFOAM was removed by providing the flow data manually.

The user gives the input for the values of $\tilde{Z}, \tilde{C}$ and $\widetilde{Z''}$, 105 values for $\tilde{Z}$, 105 values for $\tilde{C}$ and 153 values for $\widetilde{Z''}$. Thus, the data is generated again with respect to $\tilde{Z}, \tilde{C}$ and $\widetilde{Z''}$.

The plots of the new data for all the reactive scalars are plotted. The function is plotted with respect to two variables at a time, thus, each reactive scalar has three plots each.

2.7 Data plotting

In order to plot the variation of the reactive scalars with respect to $\tilde{Z}, \tilde{C}$ and $\widetilde{Z''}$. The different approaches used for data plotting were:

a. iPyvolume
    - Advantages

- Produces 4D plots interactive plots, with 4th dimension depicted by color.
- Dynamic parameters are available.

- Disadvantages
  - Only useful for running on Jupyter notebooks, can not run on standalone server/local machines.
  - Plots can only be saved as static entities.

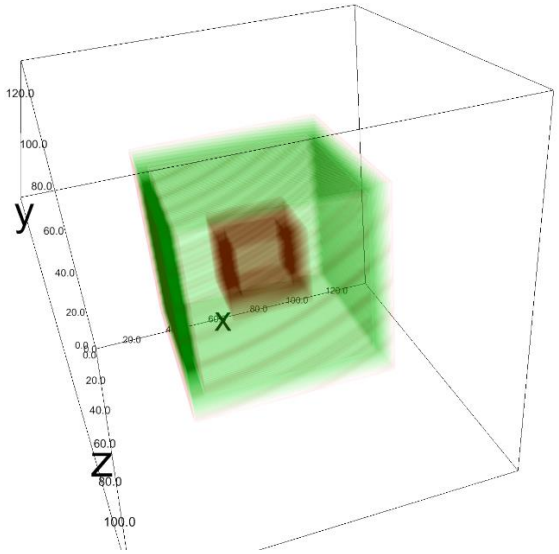


Figure 2.2: Plot in iPyvolume

b. Mayavi

- Advantage:
  - Produces 3D interactive plots.
- Disadvantage:
  - Requires the setup of a virtual environment like anaconda.
  - No dynamic parameters.
  - Compatible version of vtk not being available for Python 3.8 and above.
  - Plots can only be saved as static entities.
  - Function has to be an expression of coordinates and not an array.

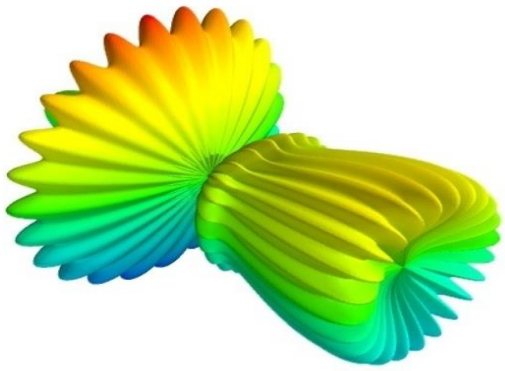


Figure 2.3: Plot in Mayavi

c. Plotly.go

- Advantage:
  - Produces 4D plots interactive plots, with 4th dimension depicted by color.
- Disadvantage:
  - No dynamic parameters.
  - Plots can only be saved as static entities.
  - Function has to be an expression of coordinates and not an array.

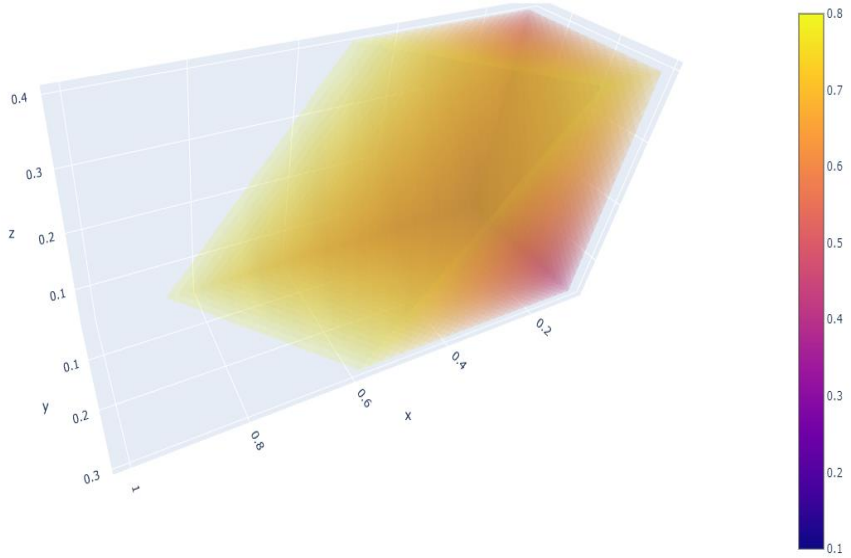


Figure 2.4: Plot in Plotly.go

d. Paraview

- Advantage:
  - Produces 3D volume plots.
- Disadvantage:
  - No dynamic parameters.
  - Plots can only be saved as static entities.
  - Unable to handle generated files of size around ~60MB

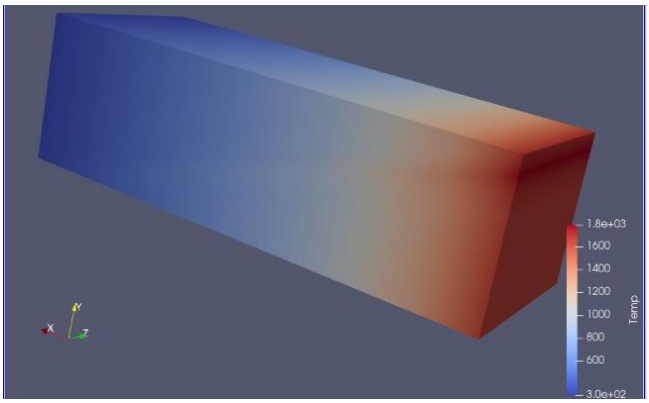


Figure 2.5: Plot in Paraview

e. Scatterplot

- Advantage:
  - Produces 3D scatter plots from three independent arrays.
- Disadvantage:
  - No dynamic parameters.
  - Plots can only be saved as static entities.
  - Lacks 3D surfaces that makes the plot hard to comprehend.

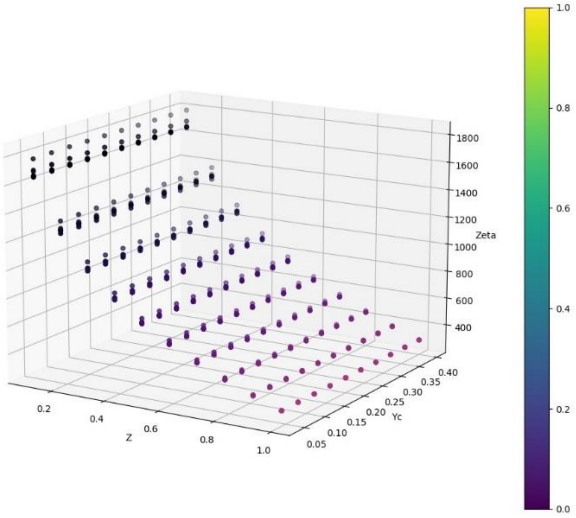


Figure 2.6: Scatterplot

f. Triangular contour plots

- Advantage:
  - Produces 3D contour plots that are color-coded.
  - Uses three independent arrays.
- Disadvantage:
  - Plots can only be saved as static entities.
  - It only plots two variables at a time.
  - Lacks 3D continuous surfaces that makes the plot hard to comprehend.

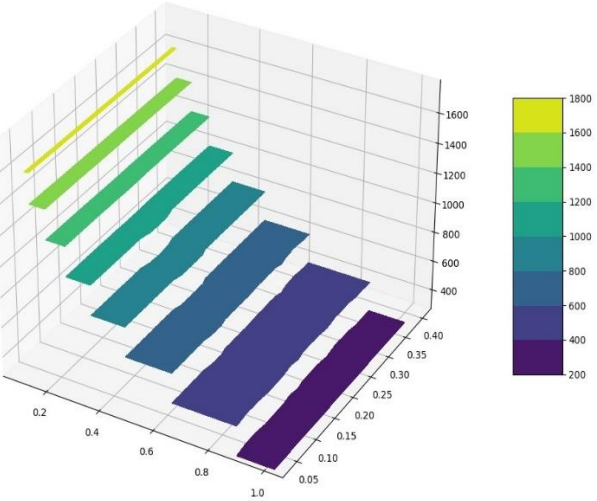


Figure 2.7: Triangular contour plot

g. Triangular surface plots

- Advantage:
  - Produces 3D surface plots from arrays.
  - Uses three independent arrays.
- Disadvantage:
  - It only plots two variables at a time.
  - Plots can only be saved as static entities.

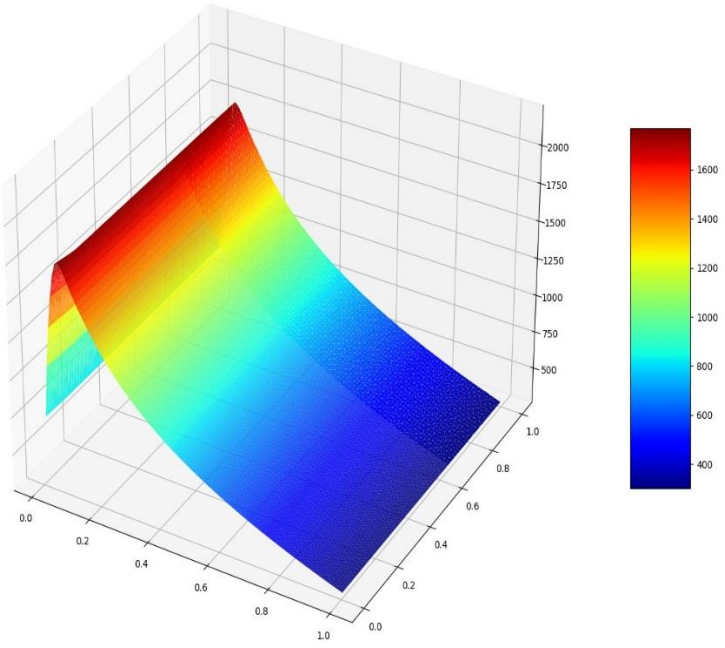


Figure 2.8: Triangular surface plot

The triangular surface plot was selected among all the available options, based on its easy application, availability and comprehensibility. All the species were plotted with respect to two variables at a time. Thus, there were 3 graphs corresponding to each species.

Chapter 3: Artificial Neural Network

3.1 Introduction

The OpenFOAM – FPV framework that was used, requires the lookup tables to interpolate the values of reactive scalars that are obtained from the chemistry solver. uses lookup table generation method to interpolate values of reactive scalar obtained from chemistry solver into the flow solver. A total of 153 flamelet tables are constructed containing 78 flamelets each. These flamelets are flamelet solutions obtained from chemistry solver whereas flamelet tables are constructed using OpenFOAM utility using assumed PDF approach. Each flamelet table is a 3D table which is the function of mixture fraction, mixture fraction variance, and progress variable and consists of data of 32 species along with temperature and reaction source term of progress variable. The mixture fraction grid consists of 105 points, the mixture fraction variance grid has 153 points and the progress variable grid has 105 points and the storage space required for storing a double precision float is 8 bytes = 8×10^{-6} MB. Therefore, the data generated on one core will be: $105 \times 153 \times 105 \times 34 \times 8 \times 10^{-6} = 458.82$ MB.

Now suppose we want to parallelize the training process on a cluster of 32 cores, thus, the amount of data generated will be: $458.82 \times 32 = 14682$ MB = 14.7 GB. If we want to use an even more detailed mechanism, or want to enhance the model by accounting for effects like enthalpy defect, high pressure or increasing the number of grid points to make the discretization more finer, it will generate even more data. Suppose we want to take into account the effect of enthalpy defect, adding another dimension of 100 grid points, the total storage space required becomes $14.7 \times 100 = 1470$ GB, which is quite costly, and many systems may not have that much free memory.

Thus, the huge amount of data generated and the dimensionality of data is the main issue with the OpenFOAM-FPV model. So, developing a data-driven method, like ANN is the most favored approach as compared to the FPV model that will help in the reduction of storage resources required. In this study, a well trained ANN model has been developed that is used to model turbulent Methane-Hydrogen bluff-body flame.

3.2 Machine Learning Models

A learning task can be simplified and formulated as the process of estimating the relations, which are not very apparent, between the inputs to a system and the outputs of the system, that are influenced by various paramters. This learning task is performed using a

limited number of observations, i.e., from a limited number of input and output pairs [24]. These learning tasks can be broadly classified into three groups, depending on the amount of external supervision required.

3.2.1 Supervised Learning

Supervised learning refers to the availability of correct information for the learning task. In the simplest terms, it implies to the training data that is labeled with corresponding outputs of the learning task. The unknown parameters of the learning task are determined by minimizing the cost function that depends on the training data. Supervised learning dates back to the regression and interpolation methods proposed centuring ago by Gauss [25], where the answers are calculated based on the correct mixture of interpolating values is to be found. The choice of proper loss functions reflect different constraints on the learning task such as sparsity [26][27]. An appropriate approximating function is chosen based on the prior knowledge about the data.

3.2.2 Unsupervised Learning

Unsupervised learning refers to the extraction of certain features that may be present in the data but are not readily seen. These features are extracted by implementing certain conditions, where the supervision is not needed for the result. The problems that deal in unsupervised learning include dimensionality reduction, clustering etc.

3.2.3 Semisupervised Learning

Semisupervised learning tasks can been seen as something between supervised and unsupervised learning, where partial supervision is needed. In such cases, we either have limited amount of labeled data or we have limited amount of correct information.

3.3 Neural Networks

Neural networks are the most famous constructs of supervised learning, where we have labeled data and we want to discover the relationship present between the input and the outputs. Neural networks are nonlinear function approximators. These are constructed taking the inspiration from the human brain and how it processes, stores and retireves information.

