Towards a cloud-based platform for Structural Health Monitoring: implementation and numerical issues*

Tiziana Croce[1], Maria Girardi[2], Gianmarco Gurioli[2], Cristina Padovani[2], and Daniele Pellegrini[2]

[1] Infomobility S.r.l., Via G. De Chirico 225, Rende (CS), Italy
t.croce@infomobility-italia.com
[2] Institute of Information Science and Technologies "A. Faedo" ISTI-CNR, Pisa, Italy
{maria.girardi,gianmarco.gurioli,cristina.padovani,daniele.pellegrini}@isti.cnr.it

Abstract. Structural Health Monitoring (SHM) is increasingly important in protecting and maintaining architectural heritage. Its main goal is to distinguish ordinary fluctuations in a building's response from other, possibly anomalous, behaviour. SHM starts setting sensors to measure accelerations or velocities and other environmental parameters over time at fixed points of the structure. The time-series processing makes it possible to perform modal tracking and damage/anomaly detection while correlating dynamical and environmental parameters. In practice, these activities are conducted separately, using different numerical codes. Thus, the idea is to take the first step to distance from such practice, leveraging the MOSCARDO system, which encompasses a Wireless Sensor Network (WSN) and a platform designed according to a cloud architecture that provides services for storing and processing data from the WSN. We employ a code based on the Stochastic Subspace Identification (SSI) technique to improve the system's capabilities, and we exploit the SSI's theoretical features to get an efficient implementation that will be integrated into the cloud-based platform. This pipeline is here presented considering data collected from a monitoring campaign on the "Matilde donjon" in Livorno (Italy) and reporting preliminary numerical results on the identification of the modal parameters.

Keywords: Structural Health Monitoring · Automated Operational Modal Analysis · Stochastic Subspace Identification · Modal Tracking.

1 Introduction and Main Motivations

Long-term dynamic monitoring of historical towers and buildings is aimed at identifying the modal parameters (natural frequencies, i.e. poles, modal shapes

* This research was funded by Infomobility S.r.l., ISTI-CNR and the Region of Tuscany through the fellowship programme "FSC Bando Assegni di Ricerca 2021" (STRENGTH project, 2022 - 2024). The support of CNR (Revolution Project, Progetti di Ricerca@CNR, 2022–2024) is gratefully acknowledged.

and damping ratios) of a structure and tracking them over time to detect possible anomalies in the vibrational features. To do that, it is common to work within the framework of Operational Model Analysis (OMA) [2] and exploit ambient vibration sources available on in-operation structures (crowd, wind, traffic, etc.). In this case, the input excitation is unknown and assumed to be a white Gaussian noise, thus ensuring that all the vibration modes are suitably excited.

This assumption is a basic hypothesis of output-only algorithms [2], common to both the (parametric) time-domain identification techniques, where outputs are estimated by means of correlation functions on a subset of the input dataset, and the (non-parametric) frequency-domain identification methods, where the processing of input data is achieved via Fourier's transforms.

Time-domain techniques have turned out to be more effective when dealing with civil engineering applications. Among several available techniques (AutoRegressive Moving Average model [5], AutoRegressive/Poly Reference model [14], Ibrahim Time Domain method [3]), the Stochastic Subspace Identification (SSI) [13] is nowadays the most popular to combine the advantages of the previous ones. The method, stemming from the Eigensystem Realisation Algorithm [4, 7], deals with a stochastic state-space model, in which the dynamic identification is performed without addressing nonlinear computations.

In this paper, we focus on the reference-based data-driven stochastic realization algorithm [9], which refers to the basic state-space formulation reported below for the k-th time instant:

$$\begin{aligned} x_{k+1} &= Ax_k + w_k \\ y_k &= Cx_k + v_k, \end{aligned}$$

in which the input is modelled by the noise terms $w_k \in \mathbb{R}^n$, $v_k \in \mathbb{R}^m$ ($n, m \in \mathbb{N}$), such that $\mathbb{E}[w_k] = 0 = \mathbb{E}[v_k]$, and covariance matrices

