PyOMA_GUI: users manual

D.P. Pasca, A. Aloisio, M.M. Rosso, S. Sotiropoulos

Ver 1.1

UNIVERSITÀ DEGLI STUDI DELL'AQUILA
Treteknisk
Politecnico di Torino

PyOMA_GUI: users manual

D.P. Pasca, A. Aloisio, M.M. Rosso, S. Sotiropoulos

Ver 1.1

The PyOMA support team

As a part of

ArtIStE - Artificial Intelligence in Structural Engineering

Head of research lab: Professor Giuseppe Carlo Marano

Contents

1. Updates of the newer releases

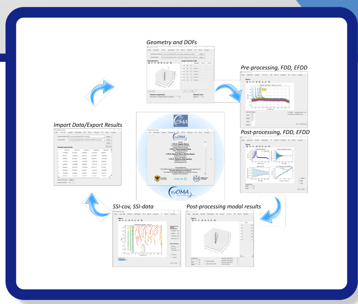
1.1 Last update of PyOMA_GUI users manual

October 29, 2022

1.2 Update version 1.1

- Minor bugs fixed in selecting identified peak frequencies;
- Minor bugs fixed in scale factor for visualizing deformed shape;
- Minor bugs fixed in general;
- Since GitHub does not allow pushing single files exceeding 100MB (total PyOMA_GUI.exe file is 104 MB) it is necessary to download the two archives and then extract it on local machine.

2. Introduction



In the structural health monitoring (SHM) paradigm, operational modal analysis (OMA) comprises several techniques and algorithms for estimating the dynamic characteristics of a structure in operational conditions from its vibration response. The OMA method has been spreading in the last years due to multiple advantages compared to input-output identification methods. In the current work, the authors present the implementation of a Python module named PyOMA and its Graphical User Interface (GUI) PyOMA_GUI. This software provides a user-friendly framework for the first time in the Python environment for estimating the experimental modal parameters (natural frequencies, mode shapes, damping ratios) of a structure from output-only vibration measurements in operational conditions.
PyOMA module is an open-source Python module that implements a complete output-only OMA framework for researchers, engineers, and practitioners. Through the implemented functions, it is possible to estimate the modal parameters of a civil structure using dynamic identification techniques derived from stochastic subspace identification (SSI) and frequency domain decomposition (FDD) [1], as illustrated in Fig.2.1. In addition, the authors provided a graphical user interface software version of the current module, called PyOMA_GUI, see Fig.2.2. The PyOMA_GUI aims to improve the appeal of the existing open-source Python OMA module, which has already been used in several applications. Not secondarily, the graphical user interface does not require any Python expertise or Python coding knowledge pre-requisite.
The present users manual is based on the research article [2] from the same authors.

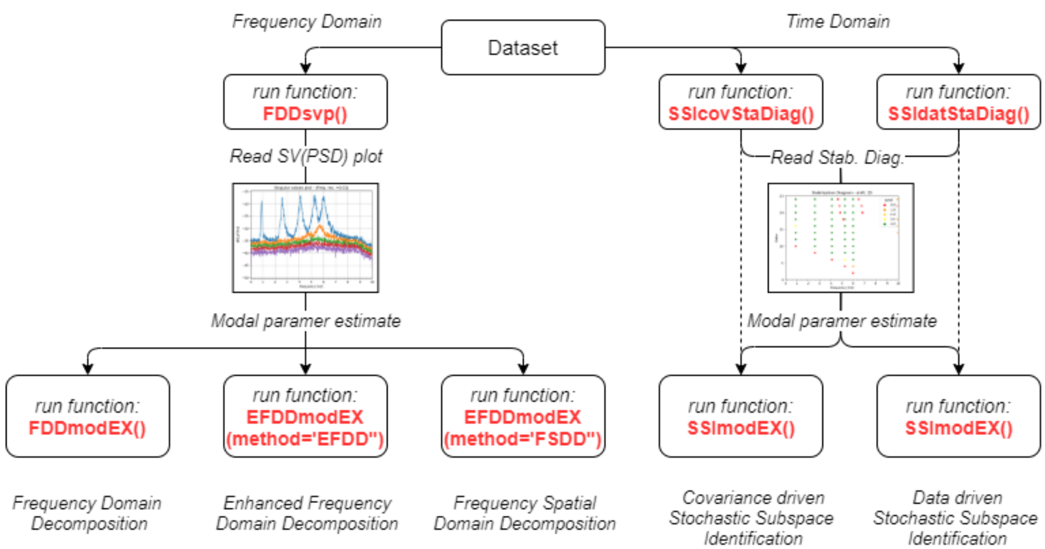


Figure 2.1: Functions implemented in PyOMA python module.

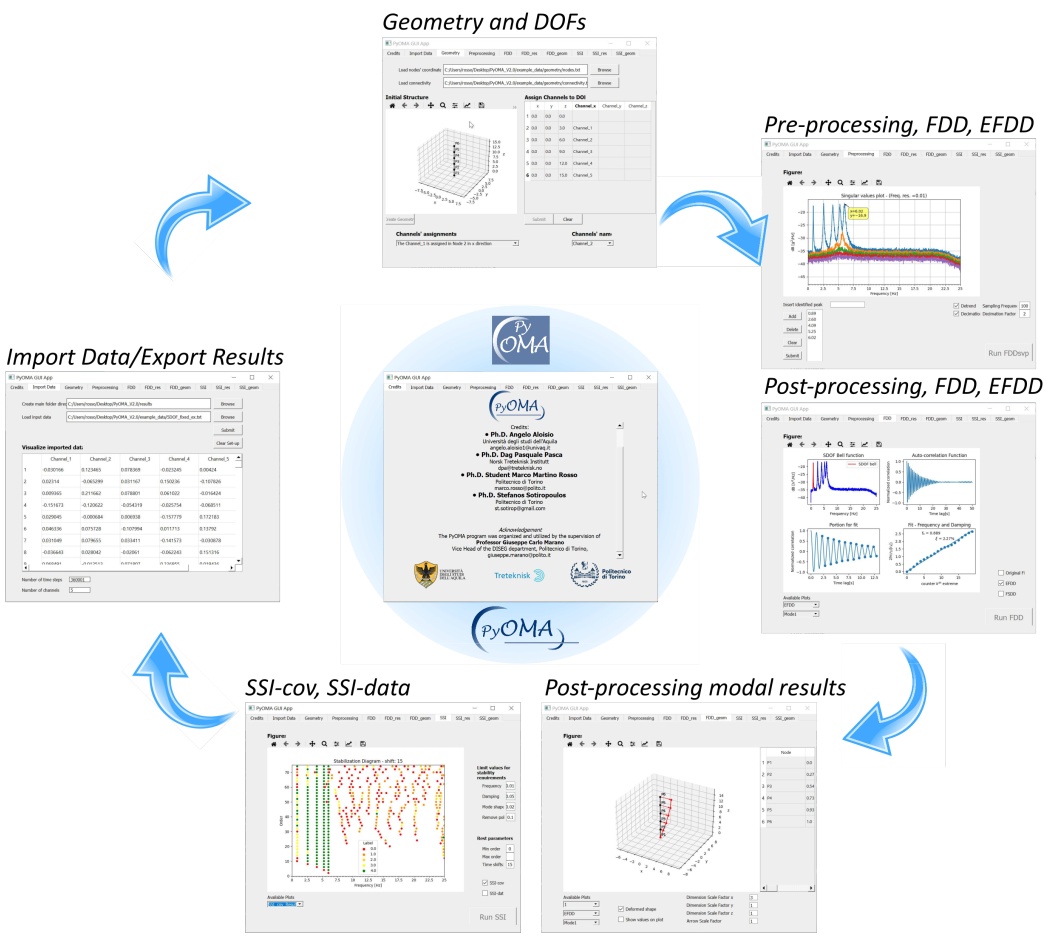


Figure 2.2: Functions implemented in PyOMA python module.

2.1 Example considered in the following chapters

In the following chapters, the users manual presents the main functionalities of the PyOMA_GUI software, based on a specific example. The output-only vibration data needed to carry on the proposed example have been generated with the function `oma.Exdata()` of the PyOMA python module.
Let us consider a 5 Degrees of Freedom (DOF) shear-type frame with lumped mass m = 25.91 [Ns^2/mm] at each floor, and the same story stiffness k = 10000 [N/mm] to all stories. Solving the eigenvalue problem gives the natural frequencies of the system:

$$f_n = \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{Bmatrix} = \begin{Bmatrix} 0.88995 \\ 2.59776 \\ 4.09511 \\ 5.2607 \\ 6.0001 \end{Bmatrix} [Hz]$$

And the (unity normalized) mode shapes:

$$\Phi = \left[\{\phi_1\} \quad \{\phi_2\} \quad \{\phi_3\} \quad \{\phi_4\} \quad \{\phi_5\}\right] = \begin{bmatrix} 0.28463 & -0.763521 & 1 & 0.918986 & -0.5462 \\ 0.5462 & -1 & 0.28463 & -0.763521 & 0.918986 \\ 0.763521 & -0.5462 & -0.918986 & -0.28463 & -1 \\ 0.918986 & 0.28463 & -0.5462 & 1 & 0.763521 \\ 1 & 0.918986 & 0.763521 & -0.5462 & -0.28463 \end{bmatrix}$$

The damping matrix is calculated assuming a constant damping of 2% to all modes. Synthetic signals, corresponding to the acceleration time history at each floor, are generated by the function using scipy's `signal.StateSpace` class. All the 5 DOF are excited by a Gaussian white noise input, then the results from each channel are polluted with a noise source with SN=10%.

3. Import data

Figure 3.1: PyOMA_GUI.exe icon.

When the user double click on the *PyOMA_GUI.exe* (Fig.3.1), the first *Credits*'s tab appears (Fig.3.2)

The first tab of the software is the *Import data* tab (Fig.3.3). Firstly, the user has to select an **empty folder** which would be the folder in which the software will automatically export the results, in terms of graphs images, and text files. This folder can be in any location of the O/S. It can be an existing folder or a new one that the user may create on the dialog box (Fig.3.4).

Thereafter, the user must upload the input data file with the Browse button. The current admissible file formats are **.txt, .xls, .xlsx, .csv**. The files have to be pre-processed externally, e.g. removing any textual headers, eventually providing to the software numerical columns data only (Fig.3.5). The first five lines of the uploaded data will be displayed in the table below (Fig.3.6). The user has the possibility to change the names of the channels, automatically set to *Channel_x*, for *x* as an integer value varying from 1 to the total number of columns of the imported file. With a double click on the headers' names, the user may manually modify the name referring to each specific column.

At the bottom of the Import data tab, the software will show the number of time steps and the number of channels retrieved from the imported data (Fig.3.7). In order to proceed, the user must push the submit button (Fig.3.8). Lastly, there is the Clear Set-up button that clears a whole set-up, to start from the beginning of a new analysis.

Figure 3.2: First view at the software opening: *Credits*' tab.

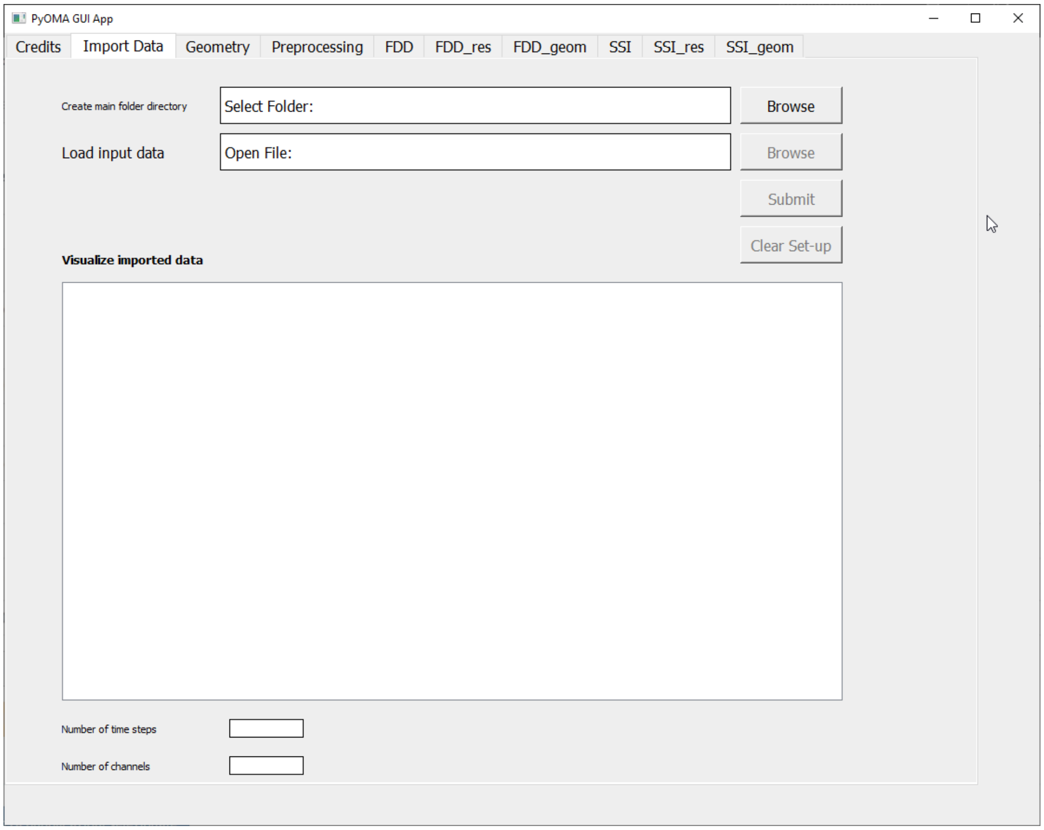


Figure 3.3: Second tab of the software: *Import data* tab.

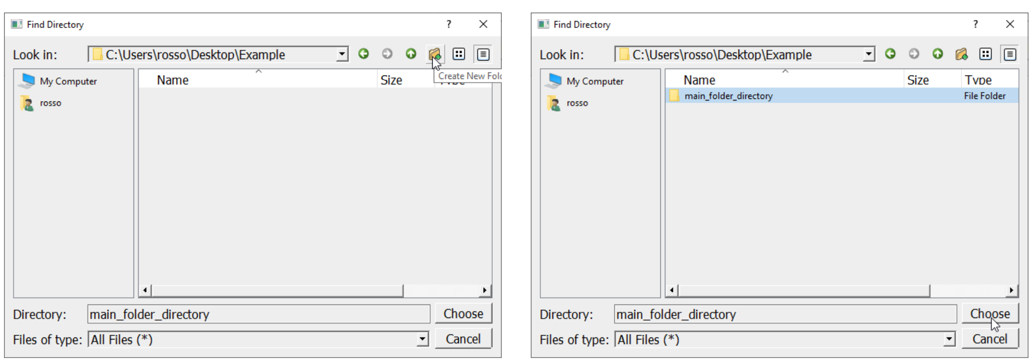


Figure 3.4: Create a new empty working directory.

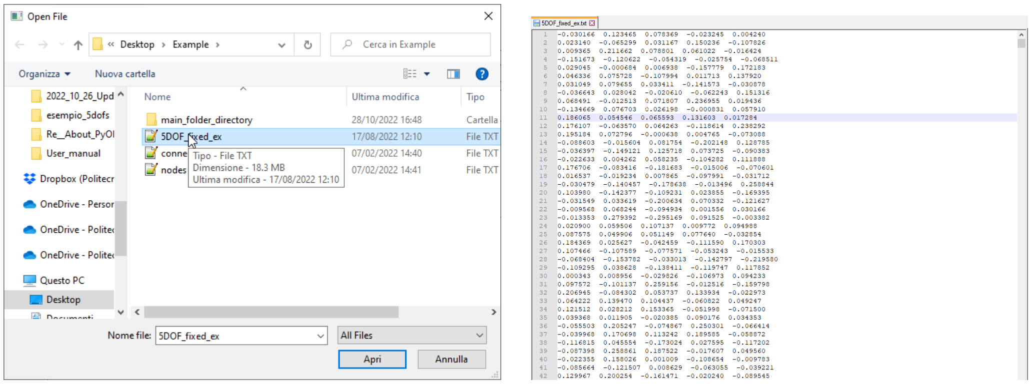


Figure 3.5: Create a new empty working directory.

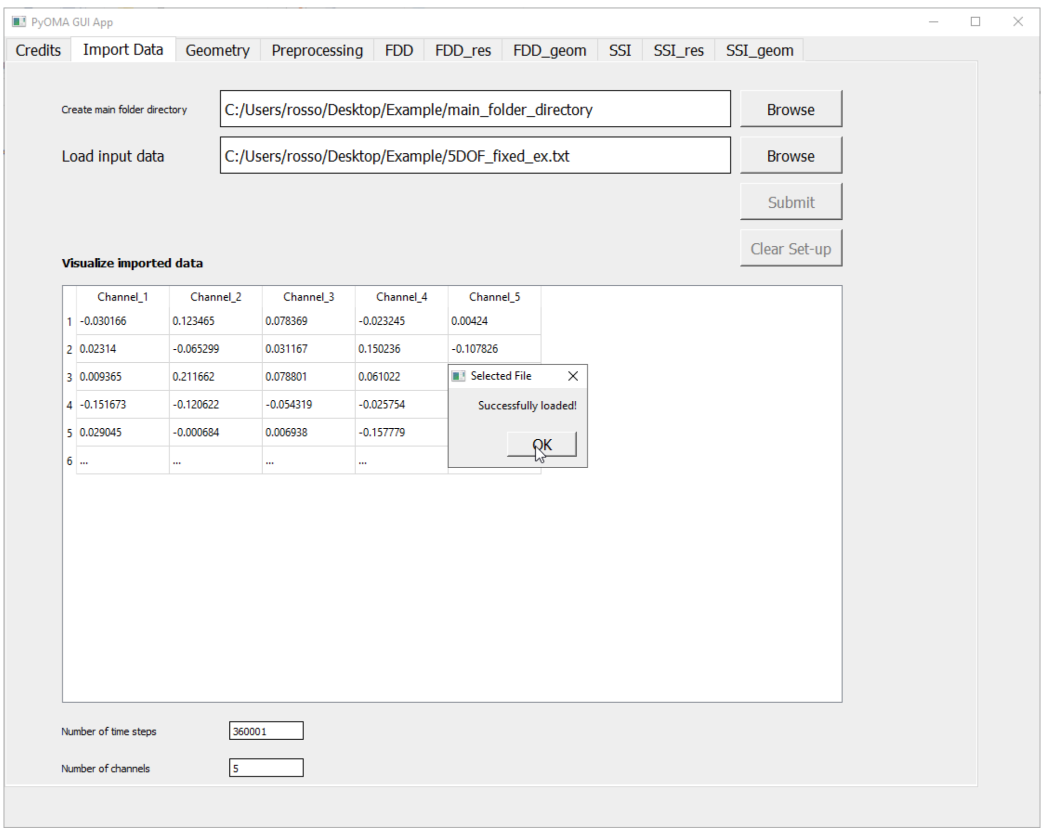


Figure 3.6: Create a new empty working directory.

