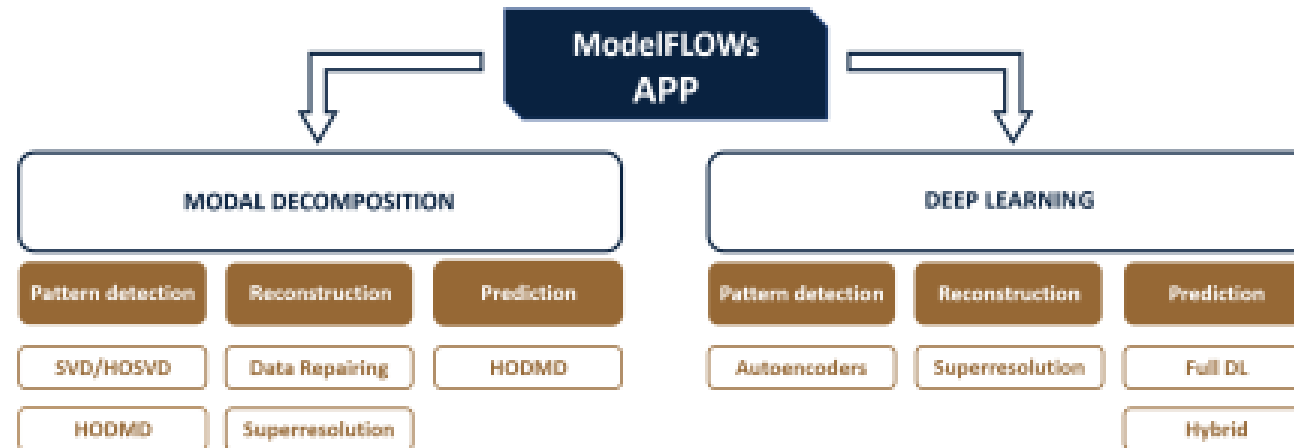
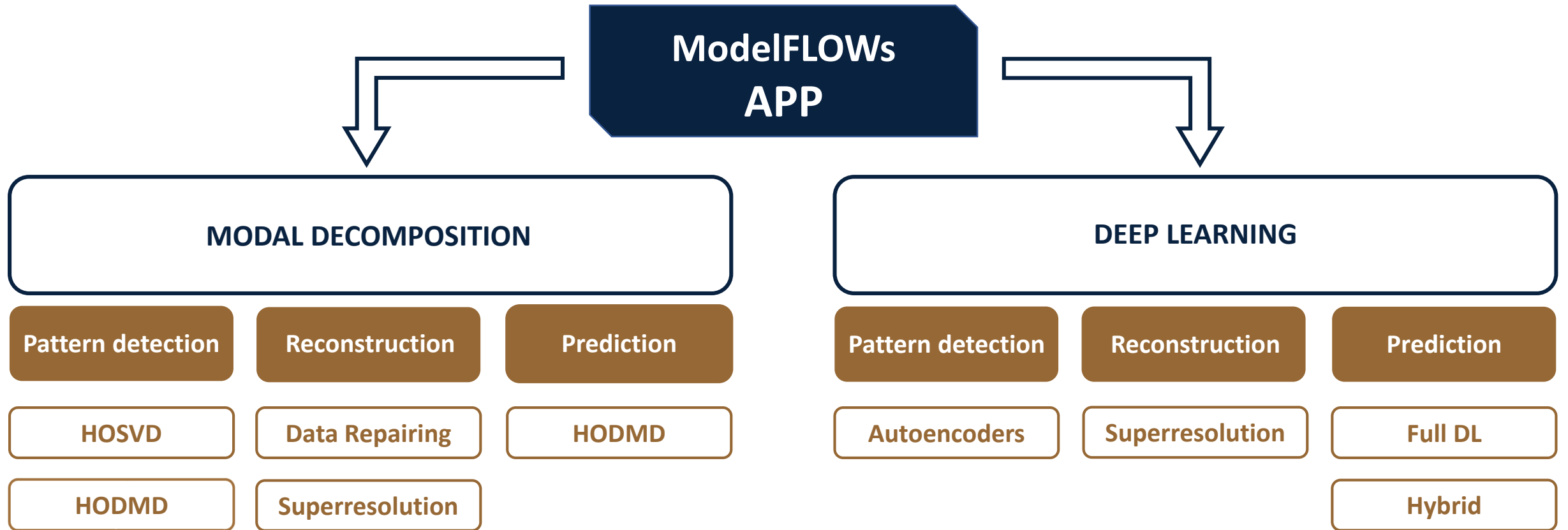


# ModelFLOWS-app

## Installing and using ModelFLOWS-app



# Open Source Software



# Download & Installation

## How can I download ModelFLOWS-app?

ModelFLOWS-app is easy to download:

- 1) Head over to the ModelFLOWS-app GitHub repository ([github.com/modelflows/ModelFLOWS-app](https://github.com/modelflows/ModelFLOWS-app))
- 2) Find the version that you would like to download (mains/desktop/web-browser demo)
- 3) Click on “main” and then the branch corresponding to the version you would like
- 4) Once in the desired branch, click on “code” and select “Download ZIP”
- 5) Done! You have downloaded ModelFLOWS-app

# Download & Installation

## How can I download ModelFLOWS-app?

1

The screenshot shows the GitHub repository page for 'modelflows ModelFLOWS-app'. The interface is annotated with four numbered steps:

- Step 1:** A blue box highlights the 'main' branch dropdown menu in the top left corner.
- Step 2:** A red box highlights the 'Switch branches/tags' dialog box. Within this dialog, the 'Find or create a branch...' search bar is highlighted in blue, and the 'web-browser-version' branch is highlighted in red.
- Step 3:** A yellow box highlights the 'Code' dropdown button in the top right corner of the repository view.
- Step 4:** A green box highlights the 'Download ZIP' option in the bottom section of the 'Code' dropdown menu.

The repository view shows the 'web-browser-version' branch is selected, which is 4 commits ahead and 6 commits behind the 'main' branch. The file list includes 'v0.1\_web-browser', 'LICENSE', and 'Tutorial v0.1 ModelFLOWS-app.pdf'.



ModelFLOWS

# Download & Installation

## How do I “Install” ModelFLOWS-app?

ModelFLOWS-app doesn't require installation, but you may need to install some libraries to your environment.

This application has been developed using Python 3.9 and is compatible with all later versions.

The required libraries are:

Library	Version
tensorflow	2.10
protobuf	3.20
scikit-learn	1.2
keras-tuner	Latest
scipy	1.9.3
numba	0.56.4
hdf5storage	Latest
ffmpeg	Latest

If you are going to use the web-version demo, you will also have to install:

Library	Version
streamlit	1.17
streamlit-option-menu	0.3.2

# Download & Installation

## How do I “Install” ModelFLOWS-app?

There are two options to install these libraries, both are done through the command prompt/line:

Option 1 (**FAST**): In command prompt, head to the ModelFLOWS-app directory (the folder that you have just downloaded) and run the following command:

```
C:\Users\...\Desktop>cd v0.1_web-browser  
C:\Users\...\Desktop\v0.1_web-browser>sudo pip install -r Requirements.txt
```

This will read the *Requirements.txt* file, which is located inside the folder and contains a list of all the necessary libraries and corresponding versions, and install them automatically.

