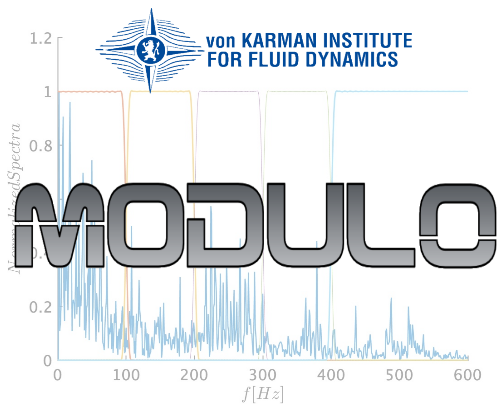
The von Karman Institute For Fluid Dynamics

ENVIRONMENTAL AND APPLIED FLUID DYNAMICS DEPARTMENT

User Manual for the *MODal mULtiscale pOd* (MODULO) Software



Davide Ninni Miguel Alfonso Mendez

March 2020

Contents

Chapter 1

The MODal mULtiscale pOd (MODULO)

MODal mULtiscale pOd (MODULO) is a software developed at the von Karman Institute to perform data-driven decomposition for complex fluid flows. Its main menu is shown in figure 1.1.

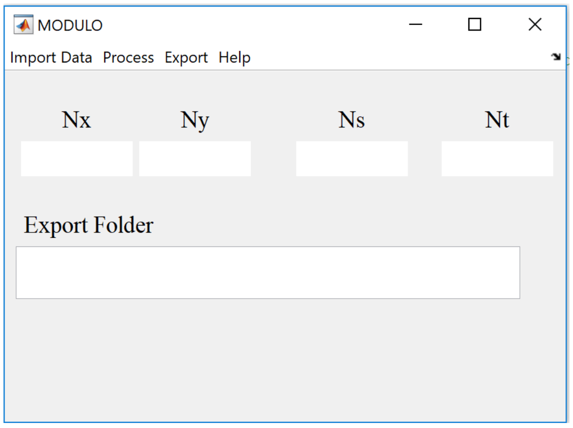


Figure 1.1: Main Menu of MODULO

It is composed by four not editable strings giving information about the mesh and the number of time steps (see section 2) and one editable string for choosing the folder where the results will be saved (see section 4). The upper part of the GUI shows four menus for:

- import data (`Import Data` menu), presented in section 2;
- perform modal analysis (`Process` menu), presented in section 3;
- control plotting and exporting folder (`Export` menu), presented in section 4;
- help about the software (`Help` menu), presented in section 5.

Chapter 2

Import Data

By clicking on **Import Data**, the user can choose if importing 1D or 2D quantity data, as shown in figure 2.1. In particular, in 2D test cases, the quantity can be a scalar (e.g., the vorticity) or a vector (e.g., a velocity field); in 1D test cases, only a scalar quantity can be imported. Before importing data, the user can choose to remove the mean flow by activating the corresponding tick in the popup menu. This option removes the temporal average in each spatial point, very interesting if the dataset is statistically stationary.

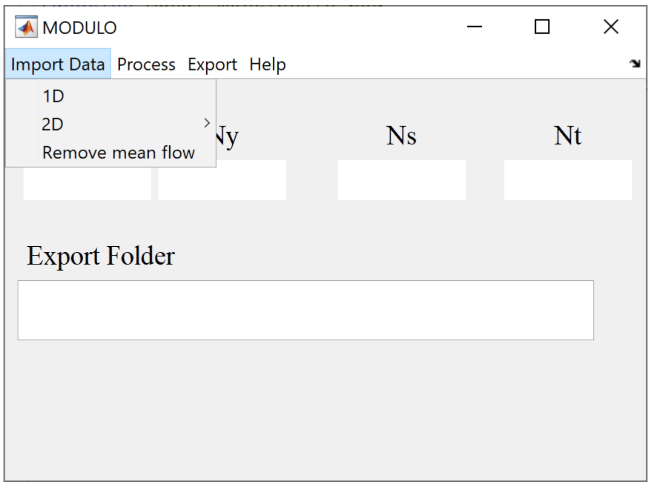


Figure 2.1: Importing data

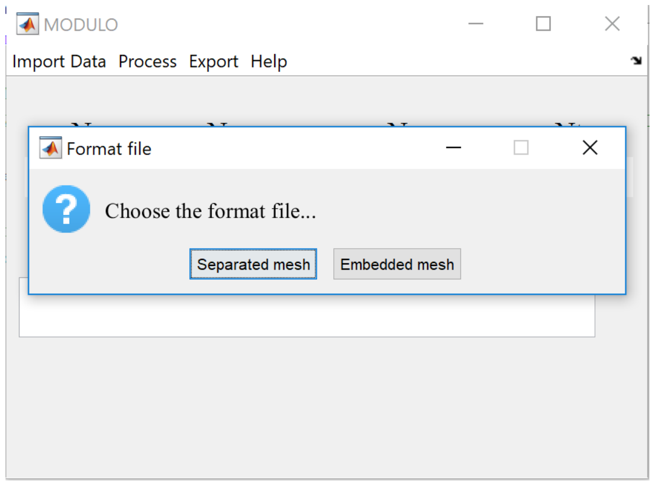
While importing data, the user has to choose a set of files. These files can be organized in two ways:

- one file for each time step, containing both the mesh and the solution (*Embedded mesh*)
- one separate file for the mesh and other files containing only the solution, one for each time step (*Separated mesh*)

This can be chosen from the question dialog appearing immediately after the click on 1D or 2D (scalar or vector). A preview is shown in figure 2.2.

In the current version, MODULO assumes that the mesh is Cartesian and has equal sampling in all directions. Examples of data files accepted by MODULO are shown in the five exercises available in the repository; the user is invited to consult these before proceeding further.

If the *Separated mesh* format is chosen, the user will be asked to select the mesh file first, and then the files containing the solutions. After selecting the files, `Nt` in the main menu will be updated: this is the total number of imported snapshots. After this, an interactive GUI will appear, as explained in section 2.1.

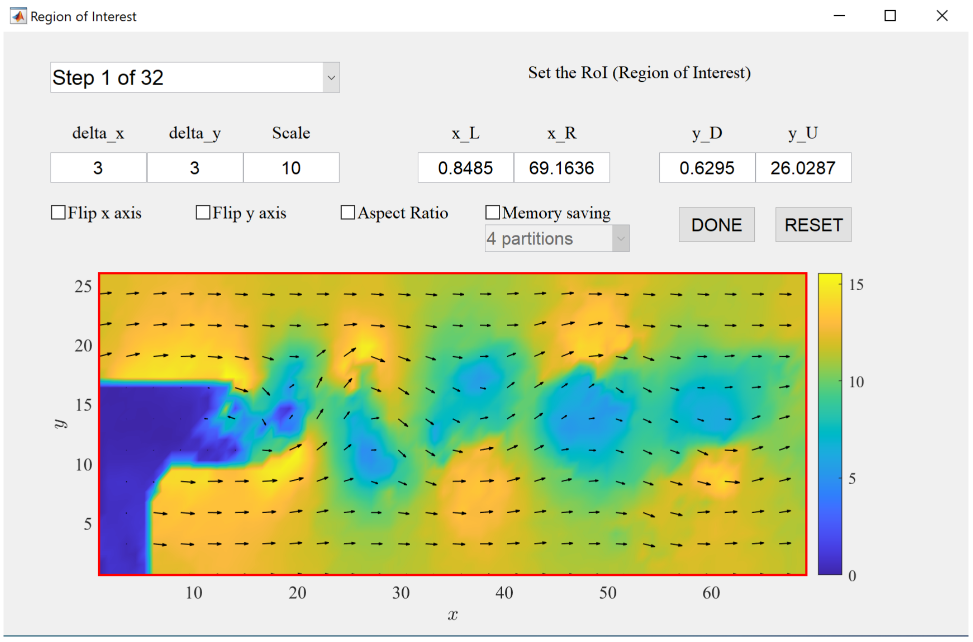


Choose the format file...

