CYPHER Training School

Machine Learning for Reacting Flows
Dynamical System Modelling
Hands-on session

Anh Khoa Doan

TU Delft

Short instructions

Scripts and data available here:

https://github.com/adoanTUD/CYPHER_MLSchool

Same instructions as this morning/yesterday (search for adoanTUD on google colab)

Slides also available on the git repo

Structure of the session

1. 5-minute recap
2. Guide through example notebook: Lorenz system
3. Description of the exercise: Learning flame dynamics

Modelling of time series with feedforward neural networks

- Assume we want to model a time-dependent process:

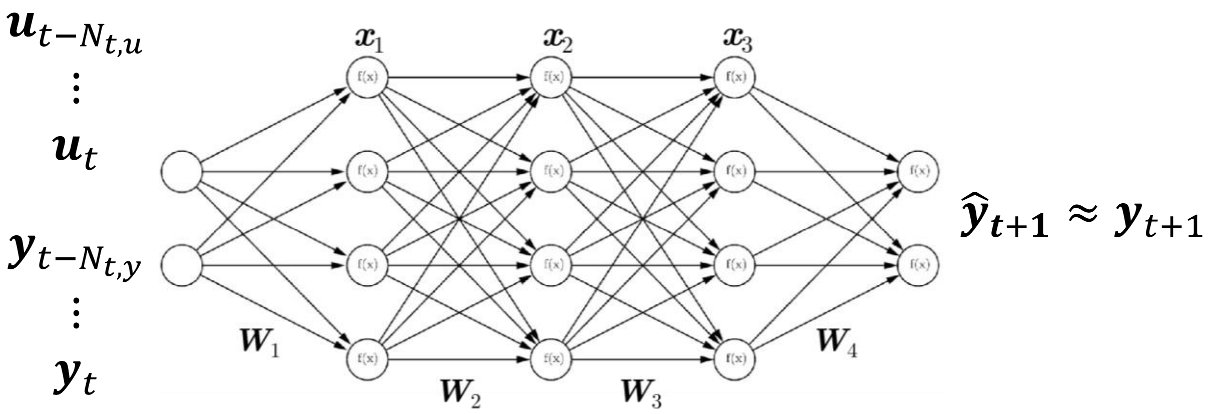
$$\dot{\boldsymbol{y}}(t) = \mathcal{F}(\boldsymbol{y}, \boldsymbol{u})$$

$$\rightarrow \boldsymbol{y}_{t+1} = \mathcal{F}_d\left(\boldsymbol{y}_t, \dots, \boldsymbol{y}_{t-N_{t,y}}, \boldsymbol{u}_t, \dots, \boldsymbol{u}_{t-N_{t,u}}\right)$$

Typical feedforward-based architecture

Time series of past input and prediction

$N_{t,u}$, $N_{t,y}$: duration of "relevant past information"



Modelling of time series with recurrent neural network

- Assume we want to model a time-dependent process:

$$\dot{\boldsymbol{y}}(t) = \mathcal{F}(\boldsymbol{y}, \boldsymbol{u})$$

$$\rightarrow \boldsymbol{y}_{t+1} = \mathcal{F}_d\left(\boldsymbol{y}_t, \dots, \boldsymbol{y}_{t-N_{t,y}}, \boldsymbol{u}_t, \dots, \boldsymbol{u}_{t-N_{t,u}}\right)$$

Two possible recurrent approaches with RNNs:

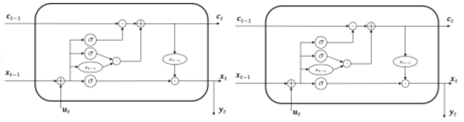
Time series of past input and prediction

$N_{t,u}$, $N_{t,y}$: duration of “relevant past information”

“Many to one” approach:

$\boldsymbol{u}_{t-N_{t,u}}$
$\vdots$
$\boldsymbol{u}_t$

$\boldsymbol{y}_{t-N_{t,y}}$
$\vdots$
$\boldsymbol{y}_t$



$\hat{\boldsymbol{y}}_{t+1}$

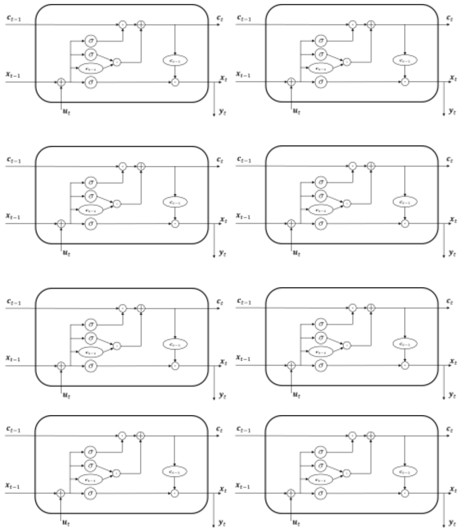
“Many to many” approach:

$[\boldsymbol{u}_1, \boldsymbol{y}_1]$ → $\hat{\boldsymbol{y}}_1$

$[\boldsymbol{u}_2, \boldsymbol{y}_2]$ → $\hat{\boldsymbol{y}}_2$

$\vdots$

$[\boldsymbol{u}_t, \boldsymbol{y}_t]$ → $\hat{\boldsymbol{y}}_{t+1}$



Structure of the session

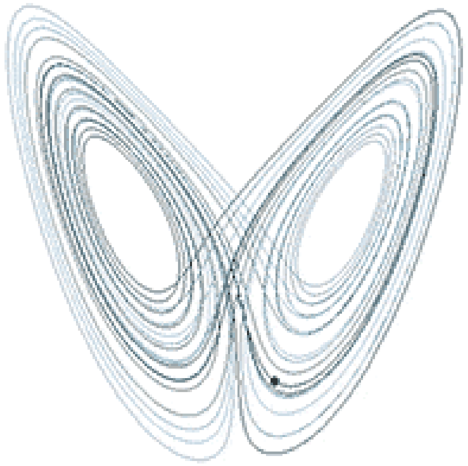
1. 5-minute recap
2. **Guide through example notebook: Lorenz system**
3. Description of the exercise: Learning flame dynamics

Guided exercise: Lorenz system

- Developed originally as a model of atmospheric convection

$$\frac{dx}{dt} = \sigma(y - x), \frac{dy}{dt} = x(\rho - z) - y, \frac{dz}{dt} = xy - \beta z$$

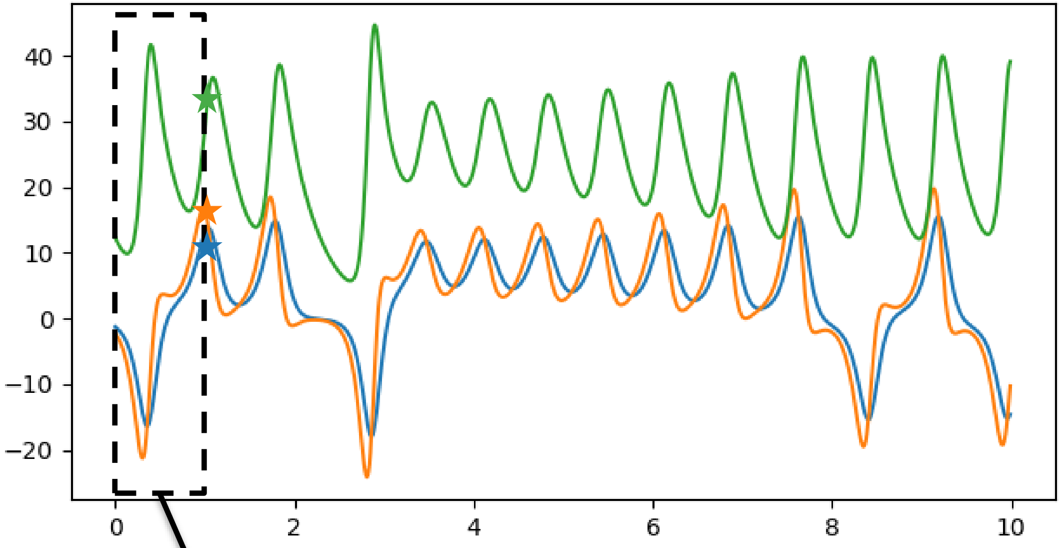
- Simplified model that exhibits chaotic behaviour.



