Analysis and optimization of a hybrid testing methodology using a small-scale set-up

18TH WORLD CONFERENCE ON EARTHQUAKE ENGINEERING
WCEE2024
MILAN

MILAN, ITALY
30TH JUNE - 5TH JULY 2024
www.wcee2024.it

ANALYSIS AND OPTIMIZATION OF A HYBRID TESTING METHODOLOGY USING A SMALL-SCALE SET-UP

G. van Vugt[1], J. Henriques[2], B. Vandoren[2] & H. Degée[2]

[1] Hasselt University, Hasselt, Belgium, glenn.vanvugt@uhasselt.be

[2] Hasselt University, Hasselt, Belgium

Abstract: *Hybrid testing provides an efficient and less costly way to explore the response of structural systems to realistic dynamic or seismic loading. However, the required equipment to execute hybrid tests are high-cost tools. To get insight in the hybrid testing methodology, a small-scale set-up has been developed in this project.*

An Arduino UNO controls the system that imposes the displacement to a linear actuator. Connecting the small-scale set-up, i.e., the Arduino UNO, to a PC allows imposing a time history to the physical substructure. A load cell measures the restoring force which will be communicated to the PC by the Arduino UNO using the serial monitor. Numerical integration based on the Gravouil-Combescure scheme with Classic Lagrange Multipliers (CLM) determines the displacement for the next time step. A correction on the measured restoring forces is applied to mitigate experimental errors.

This paper describes hot spots of the methodology and the results of a demonstrative experimental test using a MATLAB and Python implementation. The experiment consists of a 4 degree of freedom (DOF) numerical model combined with a 1 DOF physical specimen. The installed linear actuator only has one gearing option, which leads to a possible overshooting loop.

Interesting conclusions can be drawn from the analysis of the small-scale set-up in view of its future upscaling and implementation of the hybrid test method at laboratory scale. Firstly, the linear actuator requires a non-negligible amount of time to reach the imposed displacement which imposes boundary conditions in the directives. Secondly, a velocity-controlled actuator is essential in the exploitation of hybrid testing. Thirdly, the displacement tolerance influences the stability of the system. If one increases the displacement tolerance, the risk of an overshooting loop decreases. However, the accuracy might be influenced. Good balance must therefore be found between stability and accuracy.

1 Introduction

1.1 Hybrid test principles

This paper details the development of a small-scale set-up to execute low-cost hybrid tests. The term hybrid test covers physical-numerical simulations of seismic response, in which a part of a structure is isolated and tested experimentally. This specimen, or physical substructure (PS), contains a region of interest while the remainder of the structure, the numerical substructure (NS), is simulated numerically. A time stepping analysis algorithm solves the coupled equations of motion, which determines the displacement to impose on the PS. The measured restoring forces serve as input values for the next time step.

Hybrid tests offer a powerful way of verifying the performance of seismic-resisting structural systems and can provide valuable data for the development and calibration of nonlinear numerical models of structures and elements (McCrum & Williams, 2016).

Full-scale hybrid test equipment contains expensive tools. Furthermore, preparing, executing and break down a full-scale hybrid test requires a high amount of time, space and resources. Hence, a small-scale set-up provides an ideal opportunity to get acquainted with hybrid testing without wasting resources.

1.2 Background of hybrid testing

McCrum and Williams (2016) issues the history of different hybrid testing methods, e.g., pseudo-dynamic testing, real-time hybrid testing (RTHT) etc. Takanashi et al. (1975) introduced conventional pseudo-dynamic testing, while Dermitzakis and Mahin (1985) defined the concept of substructured pseudo-dynamic testing. Pseudo-dynamic testing entails the application of slowly varying forces to a structural model.

Numerical integration techniques compute displacements for the next time step, which are then imposed on the PS. Once the computed displacements are applied, restoring forces will be measured and utilized to compute the next time step. Figure 1 illustrates the substructure pseudo-dynamic test method. This approach permits to physically examine only the most critical zone of a structure, resulting in cost reduction compared to quasi-static testing. Pseudo-dynamic testing is time-consuming, as each time step requires computational determination, which allows relaxation. Therefore, pseudo-dynamic testing only applies for rate-independent experiments.

RTHT provides the possibility for rate-dependent experiments. Nonetheless, RTHT introduces control issues, i.e., delay because of numerical simulation, communication protocols and actuator properties.

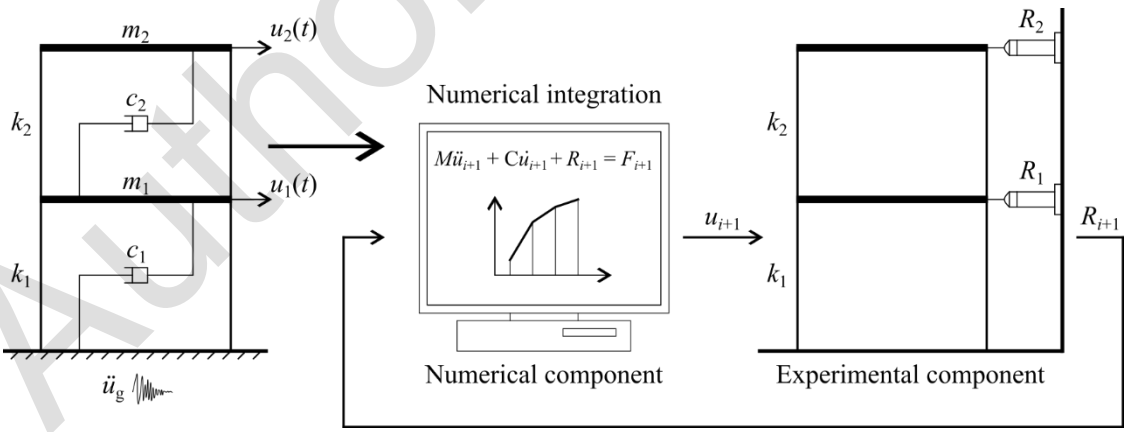


Figure 1. Schematic of pseudo-dynamic test method (McCrum and Williams, 2016).

