

CYPHER Training School

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

Anh Khoa Doan

# Short instructions

Scripts and data available here:

[https://github.com/adoanTUD/CYPHER\\_MLSchool](https://github.com/adoanTUD/CYPHER_MLSchool)

Upload data into google drive

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:

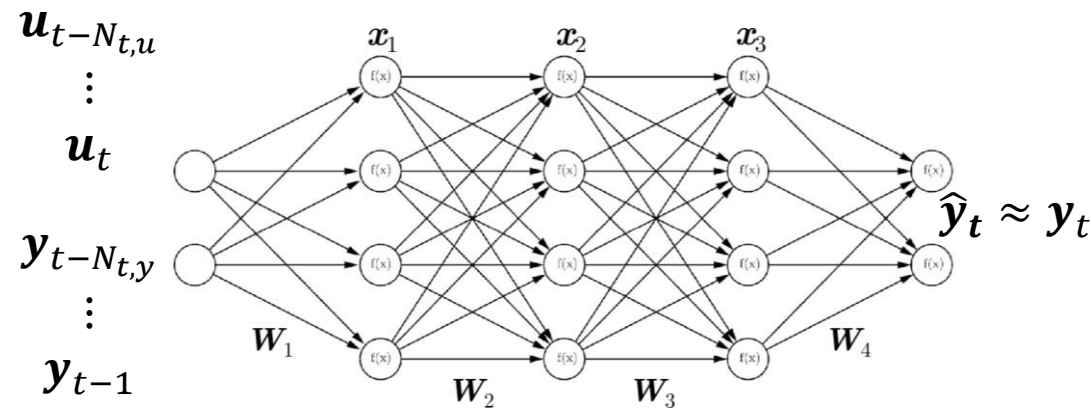
$$\dot{\mathbf{y}}(t) = \mathcal{F}(\mathbf{y}, \mathbf{u})$$

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

## 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

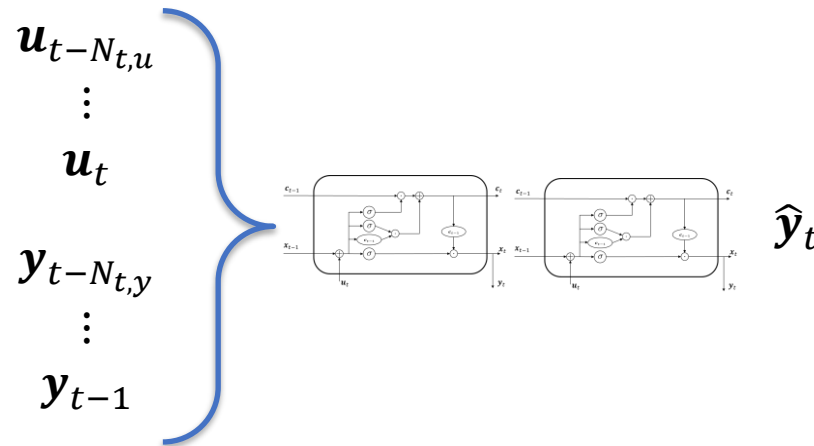
- Assume we want to model a time-dependent process:

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

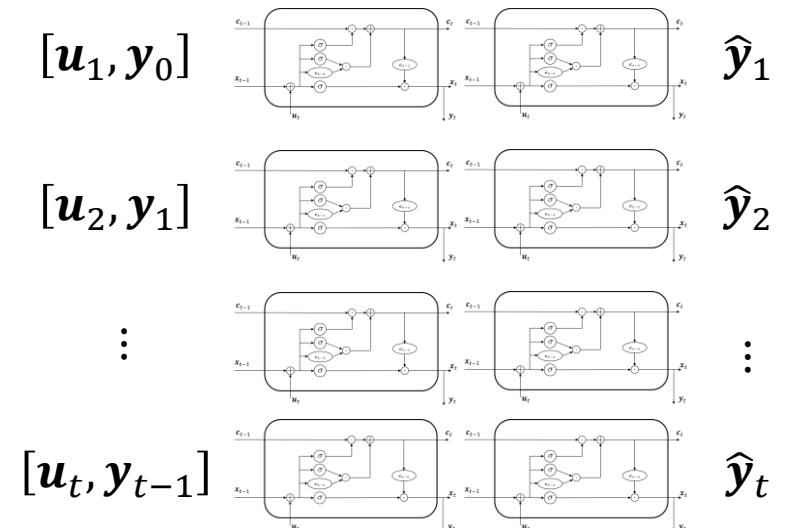
$$\rightarrow \mathbf{y}_t = \mathcal{F}_d \left( \mathbf{y}_{t-1}, \dots, \mathbf{y}_{t-N_{t,y}}, \mathbf{u}_t, \dots, \mathbf{u}_{t-N_{t,u}} \right)$$