$\rho = 28, \sigma = 10, \beta = 8/3$

Guided exercise: Lorenz system

Data: Time series of the Lorenz system

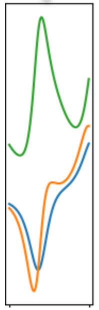


input

$\boldsymbol{x}_{t-N}$

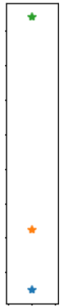
$\vdots$

$\boldsymbol{x}_{t-1}$



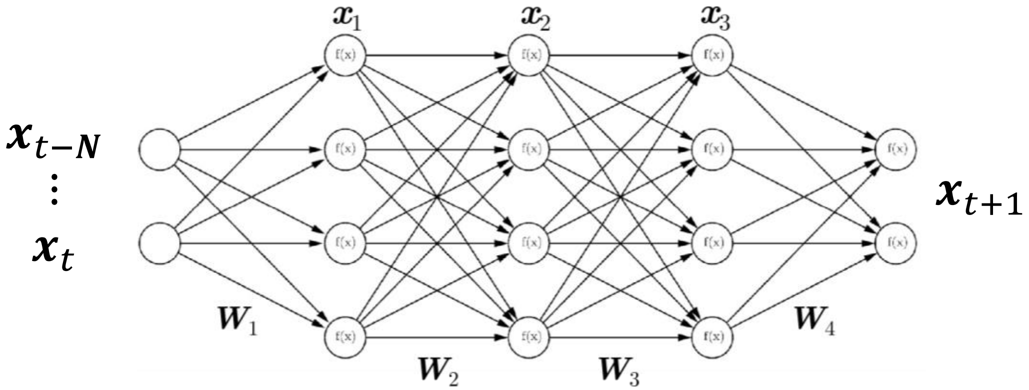
target

$\boldsymbol{x}_t$



Objective: predict next time step based on previous ones:

$$\boldsymbol{x}(t+1) = \mathcal{F}\big(\boldsymbol{x}(t), \ldots, \boldsymbol{x}(t-N)\big)$$



Steps:

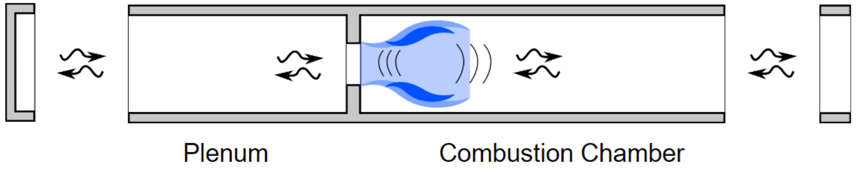
1. Restructure datasets into appropriate (input,pair)
2. Design the network
3. Train

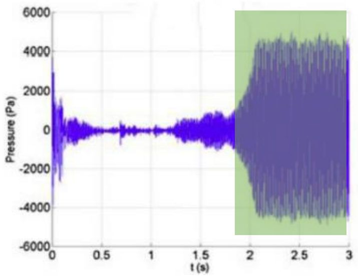
Structure of the session

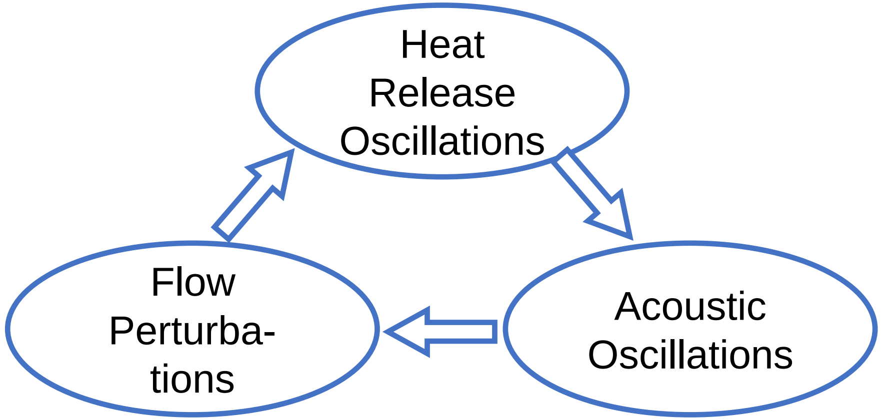
1. 5-minute recap
2. Guide through example notebook: Lorenz system
3. **Description of the exercise: Learning flame dynamics**

Thermoacoustic instabilities result from heat release rate oscillations

- Thermoacoustic instabilities: High amplitude pressure oscillations in ‘lean premixed’ combustors