According to the universal approximation theorem [28], any kind of function can be estimated with the application of a sufficiently large and deep neural network. The popularity of neural

networks can be accredited to their modular structure, which is based on the neurons, that acts as the central building element. Each neuron is associated with a number of inputs, a number of outputs, one activation function. Everytime a neuron receives an input, the activation function fires based on some threshold criteria, and the neuron produces an output. Multiple neurons can be combined into various configurations based on the type of problem and type of data involved. Forward propagating neural networks are the most common structures that consists of layers of neurons, where the output of one layer, acts as the input for the next layer. Every neural network has an input layer, that receives the labeled, input data, and an output layer, that produces the prediction for the repective labeled quantities. A nonlinear optimization method, like backpropagating neural network [29], are employed in learning tasks where we want to identify the network weights in order to minimize the error between the predicted and labeled training data. Deep neural networks consists of multiple layers where each layer contains multiple neurons and various types of nonlinear activation functions. Neural networks have been widely employed to model heat transfer [30], turbomachinery [31], turbulent flows [32], and other problems involved in aeronautics [33].

Even though neural networks are widely known and they hold many promising applications, there are several challenges while implementing a neural network. On a broader level, neural networks are just interpolating architectures, thus, the function that is to be estimated, is only estimated efficiently under the span of the sampled data that is used to train the network. Thus, neural networks must be implemented with utmost care when they are to be used for an extrapolation task. In many learning problems, like speech recognition or computer vision, the data present is so vast that, crudely, it can be estimated the all future data that will be generated, will roughly be an interpolation of the present data that can be used for training of the neural network. For problems like speech recognition, computer vision, etc., the data is being continuously generated and thus, the training data present is very vast. This is not the case for fluid mechanics problems, because training of that scale has not been achieved till date. Similarly, neural networks are prone to problems like overfitting, wherein, the model performs exceptionally well on training data, but very poorly on testing data, or underfitting, where a lact of sufficient amount of training data leads to an ill-trained neural network. The problem of overfitting can be taken care of by using cross-validating the model on carefully chose test dataset. The best practices that are to be followed while constructing a neural network are discussed in [34]. Finally, since fluid flows are physical in nature, thus, they follow certain

symmetries, laws and constraints. The physics of the problem must be incorporated in the model, at least to an extent.

3.4 Structure

An open-source python library, Pytorch was used to construct, train, validate and test the ANN model for predicting the reactive scalar quantities. A systematic study was performed to find the optimal number of hidden layers and neurons in each layer that are required to be used.

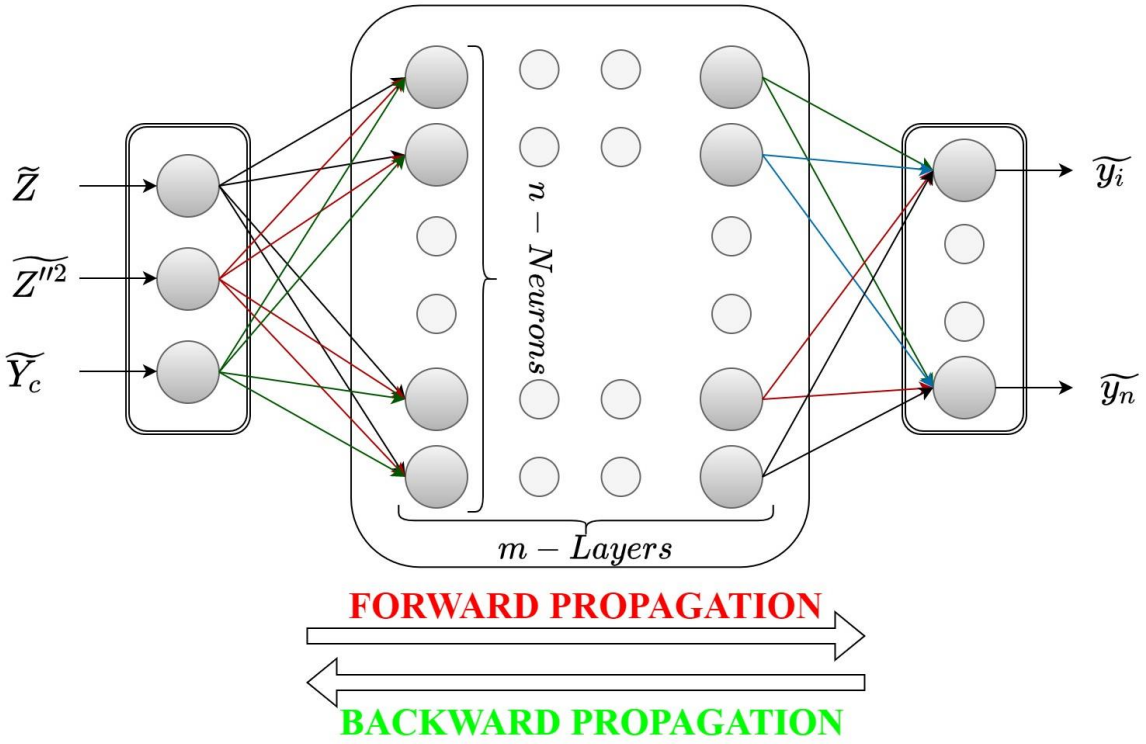


Figure 3.1: Basic structure of ANN

The ANN model consisted of 3 input – 34 output systems (32 species + temperature + reaction source term of Progress variable). The model is characterized by a number of independent parameters that collectively dictate the performance of the ANN in predicting the values.

a. Weights: These are values that are used to signify the importance of a certain feature in the final output. So, we use weights to amplify/diminish the importance of a feature. Weights signify the relative importance of a feature.

b. Weight initialization techniques

These can be used to reach convergence faster. Let f_i be the number of inputs and f_o be the number of outputs from a neuron. The weight for neuron i in layer j is, W_{ij}, and this can be calculated using the following techniques:

1. Uniform distribution: The weights are chosen from a uniform distribution in the given bounds.

$$W_{ij} \sim U\left[\frac{-1}{\sqrt{f_i}}, \frac{1}{\sqrt{f_i}}\right]$$

2. Xavier and Groat techniques

   i. Xavier Normal distribution and

$$W_{ij} \sim N(0, \sigma)$$

$$\sigma = \sqrt{\frac{2}{f_i + f_o}}$$

   ii. Xavier Uniform distribution

$$W_{ij} \sim U\left[\frac{-\sqrt{6}}{\sqrt{f_i + f_o}}, \frac{\sqrt{6}}{\sqrt{f_i + f_o}}\right]$$

3. He Init techniques

   i. He Normal distribution

$$W_{ij} \sim N(0, \sigma)$$

$$\sigma = \sqrt{\frac{2}{f_i}}$$

   ii. He Uniform distribution

$$W_{ij} \sim U\left[-\sqrt{\frac{6}{f_i}}, \sqrt{\frac{6}{f_i}}\right]$$

c. Learning rate (α): It controls the amount by which the weights are changed in each updating step.

d. Activation function: This can be thought of as a trigger, that causes a certain kind of reaction.

1. ReLU (Rectified Linear Unit): It converts the y value into its corresponding value if the y is positive, and if it is negative, then the output is 0.