The equation of motion, as formulated in Eq. (1), is solved over a series of time steps Δt, i.e., at the (i + 1)th step of:

$$\boldsymbol{M}\ddot{\boldsymbol{u}}_{i+1} + \boldsymbol{C}\dot{\boldsymbol{u}}_{i+1} + \boldsymbol{R}^{\mathrm{N}}_{i+1} + \boldsymbol{R}^{\mathrm{P}}_{i+1} = \boldsymbol{F}_{i+1} \quad (1)$$

where $\boldsymbol{M}$ is the mass matrix, $\ddot{\boldsymbol{u}}_{i+1}$ the nodal acceleration vector, $\boldsymbol{C}$ the damping matrix, $\dot{\boldsymbol{u}}_{i+1}$ the nodal velocity vector, $\boldsymbol{R}^{\mathrm{N}}_{i+1}$ the restoring force vector for the NS, $\boldsymbol{R}^{\mathrm{P}}_{i+1}$ the restoring vector for the PS, $\boldsymbol{u}_{i+1}$ the nodal

displacement vector and $\boldsymbol{F}_{i+1}$ the external excitation force applied to the system. For a linear NS, $\boldsymbol{R}_{i+1}^{\mathrm{N}} = \boldsymbol{K}^{\mathrm{N}}\boldsymbol{u}_{i+1}^{\mathrm{N}}$($\boldsymbol{K}^{\mathrm{N}}$ is the NS stiffness matrix and $\boldsymbol{u}_{i+1}^{\mathrm{N}}$ is the NS nodal displacement vector).

Pseudo-dynamic testing involves the physical measurement of the stiffness term while modelling inertial and damping forces numerically. Thus, pseudo-dynamic testing requires a slow loading rate to not induce damping or inertia responses. An extended timescale is used to execute a pseudo-dynamic test, providing time required for numerical integration of the equation of motion and communicating and imposing the resulting displacements by the actuators. Therefore, time-dependent behaviour is not recorded.

Conventional pseudo-dynamic consists of a ramp-hold load pattern. Magonette (2001) introduced the continuous pseudo-dynamic method, allowing the actuator to avoid the hold period. However, in the early stages of its development, continuous hybrid testing ran into overshooting issues (Takanashi (1987), Takanashi and Fenves (2006)).

1.3 Experimental errors

Shing and Mahin (1987) observed that experimental errors tend to increase during the step-by-step integration process. This study revealed that the rate of cumulative error growth is significantly influenced by the natural frequency of the specimen and the integration time step. Particularly, as the natural frequency and time step increase, the rate of cumulative error growth also expands, signifying an elevated sensitivity to experimental errors in higher modes of MDOF systems compared to their lower equivalents.

Bursi and Shing (1996) provides an overview of implicit time-stepping algorithms utilized in pseudo-dynamic tests. It focuses on the examination of α-C, a modified Newton iteration-based algorithm developed by Shing and Vannan (1991), and α-OSM (Operator-Splitting Modified) algorithms through a systematic approach, containing theoretical analysis, numerical simulations, and verification experiments. The research findings reveal that the α-C algorithm effectively mitigates the impact of experimental errors. However, it is observed that undershooting errors introduce energy-adding effects using the α-OSM algorithm. Those effects resist suppression through algorithmic damping. The implementation of I-Modification shows to be an effective strategy to correct the influence of experimental errors when dealing with the α-OSM algorithm.

Fu et al. (2018) investigates error propagation properties of integration algorithms for pseudo-dynamic tests. The research develops a new algorithm by introducing two additional coefficients to the Chen-Ricles algorithm. In addition, a parameter, i.e., degree of nonlinearity, is described to model stiffness evolution for nonlinear structures.

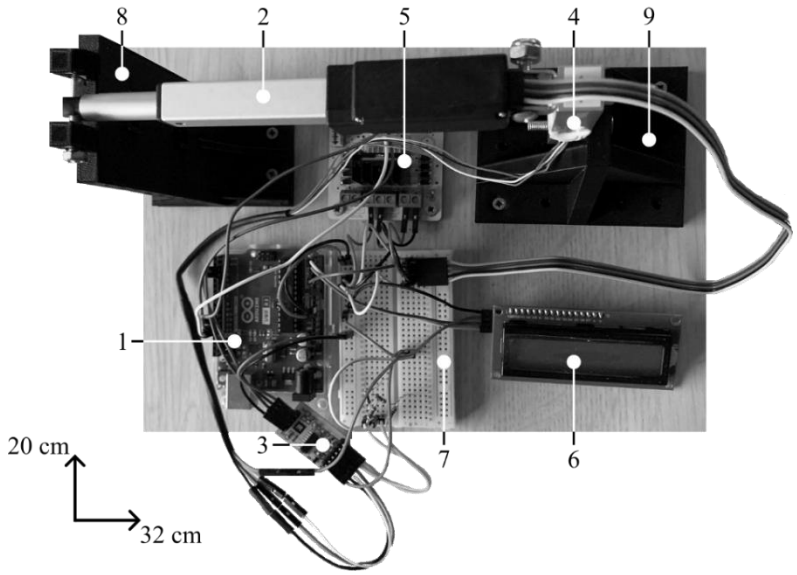
Özturk et al (2023) compares the α-OSM and Central Difference Method (CDM) using numerical analyses and CDM for the experimental analyses to investigate the effects combined with various time step interval values. This research studies the behaviour of a 1 DOF cantilever column system under the effect of a pulse load and earthquake record. The experimental results show to be deteriorated at small time interval values.