$$\mathbb{E}\left[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q^\top & v_q^\top \end{pmatrix}\right] = \begin{pmatrix} Q & S \\ S^\top & R \end{pmatrix} \delta_{pq},$$

being Q, R, S suitable matrices to describe the block structure of the covariance matrices, δ_{pq} the Kronecker's delta and with $\mathbb{E}$ denoting the expected value operator. A is the $n \times n$ state matrix, C is the $m \times n$ influence matrix, while $x_k \in \mathbb{R}^n$ (n being the model order) and $y_k \in \mathbb{R}^m$ are the state vector and the vector of observations at the k-th time instant, respectively. Moreover, the stochastic process is assumed to be stationary with zero mean ($\mathbb{E}[x_k] = 0$), the state covariance matrix $\Sigma = \mathbb{E}[x_k x_k^\top]$ is independent of the time index k and w_k, v_k are independent of the actual state, in the sense that $\mathbb{E}[x_k w_k^\top] = 0$, $\mathbb{E}[x_k v_k^\top] = 0$.

Although SSI methods are not totally free of drawbacks, their practical embedding into an automatic identification framework is hindered by a certain amount of interaction required to the user, consisting in the manual and subjective selection of the meaningful natural frequencies of the structure under examination from the stabilization diagrams. Moreover, such a non automatic feature significantly increases the elapsed time effort when a great amount of

This is an original manuscript of an article published by Springer in Lecture Notes in Civil Engineering - Experimental Vibration Analysis for Civil Engineering Structures EVACES 2023 - Volume 1, available at: https://doi.org/10.1007/978-3-031-39109-5

data has to be processed. These considerations have been formerly taken over in [11], but the development of a fully automatic SSI framework is still an open issue.

The main aim of this paper is thus to take the first step towards an automatic use of the reference-based data-driven SSI algorithm, following two subsequent stages. At first, an efficient implementation of the algorithm is carried out, grounded on state–of–art software. Then, the implementation is embedded into an unsupervised hierarchical clustering strategy, to automatically select the stable eigenfrequencies and the corresponding mode shapes without any actions needed by the user, after setting some parameters described below. This even paves the way for integrating such a strategy into the cloud-based platform of the MOSCARDO system [15] and enrich its capabilities, that is our end as well.

The paper is organized as follows. The next section recalls the MOSCARDO system. Section 3 presents the developed automatic OMA framework, applied in Section 4 to data collected during the monitoring campaign on the "Matilde donjon" in Livorno (Italy) [1] . Some numerical results on the identification of the tower's modal properties are shown and compared with those given by the commercial software MACEC [10]. Section 5 states conclusions and perspectives.

2 The MOSCARDO System

MOSCARDO is a system for the structural health monitoring of ancient constructions. It integrates Information and Communication Technologies (wireless sensor networks, signal processing and computer vision) with methodologies coming from engineering and computational mechanics, to develop complex predictions and promptly operate if needed. The system was developed within the MOSCARDO project conducted by Infomobility S.r.l., Engineering Italy Solutions S.r.l., ISTI-CNR and the University of Florence from April 2016 to October 2018 and funded by the Region of Tuscany and the Italian Ministry of Education, University and Research (PAR–FAS 2007–2013).

The MOSCARDO infrastructure is composed by Wireless Sensor Networks (WSNs) for the acquisition of structural (accelerations and displacements at prescribed points of the structure) and environmental data (pressure, humidity and temperature), coupled with a flexible and reliable Internet of Things communication infrastructure. A further essential component of the system is the so-called Monitoring Control Center (MCC), a cloud-based platform to provide services for storage, processing, and interpretation of data coming from the WSNs. The MCC architecture collects and stores large amounts of data to monitor the structure in time and provide continuous supervision.

Our goal is the development of automated management tools that allow tracking in time the structural behaviour of the monitored building and obtaining prompt information on any damage in standard conditions or during events of great significance (such as crowded demonstrations or earthquakes). In fact, such a capability is currently carried out offline and is not yet integrated within the

infrastructure. The goal is thus equipping the MCC platform with online SHM tools that do not require actions by the user throughout the running.