$$ReLu(y) = \max(y,\ 0)$$

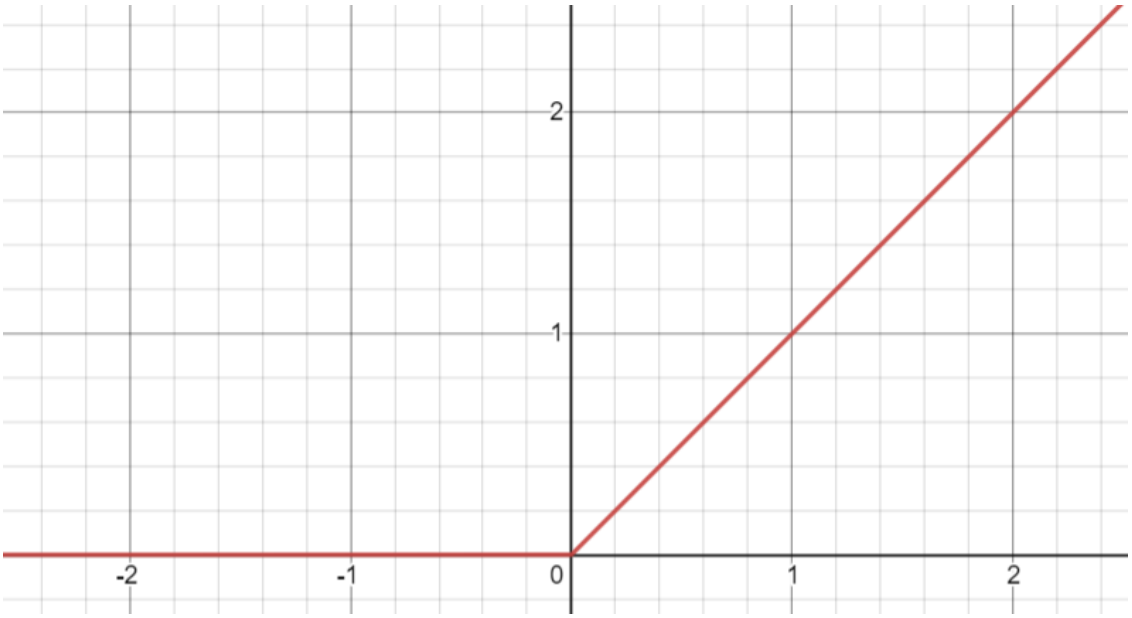


Figure 3.2: Graph of ReLU

2. Leaky ReLU: Instead of giving no output when input is negative, it gives a mild output when the signal is below zero.

$$Y\ = max(0.01 * y,\ y)$$

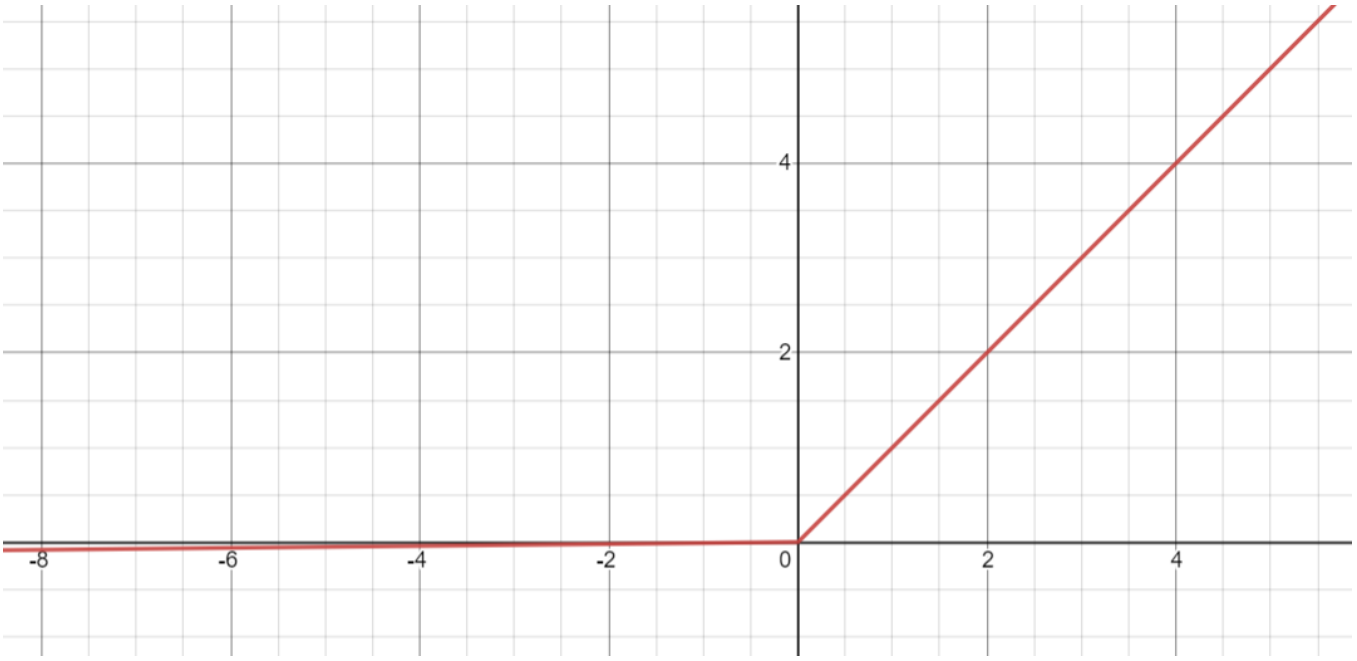


Figure 3.3: Graph of Leaky ReLU

3. ELU (Expoential Linear Unit)

$$Y = y, \ if \ y > 0$$

$$Y = a * (e^{y} - 1), \ otherwise$$

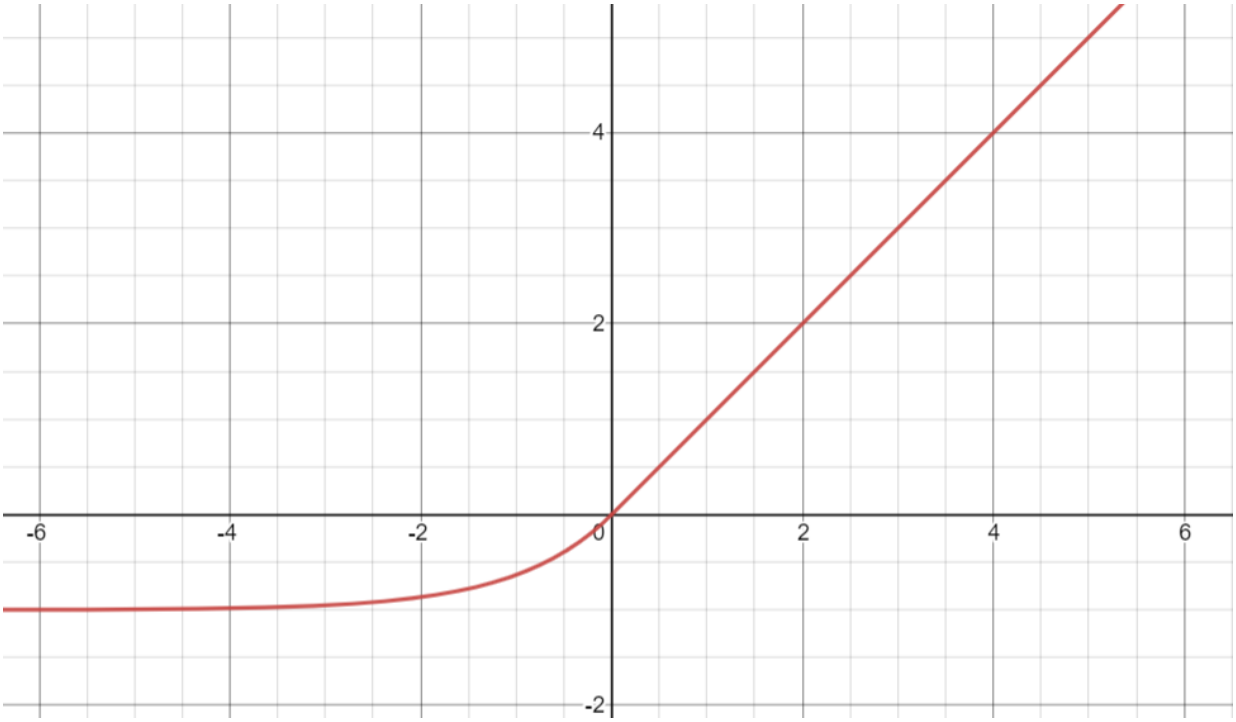


Figure 3.4: Graph of ELU

4. Sigmoid: It is used to convert the y-value into a value between 0 and 1.

$$Y = Sig(y) = \frac{1}{1 + e^{-y}}$$

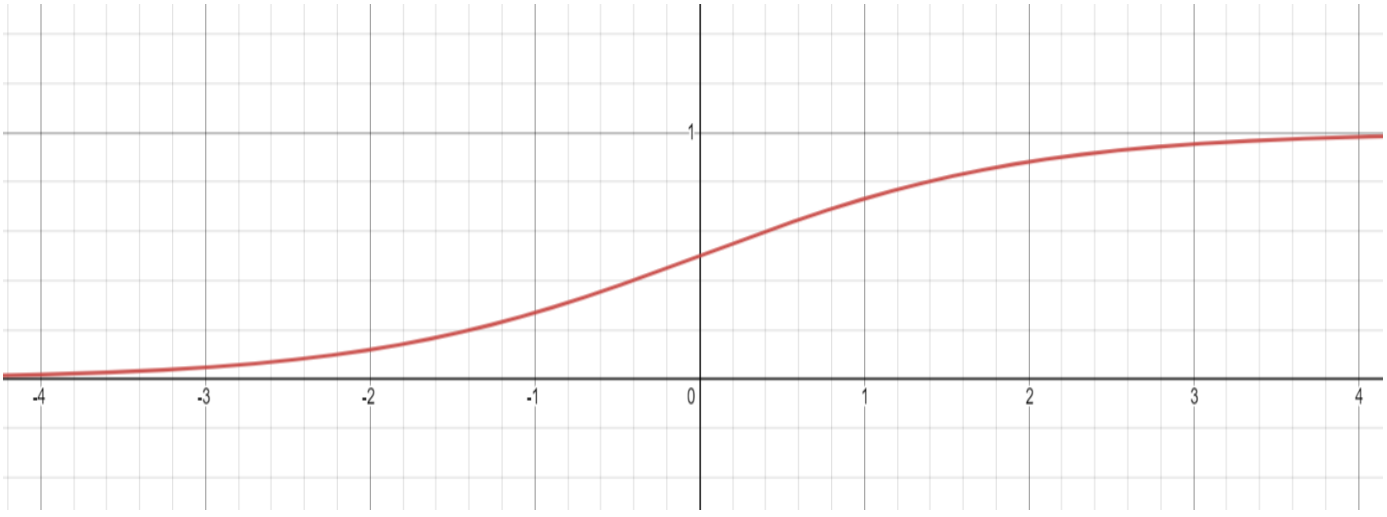


Figure 3.5: Graph of Sigmoid

5. Swish

$$Y = y * Sig(y)$$

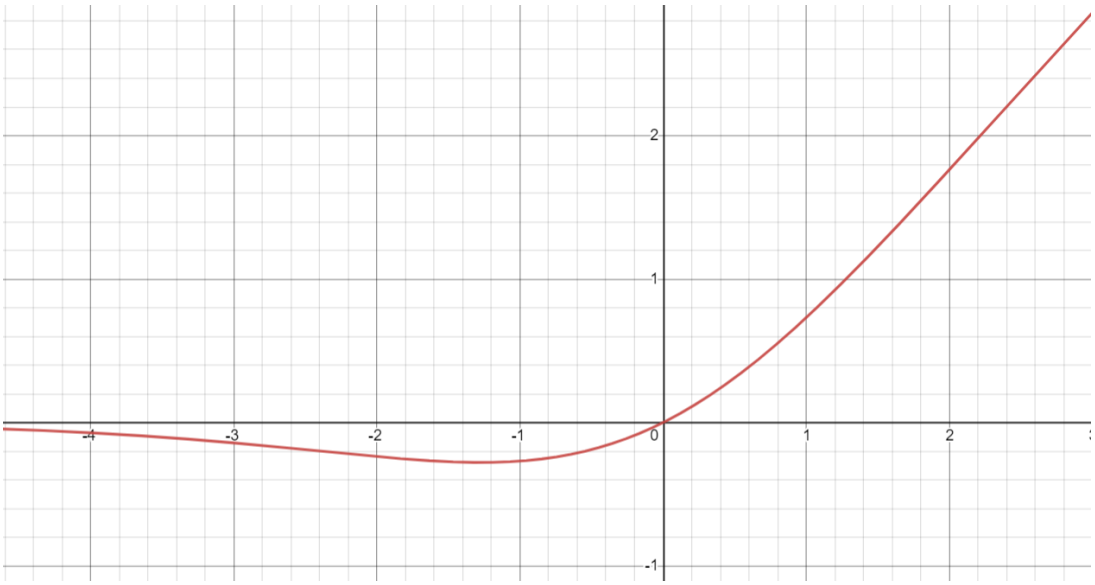


Figure 3.6: Graph of Swish

6. Soft Plus

$$Y = ln(1 + e^{y})$$

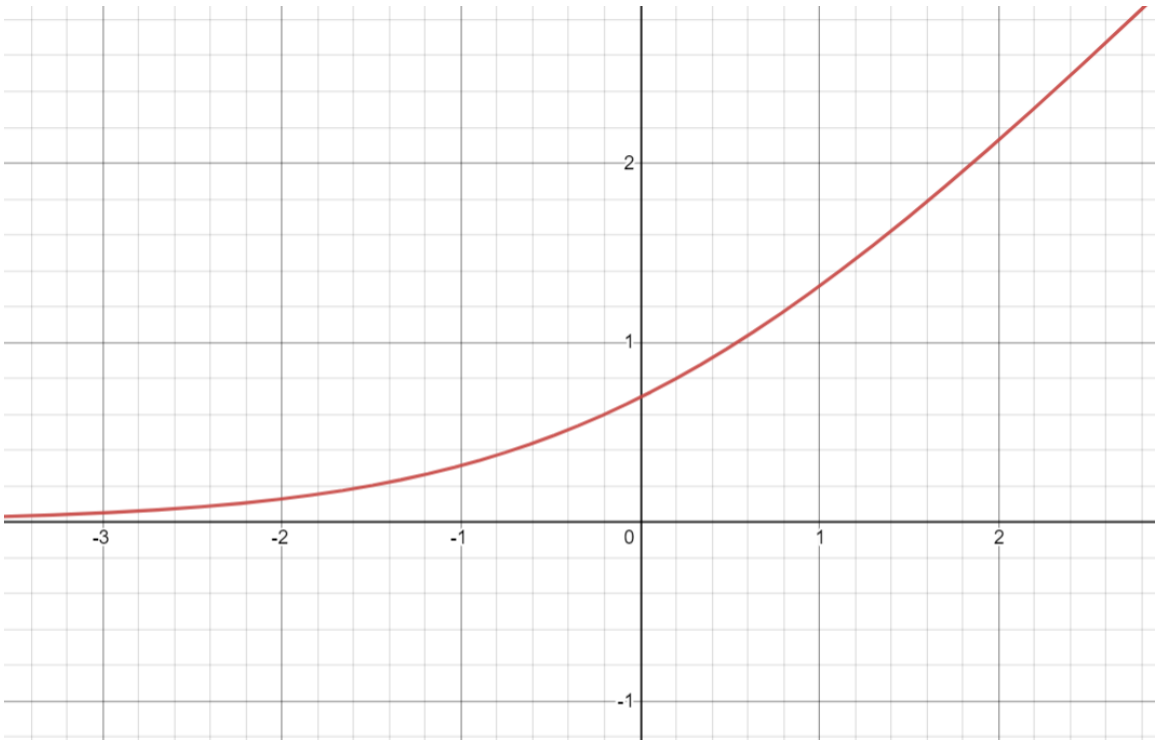


Figure 3.7: Graph of Softplus

e. Loss function: These are used to calculate the loss or error after each epoch and then the weights are corrected in the backpropagation step. Here, y is the correct value and $\hat{y}$ is the predicted value.

1. Squared error loss

$$Loss = \sum (y - \hat{y})^2$$

2. Absolute error loss

$$Loss = \sum |y - \hat{y}|$$

3. HUBER loss

$$Loss = \sum (y - \hat{y})^2,\ if\ |y - \hat{y}| \leq \delta$$

$$Loss = \sum (\delta * |y - \hat{y}| - 0.5 * \delta^2)\ ,\ otherwise$$

where δ is a predefined quantity.

f. Optimizers

1. Gradient Descent

$$\theta_j = \theta_j - \alpha \sum [(h_\theta(x_i) - y_i * x_i)]$$

2. SDG (Stochastic Gradient Descent)

$$\theta = \theta - \alpha(h_\theta(x) - y * x)$$

3. Mini Batch SDG

$$\theta_j = \theta_j - \alpha \sum_{i=1}^{i=m<N} [(h_\theta(x_i) - y_i * x_i)]$$

4. ADAM

$$W = W - \alpha \frac{vdW}{sdW^2 + \rho}$$

$$b = b - \alpha \frac{vdb}{sdb^2 + \rho}$$

The accuracy of prediction of the ANN depends on the training paramters and complexity of the model. Accounting for the deviation of ANN predictions from FPV data for each parameter is not feasible, thus to get a better understanding of the correlation between the FPV data and

ANN predictions, we can use a measure of the linear correlation between the two. We have used PPMCC (Pearson product-moment correlation coefficient), denoted by R [23]. It is the ratio between the covariance of two variables and the product of their standard deviations. It gives a value between [-1, 1], where -1 depicts a negative correlation, meaning the two quantities vary inversely, 1 depicts a positive correlation, meaning the two quantities vary proportionaly, and 0 depicts that the two quantities are not related to each other.

$$R = \frac{\sum(y_{ANN,j} - \bar{y}_{ANN})(y_{FPV,j} - \bar{y}_{FPV})}{\sqrt{\sum(y_{ANN,j} - \bar{y}_{ANN})^2}\sqrt{\sum(y_{FPV,j} - \bar{y}_{FPV})^2}}$$

Here, $y_{ANN,j}$ is the j[th] datapoint predicted by the ANN

$y_{FPV,j}$ is the j[th] datapoint of the FPV model

$\bar{y}_{ANN}$ is the average predicted value of that species

$\bar{y}_{FPV}$ is the average value of the species

3.5 Specifications of Models

We have developed 9 different models that vary in the number of layers, nodes, training epochs, learning rate etc. to understand the effect of various parameters on the training of the ANN. The following table lists the details about the models.

Model number	Number of layers	Number of nodes in each layer
Model 1	1	150
Model 2	5	50-100-100-75-50
Model 3	8	25-25-50-70-70-50-25-25
Model 4	1	50
Model 5	5	50-50-50-50-50
Model 6	8	50-50-50-50-50-50-50-50
Model 7	7	1-2-3-4-3-2-1
Model 8	7	25-25-50-50-50-25-25
Model 9	7	75-75-125-125-125-75-75

