Industrial Scale NLRH Analysis Using OpenSees and Comparison with Perform3D

Craig B. Goings, S.E., Principal
Ayush Singhania, E.I.T., Associate Project Consultant
Benjamin Weaver, S.E. Senior Consulting Engineer
Simpson Gumpertz & Heger
San Francisco, CA
Pearl Ranchal, P.E., Designer
Degenkolb Engineers
San Francisco, CA

Abstract

OpenSees is an open source finite element software framework that has been broadly used in academia by researchers in the earthquake engineering community for simulating seismic response of structures. However, OpenSees has not gained popularity in the professional community partly due to its lack of a graphical user interface (GUI) and a popular scripting language. In 2017, OpenSees interpreter interface was modified to provide multi-interpreter capabilities and a Python interpreter, OpenSeesPy, was introduced for the first time.

To provide professionals with an option of using OpenSees instead of other analytical tools with nonlinear capabilities, existing industry standard software tools are adapted to serve as the GUI front-end for the OpenSees analytical engine. A framework is developed to automate the generation of a model in OpenSees utilizing ETABS as the GUI. An analytical case study comparing results from OpenSees and Perform3D is presented. Workflow, modeling recommendations and limitations of this modeling process are also presented.

Introduction

Engineers use commercial software such as Perform3D, SAP2000 and ETABS to run nonlinear analyses. These commercial software programs provide structured, user-friendly modeling environments and favor stable modeling elements and solution strategies but do not always support novel nonlinear elements and solution strategies developed in academia and implemented in OpenSees. OpenSees, on the other hand, has mostly been used as an academic research tool and has not gained popularity among professionals due to its lack of a GUI and its unwieldy, unstructured modeling environment.

These issues can be resolved by developing a modeling environment that employs the user-friendly, GUI-oriented modeling environment of ETABS or SAP2000 as the front-end to an OpenSees-powered computational backend. Such an environment would allow engineers to model structural geometry, static loads, elastic components, and certain nonlinear components within ETABS or SAP2000, while organizing information that cannot be captured within ETABS or SAP2000 in a companion spreadsheet. Once this pre-processing phase is complete, the modeling data in ETABS/SAP2000 and spreadsheet components can be parsed and combined using a Python-based environment and fed into OpenSees for computational processing using the OpenSeesPy library.

This paper describes using OpenSees via OpenSeesPy, a Python interpreter for OpenSees to analyze a five-story steel moment frame structure as a case study. We used the ETABS graphical user interface along with OpenSeesPy to automate the generation and analysis of the OpenSees model. We then compare results of modal, nonlinear pushover, linear and nonlinear response history analysis performed in OpenSees using OpenSeesPy, with results from similar models in CSI Perform3D and CSI ETABS.

Why use OpenSees for NLRHA?

The OpenSees framework is open source. This provides engineers with source-code level control of a sophisticated nonlinear finite element analysis tool without incurring any software licensing costs. OpenSees also possesses several analytical features and structural elements based on state-of-the-art research that can, in certain cases, capture structural behavior better and faster than commercial software. We highlight some examples below.

In some cases, OpenSees can run analysis faster by better leveraging the capabilities of modern computing hardware. OpenSees features partitioning of analysis case between multiple CPU cores so that an analysis case can be run in parallel on several processors. Additionally, OpenSees also supports GPU-powered solvers (such as CuSP and CULASparse) that have been reported to deliver up to 15-fold reduction in computation time relative to CPU-based solvers for large nonlinear models (Tian et al., 2015). In contrast, most commercial software is constrained to run one analysis case per CPU core in a serial manner, utilizing all cores. The advantage this presents to OpenSees is that it can utilize multiple processors when needed computationally during an analysis case, rather than a single processor. Read and write speed often become a bottle neck during parallel processing situations which can be alleviated in OpenSees by using commonly available modern storage technologies such as non-volatile memory solid state drives.

OpenSees enables engineers to use the latest elements, solution algorithms and solvers, and create their own if desired. Analysis-oriented academic research is often accompanied by implementation of modules in OpenSees. Although such academic modules may require testing, benchmarking, and modification for adaption in profession analysis, the benefit of such one-time investments can often lead to innovative, more realistic and faster solutions. In addition, commercial software packages are often opaque regarding the details of the solution algorithms and solver techniques they employ. However, OpenSees provides engineers with extensive, transparent control over the details of their analysis. For simple projects, this extra functionality and transparency may be of minimal benefit. However, in certain specific cases, such additional control may be very beneficial. Moreover, it is always prudent for engineers to be aware of the strengths and weaknesses of the nonlinear elements, solution strategies and solvers they are using in their analysis, especially when modeling conditions close to collapse.

Concentrated plasticity flexural hinges using the modified Ibarra-Medina-Krawinkler (Lignos and Krawinkler, 2011) model are a good example of the types of elements that are based on relatively recent research and have OpenSees implementations. This hinge element is available in OpenSees with three hysteretic response models: bilinear, peak-oriented and pinched. The hysteretic characteristics of these hinges were calibrated with respect to more than 350 experiments of steel beams with different beam-to-column connection types. These relationships were adopted by PEER/ATC-72. Given a certain backbone, these experimentally calibrated hysteretic models can better represent the hysteretic dissipation of energy in a steel framed building compared to the limited hysteretic models available within commercial software that often require calibration to experimental datasets.

OpenSees allows engineers to perform modal analysis at any state of the model, i.e. before, after or in the middle of a ground motion. This can help alleviate uncertainty regarding the upper bound of period range of interest associated with the lengthening of the fundamental period of a softening nonlinear model. Such period ranges of interest are often required for: ground motion selection and scaling and anchoring Rayleigh damping models.

Finally, the use of initial stiffness proportional Rayleigh damping with concentrated plasticity elements has been shown to cause large spurious damping forces (Chopra, 2016) within structural models during nonlinear response history analyses. Perform3D is limited to using initial stiffness proportional Rayleigh damping, while OpenSees also allows the use of tangent stiffness proportional Rayleigh damping which has been shown to prevent the occurrence of large spurious damping forces during nonlinear response history analyses. In the case study accompanying this paper, we observed that Perform3D produced satisfactory results with initial stiffness-proportional Rayleigh damping because its uses “compound” elements. Such elements mask the degrees of freedom (DOFs) associated with zero-length, rigid beam hinges and prevent these DOFs from entering the global stiffness matrix directly. This solution is akin to the use of penalty elements (Chopra, 2016) that have stiffness, but no damping associated with them. This solution works well in the context of frame and shell elements where compound assignments are required but may not be applicable to support/foundation/soil spring elements where DOF-masking using “compounds” is not applicable. Using modal damping, which is available in both OpenSees and Perform3D, also prevents spurious damping forces. While damping remains a topic of debate among academic researchers, it’s useful for practicing engineers to have software that is not limited in functionality so that engineers can make their own decision and select damping models based on varying engineering considerations of their analysis models, which OpenSees supports.