Two possible recurrent approaches with RNNs:

“Many to one” approach:



“Many to many” approach:

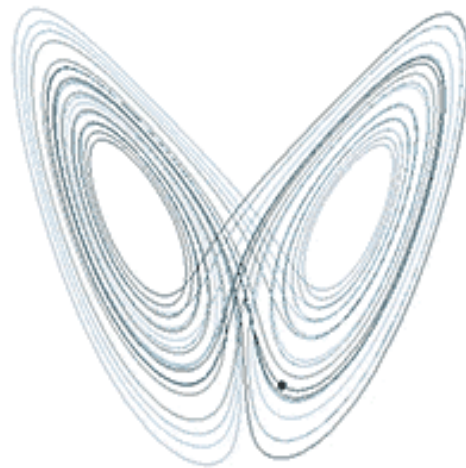


# Structure of the session

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

# Guided exercise: Lorenz system

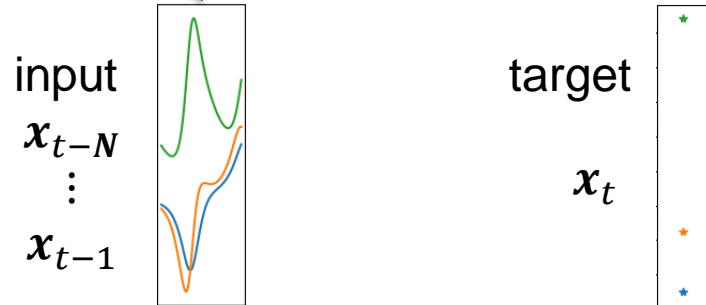
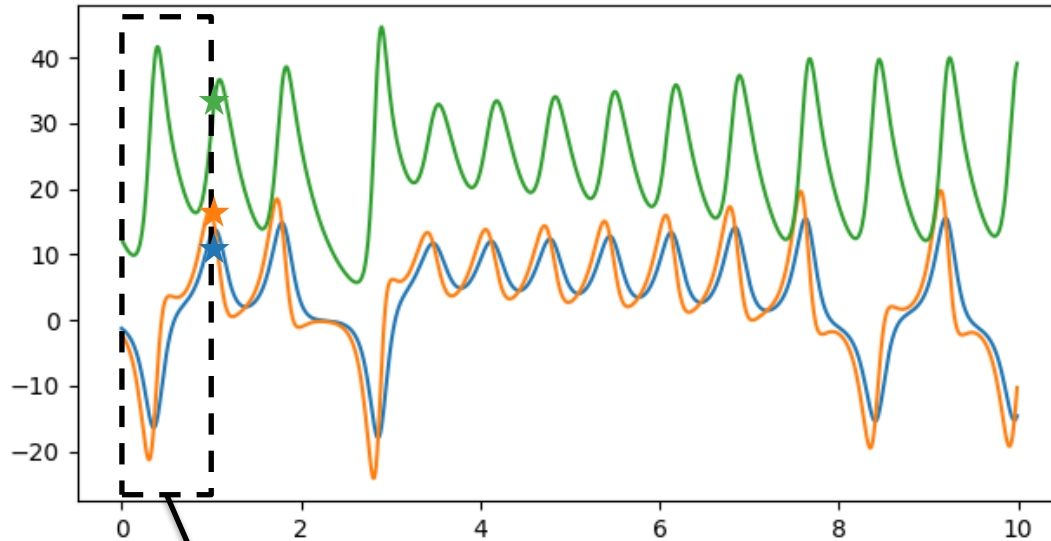
- Developed originally as a model of atmospheric convection
$$\frac{dx}{dt} = \sigma(y - x), \frac{dy}{dt} = x(\rho - z) - y, \frac{dz}{dt} = xy - \beta z$$
- Simplified model that exhibits chaotic behaviour.



