ME 399
Project Course Report

Supervisor: Prof. Dilip S. Sundaram
Mechanical Engineering

Problem Statement

~ We are trying to develop a hybrid Machine learning based solver to solve 0D combustion chemistry problems.

~ We train a Multi-layer perceptron(MLP) model on H2/Air combustion 0D simulation data. Using this trained model we successively predict the states of the 0D system(i.e T, P, X_k).

~ As per Ref. [1] ML based solvers implemented using MLP/ GPAR(Gaussian Process Autoregressive Regression) have the potential to hugely improve the simulation performance.

Comparative performance evaluation of the models.

ML model	Split set	R^2	MAE	Relative speed-up (t_{0D}/t_{ML})
GPR	Train	0.998	1.61×10^{-8}	1.9
	Validation	0.998	1.62×10^{-8}	
	Test	0.997	2.12×10^{-8}	
GPAR	Train	0.999	1.39×10^{-8}	2.1
	Validation	0.999	1.41×10^{-8}	
	Test	0.998	1.53×10^{-8}	
ANN	Train	0.993	6.24×10^{-8}	3.0
	Validation	0.992	9.02×10^{-8}	
	Test	0.988	2.80×10^{-7}	

Note: R^2 is a commonly used performance metric. A model with R^2=1.0 is a perfect model.

Literature Review

January

~ I read about the application of Machine learning in predicting fluid flows and the maths behind it.

~ Since fluid flows have a lot of redundant data associated with it, it becomes imperative to reduce the dataset.

This reduction of dataset is done using techniques such as **PCA**(Principal Component Analysis), **POD**(Proper Orthogonal Decomposition) and Dynamic Mode decomposition(**DMD**).

~ I took a course on Physics Informed Machine Learning. This course taught me about curating the right data; designing an architecture; crafting a loss function; deploying the optimisation process.

~ I learnt about **Physics Informed Neural Networks(PINNs)**. PINNs is a technique through which we can not only train our ML model based on the existing data but also on the underlying physics governing equations. A detailed code implementation of PINNs can be accessed here.

~ I read about **combustion chemistry** from the book by *Stephen R Turns*.

In the book i read on topics like Reaction mechanism, Elementary reaction rates, 0D reactor models, etc.

OpenFoam Programming

February

~ I started by understanding about the basic implementation of simulations in openfoam.

I read the first 30 pages of the official openfoam UserGuide where i got accustomed to the general syntax rules, file structure of simulation cases, etc.

Then I implemented my first simulation case i.e Lid driven Cavity flow by reading the official openfoam Tutorial Guide. A detailed report of the implementation of Lid Driven Cavity flow can be accessed here.

~ I took a course on basic openfoam programming wherein we were taught about the fundamental openfoam data structures, developing custom solvers, etc.

During the course I simulated 1D heat diffusion case after developing a custom LaplacianFoam solver. A detailed implementation report of the simulation can be accessed here.

Chemfoam Solver March

~The chemFoam solver in OpenFOAM is designed for simulating chemical kinetics in zero-dimensional (0D) reactors. It employs a segregated approach, solving individual governing equations for species mass fractions, energy, and pressure with explicit coupling through reaction source terms.

~ Since the computational domain consists only of one cell, the chemical reactions are the only mechanism influencing the evolution of the species concentration and the temperature.

~ Solution scheme of the chemfoam solver

1. Species conservation

$$\frac{dY_k}{dt} = \frac{\dot{\omega}_k}{\rho}$$

- Reaction rates w_i computed via. chemFoam/solveChemistry.H with Arrhenius kinetics using the latest species mass fractions Yi.
- The chemFoam/Yeqn.H file updates the species mass fractions Y_i using the reaction rates computed in the previous step.

2. Energy Equation

$$\frac{dT}{dt} = \frac{1}{C_{p.\,mix}} \frac{q}{\rho}$$

The above equation updates temperature T via enthalpy-based formulation of q.

After temperature updation, the specific mass enthalpies are updated using the chemFoam/hEqn.H file.

