

## Dynamic Modeling Of Mini SR-30 Gas Turbine Engine

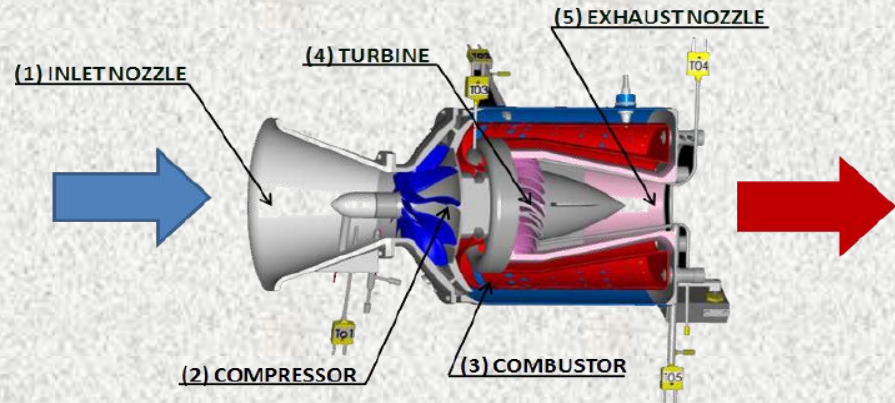


Photo Courtesy Turbine Technologies LTD

Presented By:  
Prof. P. S. V. Nataraj

# Outline

---



- **Quick glance at deep learning**
- **Introduction to gas turbine engine**
- **First principle based modeling**
- **Deep learning based modeling**
- **Results and validation**
- **Conclusion**

# Quick Glance at Deep Learning

---



## A brief Introduction:

- **1943** - Walter Pitts and Warren McCulloch, gave us that piece of the puzzle when they created the **first mathematical model** of a neural network.
- **1946** - John Mauchly & J. Presper Eckert develop world's first digital computer '**ENIAC**' .
- **1952** — Arthur Samuel writes the **first computer program** capable of learning.
- **1958** — Frank Rosenblatt designs the **Perceptron**, the first artificial neural network.

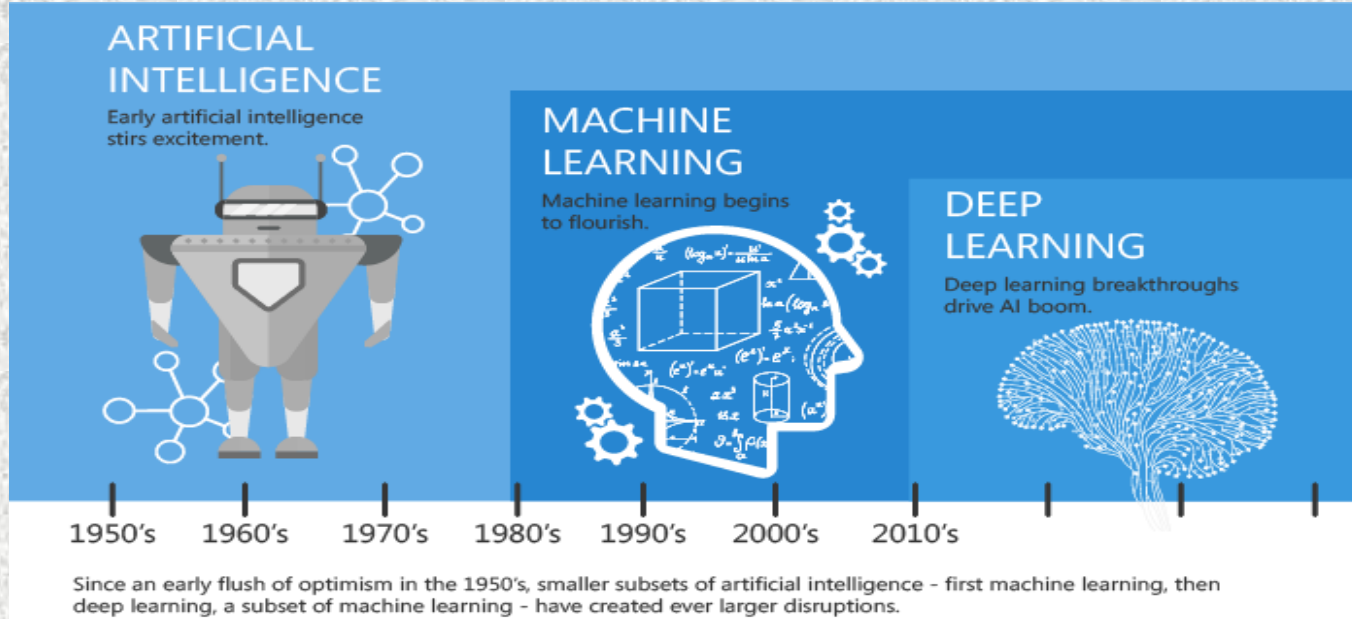
## What has fueled the development of deep learning?

1. Explosion of data.
2. Cheap computing cost – CPUs and GPUs.
3. Improvement of machine learning models.

# Quick Glance at Deep Learning



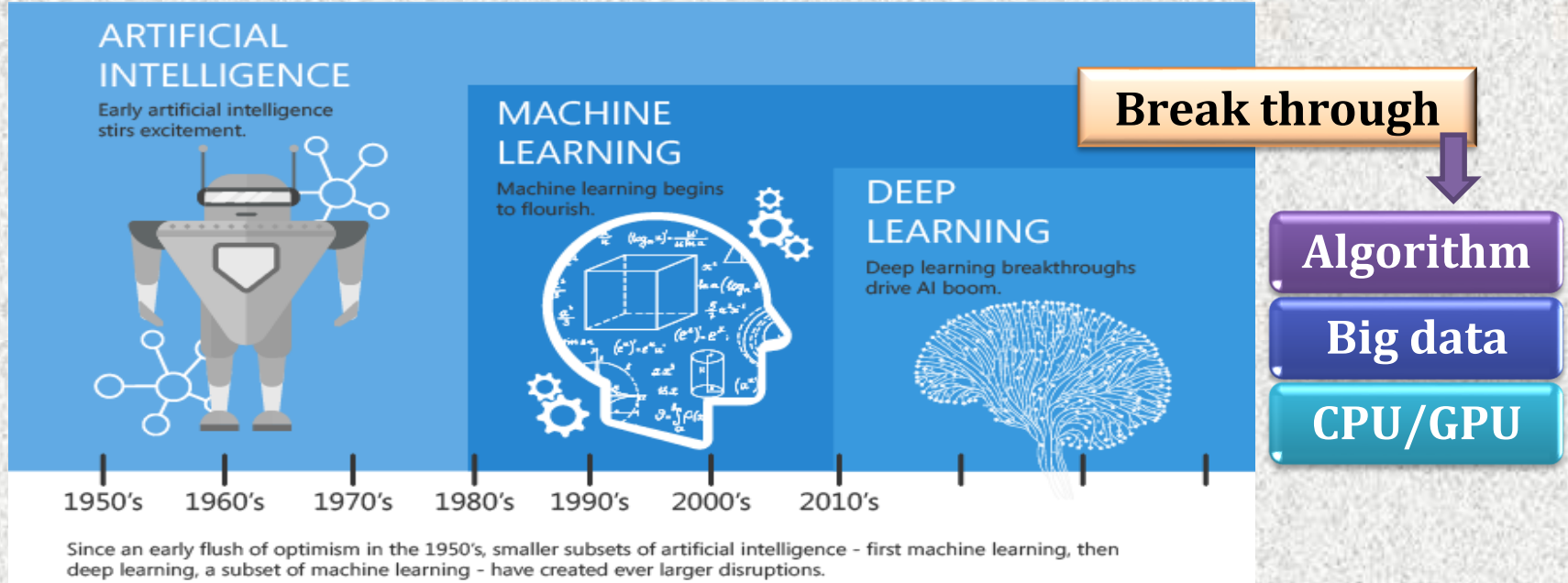
## Evolution of Deep Learning



# Quick Glance at Deep Learning



## Evolution of Deep Learning

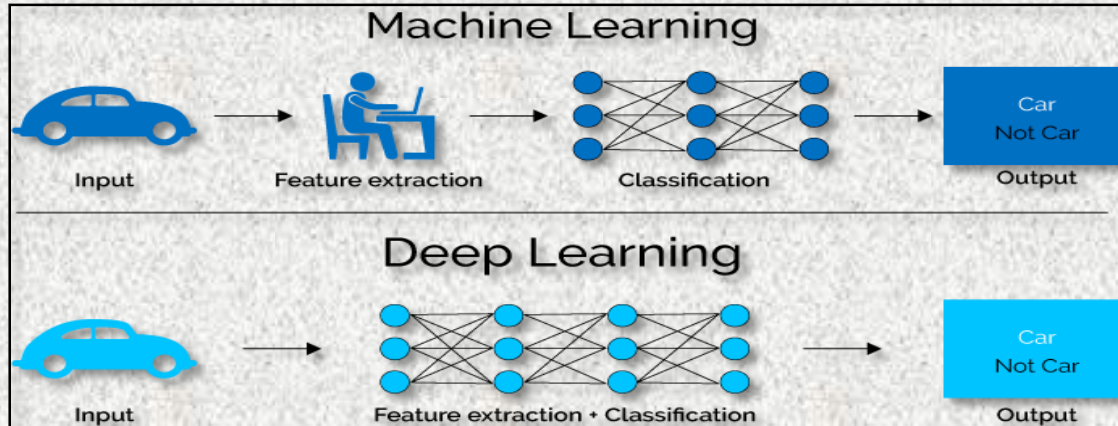


# Quick Glance at Deep Learning



## Deep Learning revolutionized Machine learning:

- Deep learning don't need to provide features ahead of time, it learns features at different levels by itself.
- Same deep learning architecture can be trained to accomplish different tasks.