Figure 2.2: Question dialog to choose the format of the files

2.1 *Region of Interest* (RoI)

After the user has chosen the files, an interactive GUI appears. A preview is shown in figure 2.3. This window is *modal*: the user must act on it before doing anything else. If this window is closed without selecting a RoI, the importing will be aborted.



Step 1 of 32

Set the RoI (Region of Interest)

delta_x	delta_y	Scale	x_L	x_R	y_D	y_U
3	3	10	0.8485	69.1636	0.6295	26.0287

☐Flip x axis ☐Flip y axis ☐Aspect Ratio ☐Memory saving

4 partitions

Figure 2.3: Region of Interest and plotting parameters.

A 2D vector test case is taken as an example. In this window, the user can decide which time step to plotting, from the pop-up menu on the left upper corner (indicating 'Step 1 of 32'). The plotting window can be zoomed.

The scope of this window is to visualize the data, define a region of interest (RoI) within which the decomposition should be performed, and prepare the plotting of the spatial structures. The RoI can be set by changing the values of `x_L`, `x_R`, `y_D` and `y_U` (thus left, right, bottom and upper limit). Their default values consider the entire domain from the snapshot. A red rectangle will highlight the part of the mesh that will be taken into account. Observe that **in 1D test cases, `y_D` and `y_U` are disabled.**

For plotting purposes, the user can edit the values of `delta_x`, `delta_y` and `Scale`. **These**

are enabled only for 2D vector test cases and are used only for plotting purposes. The entries `delta_x` and `delta_y` allow for controlling the density of the quiver plot, possibly reducing the number of arrows: if $\delta_x = 5$, arrows one arrow over 5 will be plotted in the x direction. The default value for these skipping parameters is 3. By changing `Scale`, the length of the arrows is edited for the same reason (its default value is 10). In this way, the user can easily identify the RoI.

If necessary, the user can flip the axes, by activating the check-boxes `Flip x axis` or `Flip y axis`. By activating `Aspect ratio`, the plot will be re-adapted so that the axes have the same scale.

Observe that, besides the RoI definition, none of the other plotting parameters has any influence on the decomposition. However, the spatial structures produced by the decomposition will be saved with the plotting parameters selected at this stage. Observe that at this step, the functionality `Memory saving` is not taken into account: at this stage, only the snapshot selected in the upper left pop-up menu is loaded. The `Memory saving` functionality is described in 2.1.1.

On the right, two buttons are available. The user can reset the RoI to the entire mesh through `RESET` button. By clicking on `DONE`, the matrix D will be constructed. A wait-bar will indicate the progress of the computation, as shown in figure 2.4.

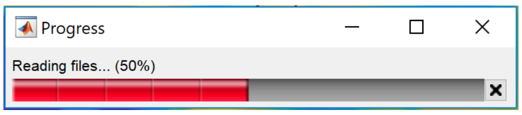
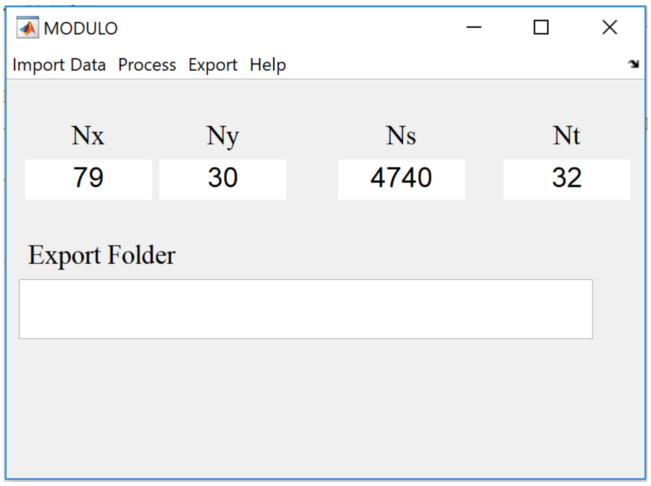


Figure 2.4: Progress of the construction of D matrix

Once D is prepared, the user is addressed back to the main menu, in which the values of `Nx`, `Ny` and `Ns` are updated according to the RoI chosen by the user, as shown in figure 2.5. `Nt` is the number of time steps, available before the definition of the RoI.



Nx	Ny	Ns	Nt
79	30	4740	32

Figure 2.5: Update of the main menu after the user has chosen the RoI

In particular, one should notice that:

- $N_s = 2 \cdot N_x \cdot N_y$ in 2D vector test cases (as in figure 2.5)
- $N_s = N_x \cdot N_y$ in 2D scalar test cases
- $N_s = N_x$ and $N_y = 0$ in 1D test cases

At this stage, the user can choose the exporting folder and perform the decomposition. These functionalities are explained in next sections.

2.1.1 Memory Saving

`Memory saving` is a checkbox that allows for saving memory during the computation, at the cost of increasing the computational time. The data matrix $\mathbf{D}$ can sometimes be too large, especially in 2D vector test cases, for machines that have less than 16GB of RAM. To better understanding the sense of this functionality, an example is presented here.

Consider a 2D vector test case with $n_t = 13566$ time steps, with $n_x = 79$, $n_y = 30$ (the RoI corresponds to the entire spatial domain) one has $n_s = 4740$. Hence the size of $\mathbf{D}$ is 4740×13566. A single element of this matrix occupies 8 bytes, so the entire matrix occupies 0.514 GB. With a finer mesh, this can easily reach 1 GB. Moreover, the `Maximum possible array` (see section 5) depends on the tasks running on the machine at that moment. If one adds the computation of the matrices involved in the factorizations, it is easy to see that the decomposition can demand too much memory to the machine; thus, a memory saving strategy has been implemented.

The main objective of this functionality is to by-pass the preparation of the data matrix $\mathbf{D}$: all the operations involving its columns (snapshots) or rows (time series) are performed by reading the files *during* the calculations. This brings the memory requirements to the minimum but increases the computational time. In order to accelerate this process, it is possible to introduce a certain number of partitions, defining the number of files that will be read and stored in memory during this process.

Chapter 3

Process

All the decompositions available in MODULO are here briefly presented. A more detailed review of the theoretical background is provided in the first three video tutorials. For more background on the theoretical derivations, and for some experimental and numerical test cases, the reader is referred to [1–5].