2 Small-scale set-up

2.1 Small-scale set-up configuration

Figure 2 illustrates the small-scale set-up. An Arduino UNO (1) controls the entire system. It directs the linear actuator (2) to extend, contract or stay stationary. The Actuonix L16-50-150-6-P linear actuator comes with a built-in potentiometer which measures the actual position of the linear actuator. A HX711 load cell amplifier (3), used to get measurable data out from a load cell, is connected to an EstarDyn load bar (4) with a capacity up to 200 N in compression and 100 N in tension. This load bar is attached to a load bar support (9).

A L298N H-bridge (5), installed between the Arduino UNO and the linear actuator, reverses polarity if needed. Regular polarity makes the linear actuator extend, reversed polarity causes contraction. An LCD (6) displays displacement and force on-the-fly. A breadboard (7) connects the components through jumper wires. The actuator end is supported by a 3D-printed specimen (8) where displacements will be applied upon. This specimen represents a 1 degree of freedom (DOF) PS.



1	Arduino UNO
2	Actuonix L16-50-150-6-P linear actuator
3	HX711 load cell amplifier
4	EstarDyn load bar 20 kg
5	L298N H-bridge
6	LCD
7	Breadboard
8	Specimen
9	Load bar support

Figure 2. Small-scale set-up.

2.2 Calibration of the small-scale set-up

The load cell consists out of four strain gauges hooked up in a Wheatstone bridge. The strain gauge resistance is proportional to the applied load, which allows one to calculate the weight of objects.

Enhancing the load cell in a common scale set-up allows calibration, as illustrated by Figure 3. Five different objects were weighted on a calibrated lab scale before determining the load cell's gain factor λ

$$\lambda = \frac{m_1 - m_0}{m_{\text{ref}}} \tag{2}$$

where m_0 is the tare value, m_1 is the measured value, and m_{ref} is the load value measured using the calibrated laboratory scale.

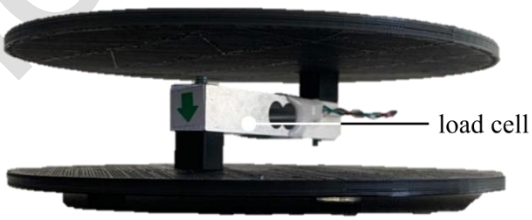


Figure 3. Calibration scale using load cell.

2.3 Controlling the small-scale set-up

An Arduino UNO operates as the controller of the small-scale set-up. The algorithm resets the linear actuator position, i.e., to the middle of the total stroke length, during the initialisation phase whereafter the position and force output are set to zero.

While performing a test, the Arduino UNO directs the linear actuator to extend, contract or stay stationary, based on the difference between the current position and the desired position. An internal potentiometer allows the linear actuator to measure the current flowing through, which is translated by the controller algorithm to a position by linear interpolation. Eq. (3) presents the calculation of the current actuator position x_{actuator}

$$x_{\text{actuator}} = \frac{E_{\text{actuator}}}{1023\ \text{V}} \cdot \ell_{\text{actuator}} - \frac{\ell_{\text{actuator}}}{2} \tag{3}$$

where E_{actuator} is the measured voltage, and ℓ_{actuator} is the total stroke length of the linear actuator.

When defining the discrepancy Δx between the desired actuator position x_{c} and the current actuator position x_{a}, three possible scenarios emerge: (i) $\Delta x > x_{\text{tol}}$, resulting in actuator extension; (ii) $\Delta x < -x_{\text{tol}}$, leading to actuator contraction; or (iii) $-x_{\text{tol}} \leq \Delta x \leq x_{\text{tol}}$, with x_{tol} denoting the allowed tolerance. An internal clock inside the Arduino UNO executes the algorithm every 1 millisecond, and a upper limit has been set on the maximum executions steps to prevent an infinite overshooting loop.

2.4 Time integration

Typically, the PS is characterized by a limited number of DOFs, often fewer than five, making an accurate characterization of the tangent stiffness challenging. In contrast, the NS is characterized by a larger number of DOF, potentially up to 100, and allows for the evaluation of the Jacobian of the restoring force. Consequently, when interface masses are partitioned properly, it becomes feasible to employ an explicit time integration scheme for solving the PS response, while the NS response is solved through an implicit time integration scheme. This partitioned approach stands in contrast to a monolithic time integration strategy which does not allow to couple different time integration schemes. Gravouil and Combescure (2001) have introduced a domain decomposition method to solve time-dependent nonlinear problems.

Partitioned time integration indicates a class of algorithms to couple multiple monolithic time integration processes, e.g., based on the Newmark scheme. Analogous to the Newmark-Newton-Raphson algorithm, additional kinematic constraints are enforced using Lagrange multipliers. The coupled equations of motion for a general hybrid model, consisting out of one PS and one NS, can be expressed as:

$$\begin{cases} \boldsymbol{M}^{\text{P}}\ddot{\boldsymbol{u}}_{i+1}^{\text{P}} + \boldsymbol{C}_{\text{p}}^{\text{P}}\dot{\boldsymbol{u}}_{i+1}^{\text{P}} + \boldsymbol{r}^{\text{P}}\left(\boldsymbol{u}_{i+1}^{\text{P}}, \dot{\boldsymbol{u}}_{i+1}^{\text{P}}\right) = \boldsymbol{B}_f^{\text{P}}\boldsymbol{f}^{\text{P}}(t_{i+1}) + \boldsymbol{B}_v^{\text{P}}\boldsymbol{\Lambda}_{i+1} \\ \boldsymbol{M}^{\text{N}}\ddot{\boldsymbol{u}}_{i+1}^{\text{N}} + \boldsymbol{C}_{\text{p}}^{\text{N}}\dot{\boldsymbol{u}}_{i+1}^{\text{N}} + \boldsymbol{r}^{\text{N}}\left(\boldsymbol{u}_{i+1}^{\text{N}}, \dot{\boldsymbol{u}}_{i+1}^{\text{N}}\right) = \boldsymbol{B}_f^{\text{N}}\boldsymbol{f}^{\text{N}}(t_{i+1}) + \boldsymbol{B}_v^{\text{N}}\boldsymbol{\Lambda}_{i+1} \end{cases} \tag{4}$$

