
Digital Standardization of Seismic Compliance: An Open and Automated Framework for Non-Structural Systems

N. Shokrollahi, & M. Bishop

Brevity Ltd, Auckland, New Zealand

ABSTRACT

The seismic design and compliance verification of architectural non-structural elements (NSE) remains fragmented due to reliance on disparate proprietary methodologies. This fragmentation leads to inconsistent safety margins, limited design reproducibility, and elevated regulatory risk. This paper presents a standardised, open-source computational framework that automates the end-to-end seismic design and verification process through a defined Automated Compliance Pathway (ACP). The framework integrates four core computational modules: automated geometry system creation, parametric engineering methods, structural analysis using open-source finite element analysis (FEA) libraries, and automated compliance verification against regulatory performance criteria. By embedding machine learning, mathematical algorithms, and Python standard function tools within a collaborative GitHub environment under a permissive open-source licence, the proposed framework ensures consistency in design practice, enhances traceability in compliance verification, and accelerates the adoption of digital engineering standards for NSE.

1 INTRODUCTION

1.1 Background and Motivation

Non-structural elements in buildings, while not part of the primary load-carrying system, represent a significant portion of total building value and pose substantial risks during seismic events (Taghavi and Miranda 2003). Industrial storage racking systems, suspended ceiling grids, and partition walls are ubiquitous across commercial, industrial, and institutional facilities. Their failure during earthquakes can result in life safety hazards, business continuity disruption, and substantial economic losses, typically accounting for 50–70% of total earthquake-related building repair costs (Miranda et al. 2012; Dhakal 2010).

Despite their critical importance, the seismic design of NSE remains characterised by:

- Fragmented methodologies with inconsistent assumptions
- Limited reproducibility due to closed-source calculations
- Regulatory uncertainty from varying code interpretations

- Insufficient documentation for compliance verification

1.2 Research Objectives

This research develops a comprehensive framework that:

1. **Automated Geometry Creation:** From AutoCAD section extraction to whole system generation
2. **Parametric Engineering Methods:** Parametric design workflows for NSE structural analysis
3. **Analysis:** Integration of FEA and hand-calculation analysis to parametric systems
4. **Compliance:** Automated compliance verification against regulatory criteria

1.3 Scope of Application

The framework currently supports three categories of architectural NSE, each with distinct analysis requirements ([Table 1](#)).

Table 1: Scope of application for the three categories of architectural non-structural elements

System Type	Parametric Variables	Analysis Method	AI/ML Applications
Industrial Storage Racking	Bay count, levels, bracing type, geometry	3D FEA (Pynite)	Parametric modelling, modal analysis
Suspended Ceiling Grids	Grid spacing, tile dimensions, brace pattern	Code-based calculations	Voronoi optimisation, K-means clustering
Interior Partitions	Stud spacing, height, bracing requirements	Code-based calculations	Placement optimisation

2 FRAMEWORK ARCHITECTURE

2.1 Parametric Design Principles

The framework is built upon parametric design principles, where structural systems are defined by a minimal set of independent parameters from which all geometric and structural properties are derived. This approach enables design exploration through rapid evaluation of multiple configurations, optimisation through systematic improvement of structural performance, sensitivity analysis for understanding parameter influence on outcomes, and automation of end-to-end processing without manual intervention.

2.2 System Architecture

The Automated Compliance Pathway (ACP) framework comprises four interconnected computational modules operating on parametric definitions ([Figure 1](#)).

