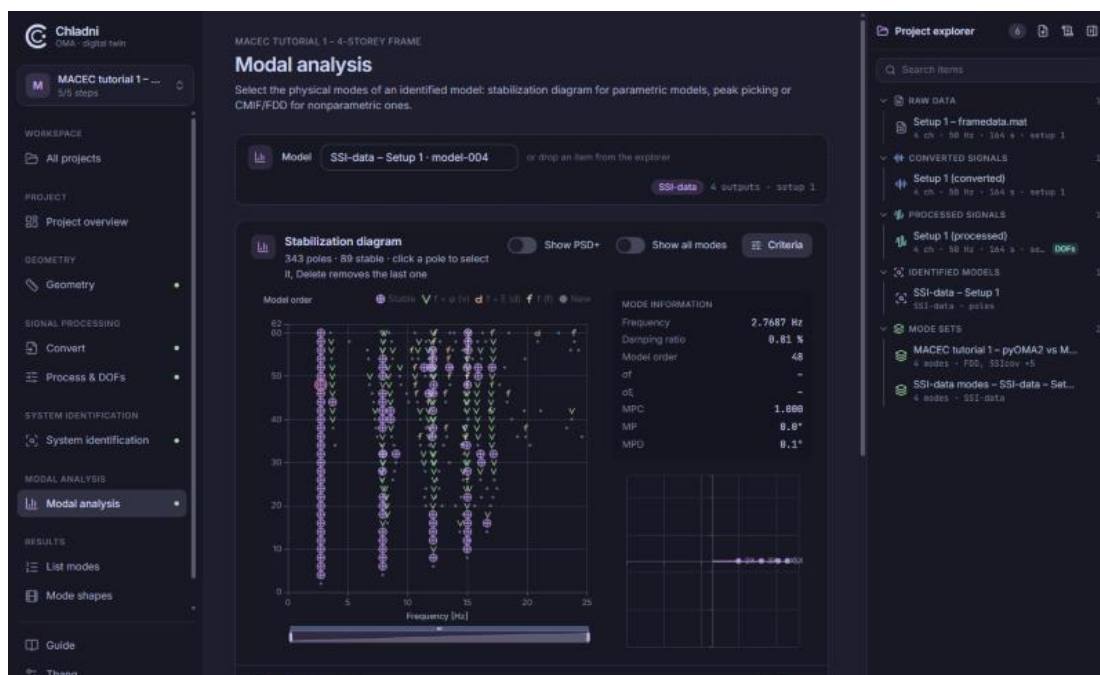


Chladni

Sổ tay sử dụng

Phân tích modal thực nghiệm, vận hành và kết hợp
theo dự án – với pyOMA2 và OpenSeesPy



Lê Xuân Thắng

Phiên bản 0.3 · Ngày 19 tháng 10 năm 2026

Mục lục

1	Giới thiệu	2
1.1	Các loại phân tích	2
1.2	Dự án, mô hình FE và giao diện	2
1.3	Tài liệu liên quan	3
2	Bắt đầu	4
2.1	Yêu cầu và cài đặt	4
2.2	Khởi động	4
2.3	Ứng dụng độc lập trên macOS	4
2.4	Hồ sơ người dùng, đăng nhập và đăng xuất	5
2.5	Trợ giúp, phím tắt và bảng lệnh	5
3	Cấu trúc và quy ước	6
3.1	Cấu trúc chương trình	6
3.2	Các mục và nguồn gốc	6
3.3	Định dạng dữ liệu đo được hỗ trợ	7
3.4	Import và Convert – định dạng làm việc của Chladni	7
3.5	Hình học	8
3.6	Bậc tự do đo (DOF) và hướng	8
3.7	Thư mục project	9
3.8	Logfile và chế độ batch	9
4	Giao diện	10
4.1	Bố cục	10
4.2	Projects và tổng quan project	11
4.3	Geometry	11
4.4	Convert	12
4.5	Process & DOFs	13
4.6	System identification	14
4.7	Modal analysis	15
4.8	Results	16
5	Danh mục hàm (chế độ batch)	19
5.1	Ví dụ	20
6	Ví dụ kiểm chứng	21
6.1	Khung cắt 4 tầng	21
6.2	Hệ 2 bậc tự do có lực đo (EMA/OMAX)	21
A	Thông báo lỗi thường gặp	22
B	Viết tắt	23

Chương 1

Giới thiệu

Chladni là một chương trình phân tích modal: từ dữ liệu dao động đo được, nó xác định các đặc trưng modal của kết cấu – tần số riêng, hệ số cản, dạng mode – và so sánh chúng với mô hình phần tử hữu hạn. Tên chương trình đặt theo Ernst Chladni (1756–1827), người đầu tiên làm cho dạng dao động “nhìn thấy được” bằng cát rắc trên tấm kim loại.

Chladni chia phân tích modal thành ba bước tách bạch:

1. **xử lý tín hiệu**: đưa dữ liệu đo về dạng làm việc của chương trình, làm sạch và gán vị trí đo;
2. **nhận dạng hệ** (system identification): ước lượng một mô hình toán học của hệ từ dữ liệu – phổ PSD^+/FRF (phi tham số) hoặc các mô hình không gian trạng thái/đa thức ở nhiều bậc (tham số);
3. **phân tích modal**: rút các mode vật lý từ mô hình đã nhận dạng (biểu đồ ổn định, peak picking).

Sự phân biệt giữa nhận dạng hệ và phân tích modal được giữ cả trong giao diện lẫn trong các hàm: một mô hình được nhận dạng một lần và có thể được “đọc” bằng nhiều cách chọn mode khác nhau.

1.1 Các loại phân tích

Loại	Đo được	Phương pháp trong Chladni
OMA (vận hành, output-only)	chỉ đáp ứng	$\text{PSD}^+ + \text{PP/FDD}$, SSI-data, SSI-cov (/ref), pLSCF
EMA (thực nghiệm, input-output)	lực và đáp ứng	$\text{FRF}(\text{H1}) + \text{PP/CMIF}$, pLSCF tất định
OMAX (kết hợp)	lực + kích thích môi trường	CSI (SSI có đầu vào đo)

1.2 Dự án, mô hình FE và giao diện

- **Dự án**: mỗi công trình là một thư mục tự chứa, có thông tin công trình, bản vẽ, lịch sử, dữ liệu, kết quả và `logfile.py`; xuất/nhập thành một file ZIP.
- **Mô hình FE**: nhập mô hình OpenSeesPy/Tcl để có hình học và dạng mode FE (so MAC), hoặc mô phỏng dữ liệu đo.
- **Giao diện desktop** chạy offline, song ngữ Anh/Việt, giao diện sáng/tối.

1.3 Tài liệu liên quan

Các case study được viết thành tutorial riêng trong D:\Working\NCREP\Examples: z24/ (cầu Z24, 9 setup, hai trụ). Bài toán kiểm chứng khung cắt 4 tầng có sẵn dưới dạng project mẫu (chương 6).

Chương 2

Bắt đầu

2.1 Yêu cầu và cài đặt

Chạy từ mã nguồn, Chladni dùng được trên Windows (đã thử trên Windows 11) và Linux. Cần uv (quản lý Python) và Node.js (chỉ để build giao diện một lần). Python 3.12 được uv tự cài; OpenSeesPy hiện chỉ hỗ trợ bản này. Cửa sổ desktop dùng Qt 6 (PySide6), được uv sync cài cùng các thư viện khác. Trên Mac Apple Silicon có thể dùng bản đóng gói sẵn (mục 2.3), không cần cài gì thêm.

```
cd D:\Working\NCREP\Chladni
uv sync                                # môi trường Python + thư viện
cd web && npm install && npm run build  # build giao diện (một lần)
cd ..
```

2.2 Khởi động

uv run chladni desktop	ứng dụng trong cửa sổ riêng, chạy offline
uv run chladni serve	cùng giao diện, mở trong trình duyệt (http://localhost:8000)
uv run python scripts/install_shortcut.py	tạo lối tắt Chladni trên Desktop và Start Menu

Các lệnh phải chạy trong thư mục Chladni; từ nơi khác dùng `uv run -directory <đường dẫn>\Chladni chladni desktop`. Các project được lưu trong Chladni\projects; biến môi trường CHLADNI_PROJECTS chuyển chúng sang thư mục khác.

2.3 Ứng dụng độc lập trên macOS

Bản macOS (Apple Silicon, arm64) đã chứa sẵn Python, giao diện, cửa sổ Qt và các thư viện tính toán: người dùng không cần Python, Node.js, uv hay mã nguồn. Mở file Chladni-<phiên bản>-arm64.dmg và kéo Chladni vào Applications. Kiểm tra file tải về bằng checksum SHA-256 trong SHA256SUMS.

- **Dữ liệu:** project, hồ sơ người dùng, bộ tham số hiệu chỉnh, kết quả và log nằm trong ~/Library/Application Support/Chladni/. Thư mục này được giữ nguyên khi thay bản ứng dụng mới. Biến CHLADNI_HOME chuyển toàn bộ dữ liệu sang thư mục khác, CHLADNI_PROJECTS chỉ chuyển các project.
- **Chuyển project từ bản chạy mã nguồn:** project cũ không được chép tự động; xuất ZIP ở trang **Projects** rồi **Import** trong ứng dụng.