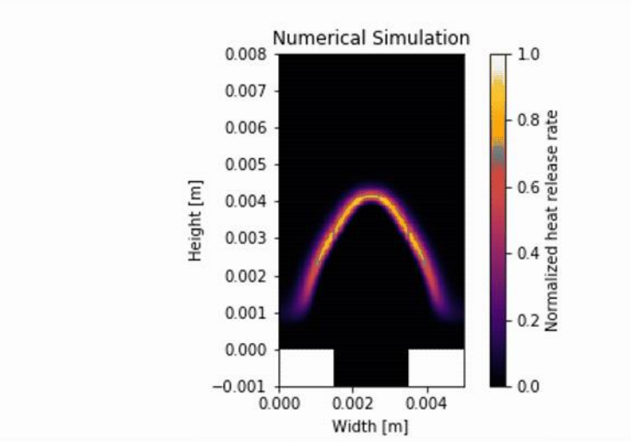


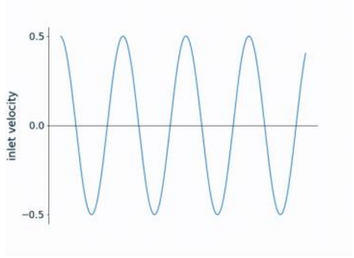


- Prediction requires a coupling between:
  - *Acoustic model*
  - $\boxed{\textit{Flame model: } \dot{Q}' = f(u')}$

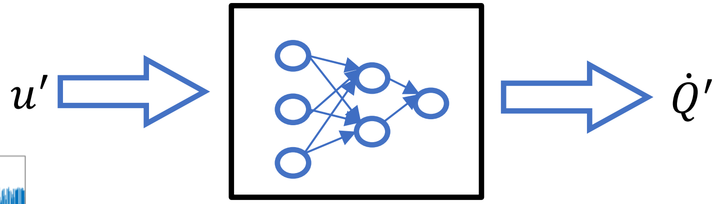
NN must be trained to learn flame dynamics ($\dot{Q}' = f(u')$) of a laminar slit flame

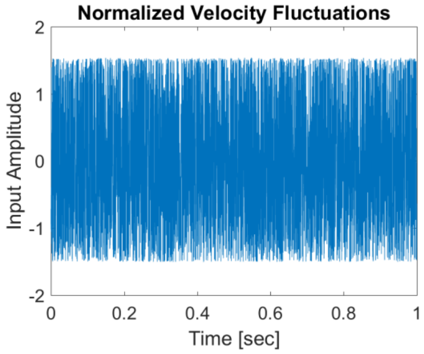
- Premixed methane-air laminar slit burner (equivalence ratio of 0.8)
- Fully resolved with DNS

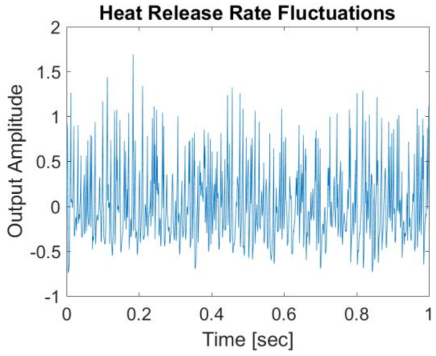




Objective: Design a neural networks trained using the data from broadband simulation







Similar steps as for Lorenz case except input and output are different signals

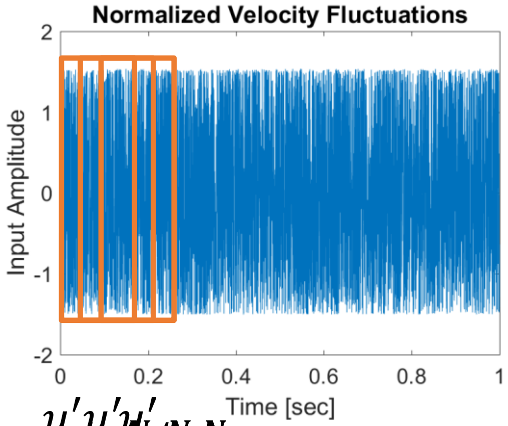
Kornilov et al. PROCI (2007)
Tathawadekar et al., PROCI (2021)