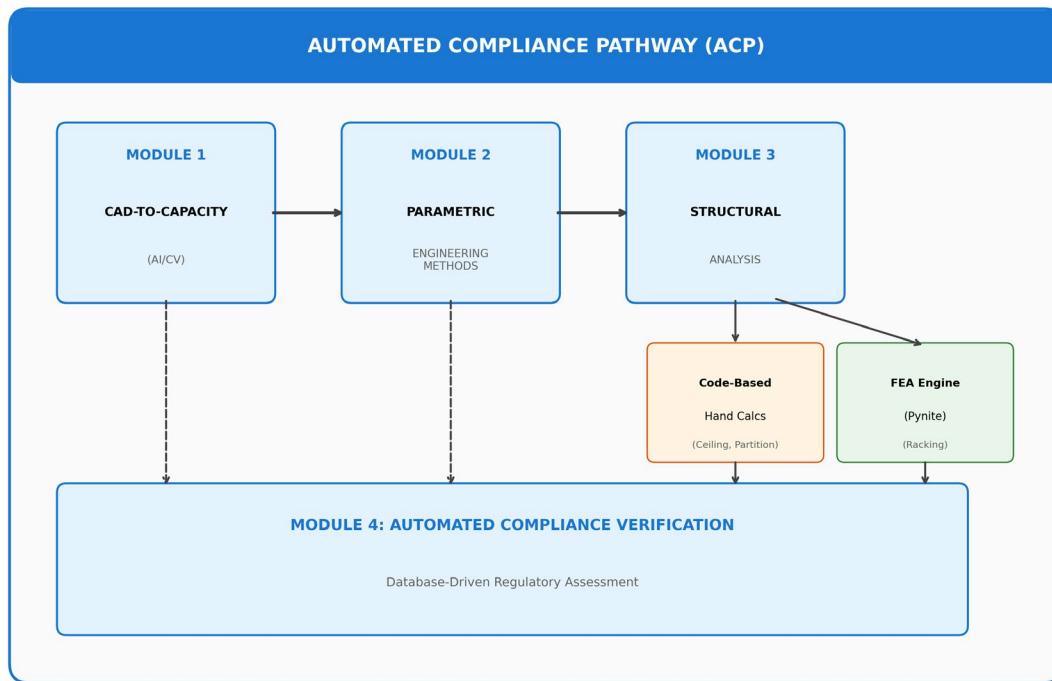


Figure 1: Framework architecture showing the four computational modules of the Automated Compliance Pathway (ACP)

The structural analysis approach varies by NSE type. Industrial storage racking requires full three-dimensional finite element analysis (FEA) due to its structural complexity. Suspended ceiling grids and partition walls utilise code-based hand calculations as prescribed by relevant standards.

2.3 Data Flow Architecture

The framework implements a unidirectional parametric data flow through the four computational modules (Table 2).

Table 2: Parametric data flow through the four computational modules

Stage	Input	Processing	Output
1. Geometry	AutoCAD DXF	Computer Vision	Section Properties
2. Definition	GeoJSON Boundaries	Parametric Generation	Layout Geometry
3. Analysis	Parametric Model	FEA or Code-Based Calculations	Structural Response
4. Verification	Analysis Results	Database + Standard Library	Compliance Status

3 AUTOMATED GEOMETRY CREATION

The framework automates the extraction of structural section properties directly from native AutoCAD DXF files using computer vision techniques, eliminating manual property lookup and ensuring actual manufactured section geometries are used in structural calculations. The pipeline integrates Harris corner detection (Harris and Stephens 1988) for geometric feature extraction, Shapely-based polygon processing (Gillies et al. 2007)

for computational geometry operations, and the SectionProperties library (King 2019) for structural property calculation including cross-sectional area, centroid coordinates, second moments of area, section moduli, and warping properties for cold-formed steel sections (Figure 2). Computed section properties are automatically exported to a Firebase/Firestore database (Google 2014), creating a continuously growing capacity library that enables real-time property lookup during design.