Option 2 (**SLOW**): Install each library manually (you don't need to be inside the directory).

```
C:\Users\...\Desktop>pip install tensorflow == 2.10
```

# Download & Installation

## How do I open ModelFLOWS-app?

This is also done in the command prompt/line. Each version (web-browser/desktop) requires a different, but short and easy to remember command (inside the app directory):

For the web-browser demo: *streamlit run ModelFLOWS\_web.py*

```
C:\Users\      \Desktop\v0.1_web-browser>streamlit run ModelFLOWS_web.py
```

For the desktop version: *python ModelFLOWS\_app.py*

```
C:\Users\      \Desktop\v0.1_desktop>python ModelFLOWS_app.py_
```

# Download & Installation

## Once the app is open...

Both versions of ModelFLOWS-app have very intuitive user-friendly menus, allowing you to easily navigate between the different options:

```
ModelFLOWS-app
-----
ModelFLOWS Application

Authors: ModelFLOWS Research Group - E.T.S.I. Aeronautica y del Espacio - Universidad Politécnica de Madrid
Version: 0.1

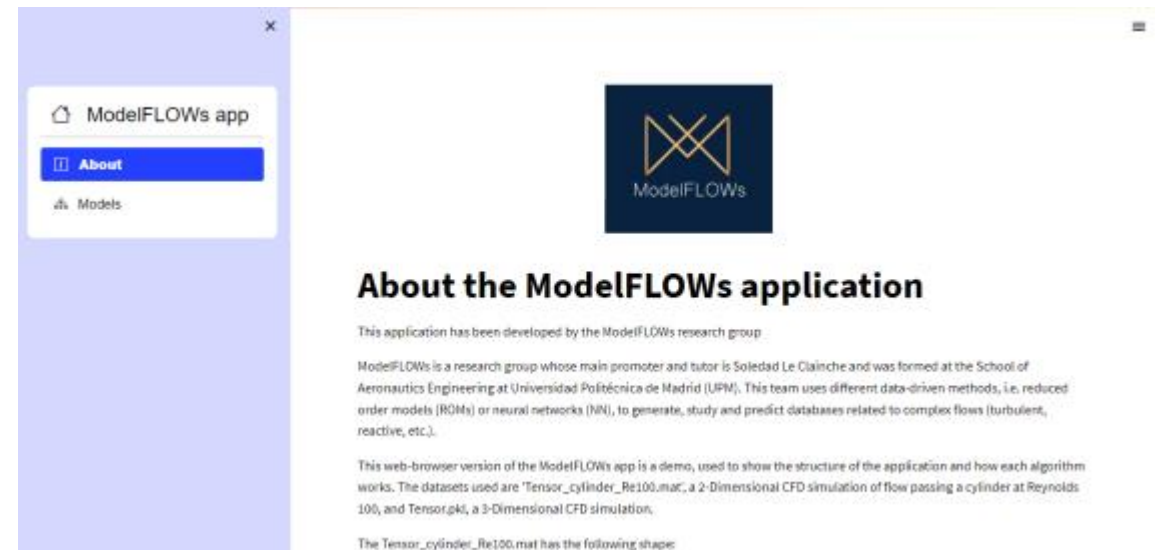
This data-driven application consists of two modules:
- Modal Decomposition
- Deep Learning

Both blocks consist of algorithms capable of:
- detecting patterns,
- repairing and enhancing data,
- and predicting data from complex flow databases

The databases can be in the following formats:
- MATLAB ".mat"
- Numpy ".npy"
- Pickle ".pkl"
- Pandas ".csv"
- h5 ".h5"

This application takes in databases with the following shapes:
- 1D Arrays -> (1, n)
- 2D Matrices -> (x, y)
- 3D Tensors -> (y, x, t)
- 4D Tensors -> (m, y, x, t)
- 5D Tensors -> (m, y, x, z, t)

IMPORTANT: The data that defines the spatial mesh (x, y, z) must be located in the specified positions
For more information please visit: https://modelflows.github.io/modelflowsapp/
```





# Using ModelFLOWS-app

Differences between versions:

Desktop version	Web-browser version
Runs in command prompt/line	Runs in the web-browser (Chrome/Mozilla/Edge...)
User selects the database (.mat, .npy, .csv, .pkl, .h5)	Fixed databases
Flexible	Limited parameter configuration

This tutorial will focus on the web-browser version, since its has been created as a demo for unexperienced users.

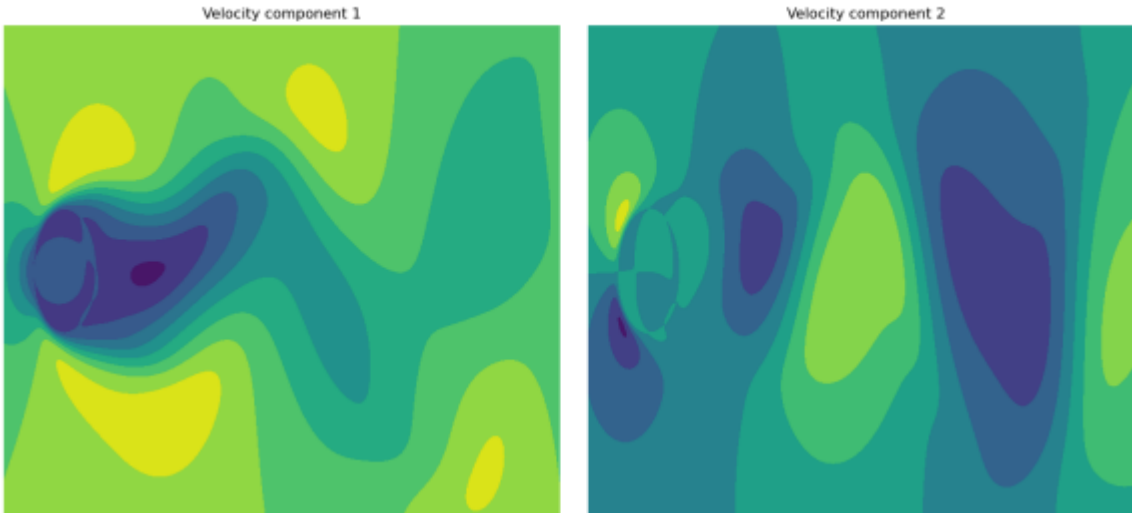
# Using ModelFLOWS-app

## Databases

The following databases are provided:

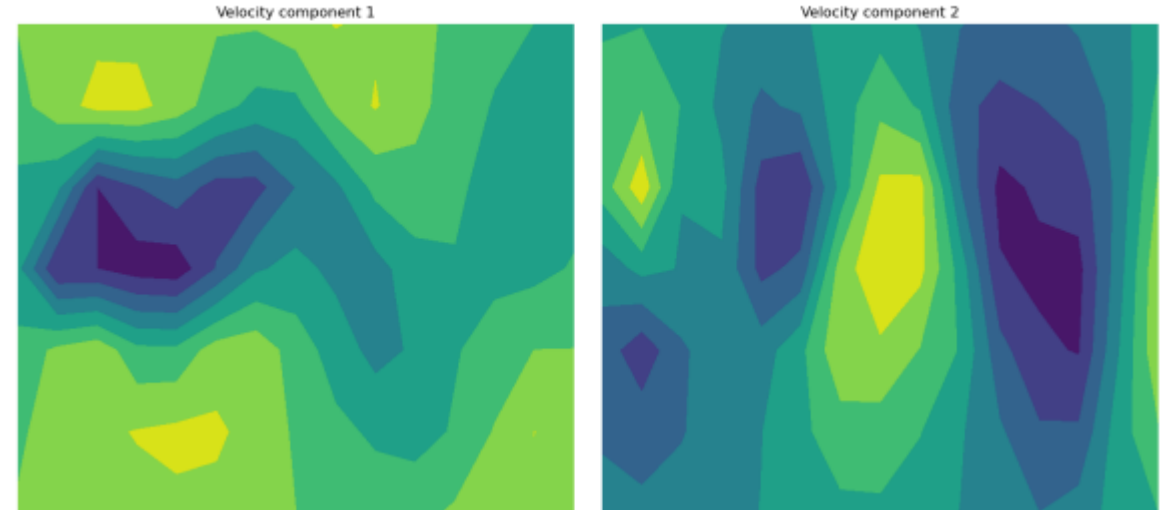
1.1) *Tensor\_cylinder\_Re100.mat* (used in most modules)

Snapshot 1 - Velocity components



1.2) *DS\_30\_Tensor\_cylinder\_Re100.mat*

Snapshot 1 - Velocity components

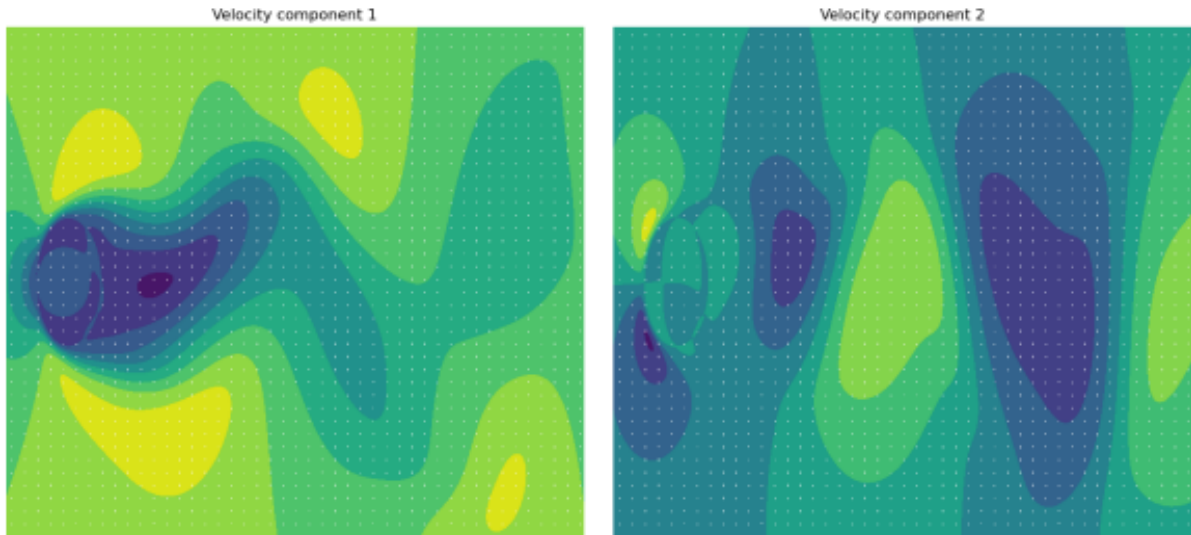


ModelFLOWS

# Using ModelFLOWs-app

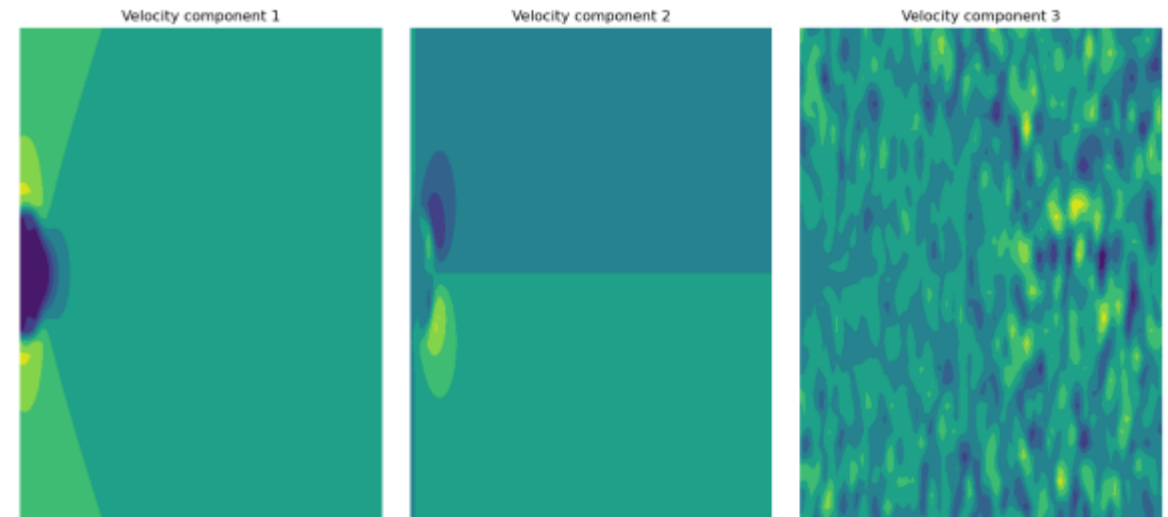
## 1.3) *Gappy\_Tensor\_cylinder\_Re100.mat*