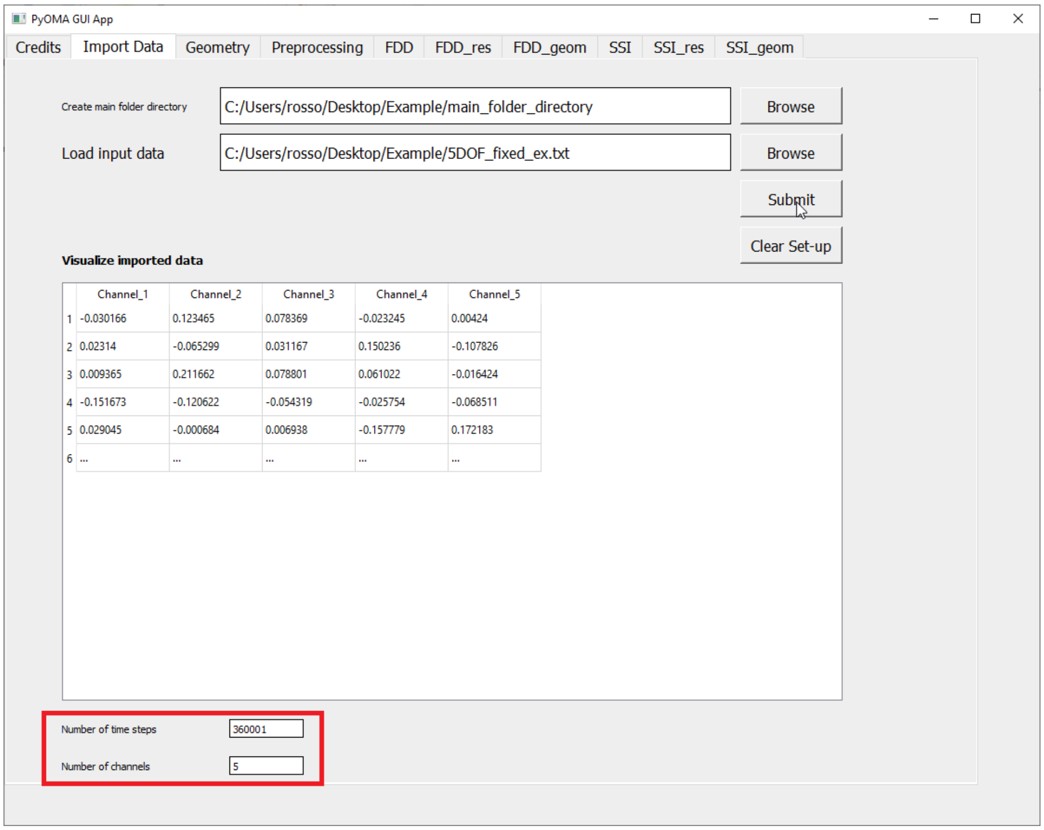


Figure 3.7: Create a new empty working directory.

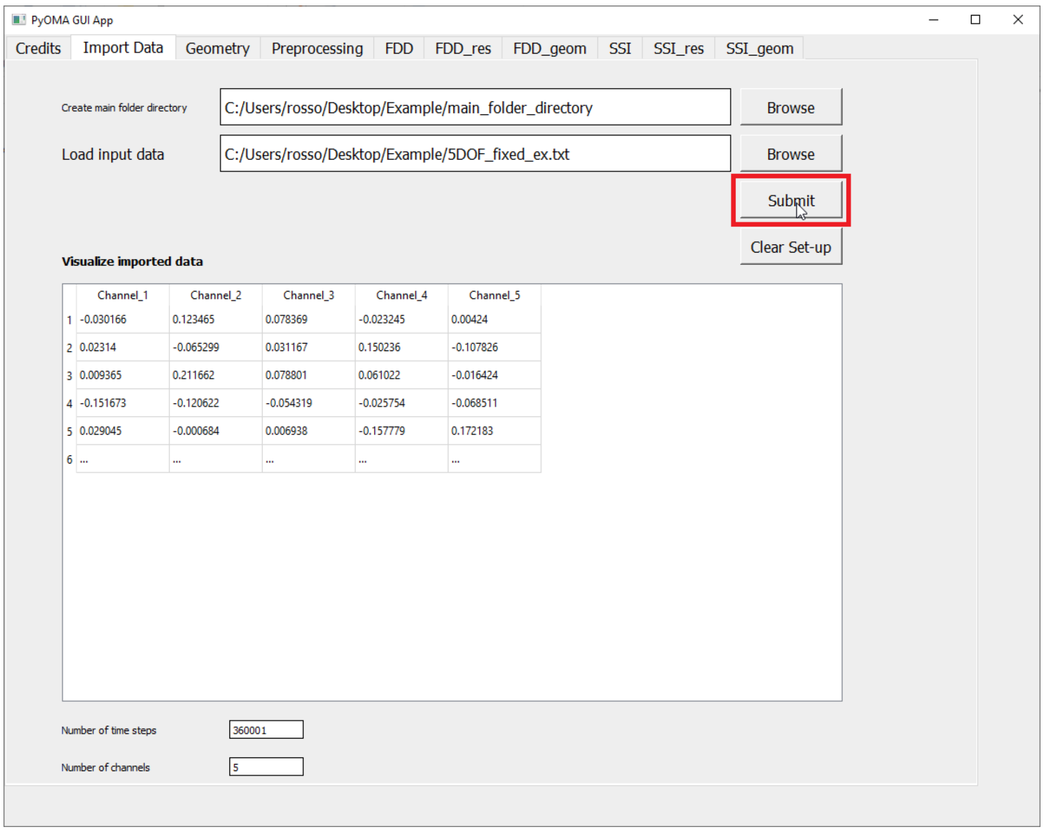


Figure 3.8: Create a new empty working directory.

4. Geometry

The second tab of the PyOMA_GUI software is the *Geometry* tab, see Fig.4.1. In this tab, the user has to browse and upload the text files (**txt** format) with the nodes' coordinates (Fig.4.2 and Fig.4.3) and connectivity (Fig.4.4) to visualize a schematic visualization of the structures and the monitored degree of freedoms (DOFs) and sensors' locations. Pushing the button Create Geometry the user can see a wire-frame model of the imported structure (Fig.4.5).
As shown in Fig.4.6, in the table **Assign Channels to DOF**, the user must manually connect the proper DOF of each node of the table (Channel_x, Channel_y, Channel_z) the name of the imported channel, referred to the specific column of the imported data from the previous tab. The user can check the correct name of the channel in the combo-box Channels' name (Fig.4.7). After pressing the submit button, the user may check the correct assignments of each DOF to the specific channel of the imported data file in the combo-box below (Fig.4.8).

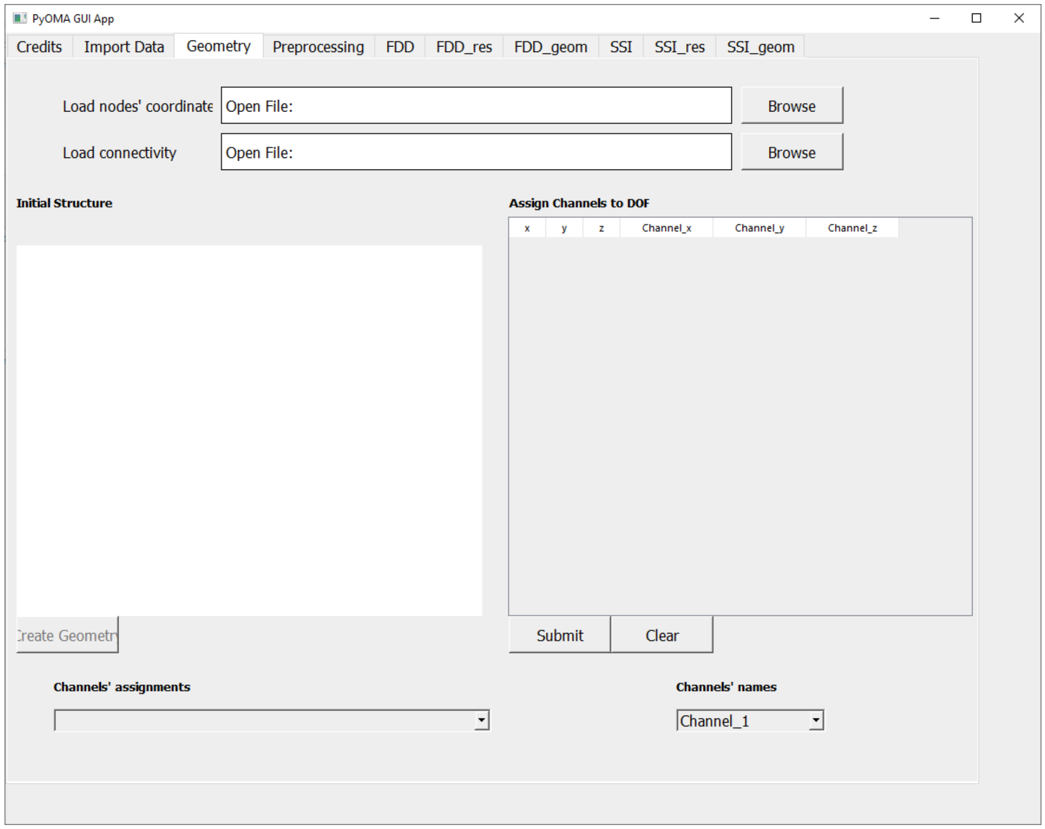


Figure 4.1: *Geometry* tab.

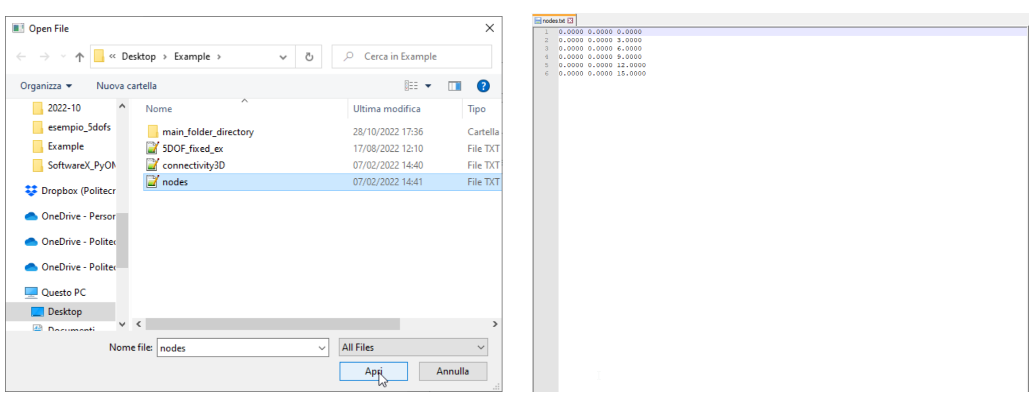


Figure 4.2: Upload nodes file.

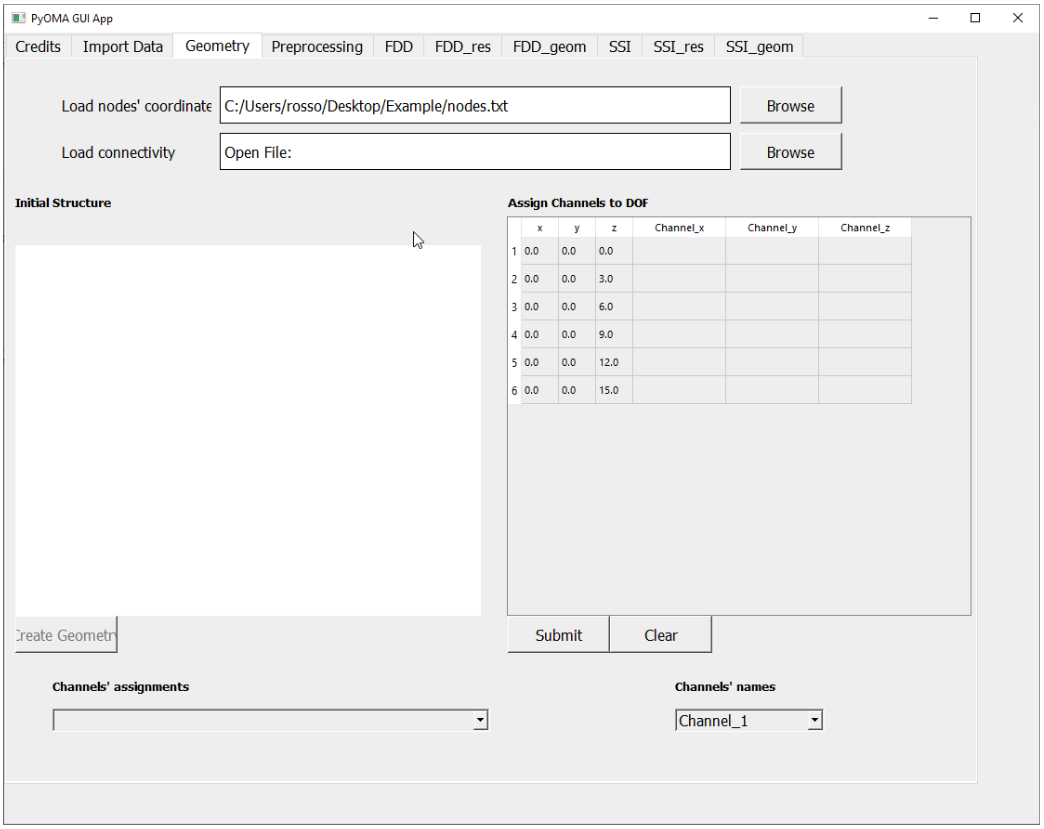


Figure 4.3: *Geometry* tab after uploading the nodes file.

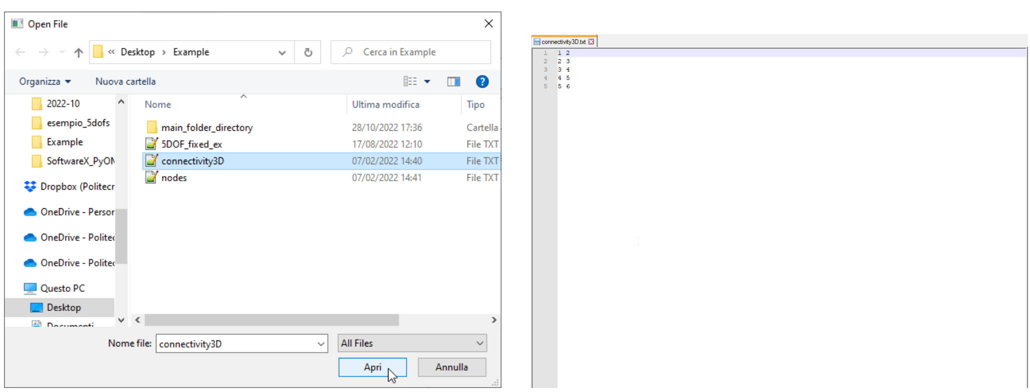


Figure 4.4: Upload nodes' connectivity file.

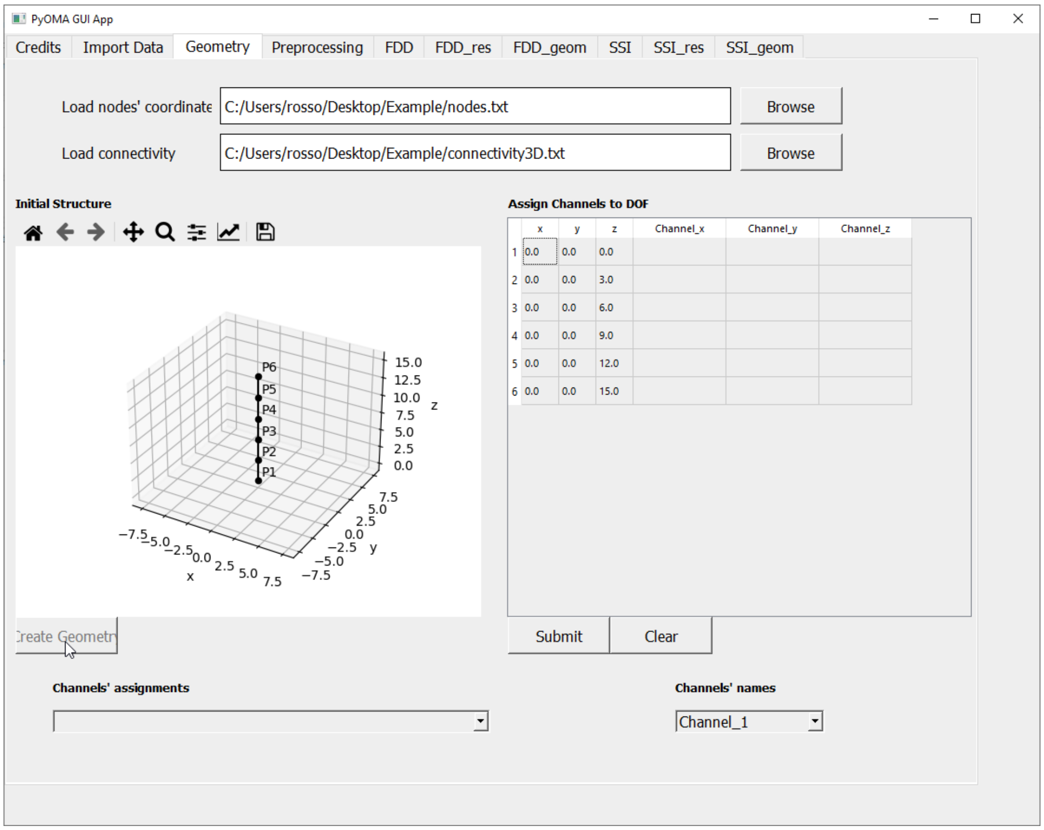


Figure 4.5: Graphical check of the imported nodes and connectivity.

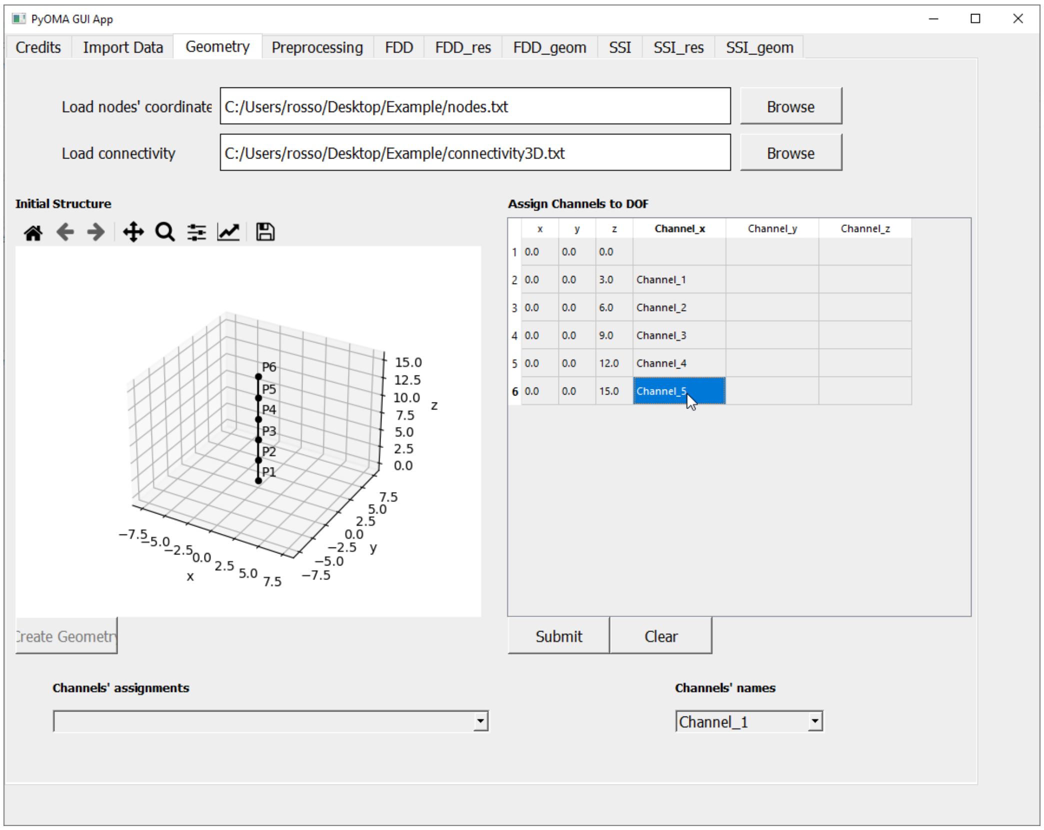


Figure 4.6: Manual assignment of the imported data channels to each node DOF.