3. Thermodynamic Updates:

The below equation updates mean density of the mixture using the ideal gas law as:

$$\rho = \frac{p\,MW}{R\,T}$$

~ Key Limitations of the chemfoam solver:

1. Explicit Coupling: Reaction terms lag by one-time step, limiting accuracy for stiff mechanisms.
2. Time-Step Sensitivity: Requires small Δt for ignition/detonation cases

H2/Air OD constant-pressure combustion simulation

~ The 0D hydrogen-air combustion simulation models the chemical kinetics of hydrogen oxidation in a homogeneous reactor under constant pressure. This setup isolates the effects of reaction chemistry without the influence of fluid dynamics, making it ideal for studying ignition delays, reaction pathways, and temperature evolution.

~ The hydrogen-air mixture reacts via detailed chemical mechanisms (e.g., GRI-Mech which is defined in the chemfoam/chem.inp file).

~ We measure the concentrations of H2, O2, H2O, H2O2, H, O, OH, Ar, HO2, N2 ,Temperature T, pressure P and density ϱ of the 0D combustor.

~ A detailed openfoam implementation of H2/Air combustion test case can be accessed here.

~ We get the simulation data in the test_case/chemfoam.out file as:

t	T	Qdot	P	H2	H	O2	O	OH	HO2	H2O2	H2O	AR	N2
1.000000e-07	1000.0	-824.946	202650	0.004231	2.100800e-13	0.004231	2.290120e-15	2.021970e-15	2.198430e-13	1.401150e-18	4.112770e-16	0	0.01591
3.000000e-07	1000.0	-789.345	202650	0.004231	6.033260e-13	0.004231	1.880080e-14	1.359050e-14	6.820920e-13	1.290390e-17	8.812880e-15	0	0.01591
7.000000e-07	1000.0	-717.125	202650	0.004231	1.324740e-12	0.004231	8.685340e-14	4.826390e-14	1.689940e-12	7.327670e-17	7.892900e-14	0	0.01591
1.500000e-06	1000.0	-575.328	202650	0.004231	2.654300e-12	0.004231	3.030020e-13	1.314200e-13	4.011170e-12	3.623060e-16	4.914500e-13	0	0.01591
3.100000e-06	1000.0	-292.445	202650	0.004231	5.269190e-12	0.004231	8.624610e-13	3.189510e-13	9.818610e-12	1.754250e-15	2.564460e-12	0	0.01591

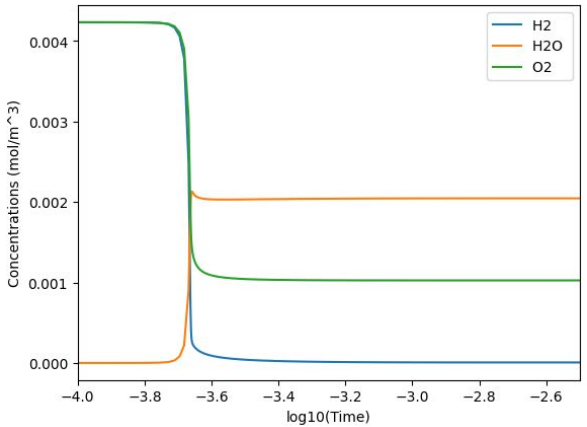
Simulation Results

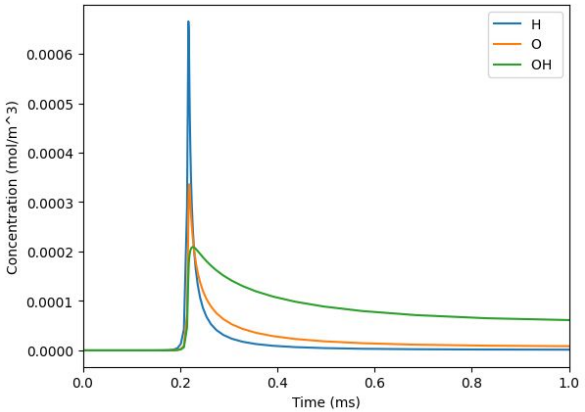
~ I ran the H2/Air combustion simulations with different initial Temperature and Pressures.

I generated **a large dataset** to train the MLP model. I ran the simulation for every binary combination of Initial Temp. T = 700, 750, 800, 850, 900, 950, 1000, 1050, 1100, 1150, 1200, 1250, 1300, 1350, 1400, 1450, 1500 Kelvin and constant Pressure P = 0.1, 0.2, 0.5, 1, 2, 5 atm.