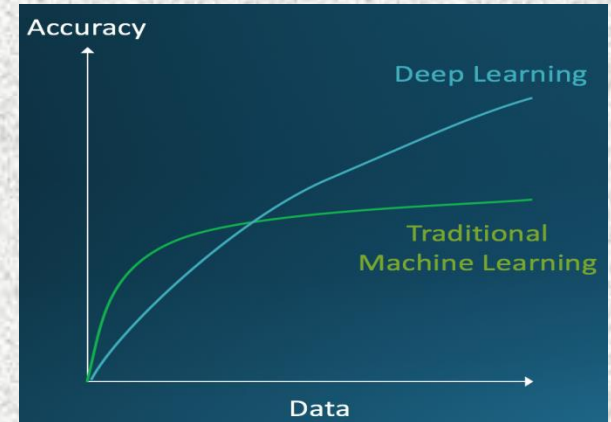
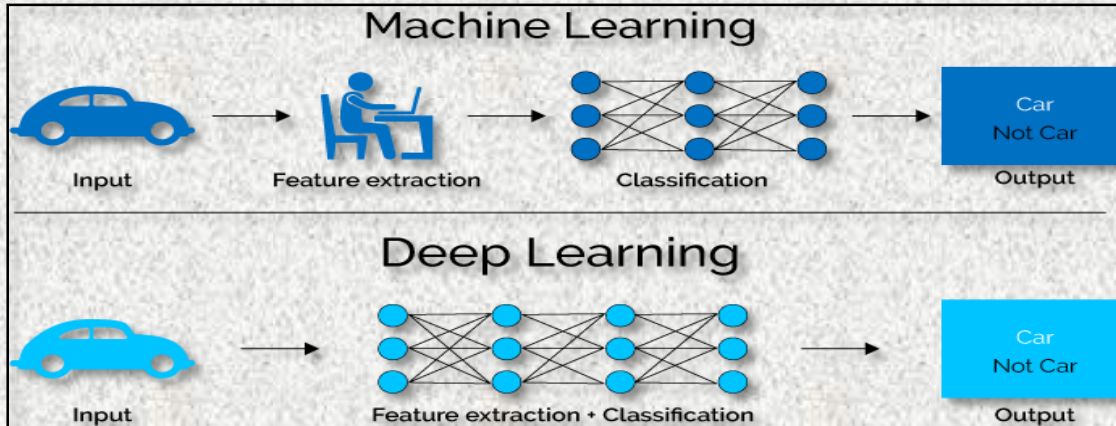


# Quick Glance at Deep Learning



## Deep Learning revolutionized Machine learning:

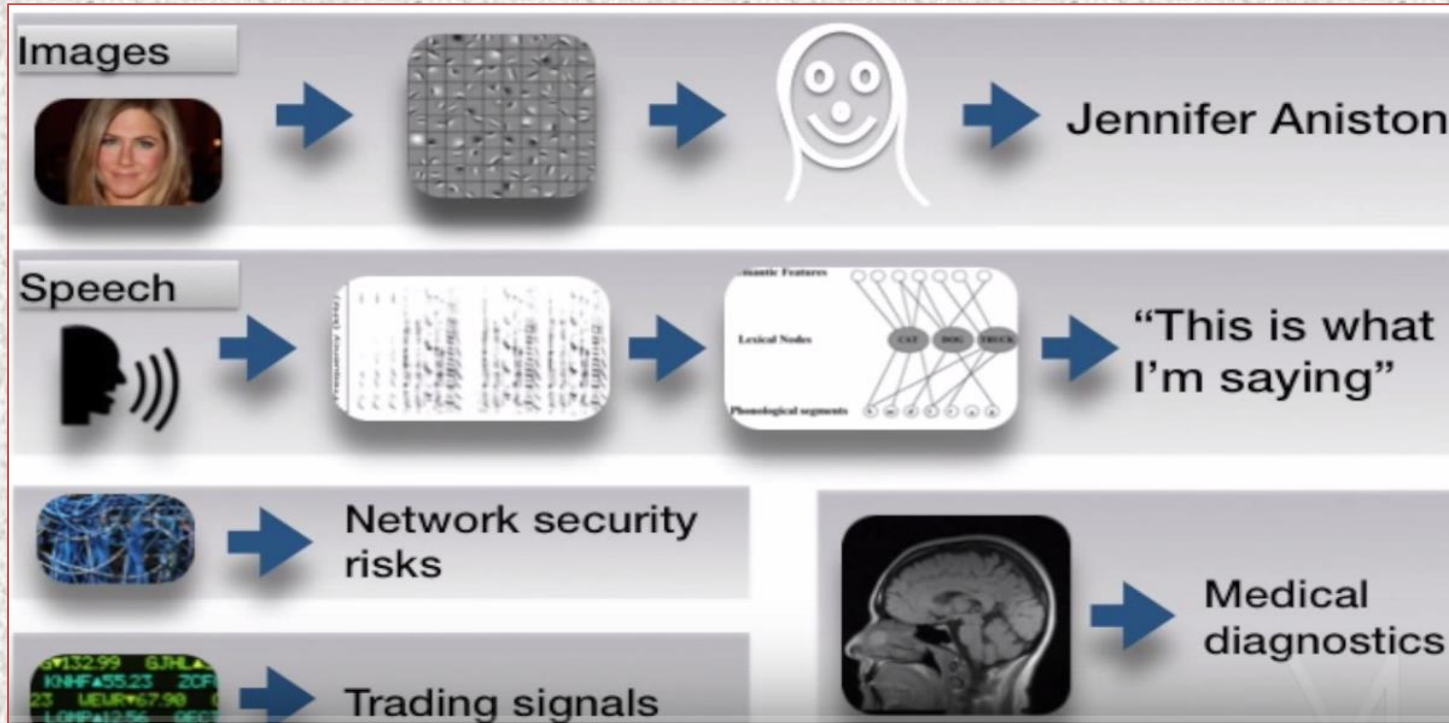
- Deep learning don't need to provide features ahead of time, it learns features at different levels by itself.
- Same deep learning architecture can be trained to accomplish different tasks.



# Quick Glance at Deep Learning



## Major area of research





# Quick Glance at Deep Learning



## Applications in various sector

- Predictive maintenance or condition monitoring
- Warranty reserve estimation
- Propensity to buy
- Demand forecasting
- Process optimization
- Telematics

### Manufacturing



- Predictive inventory planning
- Recommendation engines
- Upsell and cross-channel marketing
- Market segmentation and targeting
- Customer ROI and lifetime value

### Retail



- Alerts and diagnostics from real-time patient data
- Disease identification and risk stratification
- Patient triage optimization
- Proactive health management
- Healthcare provider sentiment analysis

### Healthcare and Life Sciences



- Aircraft scheduling
- Dynamic pricing
- Social media – consumer feedback and interaction analysis
- Customer complaint resolution
- Traffic patterns and congestion management

### Travel and Hospitality



- Risk analytics and regulation
- Customer Segmentation
- Cross-selling and up-selling
- Sales and marketing campaign management
- Credit worthiness evaluation

### Financial Services



- Power usage analytics
- Seismic data processing
- Carbon emissions and trading
- Customer-specific pricing
- Smart grid management
- Energy demand and supply optimization