Computational Differences Between OpenSees and Perform3D

Computationally, nonlinear analysis is a two-step problem. 1) setting up the problem, i.e. constructing the mathematical model, selecting and scaling ground motions, etc. and 2) solving the nonlinear problem. Perform3D, by using its reliable event-to-event solution strategy, insulates the engineer from the details of step two to quite a significant degree. However, OpenSees is designed to give engineers total control over both analysis steps. Given the objectives and nature of the nonlinear analysis, the extra functionality provided by OpenSees may or may not be desirable. For a relatively simple structure and loading, Perform3D can be adequate and efficient.

Perform3D's event-to-event strategy is stable, i.e. rarely has convergence issues in typical models. However, for large nonlinear models with features such as: multiple sources of nonlinearity, fiber hinges, or P-Delta dominated behavior, an event-to-event strategy can be slow compared to iterative strategies as it may have to contend with a large number of "events" – instances of appreciable change in element stiffness such as yielding, unloading, etc. (Prakash, 1992; Mondkar and Powell, 1978; CSI, 2018b). Occurrence of every such "event" triggers computationally expensive revisions of the global stiffness matrix. When several events occur within the same timestep, the software is forced to solve for the sequential occurrence of these events by dividing a timestep into several substeps such that only one event occurs per substep, hence this strategy is called event-to-event. This division of timesteps into substeps can also be very time intensive because of repeated state determinations and stiffness matrix assembly steps (Mondkar and Powell, 1978). Specifying a large (>2% in Perform3D) overshoot factor (OSF) can help reduce analysis runtime by allowing an "event" to comprise stiffness revision of multiple elements (CSI, 2018b). While specifying large OSFs has little effect on peak responses, it has been observed to compromise quality of results associated with post-peak behavior, such as: residual drift, residual hinge rotations and post-peak portions of pushover curves. This effect can occur even when good analytical energy balance (>95%) is achieved with large OSFs. Also, Perform3D's event-to-event strategy does not iterate to equilibrium between external and internal forces at individual timesteps but instead adds the equilibrium error of a timestep to the external load vector of the subsequent timestep (CSI, 2018b). As a result, larger OSFs can also produce large equilibrium errors that can translate to larger energy unbalances and potentially compromised post-peak responses.

In contrast, OpenSees relies on Newton-type iterative solution algorithms such as: Newton-Raphson, Newton with Line Search, Krylov-Newton (Scott and Fenves, 2010), etc., for nonlinear response history analysis. These solution strategies require parameter tuning which often need to be thoughtfully determined before such strategies can be used effectively. As such, these strategies are not as reliable as Perform3D's event-to-event strategy and engineers are advised to evaluate and select strategies most appropriate for their analysis. However, their reliability can be improved by specifying a "suite of solution strategies" such that the program falls back on a different set of solution parameters or an entirely different solution strategy if one fails to converge. These iterative strategies are also not constrained to executing "events" sequentially and seek equilibrium at every timestep by iterating between converged states of the mathematical model. Furthermore, given that OpenSees is open source, a hybrid strategy that combines the best features of event-to-event and iterative strategies can be implemented. OpenSees also provides engineers with direct control over analytical parameters such as numbering technique, solver selection, iterative algorithm, integration method and convergence criterion. Although ETABS and SAP2000 provides some of these choices, the options are not as extensive as OpenSees. Perform3D provides no control over numbering scheme, solver selection or integration method.

In the past, when computing resources were limited in both computing power and availability, event-to-event strategies served as resource-efficient alternatives to Newton-type iterative strategies. With the current widespread availability of inexpensive and powerful computing resources, Newton-type iterative strategies can be used to perform nonlinear analysis faster and more efficiently than event-to-event strategies for highly nonlinear structures. These are also some of the reasons why ETABS and SAP2000 prefer iterative strategies to event-to-event strategies even though both strategies are available within these software packages.

Convergence in this context can be determined through one of several criteria available in OpenSees. For example: an iterative strategy may be said to have converged at a timestep when the norm of the displacement increment vector or the load unbalance vector or the energy increment shrinks to a pre-determined tolerance. Several vector norms may also be used to evaluate these convergence tests such as: 2-norm (square root of sum of squares), 1-norm (sum of absolute values) or max-norm (absolute maximum). For highly nonlinear structural models, many of the iterative strategies available in OpenSees will require fewer revisions of the stiffness matrix compared to the event-to-event strategy and so the analysis will run faster.

Case Study Building Description

This case study presents analyses of the steel moment frame building presented in Chapter 8 of FEMA P-2006 (FEMA, 2018). This building was constructed in 1985 using the 1982 Uniform Building Code, with a footprint of 217ft x 217ft (Figure 1). The building has five stories above the seismic base, with the first story height of 16ft and typical story height of 14'-3" (Figure 2). The base of the structure is supported on spread footings with grade beams connecting the footings around the perimeter. The seismic force resisting system comprises perimeter steel moment frames (SMF) with rigid diaphragms – Type S1 per ASCE 41 (ASCE, 2017). The plan shown in Figure 1 highlights the SMF layout in Red. The moment frame has Pre-Northridge welded unreinforced flange (WUF) connections. A typical perimeter frame elevation is shown in Figure 2. Columns are identified as ASTM A572 GR50 steel, beams are identified as ASTM A36 steel, and grade beams with concrete strength of 3 ksi.