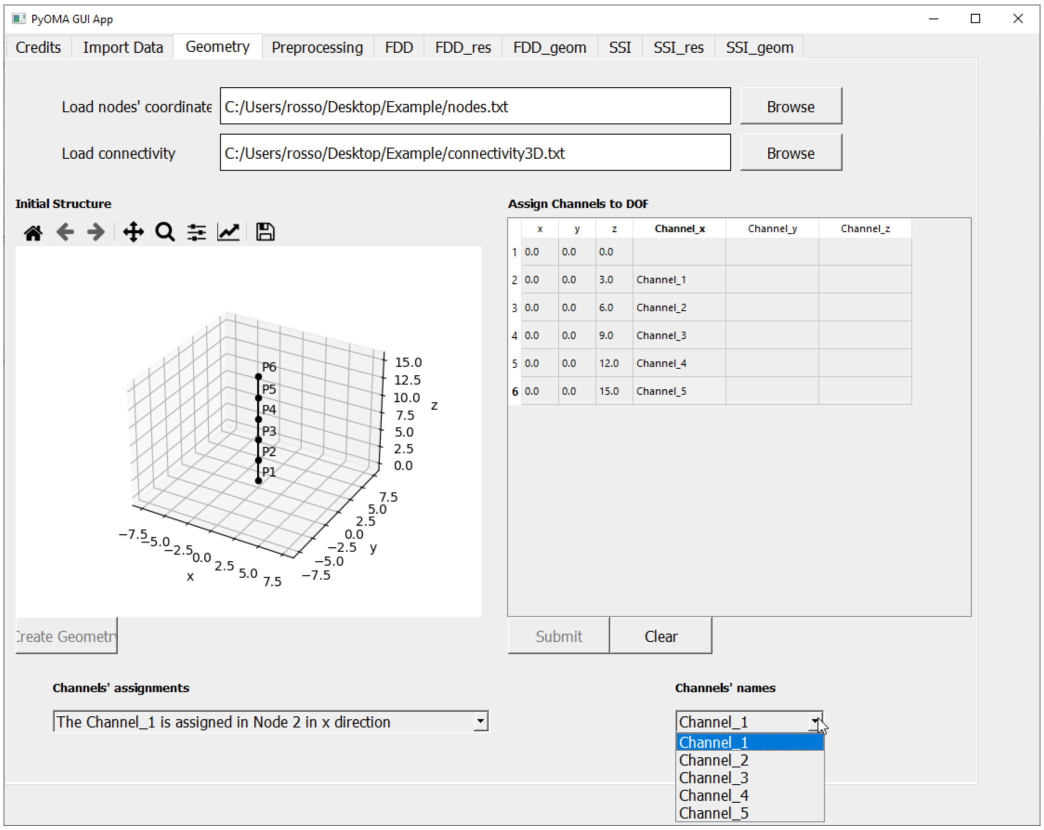


Figure 4.7: How to check the names of the imported data channels.

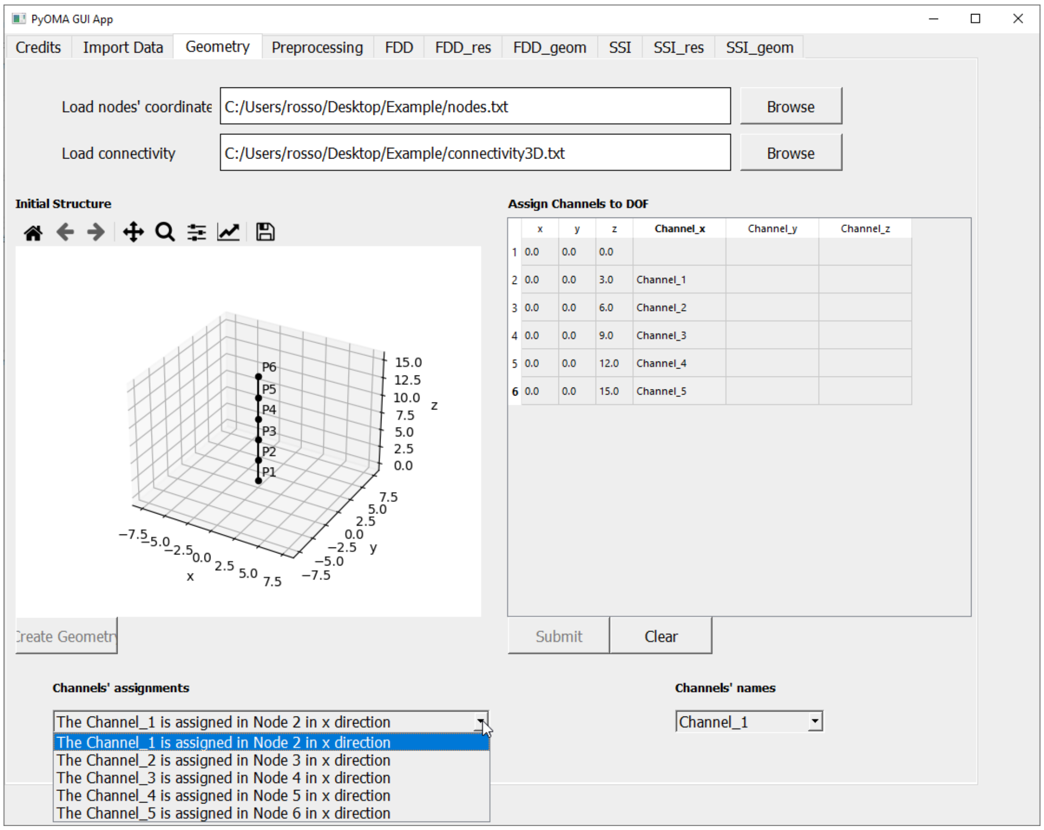


Figure 4.8: How to check if the nodes have been correctly assigned.

5. PreProcessing

In the tab *PreProcessing* (Fig.5.1), the user has the possibility to perform a prior signal processing in terms of detrending of the imported data files and a subsequent decimation procedure. At first, the user has to provide the **sampling frequency** of the signals and sign the corresponding checkboxes to perform detrening and/or decimation on the imported signals (Fig.5.2). The preprocessing is executed when the user clicks on the **Run FDD_svp** button (Fig.5.3). The command will run the function `oma.FDDsvp` of the PyOMA python module to perform the Frequency Domain Decomposition (FDD) algorithm, which returns a plot of the Singular Values (SV) of the Power Spectral Density (PSD) matrix, and a dictionary that contains the results that will be processed later to extract the modal properties. In the figure, the user can select the peak of interest and with the button Add, it is added to the list of the identified peaks. The rest buttons are to help us with the list (Delete a single item or Clear the whole list). With the submit button we are locking the selection and we are ready to proceed with the analysis.
At this point, the user has the possibility to perform the **peak peaking** procedure in order to select directly by clicking on the singular values graph (Fig.5.4), the peak corresponding to the natural frequencies of the structural system. After the click on a certain peak, the user can save the selected peak in the chosen peaks by clicking on the add button (Fig.5.5). After the peak selection, the user has to click submit button to proceed with the OMA analysis (Fig.5.6).

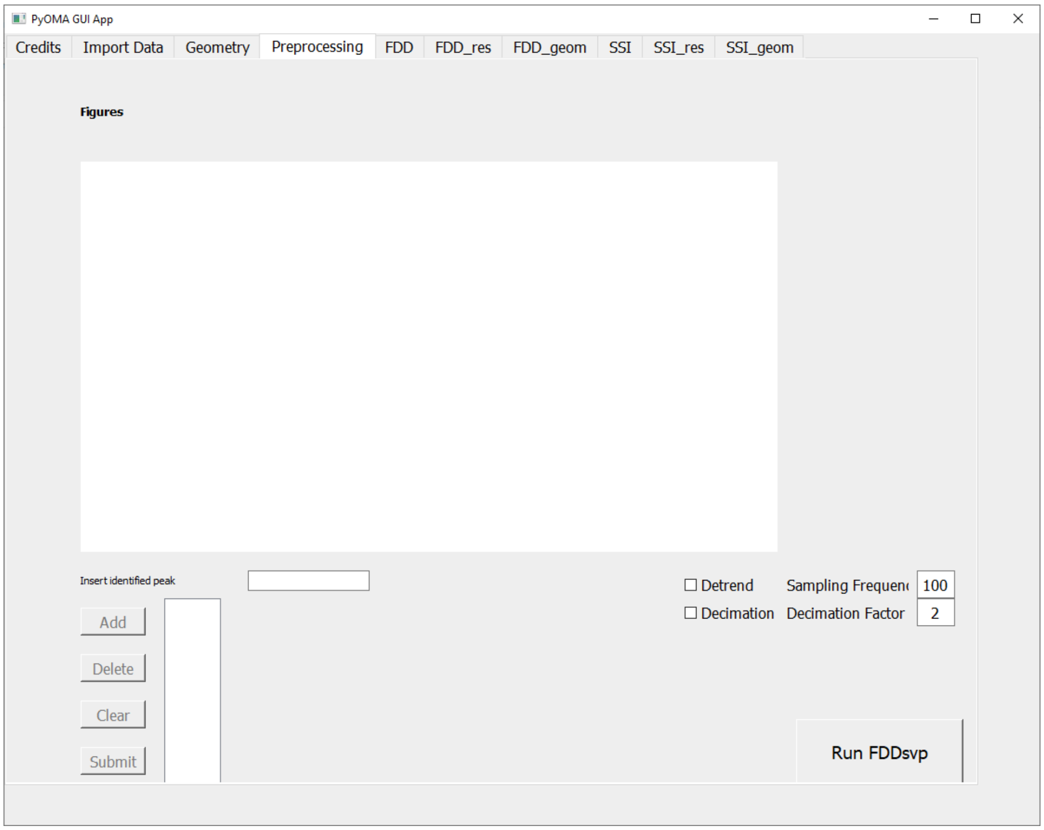


Figure 5.1: *Preprocessing* tab.

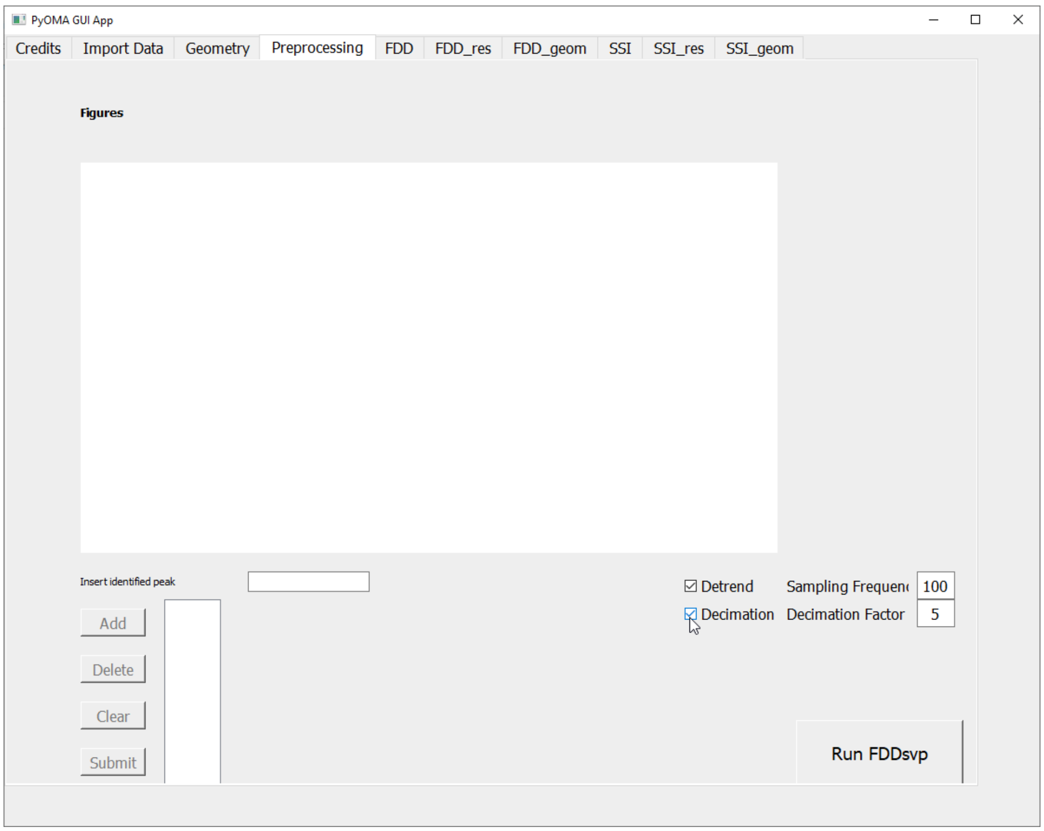


Figure 5.2: Sign the checkboxes to perform detrening and/or decimation on the imported signals.

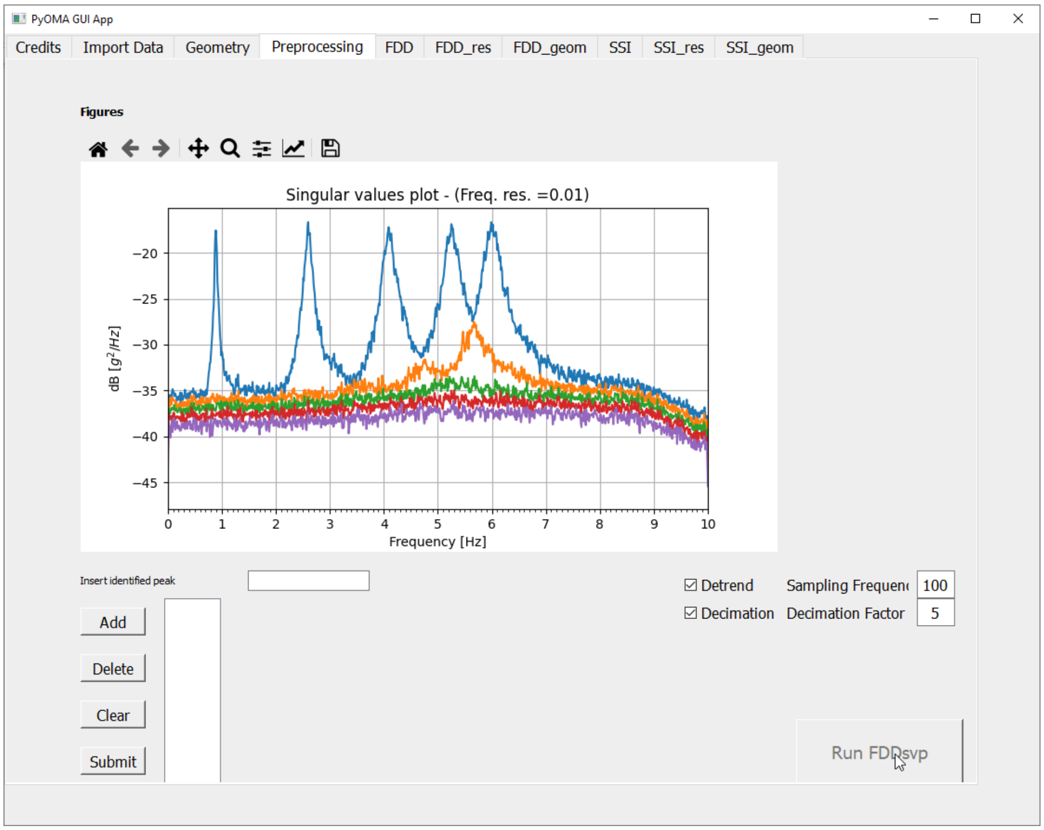


Figure 5.3: The preprocessing is executed when the user click on the **Run FDD_svp** button.

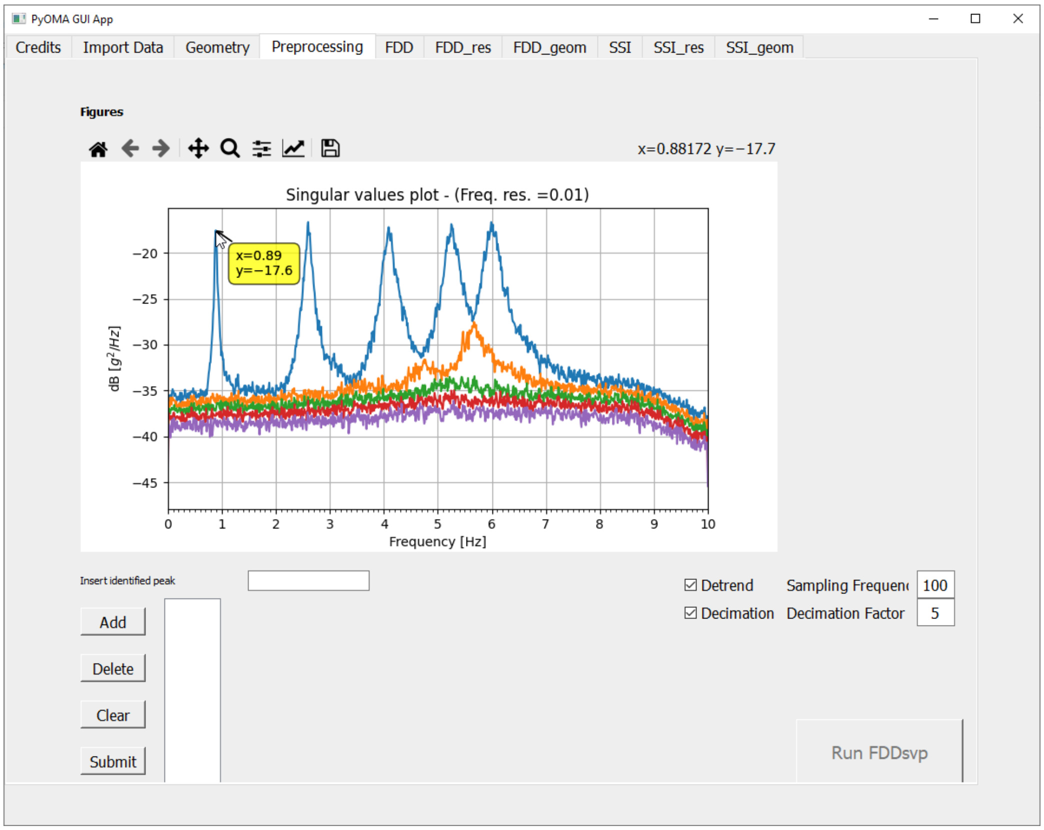


Figure 5.4: Select the peak of interest directly on the graph.