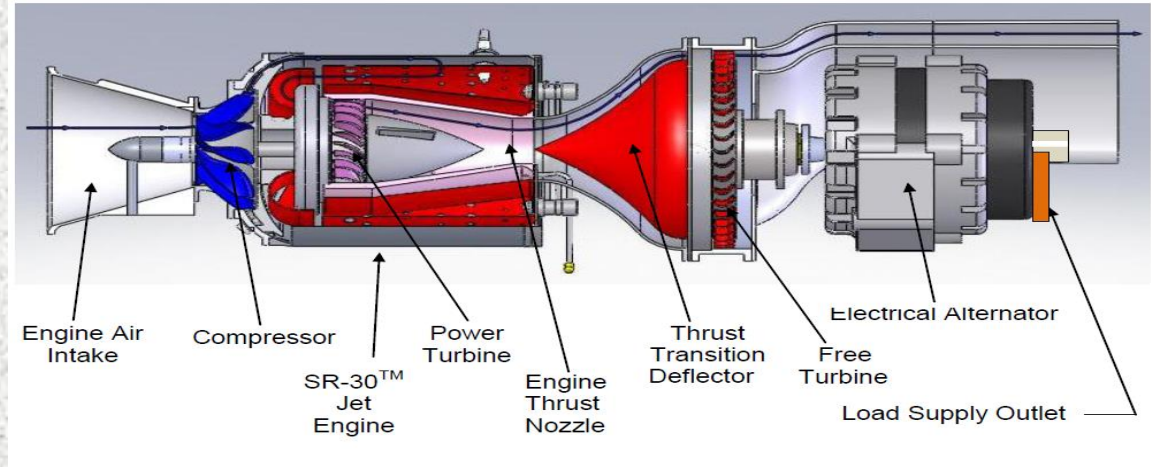
### Energy, Feedstock, and Utilities



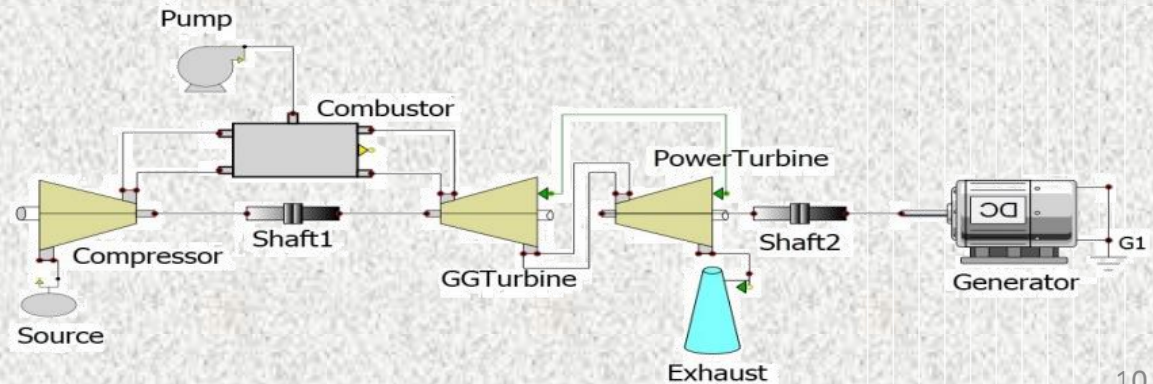
# SR-30 Gas Turbine Engine



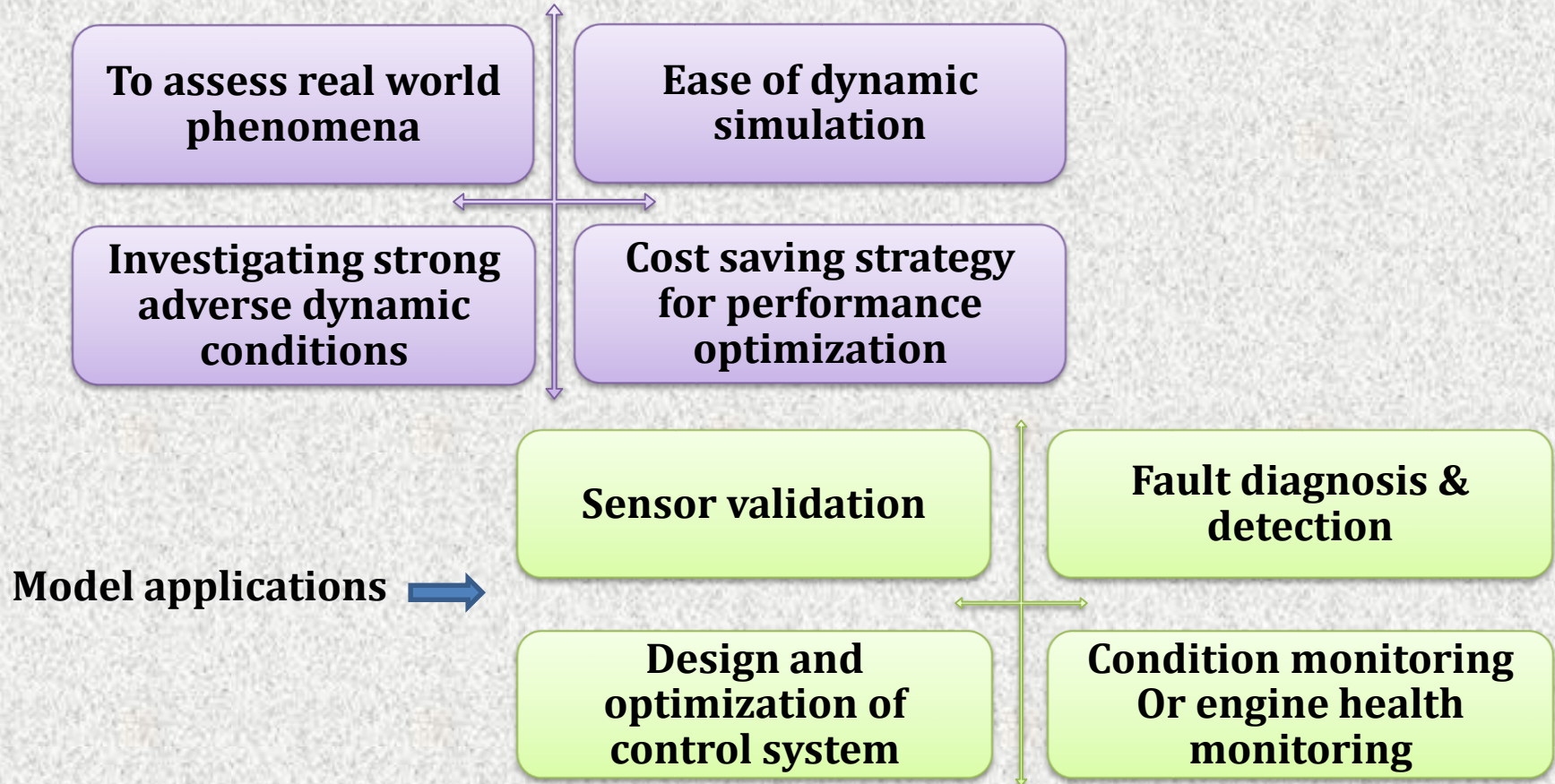
**Fig : Cross-sectional view of laboratory SR-30 engine**