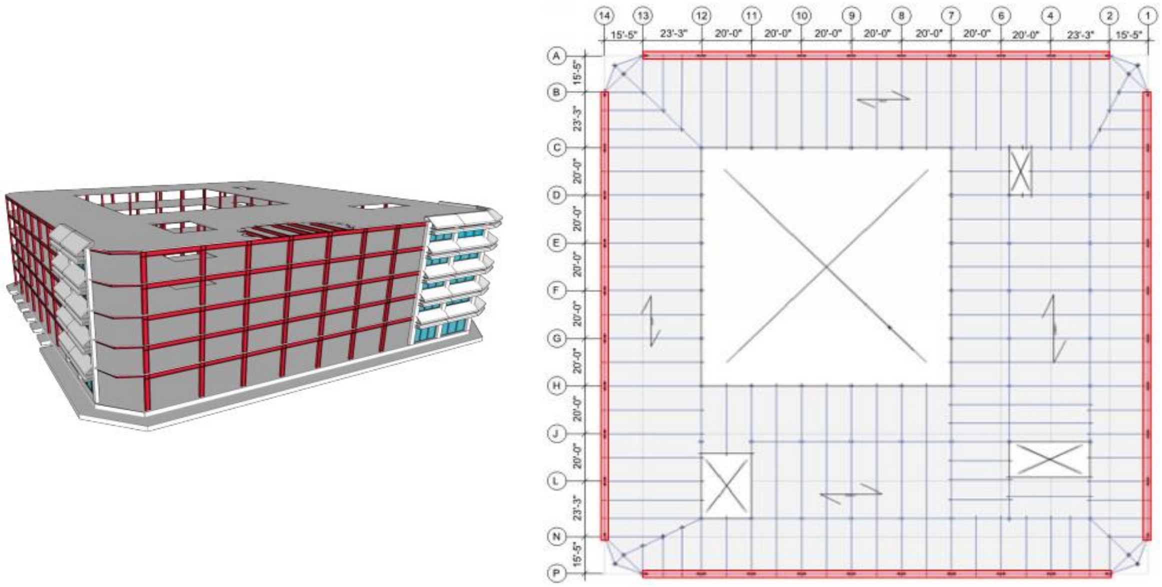

Figure 1: Three-Dimensional View of the Steel Moment Frame Building and Typical Floor Plan

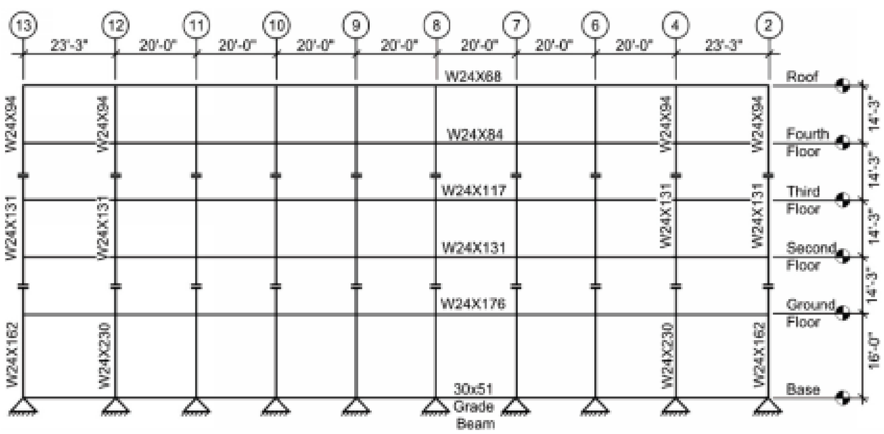

Figure 2: Typ. Moment Frame Elevation (Gridline P)

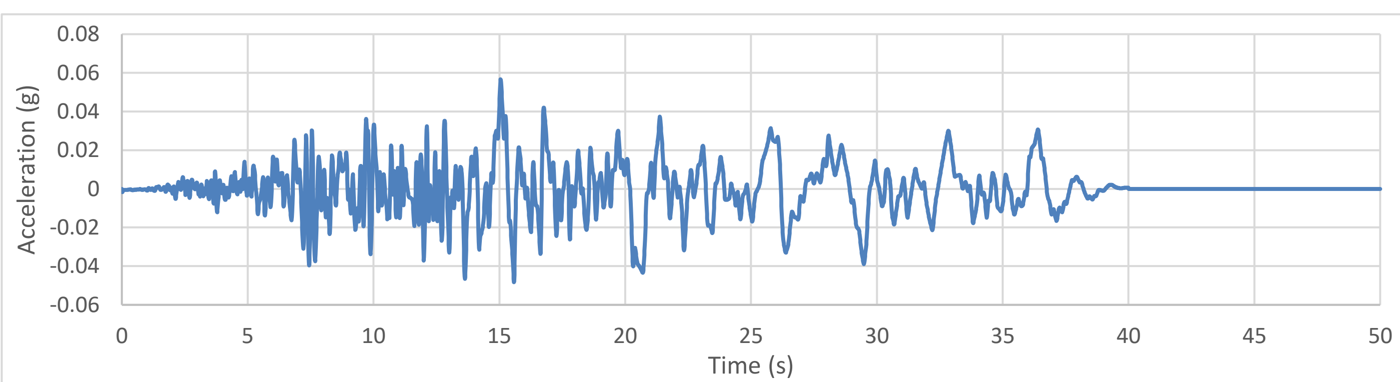

Figure 3: Modified Borrego Mtn Ground Motion (Record Sequence #36)