3.1 Proper Orthogonal Decomposition (POD)

Let $\mathbf{D}(x_i, t_k) = \mathbf{D}[i,k] \in \mathbb{R}^{n_s \times n_t}$ be a data matrix. Each column represents a 'snapshot' at a certain time step, in the domain of interest:

$$\mathbf{D} = \begin{bmatrix} d_1[1] & \cdots & d_1[k] & \cdots & d_1[n_t] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ d_{n_s}[1] & \cdots & d_{n_s}[k] & \cdots & d_{n_s}[n_t] \end{bmatrix} \tag{3.1}$$

Performing the *Proper Orthogonal Decomposition* (POD) of data that is uniformly sampled in space and time is equivalent to performing a *Singular Value Decomposition* (SVD) of D, which it is here denoted as:

$$\mathbf{D} = \sum_{r=1}^{rk(\mathbf{D})} \sigma_r \, \phi_r \, \psi_r^T = \Phi_P \, \Sigma_P \, \Psi_P^T \tag{3.2}$$

where $rk(\mathbf{D})$ is the rank of D, and both Φ_P and Ψ_P are orthogonal and orthonormal, hence $\Phi_P^T \Phi_P = \mathbf{I} \in \mathbb{R}^{n_s \times n_s}$ and $\Psi_P^T \Psi_P = \mathbf{I} \in \mathbb{R}^{n_t \times n_t}$. The POD is derived under the constraints of error minimization (or, equivalent, amplitude maximization), i.e. the approximation constructed by including the leading R modes minimizes the associated l2 error, i.e.

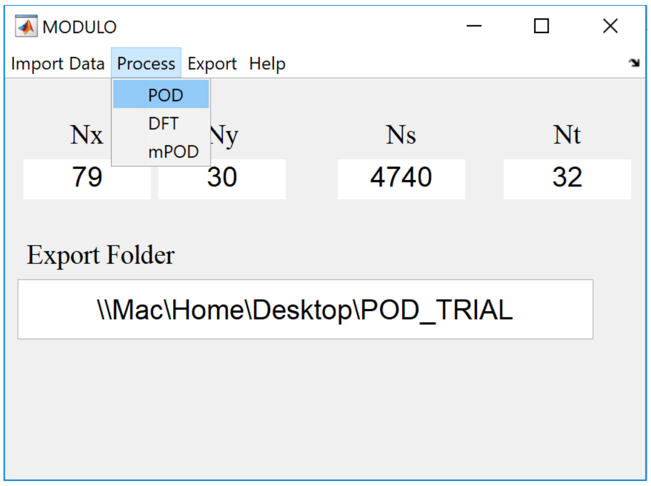
$$\text{Given} \quad \tilde{\mathbf{D}} = \sum_{r=1}^{R} \sigma_r \, \phi_r \, \psi_r^T = \tilde{\Phi}_P \, \tilde{\Sigma}_P \, \tilde{\Psi}_P^T \quad \text{such that} \quad min||\mathbf{D} - \tilde{\mathbf{D}}|| \tag{3.3}$$

The POD algorithm implemented in MODULO is the popular snapshot POD (see video tutorial two and [3] for more background). The first step consists in computing the temporal correlation matrix $\mathbf{K} = \mathbf{D}^T \mathbf{D}$. The temporal structures are the eigenvectors of this matrix. Hence the second step consists of solving the eigenvalue problem $\mathbf{K} = \Psi_P \Sigma_P^2 \Psi_P^T$. Finally, as for all the decompositions implemented in MODULO, the last step is the data projection to compute the spatial structures, that is

$$\Phi_P \, \Sigma_P = \mathbf{D} \, \Psi_P \rightarrow \Phi_P = \mathbf{D} \, \Psi_P \, \Sigma_P^{-1} \tag{3.4}$$

In the case of the POD, the matrix of amplitudes Σ_P is known from the eigenvalue decomposition [3]. In the DFT or the mPOD, this matrix is computed by normalizing all the columns of $\mathbf{D}\,\Psi$.

To plot the temporal structures and to study their frequency content, a sampling frequency has to be input by the user, as shown in figure 3.2. An input dialog will appear as soon as the POD is chosen in the menu `Process`. **It is important to choose an exporting folder before performing any decomposition**. This can be done manually or from the menu `Export`. Moreover, the user can select a limited range of modes to be exported, as shown in figure 3.3. Of course, these quantities have to be positive and refer to modes that exist. Thus, an error will occur if `r2` (or `r1`) is bigger than the number of columns of the temporal basis or if `r1` is smaller than one.