~ **For Initial T = 1000 K and P = 2 atm simulation,** I got the following results

The Ignition delay time
(i.e time at which Qdot is max.)
was found to be 0.218ms.





Problem Statement Revisited…

~ In combustion chemistry simulation of 1D 2D 3D flows we have to solve the CFD part as well as the combustion kinetics part. The CFD time scale is much larger than the chemical time scale because combustion kinetics can be a very fast.

~ Since the t_{chem} is very small, we do not run the simulation at this time scale!! Basically we run the simulation at t_{CFD} time scale and we do an integration of the chemical kinetics equations to get the concentrations of species at the next t_{CFD} time step.

~ The bottleneck of the whole approach is the calculation of chemical kinetics, which requires the integration of a stiff system of ODEs. This chemical integration takes about 90-95% of the total simulation time.

~ **Nikitin et. al.** wrote 2 papers(Ref [5] and [6]) to solve this problem for H2/air combustion case.

~ To speed up numerical simulations, the paper presents a solution of the chemical kinetics problem using artificial neural network approach. Using the architecture of a multilayer neural network with bypass connections, namely residual network, it is possible to obtain a fast and reliable solution to the problem.

~ The neural network is trained to predict the state of the system only one t_{chem} time step ahead. Using it in a recursive mode, it is possible to forecast for thousands of steps(i.e t_{CFD}) without loss of accuracy.

~ Let us now talk about the papers in detail..

A neural network approach to solve the problem

April

~ To **train the network,** a large data set is required, so a numerical solution of the system was processed for a large set of initial data. The numerical implementation is based on the Novikov method from the class of Rosenbrock methods for stiff systems of ordinary differential equations;

~ We normalize the dataset as:

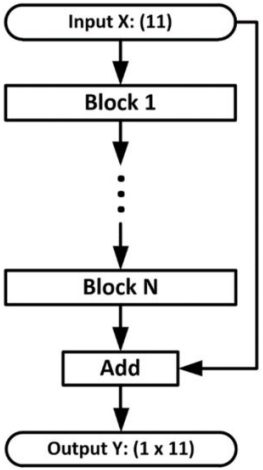
$$\begin{cases} \widetilde{T} = \dfrac{T - \langle T \rangle}{\sigma_T} \\ \widetilde{X} = \dfrac{X^* - \langle X^* \rangle}{\sigma_{X^*}}, \; X^* = \ln\left(1 + \dfrac{X}{\varepsilon}\right), \; \varepsilon = 10^{-10} \end{cases},$$

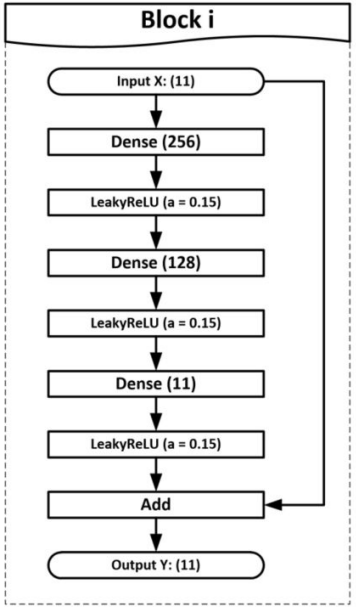
Molar densities were additionally made logarithmic. Means and variances were calculated over the entire training sample.

~ We initialize a MLP architecture such that we input the

initial state(11 variables i.e 10 X_k and T) and

get the output state(i.e the predicted output state at t_{chem}= 1e-5).





~ Thus, we have a neural multi-layer feedforward network with detours. The number of blocks can be changed to achieve greater accuracy and reliability or speed of operation. The computational complexity of a block is about 74 000 flop operations.