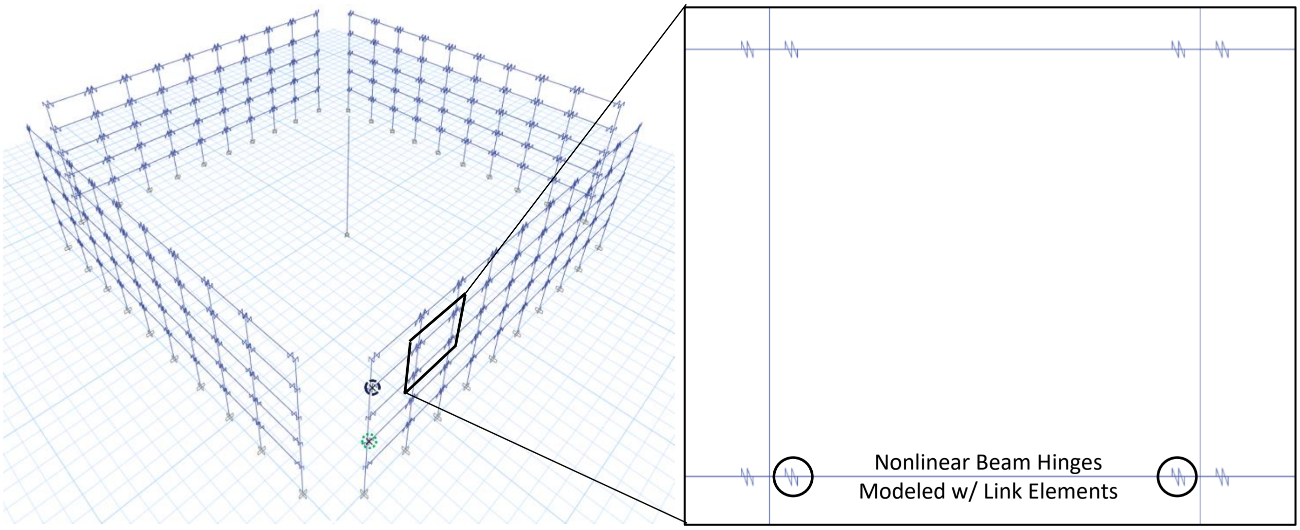

Figure 4: Nonlinear ETABS Model with Beam Hinges Modeled using Link Elements; Hinge #20275 (in black) dashed; Hinge #20283 (in green) dotted

Ground Motion Input

This paper presents analysis using the Modified Borrego Mtn ground motion – RSN#36 (OpenSees Wiki, 2010) with scale factor 1, 3.0, and 5.0. The time-history for scale factor 1 is shown in Figure 3.

Description of Analysis Models

We developed a linear elastic ETABS model based on the pre-Northridge steel moment frame building discussed above. The model comprises of frame elements modeled with appropriate sections and expected material properties per tables 9-1, and 9-3 of ASCE 41. We modeled the diaphragms as rigid at all levels. We also modeled a set of columns at the plan center (Figure 4) to represent the gravity system and mass. To simplify the analysis, we omitted modeling the gravity load in the current study, but it can be incorporated within the modeling workflow presented here with relative ease.

We converted the linear ETABS model into an OpenSees model with one click using a custom Python-based ETABS-to-OpenSees (E2O) converter tool that we developed as part of this study. This converter uses the CSI OAPI to retrieve structural data (model geometry, connectivity, material and section properties, etc.) from the ETABS model and organizes it within Python data structures (Moore et al., 2019). We use E2O's routines in conjunction with the Python data structures to autogenerate an OpenSees model via OpenSeesPy. The Python source code for this converter tool is available on the project GitHub page on an open-source basis.

We manually converted the linear ETABS model into nonlinear ETABS (Figure 4) and Perform3D (Figure 5) models. We then used the E2O converter to autogenerate the nonlinear OpenSees model (Figure 6) using the nonlinear ETABS model. All nonlinear models captured nonlinearity at beam ends using concentrated plasticity hinge elements, with columns and panel zones modeled linear elastic. We modeled beam hinges using link elements in ETABS, semi-rigid moment connection hinges in Perform3D and 'bilin' uniaxial material hinges in OpenSees using the Modified Ibarra-Medina-Krawinkler (IMK) deterioration model with bilinear hysteretic response (Lignos and Krawinkler, 2011). Since the beam hinges are non-rigid and connected in series with elastic beam segments, we adjusted their elastic stiffness to approximately three orders of magnitude greater than the elastic stiffness of the beam so that the global stiffness of the assembly is approximately equal to the stiffness of the actual beam.

While nonlinear response history analysis can be run in ETABS, our main use of the nonlinear ETABS model is to have it serve as a database of structural information for the OpenSees model, and to utilize the ETABS GUI interface.

Analysis Model Conversion to OpenSees

As discussed earlier, the lack of a fully functional graphical user interface (GUI) continues to remain a barrier for the industry-wide adoption of OpenSees as a viable structural analysis software. One potential solution to this problem involves programmatically tying the OpenSees analytical engine into the GUI of commercially available structural analysis software that an engineer may already be familiar

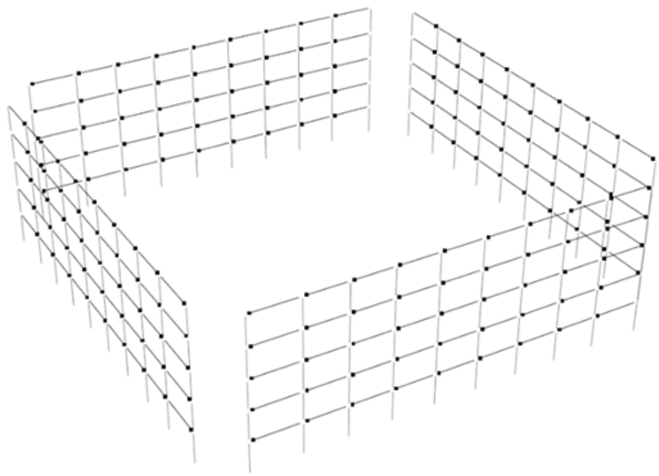

Figure 5: Perform3D Model

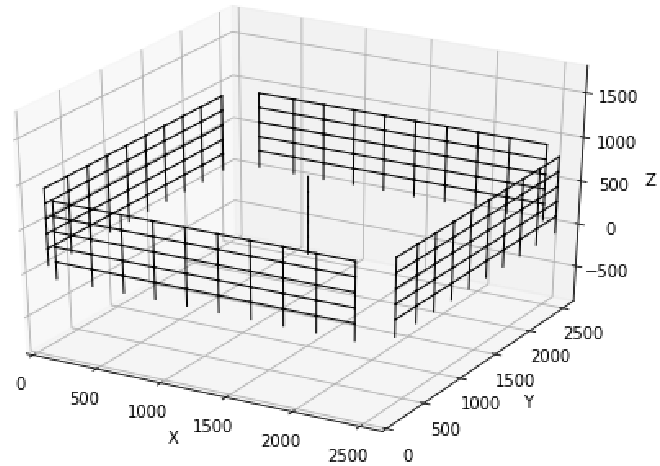

Figure 6: OpenSees Model

with. Such a combination will allow engineers to pair the convenience of graphical modeling workflows of commercial software with OpenSees' analytical capabilities.