Snapshot 1 - Velocity components



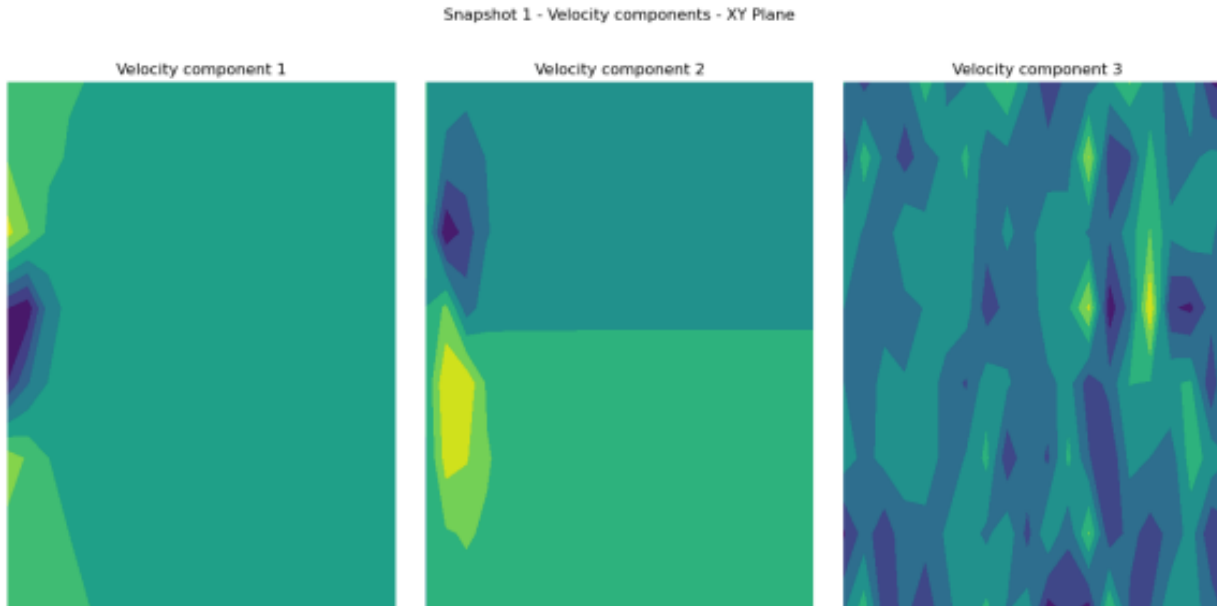
## 2.1) *Tensor.pkl*

Snapshot 1 - Velocity components - XY Plane



# Using ModelFLOWSs-app

## 2.2) *DS\_30\_Tensor.pkl*



**Download the tutorial pdf for more info!**

***Gappy\_Tensor\_cylinder\_Re100.mat***: This variant is a “gappy” version of the database. This means that the database has missing velocity measurements, which have been substituted by NaN (Not A Number) values. It is important to note that the spatial mesh is not reduced in size. See Fig. 6.

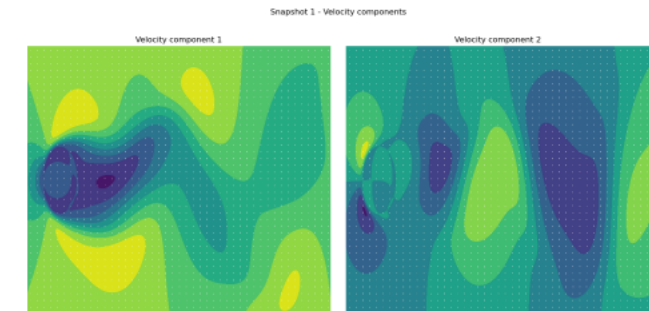


Figure 6: First snapshot velocity components for the *Gappy\_Tensor\_cylinder\_Re100.mat* database.

***Tensor.pkl***: This database corresponds to the 3D wake behind a circular cylinder at Reynolds number 220 and with spanwise wavenumber 4. The database is given in tensor form and is formed by three velocity components (streamwise, normal spanwise), the spatial resolution along the streamwise, normal and spanwise components is composed by  $100 \times 40 \times 40$  grid points and the temporal resolution is represented by 150 snapshots equi-distant in time with time interval  $\Delta t = 1$ . See Fig. 7.

# Using ModelFLOWS-app

## How to run a case

### Singular Value Decomposition, SVD

SVD (Singular Value Decomposition) is a mathematical technique used to analyze and reduce the dimensionality of databases. It decomposes a given database into a set of orthogonal modes, capturing the most important patterns and variability in the data. This allows for efficient storage, compression, and analysis of the information, enabling tasks such as visualization, model reduction, and optimization.

### SVD - Parameter Configuration

Select number of modes to retain during SVD

18

Calculate

Made with Streamlit

## How to run a new case



Run complete!

All files have been saved to C:\Users\ASTHON\Desktop\v 0.1 - ModelFLOWS\_webapp/SVD\_solution/n\_modes\_

Press 'Refresh' to run a new case

Refresh



ModelFLOWS

# Using ModelFLOWS-app

## Modal Decomposition – Pattern Detection

### Singular Value Decomposition, SVD

Select an option

Modal Decomposition ▼

Select an action

Pattern detection ▼

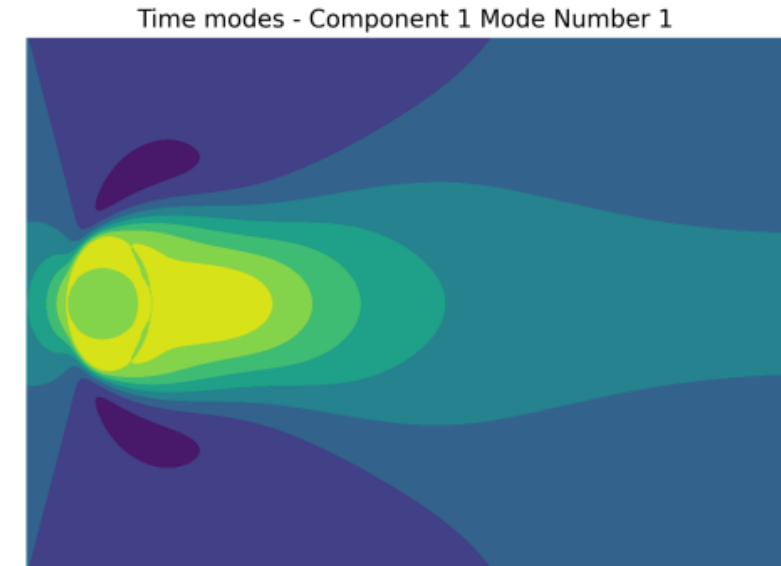
Select an algorithm

☒ SVD

☐ HOSVD

☐ HODMD

Parameter	Default value
Number of SVD modes to retain	18



SVD is a mathematical technique used to analyze and reduce the dimensionality of databases. It decomposes a given database into a set of orthogonal modes, capturing the most important patterns and variability in the data.