- **Project mẫu:** đi kèm ứng dụng và chạy offline.
- **Ký số:** bản dựng chỉ có chữ ký ad-hoc, chưa được Apple notarize. Lần đầu mở, macOS có thể chặn ứng dụng; khi đó vào **System Settings** → **Privacy & Security** và chọn **Open Anyway**.
- **Khi không mở được:** xem `desktop.log` trong thư mục dữ liệu.

Cách build bản này và quy trình phát hành qua GitLab CI có trong `docs/STANDALONE_MACOS.md`.

2.4 Hồ sơ người dùng, đăng nhập và đăng xuất

Chladni là ứng dụng chạy trên máy, không có tài khoản trên máy chủ: mỗi người dùng có một **hồ sơ cục bộ** (tên, đơn vị, e-mail và cài đặt riêng), không cần mật khẩu. Lần đầu mở ứng dụng, màn hình chào yêu cầu tạo hồ sơ; sau đó chọn hồ sơ để vào. **Đăng xuất** (menu tài khoản ở góc dưới thanh trái) quay lại màn hình chào để người khác dùng máy. Các dự án dùng chung cho mọi hồ sơ trên máy.

Menu tài khoản: **Account settings** (Ctrl+,): ngôn ngữ (English/Tiếng Việt – thuật ngữ như MAC, PSD, SSI giữ nguyên), giao diện (bốn bảng màu Catppuccin, theo hệ thống hoặc cố định; 14 màu nhấn), tên và đơn vị; chuyển nhanh ngôn ngữ và giao diện sáng/tối; **Sign out**.

2.5 Trợ giúp, phím tắt và bảng lệnh

- **Help** (thanh trái, kèm số phiên bản): hướng dẫn trong ứng dụng, sổ tay này (mở bằng trình đọc PDF của máy), phím tắt, bảng lệnh, **What's new** (nhật ký thay đổi – tự hiện một lần sau mỗi lần cập nhật), **Report a problem** (trang báo lỗi của dự án) và **About Chladni** (phiên bản, bản dựng, phiên bản các thư viện tính toán, thư mục dự án).
- **Bảng lệnh** (Ctrl+K, hoặc ô **Search or jump to...** dưới ô chọn project): gõ để tìm trang, project hoặc thao tác (đổi ngôn ngữ, giao diện, ẩn/hiện Explorer, đăng xuất...), không cần gõ dấu; mũi tên để chọn, Enter để thực hiện.
- **Phím tắt:** ? mở danh sách đầy đủ.

Chương 3

Cấu trúc và quy ước

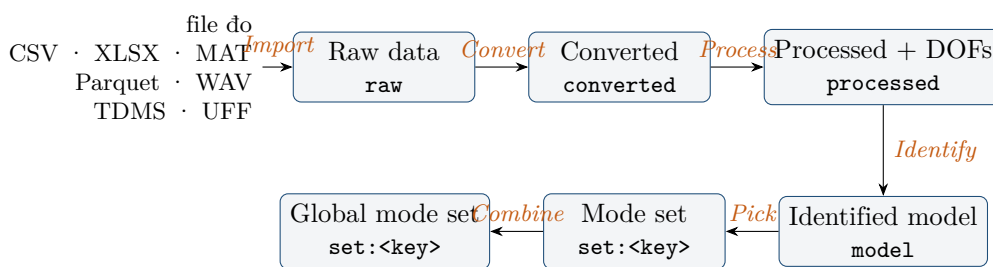
3.1 Cấu trúc chương trình

src/chladni/workbench/	lỗi phân tích: các mục, xử lý tín hiệu (signals), nhận dạng hệ (ident , emax), phân tích modal (modal), chế độ batch (batch)
src/chladni/project/	dự án, hình học, mô hình FE, xuất kết quả (CSV, UFF), dạng mode trên hình học
src/chladni/api/	máy chủ FastAPI (giao diện gọi qua <code>/api/...</code>)
web/	giao diện React

Các thuật toán nhận dạng dùng pyOMA2 [2] (SSI, pLSCF, FDD) và mã riêng của Chladni (PSD⁺, H1, pLSCF tắt định, tiêu chí ổn định, MP/MPD, ghép setup); mô hình FE dùng OpenSeesPy [3].

3.2 Các mục và nguồn gốc

Mỗi bước phân tích tạo ra một **mục** (item) được lưu trong project và hiện ở **Project explorer** (hình 3.1). Mỗi mục biết nó được tạo từ mục nào (nguồn gốc); xóa một mục thì mọi mục sinh ra từ nó cũng bị xóa.



Hình 3.1: Các mục của một project và thao tác tạo ra chúng.

Mục	Nội dung
Raw data	tín hiệu như đã ghi (V, counts hoặc đơn vị vật lý), đọc một lần vào định dạng nội bộ
Converted signals	tín hiệu làm việc: SI, nhãn, loại đại lượng của từng kênh
Processed signals	sau tiền xử lý; mỗi kênh gắn nút, hướng, cờ tham chiếu
Identified models	PSD ⁺ /FRF, hoặc cực của mọi bậc mô hình
Mode sets	các mode đã chọn: f , ξ , σ , ϕ , MPC, MP, MPD, bậc

3.3 Định dạng dữ liệu đo được hỗ trợ

Mỗi file là **một setup**; mỗi cột (hoặc mỗi track) là **một kênh**.

Bảng 3.1: Định dạng file đo nhập được (**Import files**, hoặc kéo file vào trang/Explorer).

Đuôi file	Quy tắc đọc
.csv .txt .dat .asc	bảng văn bản; dấu phân cách , ; tab hoặc khoảng trắng (tự nhận); dấu thập phân chấm hoặc phẩy; dòng bắt đầu bằng # là chú thích; hàng tiêu đề tùy chọn (có thì tên cột thành tên kênh, không có thì CH1, CH2, ...); cột thời gian tùy chọn (tên t, time, time [s], time_s, ... hoặc cột đầu có bước đều) – có cột này thì f_s được tính tự động
.xlsx	Excel: sheet đầu tiên, cùng quy tắc như bảng văn bản
.parquet .pq	bảng số Apache Parquet, đọc trực tiếp (không qua CSV); tên cột là tên kênh, giữ nguyên thứ tự cột và mọi mẫu. Cột thời gian tùy chọn (tính bằng giây, cùng các tên như bảng văn bản, ở vị trí bất kỳ) cho f_s ; với file có cột tên 0, 1, ... hoặc f0, f1, ... (do công cụ chuyển đổi tạo ra), cột đầu chỉ được coi là thời gian khi nó bắt đầu từ 0 và tăng đều. Yêu cầu chặt hơn bảng văn bản: mọi cột phải là số thực, không có ô trống/NaN/vô cực, tên cột không trùng, nhiều nhất một cột thời gian và cột này phải tăng đều (sai lệch $\leq 0,1\%$). f_s nhập tay (nếu có) được ưu tiên hơn f_s tính từ cột thời gian
.mat	MATLAB v5–v7.2 và v7.3 (HDF5): biến data (hoặc ma trận số lớn nhất), kênh $\times$ mẫu hay mẫu $\times$ kênh đều được; biến fs nếu có; một mảng chuỗi cùng số phần tử với số kênh (ví dụ labels, labelshulp) được dùng làm tên kênh; file chuẩn bị cho MACEC đọc được trực tiếp
.wav	WAV PCM hoặc float, mỗi track một kênh, f_s đọc từ file; số nguyên PCM được chia về ± 1 (nhân lại với toàn thang của thiết bị ở bước Convert)
.tdms	LabVIEW TDMS: mọi kênh số của mọi group (tên group/kênh khi có nhiều group); f_s từ thuộc tính wf_increment (hoặc fs, SampleRate); các kênh khác độ dài bị bỏ qua (có ghi chú)
.uff .unv	Universal File, dataset 58 (bản ghi thời gian, ASCII hoặc nhị phân 58b): mỗi dataset một kênh, tên theo nút và hướng đáp ứng (ví dụ 10Z), $f_s = 1/\Delta t$ của trục hoành

Khi file không có thông tin f_s (bảng không có cột thời gian, .mat không có fs), phải nhập f_s trong ô fs trước khi import. File nhị phân khác bị từ chối kèm thông báo; hãy xuất chúng sang một trong các định dạng trên bằng phần mềm thu dữ liệu.

Trong quy trình dự án OMA (khai báo kênh theo setup), file Parquet được ghép theo tên kênh nếu file có đủ các cột trùng tên; nếu không, các cột dữ liệu đầu tiên được lấy theo thứ tự kênh của setup.

3.4 Import và Convert – định dạng làm việc của Chladni

Import đọc file đo *một lần duy nhất* và lưu nó vào project ở định dạng nội bộ (file nén data.npz trong thư mục của mục: ma trận số thực mẫu $\times$ kênh và f_s). Từ đó mọi thao tác đọc thẳng mảng số này – không phân tích lại CSV hay Excel, không phụ thuộc file gốc còn hay mất.