**Fig : schematic flow diagram of laboratory engine**



# Why Engine Model is Necessary??



# Dynamic Model of SR-30 Engine

---



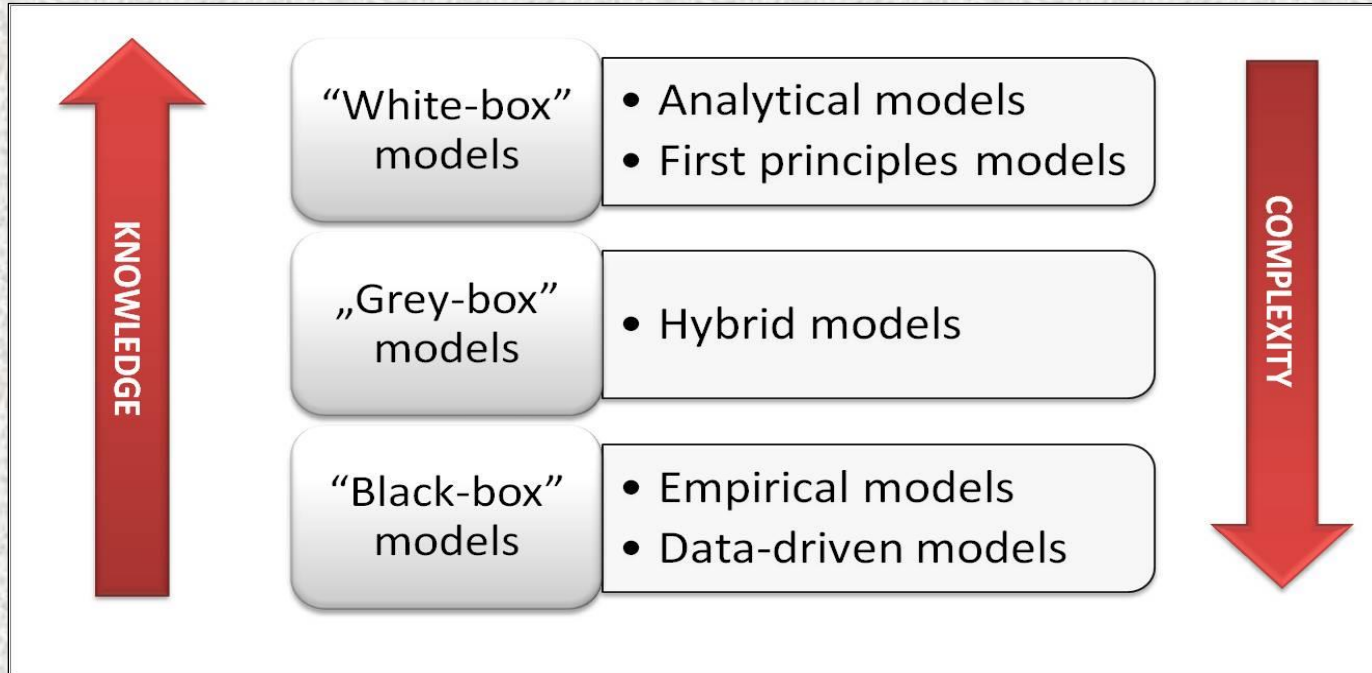
## Objective:

- Develop a non-linear dynamic engine model.
- Simulate the steady state and transient performance.
- Integrate the developed gas turbine model with multi-disciplinary systems.

## Challenges:

- Experimental data
- Characteristics map of engine components.
- Tuning of characteristics maps.
- Simulate the model over full operating range.

# Approaches for Modeling Dynamic Systems



# Problem Statement

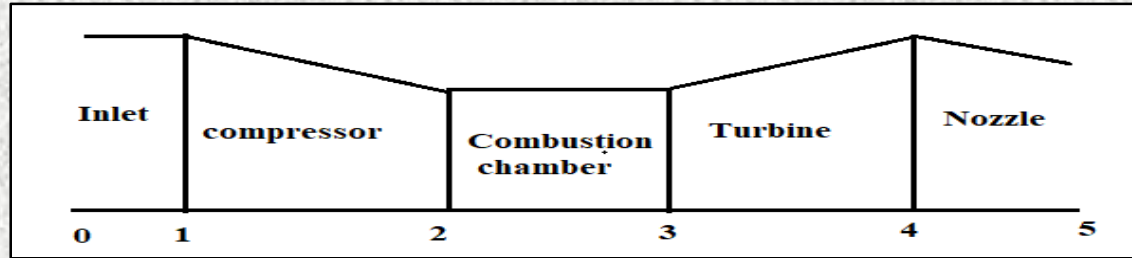
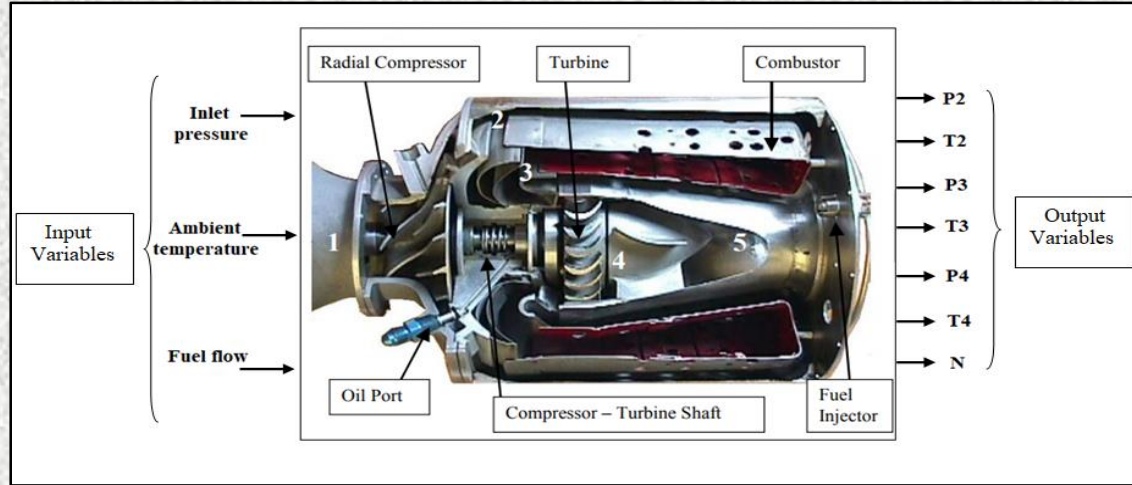


Fig : Illustration of input and output variables of the model

# First Principle Modeling Approach



## State variable method:

### 1 - Selection of state Variable

$$x = [P_2 \ P_4 \ N]$$

### 2 - Compressor calculation

$$\left(\frac{P_2}{P_1}, \frac{N}{\sqrt{T_1}}\right) \xrightarrow{\text{Compressor map}} (\dot{m}_c, \eta_c)$$

$$(T_1, \eta_c, PR_{21}) \xrightarrow{\text{isentropic equation}} (T_2)$$

$$W_c = \dot{m}_c c_p (T_2 - T_1)$$

### 3 - Combustion chamber

$$(P_2, \sigma_{cc}) = P_3$$

$$(LHV, \dot{m}_c, \dot{m}_f) \xrightarrow{\text{Energy balance}} T_3$$

### 4- Turbine calculation

$$\left(\frac{P_3}{P_4}, \frac{N}{\sqrt{T_3}}\right) \xrightarrow{\text{turbine map}} (\dot{m}_t, \eta_t)$$

$$(T_3, \eta_t, PR_{34}) \xrightarrow{\text{isentropic equation}} (T_4)$$

$$W_t = \dot{m}_t c_p (T_3 - T_4)$$

### 5- Nozzle equation

$$\left(\frac{P_4}{P_5}\right) \xrightarrow{\text{nozzle map}} (\dot{m}_t, \eta_t)$$