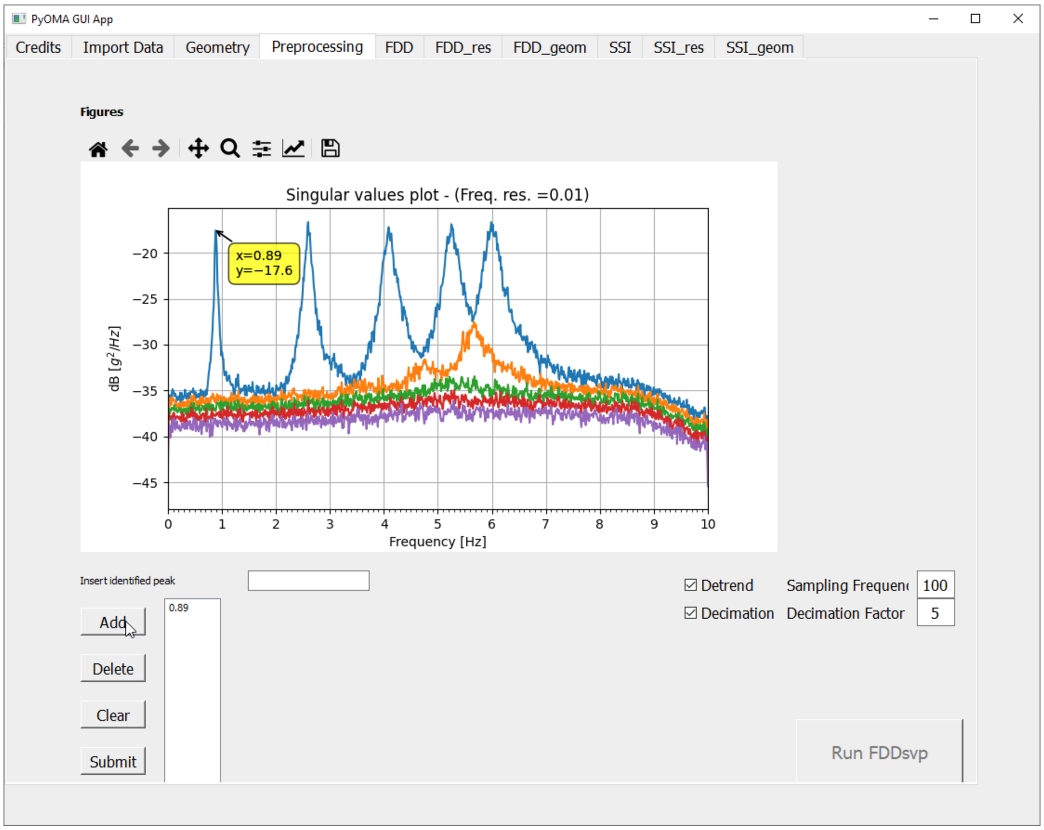


Figure 5.5: Add the peak of interest to the list of the saved peaks.

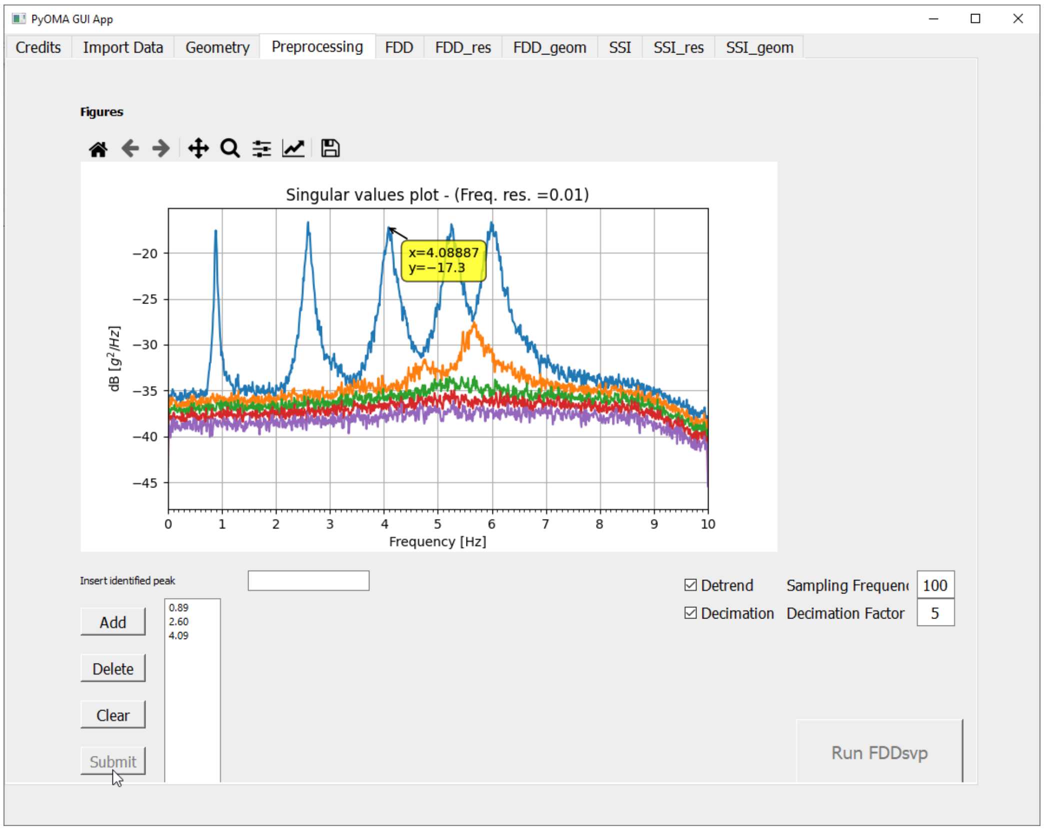


Figure 5.6: Submit the peaks of interest list to proceed with the OMA.

6. FDD

The current tab is the *FDD* tab, see Fig.6.1. This tab provides the user the ability to perform the frequency domain dynamic identification methods on the imported data based on the frequency domain decomposition (FDD) algorithm.
The user has to sign the checkboxes corresponding to the methods of interest and then push the button *Run FDD*. In the dropdown menu, there are the available graphs produced from the enhanced frequency domain decomposition (EFDD) and the Frequency-Spatial Domain Decomposition (FSDD) method [3]. These checkboxes allow the user to customize the return, permitting the extraction of the single DOF (SDOF) bells extraction from the PSD peaks [1], one for each mode, as depicted for instance in Figs. 6.2-6.3-6.4-6.5. The visualized figure is **refreshed** every time the user selects the desired mode from the dropdown menu is performed (Fig.6.5). The results of the analyses are automatically stored in the results directory previously selected in the *Import data* tab.
In detail, the first two checkboxes will run the function `oma.FDDmodEX` of the PyOMA python module and/or the `oma.EFDDmodEX` function to extract the modal information according to the FDD method and/or the EFDD method respectively. These functions return a dictionary that contains the results of the identification in terms of modal properties, which are shown in the next tab *FDD_res*. Specifically, the `oma.FDDmodEX` function will only extract the natural frequency and the mode shape, according to the original FDD algorithm as presented in [4]. On the other hand, the `oma.EFDDmodEX()` function extracts the modal properties (frequencies, mode shapes, damping) according to the EFDD algorithm as presented in [5]. The last checkbox will run the FSDD method which extracts the modal properties (frequencies, mode shapes, damping) according to the Frequency-Spatial Domain Decomposition method argued in [3]. The latter method isolates the modal coordinates by modal filtering and provides enhanced output PSD estimates, which yield better auto-correlation functions. These functions take as input the list of peaks previously identified in the singular values diagram provided by the function `oma.FDDsvp`.

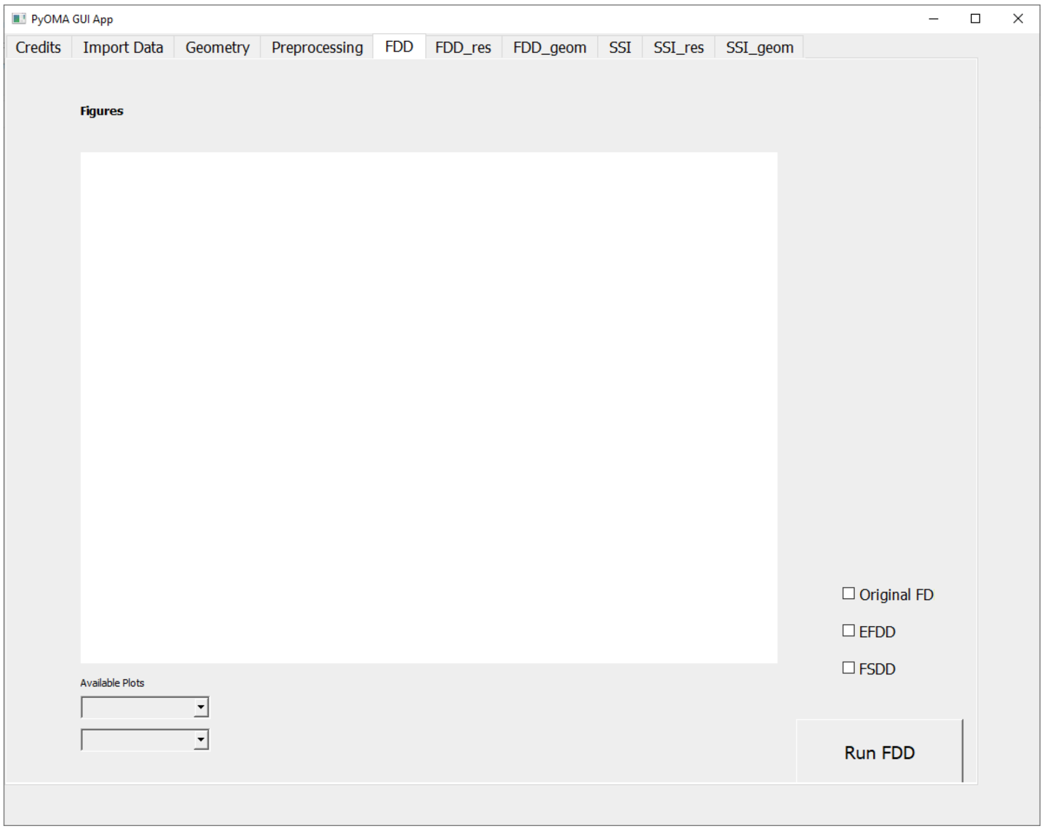


Figure 6.1: *FDD* tab.

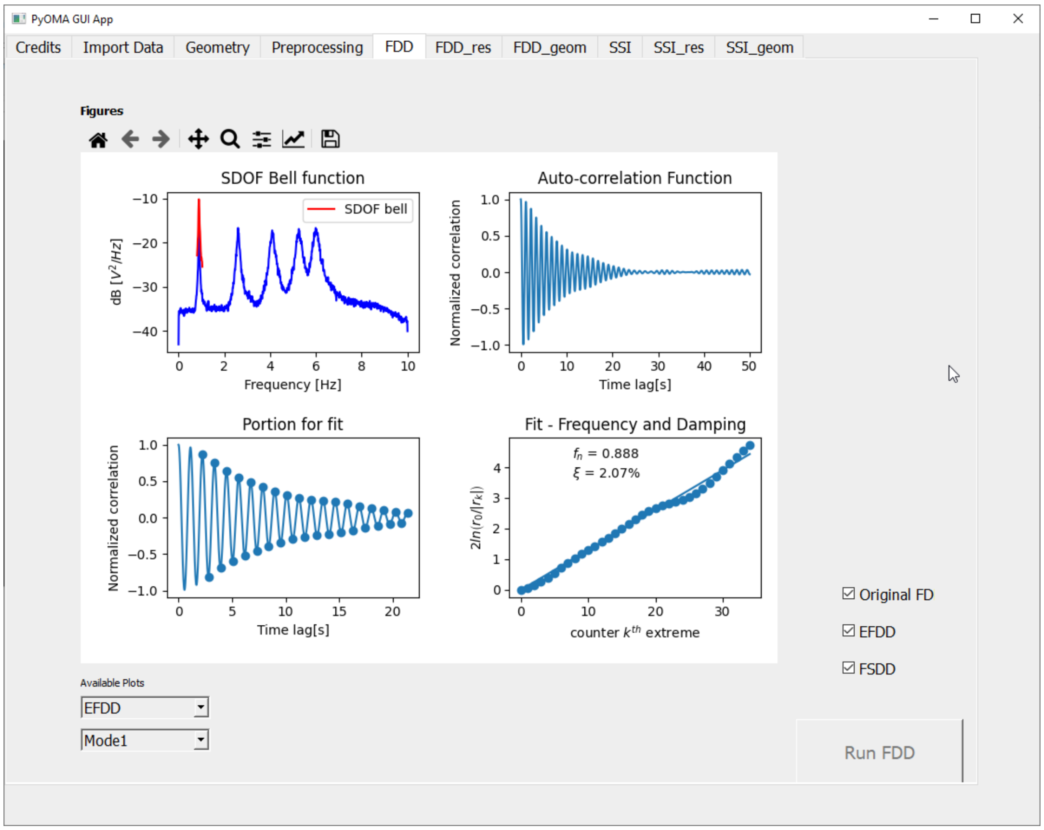


Figure 6.2: *FDD* tab.

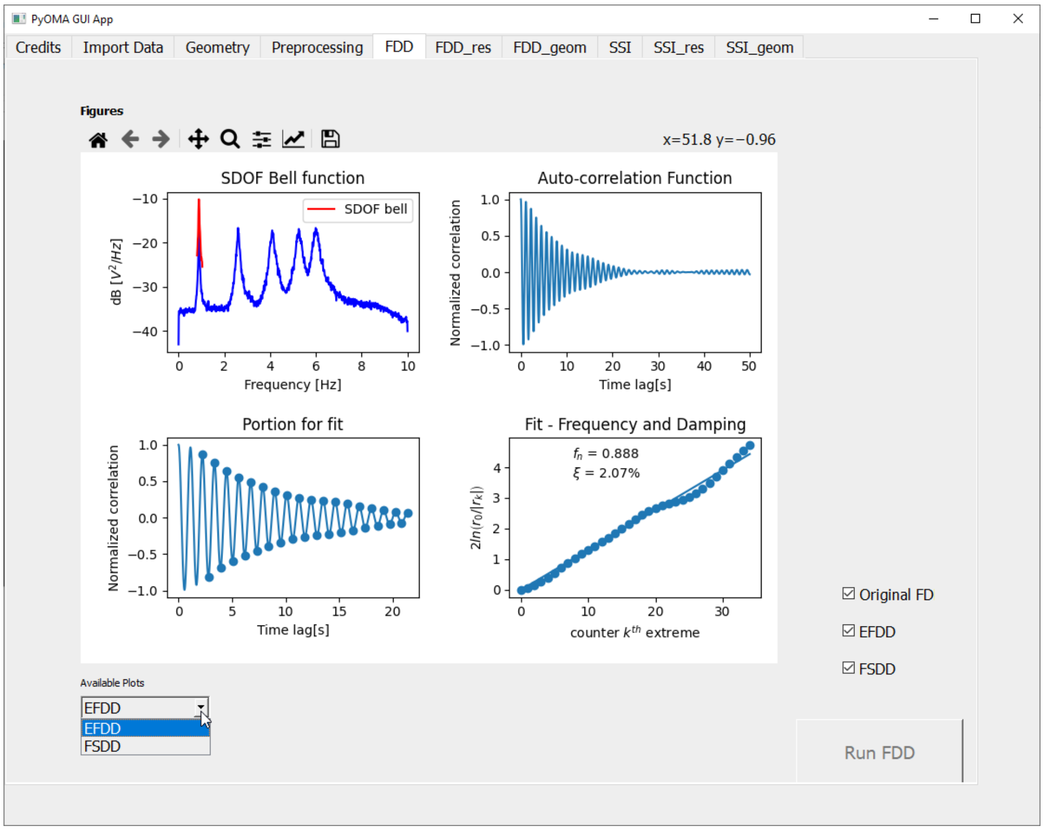


Figure 6.3: *FDD* tab.

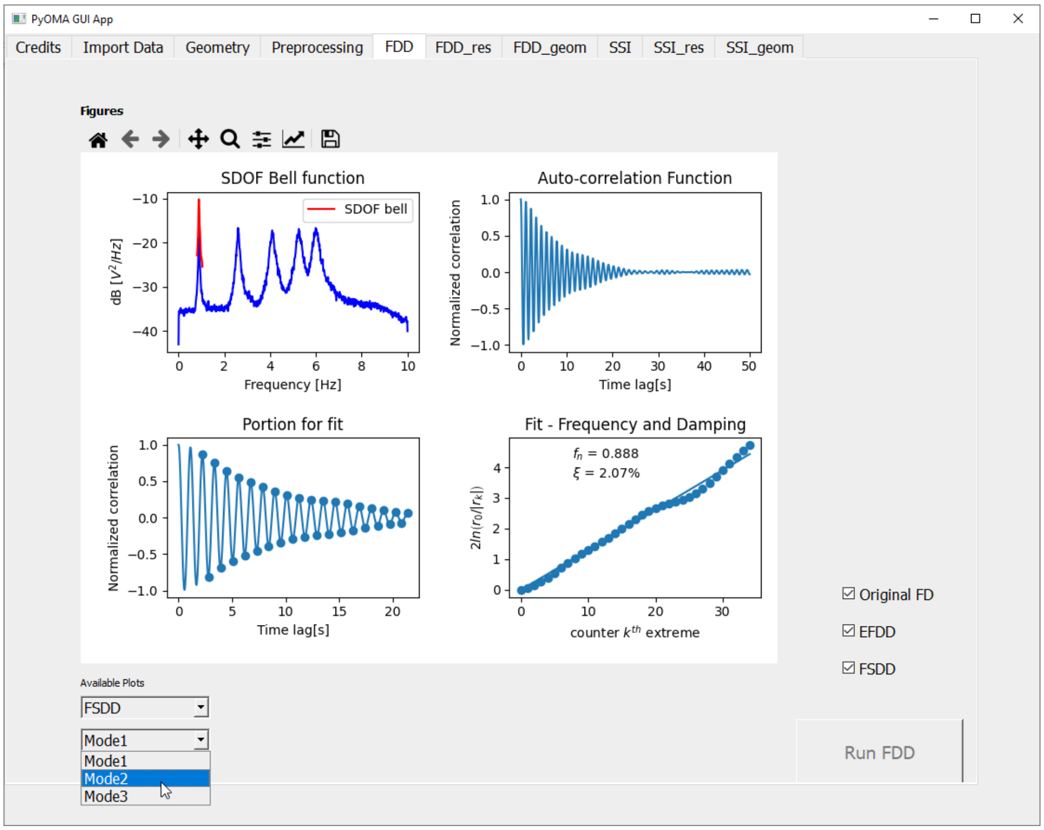


Figure 6.4: *FDD* tab.

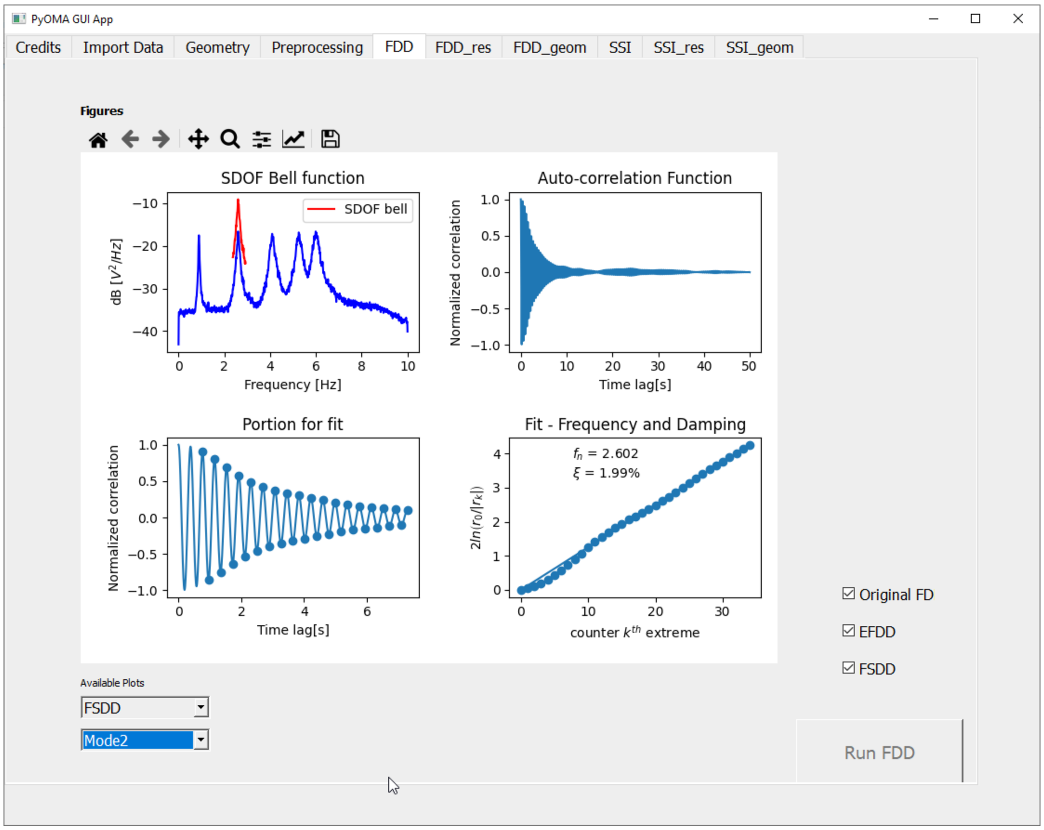


Figure 6.5: *FDD* tab.