# Using ModelFLOWS-app

## Modal Decomposition – Pattern Detection

### Higher-Order Singular Value Decomposition, HOSVD

Select an option

Modal Decomposition ▼

Select an action

Pattern detection ▼

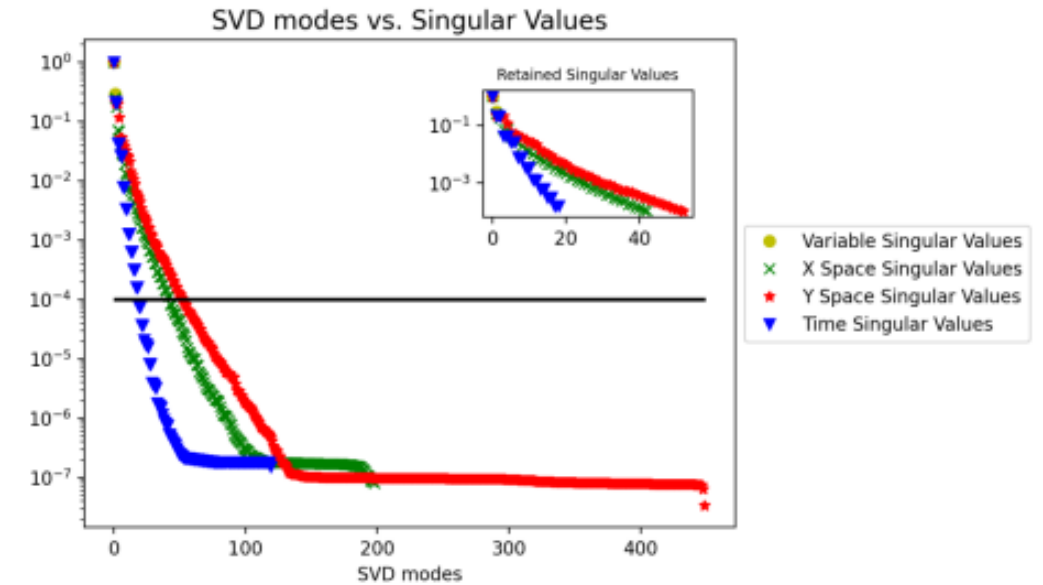
Select an algorithm

☐ SVD

☒ HOSVD

☐ HODMD

Parameter	Default value
SVD tolerance	$1e^{-4}$
Number of snapshots	120



In multilinear algebra, HOSVD of a tensor is a specific orthogonal Tucker decomposition. It may be regarded as one generalization of the matrix singular value decomposition.

# Using ModelFLOWS-app

## Modal Decomposition – Pattern Detection

Higher-Order Dynamic Mode Decomposition, HODMD (I)

Two types: **HODMD** (for matrices, applies SVD) and **Multi-dimensional HODMD** (For tensors, applies HOSVD)

There are also two versions for Multi-dimensional HODMD: Non-iterative and iterative

The screenshot displays the ModelFLOWS-app interface with the following configuration:

- Select an option:** Modal Decomposition
- Select an action:** Pattern detection
- Select an algorithm:** HODMD (selected with a radio button)
- Which type of HODMD would you like to perform:** HODMD (selected from a dropdown menu)
- Which type of HODMD would you like to perform:** Multi-dimensional HODMD (selected from a dropdown menu)
- Select a version:** Non-iterative (selected from a dropdown menu)

The interface also includes a text box with the message: "The Multi-dimensional HODMD has two variants: Non-iterative and iterative."



ModelFLOWS

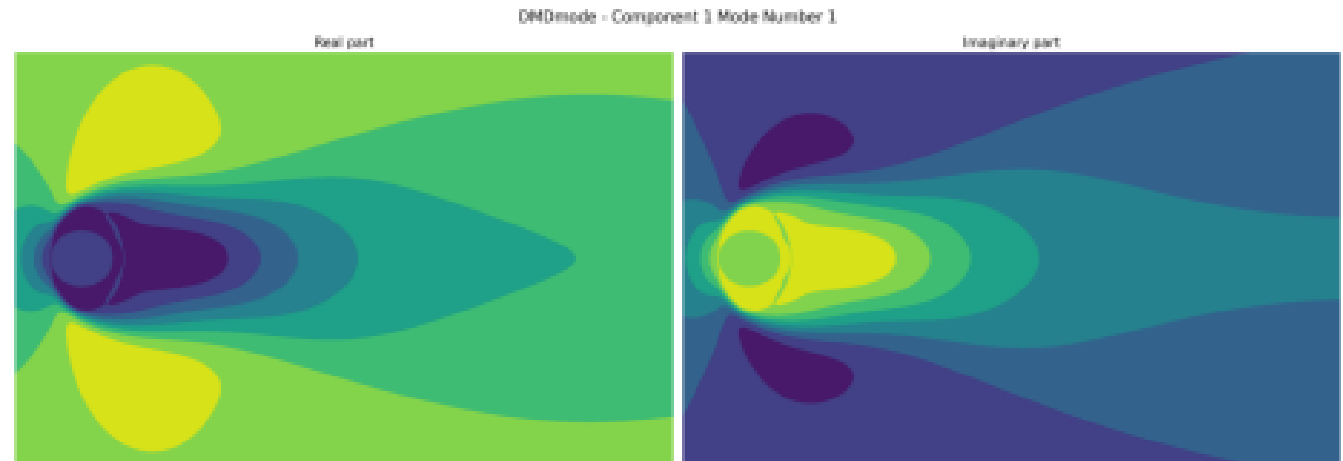
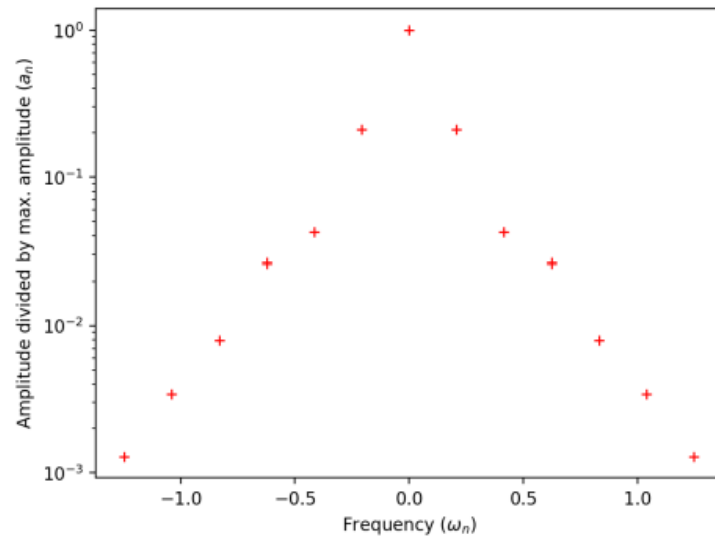


# Using ModelFLOWS-app

## Modal Decomposition – Pattern Detection

### Higher-Order Dynamic Mode Decomposition, HODMD (II)

Parameter	Default value
SVD tolerance	$1e^{-10}$
HODMD tolerance	$1e^{-3}$
Number of windows in HODMD (d)	15
Number of snapshots	151
Time interval	1



HODMD is a data-driven method generally used in fluid dynamics and in the analysis of complex non-linear dynamical systems modeling several complex industrial applications.