Table 3.1: Properties of all ANNs

Chapter 4: Results and Discussions

4.1 Tabulated Chemistry Results

Using the flamelet tables constructed previously, we generated the data with respect to all the three variables, $\tilde{Z}, \tilde{C}$ and $\widetilde{Z''}$. The variation is shown in the following graphs:

a. Temperature

i. Z and Zeta

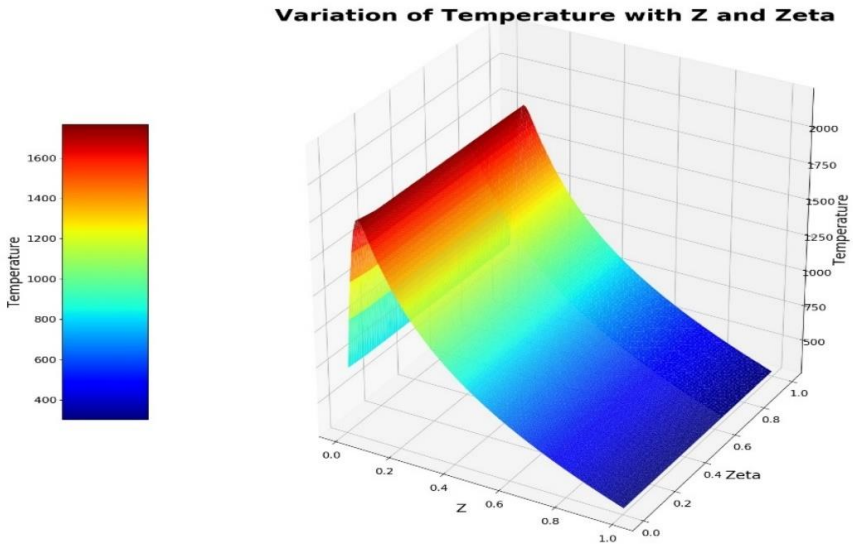


Figure 4.1: T vs. Z and Zeta

ii. Z and Yc

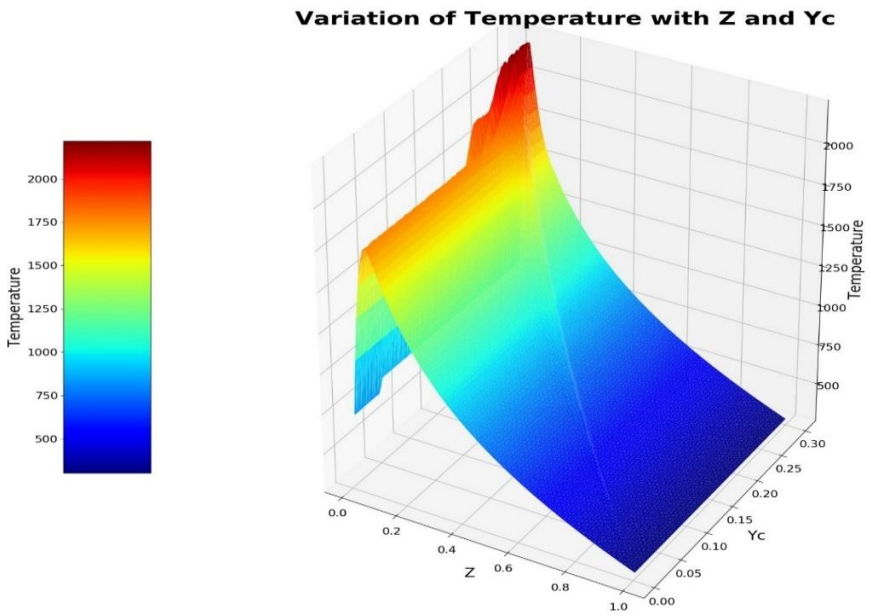


Figure 4.2: T vs. Z and Yc

iii. Zeta and Yc

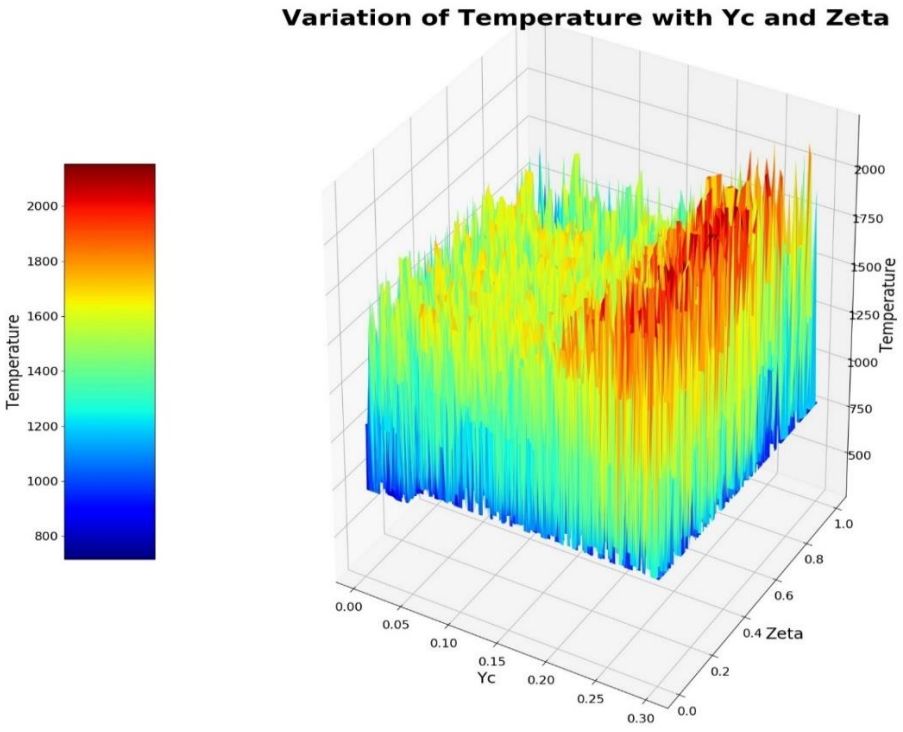


Figure 4.3: T vs. Yc and Zeta

b. OmegaYc

i. Z and Zeta

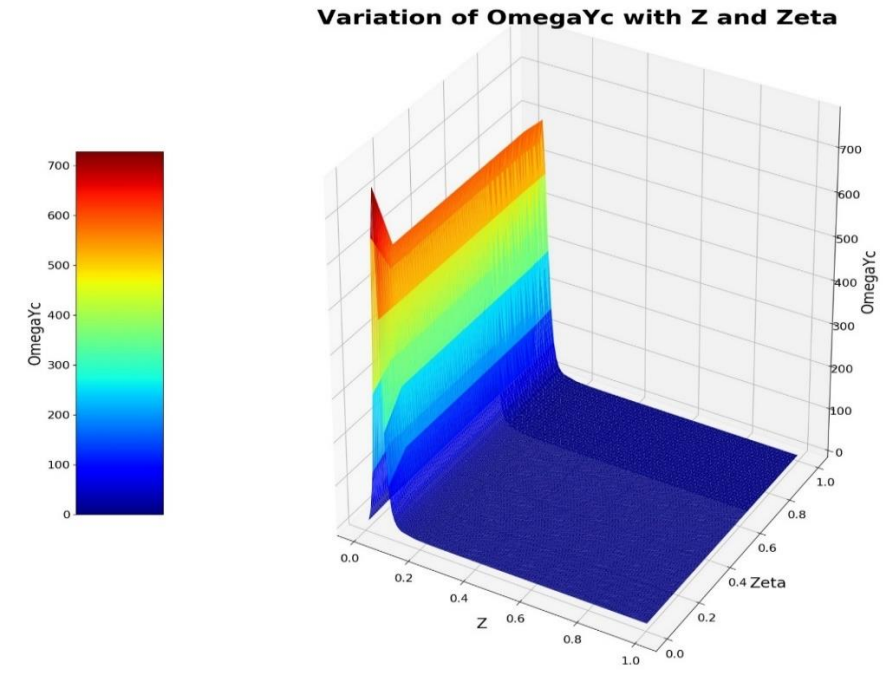


Figure 4.4: OmegaYc vs. Z and Zeta

ii. Z and Yc

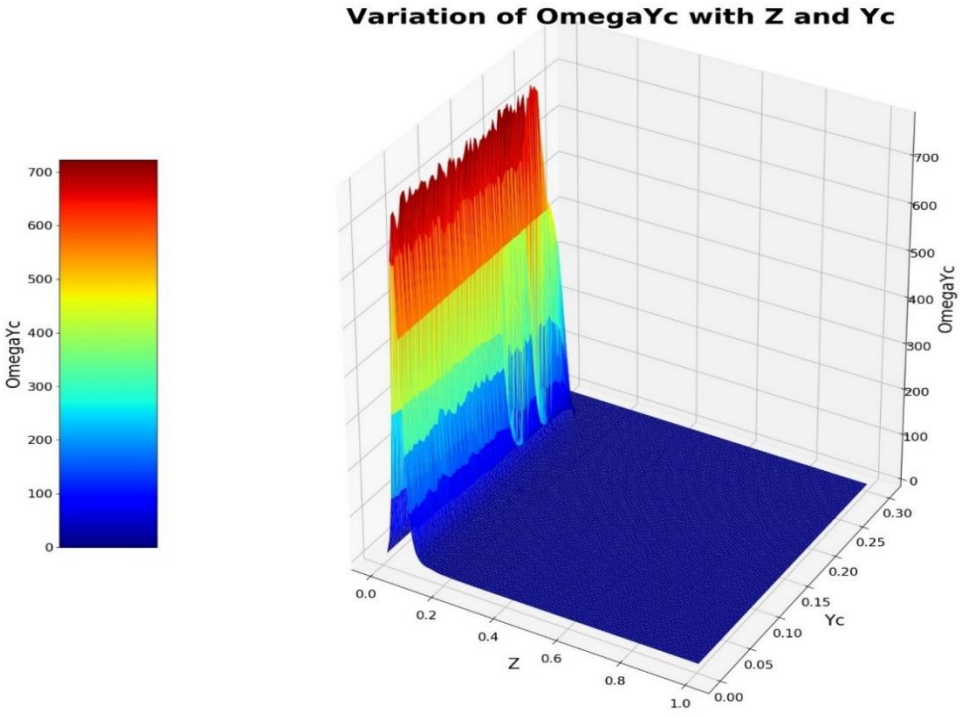


Figure 4.5: OmegaYc vs. Z and Yc

iii. Zeta and Yc

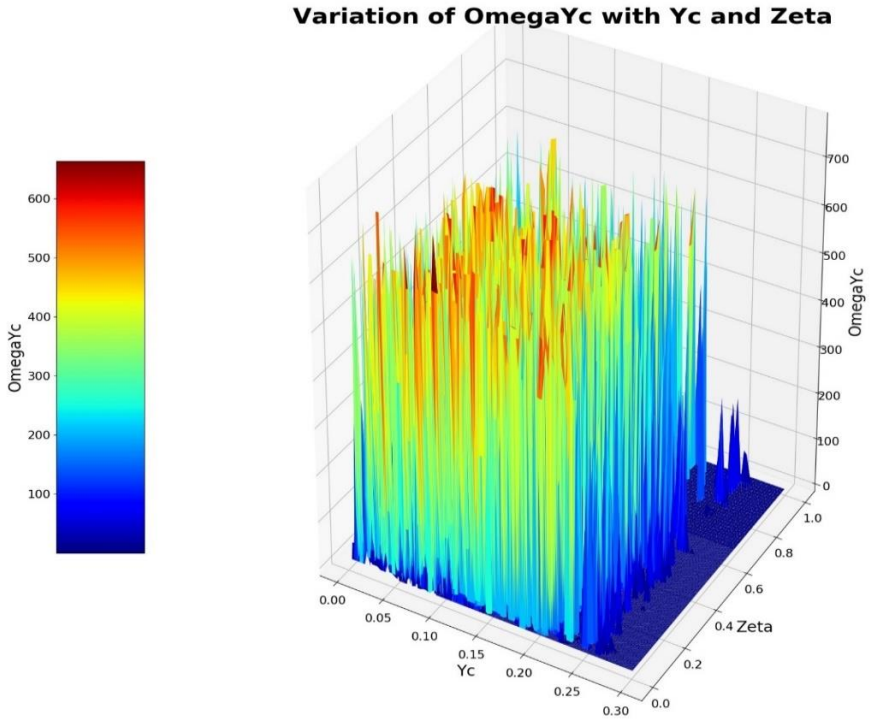


Figure 4.6: OmegaYc vs. Yc and Zeta

c. CO_2

i. Z and Zeta

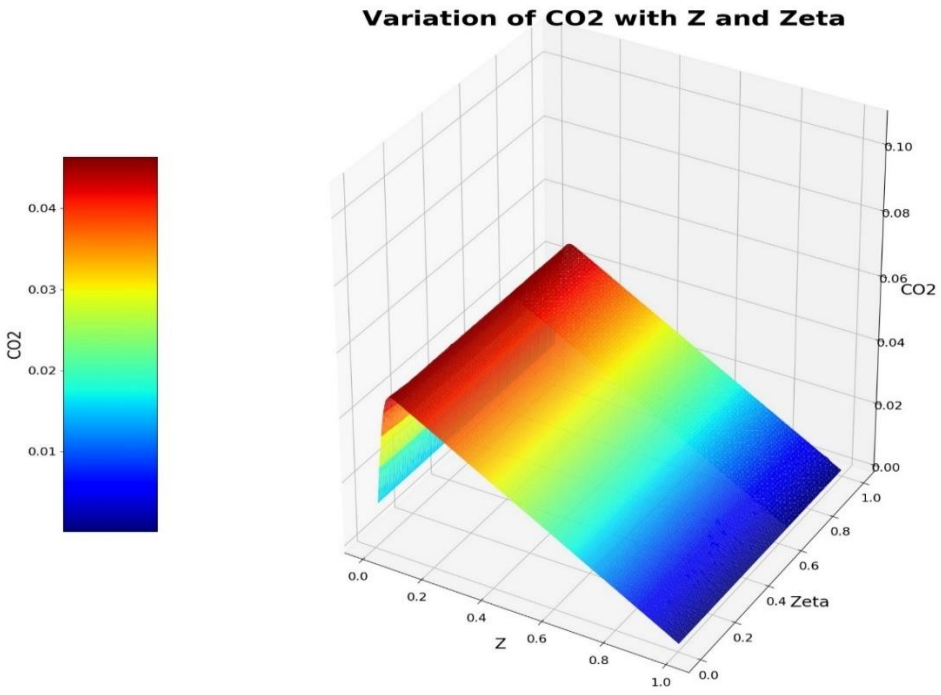


Figure 4.7: CO_2 vs. Z and Zeta

ii. Z and Yc

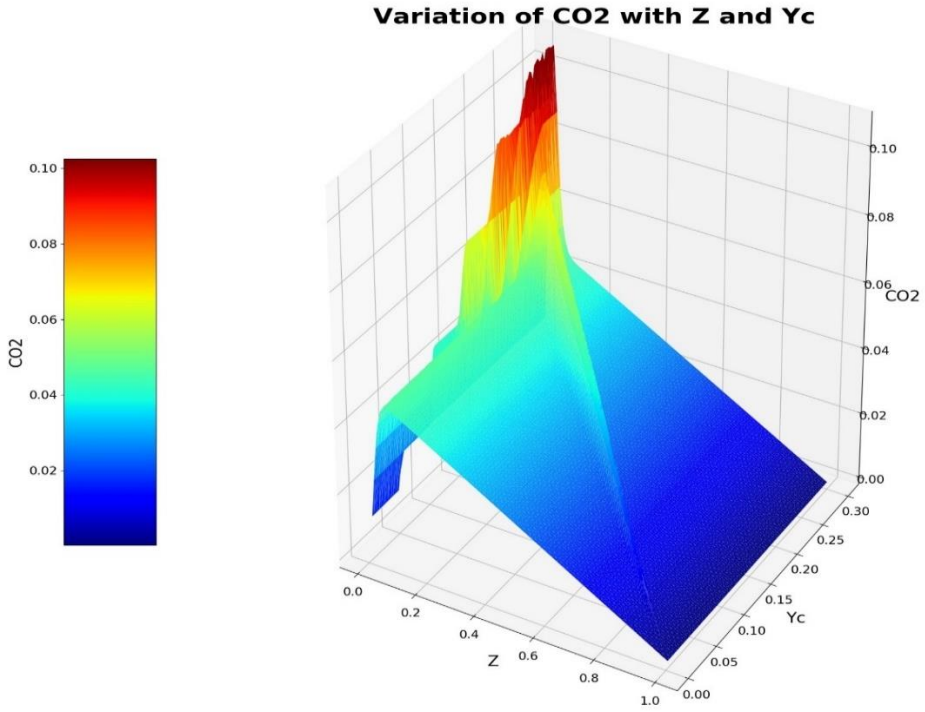


Figure 4.8: CO_2 vs. Z and Yc

iii. Zeta and Yc

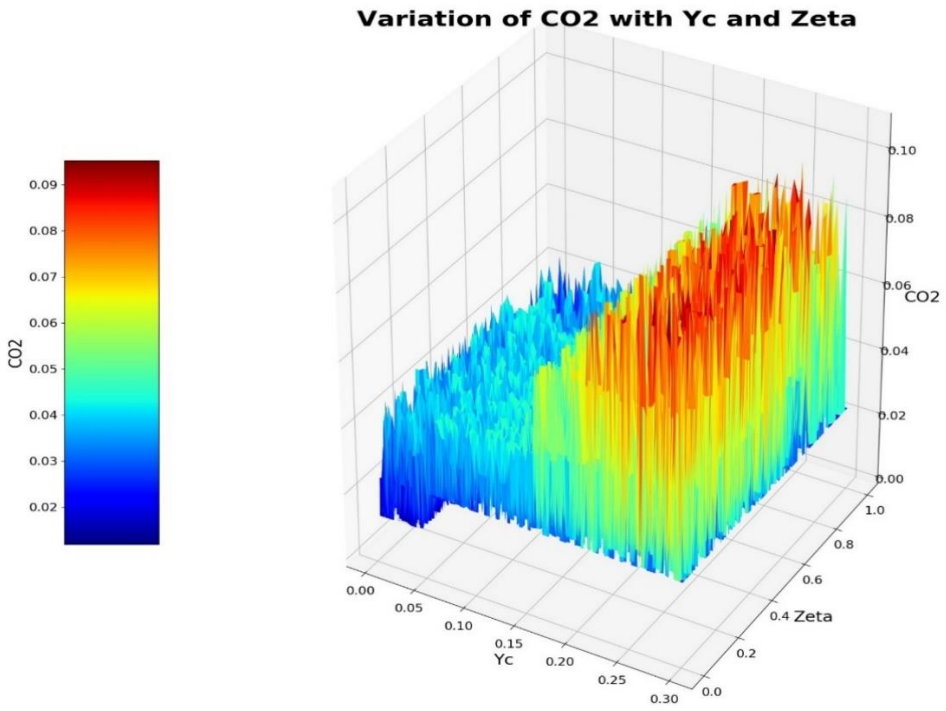


Figure 4.9: CO_2 vs. Yc and Zeta

d. H_2O

i. Z and Zeta

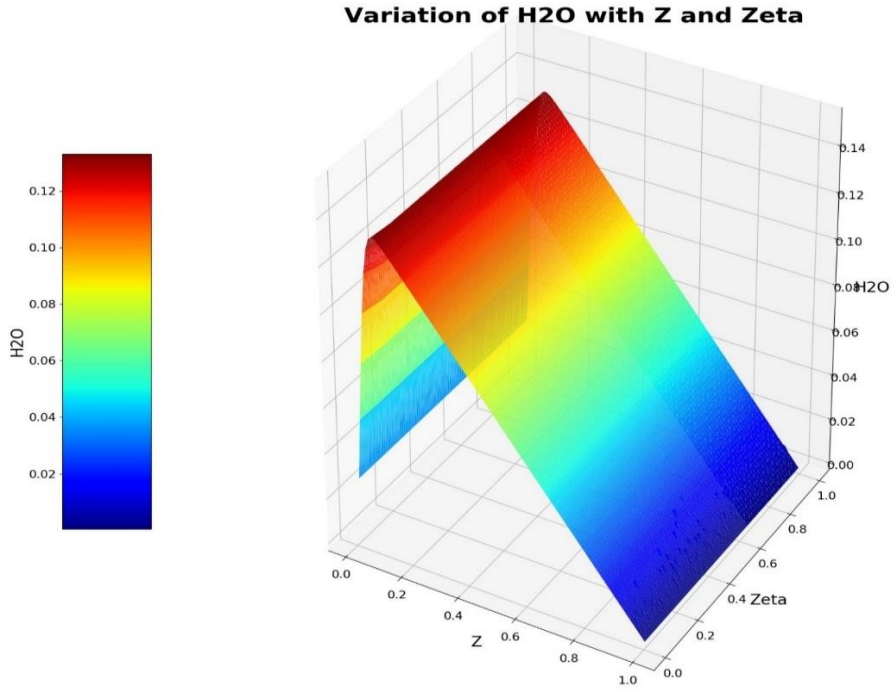


Figure 4.10: H_2O vs. Z and Zeta

ii. Z and Yc

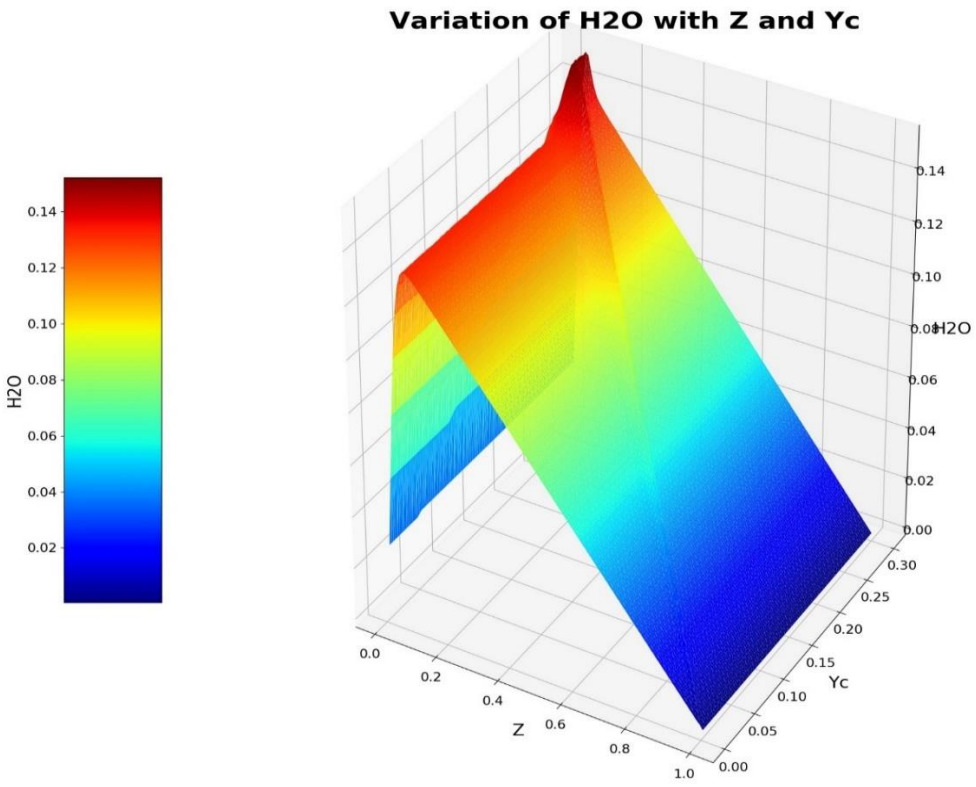


Figure 4.11: H_2O vs. Z and Yc

iii. Zeta and Yc

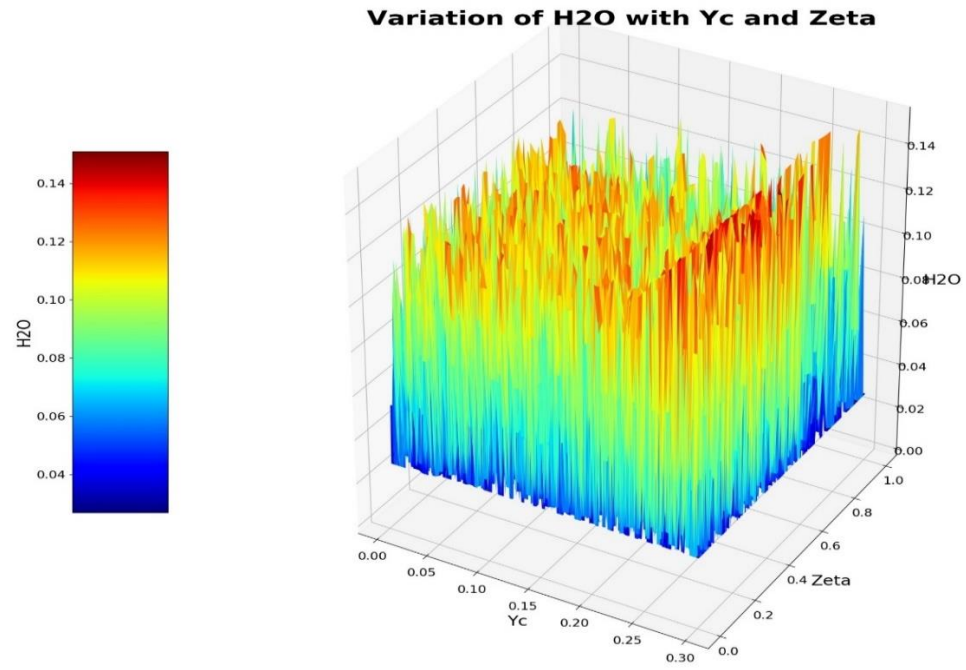


Figure 4.12: H_2O vs. Yc and Zeta

e. CH_3OH

i. Z and Zeta

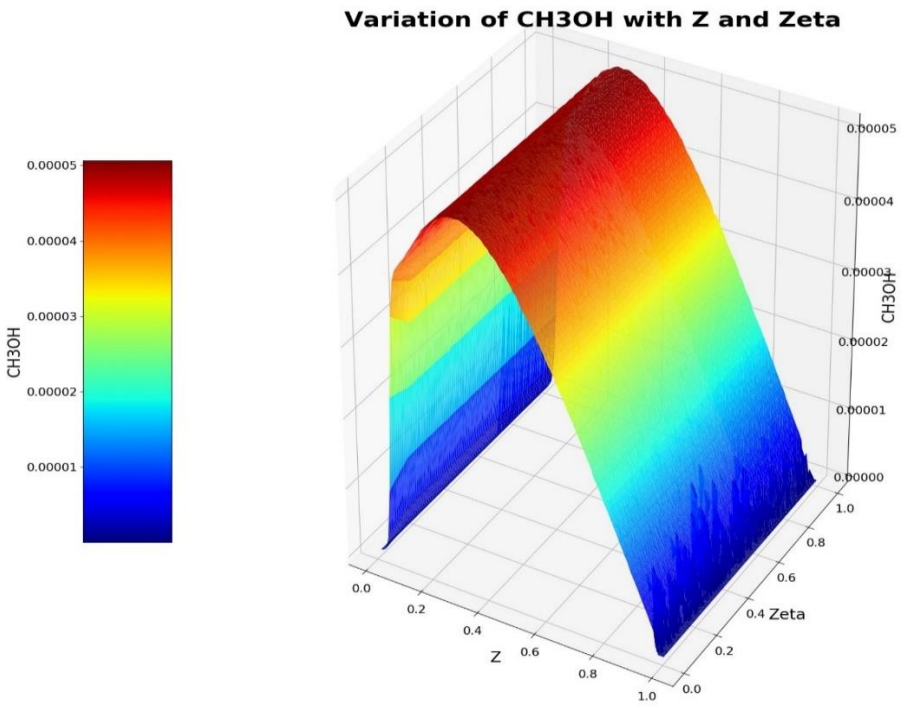


Figure 4.13: CH_3OH vs. Z and Zeta

ii. Z and Yc

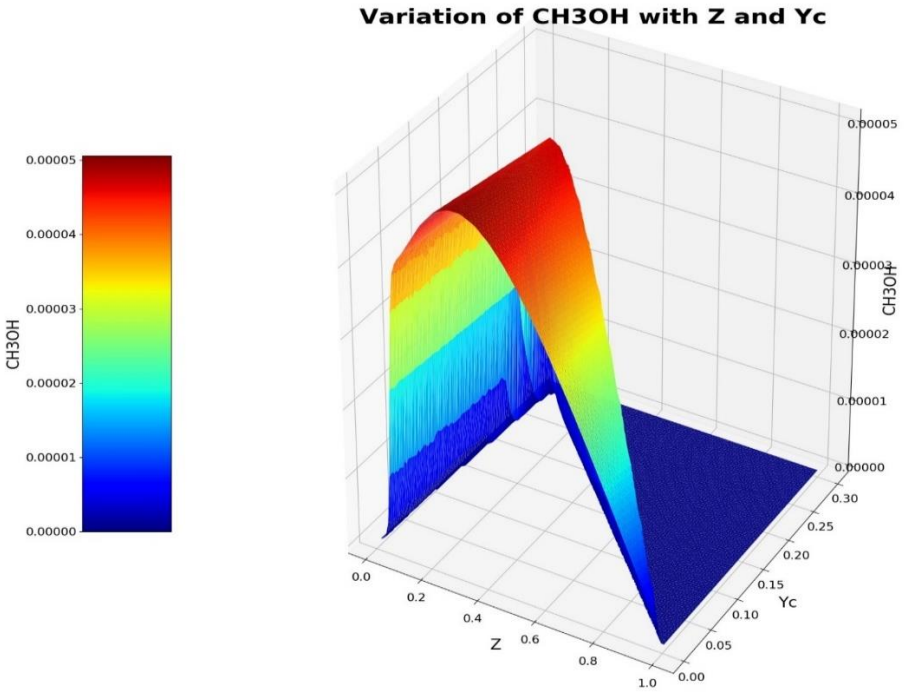


Figure 4.14: CH_3OH vs. Z and Yc