$$\boldsymbol{B}_v^{\text{P}^{\text{T}}}\dot{\boldsymbol{u}}_{i+1} + \boldsymbol{B}_v^{\text{N}^{\text{T}}}\dot{\boldsymbol{u}}_{i+1} = \boldsymbol{0} \tag{5}$$

where $\boldsymbol{M}$ is the mass matrix, $\ddot{\boldsymbol{u}}_{i+1}$ the nodal acceleration vector, $\boldsymbol{C}_{\text{p}}$ the damping matrix, $\dot{\boldsymbol{u}}_{i+1}$ the nodal velocity vector, and $\boldsymbol{u}_{i+1}$ the nodal displacement vector, $\boldsymbol{r}$ the restoring force vector, $\boldsymbol{B}_f$ a Boolean matrix used to collocate the externally applied loading $\boldsymbol{f}(t)$ to the set of loaded DOF, and $\boldsymbol{B}_v$ a Boolean matrix used for collocating the Lagrange multiplier $\boldsymbol{\Lambda}$.

For seismic loading conditions:

$$\boldsymbol{B}_f^{\blacksquare} = -\boldsymbol{M}\boldsymbol{p} \tag{6}$$

$$\boldsymbol{f}(t) = \ddot{\boldsymbol{u}}_{\text{g}}(t) \tag{7}$$

where $\boldsymbol{p}$ is a Boolean vector, $\ddot{\boldsymbol{u}}_{\text{g}}(t)$ the ground motion acceleration time-history, and ■ denotes that the equation is applicable to both PS and NS. The superscript P indicates the PS, while the superscript N indicates the NS.

In principle, one could solve the Newton-Raphson scheme to simultaneously solve the system of equations in Eq. (4) and (5) for a generic time step. However, for enhanced computational efficiency, a two-stage solution strategy is adopted. Abbiati (2022) provides a detailed account of the integration scheme, implemented using MATLAB or Python.

Furthermore, the I-modification technique illustrated by Nakashima and Kato (1988) is implemented to correct restoring forces to mitigate experimental errors, as demonstrated in Eq. (8)

$$\tilde{\boldsymbol{r}}_{i+1}^{c} = \tilde{\boldsymbol{r}}_{i+1}^{a} - \mathbf{K}(\tilde{\boldsymbol{u}}_{i+1}^{a} - \tilde{\boldsymbol{u}}_{i+1}^{r}) \tag{8}$$

in which $\tilde{\boldsymbol{u}}_{i+1}^{r}$ is the exact numerical solution, $\tilde{\boldsymbol{u}}_{i+1}^{a}$ and $\tilde{\boldsymbol{r}}_{i+1}^{a}$ experimental feedback values, and $\tilde{\boldsymbol{r}}_{i+1}^{c}$ the corrected restoring force vector.

2.5 PC-connection

Newly developed MATLAB and Python classes have been introduced to create an object, including essential properties to connect with the Arduino UNO, i.e., (i) port name, (ii) baud rate, (iii) serial port time-out, and (iv) a handle to the serial port. Both classes contain functions to (i) create an object, (ii) create serial communication, (iii) set the actuator position, (iv) get the object state and (v) delete the object.

A USB-B cable connects the Arduino UNO with a PC, allowing bidirectional communication between the created object and the Arduino UNO. Noteworthy, the built-in serialport function orchestrates the MATLAB-Arduino communication. However, it lacks the capability to wait for the Arduino to complete its execution. To ensure the actuator has sufficient time to reach the desired displacement, a 5-second pause is integrated into the MATLAB script.

Simultaneously, a Python class has been developed, containing the same essential properties and functions of the MATLAB counterpart, facilitating interaction with an Arduino UNO. A significant distinction is in the management of Python-Arduino communication, using the *serial* module. In contrast to the MATLAB approach, the serial module offers functionality to wait for the Arduino to complete its execution.

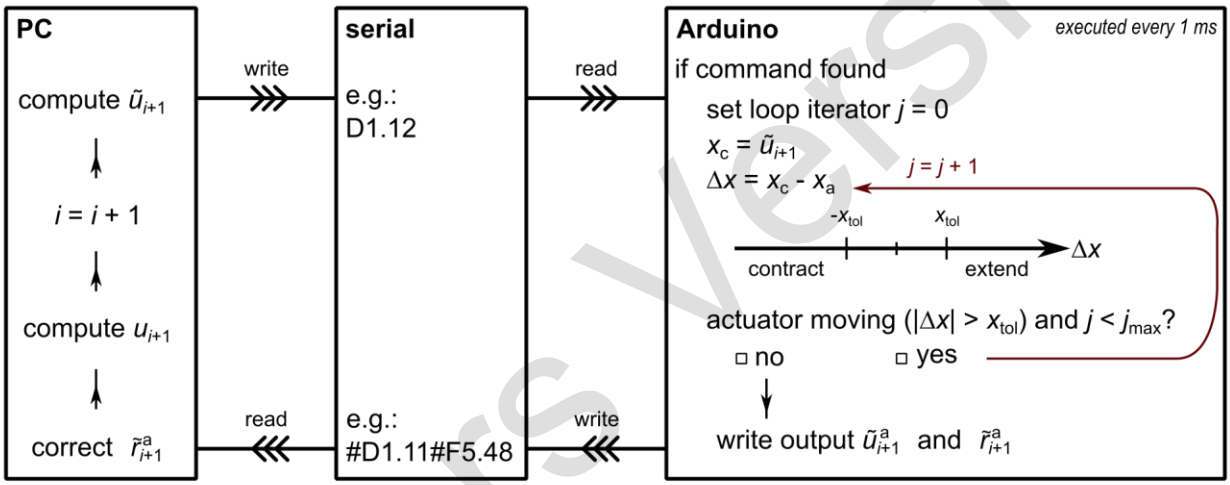


Figure 4. Controlling algorithm and communication with PC.

