ME5311 - Project 1: Data analysis (50% of the final grade)

Instructions

Each student should use the dataset provided for the project, and conduct a thorough analysis of the dataset, choosing some of the tools that you learned from lectures 1, 2, 3, and 4.

Please note that this project focuses exclusively on data analysis. Modelling the system dynamics or performing prediction tasks must **NOT** be done for the Project 1 (this will be the focus of Project 2).

Report

You are required to compile a **brief report** using the template provided on Overleaf without changing the font style, size, or document margins. The report should be:

- maximum 3-page long (excluding references);
- containing a maximum of 1 figure;
- containing a maximum of 20 references (note, references are not counted for the page limit).

The Overleaf template can be found here: https://www.overleaf.com/read/xxhqcbxktxxq#b2892b
We ask you to copy the project and start drafting your own report. For those who have no experience with LaTeX, the only files you need to modify are: ME5311-template.tex (for the report), and ME5311-references.bib (for the bibliography), there are several tutorials online freely available regarding the use of LaTeX.

Submission

The report of Project 1 is due **by 20th March at 23:59** (no late submission will be accepted). The project is worth 50% of the final grade.
You are required to submit 3 files (where XXXXXXXX represent your student id, e.g. A0123456R):

- **XXXXXXXX_report.pdf** file containing the report.
- **XXXXXXXX_code.zip** file containing all the code you wrote (either Matlab or Python, other programming languages are not accepted).
- **XXXXXXXX_selfeval.txt** file where you self-evaluate your own report.

We will be ultimately grading all your projects, based on the report; however, we ask you to grade your own project, to help you develop a critical eye towards your own work. Failing to comply with these rules (as well as changing the report style, font or borders) may be treated as a non-submission and/or penalized.

AI Policy

We follow the University AI policy. You are required to mention any eventual AI tool usage (in the full process, whether it be code, report writing, or something else). For more information about NUS AI Policy visit link. A single sentence at the end of the report is enough. Simply state what type of tool you used and for what scope you used it; no declaration means that you did not use any AI tool (this sentence will not count towards the 3-page limit).

Analysis guidelines

The objective is to explore the dataset using data-driven analysis tools and to extract meaningful structural and statistical information directly from the data.

Students are encouraged to consider, but are not limited to, the following questions:

- What are the dominant spatial structures present in the data, and how can they be identified using techniques such as singular value decomposition (SVD)?

- How is variance or energy distributed across spatial scales, and what does spectral analysis reveal about the characteristic length scales of the system?
- Can the presence of an externally imposed, spatially periodic input be inferred directly from the data, for example through distinct features in the Fourier spectrum?
- How do the spatial and temporal characteristics of the data reflect underlying symmetries or anisotropies of the system?

Modelling the system dynamics or performing prediction tasks must **NOT** be done for the Project 1 (this will be the focus of Project 2).

Dataset (~ 1 GB): Spatio-temporal Data

Dataset description

The dataset is available at the following link `https://nusu-my.sharepoint.com/:f:/g/personal/e1500557_u_nus_edu/IgBD9b1ASir8Q5bwz7O3XYewAa-ox6PVUEoWV0TZ0xsjSCM`. The dataset considered in this project consists of time-resolved snapshots of a two-component vector field evolving on a two-dimensional spatial domain. The system is defined on square domain of size $L \times L$, with periodic boundary conditions.

The data represent the evolution of a spatio-temporal dynamical system subject to a prescribed, spatially periodic external forcing. This external for introduces a preferred spatial scale characterized by a certain wavenumber, which imprints a dominant length scale onto the system dynamics and is expected to be detectable directly from the data.

The vector field is discretized on a uniform Cartesian grid with a spatial resolution of 64×64, and both components of the field are recorded at each grid point. The dataset is sampled at a constant temporal interval of $\Delta t = 0.2$ (in simulation time units) and spans a total duration of 3000 time units, resulting in a long, time-resolved sequence of snapshots.

Each snapshot therefore contains 8,192 degrees of freedom corresponding to the two field components over the spatial grid. This dataset provides a clean and well-controlled example of a spatio-temporal system, making it well suited for data-driven analysis techniques such as modal decomposition, spectral analysis, and statistical characterization of spatial and temporal structures.

The vector field is stored in "vector_64.npy" (or vector_64.mat if you prefer to use MATLAB). One snapshot of the field is provided in Fig. 1

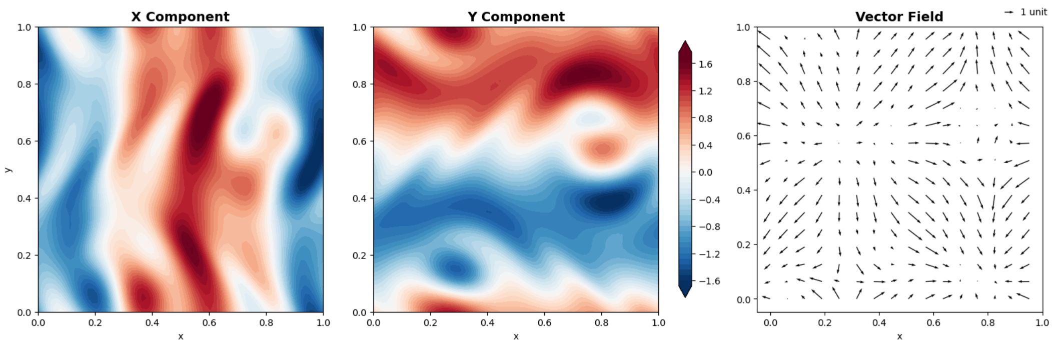

Figure 1: Representative snapshot of the two-dimensional system. Contour plots show the instantaneous X and Y components (left and middle), while the right panel shows the corresponding vector field.

How to load the data

An example of how you can load the data in Python is reported below (file is in the same folder as the dataset):

```python
import numpy as np

# dimensions of data
nx = 64
ny = 64
nt = 15000
shape = (nt, ny, nx, 2)

# load the data
vectors = np.load('vector_64.npy')  # shape (15000, 64, 64, 2)
print("Loaded data with shape:", vectors.shape)
```

Listing 1: Loading the dataset in Python. See file load_data.py

Similarly, you can load the data in Matlab as follows (file is in the same folder as the dataset):

```matlab
% dimensions of data
nx = 64;
ny = 64;
nt = 15000;
shape = [nt, ny, nx, 2];

```

```matlab
% load the data from .mat file
data = load('vector_64.mat');

% inspect variable names in the file (optional)
disp(fieldnames(data));

% the variable is called vector
vector = data.vector;

fprintf('Loaded data with shape: [%s]\n', num2str(size(vector)));
```

Listing 2: Loading the dataset in Matlab. See file load_data.m