# First Principle Based Engine Simulator

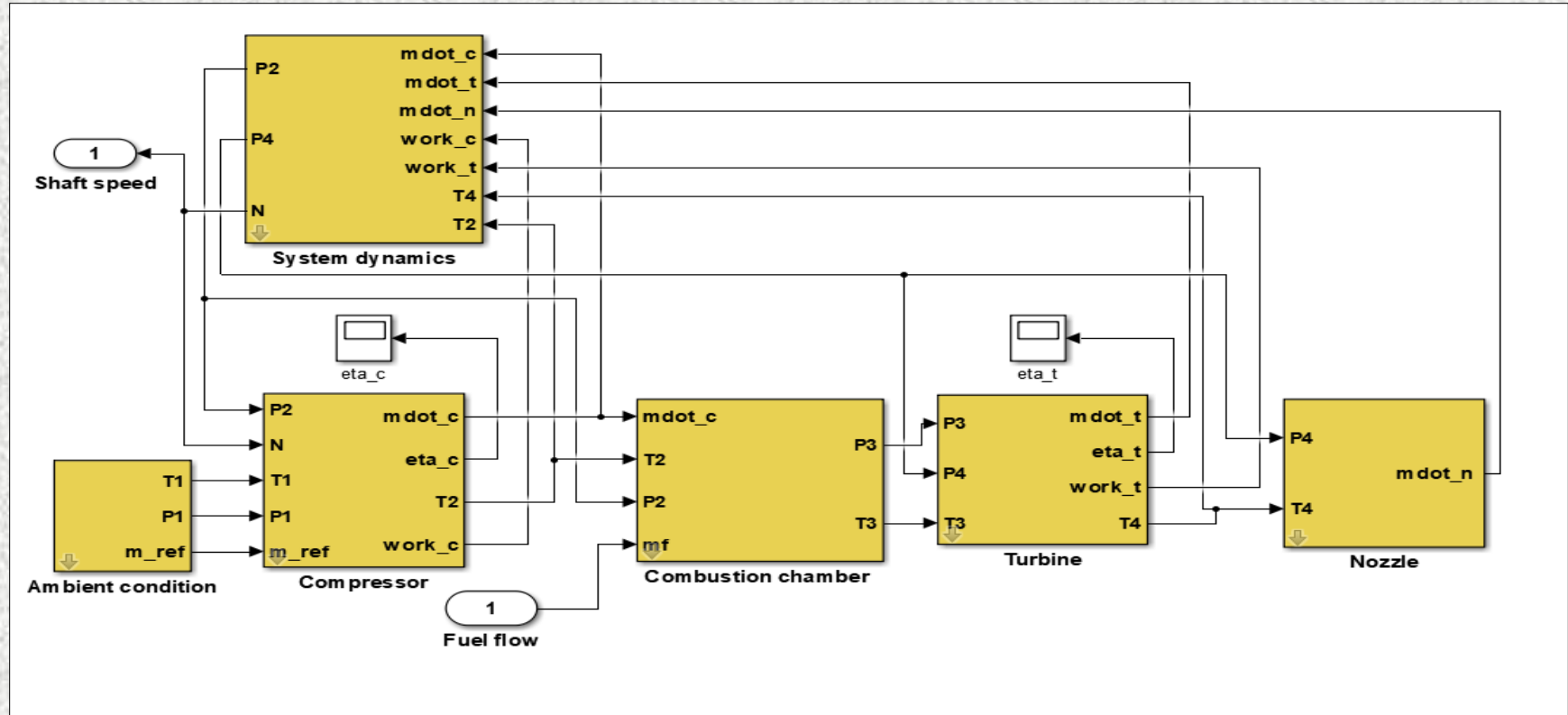
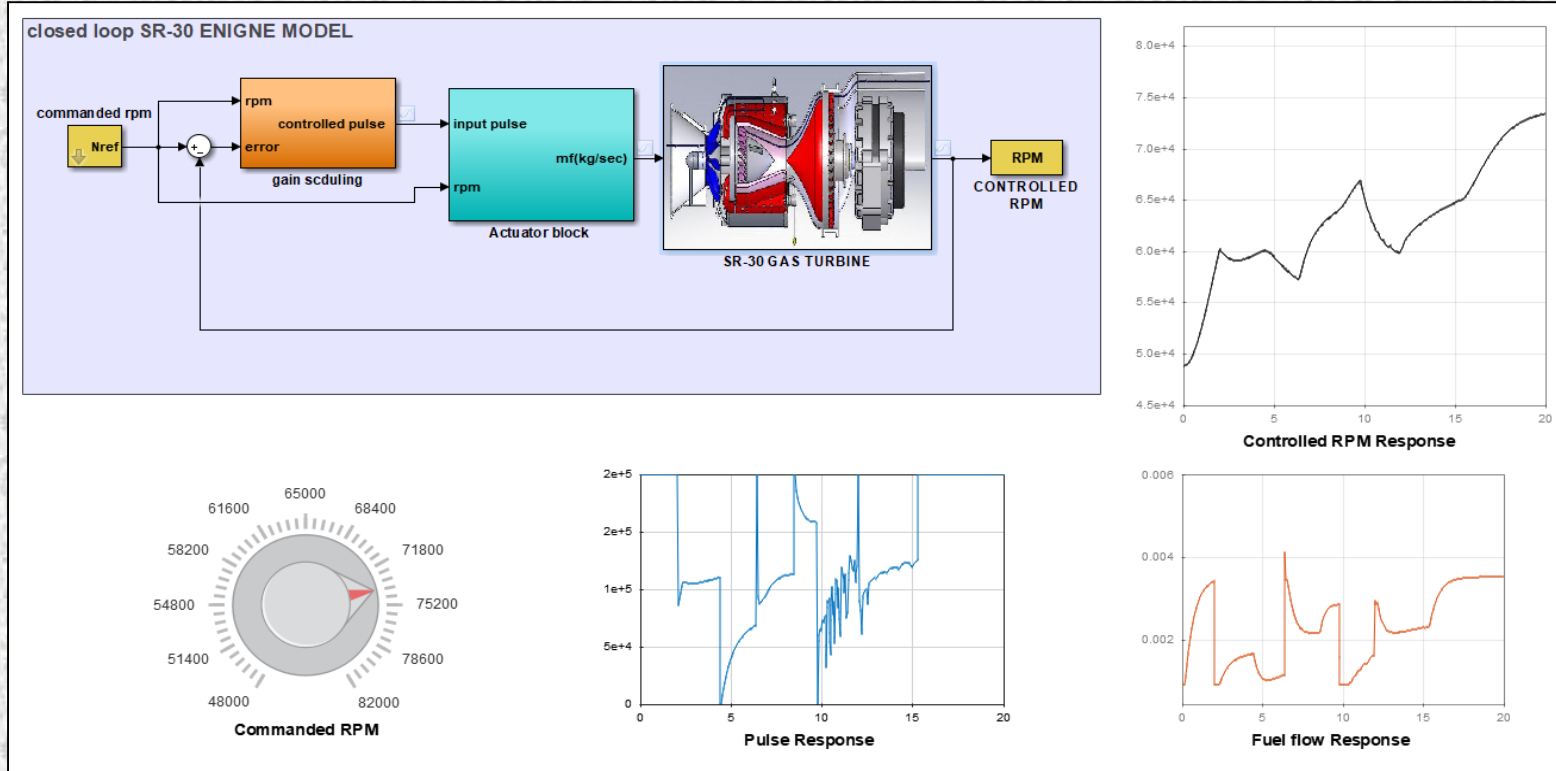


Fig : Component-wise Simulink Model of SR-30 Gas Turbine Engine



# First Principle Based Engine Simulator



**Fig : Closed loop Model of SR-30 Gas Turbine Engine along with dashboard tool**

# Motivation for Data Driven Techniques

---



1. White-box or First principles modeling approaches **rely on thermodynamic and energy balance equations**. Hence, assumptions and linearization methods are required to simplify and solve complex dynamics.
2. Models and control systems designed using **simplified linearized equations** are not accurate enough to capture system dynamics precisely.
3. The **unavailability of component maps** is also one of the key reason to shift on data driven modeling techniques.
4. Thus, Deep learning is a fair alternative to white box model as it is independent of the system dynamics with an objective of maximize system robustness, output power and efficiency.

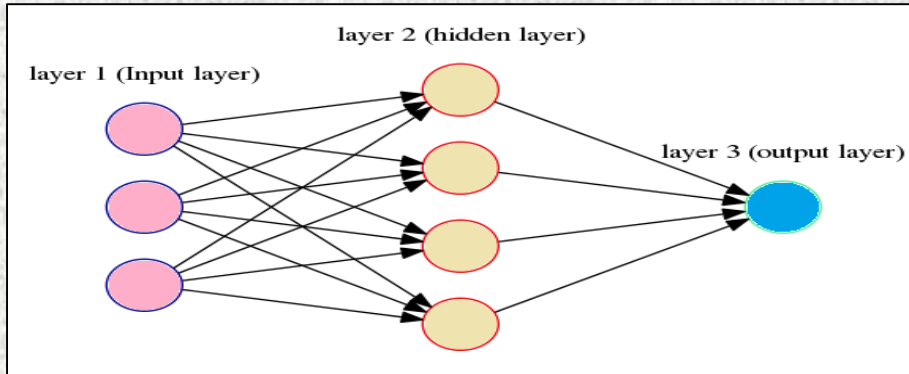
# Neural Network Architecture



The model can be mathematically represented as:

$$y(t) = f (y(t - 1), y(t - 2), \dots, y(t - n_y), u(t-1), u(t-2), \dots, u(t- n_u))$$

where  $y (.)$  is Output,  $u (.)$  is Input and  $n$  represents the Delay unit.



**Fig : Network Architecture**

# Neural Network Architecture



The model can be mathematically represented as:

$$y(t) = f(y(t-1), y(t-2), \dots, y(t-n_y), u(t-1), u(t-2), \dots, u(t-n_u))$$

where  $y(\cdot)$  is Output,  $u(\cdot)$  is Input and  $n$  represents the Delay unit.