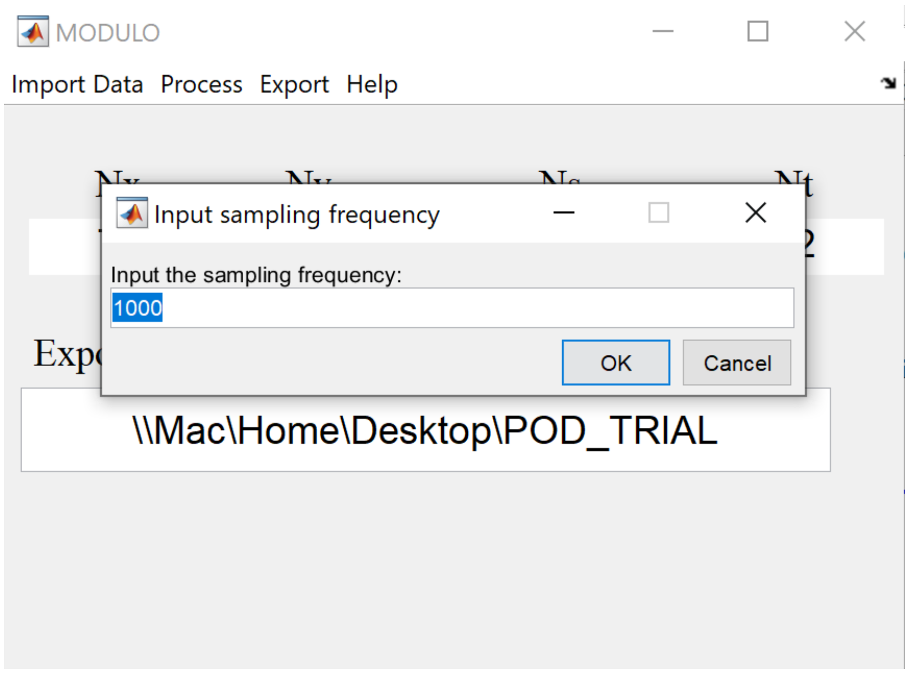


Nx	Ny	Ns	Nt
79	30	4740	32

Export Folder

\\Mac\Home\Desktop\POD_TRIAL

Figure 3.1: Process POD



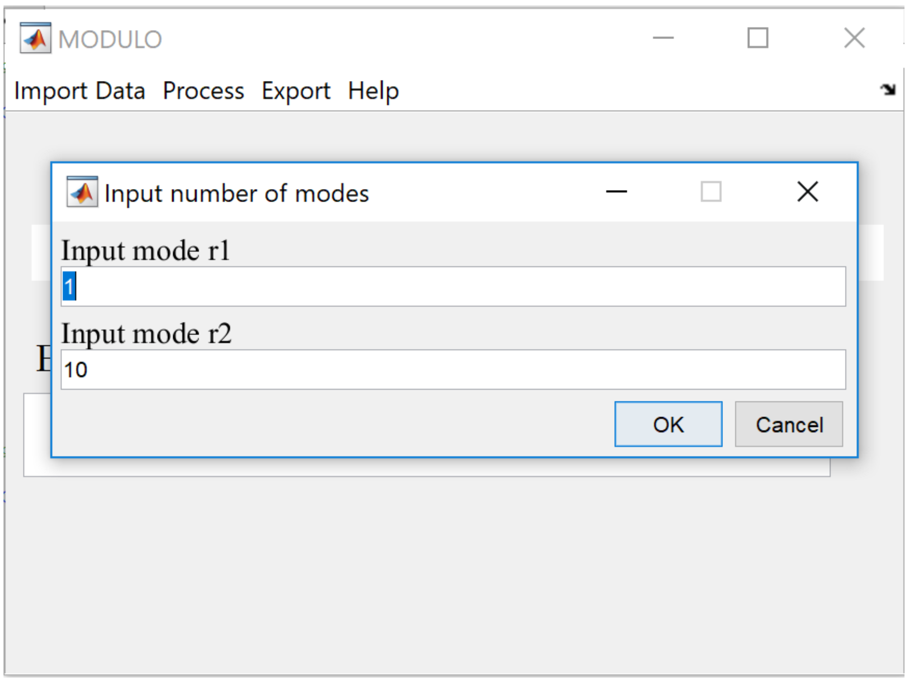
Input the sampling frequency:

1000

\\Mac\Home\Desktop\POD_TRIAL

Figure 3.2: Input dialog for the sampling frequency

Once the sampling frequency and the range of modes have been introduced, the plotting part will start. The structures that will be plot are consistent with the choice of the user in the menu `Export`. At the end of the decomposition, excel files containing the mesh and the modes



Input mode r1

1

Input mode r2

10

Figure 3.3: Input range of modes to be exported

will be saved in the exporting folder chosen by the user, *regardless of the choice of plot*. If it is a 2D test case, the modes will be exported as matrices to be consistent with the mesh format.

3.1.1 POD with memory saving

The time correlation matrix $\mathbf{K}$ is computed as $\mathbf{K} = \mathbf{D}^T\,\mathbf{D}$ and can therefore be constructed as a series of scalar products between the rows of $\mathbf{D}$. If the memory saving option is chosen, this construction is performed without preparing $\mathbf{D}$. In this case, MODULO reads the data from the files (for each time step, in each space point) in order to build two rows of $\mathbf{D}$, thus d_i and d_j, and finally the related inner product. Since this matrix is symmetric, i.e.. $\mathbf{K}(j,i) = \mathbf{K}(i,j)$, only half of it is actually computed. Therefore the for loop becomes faster and faster as the calculation proceeds (see video tutorial 8).

To speed up the computation while leveraging only marginally on the computer memory, the calculation is divided into partitions, and the inner product is computed in groups of vectors D_i and D_j. These groups are small matrices of size $dimension \times n_t$, where $dimension = n_t/partitions$. Once the correlation matrix is ready and its eigendecomposition performed, the final step requiring the definition of the dataset matrix $\mathbf{D}$ is the projection to compute the spatial structures (see equation 3.4). If the memory saving is active, MODULO reads from each of the files to build the rows of $\mathbf{D}$ and, for each mode, it computes the spatial structures using the inner product with the corresponding temporal structure:

$$\Phi_P[i,r] = \frac{D[i,:]\,\Psi_P[:,r]}{\Sigma[r,r]}\,, \tag{3.5}$$

where r is the corresponding mode and i is the spatial point.

3.2 Discrete Fourier Transform (DFT)

The Discrete Fourier Transform (DFT) is not a data-driven decomposition and its temporal structures are computed a priori, regardless of the data at hand, from the sampling frequency.

In particular, the temporal basis is provided by the well known Fourier Matrix:

$$\Psi_F = \frac{1}{\sqrt{n_t}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & w & w^2 & \cdots & w^{n_t-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & w^{n_t-1} & w^{2\,(n_t-1)} & \cdots & w^{(n_t-1)^2} \end{bmatrix} \tag{3.6}$$

where $w = e^{2\pi\, i/n_t}$ with $i = \sqrt{-1}$. This matrix can be computed using the Fast Fourier Transform, through Matlab function `fft`, of the identiy matrix and then conjugate and normalizing by $\sqrt{n_t}$. Once this matrix is given, the spatial structures can be computed as:

$$\Phi_F\, \Sigma_F = \mathbf{D}\, \overline{\Psi}_F \rightarrow \Phi_F = \mathbf{D}\, \overline{\Psi}_F\, \Sigma_F^{-1} \tag{3.7}$$

where $\overline{\Psi}_F$ denotes the conjugate of Ψ_F, which is also its inverse since this matrix is also symmetric ($\Psi_F^T = \Psi_F$). Note that the multiplication by $\overline{\Psi}_F$ is equivalent to computing the FFT. Another essential difference with respect to the POD is that the amplitudes are not known before the projection step. Hence these have to be computed via normalization, that is $\sigma_{Fr} = ||\Phi_F[:, r]\, \sigma_{Fr}|| = ||D\overline{\psi}_{Fr}||$. Finally, the obtained spatial structures are complex. Thus, for plotting purposes, only the real part is separated from the imaginary part, while the output contains both in different sheets of the excel file produced. For what concerns the exporting part, as explained in section 3.1, the user has to input a sampling frequency and a range of modes.

3.2.1 DFT with memory saving

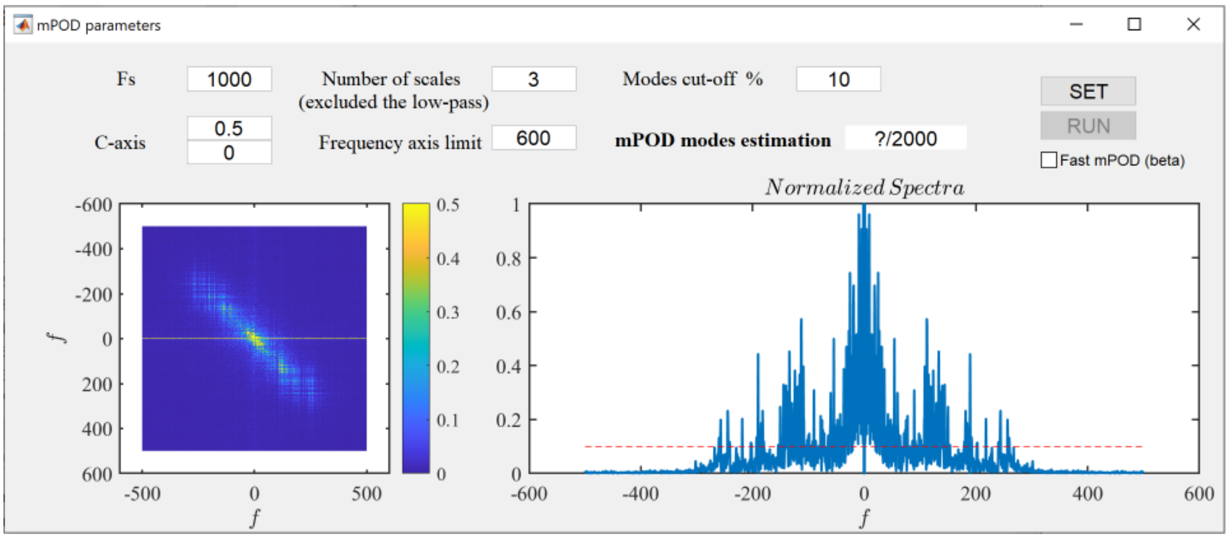
As for the POD, the memory saving option allows computing the DFT without assembling first $\mathbf{D}$. The key difference is that the DFT does not require time correlation matrix $\mathbf{K}$. Hence the memory saving calculates the spatial structures by performing a set of FFTs in each spatial location and then assemblies the spatial structures by normalizing the results for each of the available frequency bins.