3 An Automatic OMA Pipeline for Dynamic Identification of Modal Parameters

The developed OMA framework consists of the following steps. As the MCC is a Python–based platform, we have implemented the first three steps by using Python 3.9.13, while the last two are based on MATLab R2019b.

Step 1. Dataset recovery, assembling and storing. The structure is provided with sensors located at different positions measuring one or more quantities with a prescribed sampling frequency. Thus, at each time slot, a total number of n_c time series of length n_t (n_c, $n_t \in \mathbb{N}$) is measured on the structure. Let us consider time slots lasting 1 hour. The first step of the pipeline consists in acquiring the times series in the right format and store them into a $n_c \times n_t$ matrix that will constitute the input dataset. To this aim, isolated missing data in each time series are restored by using a linear interpolation, while in case of massive missing data the registered time-series is ignored.

Step 2. Pre-processing of the input dataset. The input dataset is then pre-processed via detrending and filtering operations in a suitable frequency band, applying two fourth-order low-pass/high-pass Butterworth filters.

Step 3. Dynamic identification for the stabilization diagram visualization. This step is devoted to the Python implementation of SSI solvers. Specifically, we have implemented the already mentioned reference-based covariance driven stochastic realization and the reference-based data-driven stochastic realization algorithms [9, 8], whose implementations are hereafter referred to as Python SSI/cov and Python SSI/dat, respectively. We have chosen to follow the algorithms in [9] that clearly provide hints on how to computationally take advantage of the factors Q and R of the $QR-$ factorization of the block Hankel matrices needed for the projection procedure in the SSI/dat method. The obtained Python solvers take the pre-processed dataset and the sampling frequency as the input, together with suitable specifications to build the stabilization diagram (i.e., maximum and minimum order method, relative accuracy on the frequency, damping ratio and MAC value to verify the stability of the poles, maximum tolerance on the damping ratio), returning and storing the stabilization diagram and the list of stable poles with related damping ratios, frequencies and mode shapes vectors.

Step 4. Automatic estimation of the modal parameters and results export. Once the stabilization diagram is computed, we need to automatically

select the most representative poles. For such a reason, the list of poles resulting from the previous phase is detached by an agglomerative hierarchical clustering strategy ([6, 12]), that proceeds as follows. i) The matrix with the pairwise Euclidean distances between pairs of poles of the list is computed. ii) Such a matrix is used for a linkage procedure to obtain a new matrix that encodes a bottom-up tree (dendrogram) containing the hierarchical clusters of the list of poles, following the shortest distance criterion to compute the distances between clusters. iii) Clusters are constructed from the agglomerative hierarchical cluster tree of the linkage phase using a specified cutoff parameter, coming out with the cluster assignments of each pole in the list. For each cluster, the centroid is then computed to identify the resulting closest stable pole, namely the "representative" stable pole. Finally, the stable poles are listed and stored, together with the corresponding damping ratios and mode shapes.

Step 5. Post-processing for modal tracking and visualization of the results over time. The above four steps are repeated for the measurements of each time slot, in order to perform modal tracking and plot the representative stable poles with respect to time. Additionally, few post-processing operations have been considered for a suitable visualization of the results. They are shown in the figures, tables and comparisons of the next section.

4 Numerical Tests

The automatic OMA pipeline described in the previous section has been tested on the "Matilde donjon" in the Old Fortress of Livorno (Tuscany, Italy). The tower's accelerations are measured by one radial and two tangential accelerometers at the entrance of the structure (Figure 1, level 0), while at level 2 an additional radial and three tangent accelerometers are placed, for a total number of seven measurement channels. More details on the monitoring system can be found in [1]. The sensors, which were installed in October 2018 and are still running, are set to measure for 15 minutes per hour. These sensors are grouped in nodes that transmit the acceleration values via wireless to a gateway that sends the data to a remote server that hosts the MCC. A representation of the cross-sections of the tower and the exact location of the sensors are reported in Figure 1.