iii. Zeta and Yc

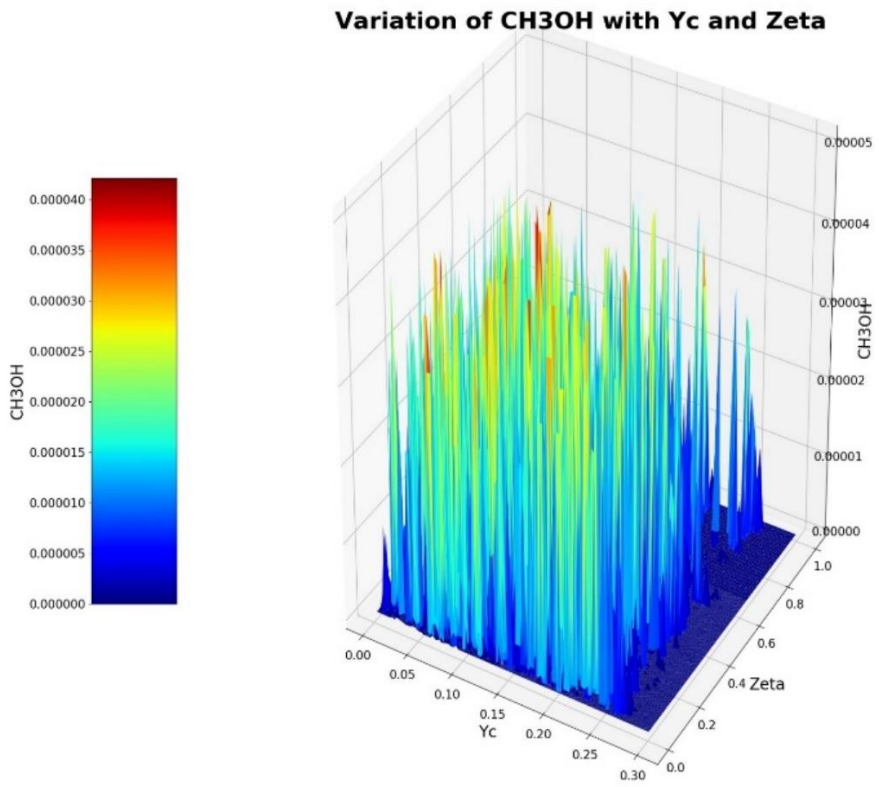


Figure 4.15: CH_3OH vs. Yc and Zeta

We can conclude the following points from the above plots:

i. All the reactive scalars vary continuously with Z and Zeta.

ii. All the reactive scalars vary almost continuously with Z and Yc.

iii. All the reactive scalars have very high fluctuations with respect to Yc and Zeta, as Zeta is a highly fluctuating quantity, it makes the plots discontinuous and appear like sharp pointed edges.

The new data generated with respect to Z, Zeta and Yc was used to train the neural network and subsequent analysis was carried out.

4.2 ANN Results

The entire FPV data was converted into a single ANN file that was gives a input to the ANN. The input was divided into three parts, training dataset, validation dataset and testing dataset. A systematic study was conducted to understand the prediction accuracy of the model based on the number of layers and nodes present in each layer. A total of 9 different models were created, all of them were trained on the same training set, then validated and tested on the sama datasets and their performace was studied. The following table shows the properties of the models.

Model number	Number of layers	Number of nodes in each layer	Least PPMCC
Model 1	1	150	0.52284
Model 2	5	50-100-100-75-50	0.98334
Model 3	8	25-25-50-70-70-50-25-25	0.94301
Model 4	1	50	0.56196
Model 5	5	50-50-50-50-50	0.96266
Model 6	8	50-50-50-50-50-50-50-50	0.98428
Model 7	7	1-2-3-4-3-2-1	0.0
Model 8	7	25-25-50-50-50-25-25	0.97219
Model 9	7	75-75-125-125-125-75-75	0.98429

Table 4.1: Properties and least PPMCC of all ANNs