7. FDD_res

In the current tab named *FDD_res* the user can inspect all the results derived from the FDD, EFDD, and/or FSDD analyses as depicted in Fig.7.1.
The results of the analyses and the SDOF bell figures have been automatically stored in the previously selected results directory in the *Import data* tab.

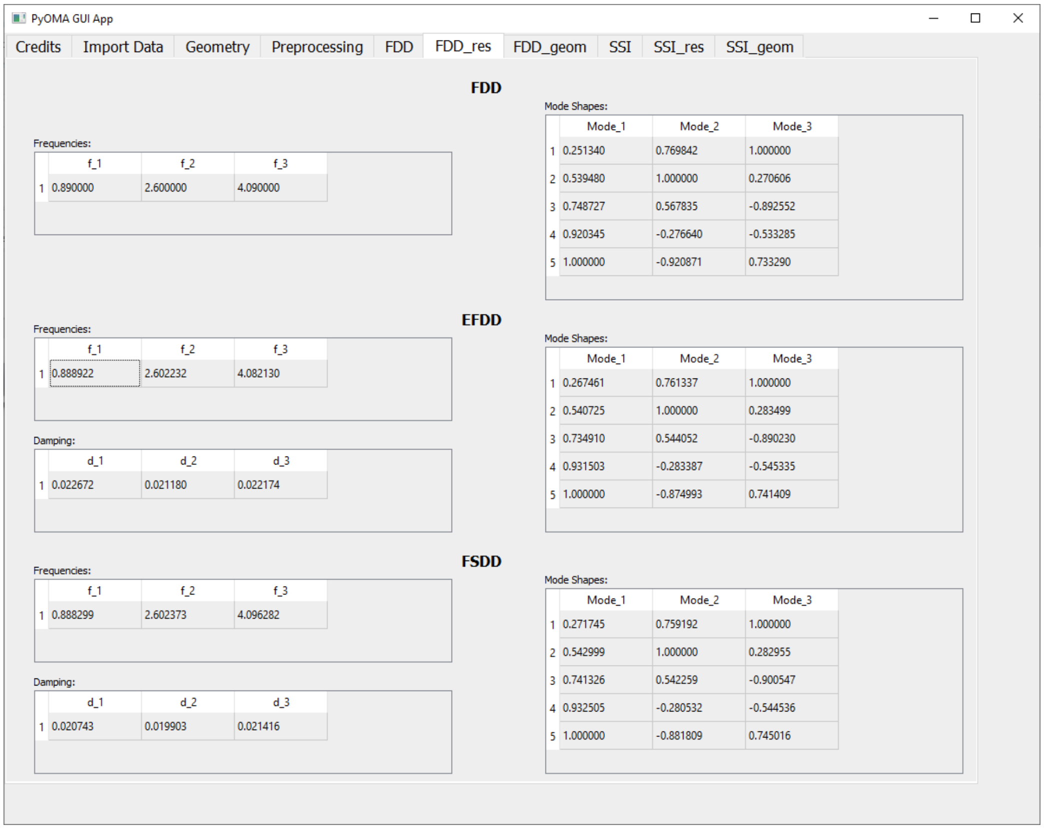


Figure 7.1: *FDD_res* tab.

8. FDD_geom

In the current tab named *FDD_geom* (Fig.8.1), the user can inspect the resulting mode shapes derived from the FDD, EFDD, and/or FSDD analyses as depicted in Figs.8.2–8.3. The resulting mode shapes can be visualized as static deformations, with the possibility to show them in several forms. As already mentioned, the figure is **refreshed** only when the mode of interest is selected from the side dropdown menu. In the side table, the values of the selected mode shape are reported for each node and each DOF, whereas the selected mode shape is represented in a vectorial way depicting red arrows (Fig.8.4). Extra abilities for the graph are provided, such as the *deformed shape* checkbox which permits to visualize of the mode shape as a piecewise line (Fig.8.6), or the *show values plot* checkbox which print the values of the vectorial components directly on the graph (Fig.8.7). Four scaling factors are also provided: the first three scale factors provide the ability to scale the deformed shape components in the global x,y, and z directions respectively (Fig.8.5), whereas the last one permits to increase or decrease the thickness of the depicted mode shape arrows (Figs.8.8–Fig.8.9).

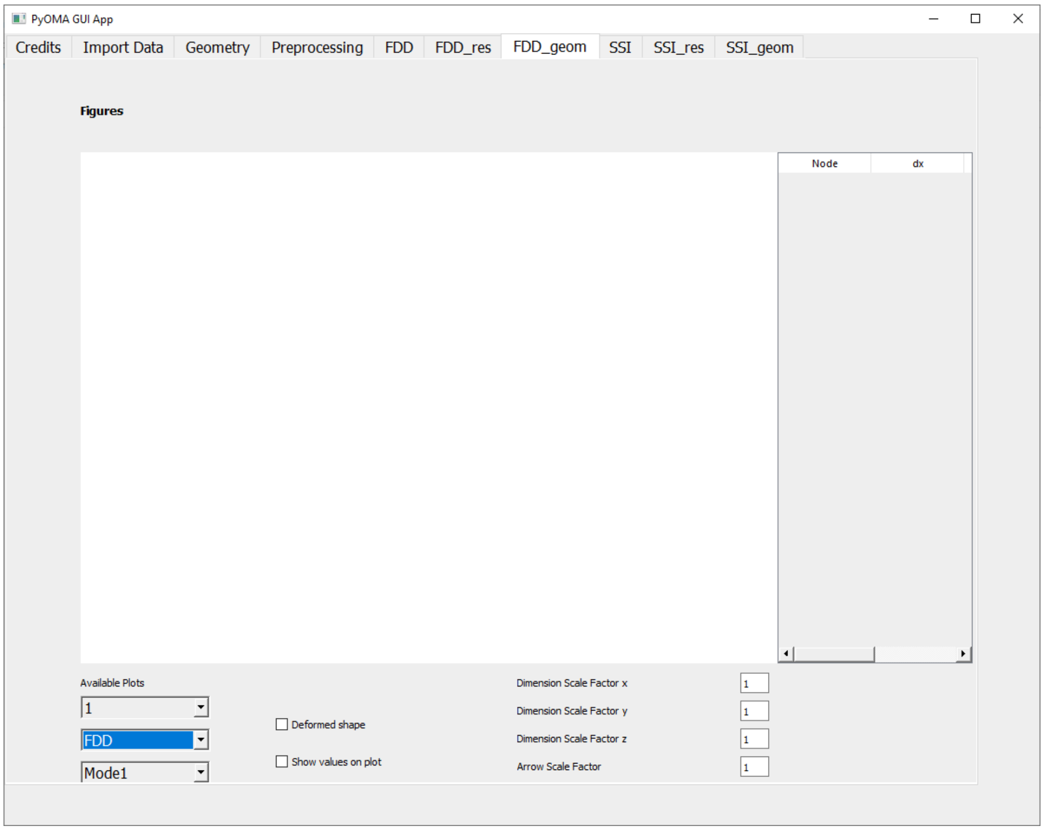


Figure 8.1: *FDD_geom* tab.

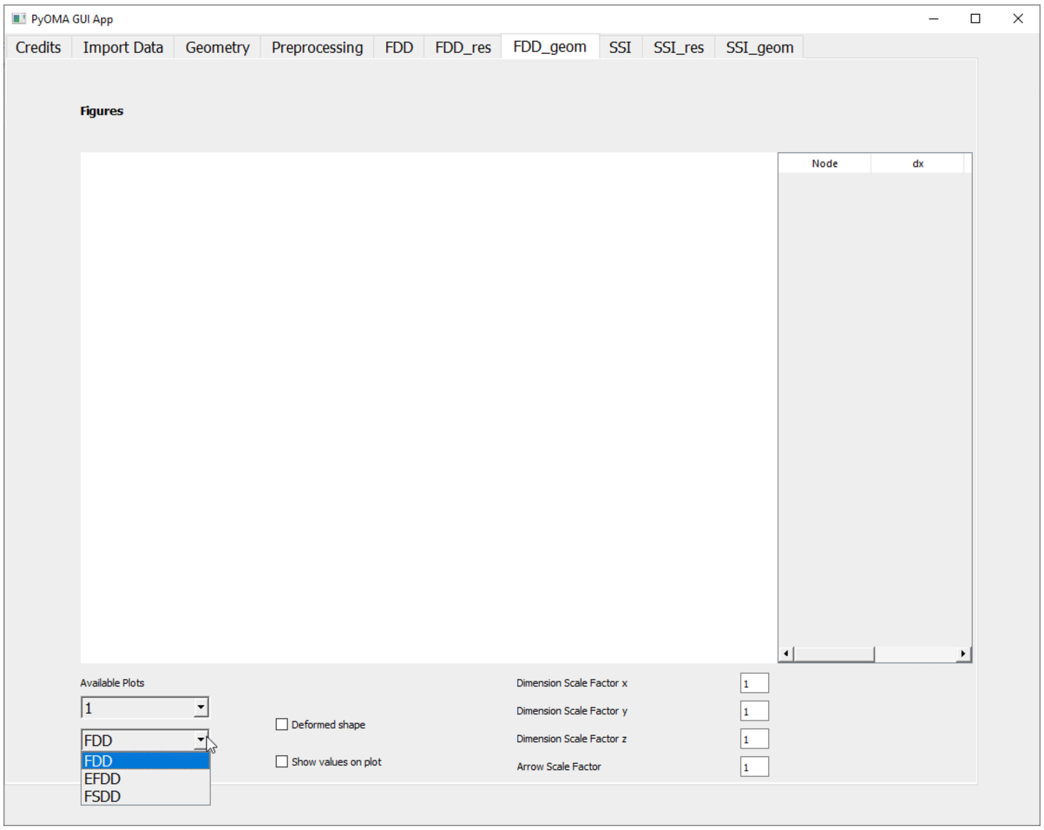


Figure 8.2: List of run methods. The user have to select the method of which is interested in.

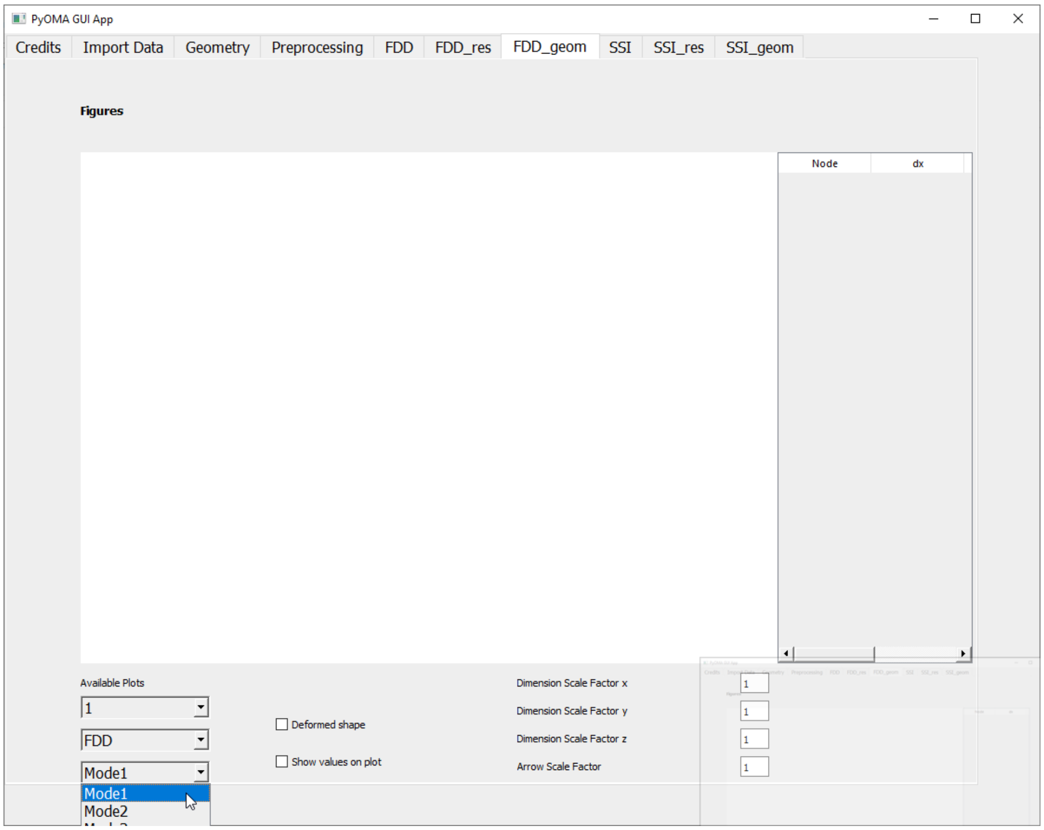


Figure 8.3: List of available mode shapes graphs for the selected method.

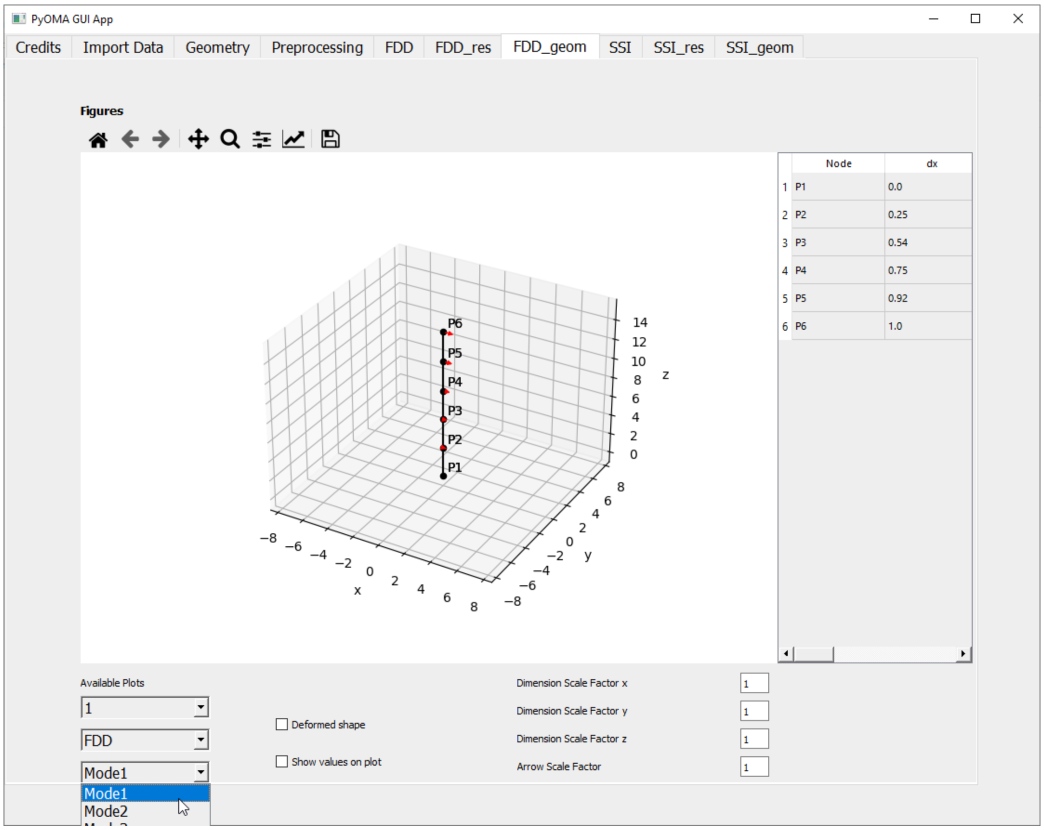


Figure 8.4: Mode shapes visualized as vectorial components only.

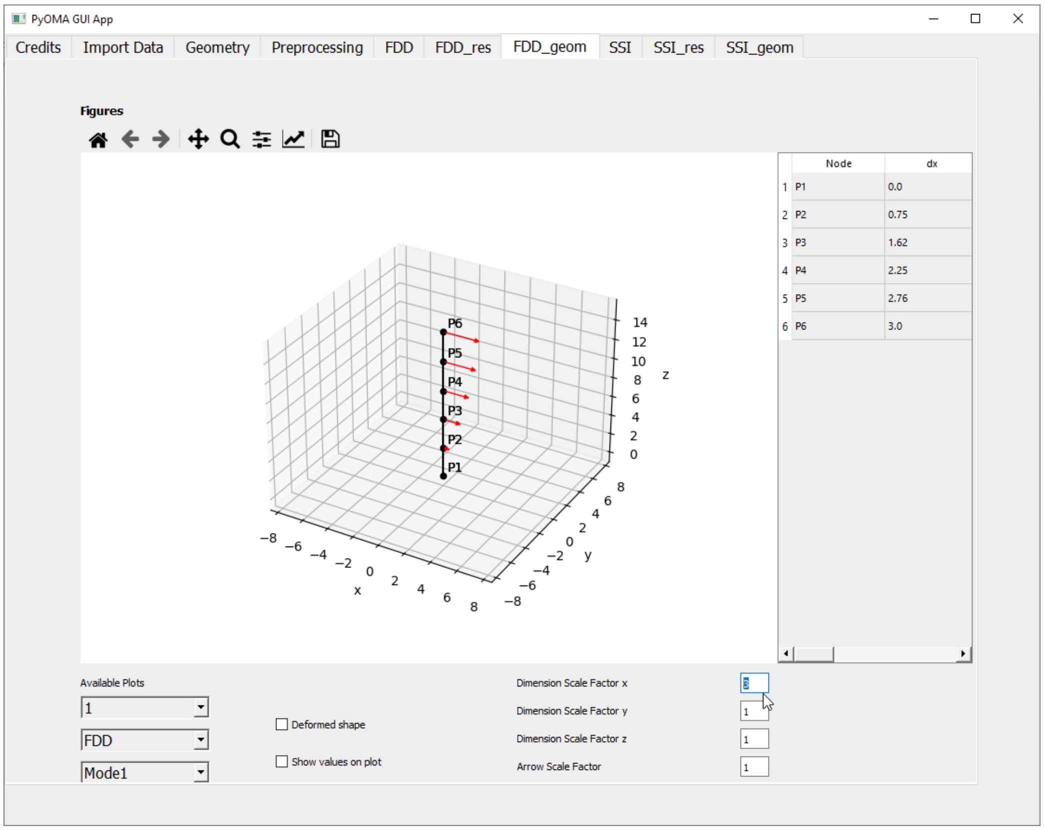


Figure 8.5: Mode shape vectorial components scaling for a better visualization.