With reference to Steps 1–4 of Section 3, the channels number $n_c = 7$ and the number of data samples values $n_t = 45000$ at each time slot are considered at Step 1, the sampling frequency is 50 Hz, the frequency band for the Butterworth filters to apply at Step 2 is 0.3–10 Hz, while 2 and 80 are the minimum and maximum model order needed in the input of Step 3. In addition, 0.01, 0.05, 0.02 are the relative errors allowed on frequency, damping and mode shapes, respectively, being 0.5 the upper bound on the damping ratio for a stable pole to be accounted; a cutoff distance value of 0.1 has been considered at Step 4 and the clusters with less than 5 elements have been neglected. Concerning this point, we notice that the cutoff distance value is a hyperparameter for our method, here

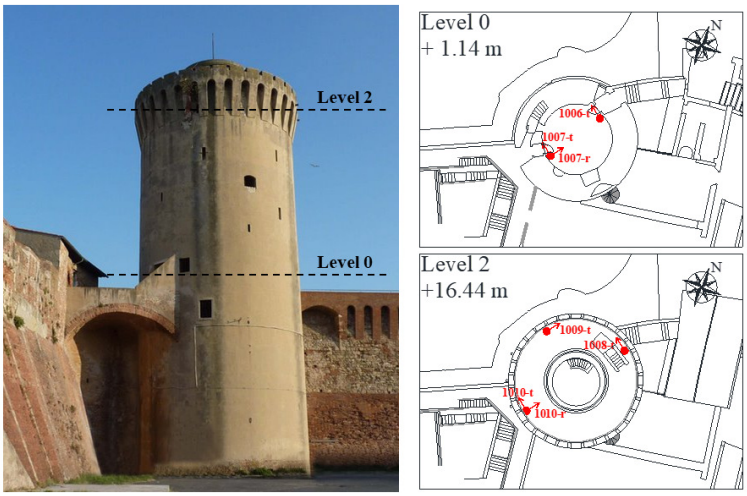

Fig. 1. Longitudinal view and cross section of the tower with the locations of the sensors: radial (r) and tangent (t) accelerometers (red).

tuned by using the results of the identification tests conducted on the tower in March, 2019, [1].

We have performed modal tracking focusing on the first year of monitoring (2019), using the Python solver SSI/dat and the corresponding method of MACEC [10], hereafter named as MACEC SSI/dat. The results obtained via Python SSI/cov and MACEC SSI/cov are here omitted for the sake of brevity. The Matlab algorithm for clustering developed in Step 4 has been also applied to the results of the MACEC stabilization diagrams, to avoid manual intervention by the user and fully automate the modal tracking procedure. The results are shown in Figures 2–3. Figure 2 compares the plot of the first two natural frequencies of the tower obtained via MACEC SSI/dat (left) and Python SSI/dat (right) in March, June, September and December 2019. The results obtained with the two codes are similar, except for the fact that the right-hand side picture of Figure 2 is more dense. This could be attributable to the fact that the MACEC solver consider more strict criteria to build the stabilization diagram. The detection of the two lowest natural frequencies, whose average values are shown in Table 1, is in accordance with the results given in [1]. The plots also show the number of sensors for which Step 1 ends up with a complete time series, addressed as the "active" sensors.

Moreover, an accurate look at the modal tracking of the two lowest frequencies highlights that the frequency values tend to increase over the hottest months

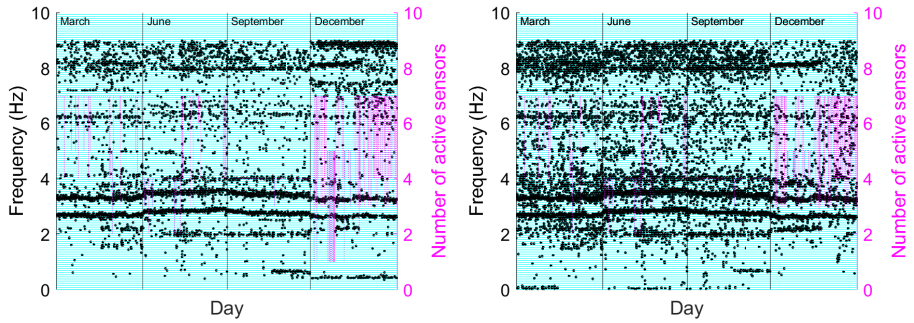

Fig. 2. Modal tracking via MACEC (left) and Python (right) solvers SSI/dat (black points) over the daily slot of hours of March, June, September and December, 2019. The auxiliary right y-axis refers to the number of active sensors, whose plot over the daily slot of hours is the magenta dashed line.

of the year. Figure 3 shows the daily variation of the two lowest frequency values due to the change of temperature measured by a meteorological station on top of the tower.