3.3 Multiscale Proper Orthogonal Decomposition (mPOD)

The Multiscale Proper Orthogonal Decomposition (mPOD) is an extension of the POD that allows deriving modes that are optimal within a certain range of frequencies. These ranges of frequencies are called scales. The concept behind this algorithm is to perform POD at different scales independently. This allows for constraining the POD optimality within non-overlapping ranges of frequencies. As shown in [3], the mPOD requires a frequency splitting vector to define the scales of interest (see also video tutorial 3). This input is used to perform a Multiresolution Analysis (MRA) of the temporal correlation matrix $\mathbf{K}$ using an appropriate filter bank. Once the correlation matrix is split into the contribution of different scales, each of these is diagonalized as in the classical POD: in this way, orthonormal bases are obtained for each scale. In the last step of the algorithm, the bases of each scale are merged into the final basis Ψ_M to perform the final decomposition $\Phi_M\, \Sigma_M = \mathbf{D}\, \Psi_M$.

After choosing mPOD, the window in figure 3.4 appears.

This window allows the user for setting the mPOD parameters. In the upper left corner, the user introduces the sampling frequency F_s, the number of intermediate (that is excluding the largest, as described below) scales (here indicated by M, from 1 to 10) and the cut-off percentage (at most 50%). This percentage indicates the number of modes that will be selected from each scales to assembly the final mPOD basis. When this is set to 50%, then in each scale only the modes that have at least half of the energy of the leading one will be retained. This


mPOD parameters

Fs 1000

Number of scales (excluded the low-pass) 3

Modes cut-off % 10

C-axis 0.5 0

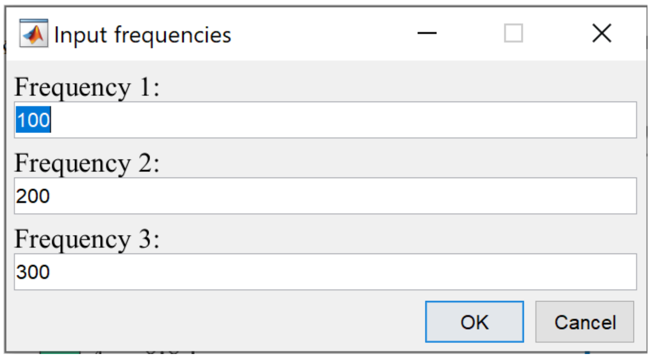
Frequency axis limit 600

mPOD modes estimation ?/2000

Figure 3.4: Parameters for preforming mPOD

allows for limiting the computational cost. `C-axis` and `Frequency axis limit` are plotting parameters; the user is invited to play with these.

The sampling frequency is necessary to build the frequency vector, hence the normalized spectra. As for the number of scales, this number is introduced **without counting the lowest scale**. Therefore, assume that one is interested in four scales, for example between $[0-100]Hz$, $[100-200]Hz$, $[200-300]Hz$ and $[300-Fs/2]Hz$. Then, the number to introduce here is 3. This is also the length of the frequency splitting vector, which in this case will be $F_V = [100, 200, 300]Hz$. This splitting vector is introduced via an input dialog window as the one shown in figure 3.5. This appears after clicking on `SET` button on the right.


Input frequencies

Frequency 1: 100

Frequency 2: 200

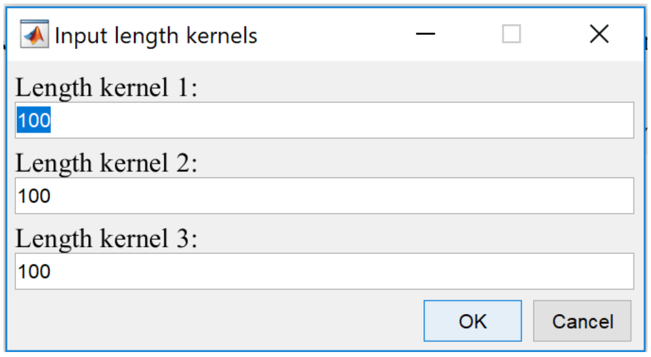
Frequency 3: 300

Figure 3.5: Input dialog for the frequency splitting vector

It is important that all these values are positive and sorted. Moreover, to avoid potential pitfalls due to spectral leakage, the last value is constrained to be at most $Fs/2 - floor(n_t/5) * \Delta f$, where $\Delta f = Fs/n_t$ is the frequency resolution. The second input dialog appearing after pressing `OK` allows for introducing the length of the FIR filter kernels, as shown in figure 3.6. These have to be positive and integer numbers.

Finally, after pressing `OK`, the last dialog box allows for introducing the keep vector (see figure 3.7). This indicates the intermediate scales that will be kept. If the keeping vector is set to a set of zeros, then only the large scale will be kept. In the previous example, if the keep vector is set to $[1, 1, 0]$, then the highest scale $[300 - Fs/2]Hz$ is removed from the decomposition. This vector allows for using the mPOD also as a powerful filter.

At this stage, after having introduced all the mPOD parameters, the software will construct the kernels on the spectra and the `RUN` button will be enabled. In the plotting window, MOD-

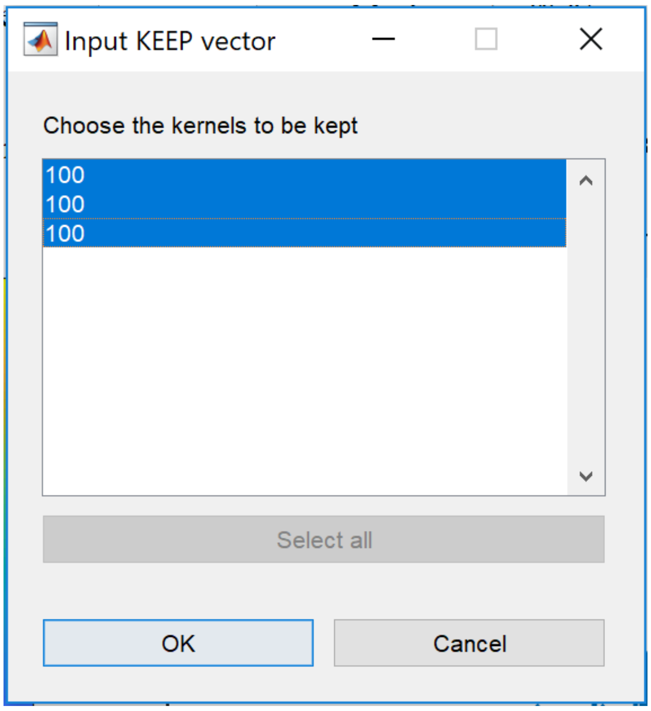

Input length kernels

Length kernel 1:
100

Length kernel 2:
100

Length kernel 3:
100

Figure 3.6: Input dialog for the kernels


Input KEEP vector

Choose the kernels to be kept

100
100
100

Figure 3.7: Input dialog for the keep vector

ULO will display the absolute value of the transfer function of each of the filters designed to isolate the various scales.