Convert đưa tín hiệu thô về **tín hiệu làm việc** chuẩn mà mọi bước sau dùng: mỗi kênh có nhãn, loại đại lượng và giá trị theo đơn vị SI. Nhờ đó:

- mọi kênh cùng một hệ đơn vị: dạng mode, phổ và đồ thị so sánh được với nhau;

- loại đại lượng giúp chương trình tự chọn vai trò của kênh (gia tốc/vận tốc/chuyển vị là output, lực là input của EMA/OMAX) và ghi đúng đơn vị trên trục;
 - nhãn kênh (ví dụ 5X, 203T) được dùng xuyên suốt: bảng DOF, ghép theo nhãn, logfile.
- Với mỗi kênh, hệ số chuyển đổi là

$$x_{\text{SI}} = \frac{x_{\text{ghi}} \cdot k_{\text{ghi}}}{S \cdot A} k_{\text{SI}}, \quad A = 10^{A_{\text{dB}}/20} \text{ nếu khuếch đại tính bằng dB}, \quad (3.1)$$

với S là độ nhạy (đơn vị ghi trên một đơn vị vật lý, ví dụ V/g), A là khuếch đại, k_{ghi} đổi đơn vị ghi (V = 1, mV = 10^{-3} , μV = 10^{-6} , counts = 1) và k_{SI} đổi đơn vị vật lý sang SI (bảng 3.2). Khi dữ liệu đã ở đơn vị vật lý, để trống **Recorded in:** chỉ k_{SI} được áp dụng.

Bảng 3.2: Đơn vị vật lý và đơn vị SI tương ứng.

Đại lượng	Đơn vị nhập	Đơn vị làm việc
gia tốc (acc)	m/s ² , g, mm/s ² , gal	m/s ²
vận tốc (vel)	m/s, mm/s	m/s
chuyển vị (disp)	m, mm, μm	m
lực (force)	N, kN, lbf	N
biến dạng (strain)	$\mu\epsilon$	$\mu\epsilon$
khác (other)	–	như đã ghi

Điểm mẫu chốt

Dạng mode OMA không có tỷ lệ tuyệt đối, nên với dữ liệu gia tốc đã cùng đơn vị (dù không rõ là m/s² hay g) chỉ cần chọn đại lượng *acc* và để đơn vị gốc. Bảng chuyển đổi chỉ bắt buộc đúng khi các kênh có độ nhạy khác nhau, khi cần biên độ thật, hoặc khi có lực đo.

3.5 Hình học

Hình học gồm bốn bảng; mỗi bảng nhập trực tiếp, dán từ Excel, hoặc import từ file:

Bảng	Cột	Ghi chú
Nodes	id, x, y, z	nút đo và các điểm cần để vẽ kết cấu
Lines	id, n1, n2, group	đoạn thẳng (dầm)
Surfaces	id, n1, n2, n3, group	tam giác (tấm tứ giác chia thành 2 tam giác)
Slave DOFs	slave, slave_dir, master, master_dir, factor	$u_{\text{dir}}(\text{slave}) = \sum \text{factor} \cdot u_{\text{dir}}(\text{master})$

File nhập: CSV/TXT (có hoặc không có tiêu đề) và **file .asc của MACEC**: grid (node x y z), beam (n1 n2), surface (n1 n2 n3, nút thứ ba bằng 0 là dầm) và slave (master mx my mz slave sx sy sz: hướng master là véc-tơ đơn vị, hướng slave đã nhân hệ số; véc-tơ slave bằng 0 nghĩa là nút đứng yên). Hoặc import một mô hình **OpenSeesPy (.py)** / **Tcl (.tcl)**: nút, phần tử và dạng mode FE được đọc từ mô hình.

3.6 Bậc tự do đo (DOF) và hướng

Mỗi kênh của tín hiệu processed được gán một nút hình học, một hướng và cờ tham chiếu. Hướng nhận các dạng: x, y, z, -x, -y, -z; một véc-tơ a, b, c (được chuẩn hóa); hoặc hai góc: θ (phương vị trong mặt phẳng xy, tính từ x) và φ (góc nâng so với mặt phẳng xy), véc-tơ

($\cos \varphi \cos \theta$, $\cos \varphi \sin \theta$, $\sin \varphi$). Bảng DOF nhập từ CSV có các cột `label`, `node`, `dir`, `ref` (tùy chọn `theta`, `phi`), ghép theo nhãn kênh.

Kênh **tham chiếu (Ref.)** là kênh có mặt (cùng nút, cùng hướng) ở mọi setup; chúng được dùng cho SSI/ref và để ghép các setup.

3.7 Thư mục project

<code>projects/<slug>/</code>	
<code>project.json</code>	tên, mô tả, thông tin công trình, lịch sử
<code>drawings/</code>	bản vẽ, tài liệu đính kèm
<code>geometry/</code>	<code>nodes.csv</code> <code>lines.csv</code> <code>surfaces.csv</code> <code>slaves.csv</code> (+ <code>model.py</code> , <code>fe_modal.npz</code>)
<code>items/<id>/</code>	<code>item.json</code> (mô tả, nguồn gốc, thông số) + <code>data.npz</code> (mảng số)
<code>results/<key>/</code>	bộ mode: <code>run.json</code> , <code>modes.npz</code> (<method> <code>__Fn</code> , <code>__Xi</code> , <code>__Phi</code> , ...), <code>dofs.csv</code>
<code>presets/conversion/</code>	bảng chuyển đổi đã lưu
<code>logfile.py</code>	<code>presets/dofs/</code> bảng DOF đã lưu nhật ký thao tác, chạy lại được

Dữ liệu trong `data.npz`:

<code>raw/converted/processed</code>	<code>data</code> (mẫu $\times$ kênh), <code>fs</code>
mô hình tham số	<code>Fn</code> , <code>Xi</code> (cực $\times$ bậc), <code>Phi</code> (cực $\times$ bậc $\times$ output), <code>orders</code> , tùy chọn <code>Fn_std</code> , <code>Xi_std</code> , <code>sv</code>
mô hình phi tham số	<code>freq</code> , <code>S</code> (tần số $\times$ output $\times$ tham chiếu/input), <code>auto</code> , <code>std</code> , <code>coh</code>

3.8 Logfile và chế độ batch

Khi một thao tác được thực hiện trên giao diện, một dòng lệnh tương ứng được ghi vào `logfile.py` của project, ví dụ:

```
from chladni.workbench import batch as wb
p = wb.open_project('frame-4-storey')
wb.process(p, item='converted-001', ops=[{'op': 'offset', 'params': {}}])
wb.ssi(p, item='processed-002', method='dat', i=34, ordmin=2, ordmax=60, step=2,
       uncertainty=True)
wb.pick_poles(p, model='model-002', picks=[{'i': 0, 'j': 24}, ...])
```

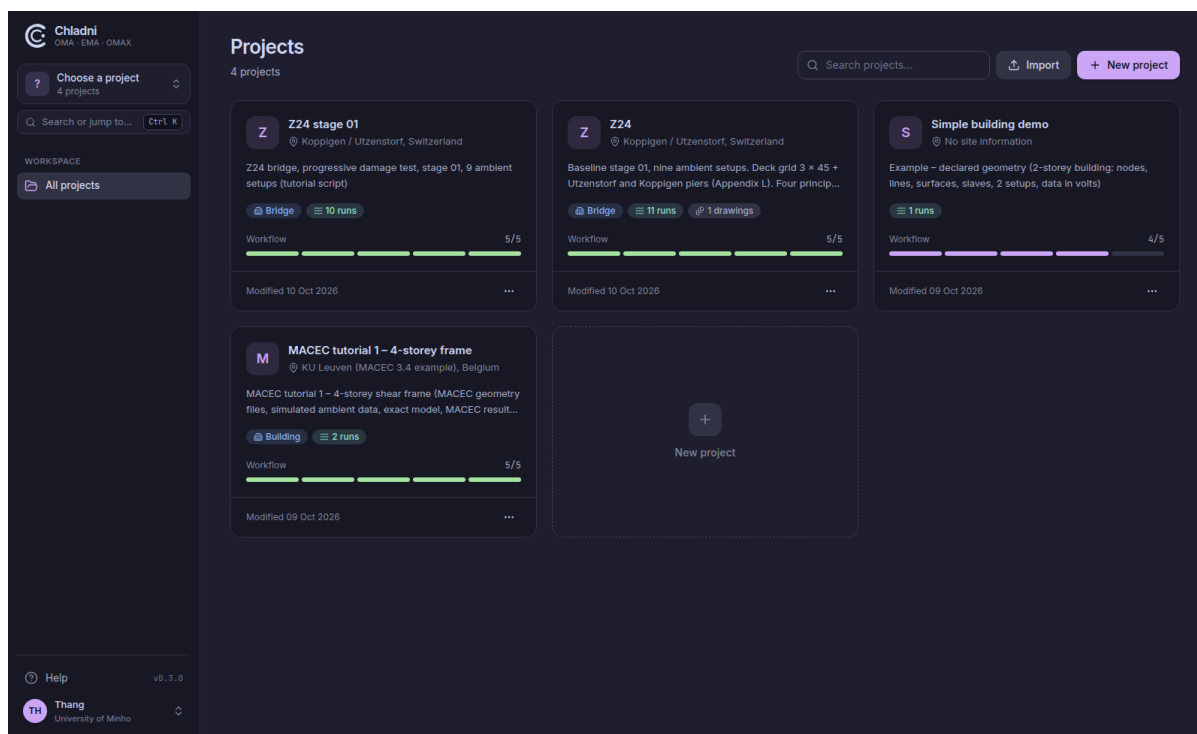
Logfile cho biết chính xác đã làm gì, cho phép làm lại toàn bộ phân tích mà không cần giao diện, và là mẫu để viết script phân tích (chương 5).