Most commercial structural analysis software allows programmatic control and data access through an application programming interface (API). Software that do not feature an API will often store structural model data in text or binary files that may be amenable to programmatic data extraction (eg: CSI Perform3D). Once structural data is extracted to and organized within a data analysis environment, such as Python or MATLAB, it can be used to automatically generate an OpenSees model. In theory, this framework can be utilized to automate the generation of Tcl scripts that can run directly on the OpenSees executable. However, in this study we emphasize a uniform Python environment aimed at minimizing the number of distinct software components. By using OpenSeesPy, this framework can be implemented practically and efficiently using a common Python platform. We adopted this pragmatic approach in this study.

As discussed previously, we developed a nonlinear mathematical model of the 5-story, steel moment frame building in CSI ETABS v18.1.1. To simplify the analytical part of this study, we only modeled nonlinearity in the beams. We assumed columns and panel zones remained elastic. We used ETABS link elements to model the concentrated plasticity rotational springs. ETABS also permits direct assignment of hinges to frame elements. However, such an approach would not identify the rotational springs and beams as distinctly connected structural elements. Modeling the rotational springs as link elements distinct from the beam elements explicitly captures the location and connectivity of the rotational springs relative to the adjacent beams, nodes and columns and simplifies the automatic generation of the OpenSees model using OpenSeesPy. Such an approach also provides flexibility to the engineer by permitting placement of hinges at any location and not just beam ends.

We used the *cDatabaseTables* interface, available within the CSI API for ETABS, to programmatically export the ETABS in/output database tables to a Python environment. These tables provide direct access to ETABS model data such as geometry, connectivity, material properties, assembled joint masses and gravity loading. Not all nonlinear properties (eg: hinge backbones) can always be retrieved from ETABS. Such information can be organized and stored in a spreadsheet. Data extracted from the ETABS model and backbone properties parsed from a spreadsheet are organized within a Python environment using instances of the data structure class *DataFrame* defined by the open-source *pandas* library - a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language (McKinney & others, 2010). DataFrames have several attributes and functionalities that make them an

```
IPython 7.8.0 -- An enhanced Interactive Python.

In [19]: print(joints_df.iloc[0:10].to_markdown(index=False))
|   UniqueName |    X |   Y |   Z |
|-------------:|-----:|----:|----:|
|           61 |  186 |   0 | 876 |
|           67 |  465 |   0 | 876 |
|           73 |  705 |   0 | 876 |
|           79 |  945 |   0 | 876 |
|           85 | 1185 |   0 | 876 |
|           91 | 1425 |   0 | 876 |
|           97 | 1665 |   0 | 876 |
|          103 | 1905 |   0 | 876 |
|          109 | 2145 |   0 | 876 |
|          115 | 2424 |   0 | 876 |
```

Figure 7: joints_df DataFrame (shown truncated)

excellent programmatic alternative to spreadsheets for engineering computations and data analyses.

We developed the E2O library of functions in Python and used them to parse the ETABS tabular data into DataFrames. For example: like a spreadsheet, the *joints_df* DataFrame is comprised of rows and columns. Each DataFrame row represents a unique node and the four DataFrame columns represent the node numbers and the nodal cartesian coordinates – x, y, z, respectively. Since the DataFrame is generated from the ETABS model, the nodal IDs and coordinates correspond exactly to the "UniqueName" ETABS attribute and the cartesian coordinates of nodes, as defined in the ETABS model. We used the *pandas.DataFrame.apply()* method to loop over each row of the *joints_df* DataFrame (Figure 7) and define nodes within the OpenSees model using the *openseespy.opensees.node()* method. We also defined and added elastic frame elements and zero length rotational springs to the OpenSees model using similar implementations within the developed library. This custom E2O Python library is available on the project GitHub page on an open-source basis.

Comparison of Results from OpenSees with Perform3D

As previously discussed, we used the ETABS GUI to model the structure manually, then used the E2O library to autogenerate the OpenSees version of the ETABS model, and we run this OpenSees model using OpenSeesPy from within a Python environment. We used OpenSeesPy to perform modal analyses, linear static analysis, nonlinear static analysis (pushover) linear response-history analysis (LRHA) and nonlinear response history analyses (NLRHA). This paper presents the result from modal, pushover, LRHA and NLRHA. We present the modal analysis results from all three software – OpenSees, ETABS, and Perform3D. We compare the results for pushover, linear and nonlinear response history analyses from two software – OpenSees and Perform3D.

Modal Response matched well between all three analyses software. Table 1 presents the results for the first four governing modes required to reach over 90% modal mass participation.

Linear RHA is performed in OpenSees and Perform3D by scaling the ground motion to a small value such that the structure remains elastic throughout ground motion duration. For this purpose, we used a ground motion scale factor (GMSF) of 1.0 with the Modified Borrego Mtn ground motion. Figure 9 shows the roof center of mass (COM) node displacement comparison between the OpenSees and Perform3D analysis results. The base shear comparison is shown in the Figure 10, and story drift comparison is shown in Figure 11. The results from OpenSees matches well with the results from Perform3D.

Table 1: Mode Shape Period Comparison

Mode	OpenSees	Perform3D	ETABS
	(sec.)	(sec.)	(sec.)
1	1.6378	1.6372	1.6340
2	1.6359	1.6353	1.6321
3	0.5660	0.5671	0.5649
4	0.5653	0.5664	0.5642

Good agreement between results of LRHA performed in OpenSees and Perform3D confirms that the autogenerated OpenSees model has been constructed with correctly assigned linear properties and structural geometry. Prior to running NLRHA, we must also confirm a good match of nonlinear properties between the Perform3D and OpenSees models. To qualify the beam hinge backbones and confirm their correct assignment in OpenSees, we performed a nonlinear pushover analysis. Hinge flexure vs rotation plots obtained from pushover analyses, run to large story drifts, should correspond to assigned backbones. Figure 8 shows good agreement between the backbones obtained from pushover analyses performed in OpenSees and Perform3D. This confirms that all geometry, linear and nonlinear properties match well between the Perform3D and OpenSees models and that NLRHA may be performed and expected to produce similar results between the two software. For very intense ground motions, it may be desirable to confirm agreement between the hysteretic models