3 Description of the case study

The experimental consists out of a numerical model with 4 DOFs coupled with a 1 DOF physical specimen. Firstly, a comparative analysis is conducted between MATLAB and Python-based simulations incorporating I-modification. Subsequently, a comparison is drawn between simulations without I-modification and those integrating I-modification, focusing on the Python-based approach. Figure 6 provides a representation of the case study, identifying the components, including the case study structure (CS), the lumped mass model (LM), the PS, and the NS.

To enhance clarity, the Boolean matrices are explicitly mentioned below.

$$\boldsymbol{B}_v^{\mathrm{P}} = [1];\ \boldsymbol{B}_v^{\mathrm{N}} = [-1 \quad 0 \quad 0 \quad 0] \tag{9}$$

The ground motion acceleration time-history employed as the basis for externally applied loading can be expressed as:

$$\ddot{\boldsymbol{u}}_{\mathrm{g}}(t) = 0.05\mathrm{g} \cdot \sin(\omega \boldsymbol{t}) \cdot \boldsymbol{W}_n \tag{10}$$

In this equation, g represents the gravitational acceleration, which is equal to 9.81 m/s², ω is the angular frequency, t denotes the time, and W_n represents an L-point symmetric Hann window (eq.(11)) applied over n time steps. In this specific study, each simulation considers a time step of 0.02 s and a total time of 5 s.

$$\boldsymbol{W}_n = 0.5\ \left(1 - \cos 2\pi \frac{n}{N}\right) \tag{11}$$

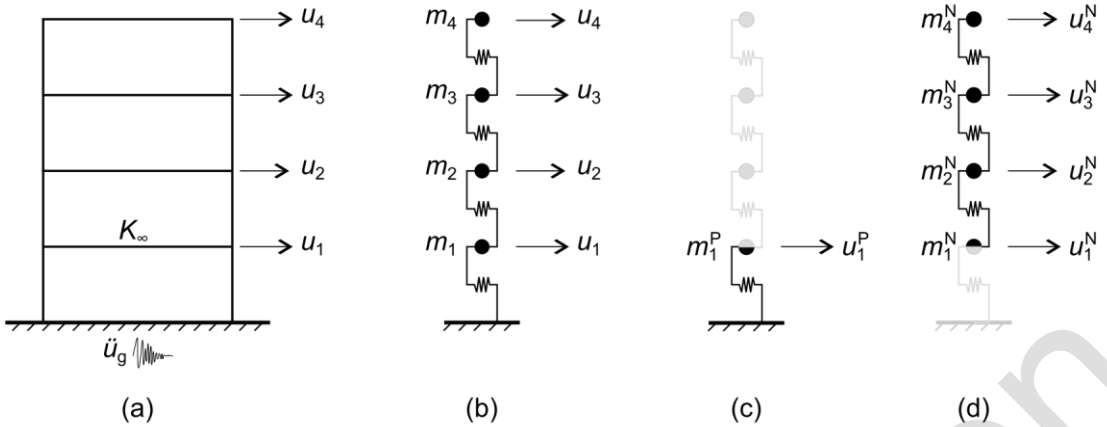


Figure 5. Case study with 4 DOFs. (a) Case study structure, (b) lumped mass model, (c) the PS and (d) the NS.

Based on the eigenfrequencies of the system, three different cases are considered, where ω assumes the following values: (i) 10 rad/s, (ii) the eigenfrequency corresponding to mode 1, and (iii) the eigenfrequency corresponding to mode 2. In all three cases, both fully numerical and hybrid simulations are conducted. In a fully numerical approach, all substructures are represented and modeled exclusively through numerical simulations. Table 1 provides an overview of the simulations.

Table 1. Overview simulations.

	Load case 1	**Load case 2**	**Load case 3**
MATLAB-script with I-modification	M1	M2	M3
Python-script with I-modification	P1	P2	P3
Python-script without I-modification	PW1	PW2	PW3

Table 2 presents the load cases with the number of time steps N, the angular frequency w, the time step dt end the total time of execution t_{max}. LC3 contains a higher angular frequency. Therefore, a reduced time step is applied in order to obtain a stable integration scheme and accurate results.

Table 2. Load case properties.

	N	**ω (rad/s)**	**Δt (s)**	**t_{max} (s)**
Load case 1 (LC1)	250	10.00	0.02	5.00
Load case 2 (LC2)	250	ω_{mode1}	0.02	5.00
Load case 3 (LC3)	250	ω_{mode2}	0.01	2.50

4 Results

Table 3 shows the eigenfrequencies of the system, which define the angular frequencies of interest for the simulations: (i) case 1 with $\omega = 10\ \mathrm{rad/s}$, (ii) case 2 with $\omega = 13.5\ \mathrm{rad/s}$ and (iii) case 3 with $\omega = 38.75\ \mathrm{rad/s}$. This leads to the illustrated applied loadings in Figure 6.

Eq. (12) converts the angular frequency to the frequency f

$$f = \frac{\omega}{2\pi} \tag{12}$$

Table 3. The eigenfrequencies of the system in radians per second.

	Mode 1	**Mode 2**	**Mode 3**	**Mode 4**
ω	13.45	38.73	59.34	72.79
f	2.14	6.16	9.44	11.58

Applying these properties to the loading procedure as described in *Description of the case study*, this results in the loading patterns as shown in Figure 6.