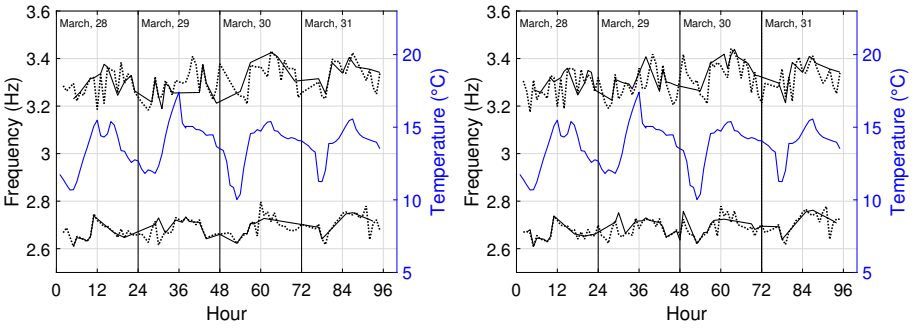

Fig. 3. Modal tracking of the two lowest frequencies of vibrations via MACEC (black line) and Python (black dashed line) solvers SSI/cov (left) and SSI/dat (right) over the daily slot of hours of March 28–31, 2019. The auxiliary right y-axis refers to the measured temperatures, whose plot over the daily slot of hours is marked as the blue line.

The monthly average values of the two fundamental frequencies and corresponding damping ratios are reported in Table 1, showing a good agreement between the MACEC and Python solvers. The relative errors of the two first frequencies calculated via MACEC and Python are reported in Figure 4; the

This is an original manuscript of an article published by Springer in Lecture Notes in Civil Engineering - Experimental Vibration Analysis for Civil Engineering Structures EVACES 2023 - Volume 1, available at: https://doi.org/10.1007/978-3-031-39109-5

errors belong to the interval $[10^{-6}, 5 \cdot 10^{-1}]$ and, for the greatest part, are lower than 10^{-2}. Finally, a comparison of the computation time for Step 3 highlights the better performance of Python with respect to MACEC, with a mean ratio of 0.38. The procedures were run on an Intel(R) Core(TM) i7, 2.67 GHz $\times$ 1 CPU, 18 GB RAM.

Table 1. Estimation of the two lowest frequencies of vibrations with related damping ratios via MACEC/Python SSI/dat in March, June, September and December, 2019.

Method	Mode 1		Mode 2	
	Freq. (Hz)	Damp. (%)	Freq. (Hz)	Damp. (%)
MACEC (March)	2.6791	3.63	3.3072	3.65
Python (March)	2.6774	3.42	3.3001	3.41
MACEC (June)	2.8381	3.92	3.5095	4.24
Python (June)	2.8413	3.62	3.5050	4.21
MACEC (September)	2.7658	3.37	3.4302	3.94
Python (September)	2.7735	3.28	3.4253	3.85
MACEC (December)	2.6810	3.78	3.3020	3.79
Python (December)	2.6936	3.81	3.3137	3.46

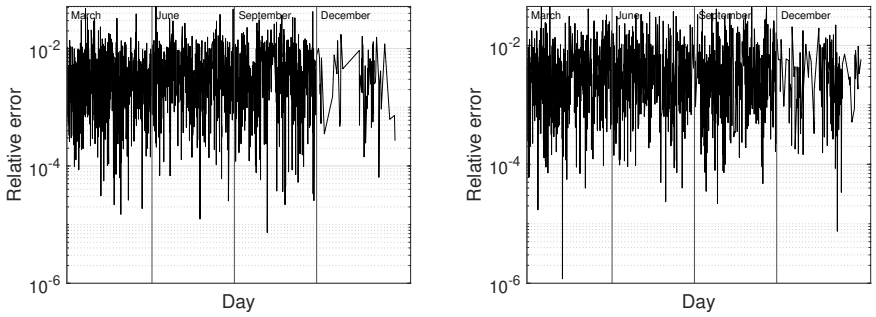

Fig. 4. Relative errors of the first (left) and second (right) lowest frequency of vibrations via Python SSI/dat, with respect to MACEC SSI/dat, versus the daily slot hours of March, June, September, December, 2019. Logarithmic scale is adopted on the y-axis.