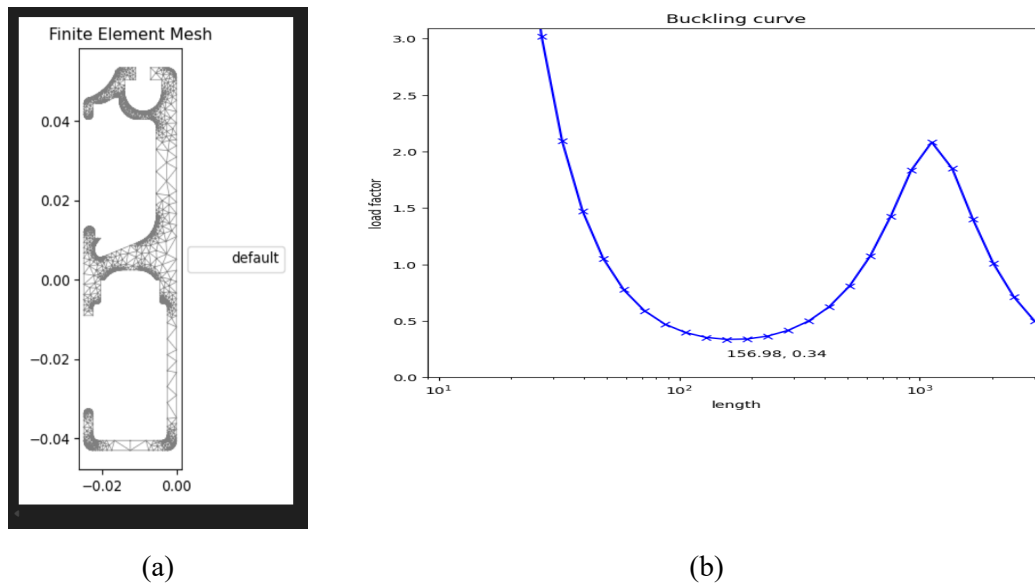


Figure 2: Computer vision processing pipeline: (a) Section property extraction showing detected corner points and computed geometric properties; (b) Cross-section mesh generation for finite element analysis of cold-formed steel profiles.

4 PARAMETRIC ENGINEERING METHODS

4.1 Engineering Cell Architecture

The framework employs modular “engineering cells”—self-contained calculation units that implement specific engineering standards. Each cell is characterised by a unique identifier for version tracking, input parameters, output variables, calculation logic, dependencies referencing prerequisite cells, and metadata including country, standard, version, and author.

4.2 Parametric Quantity Surveying System

The framework implements a parametric calculator pattern for component quantity estimation based on geometric parameters. Calculators are available for frame components, support elements, panels, and fasteners, with outputs including specimen counts and total lengths.

4.3 AI and Mathematical Algorithm for Seismic Design Components

Mathematical and AI algorithm could do complex engineering problems. In NSE application, the framework employs Voronoi diagrams (Fortune 1987) for optimal seismic brace placement in ceiling grids, ensuring each brace supports a roughly equal ceiling area while satisfying regulatory constraints. The algorithm initialises brace positions using K-means clustering (Lloyd 1982) on grid intersection points, generates Voronoi tessellation, clips cells to the ceiling polygon boundary, and iteratively adds braces at cell centroids where the area exceeds the maximum allowable limit until all cells satisfy the constraint (Figure 3).