Chương 4

Giao diện

4.1 Bố cục

- **Thanh trái:** chọn project; các bước theo ba tầng phân tích – Geometry; Signal processing (Convert, Process & DOFs); System identification; Modal analysis; Results (List modes, Mode shapes, Compare modes). Chấm xanh: bước đã có kết quả.
- **Project explorer** (bên phải, ẩn/hiện bằng nút góc trên): các mục theo nhóm. Bấm để chọn; Ctrl+bấm chọn thêm; Shift+bấm chọn dải; bấm đúp hoặc Enter mở đúng trang; kéo mục vào trang để dùng nó; kéo file từ máy vào để import; chuột phải: đổi tên (F2), xem nguồn gốc, tải về, xóa (Delete). Chọn từ hai bộ mode trở lên hiện nút **Combine setups**. Nút **Logfile** mở nhật ký.
- Mỗi trang làm việc trên mục đang chọn ở Explorer (hoặc mục mới nhất cùng loại), có thể đổi bằng ô chọn ở đầu trang.

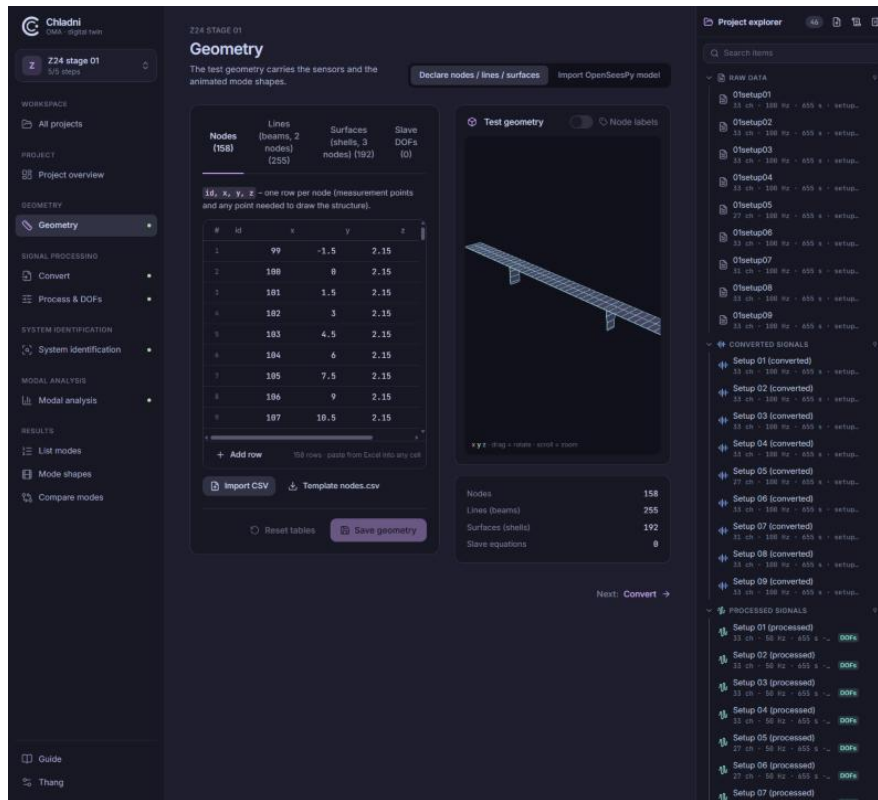


Hình 4.1: Trang Projects: mỗi thẻ là một project, thanh tiến độ cho năm bước.

4.2 Projects và tổng quan project

New project tạo project trống hoặc từ mẫu: *4-storey shear frame – benchmark* (chương 6), *Example – declared geometry* (nhà 2 tầng, dữ liệu volt có độ nhạy), *Example – OpenSeesPy model* (khung 3D). Menu “...” của thẻ: xuất ZIP, xóa; **Import** nhập project từ ZIP. Trang tổng quan của project chứa thông tin công trình, bản vẽ (kéo thả PDF/ảnh/DWG...), lịch sử và tiến độ.

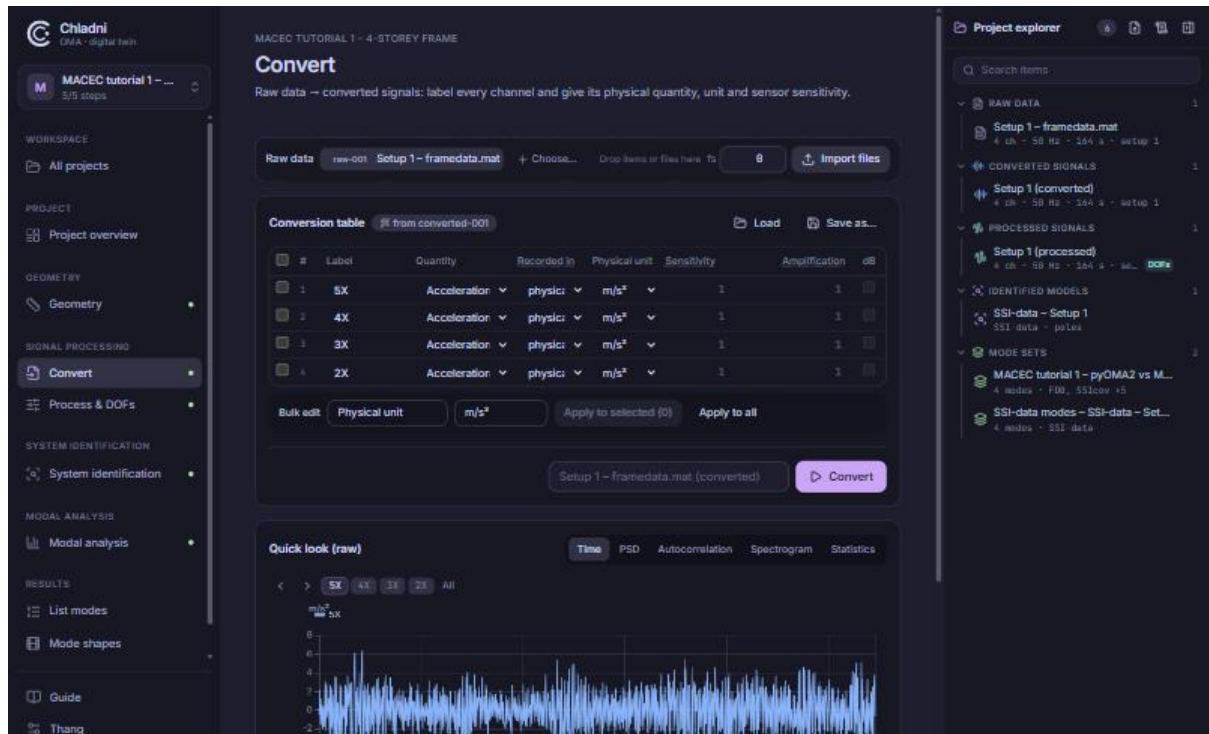
4.3 Geometry



Hình 4.2: Trang Geometry: bảng nút (trái), khung 3D (phải), thống kê.

Bốn tab **Nodes**, **Lines**, **Surfaces**, **Slave DOFs**; mỗi tab có **Import CSV** (nhận cả file .asc), **Template** (CSV mẫu), **Add row**; dán từ Excel vào ô bất kỳ. **Save geometry** kiểm tra lỗi (nút trùng, nút không tồn tại, hướng slave sai). Nút **Import OpenSeesPy model** chạy một script mô hình và lấy nút, phần tử, mode FE. **Node labels** hiện số nút trong 3D.

4.4 Convert

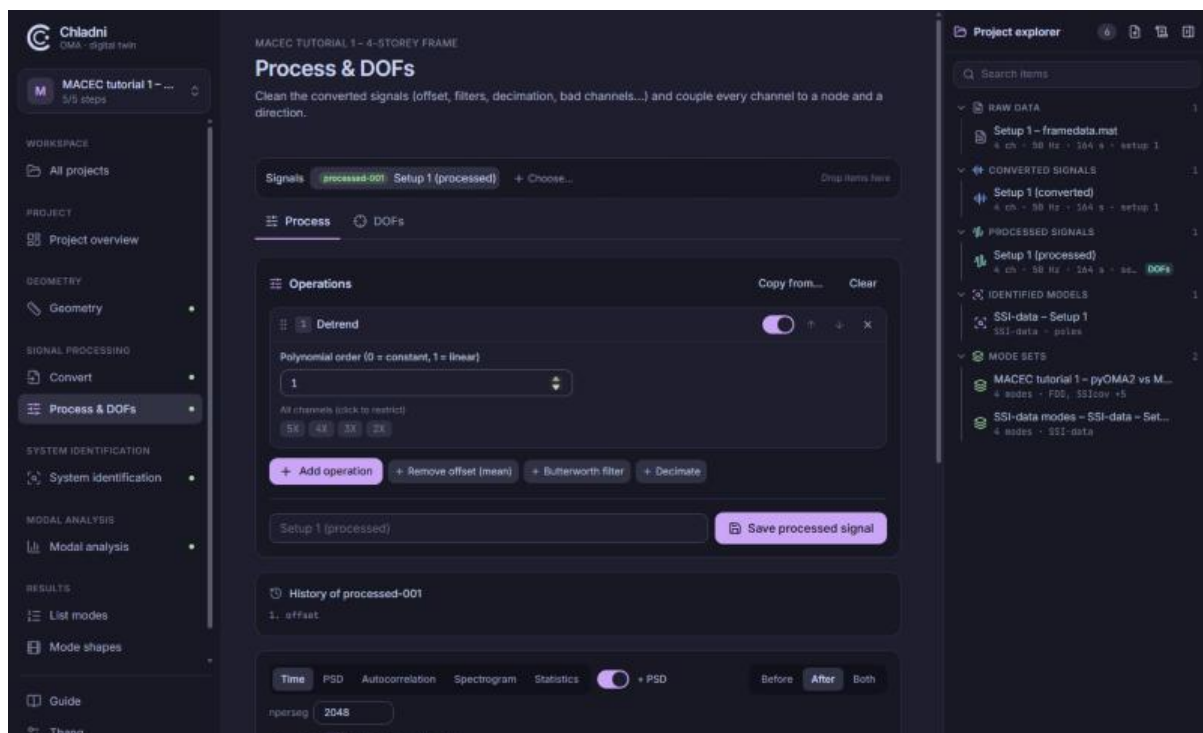


Hình 4.3: Trang Convert: mục raw đang chọn, bảng chuyển đổi, xem nhanh tín hiệu thô.