5 Conclusions

In this paper we integrated the reference-based data driven stochastic algorithms, using a Python implementation of the methods in [9], with agglomerative hierar-

chical clustering methods for modal tracking. This implementation gives a first contribution towards the development of a fully automated platform for Operational Modal Analysis. The numerical tests performed on the case study of the "Matilde donjon" in Livorno show good results. In particular, the comparison between the Python solver and the commercial software MACEC highlights the consistency of the identification results and a substantial saving in terms of computational time. These results make indeed possible the effective integration of our strategy into the Monitoring Control Center of the MOSCARDO platform mentioned in Section 2, that is our final plan for future work.

To this scope, several aspects should be further investigated, as the use of adaptive methods to automatically tune the cutoff distance in the clustering algorithm and the possibility of testing other clustering techniques. Moreover, the set of criteria considered by the Python solvers to plot the stabilization diagrams is rather essential and could be enlarged. Finally, another key aspect is the use of suitable strategies of anomaly detection, even investigating the use of deep learning techniques.

References

1. Barsocchi, P., Bartoli, G., Betti, M., Girardi, M., Mammolito, S., Pellegrini, D., Zini, D.: Wireless sensor networks for continuous structural health monitoring of historic masonry towers. International Journal of Architectural Heritage **15**(1), 22–44 (2021).
2. Brincker, R., Ventura, C.: Introduction to operational modal analysis. John Wiley & Sons (2015).
3. Ibrahim, S. R., Mikulcik, E. C.: A method for the direct identification of vibration parameters from the free response. The Shock and Vibration Inform. Ctr. Shock and Vibration Bull. Part. 4, vol. 47, pp. 183–196 (1977).
4. Juang, J. N., Pappa, R. S.: An eigensystem realisation algorithm for modal parameter identification and modal reduction. Journal of guidance, control, and dynamics **8**(5), 620–627 (1985).
5. Ljung L.: Theory for the User, System Identification (1987).
6. Murtagh, F., Contreras, P.: Algorithms for hierarchical clustering: an overview. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **2**(1), 86–97 (2012).
7. Pappa, R. S., Elliott, K. B., Schenk, A.: Consistent-mode indicator for the eigensystem realization algorithm. Journal of Guidance, Control, and Dynamics **16**(5), 852–858 (1993).
8. Pasca, D. P., Aloisio, A., Rosso, M. M., Sotiropoulos, S.: PyOMA and PyOMA_GUI: A Python module and software for Operational Modal Analysis. SoftwareX **20**, 101216 (2022).
9. Peeters, B., De Roeck, G.: Reference-based stochastic subspace identification for output-only modal analysis. Mechanical systems and signal processing **13**(6), 855–878 (1999).
10. Reynders, E., Schevenels, M., De Roeck, G.: MACEC 3.4: The Matlab toolbox for experimental and operational modal analysis. User's manual. Katholieke Universiteit, Leuven (2021).

11. Reynders, E., Houbrechts, J., De Roeck, G.: Fully automated (operational) modal analysis. Mechanical systems and signal processing **29**, 228–250 (2012).
12. Sasirekha, K., Baby, P.: Agglomerative hierarchical clustering algorithm-a. International Journal of Scientific and Research Publications **83**(3), 83 (2013).
13. Van Overschee, P., De Moor, B.: Subspace algorithms for the stochastic identification problem. Automatica **29**(3), 649–660 (1993).
14. Vold, H., Kundrat, J., Rocklin G. T., Russell, R.: A multi-input modal estimation algorithm for mini-computers. SAE Transactions, 815–821 (1982).
15. MOSCARDO Project Homepage, http://www.moscardo.it (in Italian).