# Using ModelFLOWS-app

## Modal Decomposition – Data Reconstruction

### Data Repairing

Select an option

Modal Decomposition ▼

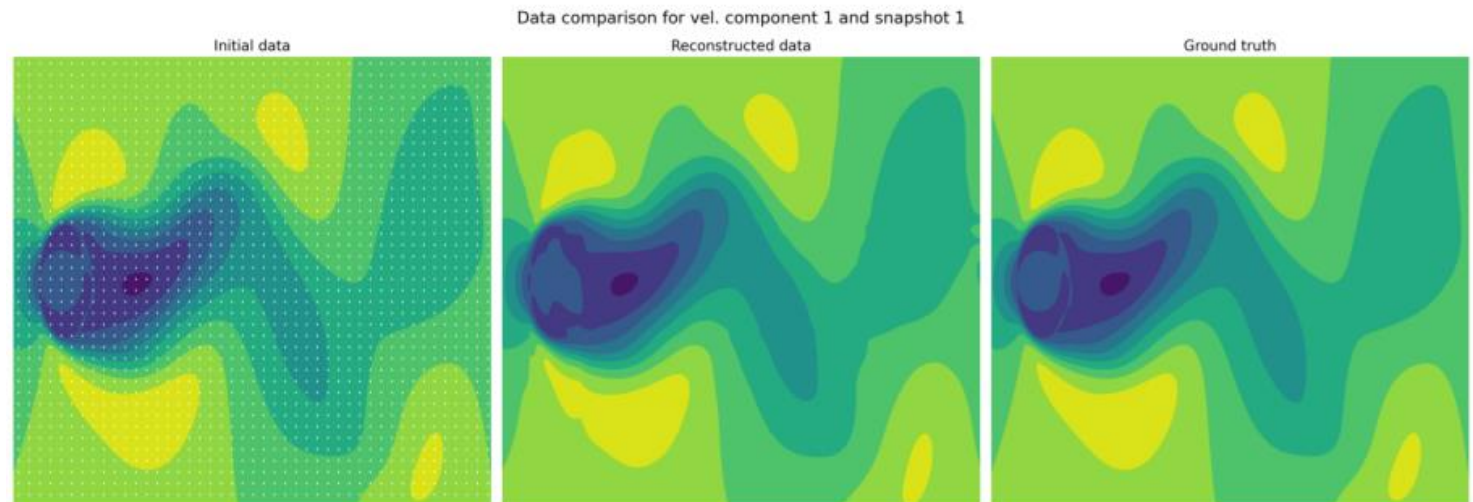
Select an action

Reconstruction ▼

Select an option

Repair

Parameter	Default value
Number of SVD modes to retain	18
Data completion	nearest value interpolation



This module takes the *Gappy\_Tensor\_cylinder\_Re100.mat* tensor, which has missing data, and fills in the gaps using different techniques (interpolation, mean value, zeros) and then HOSVD. The reconstructed data is then compared to *Tensor\_cylinder\_Re100.mat*, which is the ground truth.

# Using ModelFLOWS-app

## Modal Decomposition – Data Reconstruction

### Superresolution (Data Enhancement)

Select an option

Modal Decomposition ▼

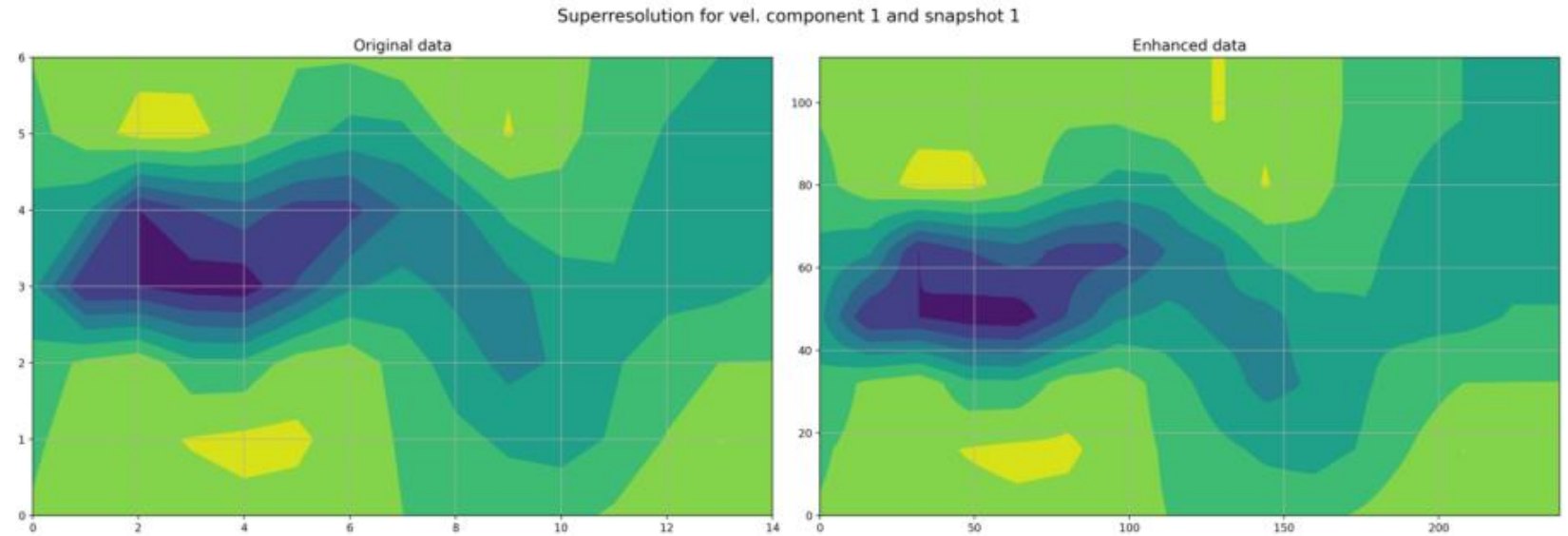
Select an action

Reconstruction ▼

Select an option

Superresolution

Parameter	Default value
Enhancement factor ( $2^{factor}$ )	$factor = 4$



This module takes the *DS\_3D\_Tensor\_cylinder\_Re100.mat* tensor, which is a low resolution database, and expands its spatial dimensions using HOSVD.

# Using ModelFLOWS-app

## Modal Decomposition – Prediction

(Multi-dimensional) Predictive Higher-Order Dynamic Mode Decomposition, (md)HODMD (I)

Select an option

Modal Decomposition ▼

Select an action

Prediction ▼

Which type of HODMD would you like to perform

Predictive HODMD |

Predictive HODMD

Multi-dimensional predictive HODMD

The Multi-dimensional HODMD has two variants: Non-iterative and iterative.


Select a version

Non-iterative

Non-iterative

Iterative

It is also possible to predict future snapshots using HODMD by reconstructing the original tensor setting the desired number of snapshots