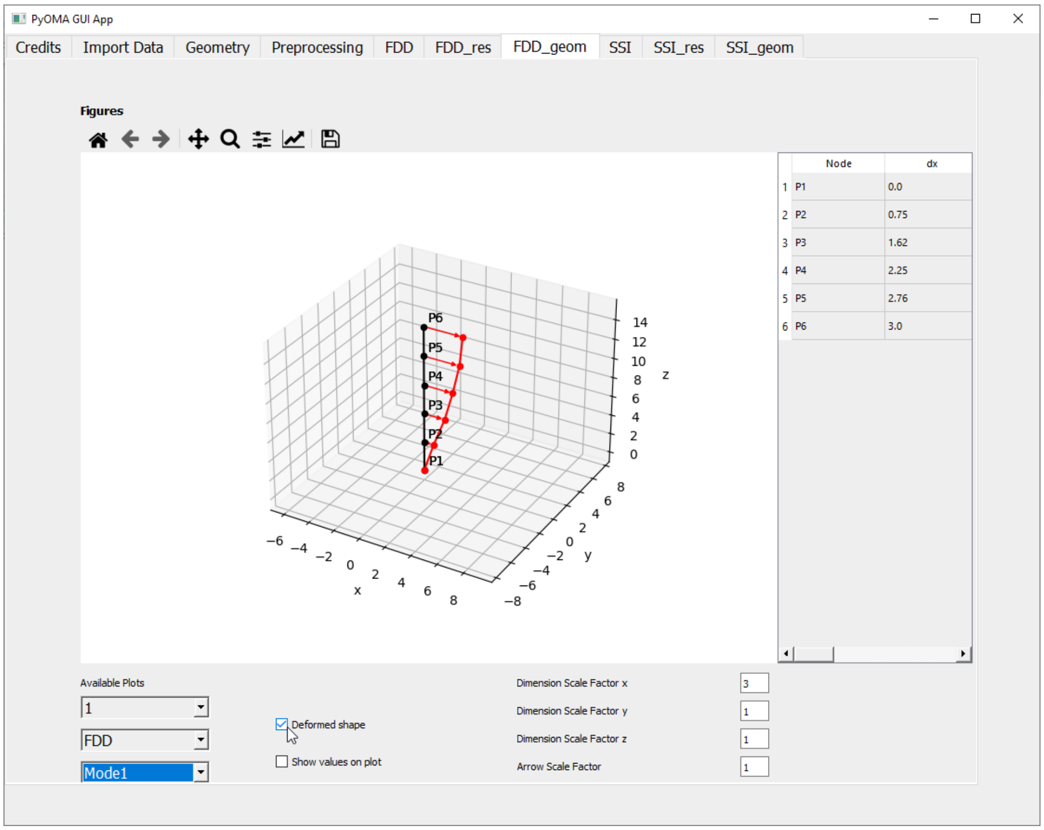


Figure 8.6: Mode shape representation with lines.

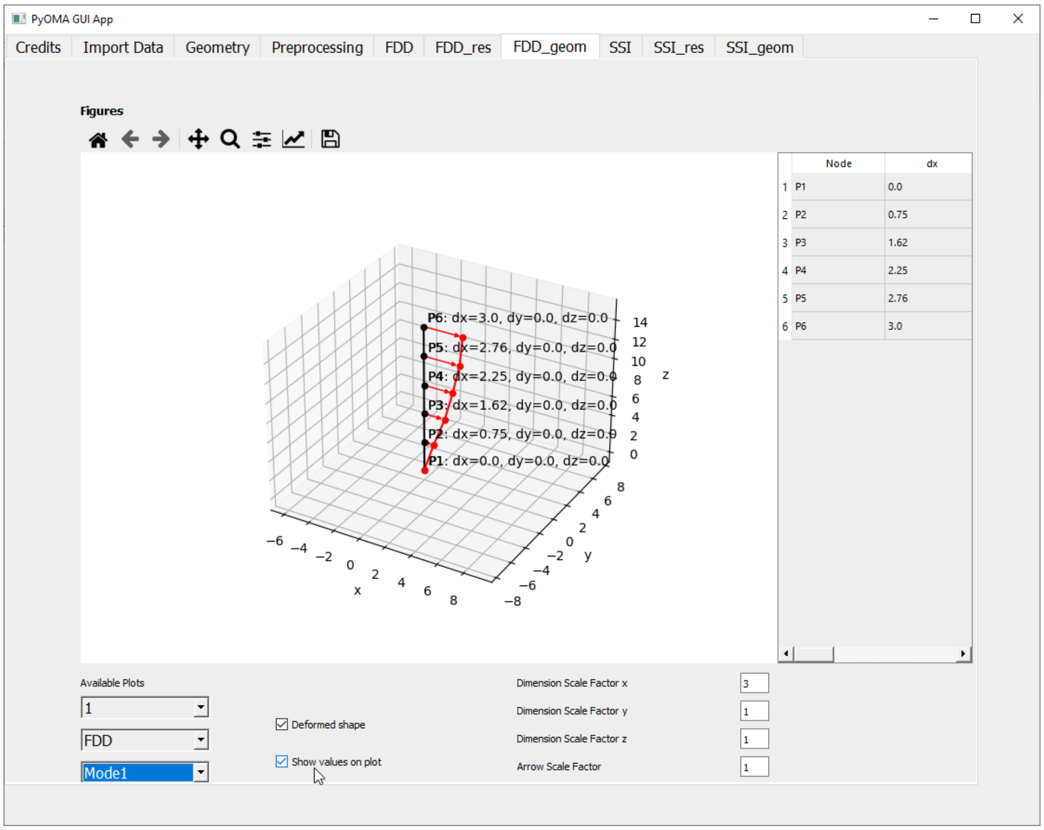


Figure 8.7: Adding annotations referred to the vectorial components of the mode shape 1.

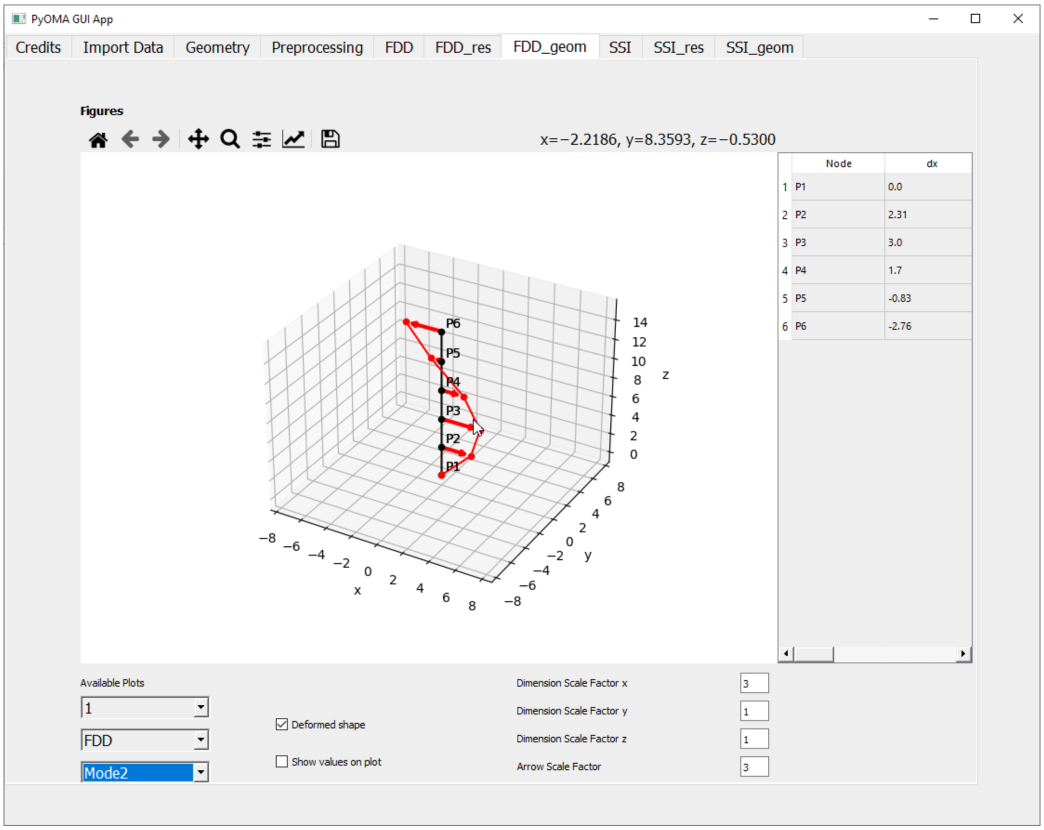


Figure 8.8: *FDD_geom* tab: mode shape 2.

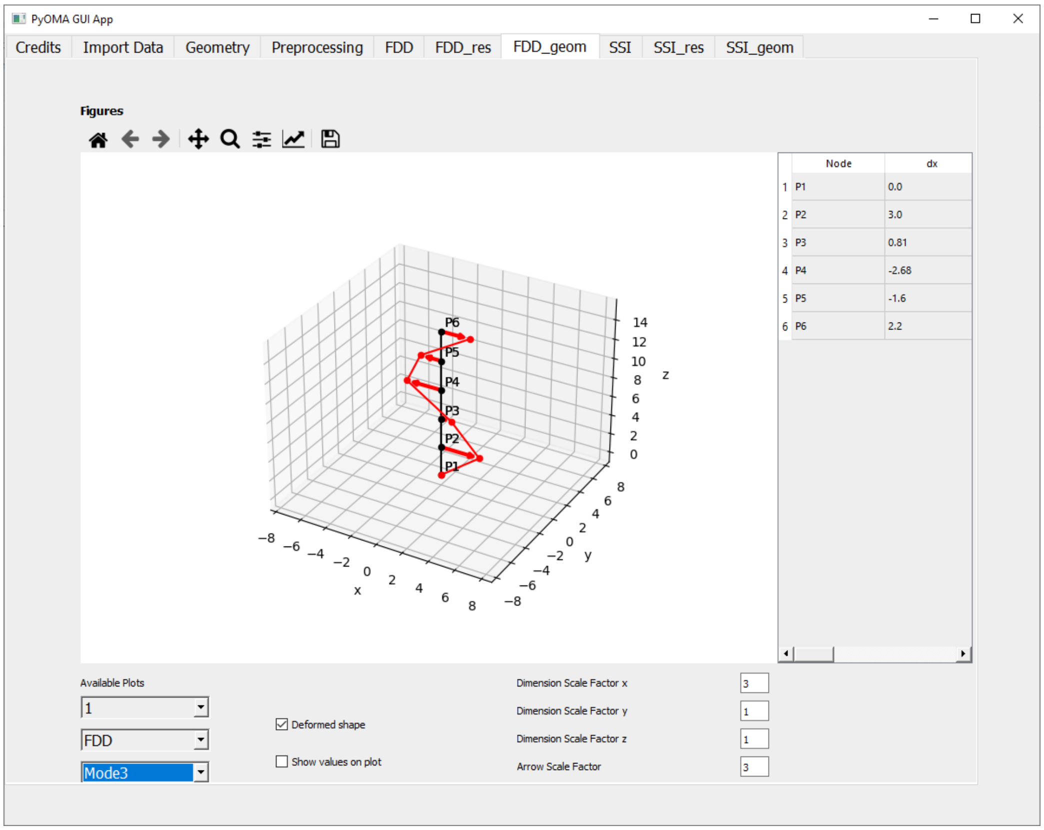


Figure 8.9: *FDD_geom* tab: mode shape 3.

9. SSI

The current tab is the *SSI* tab, see Fig.9.1. This tab provides the user the ability to perform the time domain dynamic identification methods on the imported data based on the stochastic subspace identification (SSI) algorithm.

The user has just to select the checkboxes corresponding to the desired type of SSI algorithm. The checkbox *SSI-dat* [6] performs the data-driven SSI algorithm version (Fig.9.3) which recalls the command `oma.SSIdatStaDiag` of the PyOMA python module. On the other hand, the checkbox *SSI-cov* executes the covariance-based SSI approach [7] (Fig.9.3) which recalls the command `oma.SSIcovStaDiag` of the PyOMA python module.

For these two SSI methodologies, the user is required to provide the number of block rows, better acknowledged as time lags or *time shift* parameter. The optional parameters allow the user to define the maximum model order, minimum order, and limit values to be used for the stability criteria of the poles in the stabilization diagram. The colors of the poles in Figs. 9.2 and 9.3, identified by the numbers 0.0 to 4.0 in the legend, indicate respectively: unstable, stable in frequency, stable in frequency and mode shape, stable in frequency and damping, stable in frequency damping and mode shape.

The results of the analyses and the generated stabilization diagrams are automatically stored in the results directory previously selected in the *Import data* tab.

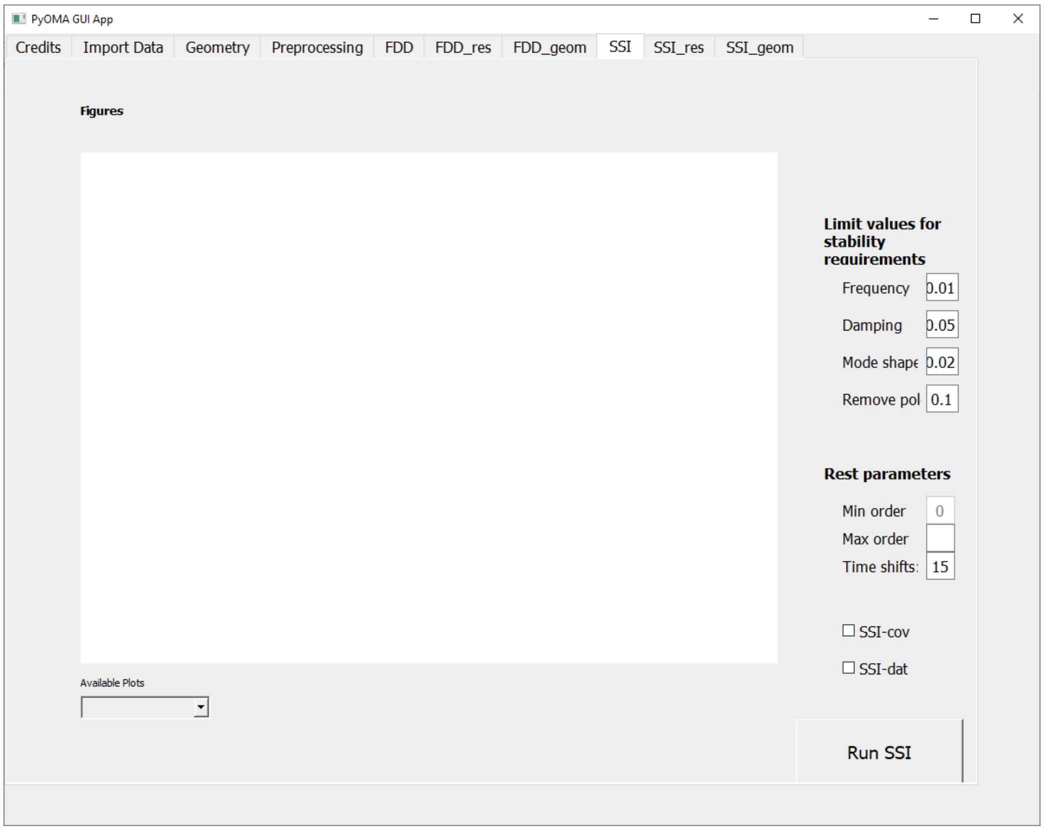


Figure 9.1: *SSI* tab.

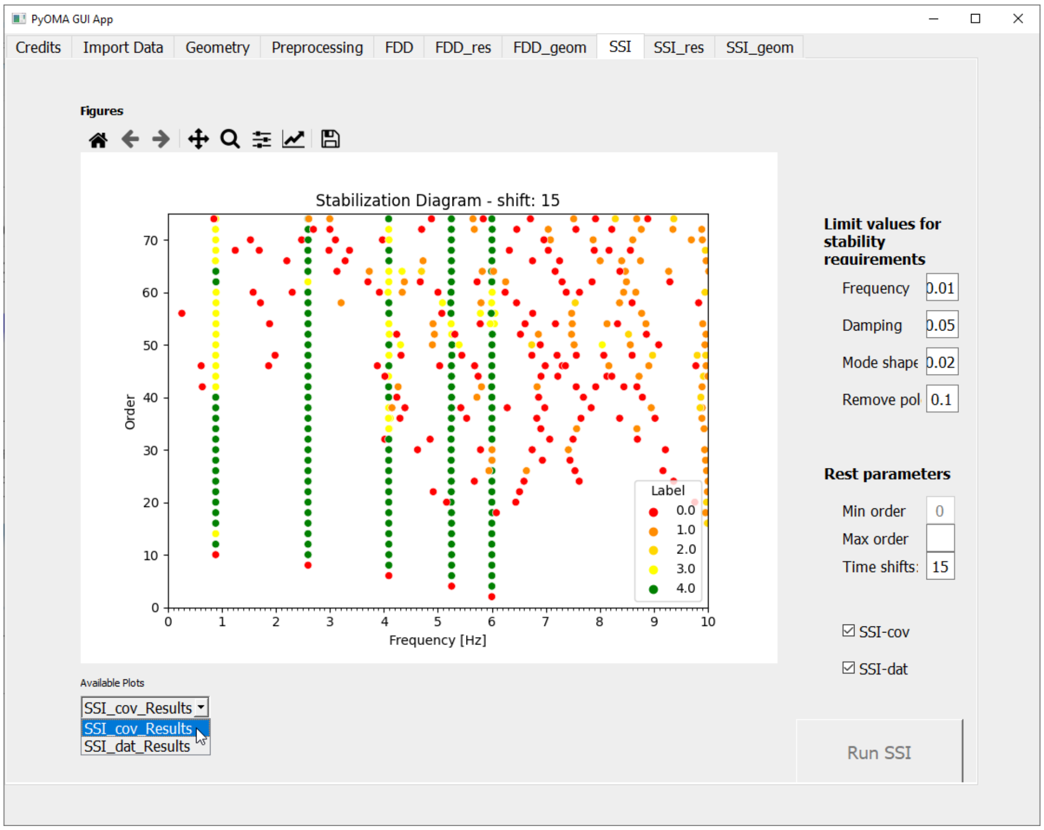


Figure 9.2: SSI-cov stabilization diagram.

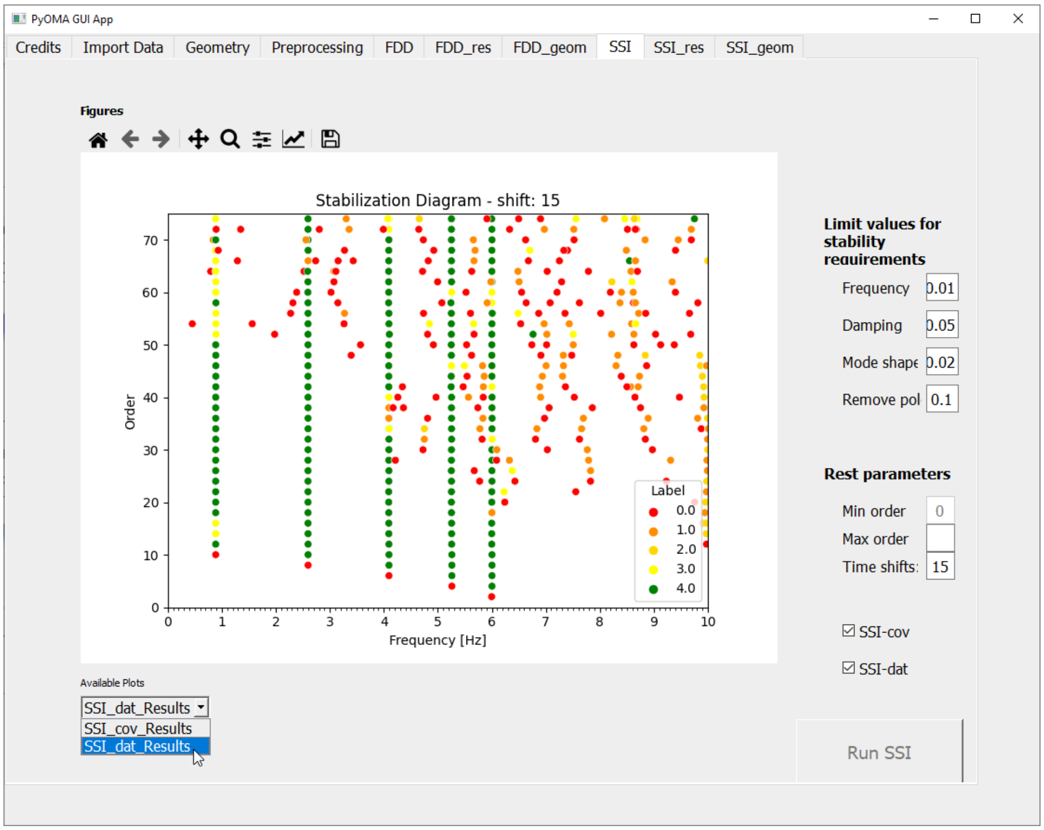


Figure 9.3: SSI-dat stabilization diagram.