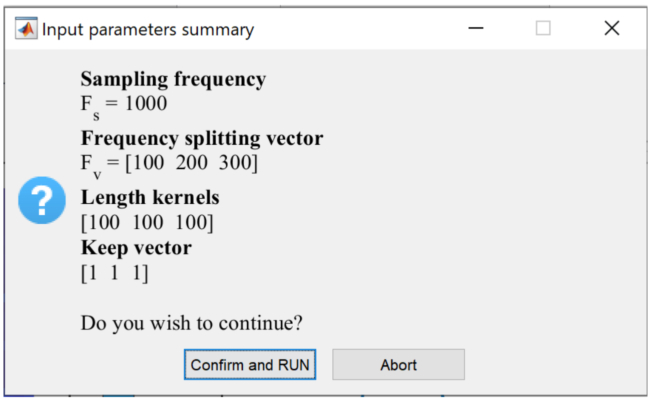
It is worth underlining that, in order to be consistent, if the user modifies the sampling frequency or the number of scales, it will be no longer possible to run the decomposition, since changing F_s or M all the other parameters have to be re-introduced.

At this point the user can modify the cut-off percentage to reduce the number of computed modes. This percentage can be at most 50%. While running the decomposition, a confirm dialog box will summarize the parameters before going (figure 3.8). Then, a multi waitbar will give information about the progress of the decomposition (figure 3.9).

As usual, once the decomposition is performed, the user can introduce the range of modes to be exported.

3.3.1 mPOD with memory saving

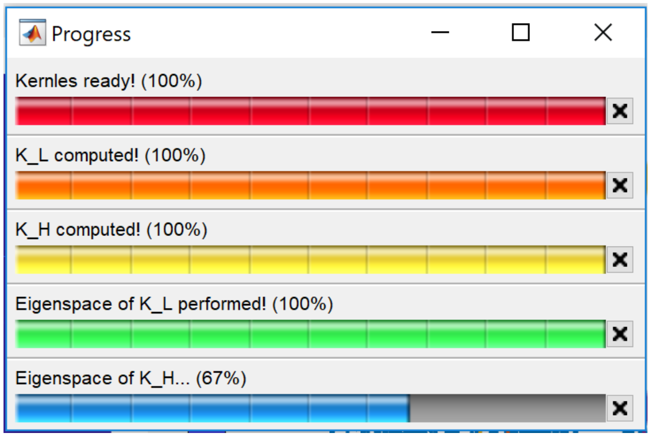
The mPOD algorithm with memory saving is very similar to the POD and the DFT. Similarly to the POD,the temporal correlation matrix is computed by running inner products over time



Sampling frequency
$F_s = 1000$
Frequency splitting vector
$F_v = [100\ \ 200\ \ 300]$
Length kernels
[100 100 100]
Keep vector
[1 1 1]

Do you wish to continue?

Figure 3.8: Summary of mPOD parameters



Kernles ready! (100%)
K_L computed! (100%)
K_H computed! (100%)
Eigenspace of K_L performed! (100%)
Eigenspace of K_H... (67%)

Figure 3.9: Decomposition progress

sequences read from the data. Similarly to the DFT, the projection and the normalization step is performed over when computing the spatial structures.

Chapter 4

Export

By clicking on **Export**, the user can decide to plot or not the computed modes, as it is shown in figure 4.1.

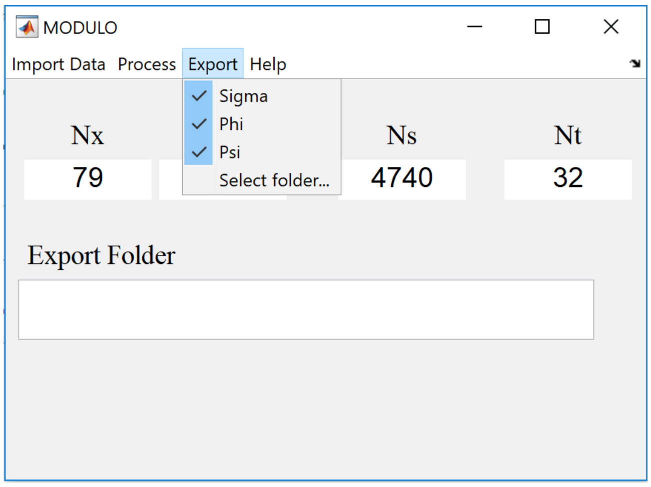


Figure 4.1: Exporting of the modes

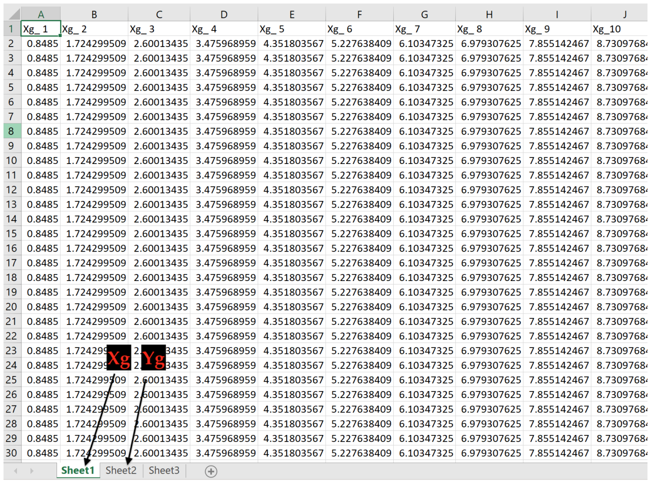
This popup menu also allows for selecting the exporting folder, where the pictures (.png format) will be saved. The path of the folder can also be edited manually from the dedicated string. *If the folder does not exist, the software will create it.* Furthermore, .xlsx files containing the data of the modes will be created by the software whether the plot is active or not. It is worth underlining that if a plot is disabled, the corresponding figure will not be saved. Nevertheless, the .xlsx files will be saved anyway.

If no folder is selected, no decomposition method can be performed.

The excel file containing the mesh is organized in two sheets (for 2D test cases). The first one contains all the `Xg` points (as a matrix) while the second one contains all the `Yg` points (as matrix), as shown in figure 4.2. These are clearly highly redundant, but the rationale behind this file is to allow direct import and plotting with no need to reconstruct the grid (e.g., using the `meshgrid` function from Matlab).

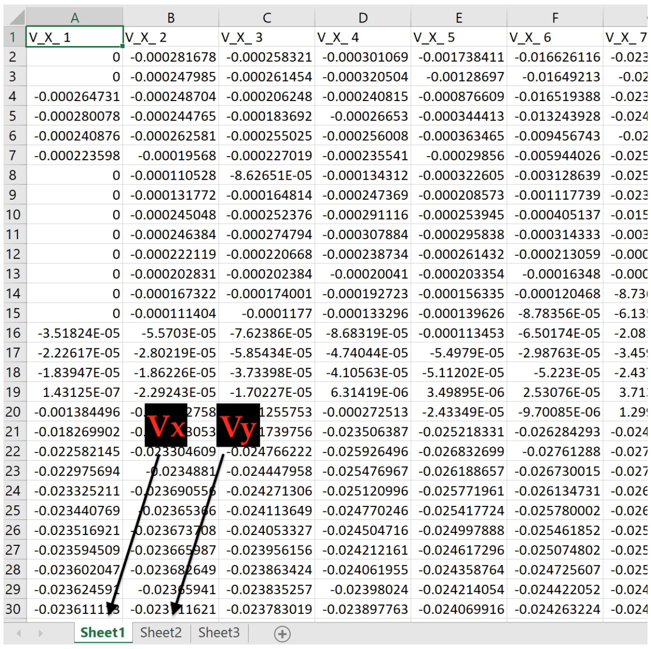
The excel file containing the spatial and temporal structures will be renamed following the number of the exported mode. The spatial structures are organized as matrices in both 2D vector and scalar test cases. In the first case, the modes will be split in two sheets for the two components, as shown in figure 4.3. Each column of the matrix gives indication about the evolution of the quantity along y axis, in a certain x point.