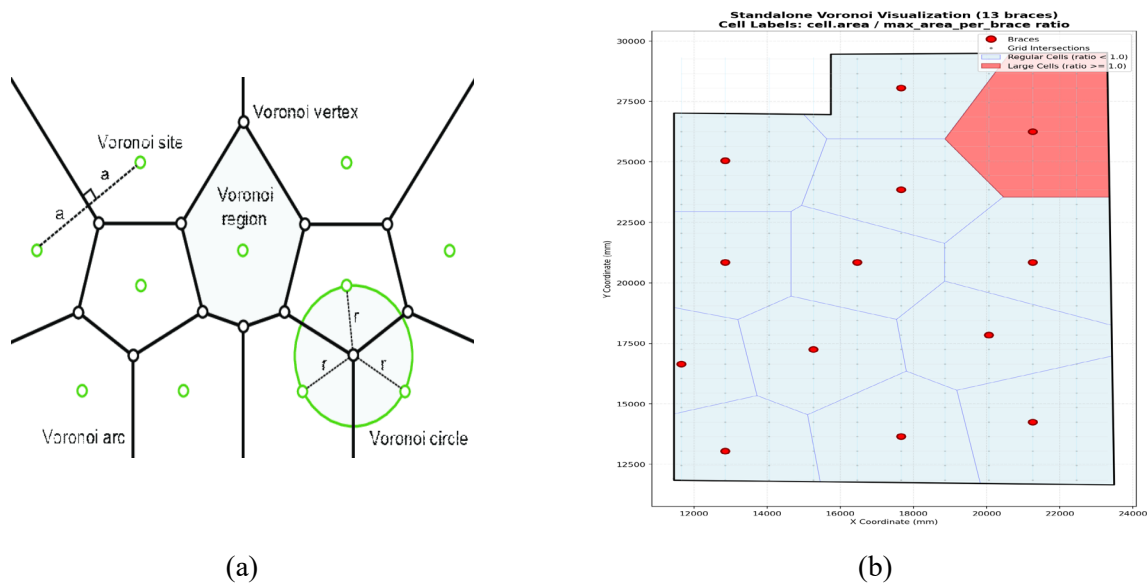


Figure 3: Voronoi-based seismic brace placement: (a) Algorithm schematic showing tessellation of ceiling polygon into cells with uniform area coverage; (b) Example application demonstrating optimised brace positions for an irregular ceiling geometry.

5 ANALYSIS

5.1 Parametric Racking System

The industrial storage racking system is defined by a minimal set of independent parameters including number of bays, number of levels, bay length, cross-aisle spacing, and level heights. The framework supports three parametric bracing configurations—Standard, X-bracing, and Z-bracing patterns—with all geometric relationships derived programmatically from the rack dimensions. For structural analysis, the framework incorporates the Pynite finite element library (Wock 2019), an open-source 3D structural analysis engine capable of full spatial structural modelling, modal analysis for natural frequencies and mode shapes, and P-Delta effects for geometric nonlinearity consideration ([Figure 4](#)).

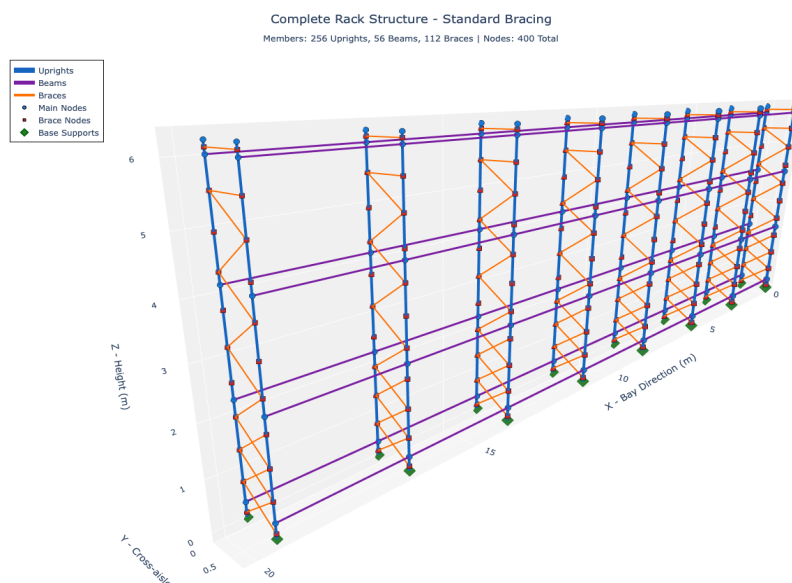


Figure 4: Parametric racking system model generated from input parameters showing uprights, beams, braces, and support nodes with standard bracing configuration.

5.2 Parametric Ceiling Grid System

The suspended ceiling grid system is parametrically defined by main tee spacing, cross tee spacing, grid margin offsets, and ceiling polygon boundary extracted from architectural drawings. The framework automatically generates the complete grid layout, calculates component quantities, and places seismic braces using the Voronoi-based optimisation algorithm to ensure code-compliant restraint spacing (Figure 5).