10. SSI_res

In the current tab named *SSI_res* the user can inspect all the results derived from the *SSI* tab as depicted in Fig.7.1. The SSI results in terms of the natural frequencies of the system are referred to the identified alignments of the stable poles. These alignments are searched in the nearby of the natural frequencies values provided in the previous peak-peaking procedure.

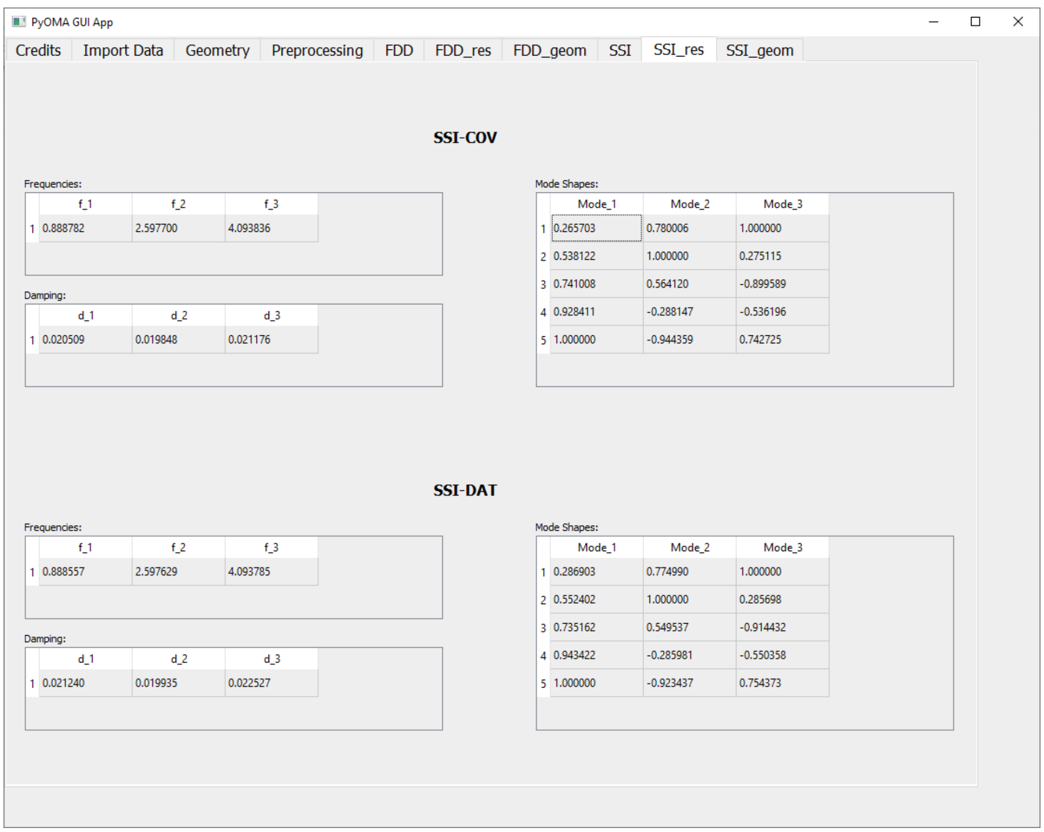


Figure 10.1: *SSI_res* tab.

11. SSI_geom

In the current tab named *SSI_geom* (Fig.11.1), the user can inspect the resulting mode shapes derived from the SSI-cov and/or SSI-dat analyses (Fig.11.2). The resulting mode shapes can be visualized as static deformations, with the possibility to show them in several forms. As already mentioned, the figure is **refreshed** only when the mode of interest is selected from the side dropdown menu. In the side table, the values of the selected mode shape are reported for each node and each DOF, whereas the selected mode shape is represented in a vectorial way depicting red arrows (Fig.11.3). Extra abilities for the graph are provided, such as the *deformed shape* checkbox which permits to visualize of the mode shape as a piecewise line (Fig.11.4), or the *show values plot* checkbox which print the values of the vectorial components directly on the graph (Fig.11.7). Four scaling factors are also provided: the first three scale factors provide the ability to scale the deformed shape components in the global x,y, and z directions respectively (Fig.11.5), whereas the last one permits to increase or decrease the thickness of the depicted mode shape arrows (Fig.11.6).

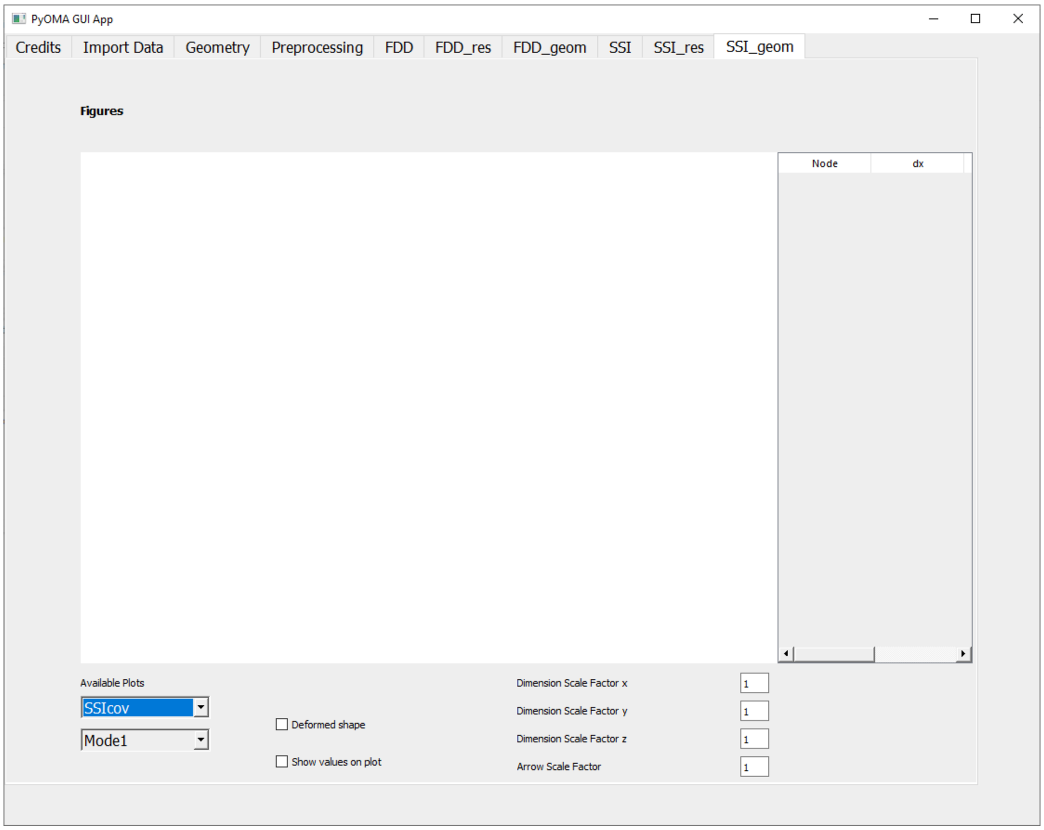


Figure 11.1: *SSI_geom* tab.

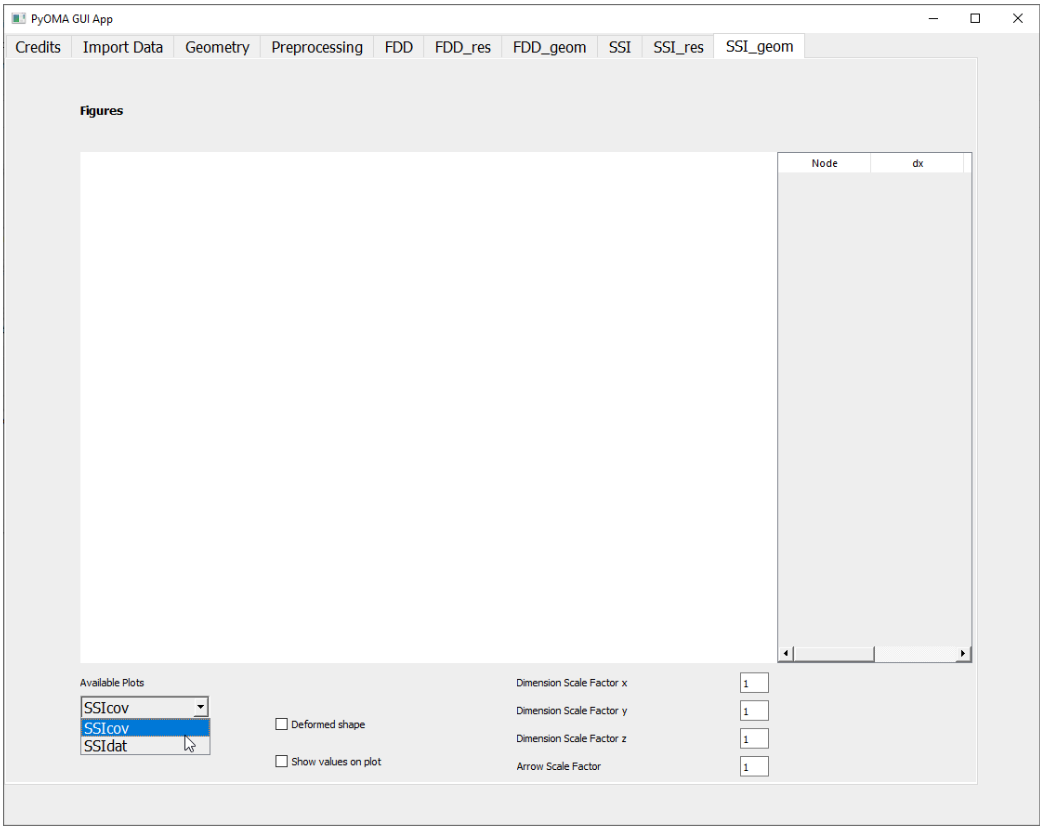


Figure 11.2: *SSI_geom* tab.

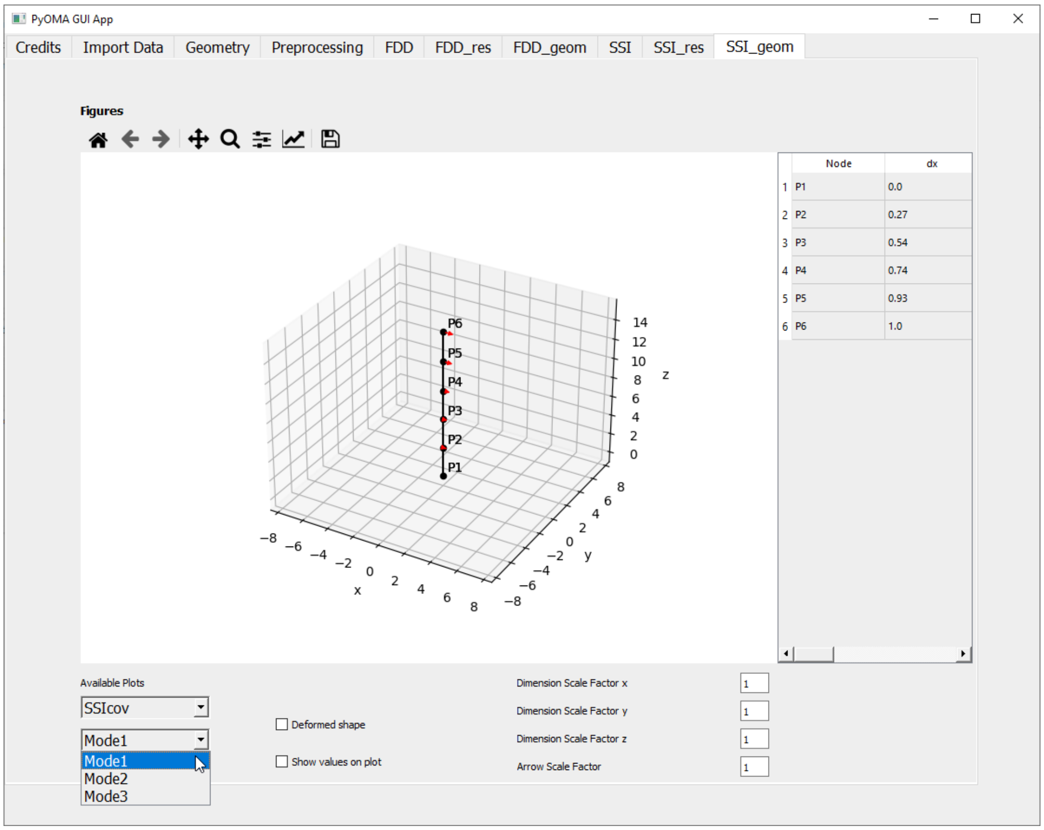


Figure 11.3: *SSI_geom* tab.

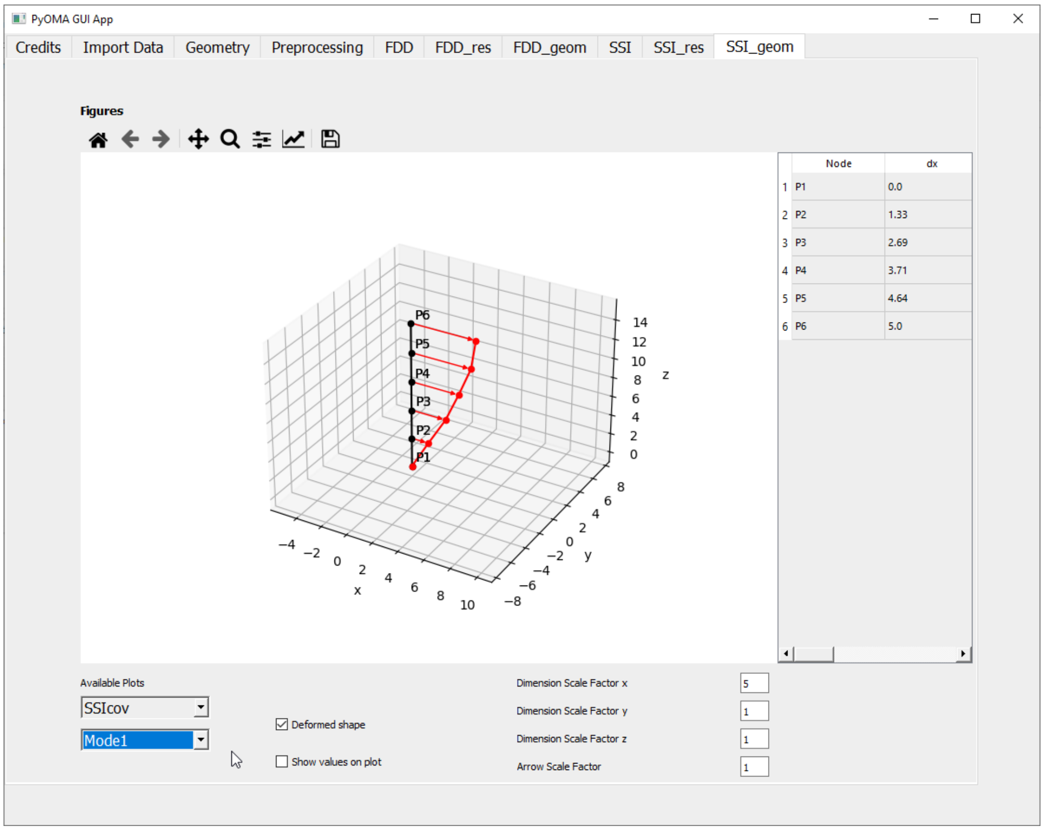


Figure 11.4: *SSI_geom* tab.

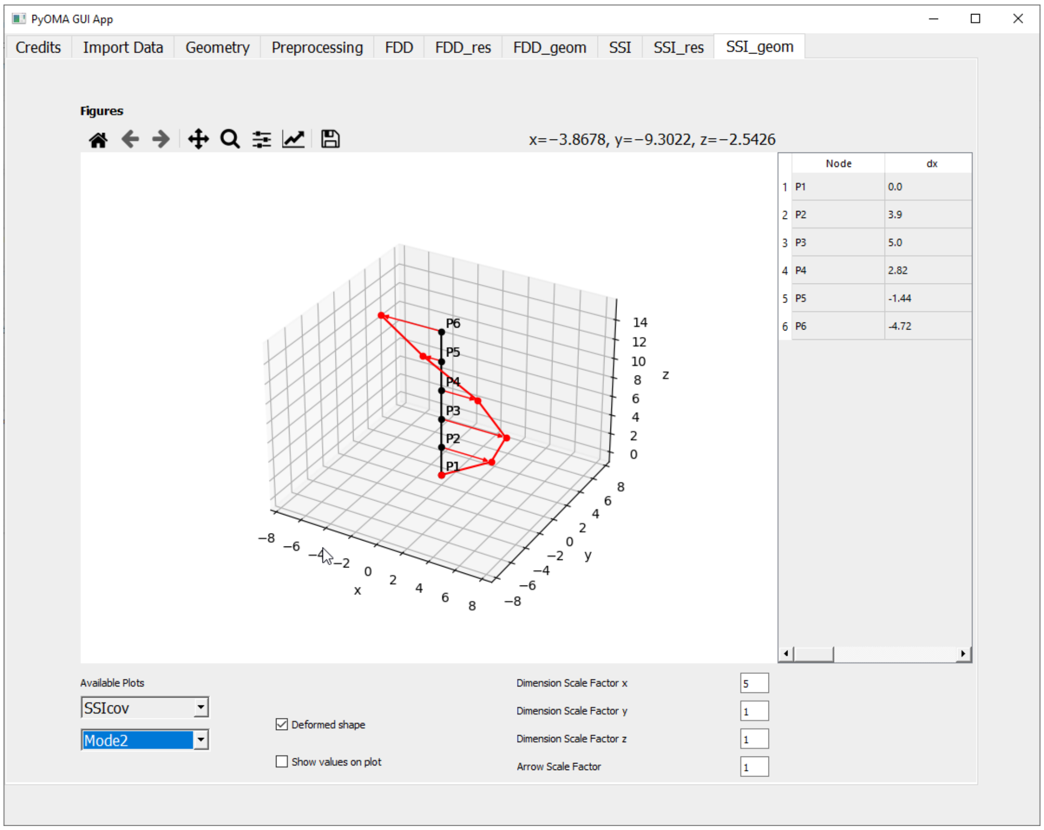


Figure 11.5: *SSI_geom* tab.