ModelFLOWS

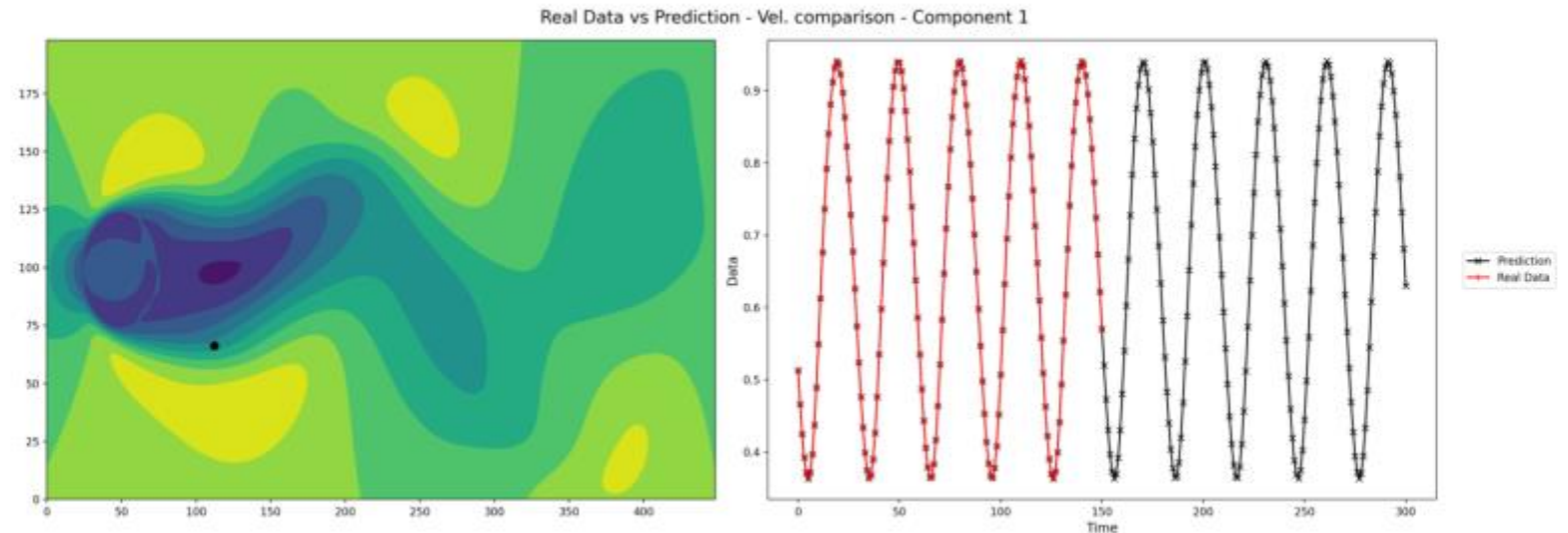
Universidad Politécnica de Madrid

# Using ModelFLOWs-app

## Modal Decomposition – Prediction

(Multi-dimensional) Predictive Higher-Order Dynamic Mode Decomposition, (md)HODMD (II)

Parameter	Default value
SVD tolerance	$1e^{-10}$
HODMD tolerance	$1e^{-3}$
Number of windows in HODMD (d)	15
Number of snapshots	151
Time interval	1
Final snapshot in the prediction interval	300



# Using ModelFLOWS-app

## Deep Learning - Pattern Detection

Autoencoders Neural Network

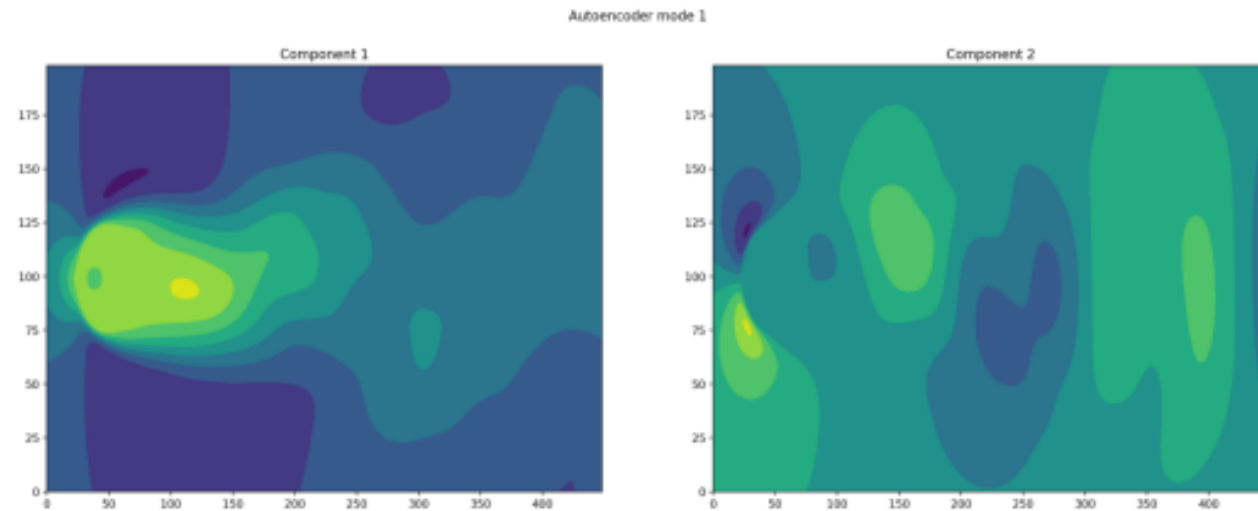
Parameter	Default value
Batch size	64
Training epochs	200
Number of autoencoder dimensions	10

Select an option

Deep Learning ▼

Select an option

Pattern detection ▼



An autoencoder is an unsupervised learning technique for neural networks that learns efficient data representations (encoding) by training the network to ignore signal "noise."

# Using ModelFLOWS-app

## Deep Learning – Data Reconstruction

### Superresolution Neural Network

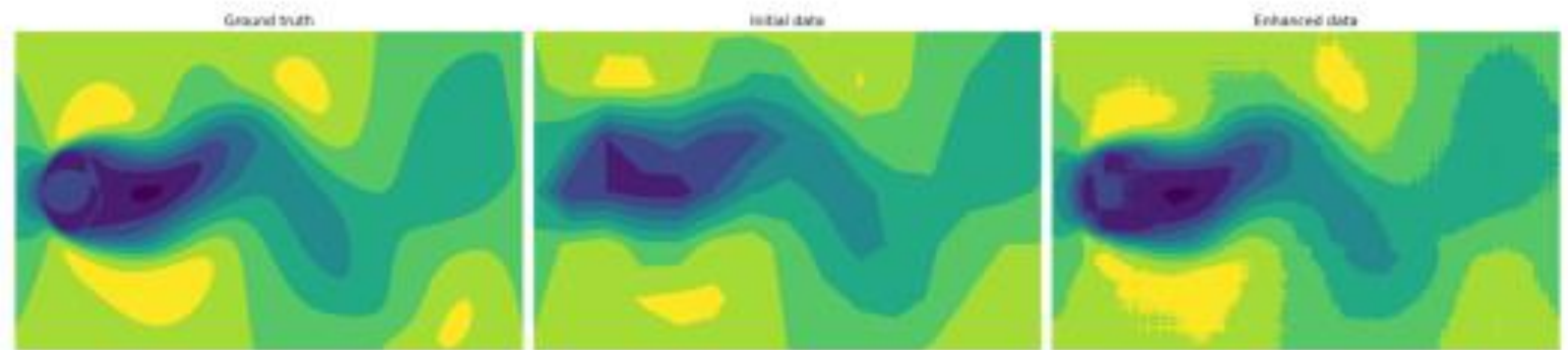
Select an option

Deep Learning ▼

Select an option

Reconstruction ▼

Parameter	Default value
Batch size	16
Neurons per layer	50
Hidden layers activation function	elu
Output layers activation function	tanh
Learning rate	$5e^{-3}$
Training data size	121
Training epochs	150



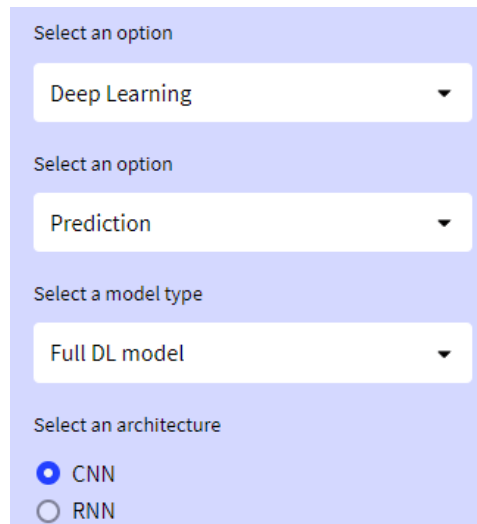
This model inputs 2D (*Tensor\_cylinder\_Re100.mat*) and 3D (*Tensor.pkl*) data. Precisely, the model inputs the downsampled and ground truth versions of each database, enhances the downsampled version's resolution, and then compares it to the ground truth. This model is hybrid, since it combines SVD with DL.