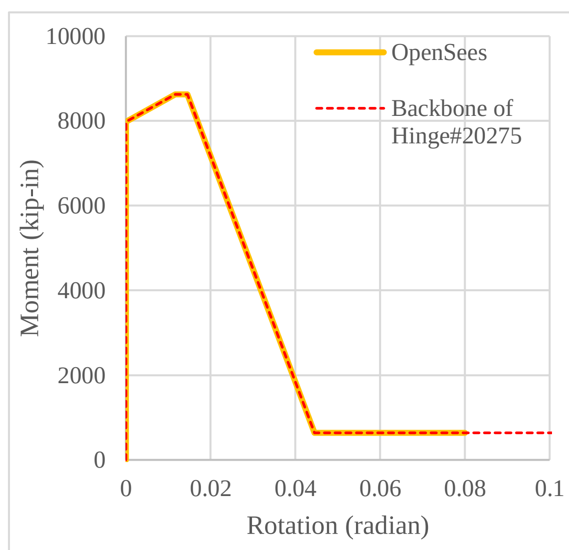

Figure 8: Pushover result for Hinge#20275

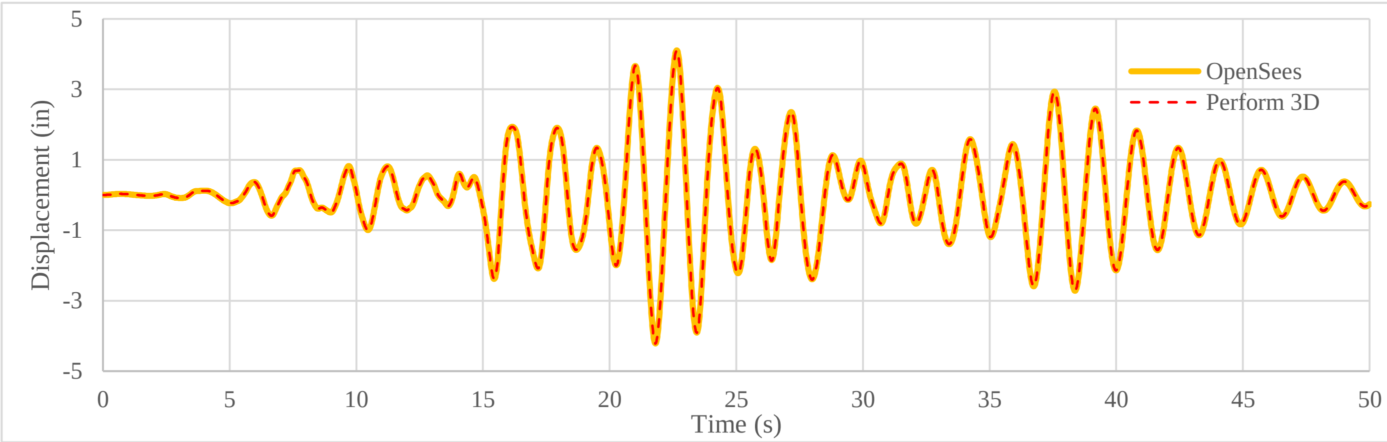

Figure 9: Roof COM node displacement (GMSF – 1.0)

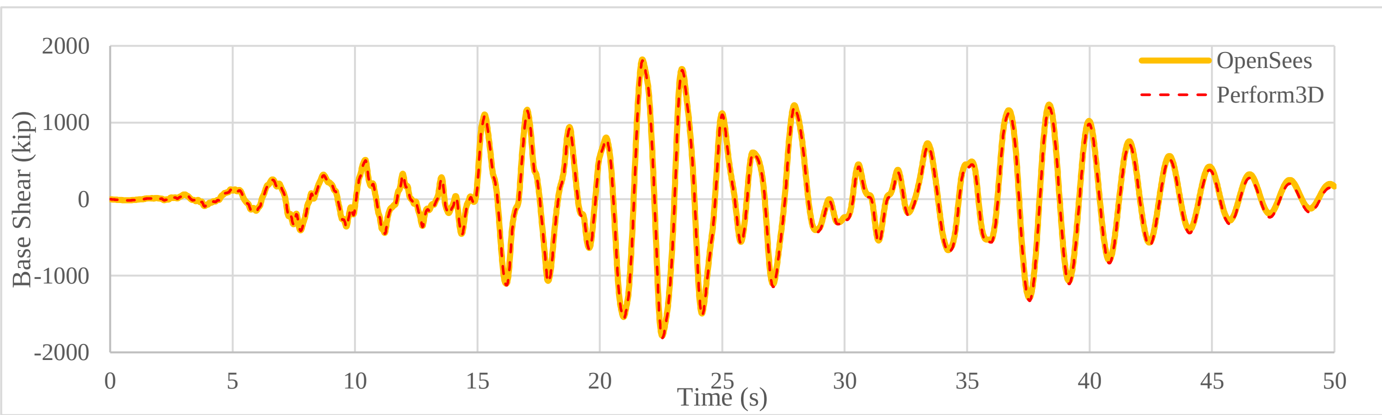

Figure 10: Base Shear (GMSF – 1.0)

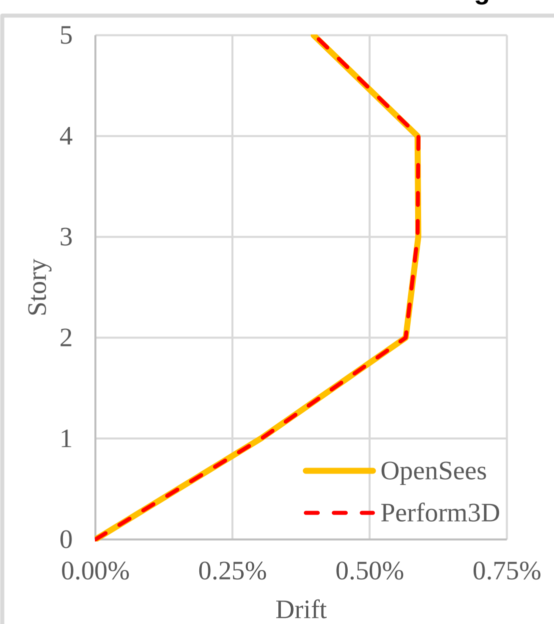

Figure 11: Story Drift (GMSF – 1.0)

used in the two software packages using cyclic pushover analyses. We did not perform this step in the current study.