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

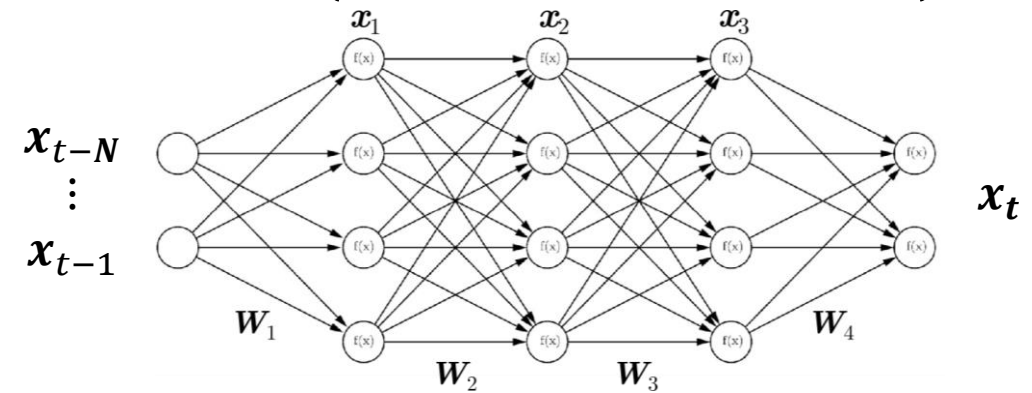
# Guided exercise: Lorenz system

Data: Time series of the Lorenz system



Objective: predict next time step based on previous ones:

$$x(t) = \mathcal{F}(x(t-1), \dots, x(t-N))$$



Steps:

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

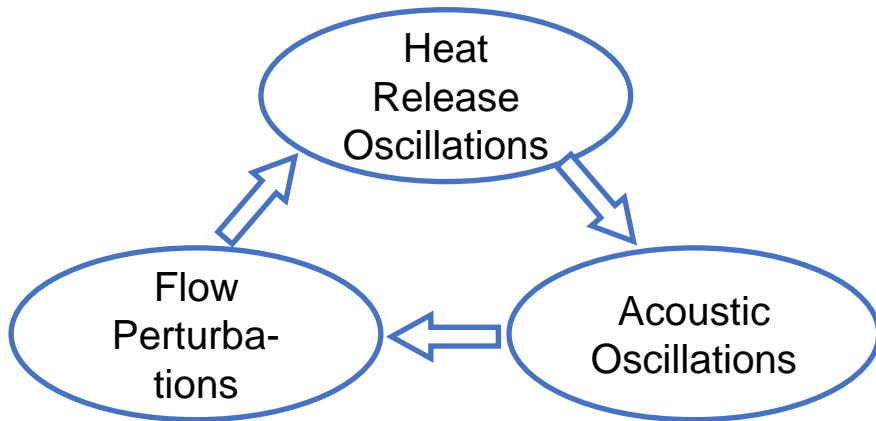
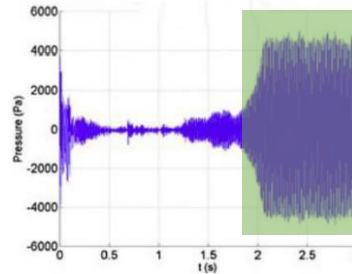
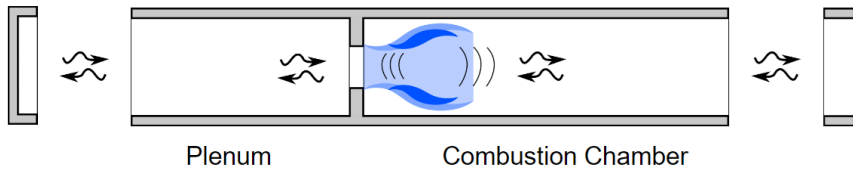


# Structure of the session

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



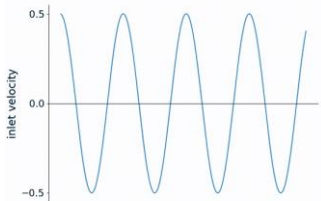
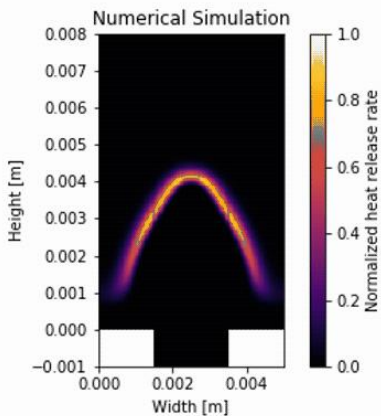
- Prediction requires a coupling between:

- *Acoustic model*

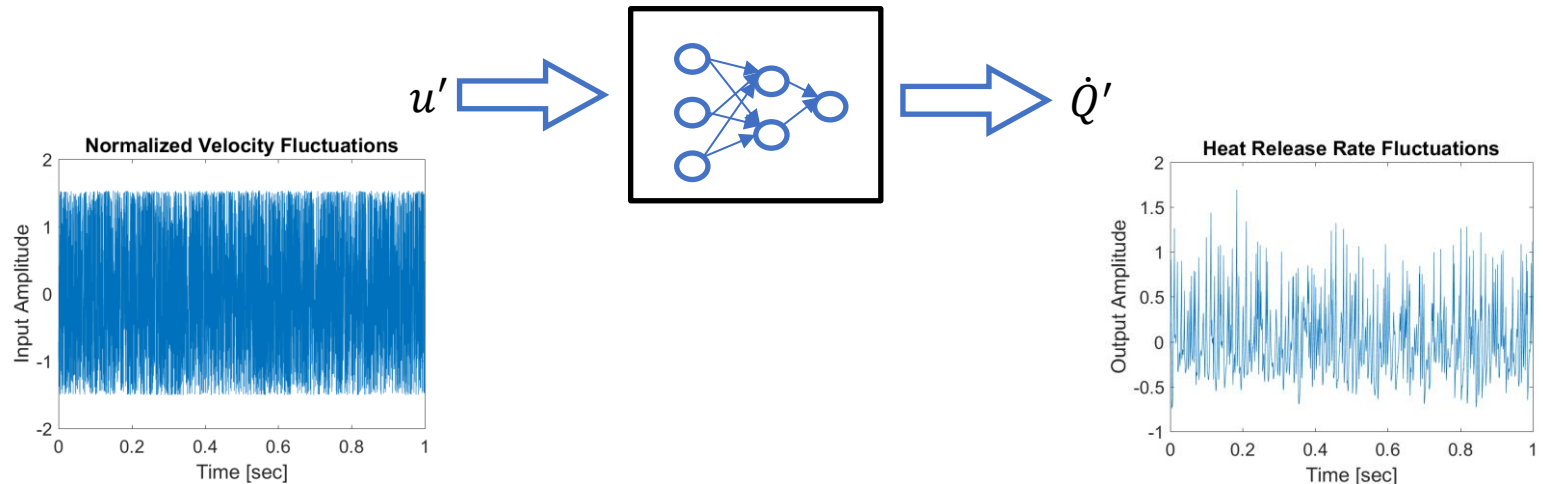
- *Flame model:  $\dot{Q}' = f(u')$*

# 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



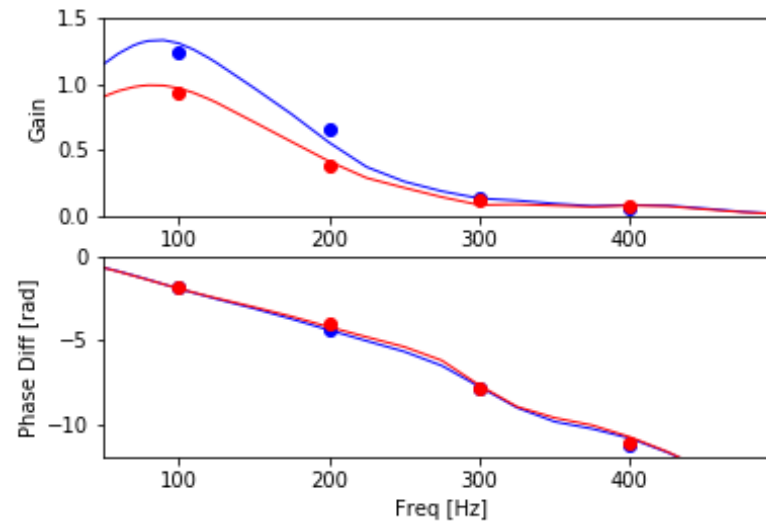
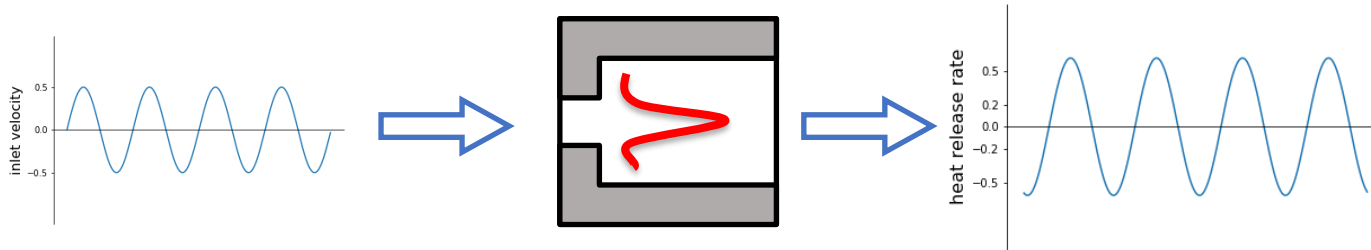
Objective: Design a neural networks trained using the data from broadband simulation



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

# Assessment of the trained neural network : comparison with harmonic forcing

- $$F(\omega, |u'|) = \frac{\dot{Q}'(\omega, |u'|)/\bar{\dot{Q}}}{u'(\omega, |u'|)/\bar{u}}$$



- 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