In 1D test cases, both the mesh and the spatial structures are organized as column vectors. The temporal structures will be print in two sheets only, as column vectors. In the first sheet



	A	B	C	D	E	F	G	H	I	J
1	Xg_ 1	Xg_ 2	Xg_ 3	Xg_ 4	Xg_ 5	Xg_ 6	Xg_ 7	Xg_ 8	Xg_ 9	Xg_10
2	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
3	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
4	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
5	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
6	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
7	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
8	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
9	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
10	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
11	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
12	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
13	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
14	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
15	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
16	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
17	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
18	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
19	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
20	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
21	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
22	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
23	0.8485	1.72429[illegible]	2.[illegible]3435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
24	0.8485	1.72429[illegible]	2.[illegible]3435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
25	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
26	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
27	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
28	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
29	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768
30	0.8485	1.724299509	2.60013435	3.475968959	4.351803567	5.227638409	6.10347325	6.979307625	7.855142467	8.7309768

Figure 4.2: Exporting mesh file



	A	B	C	D	E	F	G
1	V_X_ 1	V_X_ 2	V_X_ 3	V_X_ 4	V_X_ 5	V_X_ 6	V_X_ 7
2	0	-0.000281678	-0.000258321	-0.000301069	-0.001738411	-0.016626116	-0.023
3	0	-0.000247985	-0.000261454	-0.000320504	-0.00128697	-0.01649213	-0.02
4	-0.000264731	-0.000248704	-0.000206248	-0.000240815	-0.000876609	-0.016519388	-0.023
5	-0.000280078	-0.000244765	-0.000183692	-0.00026653	-0.000344413	-0.013243928	-0.024
6	-0.000240876	-0.000262581	-0.000255025	-0.000256008	-0.000363465	-0.009456743	-0.02
7	-0.000223598	-0.00019568	-0.000227019	-0.000235541	-0.00029856	-0.005944026	-0.025
8	0	-0.000110528	-8.62651E-05	-0.000134312	-0.000322605	-0.003128639	-0.025
9	0	-0.000131772	-0.000164814	-0.000247369	-0.000208573	-0.001117739	-0.023
10	0	-0.000245048	-0.000252376	-0.000291116	-0.000253945	-0.000405137	-0.015
11	0	-0.000246384	-0.000274794	-0.000307884	-0.000295838	-0.000314333	-0.003
12	0	-0.000222119	-0.000220668	-0.000238734	-0.000261432	-0.000213059	-0.000
13	0	-0.000202831	-0.000202384	-0.00020041	-0.000203354	-0.00016348	-0.000
14	0	-0.000167322	-0.000174001	-0.000192723	-0.000156335	-0.000120468	-8.73
15	0	-0.000111404	-0.0001177	-0.000133296	-0.000139626	-8.78356E-05	-6.13
16	-3.51824E-05	-5.5703E-05	-7.62386E-05	-8.68319E-05	-0.000113453	-6.50174E-05	-2.08
17	-2.22617E-05	-2.80219E-05	-5.85434E-05	-4.74044E-05	-5.4979E-05	-2.98763E-05	-3.45
18	-1.83947E-05	-1.86226E-05	-3.73398E-05	-4.10563E-05	-5.11202E-05	-5.223E-05	-2.43
19	1.43125E-07	-2.29243E-05	-1.70227E-05	6.31419E-06	3.49895E-06	2.53076E-05	3.71
20	-0.001384496	-0.[illegible]2758	[illegible]1255753	-0.000272513	-2.43349E-05	-9.70085E-06	1.29
21	-0.018269902	-0.[illegible]3053	[illegible]1739756	-0.023506387	-0.025218331	-0.026284293	-0.024
22	-0.022582145	-0.023304609	-0.024766222	-0.025926496	-0.026832699	-0.02761288	-0.027
23	-0.022975694	-0.0234881	-0.024447958	-0.025476967	-0.026188657	-0.026730015	-0.027
24	-0.023325211	-0.023690556	-0.024271306	-0.025120996	-0.025771961	-0.026134731	-0.026
25	-0.023440769	-0.02365366	-0.024113649	-0.024770246	-0.025417724	-0.025780002	-0.026
26	-0.023516921	-0.023673708	-0.024053327	-0.024504716	-0.024997888	-0.025461852	-0.025
27	-0.023594509	-0.023665987	-0.023956156	-0.024212161	-0.024617296	-0.025074802	-0.025
28	-0.023602047	-0.023682649	-0.023863424	-0.024061955	-0.024358764	-0.024725607	-0.025
29	-0.023624591	-0.02365941	-0.023835257	-0.02398024	-0.024214054	-0.024422052	-0.024
30	-0.0236111[illegible]3	-0.023[illegible]11621	-0.023783019	-0.023897763	-0.024069916	-0.024263224	-0.024

Figure 4.3: Exporting spatial structures

there will be the temporal basis (in time domain), while in the second one there will be the projection of them in the frequency domain.

It is worth highlighting that the exported sigmas are normalized by $\sqrt{n_s \times n_t}$: in this way their values do not depend on the number of points in space and on the number of time steps.

4.1 Exporting part for DFT

4.1.1 Exporting σ

In the case of a Discrete Fourier Transform, the temporal basis are complex harmonics. Hence, it is more interesting to plot the spectra of the σ in the frequency domain. For this reason, **only for DFT algorithm**, the temporal basis and their projections in the frequency domain are not exported. On the other hand, two different plots of the amplitudes *sigma* are exported in two sheets. In the first, the amplitudes are sorted in descending order. In the second, these are paired with the frequency bin associated to the corresponding mode.

4.1.2 ***Real*** and ***Imag*** component

As recalled in section 3.2, the spatial structures computed through DFT algorithm are complex. The real and imaginary parts are exported in the same excel file. In particular, **only for DFT decomposition**, the spatial structures will be organized in more sheets as follows:

- 4 sheets for 2D vector test cases (2 for the real parts of the components and 2 for the imaginary parts of the components)
- 2 sheets for 2D scalar or 1D test cases (1 for the real part and 1 for the imaginary part of the quantity)

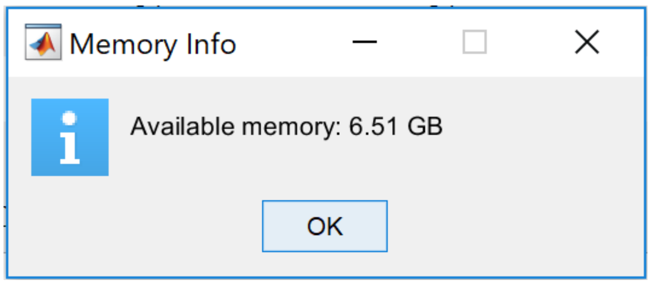
Chapter 5

Help

By clicking on `Help` the user can access to information about the software (`About`) or to the memory info.