1. **Import files** (hoặc kéo file): xem bảng 3.1. Ô **fs**: 0 = lấy từ file.
2. Chọn một hoặc nhiều mục raw (nhiều setup cùng phần cứng được chuyển đổi một lần; mỗi setup giữ tên kênh của mình). Các mục được chọn phải có cùng số kênh.
3. Điền bảng: **Label**, **Quantity**, **Recorded in**, **Physical unit**, **Sensitivity**, **Amplification**, **dB**. **Bulk edit** gán một giá trị cho các dòng đã chọn. **Save as...** / **Load**: preset *conversion factors*.
4. **Convert** tạo các mục *Converted signals*.

Quick look (raw) cho xem Time, PSD, Autocorrelation, Spectrogram, Statistics. Khi project có mô hình FE, thẻ **Simulate from the FE model** tạo dữ liệu mô phỏng (kèm DOF) với thời lượng, f_s , hệ số cản và nhiễu cho trước.

4.5 Process & DOFs

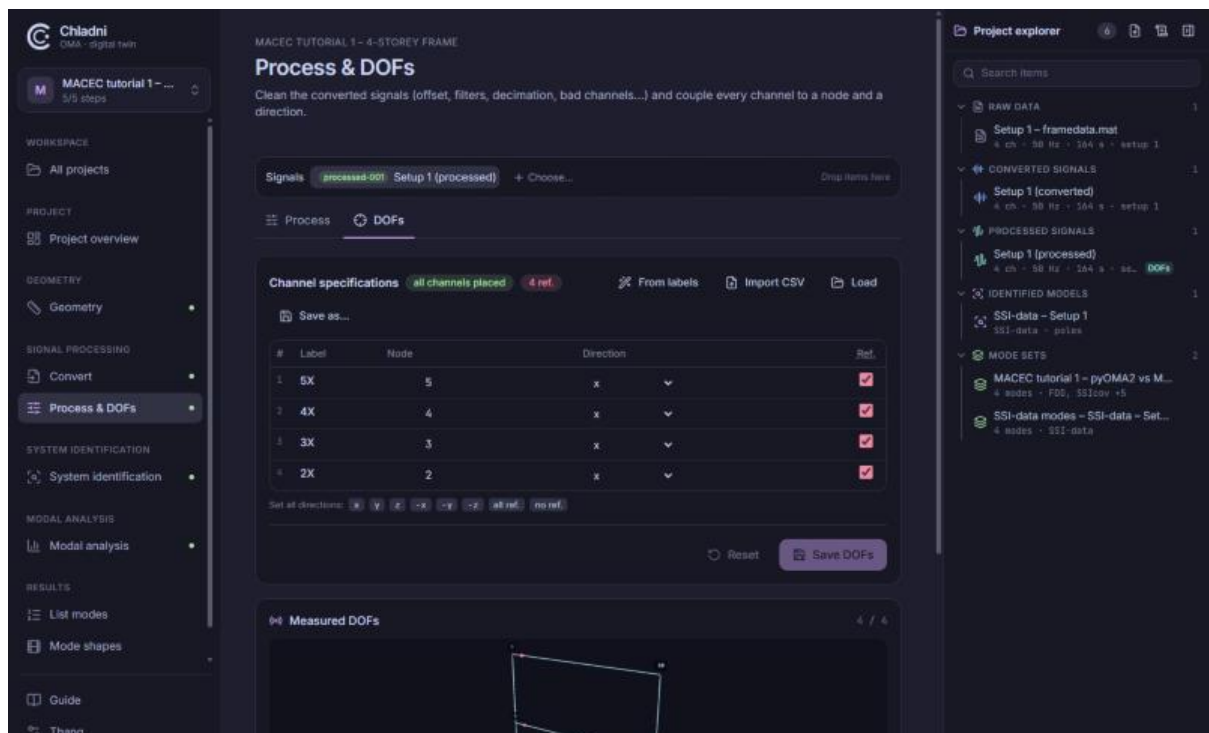


Hình 4.4: Tab Process: danh sách thao tác, lịch sử, xem trước kết quả.

Tab **Process**: thêm thao tác bằng **Add operation** hoặc các nút nhanh, kéo để đổi thứ tự, bật/tắt, chỉnh tham số; một số thao tác có thể chỉ áp dụng cho vài kênh. Kết quả hiện tức thì (**Before/After/Both**); **Save processed signal** áp dụng cho mọi mục đang chọn.

Bảng 4.1: Các thao tác tiền xử lý.

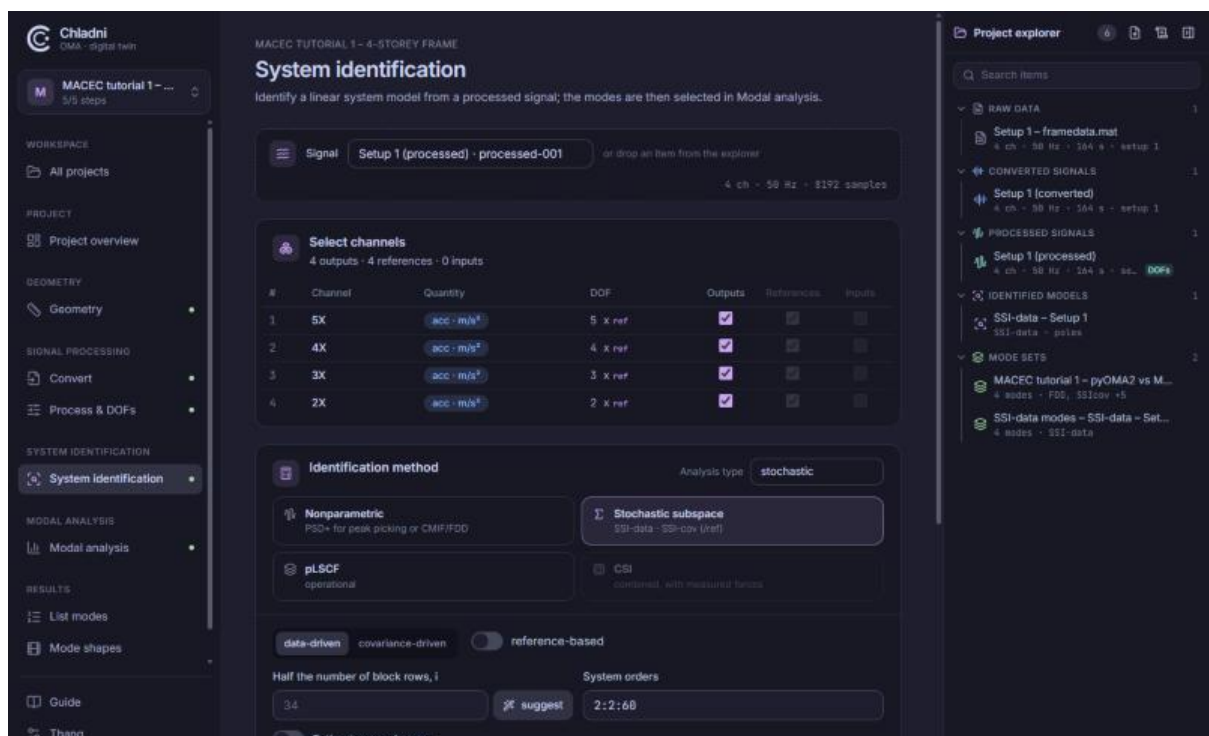
Thao tác	Tham số	Tác dụng
Delete channel(s)	kênh	loại kênh hỏng
Crop time window	bắt đầu, kết thúc [s]	giữ một đoạn bản ghi
Remove offset (mean)	—	trừ giá trị trung bình
Detrend	bậc đa thức	bỏ trôi
Remove spikes	ngưỡng $k\sigma$, nội suy/cắt	bỏ điểm bất thường
Butterworth filter	loại, f_{low} , f_{high} , bậc, zero-phase	lọc thông thấp/cao/dải/chắn
Notch filter	f_0 , Q , số họa tần	bỏ nhiễu điện lưới, máy
Decimate	hệ số q	lọc chống chồng phổ rồi giảm f_s
Resample	f_s mới	lấy mẫu lại (đồng bộ các setup)
Taper (Tukey)	α	làm mềm hai đầu
Integrate / Differentiate	số lần, tần số cắt	gia tốc $\leftrightarrow$ vận tốc $\leftrightarrow$ chuyển vị
Scale	hệ số	nhân mọi kênh



Hình 4.5: Tab DOFs: nút, hướng, tham chiếu của từng kênh; 3D các DOF đo.