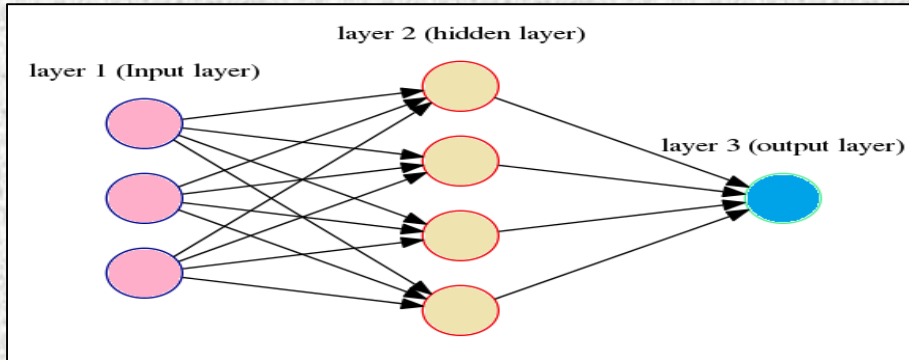
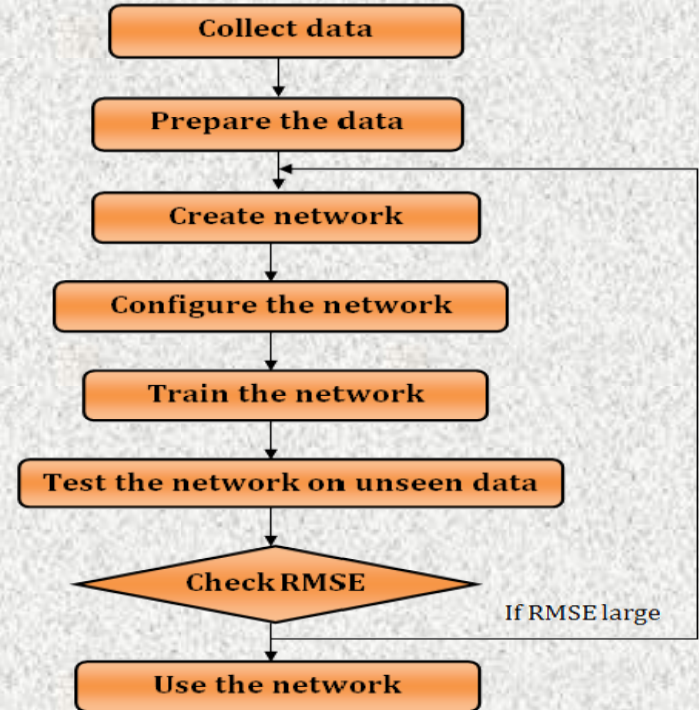


Fig : Network Architecture

## How to build Deep neural network?



# LSTM Network Architecture



- **Forget gate:**

$$f_t = \sigma(W_f[y_{t-1}, x_t] + b_f)$$

- **Input gate**

$$i_t = \sigma(W_i[y_{t-1}, x_t] + b_i)$$

$$\hat{C}_t = \tanh(W_c[y_{t-1}, x_t] + b_c)$$

- **Cell memory state**

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t$$

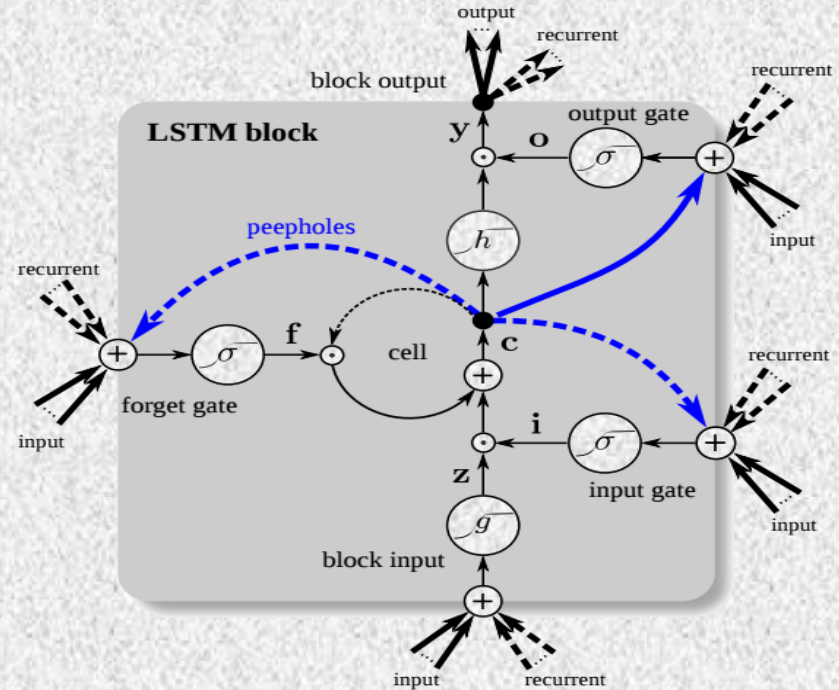
- **Output gate**

$$o_t = \sigma(W_o[y_{t-1}, x_t] + b_o)$$

$$y_t = o_t * \tanh(C_t)$$

Where:  $W$  is weight,  $b$  is bias,  $x$  is input data,  $y$  is target data.

$$\tanh(x) = \frac{2}{1+e^{-2x}} - 1 \quad \sigma(x) = \frac{1}{1+e^{-x}}$$



**Fig : Detailed schematic of LSTM block**

# Network Configuration



MATLAB R2018a - trial use

HOME PLOTS APPS EDITOR PUBLISH VIEW Search Documentation Log In

New Open Save Find Files Compare Go To Print FILE NAVIGATE EDIT BREAKPOINTS RUN

Insert Comment Indent Breakpoints Run Run and Advance Run Section Advance Run and Time

E: > richa singh > PhD > matlab\_18 work > LSTM\_simulink model\_neural >

Current Folder

- sr-30\_modified model
- GT\_experiment.csv
- GT\_old\_data\_prbs.csv
- matlab.mat
- MIMO\_ZIGZAG\_test.m
- mini\_lab\_aero\_data\_1.csv
- MISO\_testing\_example.m
- SISO\_TEST.csv
- siso\_testing\_example.m
- Zigzag\_mimo.dat

Workspace

Name	Value
data_mimo	7308x12 double
data_MIMO	7308x10 double
data_size	7308
i	1461
inputSize	3
j	7
k	3
l	7
layers	5x1 Layer
mu_XTest	[107.4929,308.6166,14...
mu_XTrain	[107.9147,308.8272,14...
mu_YTest	[5.5124e+04,406.4763...
mu_YTrain	[5.6280e+04,411.5672...

Editor - MIMO\_ZIGZAG\_test.m

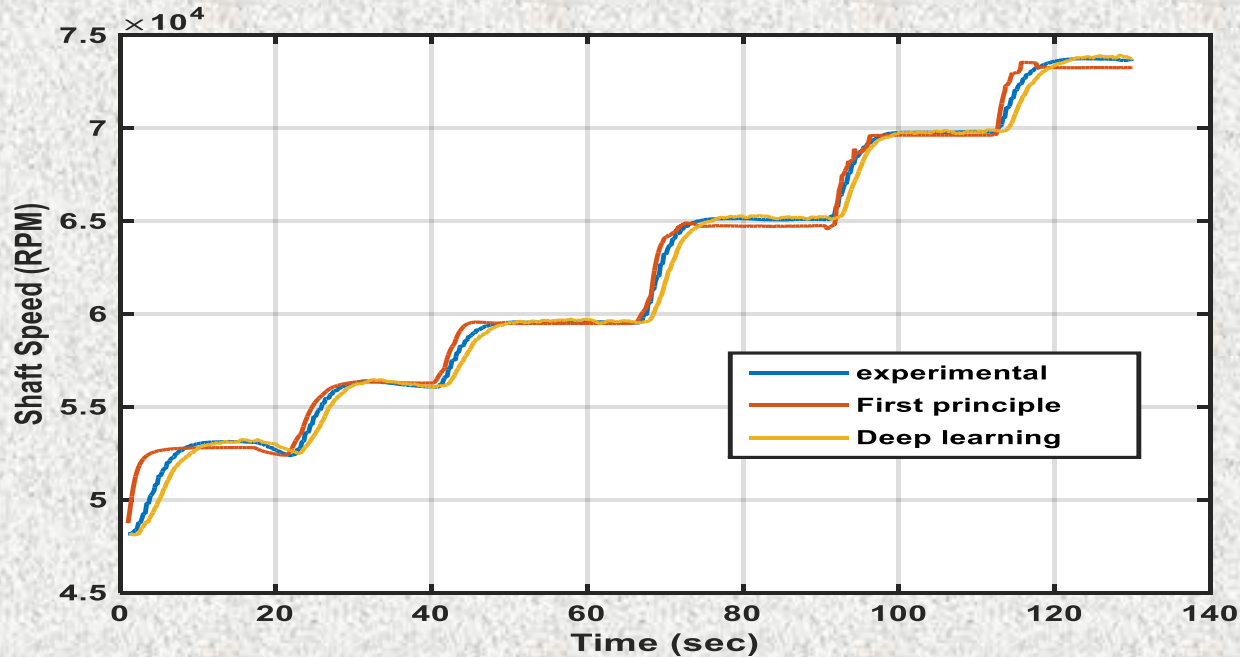
```
MIMO_ZIGZAG_test.m
51 — inputSize = 3;
52 — numResponses = 7;
53 — numHiddenUnits1 = 24;
54 — numHiddenUnits2 = 20;
55
56 — layers = [ ...
57     sequenceInputLayer(inputSize)
58     lstmLayer(numHiddenUnits1)
59     lstmLayer(numHiddenUnits2)
60     fullyConnectedLayer(numResponses)
61     regressionLayer];
62
63 — opts = trainingOptions('adam', ...
64     'MaxEpochs',250, ...
65     'GradientThreshold',1, ...
66     'InitialLearnRate',0.005, ...
67     'LearnRateSchedule','piecewise', ...
68     'LearnRateDropPeriod',125, ...
69     'LearnRateDropFactor',0.2, ...
70     'Verbose',0, ...
71     'Plots','training-progress');
```