Overall exercise

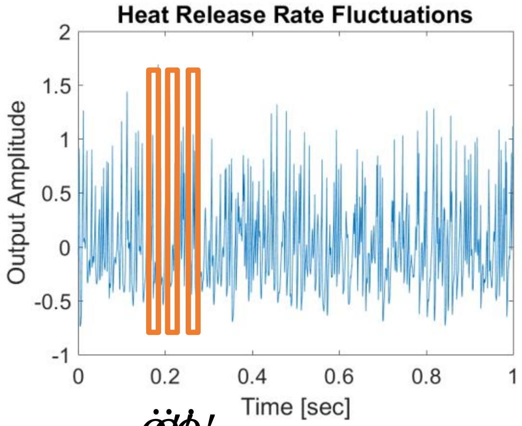
- Objective: Develop a neural network model of the flame response
- 3 training datasets provided for different amplitudes of excitations
- Two approaches possible
  - Feedforward NN
  - Recurrent neural network
- Study of the accuracy of the trained NN depending on the training dataset used
- Study on the impact of the length of the dataset used for training

Steps for the exercise

1. Preprocess the dataset
   – Consider undersampling
2. From the data provided, create the appropriate “input/output” pairs
   – Pay attention of your choice of “history” (N)
3. Design and train the network



u'_{t-N} $\vdots$ u'_t

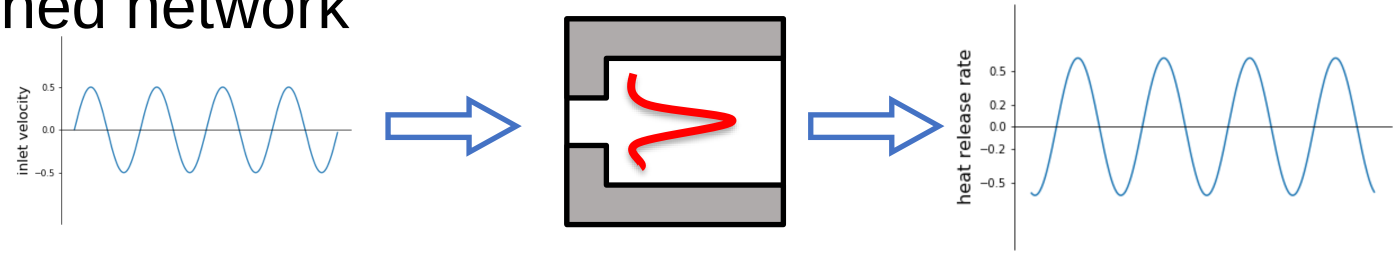


$\dot{Q}'_t$

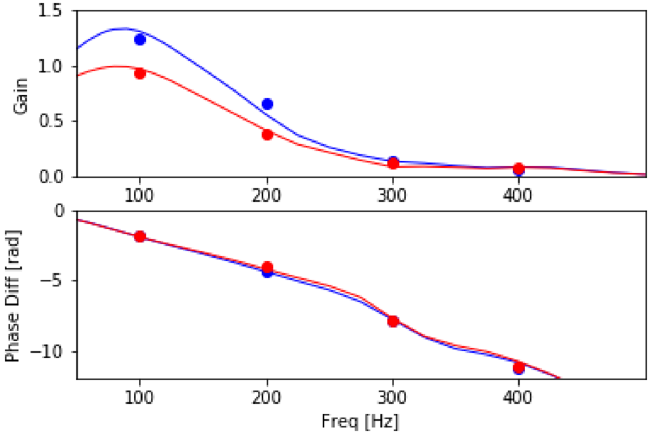
Assessment of the trained neural network : comparison with harmonic forcing

4. Compute the flame describing function $F(\omega, |u^{\prime}|) = \dfrac{\dot{Q}^{\prime}(\omega,|u^{\prime}|)/\overline{\dot{Q}}}{u^{\prime}(\omega,|u^{\prime}|)/\bar{u}}$

– Create harmonic signals at given frequencies
– Feed them to your trained network

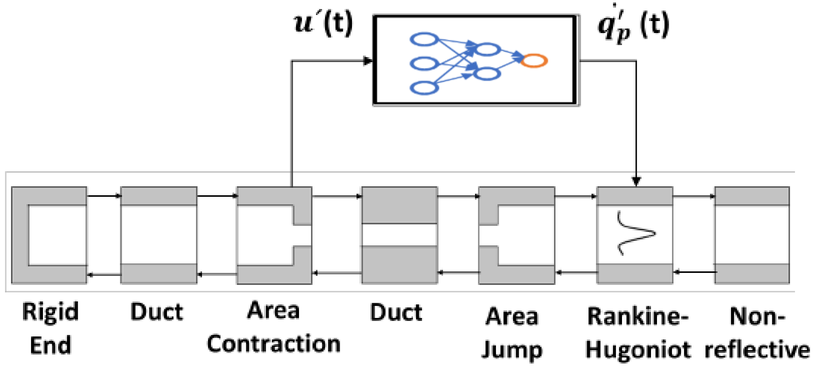


– Deduce the gain ("ratio" between input and output) and phase (normalised time shift between input/output)