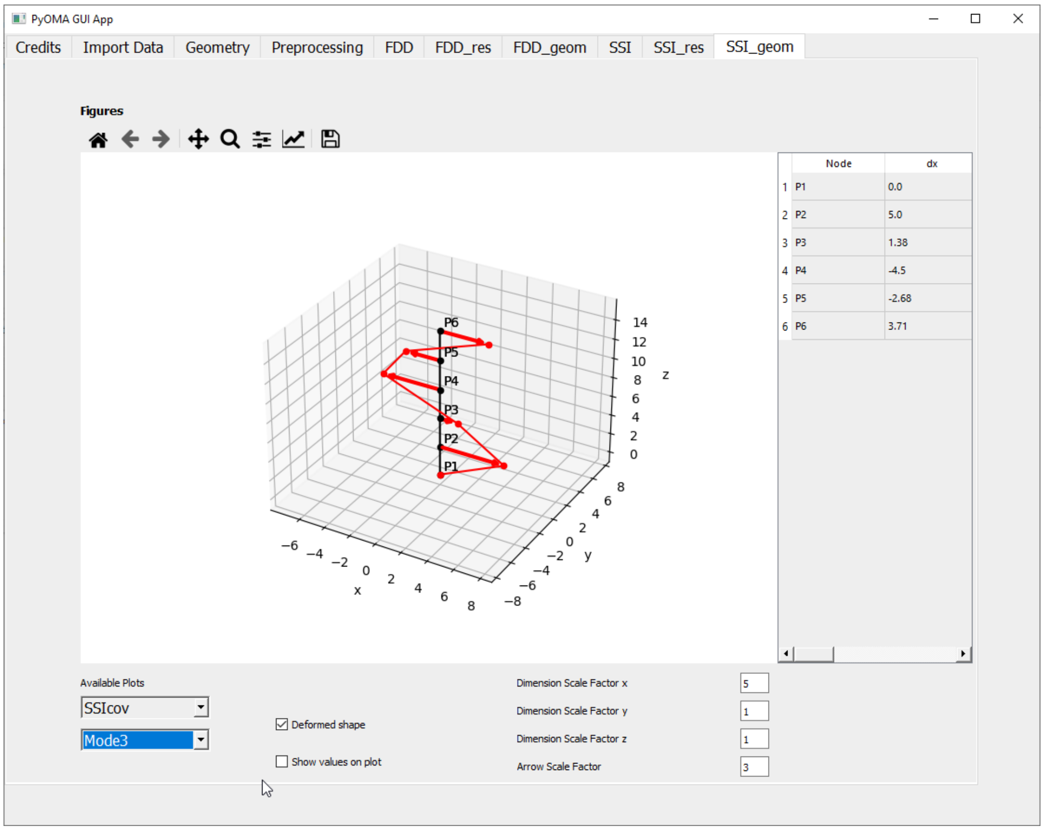


Figure 11.6: *SSI_geom* tab.

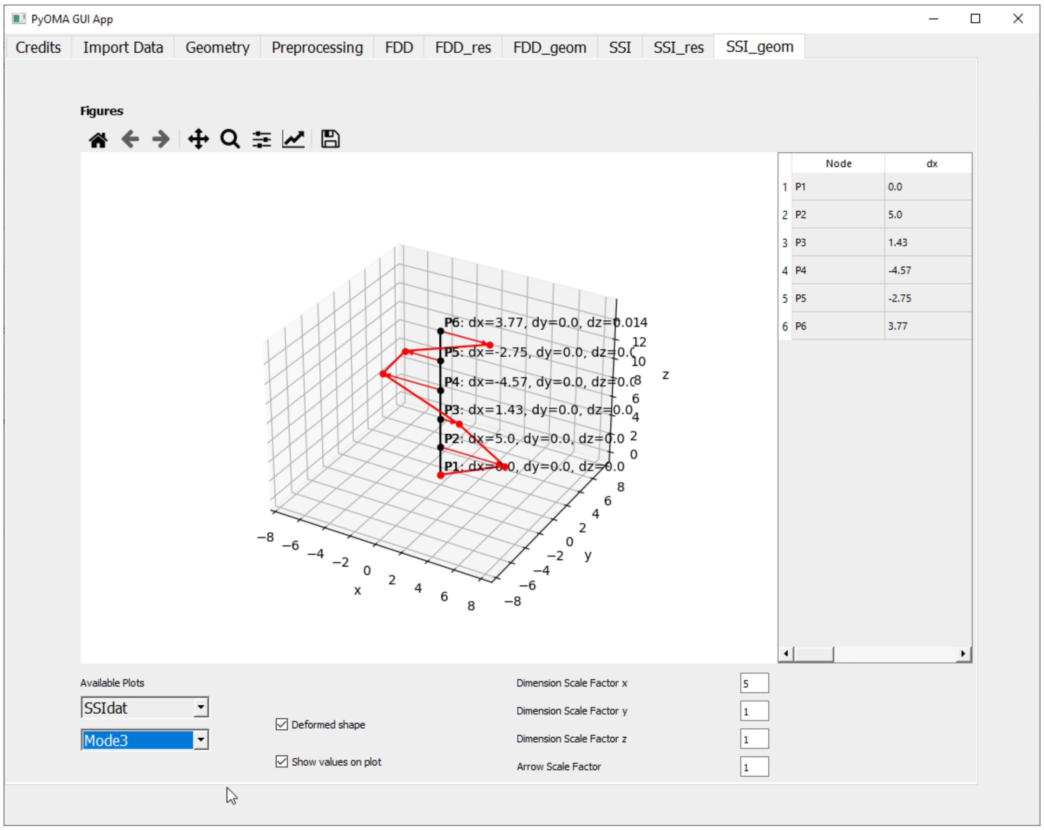


Figure 11.7: *SSI_geom* tab.

12. Example

The PyOMA_GUI has been implemented with PyQt5 library, adopting a notebook-style with tabs that noticeably help every type of user follow the right steps to perform OMA in a semi-assisted approach. In the following, the main steps in Fig. 2.2 are briefly described and discussed. However, with the software, the authors provided the files with the same example of the 5 Degrees of Freedom (DOFs) shear-type frame discussed in the previous sections.

12.1 Import Data

After running the executable PyOMA_GUI file, the user must select the first tab to import vibration data, as illustrated in Fig. 12.1. In the first place, the user must select the current working directory, i.e. a folder in which the output files will be stored. It can be in any location on the user's machine, and it can be an existing folder or a new one that may be created on the dialogue box. After that, the user must browse the input data file, which will also be displayed in the table to check the success of the data uploading. At the user's will, it is possible to customize the imported data table headers' names with a simple double click. The data visualization permits the user to check the number of time steps and the number of channels. To proceed with the following stages of the OMA procedure, the user must push the `Submit` button. Furthermore, to start a new analysis, the user may press the `Clear Set-up` button to completely reset the PyOMA_GUI software memory and cleanse the uploaded data.

12.2 Geometry

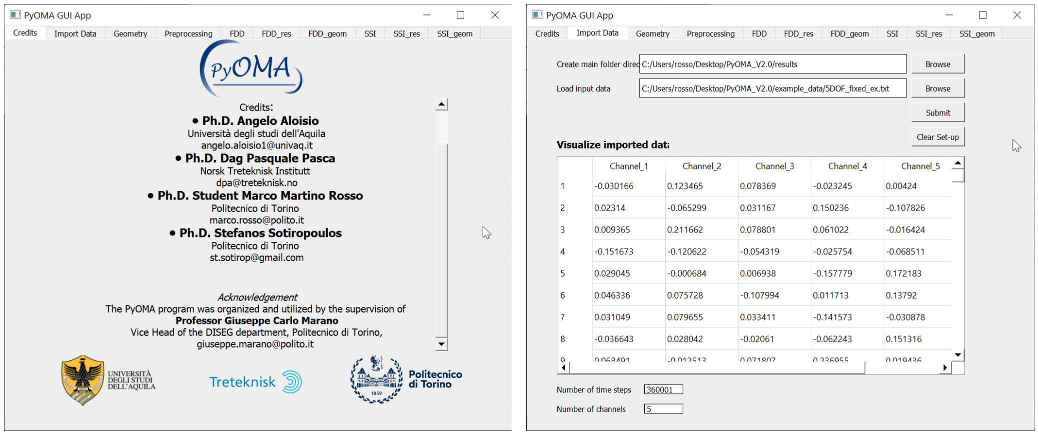
The geometry definition is fundamental to provide the user with a simplified visualization of the identified mode shapes according to the monitored DOFs and the available measurement channels, as depicted in Fig. 12.2 (a). The user must browse a text file which contains the nodes' coordinates and connectivity to visualize a starting undeformed scheme of the structure under study according to the monitored DOFs only. Pushing the button `Create Geometry` a wire-frame graph of the structure appears. In the table `Assign Channels to DOF`, the user must insert the exact index of the table (Channel_x, Channel_y, Channel_z) and the precise name of the monitored channel/DOF. Those identical names are shown in the combo-box Channels' name to remind the user, preventing him from returning to the previous imported data tab tediously. In the combo box, the exact assignments of the channels are provided.

12.3 Pre-processing and peak-peaking approach

The PyOMA_GUI allows the user to perform signal basic pre-processing procedures in the same manner as the PyOMA module, illustrated in Fig. 12.2 (b). The user must select the parameters of the problem, e.g. the sampling frequency and decimation factor and some default values are provided if the user does not set them explicitly. Thereafter, it is possible to run the `FDD_svp` function. In the SV graph figure, it is possible to perform the peak-peaking approach to select the peak of interest, and with the button `Add`, they are added to the list of the identified peaks. The other buttons close to the list permits customizing the identified peaks list, e.g. deleting a single item or clearing the whole list. With the `Submit` button, the user confirms the selection of the identified peaks, and it is possible to proceed with analyzing the results of the FDD approaches.

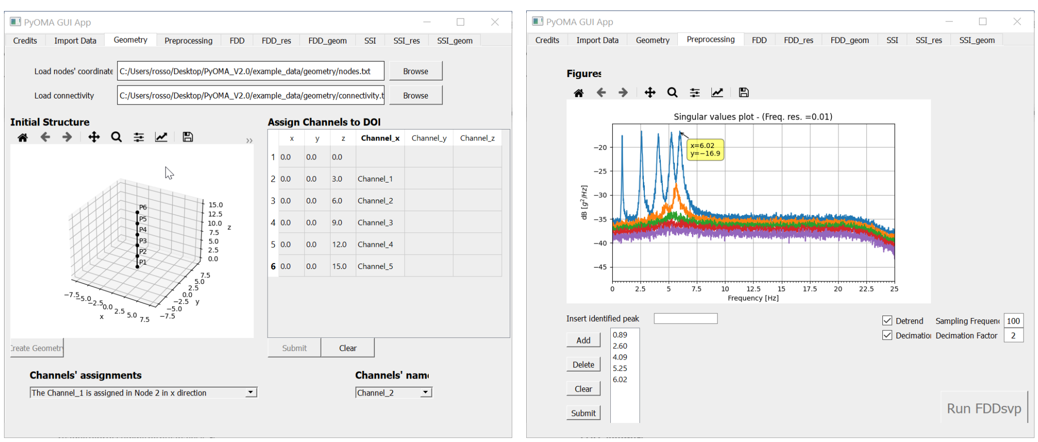
12.4 FDD, SSI and their implemented variants with PyOMA_GUI

Passing to the next tab of the notebook, it is possible to run the FDD, SSI and their implemented variants in the PyOMA module with the graphical user interface software. In the FDD tab, the user can choose in the checkboxes the methods of interest among FDD, EFDD and FSDD. Pushing the button Run FDD, the results are illustrated in terms of SDOF bell extraction for each mode. The available diagrams are listed in the dropdown menu in the bottom-left part of the window, and the above figure refreshes the user anytime to modify its selection. In the `FDD_res` tab, the user may explore all the frequency domain algorithms' results in tabular form. All the results (figures included) are automatically stored in the working directory previously selected. In the `FDD_geom` tab, the final mode shape deformations are presented in several forms, as depicted in Fig. 12.3. By selecting the desired mode, the figure is updated. In the table, the deformation on each node is presented, while extra functionalities for the plot are provided. The checkbox `Deformed shape` illustrates in the same figure the deformed structure, and the checkbox `Show values` activates the functionality to show the information of the table directly on the figure. At the same time, four scaling factors are also provided to adjust and customize the mode shape appearance. The last remaining tabs are related to the SSI method, which working mechanisms are similar and equivalent to the so-far illustrated frequency-domain methods. The stabilization diagram obtained with the PyOMA_GUI software is illustrated in Fig. 12.4.



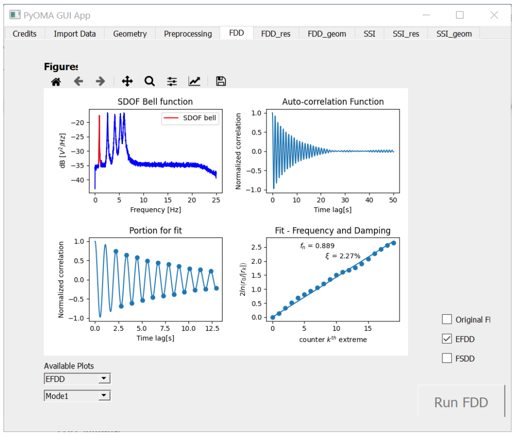
(a) (b)

Figure 12.1: PyOMA_GUI 5 DOFs shear type example overview. (a) Initial tab after starting the software execution; (b) Import data tab.

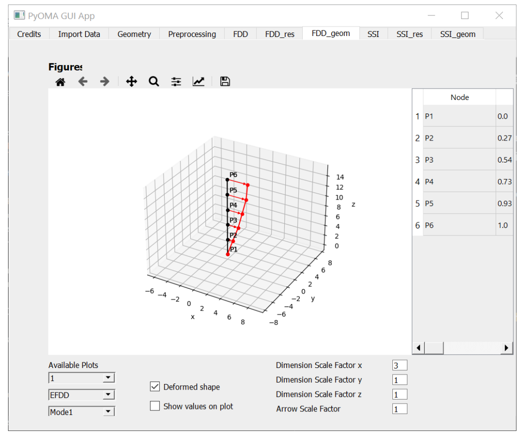


(a) (b)

Figure 12.2: PyOMA_GUI 5 DOFs shear type example overview. (a) geometry definition; (b) Preprocessing and FDD algorithm execution with SV decomposition diagram for peak-peaking identification.



(a)



(b)

Figure 12.3: PyOMA_GUI 5 DOFs shear type example overview. (a) geometry definition; (b) Preprocessing and FDD algorithm execution with SV decomposition diagram for peak-peaking identification.

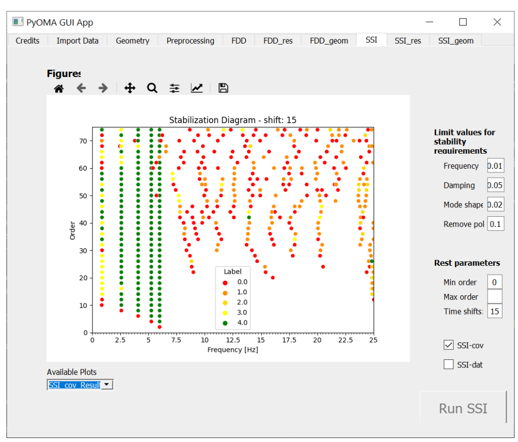


Figure 12.4: PyOMA_GUI 5 DOFs shear type example overview. Stabilization diagram retrieved with SSI algorithm.

13. Credits

13.1 Who we are

13.1.1 Head of ArtIStE research group: Prof. G.C. Marano

Figure 13.1: Prof. Giuseppe Carlo Marano

Professor Giuseppe Carlo Marano graduated cum laude in Civil Engineering at the Polytechnic University of Bari, Italy, in 1994. He spent two-year experience in Calcestruzzi s.p.a (Italcementi group), the Italian leader in concrete production. He got his Structural Engineering Ph.D. at the University of Florence in 2000, and postdoctoral fellow at Polytechnic of Bari in 2001, where achieved the associate professor position in 2011. He held the role of visiting assistant professor at Cambridge (2002), at Loughborough (2012), at Hunan University, Changsha, China (2014), and research fellow at SIBERC (Sustainable and Innovative Bridge Engineering Research Center) in Bari. In 2016, he became Structural Design full Professor at Fuzhou University, Fuzhou, China. Since 2018, he is full professor at Politecnico di Torino, Italy, and deputy director of the Department of Structural, Geotechnical, and Building Engineering. He authored four European patents and more than 300 articles in international journals or presented at conferences in structural engineering optimization, structural health monitoring, and impactful artificial intelligence civil applications.

13.1.2 Dag Pasquale Pasca

Figure 13.2: Ph.D. Dag Pasquale Pasca

Ph.D. Researcher at Norwegian Institute of Wood Technology - Norway

13.1.3 Angelo Aloisio

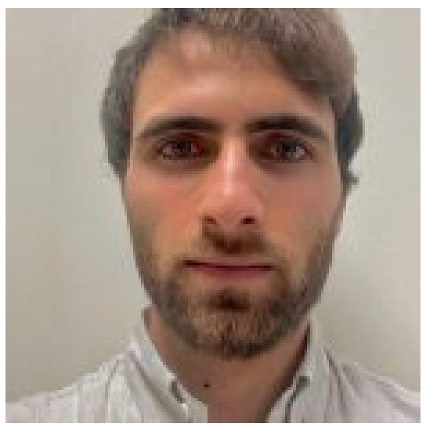


Figure 13.3: Ph.D. Angelo Aloisio

Ph.D. Research fellow Università degli Studi dell'Aquila - Italy

13.1.4 Marco Martino Rosso

Figure 13.4: Ph.D. Student Marco Martino Rosso

Ph.D. Student at Politecnico di Torino university - Italy

13.1.5 Stefanos Sotiropoulos

Figure 13.5: Ph.D. Stefanos Sotiropoulos

Ph.D. Visiting Staff at Politecnico di Torino university - Italy

13.2 How to contact us

If you have any issue, please feel free to contact us at our official e-mail address:
supportPyOMA@polito.it

13.3 How to cite

If you use this code, please don't forget to cite this work:

Dag Pasquale Pasca, Angelo Aloisio, Marco Martino Rosso et al., PyOMA and PyOMA_GUI: A Python module and software for Operational Modal Analysis. SoftwareX (2022) 101216, https://doi.org/10.1016/j.softx.2022.101216.

13.4 Acknowledgements

The developers acknowledge the meaningful contribution of Professor Rocco Alaggio from Università degli Studi dell'Aquila, who encouraged the authors to study and develop these topics. Furthermore, the developers acknowledge the meaningful contribution of Professor Giuseppe Carlo Marano from Politecnico di Torino for promoting the Graphical User Interface programming and coordinating the team activities.

Bibliography

[1] Carlo Rainieri and Giovanni Fabbrocino. Operational modal analysis of civil engineering structures. *Springer, New York*, 142:143, 2014.

[2] Dag Pasquale Pasca, Angelo Aloisio, Marco Martino Rosso, and Stefanos Sotiropoulos. Pyoma and pyoma_gui: A python module and software for operational modal analysis. *SoftwareX*, 20:101216, 2022.

[3] Lingmi Zhang, Tong Wang, and Yukio Tamura. A frequency–spatial domain decomposition (fsdd) method for operational modal analysis. *Mechanical systems and signal processing*, 24(5):1227–1239, 2010.

[4] Rune Brincker, Lingmi Zhang, and Palle Andersen. Modal identification of output-only systems using frequency domain decomposition. *Smart materials and structures*, 10(3):441, 2001.

[5] Rune Brincker, Carlos E Ventura, and Palle Andersen. Damping estimation by frequency domain decomposition. In *Proceedings of IMAC 19: A Conference on Structural Dynamics: februar 5-8, 2001, Hyatt Orlando, Kissimmee, Florida, 2001*, pages 698–703. Society for Experimental Mechanics, 2001.

[6] Peter Van Overschee and BL De Moor. *Subspace identification for linear systems: Theory—Implementation—Applications*. Springer Science & Business Media, 2012.

[7] Bart Peeters and Guido De Roeck. Reference-based stochastic subspace identification for output-only modal analysis. *Mechanical systems and signal processing*, 13(6):855–878, 1999.