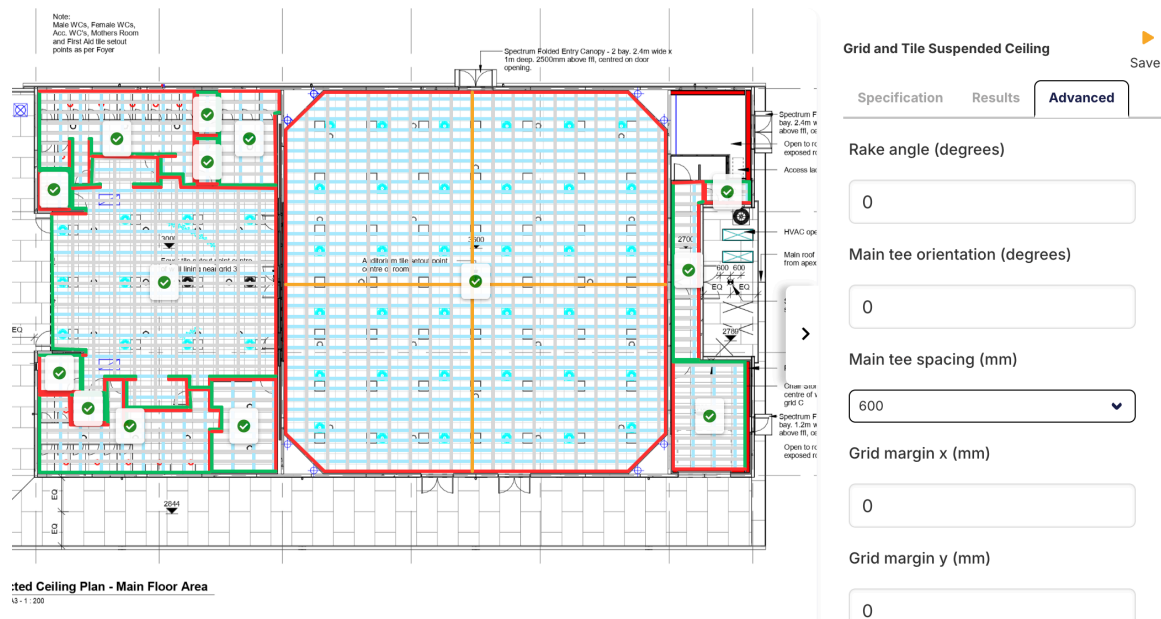


Figure 5: Parametric ceiling grid system showing automated grid generation with seismic brace placement optimised using Voronoi tessellation.

5.3 Load Analysis

The parametric load system applies forces based on model parameters. Gravity loads include self-weight automatically computed from section properties and material density, and pallet loads as level-specific loads for each bay. Seismic loads are calculated as level-specific horizontal forces per NZS 1170.5 (Standards New Zealand 2004).

6 COMPLIANCE

6.1 Compliance Service Architecture

The compliance module orchestrates all verification checks against regulatory criteria. The workflow proceeds from analysis results (demands) through the capacity database (capacities) to engineering cell execution, culminating in a comprehensive compliance report including utilisation ratios, pass/fail status, and recommendations.

6.2 Capacity Check Framework

Each structural component undergoes capacity verification per applicable standards. The framework retrieves capacity values from two sources: a capacity database with pre-computed section capacities stored in Firebase/Firestore populated via the CAD-to-capacity pipeline, and open-source standard libraries implementing engineering calculations that codify building standards. This approach ensures that capacity calculations are transparent, verifiable, consistent, and auditable.

6.3 Automated Report Generation

The framework generates comprehensive compliance reports including utilisation ratios for all components, critical member identification, drift and deflection verification, overall pass/fail determination, and improvement recommendations.

7 OPEN-SOURCE FOUNDATION

7.1 Technology Stack

The framework is built entirely upon established open-source libraries, installed via standard Python package management. This approach ensures transparency (all underlying algorithms are publicly available and peer-reviewed), reproducibility (standard libraries guarantee consistent behaviour across deployments), maintainability (community-supported packages receive regular updates and security patches), and verifiability (engineers can inspect the source code of any calculation). The open-source dependencies are summarised in Table 3.