~ We try to generate simulation data for about 4,00,000 time steps. We according divide this dataset into training, validation and testing dataset in the ratio of 0.8 : 0.1 : 0.1

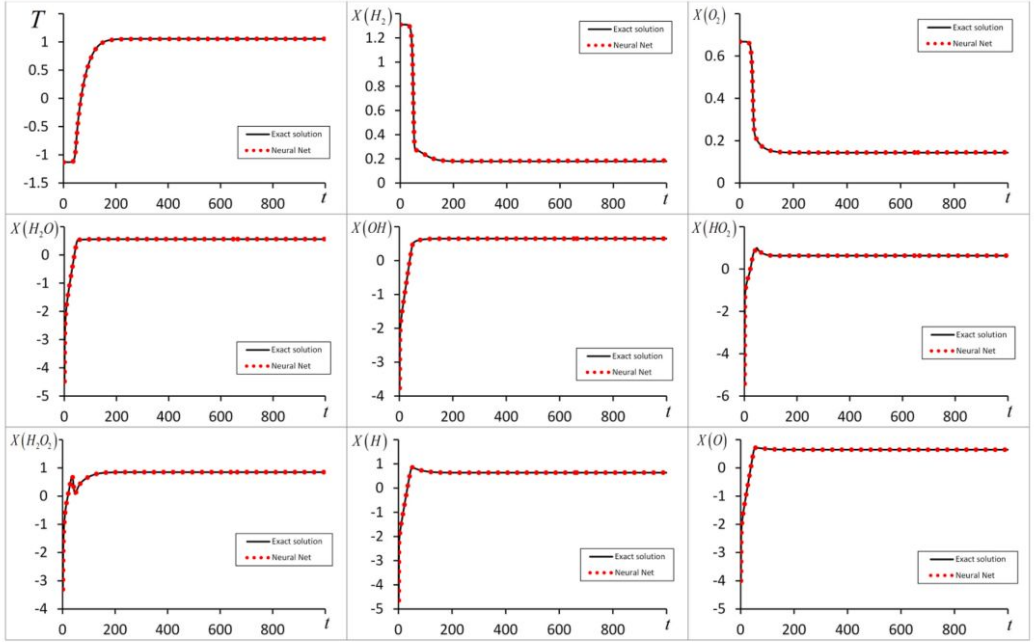
~ **There is a problem of accumulating errors:**

At every step, the network produces a result with some deviation, or an inaccurate result. No matter how small the error is, at the next step a distorted input is applied to the network input. This means that the output of the network with each cycle will deviate more and more from the ideal result.

~ **Solution to the problem:**

The increase in the number of blocks in the network and the residual connection allow us to partially combat this problem and get a high-quality result for about 1000 steps.

~ This is the expected results:



. Results of processing of the neural network with 9 blocks for temperature and molar concentration of chemical component for first 1000 time steps.

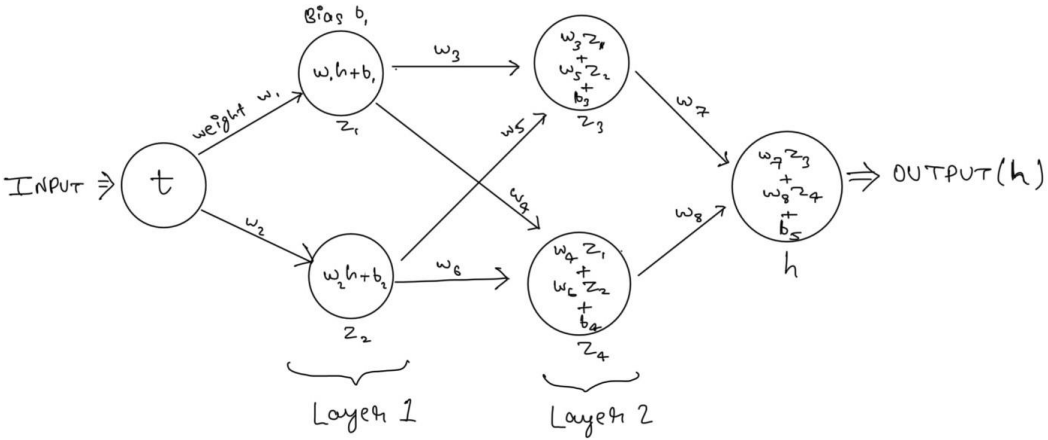
~ Hence, if some traditional solver is simulating H2/Air combustion through GRI 3.0 mechanism, then instead of numerical integration of chemical kinetics. People can use our solver as a base solver to quickly predict the chemical states upto $1000*t_{chem}$ step size.