The NLRHA can be performed with different damping models. We developed the E2O Python package to use Rayleigh damping and we are developing modal damping. We performed this case study solely using Rayleigh damping with both mass and stiffness proportional terms. OpenSees has the option of using either the initial stiffness or tangent stiffness matrix for the stiffness proportional term. In contrast, Perform3D always uses the initial stiffness matrix, as discussed previously. Although we did not run NLRHA in ETABS for this study, it is worth noting that ETABS' link elements also provide the option of using initial or tangent stiffness proportional Rayleigh damping.

As shown in Chopra A., 2016, initial stiffness proportional term with initial stiffness matrix in Rayleigh damping can lead to erroneous results caused by the spurious damping forces when used with concentrated plasticity models, as is the case

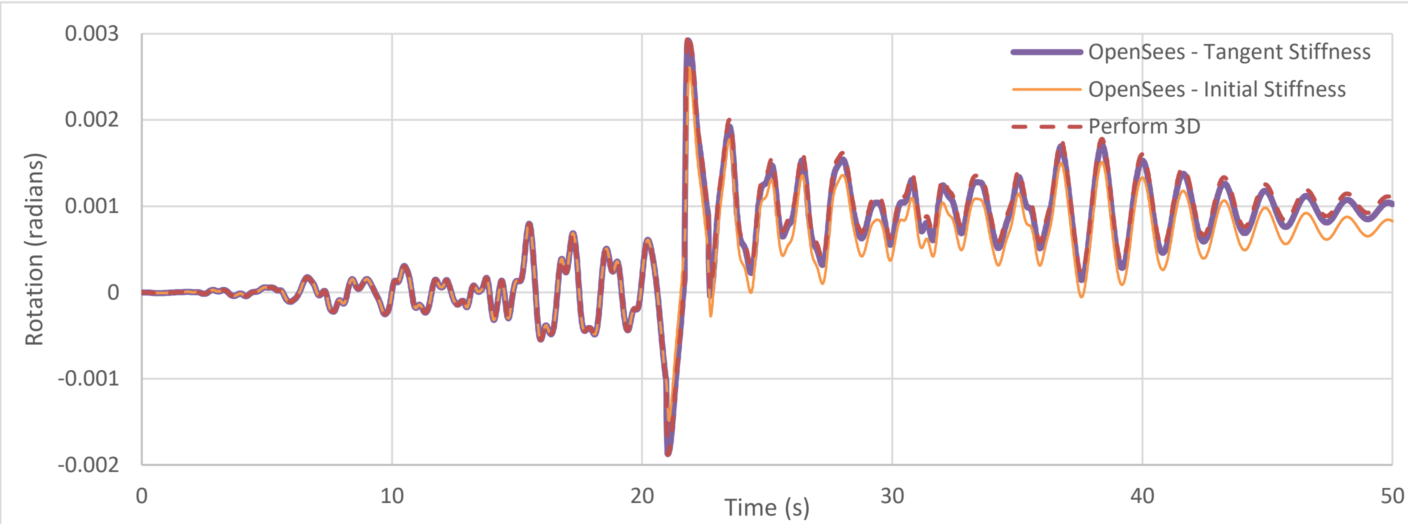

Figure 12: Plastic Rotation at Hinge #20275 (GMSF – 3.0)

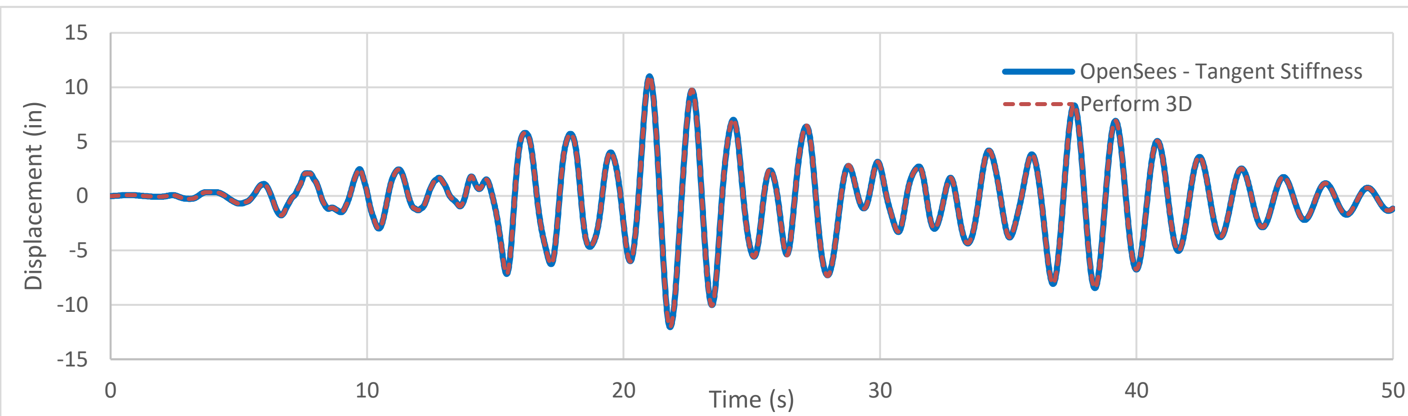

Figure 13: Roof COM Node Displacement (GMSF – 3.0)

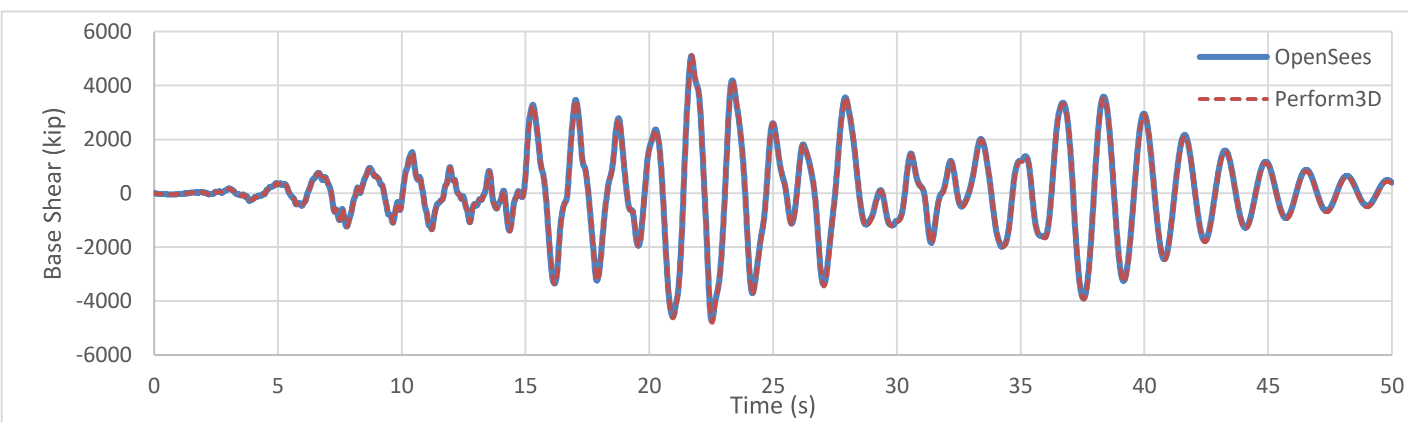

Figure 14: Base Shear (GMSF – 3.0)