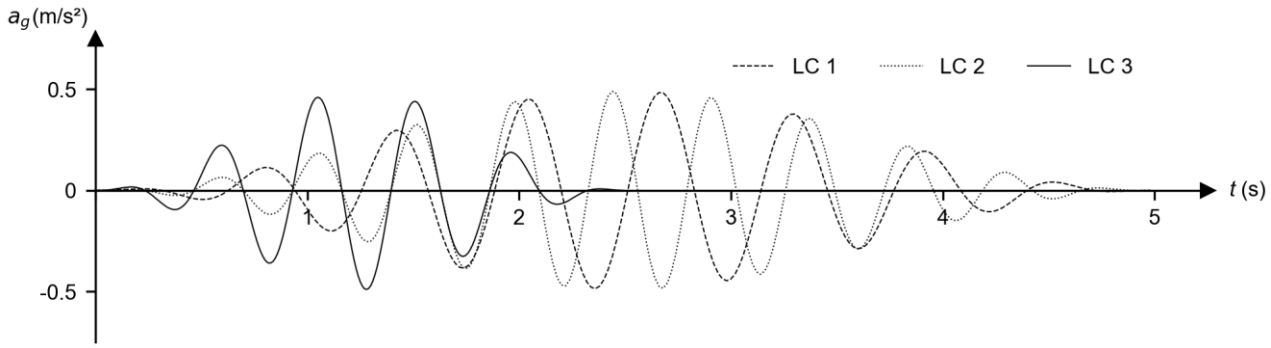


Figure 6. Applied loadings.

Figure 7 presents the displacement results from fully numerical simulations conducted with both MATLAB and Python. In the context of a fully numerical analysis, the restoring force of the PS is also computed numerically, e.g., $F_{i+1} = Ku_{i+1}$. The congruence of these displacements not only confirms the uniform and validated implementation on both platforms but also serves as a foundation for validating the results of the hybrid simulations. The accuracy of the numerical results, in comparison to the hybrid simulation results, are dependent of the estimated stiffness which is used to compute the numerical restoring force.

However, Figure 8 reveals that MATLAB executions within a hybrid simulation context encounter significant challenges, ultimately resulting in failure. This failure is attributed to serial overflows, leading to communication errors, which subsequently result in inaccurate imposed displacements and irrelevant restoring forces. Consequently, as the computed displacements keeps increasing, the displacements exceed the setup's limitations, magnifying the issues.

Therefore, the interruption of the hybrid simulation using MATLAB before its completion is necessary. In contrast, Python simulations consistently shows stable execution, as illustrated in Figure 9 for P1 and P2, where the hybrid simulation results closely match the numerical results. However, P3 exhibits a higher degree of variability in its results and does not closely align with the oscillations observed in the numerical results.

Figure 10 provides insights into the impact of employing I-modification versus its absence. Notably, for small displacements, some noise is observed in the restoring force measurements, as represented in Figure 12, where simulations P1-PW1 and P2-PW2 are comparable in their outcomes. In contrast, P3-PW3 displays a greater degree of variability in its results.

Figure 11 illustrates the presence of experimental errors. These errors are primarily attributed to the lack of a velocity-controlled actuator and the imposition of a maximum number of allowed loops to achieve the desired displacement. Consequently, intermittent measurement of relatively substantial errors occurs. Nevertheless, the incorporation of I-modification proves to be an effective remedy for addressing these errors. Notably, for smaller displacements, the setup contends with noise that has a pronounced impact on the execution, particularly in the case of LC3. This effect is most conspicuous when dealing with maximum displacements limited to the problematic range of -2.0 mm to 2.0 mm, where the noise exerts the greatest influence.

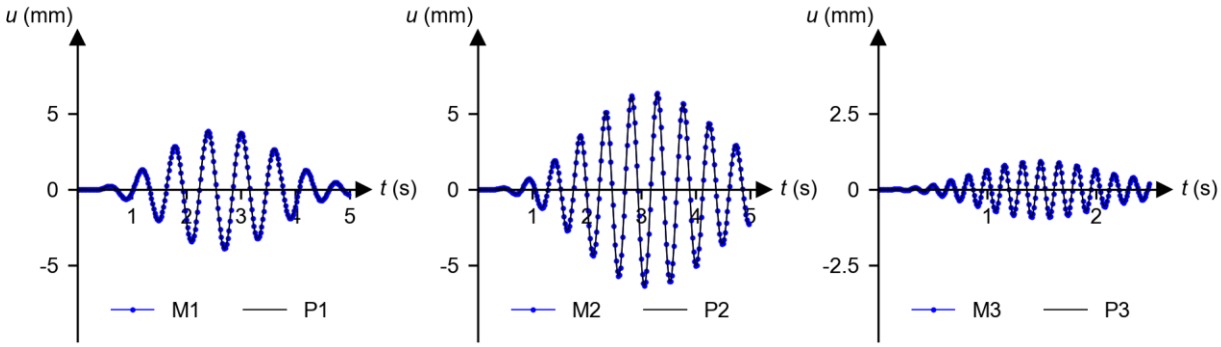


Figure 7. Displacements of PS applying fully numerical simulations using MATLAB and Python.

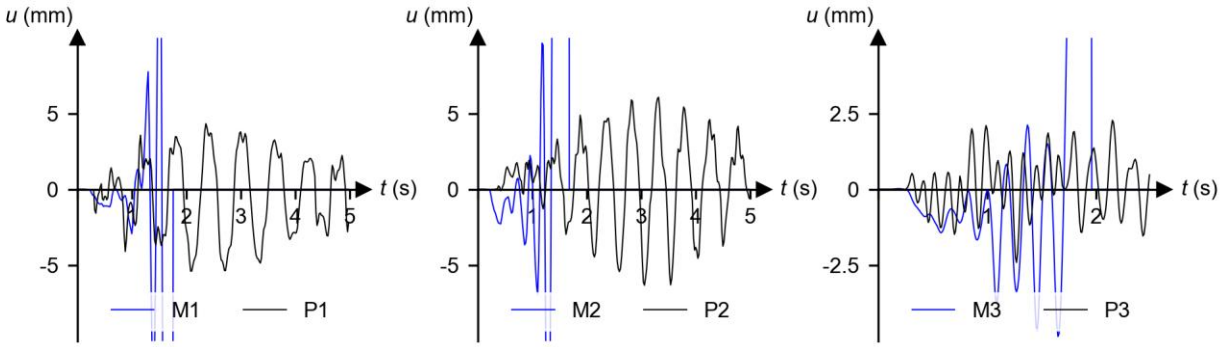


Figure 8. Displacements of PS applying hybrid simulation using MATLAB and Python with I-modification.