Tab **DOFs**: nhập nút, hướng, tham chiếu; **From labels** đọc nhãn kiểu 5X, 203-y; **Import CSV** đọc bảng label, node, dir, ref; **Copy from...** lấy từ mục khác; **Save as...** / **Load** preset; **Save DOFs**. Khung 3D vẽ mũi tên theo hướng đo, tham chiếu màu đỏ.

4.6 System identification



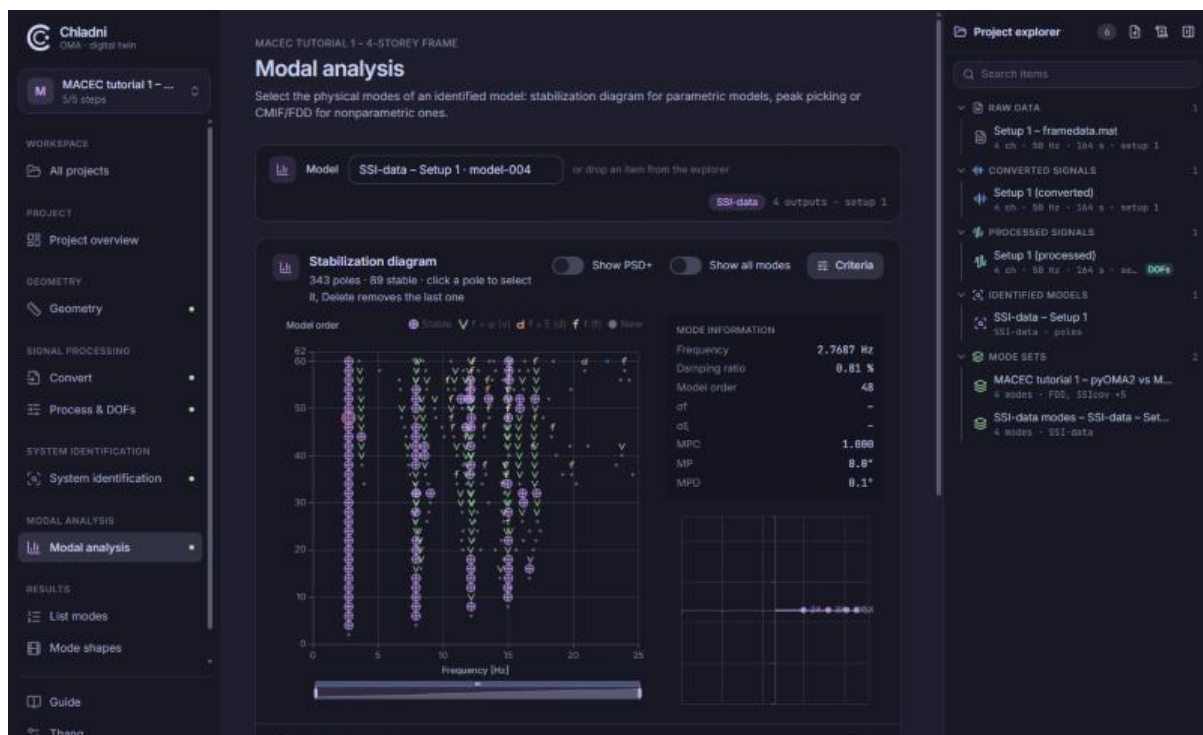
Hình 4.6: Trang System identification: chọn kênh, loại phân tích, phương pháp và thông số.

1. Chọn tín hiệu processed. Bảng **Select channels: Outputs, References, Inputs** (mặc định: kênh lực là input, các kênh còn lại là output; cờ tham chiếu lấy từ bảng DOF).
2. **Analysis type** tự gợi ý: *stochastic* (không có lực), *deterministic* hoặc *combined*.
3. Chọn phương pháp và thông số (bảng 4.2); **Resolution info** cho biết Δf , kích thước Hankel, bậc tối đa. **Calculate** chạy nền và tạo một mục *Identified model*.

Bảng 4.2: Phương pháp nhận dạng và thông số.

Phương pháp	Thông số
Nonparametric – PSD ⁺	periodogram: <i>Number of data blocks</i> N_b , <i>Time window parameter</i> m (giữ N_b/m trể đầu: ít phương sai hơn, độ phân giải thô hơn); correlogram: <i>Number of time lags</i>
Nonparametric – FRF (H1)	số khối; $H_1 = G_{yu}G_{uu}^{-1}$, kèm coherence
SSI	data-driven / covariance-driven; <i>reference-based</i> (chỉ dùng kênh tham chiếu); <i>half the number of block rows</i> i (nút suggest); <i>System orders</i> min:step:max ; <i>Estimate covariances</i> (độ bất định, số khối). Bậc tối đa tự giới hạn bởi $i \cdot n_{\text{ref}}$
pLSCF	operational (PSD) hoặc deterministic (FRF): bậc đa thức tối đa, độ dài đoạn, dải tần
CSI	i , bậc (cần ít nhất một kênh lực)

4.7 Modal analysis



Hình 4.7: Biểu đồ ổn định (trái), *Mode information* và dạng mode trên mặt phẳng phức (phải).

Mô hình tham số – biểu đồ ổn định. Mỗi cực được đánh dấu theo so sánh với bậc ngay dưới: $\oplus$ ổn định (tần số, cản và dạng mode), ν (tần số + dạng mode), d (tần số + cản), f (chỉ tần số),

· cực mới. **Criteria** (bảng 4.3) → **Apply. Show all modes** hiện cả cực vi phạm tiêu chí cứng; **Show PSD+** phủ tổng PSD. Rê chuột: *Mode information*; bấm: chọn/bỏ cực, dạng mode hiện trên mặt phẳng phức (mode thực: các véc-tơ trên một đường thẳng). Nên chọn các mode ở cùng một bậc. Phím Delete bỏ cực vừa chọn; **Save** tạo *Mode set*.

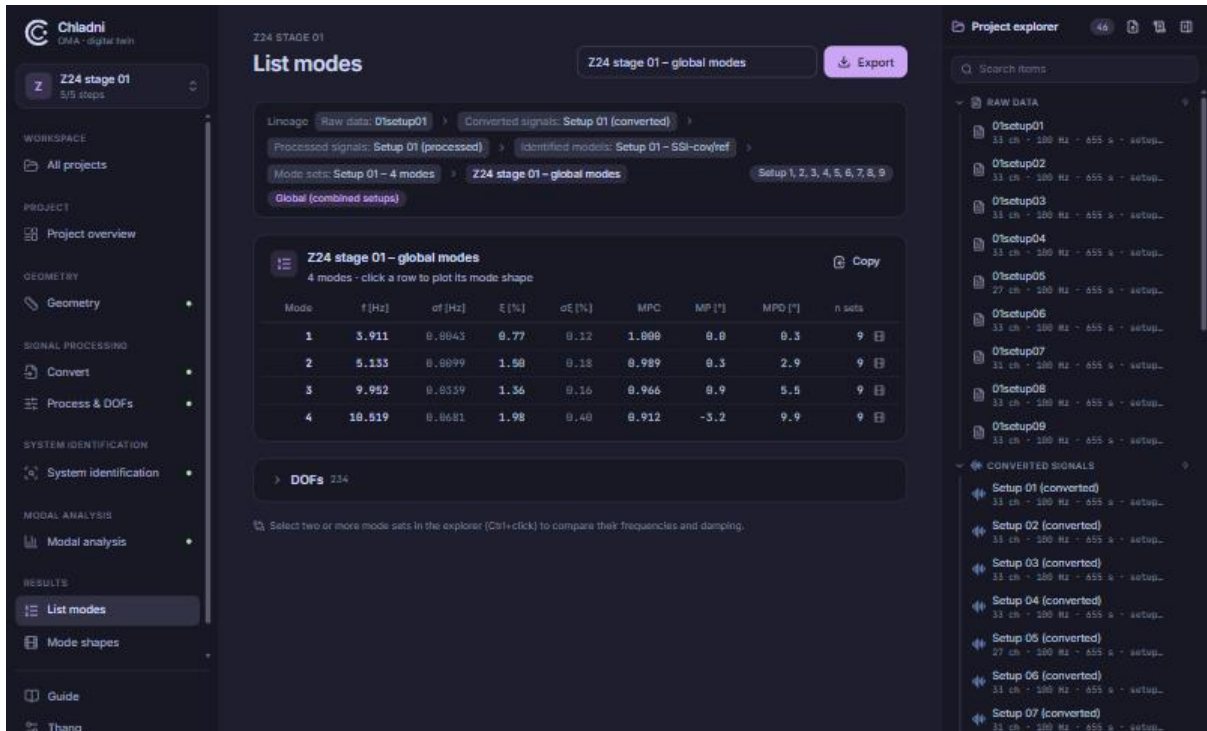
Bảng 4.3: Tiêu chí ổn định và giá trị mặc định.