In the above table, the least value of PPMCC (Pearson Product Moment Correlation Coefficient) is tabulated. The least PPMCC is calculated among all the reactive scalars, the reactive scalar with the least PPMCC is said to be the worst modeled species, and thus, can be used to compare the models. So, higher the value of the least PPMCC, better is the model.

From the above data, we can infer that Model 9 is the best model as it has the highest PPMCC. Thus, this model can be used further.

We can draw the same conclusion through the following procedure:

a. Find the model with the lowest cumulative error among all the species.
b. Once a model is finalized, find the reactive scalar that has the highest cumulative error in that particular model, thus, this species can be used to compare the performance of all the models.

The following graph shows the cumulative error across all species in every ANN model.

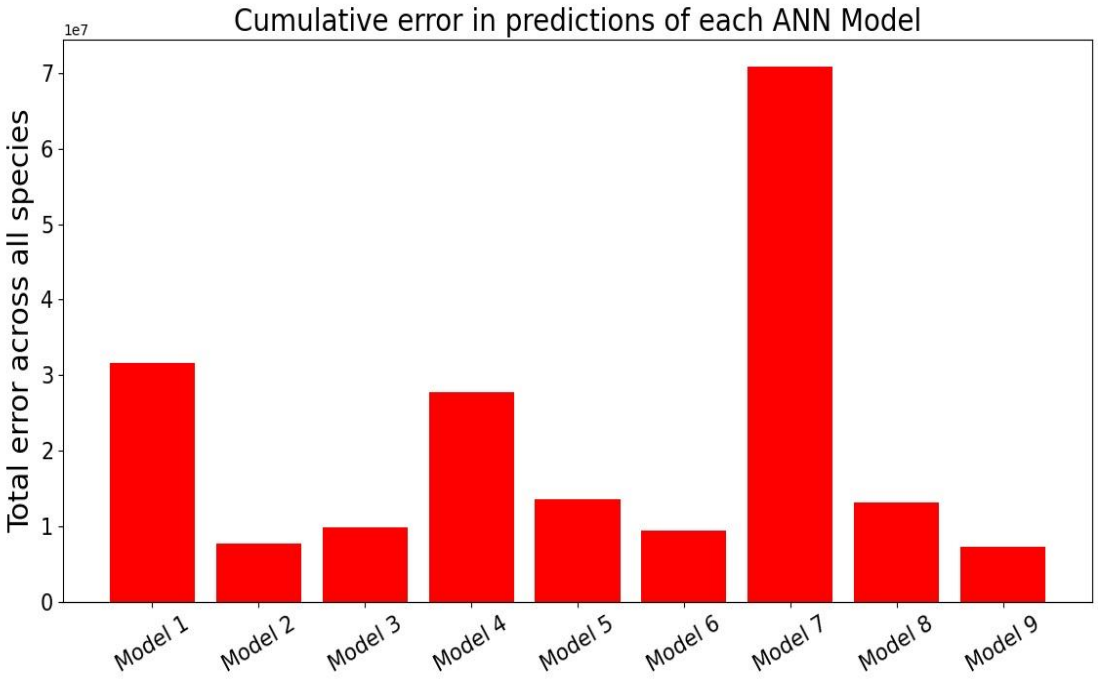


Figure 4.16: Cumulative error in each ANN

Once a model with the least error is finalized, i.e., Model 9, we find the total error in each species. This can be used to compare the models with each other.

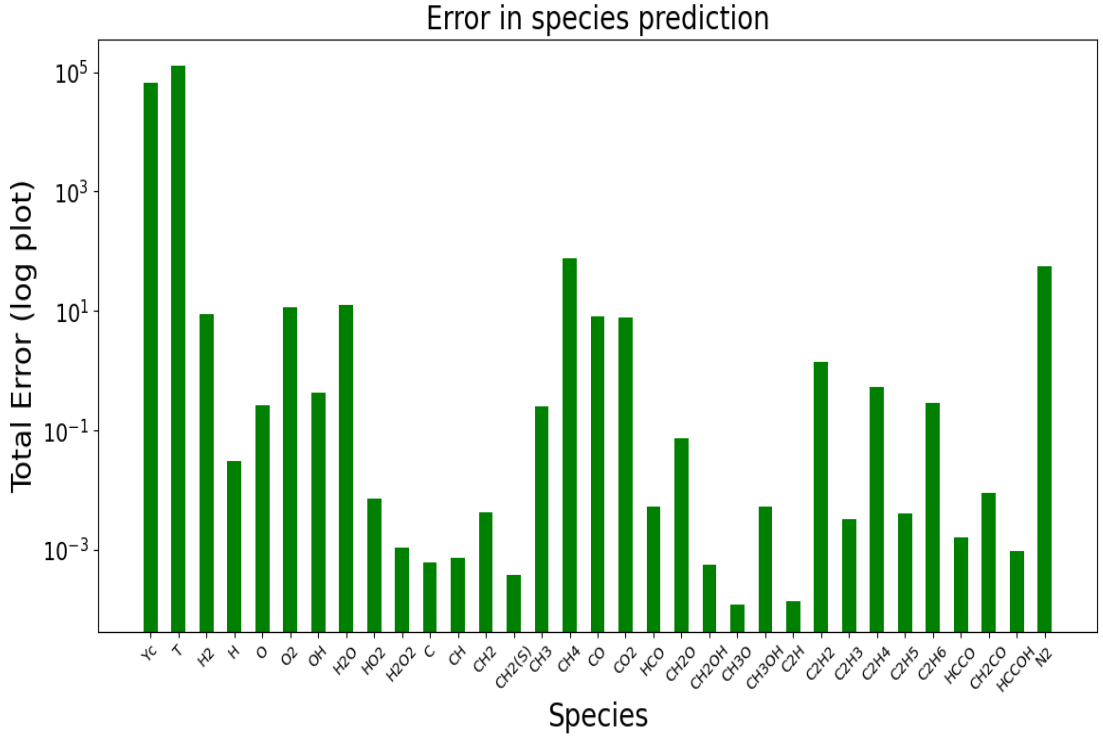


Figure 4.17: Error in each species

The following figure shows the deviation of the predicted values from the actual values. Here, the model that is closest to the x-axis is the best. We are plotting the difference between the the two corresponding datapoints of the ANN model and FPV model.

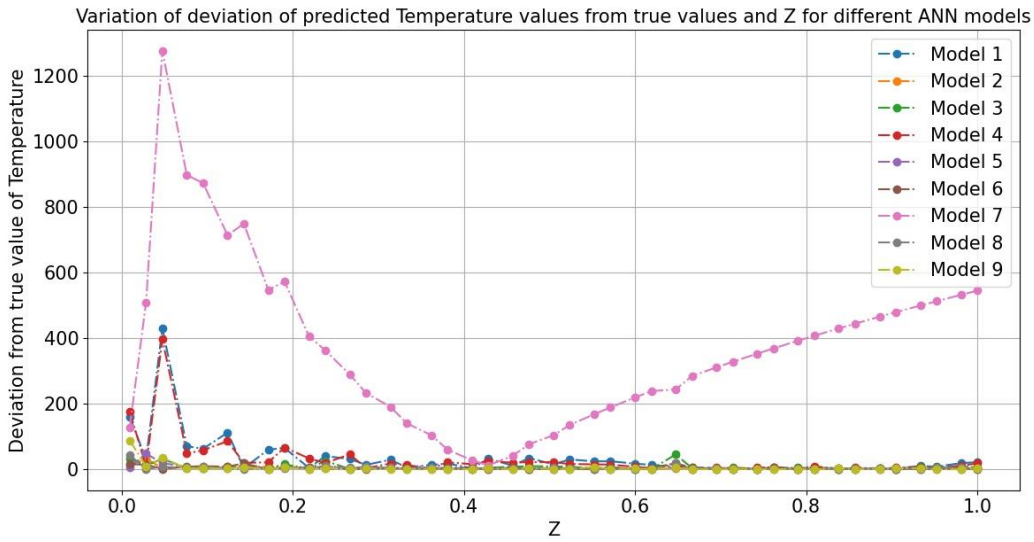
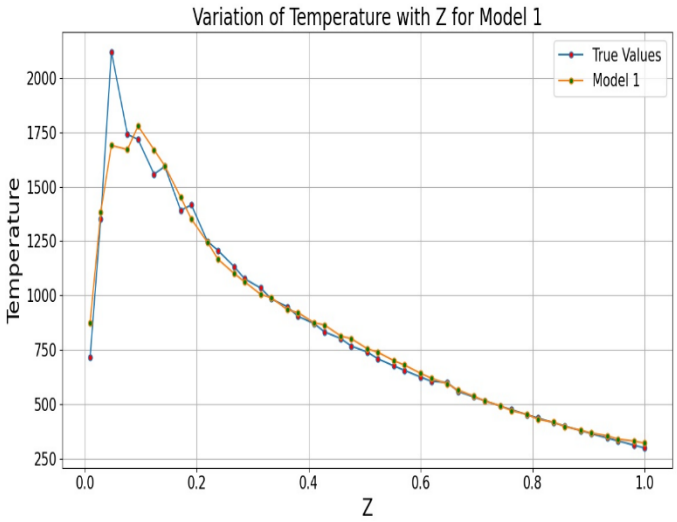
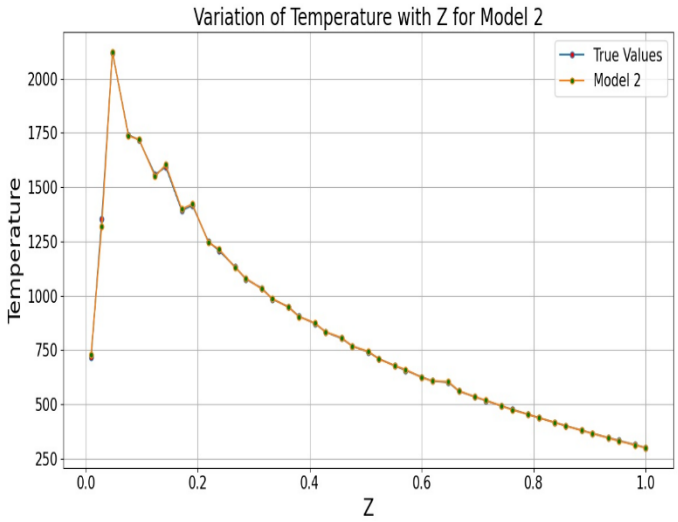
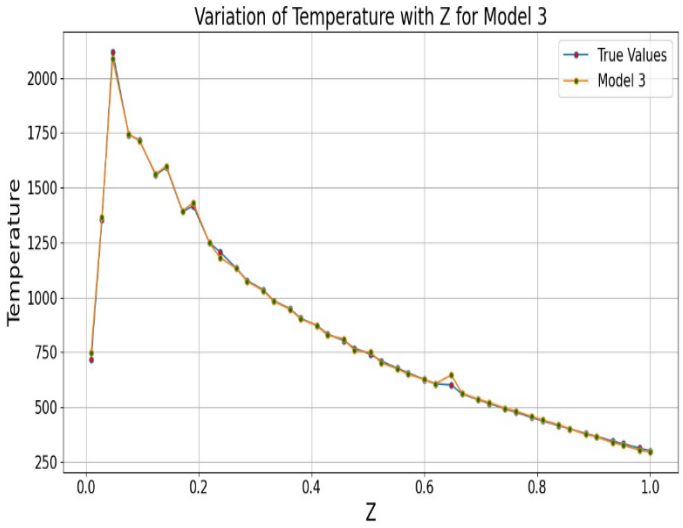