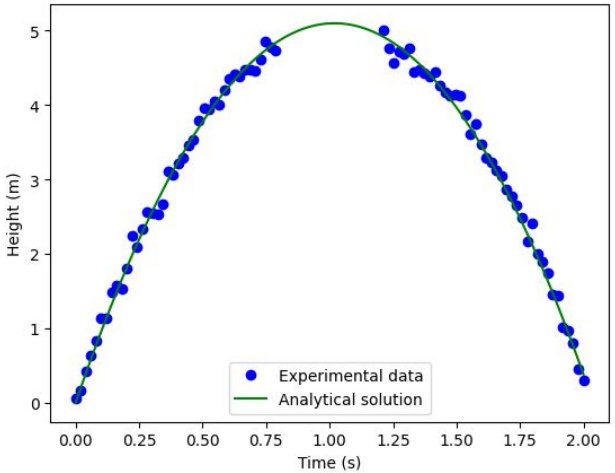
Introduction to Multi-layer perceptrons

~ To explain Multi-layer perceptrons(MLPs), let's take the example of predicting the height of a ball thrown up into the air.

~ We experimentally measured height h as a function of time t. But note that some data points are missing!! These missing data points are the ones where we want to predict the height.

~ We take a simple MLP model having 2 hidden layers with 2 neuron cells each.





~ We use the experimental height dataset to train our model.

~ We define a objective/loss function as

$$\text{Loss} = \text{MSE}(\text{Predictions}, \text{Training})$$

~ After defining the loss function, we now train our model i.e we try to modify the model weights and biases in such a way so that the loss function is minimised. This modification of weights is done using Gradient descent

$$W_{new} = W_{old} - \epsilon * (\partial \text{Loss}/\partial W)$$

$$b_{new} = b_{old} - \epsilon * (\partial \text{Loss}/\partial b)$$

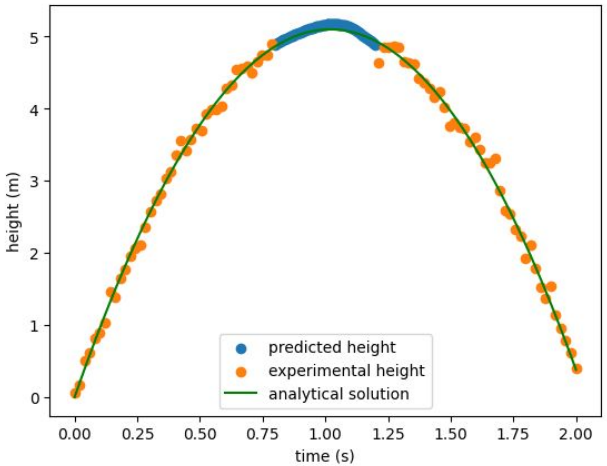
here ϵ = step-size parameter

~ We repeatedly modify every weight and bias until the loss converges to a small value.

~ Now that the model is trained, we predict for height!!

~ The code for this example can be accessed using the link at Ref [2]

~ A better Introduction to Neural networks can be accessed using the link at Ref [3]



References

[1] Üstün, C. E., & Paykani, A. (2024). Probabilistic machine learning framework for chemical source term integration with Gaussian Processes: H2/air auto-ignition case. *International Journal of Hydrogen Energy*. https://www.sciencedirect.com/science/article/pii/S0360319924028945#bb24

[2] Jupyter Notebook implementing a simple MLP network through an example https://github.com/adityaprasad2005/Hybrid_Chemfoam_solver/blob/main/intro_to_mlps.ipynb

[3] Blog on MLP: https://blog.marketmuse.com/glossary/multilayer-percpetron-definition/

[4] ChemFoam Solver Documentation: https://openfoamwiki.net/index.php/ChemFoam

[5] Nikitin, V.F. (2021). Neural network approach to solve gas dynamics problems with chemical transformations. Computers & Fluids, 218, 104873. https://www.sciencedirect.com/science/article/pii/S0094576520307347

[6] Nikitin, V. F. (2021). Approach to combustion calculation using neural network. Computers & Chemical Engineering, 151, 107361. https://www.sciencedirect.com/science/article/pii/S0094576521005750

[7] Github Repository of all the codes: https://github.com/adityaprasad2005/Hybrid_Chemfoam_solver