script Ln 59 Col 31 Trial Days Remaining: 22

# Model Validation Against Experimental Data

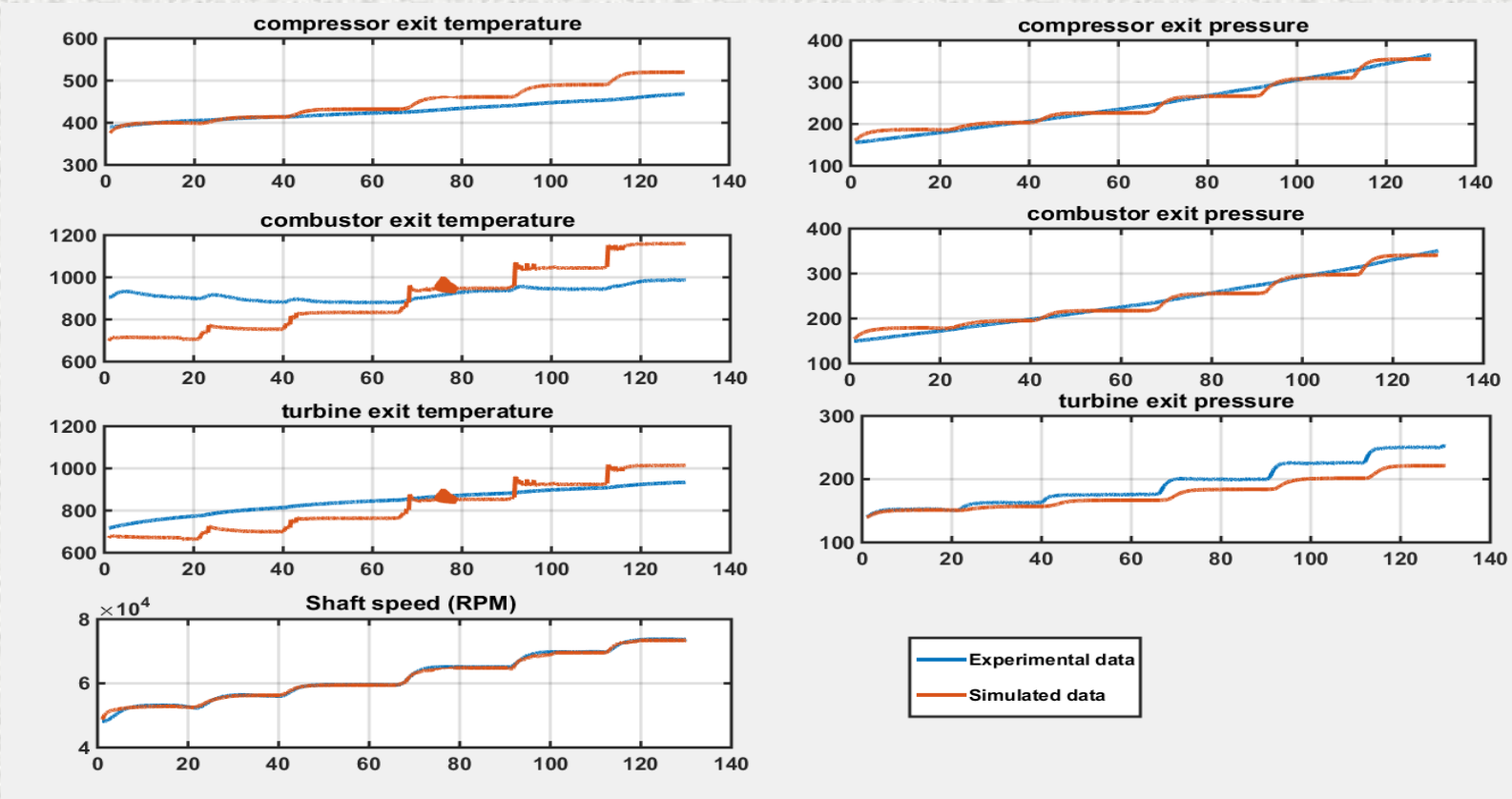


The validation results of First principle model as well as Deep learning based model against experimental data is represented for shaft speed (full range RPM)



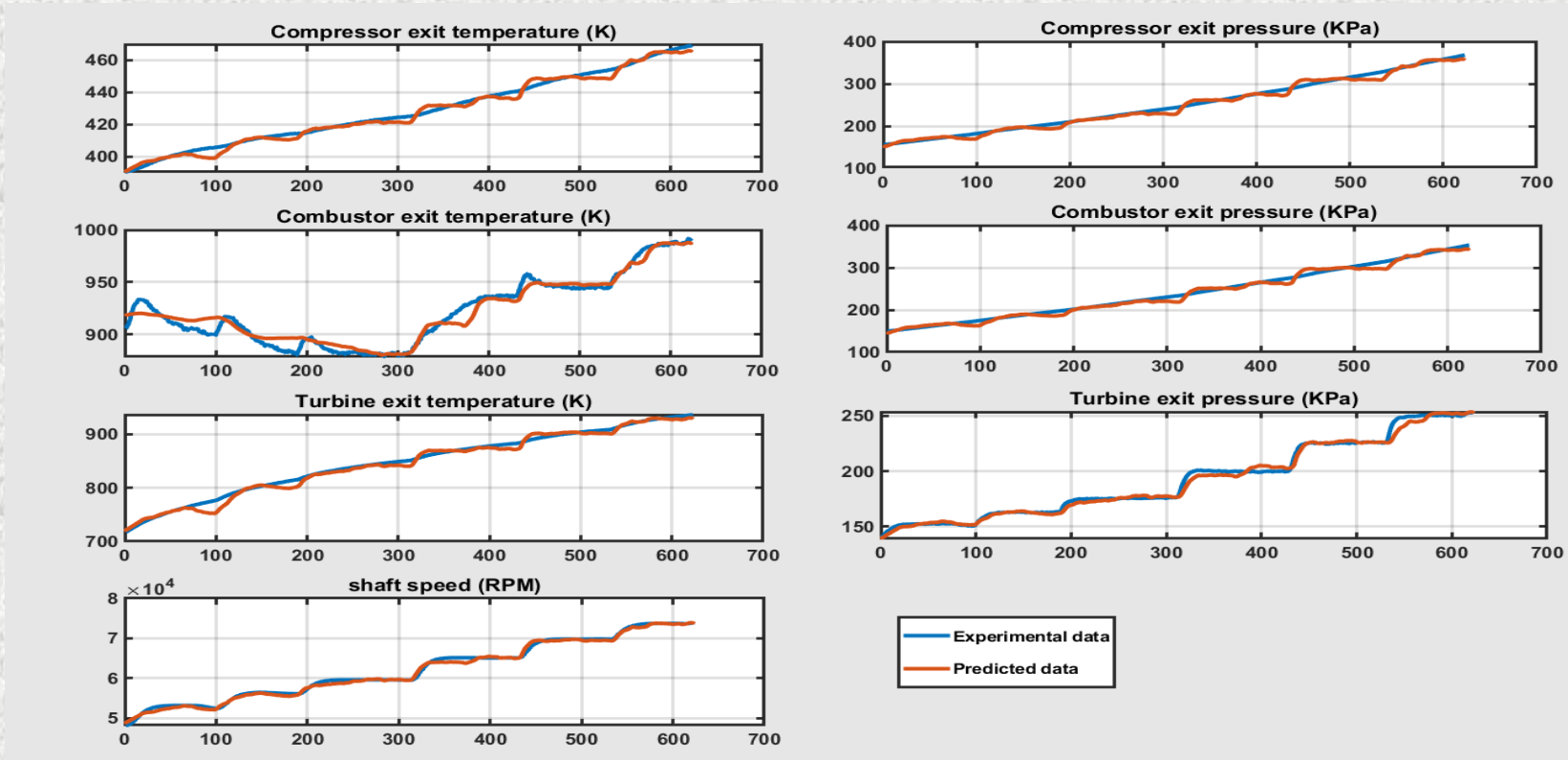
**Fig: Shaft speed validation using First Principle model and Deep Learning model**