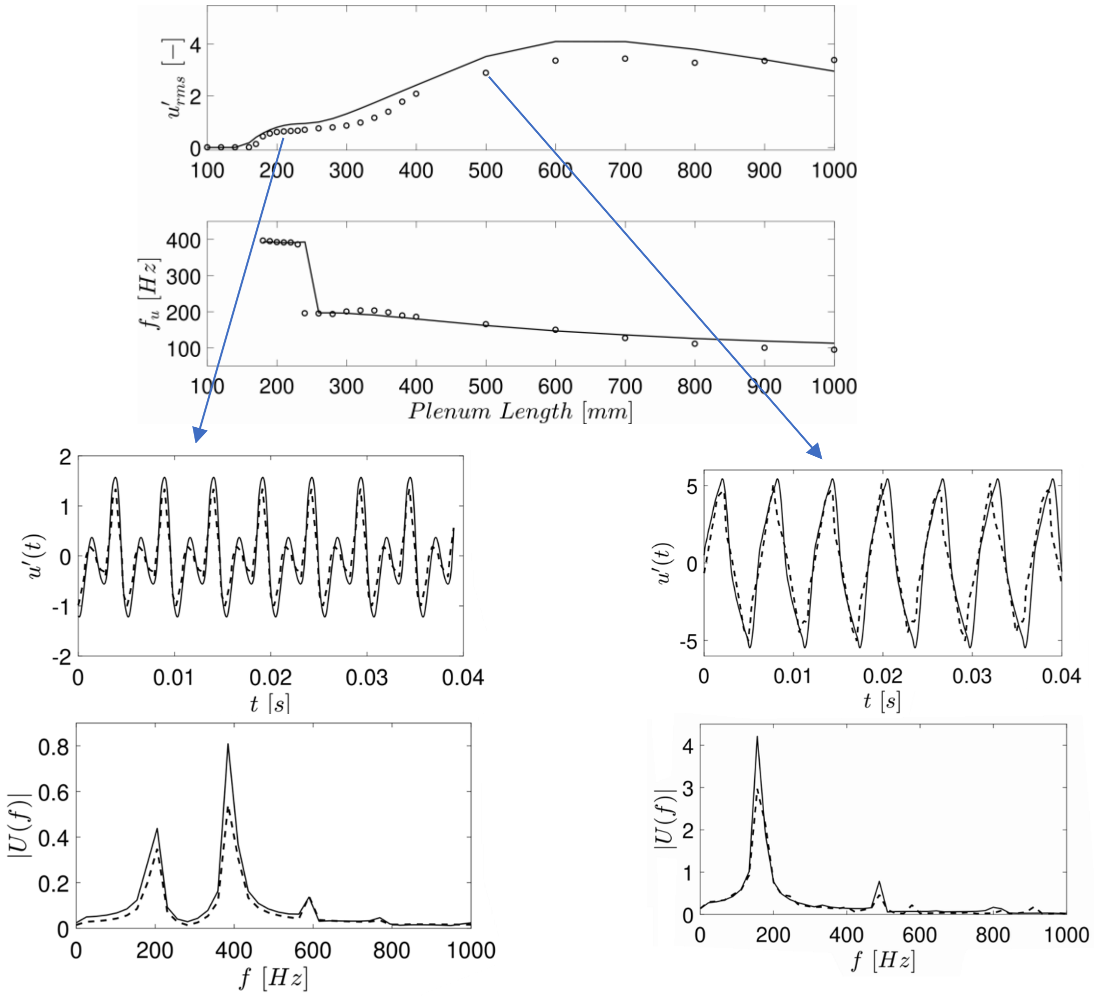


Integrated NN and acoustic network predicts accurately the thermoacoustic instability

- Trained NN with acoustic model describes the gas turbine



Stability analysis for different designs possible



Tathawadekar et al., PROCI (2021)