Tiêu chí	Ý nghĩa	Mặc định
Δf	sai khác tần số giữa hai bậc liên tiếp	1 %
$\Delta \xi$	sai khác hệ số cản	5 %
ΔMAC	1 – MAC giữa dạng mode hai bậc	2 %
$\xi_{\min}, \xi_{\max}$	khoảng hệ số cản (tiêu chí cứng)	0–20 %
MPC min	modal phase collinearity tối thiểu	0
MP max, MPD max	mean phase, mean phase deviation tối đa	90°
σ_f max, σ_ξ max	độ lệch chuẩn tối đa (khi có ước lượng bất định)	tắt

Mô hình phi tham số. *Peak picking* trên ANPSD (trung bình các PSD đã chuẩn hóa) hoặc *CMIF/FDD* trên các giá trị kỳ dị (chọn SV thứ mấy cho mode gần nhau); rê chuột xem MPC, bấm để chọn (tự bắt đỉnh lân cận).

Combine setups: chọn các bộ mode của từng setup (Explorer hoặc thẻ **Combine setups**), dung sai tần số. Các DOF chung (cùng nút, cùng hướng) dùng để đưa từng setup về cùng tỷ lệ và pha (bình phương tối thiểu phức); dạng mode toàn cục chứa mọi DOF; f và ξ lấy trung bình kèm độ lệch chuẩn. Các bộ mode có cùng DOF (đo lặp) được lấy trung bình.

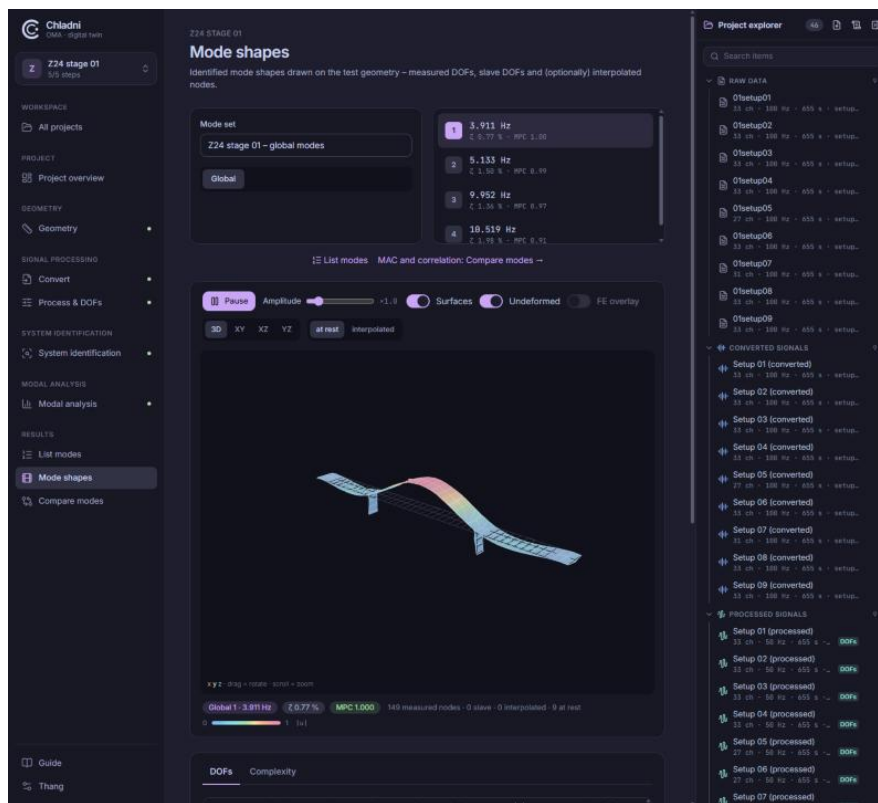
4.8 Results



Hình 4.8: List modes: nguồn gốc, bảng mode, xuất kết quả.

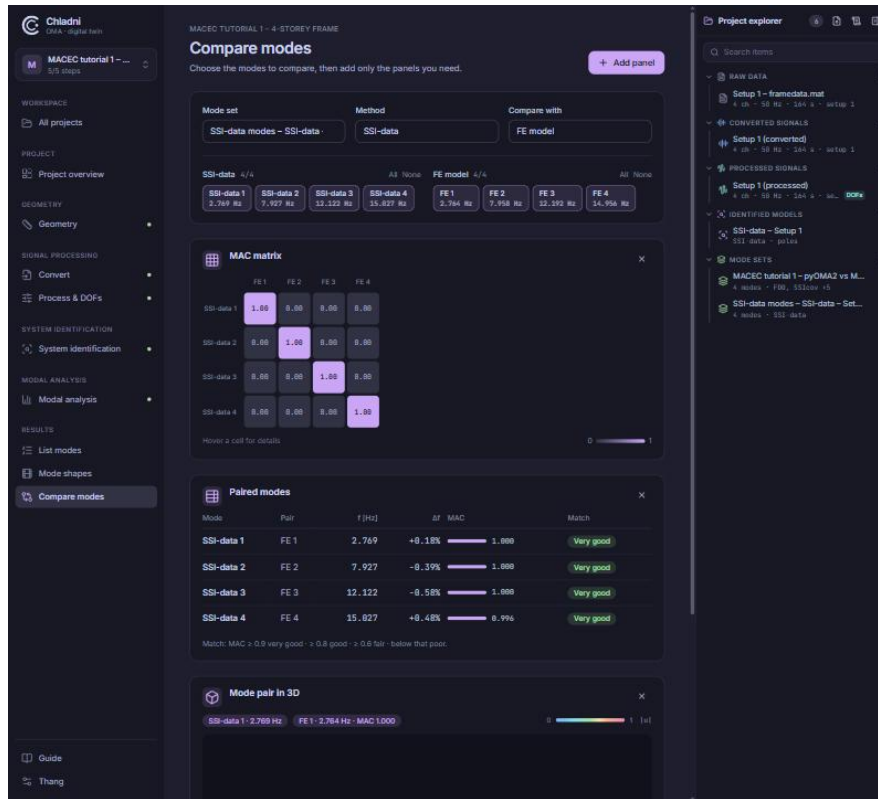
List modes: f , σ_f , ξ , σ_ξ , MPC, MP, MPD, bậc (giá trị đáng ngờ tô màu cam); bảng DOF; **Export:** modes.csv, UFF, ZIP, stabmodes.mat (tương thích MACEC), sao chép sang Excel. Chọn nhiều

bộ mode để so tần số và căn cạnh nhau.



Hình 4.9: Mode shapes: hoạt hình 3D trên hình học.

Mode shapes: hoạt hình 3D (biên độ, mặt nhìn 3D/XY/XZ/YZ, nút không đo đứng yên hoặc nội suy, phủ mode FE), các tab DOF và độ phức. Chuyển vị của nút được dựng từ các hướng đo (bình phương tối thiểu), sau đó áp phương trình slave.



Hình 4.10: Compare modes: ma trận MAC, cặp mode, cặp mode 3D.

Compare modes: chọn phương pháp A và đối tượng so sánh B (mô hình FE, phương pháp khác của cùng bộ mode, hoặc một bộ mode khác – MAC trên các DOF chung); thêm/bớt các bảng: ma trận MAC, cặp mode, cặp mode 3D, so sánh tần số, AutoMAC, hệ số cần.

Bảng 4.4: Các file xuất.

<code>modes.csv</code>	bảng mode
<code>dofs.csv</code>	DOF đo (nút, véc-tơ hướng, setup, tham chiếu)
<code><method>/mode_shapes_dofs.csv</code>	ϕ của từng DOF: phần thực, ảo, biên độ, pha
<code><method>/mode_shapes_nodes.csv</code>	chuyển vị u_x, u_y, u_z của từng nút hình học
<code><method>/modes.uff</code>	Universal File Format: dataset 15 (nút), 82 (đường), 55 (mode) – đọc được bằng ARTEMIS, MEScope, pyuff
<code><method>/mode_k_f.html</code>	hoạt hình mở bằng trình duyệt
<code>stabmodes.mat</code>	định dạng kết quả tương thích MACEC (f , ξ , m , mpc , mp , mpd , $node_num$)

Chương 5

Danh mục hàm (chế độ batch)

Module `chladni.workbench.batch` có cùng các thao tác như giao diện. Mọi hàm nhận project trước (kết quả của `open_project`) và trả về id của mục mới. Chạy script từ thư mục Chladni:
`uv run python script.py`.