Table 3: Open-source library dependencies

Category	Library	Purpose	Installation
Geometry	Shapely	Computational geometry operations	pip install shapely
Numerics	NumPy	Array operations and linear algebra	pip install numpy
Scientific	SciPy	Voronoi tessellation, spatial algorithms	pip install scipy
Machine Learning	scikit-learn	K-means clustering, KNN	pip install scikit-learn
Computer Vision	OpenCV	Harris corner detection, image processing	pip install opencv-python
Section Analysis	SectionProperties	Cross-section property calculation	pip install sectionproperties
CAD Processing	ezdxf	DXF file reading and manipulation	pip install ezdxf
FEA	Pynite	3D structural frame analysis	pip install PyNiteFEA

7.2 Reproducible Engineering Methods

The engineering methodology is documented in Jupyter Notebooks that serve as transparent, reproducible calculation records. Notebooks are available for industrial racking (FEA workflow, load application, compliance checks), suspended ceilings (brace placement algorithms, seismic verification), interior partitions (stud layout, bracing requirements), and fences/retaining walls (wind and seismic loading). These notebooks provide full transparency into the engineering calculations, allowing independent verification and audit of compliance assessments.

8 CONCLUSIONS

This research presents a comprehensive computational framework for the automated seismic design and compliance verification of architectural non-structural elements, built upon established open-source libraries. The key contributions include:

1. **Automated Geometry Creation:** Computer vision techniques enable automated section property extraction from native DXF files, eliminating manual property lookup and ensuring actual manufactured section geometries are used in structural calculations.
2. **Parametric Engineering Methods:** Voronoi tessellation combined with K-means clustering provides optimal seismic brace placement in ceiling systems, ensuring uniform load distribution while satisfying regulatory area constraints.
3. **Analysis:** Integration of open-source analysis libraries (Pynite for FEA, SectionProperties for sections) with parametric model definition enables rapid design exploration and sensitivity analysis.
4. **Compliance:** Automated verification using capacity databases populated from CAD-to-capacity pipelines ensures consistency and eliminates manual lookup errors.

The framework represents a scalable pathway toward performance-based, regulation-aligned design automation, contributing to improved seismic resilience and regulatory confidence across the construction sector. The data and workflow for this research are available at: <https://github.com/Constratum>.

9 REFERENCES

- Dhakal RP (2010). “Damage to non-structural components and contents in 2010 darfield earthquake”. Bulletin of the New Zealand Society for Earthquake Engineering, 43(4): 404–411.
- Fortune S (1987). “A sweepline algorithm for voronoi diagrams”. Algorithmica, 2(1–4): 153–174. <https://doi.org/10.1007/BF01840357>
- Gillies S (2007). Shapely: manipulation and analysis of geometric objects. <https://shapely.readthedocs.io>
- Google (2026). Firebase: app development platform. <https://firebase.google.com>
- Harris C and Stephens M (1988). “A combined corner and edge detector”. Proceedings of the 4th Alvey Vision Conference, 147–151.
- King R (2019). Sectionproperties: a python package for the analysis of arbitrary cross-sections. <https://sectionproperties.readthedocs.io>
- Lloyd SP (1982). “Least squares quantization in pcm”. IEEE Transactions on Information Theory, 28(2): 129–137. <https://doi.org/10.1109/TIT.1982.1056489>
- Miranda E, Mosqueda G, Retamales R and Pekcan G (2012). “Performance of nonstructural components during the 27 february 2010 chile earthquake”. Earthquake Spectra, 28(S1): S453–S471. <https://doi.org/10.1193/1.4000032>
- Standards New Zealand. (2004). “NZS 1170.5: Structural design actions – Part 5: Earthquake actions – New Zealand.” Standards New Zealand, Wellington, 76 pp. <https://www.standards.govt.nz/sponsored-standards/building-standards/NZS1170-5>
- Wock JC (2019). Pynite: a 3d structural analysis library for python. <https://github.com/JWock82/Pynite>