# Using ModelFLOWS-app

## Deep Learning – Prediction

### Predictive Neural Networks

Two model types are available: full deep learning, and hybrid deep learning. This last one applies SVD on the training data during the data preprocessing stage. The used architectures are Convolutional Neural Network (CNN) architecture and Recurrent Neural Network (RNN).



Select an option

Deep Learning ▼

Select an option

Prediction ▼

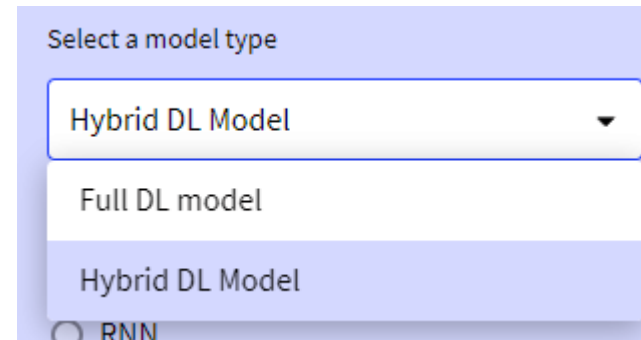
Select a model type

Full DL model ▼

Select an architecture

☒ CNN

☐ RNN



Select a model type

Hybrid DL Model ▼

Full DL model

Hybrid DL Model

☐ RNN



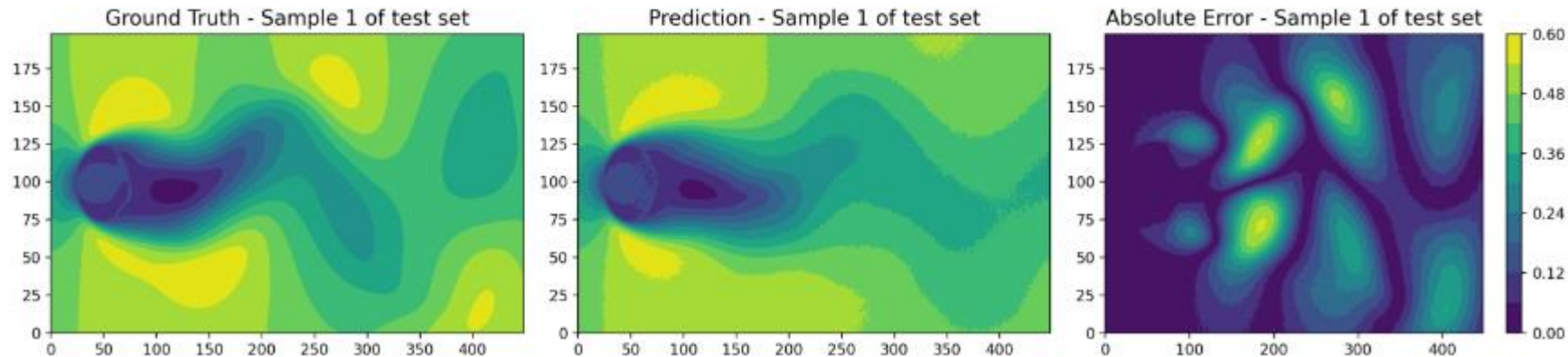
# Using ModelFLOWS-app

## Deep Learning – Prediction

Predictive Neural Networks – Full Deep Learning Model

Parameter	CNN Default value	RNN Default value
Batch size	4	4
Training epochs	3	20
Hidden layers activation function	relu	linear
Output layers activation function	linear	linear
Neurons per layer	N/A	50
Shared dims	30	50
Learning rate	$1e^{-2}$	$1e^{-2}$

## Model Training Results - RNN



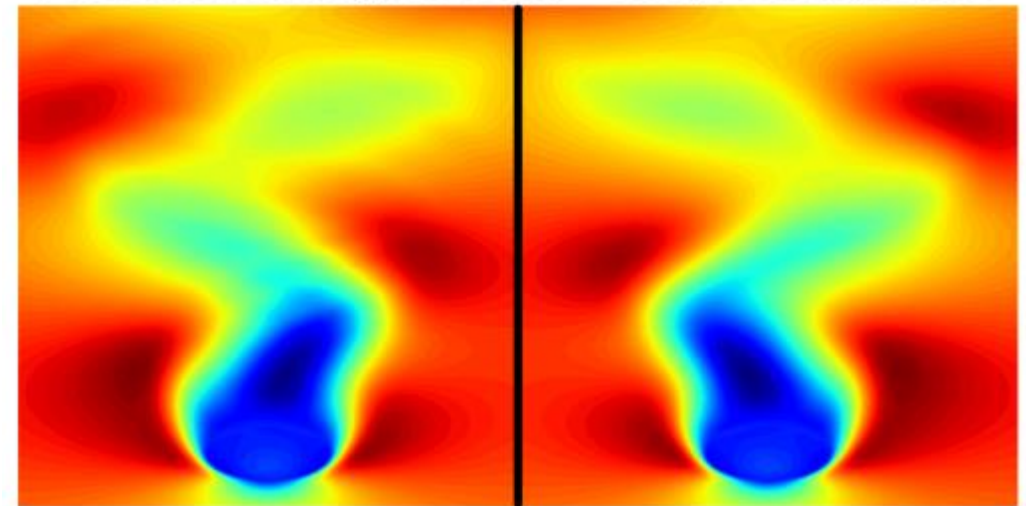
# Using ModelFLOWS-app

## Deep Learning – Prediction

### Predictive Neural Networks – Hybrid Deep Learning Model

Parameter	CNN Default value	RNN Default value
Number of SVD modes to retain	15	15
Batch size	8	8
Training epochs	200	200
Hidden layers activation function	relu	relu
Output layers activation function	tanh	tanh
Neurons per layer	30	30
Shared dims	20	20
Learning rate	$2e^{-3}$	$2e^{-3}$

Prediction vs.Original Data - Comp. 1 Snapshot 2



Two model types are available: full deep learning, and hybrid deep learning. This last one applies SVD on the training data during the data preprocessing stage. The used architectures are Convolutional Neural Network (CNN) architecture and Recurrent Neural Network (RNN).

# Thank you for your attention!

## For any help or suggestions please contact us at:

 [modelflows.mf@gmail.com](mailto:modelflows.mf@gmail.com)

Keep up to date with our work!



[linkedin.com/in/company/modelflows/](https://linkedin.com/in/company/modelflows/)



[github.com/modelflows](https://github.com/modelflows)



ModelFLOWS