5.1 Memory Info

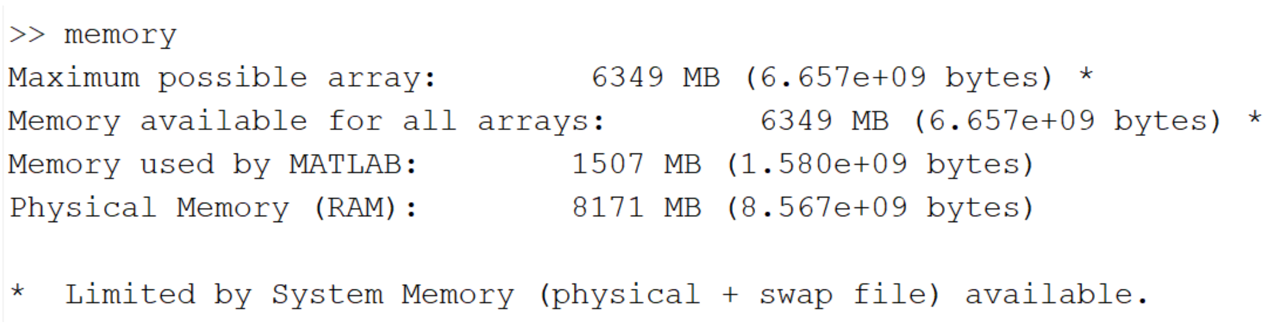
`Memory Info` gives information about the available memory for the computation, as shown in figure 5.1.



Available memory: 6.51 GB

Figure 5.1: Memory info

Note that this does not correspond to the total RAM memory, but to the `Maximum possible array` memory, as computed by Matlab. Running the command `memory` (only for Windows) in the command window, the information summarized in figure 5.2 are displayed.



```
>> memory
Maximum possible array:          6349 MB (6.657e+09 bytes) *
Memory available for all arrays:          6349 MB (6.657e+09 bytes) *
Memory used by MATLAB:          1507 MB (1.580e+09 bytes)
Physical Memory (RAM):          8171 MB (8.567e+09 bytes)

*  Limited by System Memory (physical + swap file) available.
```

Figure 5.2: Machine's memory

The one considered for this functionality is `Maximum possible array`. This value depends also on the tasks running on the machine. The risk is that the variables computed by the code can exceed this value of memory. The user can decide to save memory, as explained in section 2.1.1.

Chapter 6

Video Tutorials

Together with the executable for installing the software, a set of four video tutorials is realized on a dedicated youtube channel: `https://www.youtube.com/watch?v=ED3x00H4yN4&list=PLEJZLD0-4PeKW6Ze984q08bNz28GTntkR`. A short description of each video tutorial is presented here.

6.1 Video Tutorial 1 - Introduction and Mathematical Framework

In this video tutorial, the theoretical aspects behind the software are presented. In particular, the mathematical framework of linear data decomposition methods is summarized, recalling motivation and fundamental background. The general framework of decomposition in terms of matrix factorization is illustrated in detail.

6.2 Video Tutorial 2 - The DFT and the POD

This video presents the theoretical background for the Discrete Fourier Transform and the Proper Orthogonal Decomposition (POD), and how these fit in the matrix factorization formalism introduced in the first video. Both decompositions are tested on two of the five exercises available in our Github repository `https://github.com/mendezVKI/MODULO`.

6.3 Video Tutorial 3 - The Multiscale POD (mPOD)

This video is dedicated to Multiscale Proper Orthogonal Decomposition (mPOD), and is composed of four parts. In the first part, we review the fundamentals of Multiresolution Analysis (MRA) and its link to filter banks. In the second part, we study how filtering the data constraints the frequency content of its POD modes. The third part combines the results of the first two and presents the mPOD, including a glance at current activities on the development of the fast mPOD algorithm. Finally, in the fourth part, we take back the exercises that were analyzed during the previous video.

6.4 Video Tutorial 4 - Overview of MODULO and how to import data

This video tutorial gives an introduction of the software package, from its installation to a general overview of the graphical user interface and its main components. Specifically, the user will learn how to import data to be analyzed (1D or 2D, scalar, or vector quantities). A distinction between *'Separated mesh'* and *'Embedded mesh'* is highlighted. Moreover, the definition of the *Region of Interest* (RoI) and how to set it is given. These are the main characteristic of the software which are used in all the decomposition methods (POD, DFT, and mPOD).

6.5 Video Tutorial 5 - Performing POD

This video tutorial presents how to perform the Proper Orthogonal Decomposition (POD) in MODULO. The 1D and the 2D vector test cases analyzed in the previous videos are taken as examples and analyzed. The user will be guided through the whole process, from data importing to the definition of RoI, and the exporting of the results.

6.6 Video Tutorial 6 - Performing DFT

This video tutorial presents how to perform the Discrete Fourier Transform (DFT) in MODULO. The same exercises from the previous videos will be considered. The tutorial will highlight the major differences in the DFT, particularly linked to the imaginary nature of its modes and to its data-independent temporal structures. The video also describes how to plot the normalized spectra of the dataset.

6.7 Video Tutorial 7 - Performing mPOD

This video tutorial presents how to perform the Multiscale Proper Orthogonal Decomposition (mPOD) in MODULO. Focus is given to the GUI for setting the mPOD parameters and the associated multi-resolution analysis of the temporal correlation matrix. These include the parameters of the filter bank and the mPOD algorithm to assembly the mPOD basis from the basis of each scale.

6.8 Video Tutorial 8 - The Memory Saving Functionality

This video tutorial presents the memory saving functionality and its architecture. This functionality allows for substantial savings of computational memory, although at the cost of a significant increase in data importing time. This functionality is tested for all the decompositions, and focus is given to the definition and the impact of the dataset partitioning during all the computations.

References

[1] M. A. Mendez, D. Hess, B. B. Watz, and J.-M. Buchlin. Multiscale proper orthogonal decomposition (mpod) of tr-piv data– a case study on stationary and transient cylinder wake flows. *submitted to Meas. Sci. Tech*, 2020.

[2] M.A. Mendez. Generalized and multiscale data-driven modal analysis. In *Machine Learning for Fluid Mechanics: Analysis, Modeling, Control and Closures, von Karman Institute Lecture Series*. von Karman Institute, February 2020. Edited by M.A. Mendez, A. Ianiro, Bernd R. Noack, S.L. Brunton.

[3] M.A. Mendez, M. Balabane, and J.M. Buchlin. Multi-scale proper orthogonal decomposition of complex fluid flows. *Journal of Fluid Mechanics*, 870:988–1036, may 2019.

[4] M.A. Mendez, A. Gosset, and J.-M. Buchlin. Experimental analysis of the stability of the jet wiping process, part II: Multiscale modal analysis of the gas jet-liquid film interaction. *Experimental Thermal and Fluid Science*, 106:48–67, sep 2019.

[5] M.A. Mendez, M.T. Scelzo, and J.-M. Buchlin. Multiscale modal analysis of an oscillating impinging gas jet. *Experimental Thermal and Fluid Science*, 91:256–276, feb 2018.