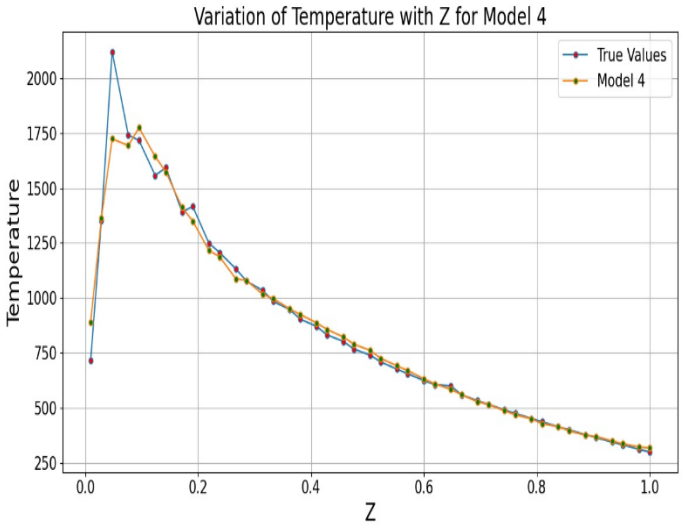
Figure 4.18: Deviation of Temperature for all models

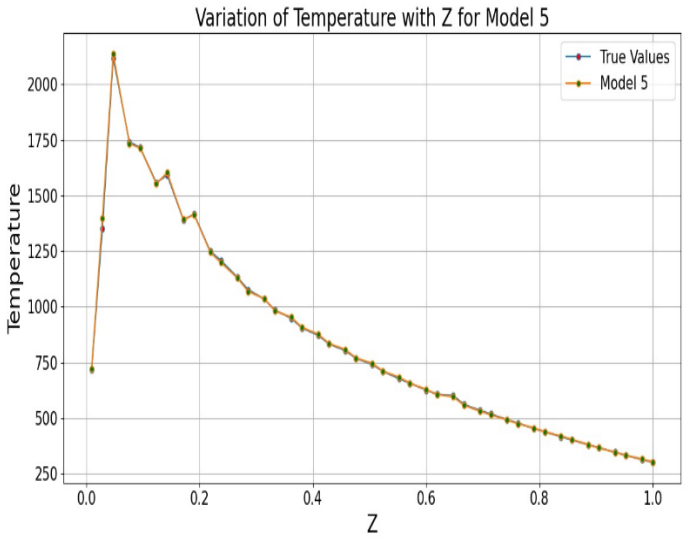
We can see in the above graph that model 7 has the largest deviation, thus, we can conclude that model 7 is of no use. But the difference is not so clear for other models. In order to tackle this problem, we plot the values of temperature for individual ANN model along with the true values as calculated using FPV model. Here, the model that gives the maximum overlap with the FPV values, is said to be the best model.

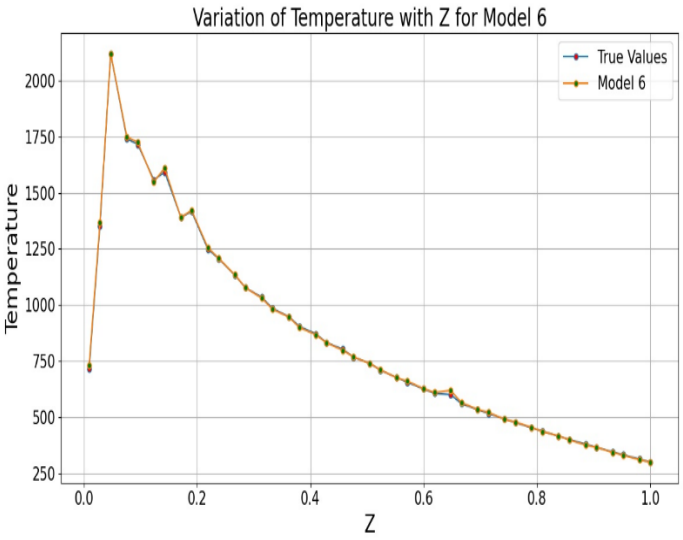


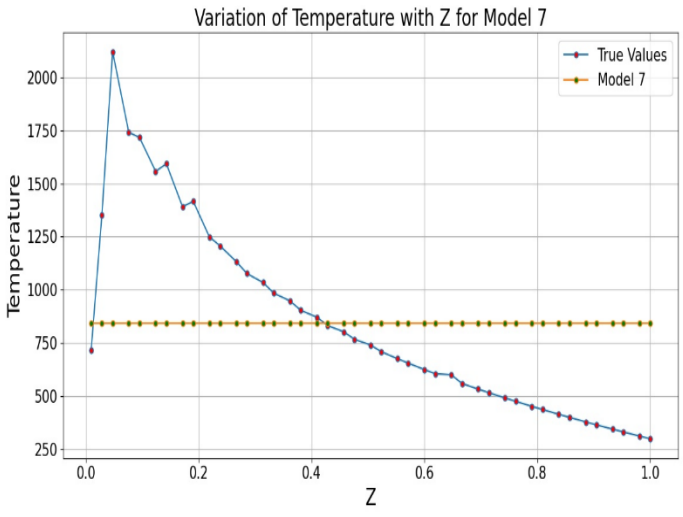












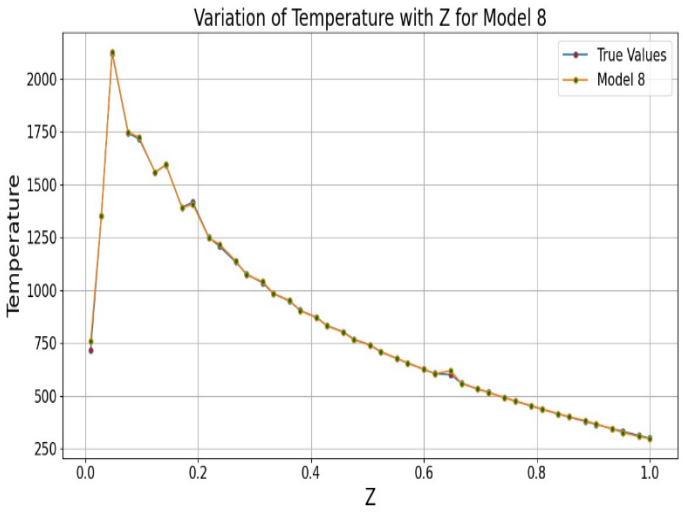


Figure 4.19: ANN data vs. FPV data for Temperature for various ANN models

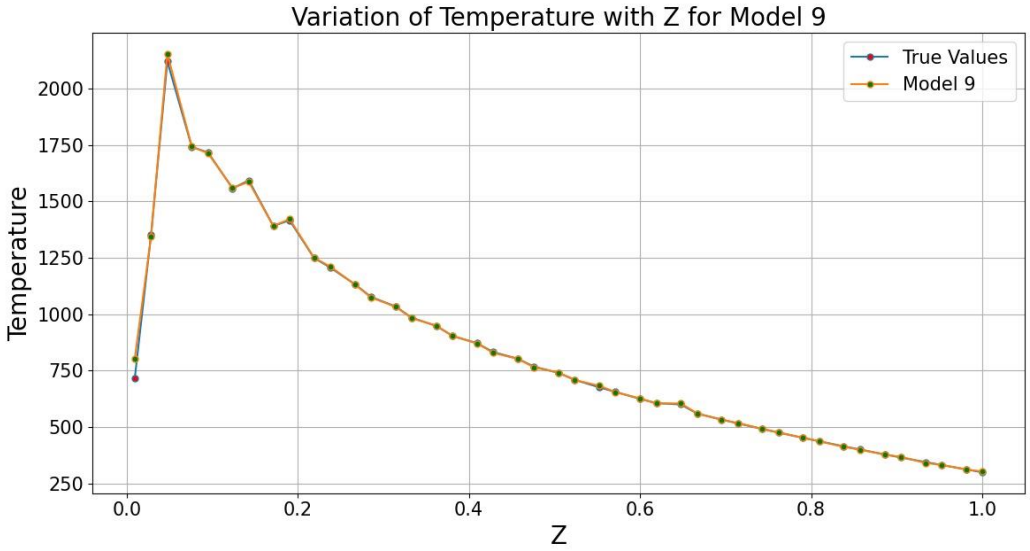


Figure 4.20: Comparison of ANN prediction and FPV data for Model 9

The values for temperature for all the ANN models and FPV model are plotted in the following graph.

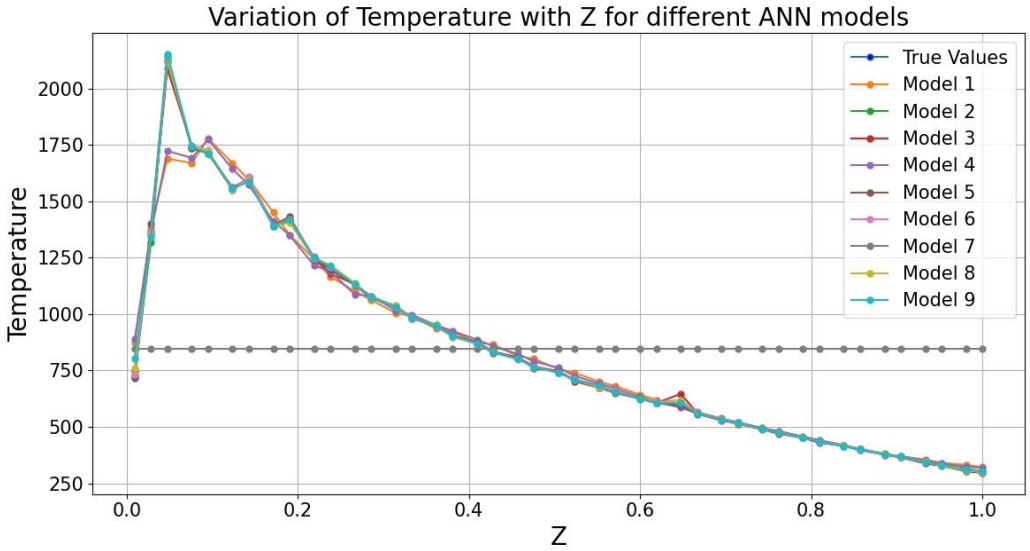


Figure 4.21: Agreement of ANN model with FPV model for Temperature

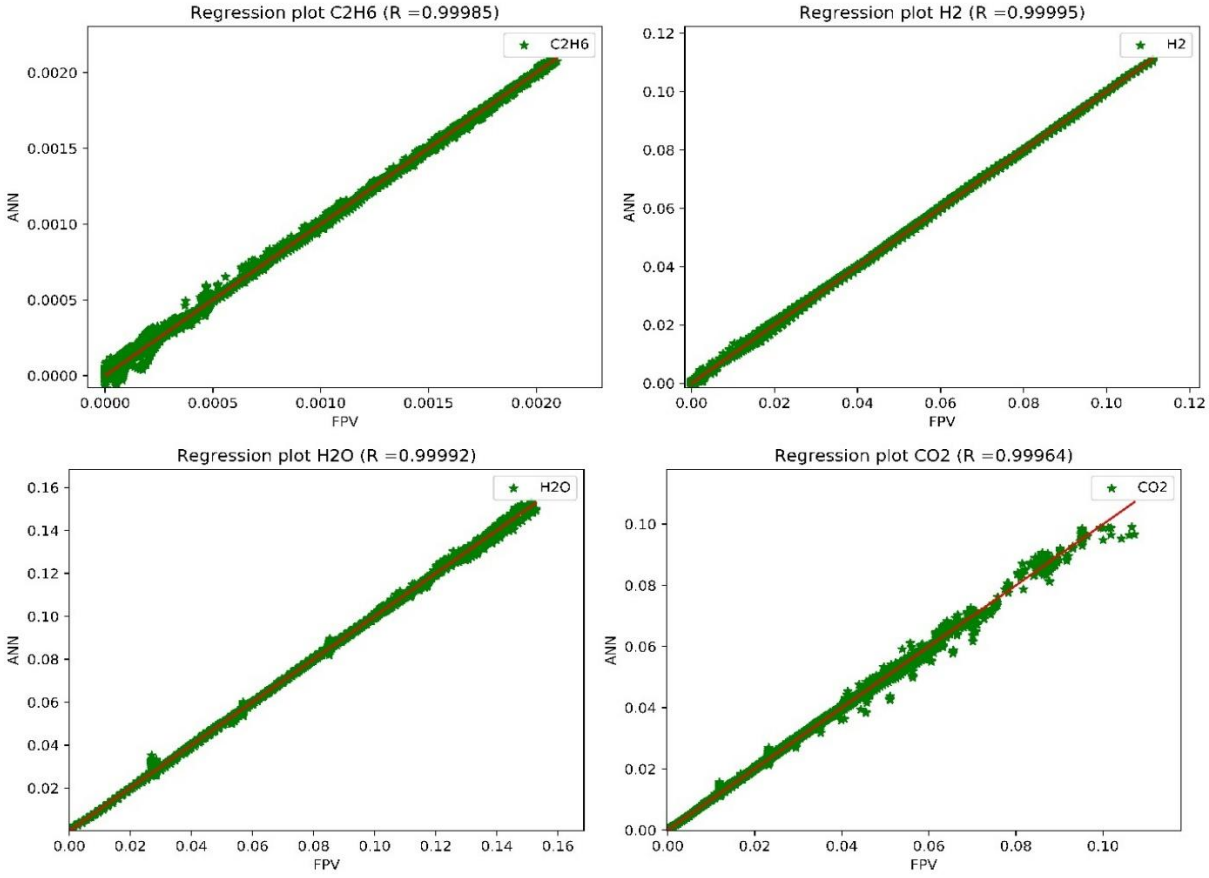
Thus, from the above analysis, we can conclude that Model 9 is the best model among all the options.