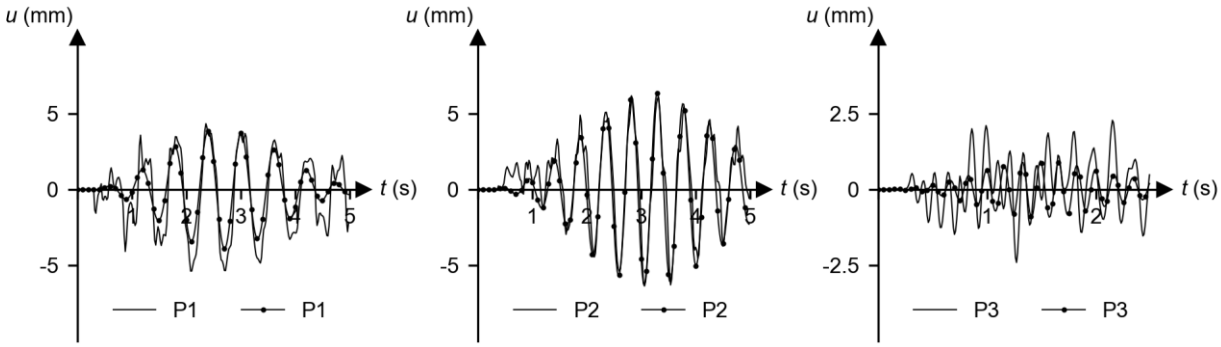


Figure 9. Displacements of PS applying hybrid simulation vs. numerical simulation using Python with I-modification.

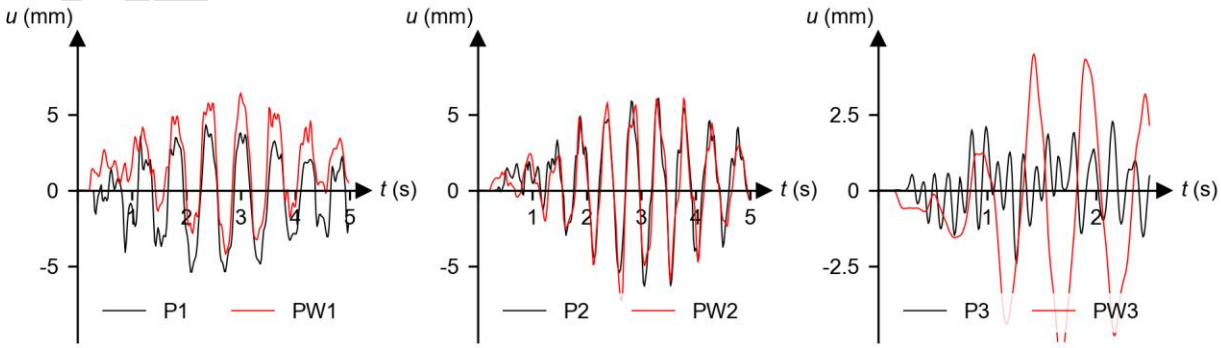


Figure 10. Displacements of PS applying hybrid simulation with and without I-modification using Python.

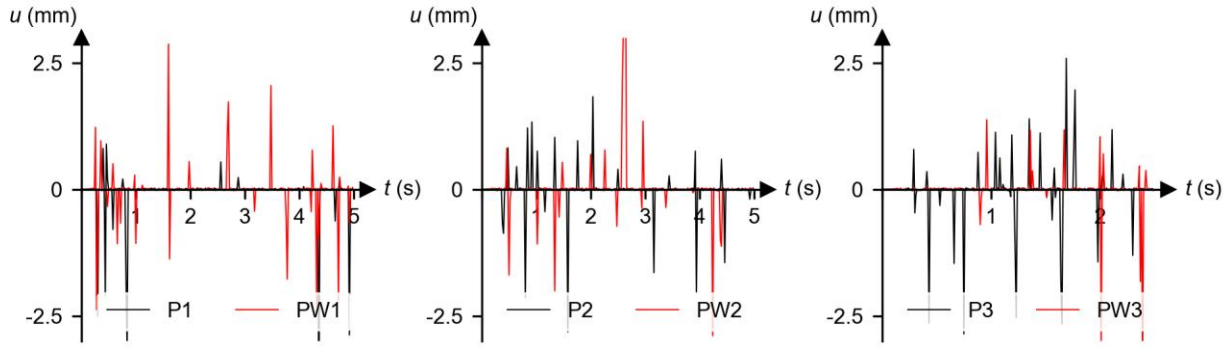


Figure 11. Experimental error.

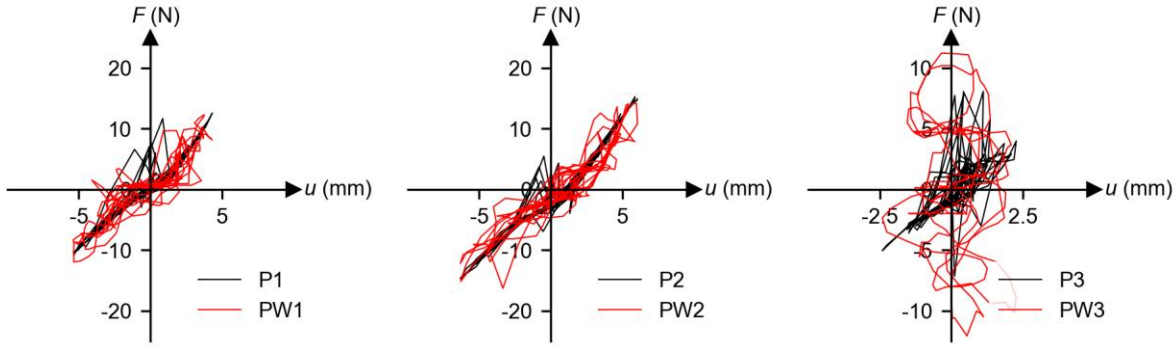


Figure 12. Hysteresis.