# First Principle Model Validation Against Experimental Data

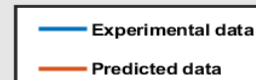
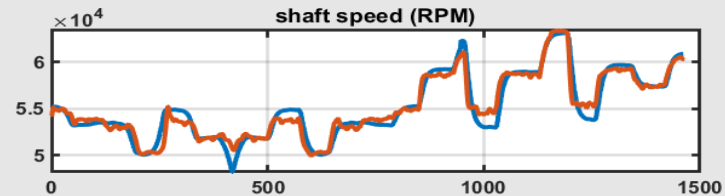
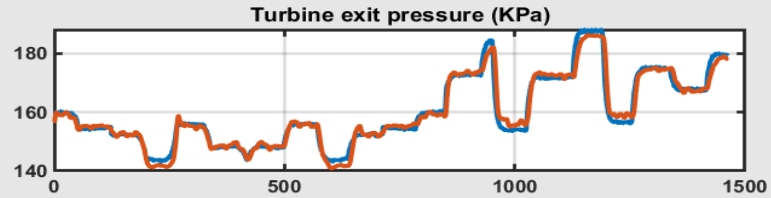
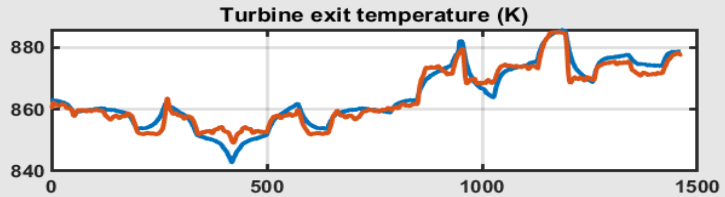
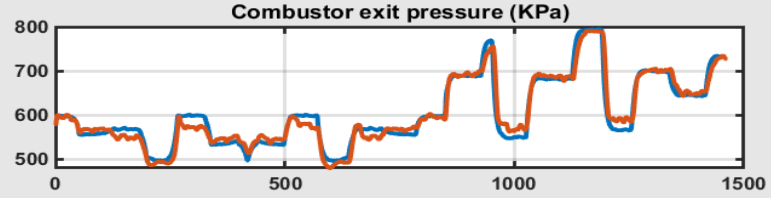
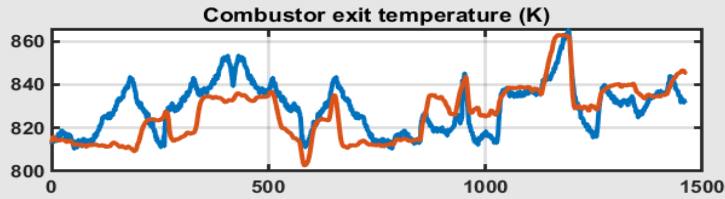
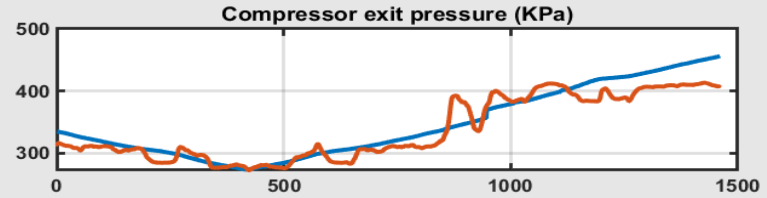
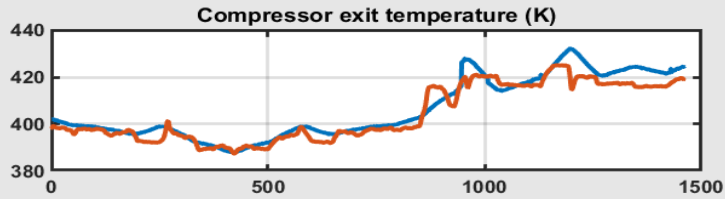




# Deep Learning Model Validation Against Experimental Data



# Deep Learning Model Validation Against Experimental Data



# Relative Error of predicted data against experimental data



$$Error = \frac{1}{N} \sum \left| \frac{y_{exp} - y_{pred}}{y_{exp}} \right|$$

Parameter	Deep Learning Approach	First Principle Approach
T2	0.002	0.0441
P2	0.0012	0.0112
T3	0.0031	0.1297
P3	0.0011	0.0255
T4	0.009	0.1428
P4	0.0036	0.0721
N	0.001	0.0014

# Conclusions

---



- First Principle based method promises good dynamic behavior when compared with the real time engine, provided that enough information is available.
- The deep learning model is trained with a set of experimental data which makes the model to learn a wide variety of engine behavior.
- The **Deep Learning** approach when compared with the First Principle model against experimental data is found to be **more efficient** in predicting behavior of system.
- LSTM performs **good with MIMO system**, however LSTM has its own disadvantage: it is slower than other normal activation functions which leads to the trial & testing process to be more slow.

# Why Matlab & Simulink?

---

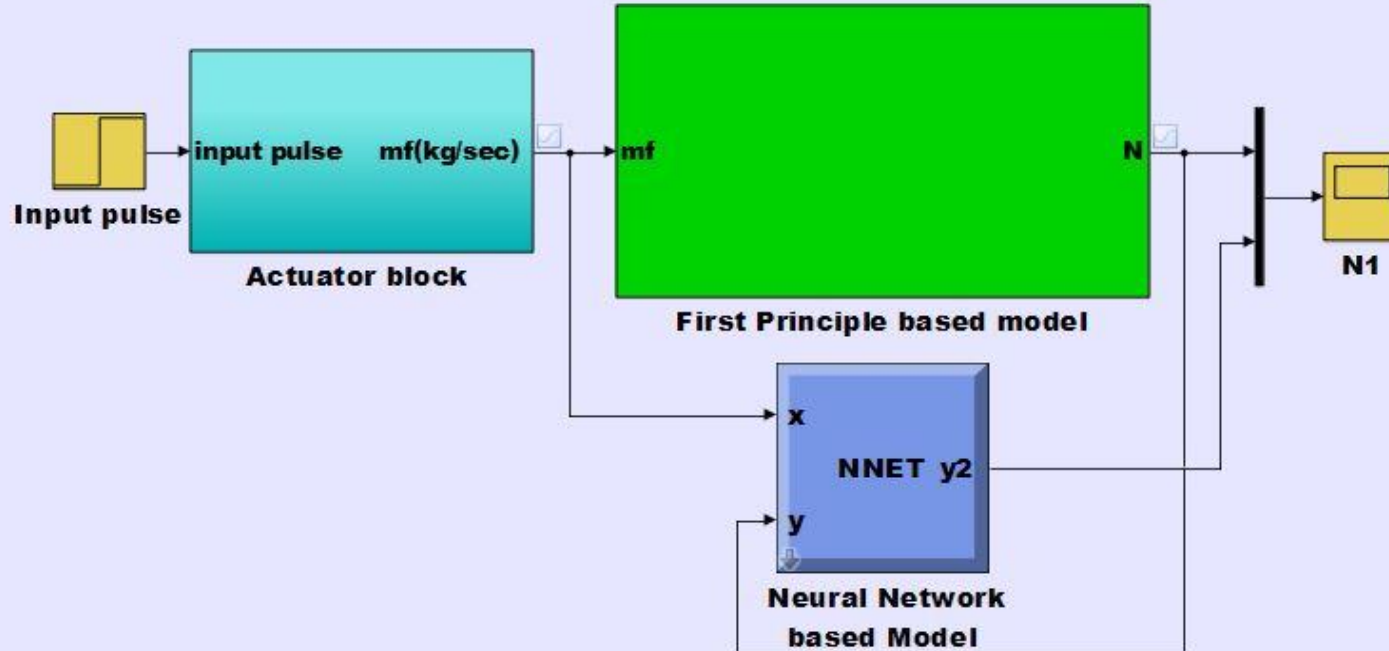


- Easy user-interface
- Less programming required while working in Simulink.
- Toolboxes are designed to integrate with parallel computing environments, GPUs, and automatic C code generation.
- Documentation is written for engineers and scientists, not computer scientists.
- Inbuilt functions are available that are required in day-to-day computation.
- MATLAB App let you start working right away and then automatically generate a MATLAB program to reproduce or automate your work.

# Future Steps in Modeling and Control of Engine



Comparison of First Principle based and NN Based GT Model



# Acknowledgements



Experimental setup :



- **Supervisor:**
  - P. S. V. Nataraj
- **Co - worker**
  - Bhagyashri Somani (deep learning)
- **Data collection from experimental setup:**
  - Swathi Surendran
  - Sanjeet Kulkarni