We can use model 9 for our purpose.

Model number	Model 9
Number of layers	7
Nodes in each layer	75-75-125-125-125-75-75
Activations functions used	Tanh and Rectilinear Linear Unit
Number of training epochs	10000
Optimizer	ADAM
Learning rate	0.001
Loss function	Mean Squared Error
Weight initialization	He Normal
Training dataset size	95% of Total data
Validation dataset size	2.5% of Total data
Testing dataset size	2.5% of Total data

Table 4.2: Specificationg of ANN Model 9

Now, we plot the linear regression graphs of major species like C_2H_6, H_2, H_2O, CO_2, and minor species CH_3O, HO_2, C_2H_2, and HCO. The Y-axis of the graph shows the ANN predicted data of the reactive scalar, whereas the X-axis shows the FPV interpolated data. It was observed that R for major species is around 0.99, whereas for minor species is around 0.98. The least value of R that we have for our ANN model is 0.98429. Thus, the worst modeled species by ANN is in 98.43% agreement with the FPV model.



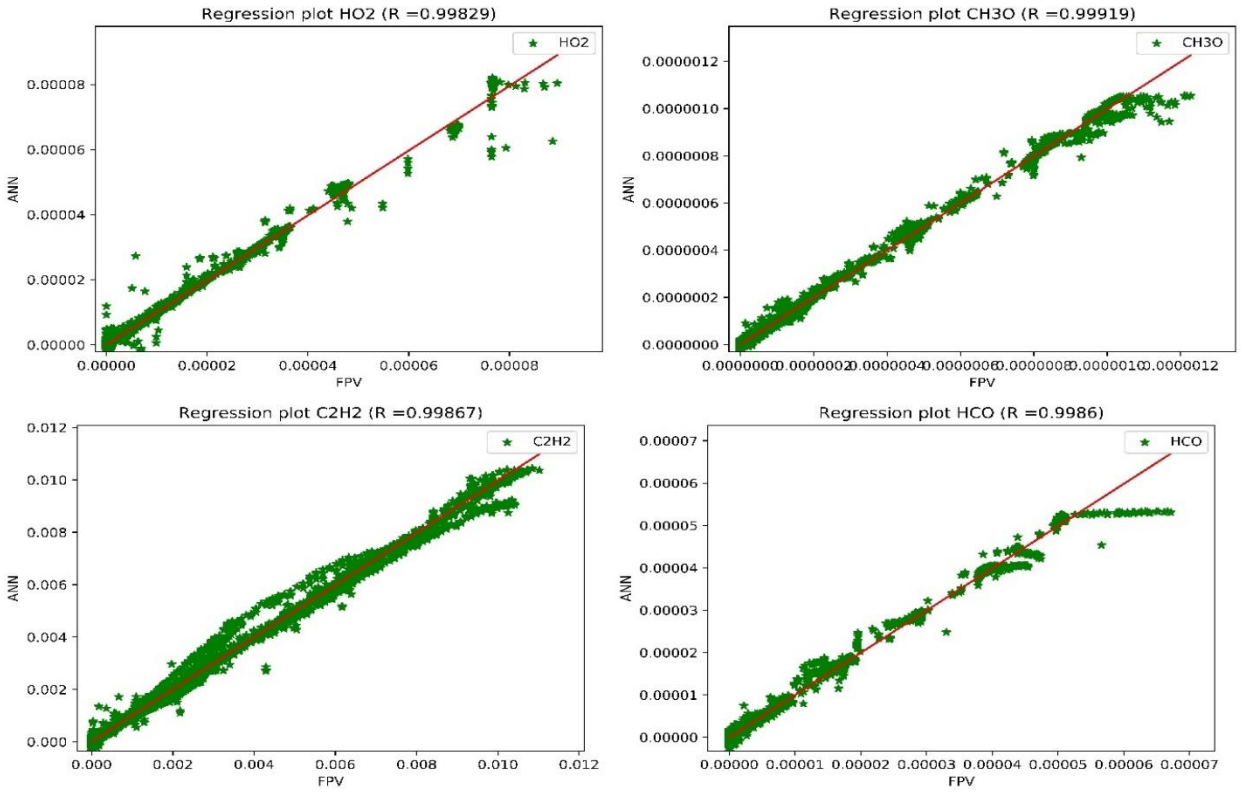


Figure 4.22: Comparison of predicted ANN data with FPV data for reactive scalars

Thus, the ANN model was trained on the new data that was generated later. In this, all the reactive scalars were a function of all the three variables, namely, Z, Zeta and Yc, instead of only two.

As compared to the previous ANN model, that was trained on the data where the reactive scalars were function of only two variables, our current model requires only 33% training epochs. Thus, the current model is 3x more efficient than the previous model. The following table summarises the difference between the previous ANN model and recent ANN model.

Model number	**Older Model**	**Model 9**
Number of layers	10	**7**
Nodes in each layer	150-150-150-150-150-150-150-150-150-150	**75-75-125-125-125-75-75**
Activations functions used	Tanh and ReLU	Tanh and ReLU
Number of training epochs	30000	**10000**
Optimizer	ADAM	ADAM
Learning rate	0.001	0.001
Loss function	Mean Squared Error	Mean Squared Error
Weight initialization	NONE	**He Normal**
Training dataset size	98% of Total data	**95% of Total data**
Validation dataset size	1% of Total data	**2.5% of Total data**
Testing dataset size	1% of Total data	**2.5% of Total data**

Table 4.3: Comparison between previous and recent ANN model

The difference between the two models are highlighted. Thus, the new model is more efficient, faster, and provides better predictions.

The following table lists the enhancements in the new ANN model.

Number of layers	33% Lesser layers
Total Nodes	55% lesser nodes
Number of training epochs	66% lesser epochs
Storage space as compared to FPV model	99.9981% lesser storage space required
Storage space as compared to previous model	0.23% lesser storage space required

Table 4.4: Enhancements in recent ANN model

Chapter 5: Conclusion and Future Scope

5.1 Conclusion

In this study, a coupled OpenFOAM-FPV turbulence combustion model has been used to model a turbulent bluff-body flame. It was observed that we were generating the reactive scalars that were function of only two variables, Z and Zeta, but while interpolation, we were interpolating those reactive scalars with respect to Z, Zeta and Yc. It was concluded that there were some issues with the reactive scalar depending only on two variables. The original approach of using the reactive scalars that depended on only two variables was changed and the scalars were generated as a function of all the three variables, and later this data was used for the training of our ANN model. Various ANN models were developed and their performance was studied. The best model from the available options was selected. It was observed that the new model's performance increased significantly, such that it required **66% lesser number of training epochs**, 3 lesser neuron layers which is **30% lesser layers** and only 675 nodes in total, i.e. **55% lesser number of nodes** as compared to the earlier model. This is a great enhancement in the model. Part of the credit goes to the weight initialization techniques that were implemented. Thus, usage of ANN model instead of FPV model has reduced the computation time as well as the storage space required. The trained model requires 281 KB of space, that is **99.9981%** reduction in memory space required as compared to the FPV model. The current model requires **0.23%** lesser space as compared to the previous ANN model.

5.2 Future Scope

The work done in the project may be extended towards the following direction:

1. Coupling of LES and FPV model may be attempted, as the coupling process of RANS and FPV model has been successfully developed, a similar approach may be extended for LES studies.
2. The validated ANN model can be coupled with OpenFOAM solver to obtain the mean reactive scalars.
3. The ANN model can be further enhanced by employing hyper parameter tuning, k-fold cross- validation procedure.
4. A different class of neural networks can be explored for modeling turbulent flames.

References

[1] Peters, N. (1984): Laminar diffusion flamelets in non-premixed turbulent combustion. Progress in Energy and Combustion Science, 3, 319–339 .

[2] Klimenko, A. Y., Bilger, R. W. (1999). Conditional moment closure for turbulent combustion, Progress in Energy and Combustion Science, 25, 595–687

[3] Pope, S. (1985). PDF methods for turbulent reacting flows, Progress in Energy and Combustion Science, 11, 119-192

[4] 4. Zhao, X., Haworth, D. C., Huckaby, D. (2012). Transported PDF modelling of nonpremixed turbulent CO/H2/N2 jet flames, Combustion Science and Technology, 184, 676-693.

[5] Kim, G., Kang, S., Kim, Y., Bilger, R.W., Cleary, M.J. (2007). Conditional moment closure and transient flamelet modelling for detailed structure and NOx formation characteristics of turbulent nonpremixed jet and recirculating flames, Combustion Theory and Modelling, 11, 527-552.

[6] Kim, S.H., Huh, K.Y., Tao, L. (2000). Application of the elliptical conditional moment closure model to a two-dimensional nonpremixed methanol bluff-body flame, Combustion and Flame, 120, 75-90

[7] Roy, R. N., Sreedhara. S. (2016). Modelling of methanol and H2/CO bluff-body flames using RANS based turbulence models with conditional moment closure model, Applied Thermal Engineering, 93, 561-570

[8] 8. Roy, R. N., Kumar, S., Sreedhara, S. (2014). A new approach to model turbulent lifted CH4/air flame issuing in a vitiated coflow using conditional moment closure coupled with an extinction model, Combustion and Flame, 161, 197-209.

[9] Roy, R. N., Kumar, S., Sreedhara, S. (2015). Predictions of lift-off of turbulent methane and propane flames issuing in cold surroundings using conditional moment closure coupled with an extinction model, Combustion and Flame, 162, 1164-1166

[10] Pierce, D. and Moin, P. (2004): Progress-variable approach for large-eddy simulation of non-premixed turbulent combustion, Fluid Mech. 504, 73–97.

[11] Ihme, M., Cha, C. M., Pitsch, H. (2005). Prediction of local extinction and re-ignition effects in non-premixed turbulent combustion using a flamelet/progress variable approach, Proceedings of the Combustion Institute, 30, 793-800

[12] Ihme, M., Pitsch, H., (2008). Modeling of radiation and nitric oxide formation in turbulent nonpremixed flames using a flamelet/progress variable formulation, Physics of Fluids, 20, 055110.

[13] Ihme, M., See, Y. C., (2010). Prediction of autoignition in a lifted methane/air flame using an unsteady flamelet/progress variable model, Combustion and Flame, 157, 1850-1862.

[14] Renzo, M. D., Coclitea, A., Tullioa, M. D. D., Palmaa, P. D., Pascazioa, G. (2015). LES of the Sandia flame D using an FPV combustion model, Energy Procedia, 82, 402-409.

[15] http://openfoam.org

[16] Pope, S. B., (2000). Turbulent Flows. Cambridge University Press.

[17] David, C., (2006) Turbulence modeling for CFD, Wilcox.

[18] Launder, B., Spalding, D., (1972). Lectures in Mathematical Models of Turbulence, New York Academic Press.

[19] http://combustion.berkeley.edu/gri-mech/new21/version12/text12.html

[20] Zhang, Y., Xu, S., Zhong, S., Bai, X. S., Wang, H., Yao, M., (2020) Large eddy simulation of spray combustion using flamelet generated manifolds combined with artificial neural networks, Energy and AI, 2,100021

[21] Pollard A, Castillo L, Danaila L, Glauser M. 2016. Whither Turbulence and Big Data in the 21st Century? Cham., Switz.: Springer

[22] Perlman E, Burns R, Li Y, Meneveau C. 2007. Data exploration of turbulence simulations using a database cluster. In Proceedings of the 2007 ACM/IEEE Conference on Supercomputing, p. 23. New York: ACM

[23] Wu X, Moin P. 2008. A direct numerical simulation study on the mean velocity characteristics in turbulent pipe flow. J. Fluid Mech. 608:81–112

[24] Cherkassky V, Mulier FM. 2007. Learning from Data: Concepts, Theory, and Methods. Hoboken, NJ: John Wiley & Sons

[25] Meijering E. 2002. A chronology of interpolation: from ancient astronomy to modern signal and image pro cessing. Proc. IEEE 90:319–42

[26] Hastie T, Tibshirani R, Friedman J, Hastie T, Friedman J, Tibshirani R. 2009. The Elements of Statistical Learning, Vol. 2. New York: Springer

[27] Brunton SL, Kutz JN. 2019. Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control. New York: Cambridge Univ. Press

[28] Hornik K, Stinchcombe M, White H. 1989. Multilayer feedforward networks are universal approximators. Neural Netw. 2:359–66

[29] Rumelhart DE, Hinton GE, Williams RJ. 1986. Learning representations by back-propagating errors. Nature 323:533–36

[30] Jambunathan K, Hartle S, Ashforth-Frost S, Fontama V. 1996. Evaluating convective heat transfer coefficients using neural networks. Int. J. Heat Mass Transf. 39:2329–32

[31] Pierret S, Van den Braembussche R. 1999. Turbomachinery blade design using a Navier-Stokes solver and artificial neural network. J. Turbomach. 121:326–32

[32] Milano M, Koumoutsakos P. 2002. Neural network modeling for near wall turbulent flow. J. Comput. Phys. 182:1–26

[33] Faller WE, Schreck SJ. 1996. Neural networks: applications and opportunities in aeronautics. Prog. Aerosp. Sci. 32:433–56

[34] Goodfellow I, Bengio Y, Courville A. 2016. Deep Learning. Cambridge, MA: MIT Press