5 Conclusions

The linear actuator requires a non-negligible amount of time to attain the imposed displacement due to several contributing factors. Firstly, the communication protocol uses the serial monitor to send the desired displacement. Secondly, the Arduino UNO issues the directive for the linear actuator to move towards the imposed displacement. Finally, the linear actuator moves towards the desired displacement target.

The presence of a single-gear linear actuator introduces a tendency to overshoot, an issue that can be mitigated by expanding the allowed displacement tolerance. However, this remedy carries the potential to compromise measurement accuracy, underscoring the necessity to strike a delicate balance between stability and precision.

Furthermore, the establishment of a stable setup is compulsory to accurately achieve the desired displacement and obtain precise restoring forces. An unstable setup is vulnerable to inducing infinite loops in the actuator's operation, consequently prolonging the execution time of tests. To overcome infinite loops, an upper limit has been imposed on the number of loops the setup may execute to achieve the desired displacement. Nevertheless, exceeding the maximum loop count could potentially result in a displacement error beyond the acceptable tolerance. The implementation of I-modification techniques on measured restoring forces proves instrumental in enhancing outcomes and achieving algorithmic stability, contingent upon the accurate estimation of the initial stiffness.

To enhance the small-scale set-up's performance, the integration of a velocity-controlled linear actuator holds the promise of establishing a control system capable of achieving more precise displacements. Additionally, a velocity-controlled linear actuator contributes to the refinement of the control algorithm. Augmenting the small-scale setup not only augments accuracy but also accommodates the testing of systems with elevated complexity, including the transition from 1 DOF to 2 DOFs. This adaptation allows for the calibration and fine-tuning of the hybrid testing procedure before its application in more resource-intensive full-scale, large-scale, or real-world scenarios.

Furthermore, an interesting approach of exploration is the introduction of specimens operating within the nonlinear range, as the current setup has primarily focused on linear-range specimens.

In conclusion, the small-scale experimental set-up serves as a valuable instrument for conducting simulations within the constraints of limited applied forces and allowed displacements. The achieved displacement and force measurements offer a degree of precision sufficient for the evaluation of adjusted or newly developed algorithms, enabling the testing of diverse scenarios and the refinement of testing procedures. This, in turn, enhances the efficacy of full-scale testing while concurrently decreasing associated costs.

Acknowledgements

This research was supported by the Special Research Fund (BOF) of Hasselt University. BOF reference: BOF21OWB11. Also, the first author would like to thank prof. Abbiati, affiliated to Aarhus University in Denmark, for the idea of creating a small-scale set-up.

References

Abbiati, G., 2022. *Numerical time integration schemes for hybrid testing,* Aarhus: Aarhus University.

Bursi, O. S. & Shing, P. S.-B., 1996. Evaluation of some implicit time-stepping algorithms for pseudodynamic tests. *Earthquake Engineering & Structural Dynamics,* 25(4), pp. 333-355.

Dermitzakis, S. N. & Mahin, S., 1985. *Development of substructuring techniques for on-line computer controlled seismic performance testing,* Berkeley: Earthquake Engineering Research Center.

Fu, B., Jian, H. & Wu, T., 2018. Nonlinear error propagation analysis of a new family of model-based integration algorithm for pseudodynamic test. *Sustainability,* 10(8), p. 2846.

Gravouil, A. & Combescure, A., 2001. Multi-time-step explicit-implicit method for non-linear structural dynamics. *International Journal for Numerical Methods in Engineering,* 50(1), pp. 199-225.

Magonette, G., 2001. Development and application of large-scale continuous pseudo-dynamic testing techniques. *Dynamic Testing of Structures,* 359(1786), pp. 1771-1799.

McCrum, D. P. & Williams, M. S., 2016. An overview of seismic hybrid testing of engineering structures. *Engineering Structures,* Volume 118, pp. 240-261.

McCrum, D. P. & Williams, M. S., 2016. An overview of seismic hybrid testing of engineering structures. *Engineering Structures.*

Nakashima, M. & Kato, H., 1988. *Experimental error growth behavior and error growth control in pseudo dynamic test.* Tokyo-Kyoto, Ninth World Conference on Earthquake Engineering.

Öztürk, S., Doran, B. & Aksoylar, C., 2023. Effect of time step interval and time integration schemes in pseudo-dynamic analysis. *Structures,* Volume 50, pp. 215-230.

Shing, P.-S. B. & Mahin, S. A., 1987. Cumulative experimental errors in pseudodynamic tests. *Earthquake Engineering & Structural Dynamics,* 15(4), pp. 409-424.

Shing, P.-S. B. & Vannan, M. T., 1991. Implicit time integration for pseudodynamic tests: Convergence and energy dissipation. *Earthquake Engineering & Structural Dynamics,* 20(9), pp. 809-819.

Takahashi , Y. & Fenves, G. L., 2006. Software framework for distributed experimental-computational simulation of structural systems. *Earthquake Engineering & Structural Dynamics,* 35(3), pp. 267-291.

Takanashi, K. & Nakashima, M., 1987. Japanese activities on on-line testing. *Journal of Engineering Mechanics,* 113(7), pp. 1014-1032.

Takanashi, K. et al., 1975. Non-linear earthquake response analysis of structures by a computer-actuator on-line system. *Transactions of the Architectural Institute of Japan,* 229(0), pp. 77-83.