Hàm	Mô tả
<code>open_project(slug)</code>	mở project theo tên thư mục
<code>import_file(p, path, fs=None, setup=None)</code>	đọc một file đo (mục 3.3) → raw
<code>simulate(p, rows, duration, fs, damping, noise_pct, seed)</code>	dữ liệu mô phỏng từ mô hình FE → raw + converted
<code>convert(p, item, rows, name=None)</code>	bảng chuyển đổi: mỗi dòng label, quantity, raw_unit, unit, sensitivity, amplification, db → converted
<code>process(p, item, ops, name=None)</code>	danh sách thao tác {op, enabled, params} (bảng 4.1; op: delete, crop, offset, detrend, despiking, filter, notch, decimate, resample, taper, integrate, differentiate, scale) → processed
<code>set_dofs(p, item, rows)</code>	mỗi kênh {node, dir, ref, theta?, phi?}
<code>nonparametric(p, item, outputs, refs, inputs, method='per' 'cor', blocks, m, lags)</code>	PSD ⁺ hoặc FRF → model
<code>ssi(p, item, method='dat' 'cov', outputs, refs, i, ordmin, ordmax, step, uncertainty, blocks)</code>	SSI → model
<code>plscf(p, item, outputs, inputs, ordmax, nxseg, method_SD, pov, f_lo, f_hi)</code>	pLSCF → model
<code>csi(p, item, outputs, inputs, i, ordmin, ordmax, step)</code>	CSI → model
<code>pick_poles(p, model, picks)</code>	picks = [{i: cực, j: chỉ số bậc}] → bộ mode
<code>pick_peaks(p, model, freqs, method='PP' 'FDD', sv)</code>	chọn vạch tần số trên mô hình phi tham số → bộ mode
<code>combine(p, sets, name, freq_tol)</code>	ghép các bộ mode của nhiều setup → bộ mode toàn cục

Các hàm tính toán dùng được độc lập:

<code>signals.factors(rows)</code>	hệ số chuyển đổi, đại lượng, đơn vị SI của từng kênh
<code>signals.apply_ops(ds, ops)</code>	áp dụng thao tác lên một tín hiệu (không lưu)
<code>ident.psd_plus(x, fs, outputs, refs, method, blocks, m, lags)</code>	PSD ⁺ (periodogram/correlogram)
<code>ident.suggest_i(ds, n_out, n_ref)</code>	gợi ý <i>i</i> cho SSI
<code>emax.h1(y, u, fs, blocks)</code>	FRF H1 và coherence
<code>emax.plscf_det(freq, H, fs, orders)</code>	pLSCF tất định
<code>modal.stabilization(arr, Criteria(...))</code>	mã ổn định của mọi cực
<code>modal.mpc(phi), modal.mp_mpd(phi), modal.mac_vec(a, b)</code>	chỉ số chất lượng dạng mode
<code>modal.anpsd(arr), modal.cmif(arr, info)</code>	ANPSD, giá trị kỳ dị CMIF

5.1 Ví dụ

```
from chladni.workbench import batch as wb

p = wb.open_project("my-bridge")
raw = wb.import_file(p, path="D:/data/setup1.csv", fs=200)
conv = wb.convert(p, item=raw, rows=[
    {"label": "1V", "quantity": "acc", "raw_unit": "V", "unit": "g",
     "sensitivity": 1.0, "amplification": 1, "db": False}, ...])
proc = wb.process(p, item=conv, ops=[
    {"op": "detrend", "params": {"order": 1}},
    {"op": "filter", "params": {"btype": "bandpass", "f_low": 0.5, "f_high": 30}},
    {"op": "decimate", "params": {"q": 2}}])
wb.set_dofs(p, item=proc, rows=[{"node": 1, "dir": "z", "ref": True}, ...])
model = wb.ssi(p, item=proc, method="cov", i=40, ordmin=2, ordmax=60, step=2)
modes = wb.pick_poles(p, model=model, picks=[{"i": 3, "j": 25}, {"i": 7, "j": 25}])
```

Script hoàn chỉnh cho một trường hợp thật: `Examples/z24/scripts/z24_analysis.py`.

Chương 6

Ví dụ kiểm chứng

6.1 Khung cắt 4 tầng

Project mẫu *4-storey shear frame – benchmark* dùng dữ liệu, hình học và mô hình chính xác công bố trong bài 1 của sổ tay MACEC [1]: nhà cắt 4 bậc tự do, $m = 2 \text{ kg/tầng}$, hai cột $k = 2500 \text{ N/m}$, $\mathbf{C} = \mathbf{K}/1000$, $f_s = 50 \text{ Hz}$, 8192 mẫu.

Bảng 6.1: Tần số [Hz]: nghiệm chính xác, MACEC (sổ tay, bảng 4.1) và Chladni (SSI-data $i = 34$, chọn ở bậc 48).

Mode	Chính xác	MACEC SSI-data	MACEC pLSCF	Chladni SSI-data
1	2,764	$2,769 \pm 0,009$	2,768	2,769
2	7,958	$7,932 \pm 0,039$	7,921	7,936
3	12,192	$12,185 \pm 0,248$	12,162	12,185
4	14,956	$15,030 \pm 0,272$	14,965	15,027

Hệ số căn ở bậc 48: Chladni 0,81 / 1,91 / 3,49 / 5,40 %, MACEC 0,81 / 1,93 / 3,65 / 5,50 %, chính xác 0,87 / 2,50 / 3,83 / 4,70 %. MAC giữa dạng mode của Chladni và của MACEC $\geq 0,999$; với dạng mode chính xác $\geq 0,997$. Chi tiết: case study *Khung cắt 4 tầng* trong ứng dụng.

6.2 Hệ 2 bậc tự do có lực đo (EMA/OMAX)

Ví dụ 1 của sổ tay MACEC: $\mathbf{M} = \text{diag}(2, 2)$, $\mathbf{C} = \begin{bmatrix} 5 & -2 \\ -2 & 5 \end{bmatrix}$, $\mathbf{K} = \begin{bmatrix} 6000 & -2000 \\ -2000 & 6000 \end{bmatrix}$, lực ngẫu nhiên ở cả hai bậc tự do, gia tốc đo ở $f_s = 100 \text{ Hz}$ với 1 % nhiễu.

Bảng 6.2: Kiểm chứng EMA/OMAX.

	f_1 [Hz]	ξ_1 [%]	f_2 [Hz]	ξ_2 [%]
Chính xác	7,1176	1,677	10,0658	2,767
H1 + pLSCF tất định ($N = 16384$)	7,118–7,120	1,72–1,73	10,059–10,067	2,78–2,82
CSI ($N = 8192$, $i = 10$)	7,1176	1,676	10,0658	2,766

MAC với dạng mode chính xác $\geq 0,9999$ cho cả hai phương pháp. Hệ số căn của pLSCF lệch khoảng 3 % do rò phổ (cửa sổ Hann); với bản ghi ngắn (8192 mẫu, 16 khối) sai số này lên tới khoảng 14 %.

Các kiểm chứng trên chạy tự động trong bộ test (`uv run pytest`).

Phụ lục A

Thông báo lỗi thường gặp

Thông báo	Cách xử lý
<i>no time column – enter the sampling frequency</i>	nhập f_s khi import
<i>no 'fs' variable in the file unsupported binary file</i>	nhập f_s , hoặc lưu fs cùng dữ liệu trong file .mat xuất sang CSV, XLSX, Parquet, MAT, WAV, TDMS hoặc UFF 58
<i>Parquet has no time column – enter the sampling frequency</i>	nhập f_s khi import
<i>Parquet time column is not uniformly sampled</i>	lấy mẫu lại dữ liệu theo bước đều trước khi import
<i>Parquet has multiple time columns</i>	giữ lại đúng một cột thời gian (giây)
<i>Parquet measurement columns must be numeric</i>	bỏ hoặc chuyển sang số các cột văn bản, bool, số phức
<i>Parquet contains missing or non-finite samples</i>	sửa các ô trống, NaN hoặc vô cực trong file
<i>Parquet channel names must be nonempty and unique</i>	đặt tên riêng cho từng cột
<i>N conversion rows for M channels</i>	các mục được chọn có số kênh khác nhau: chuyển đổi theo nhóm
<i>Nodes not in the geometry for channel(s) ...</i>	sai số nút trong bảng DOF, hoặc hình học đã đổi
<i>Reference channels must be among the outputs</i>	bỏ cờ tham chiếu ở kênh không phải output
<i>CSI needs at least one input (force) channel ... changes the sampling – apply it to all channels</i>	đặt đại lượng <i>force</i> cho kênh lực ở bước Convert decimate/resample/crop phải áp dụng cho mọi kênh
<i>has no DOF in common with the previous setups</i>	các setup cần ít nhất một DOF tham chiếu chung (cùng nút, hướng)
<i>Eigenvalue analysis failed (import mô hình)</i>	mô hình thiếu khối lượng hoặc thiếu gối

Phụ lục B

Viết tắt

ANPSD	averaged normalized power spectral density
CMIF	complex mode indication function
CSI	combined deterministic-stochastic subspace identification
DOF	degree of freedom – bậc tự do
EMA / OMA / OMAX	experimental / operational / operational with exogenous inputs modal analysis
FDD	frequency domain decomposition
FRF	frequency response function
MAC / MPC / MP / MPD	modal assurance criterion / modal phase collinearity / mean phase / mean phase deviation
pLSCF	poly-reference least squares complex frequency-domain
PP	peak picking
PSD ⁺	positive power spectral density
SSI-data / SSI-cov (/ref)	data-/covariance-driven stochastic subspace identification (reference-based)
UFF	universal file format

Tài liệu tham khảo

- [1] E. Reynders, M. Schevenels, G. De Roeck. *MACEC 3.4: The Matlab toolbox for experimental and operational modal analysis*. KU Leuven, 2021.
- [2] D. P. Pasca, A. Aloisio, M. M. Rosso, S. Sotiropoulos. PyOMA and PyOMA_GUI: A Python module and software for Operational Modal Analysis. *SoftwareX*, 20:101216, 2022.
- [3] M. Zhu, F. McKenna, M. H. Scott. OpenSeesPy: Python library for the OpenSees finite element framework. *SoftwareX*, 7:6–11, 2018.
- [4] B. Peeters, G. De Roeck. Reference-based stochastic subspace identification for output-only modal analysis. *Mechanical Systems and Signal Processing*, 13(6):855–878, 1999.