in the current study. To reduce the impact of this spurious damping forces arising from the Rayleigh damping, several researchers have proposed to modify the stiffness proportional term by replacing the initial stiffness matrix by the tangent stiffness matrix. Another modeling technique is presented in Chopra A., 2016 called the "Penalty Elements" which is commonly used at contact or sliding surfaces. Researchers have demonstrated that penalty elements can also be used to model nonlinearity in beam and column elements. Penalty elements are used to define constraint, behaviors of given degree of freedom, hence stiffness contribution of these penalty elements are not included in the damping term. This can be achieved in the OpenSees model by defining no Rayleigh damping in the zero-length element. In Perform3D elimination of spurious damping forces is achieved by defining a rotation hinge with infinite initial stiffness, moment hinge rotation type. The semi-rigid moment connection hinge in Perform3D acts like a pseudo-penalty element, which is achieved by removing the rotational degree of freedom from the global stiffness matrix by enveloping the rigid hinges within a beam compound element. For other spring type nonlinear elements in Perform3D, the Rayleigh damping can be turned off by specifying beta-k to zero. Figure 12 shows the plastic rotation of hinge#20275 (as marked in the isometric view in Figure 4). This figure compares the result between OpenSees model using Rayleigh damping with initial stiffness as well as tangential stiffness and Perform3D using the pseudo-penalty element – "compound". The plastic rotation of the hinge in OpenSees with tangent stiffness matches well with that of Perform3D. In this paper, for all following results, we compare OpenSees with Rayleigh damping using tangent stiffness with Perform3D using pseudo-penalty elements.

For the NLRHA we used a GMSF of 3.0 with the Modified Borrego Mtn ground motion. Figure 13 shows the roof center of mass node displacement comparison of OpenSees results and perform3D results. The base shear comparison is shown in the Figure 14, and the story drift comparison is shown in Figure 15.

Good agreement is found between results of NLRHA performed in OpenSees and Perform3D. Both OpenSees and Perform3D took about 10 minutes to complete the analysis. The results presented above are from GMSF of 3.0. During this analysis, strength loss does not initiate in any of the beam

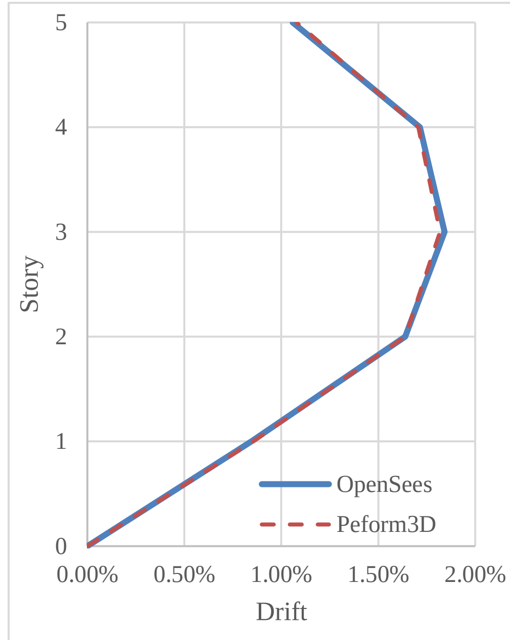

Figure 15: Story Drift (GMSF – 3.0)

hinges, i.e. maximum hinge rotations are either within yield or lie in the strain-hardening portion of the backbone. To compare the performance of the strength loss portion of the backbone curve, a higher ground motion input is required, hence we used a GMSF of 5.0.

The hysteretic model used in OpenSees is different from that of Perform3D. There are two major differences when considering no cyclic degradation, as is the case here. Perform3D uses a bilinear hysteretic model with strength limit as shown in Figure 3 of Ibarra Medina and Krawinkler (IMK) (Ibarra, et. al., 2005). Whereas OpenSees uses the modified Ibarra and Krawinkler (mod.IK) deterioration model with no such strength limit as shown in Figure 1(b) presented in Lignos and Krawinkler 2010 (Figure 16). This is shown by the green doted-line in Figure 17 where Perform3D limits the capacity in post peak cycles to the green doted-line whereas the mod.IMK model goes back to full capacity. The second major

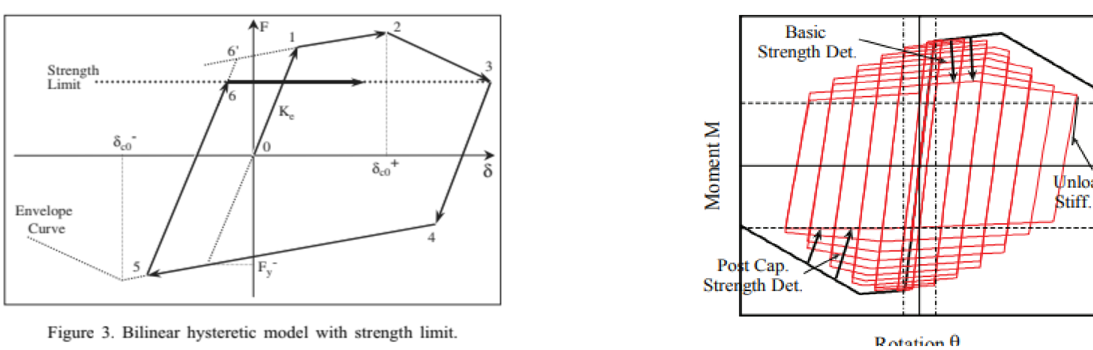

Figure 16: Perform3D (Figure 3) vs Mod.IK (Figure 1(b)) Hysteretic Model

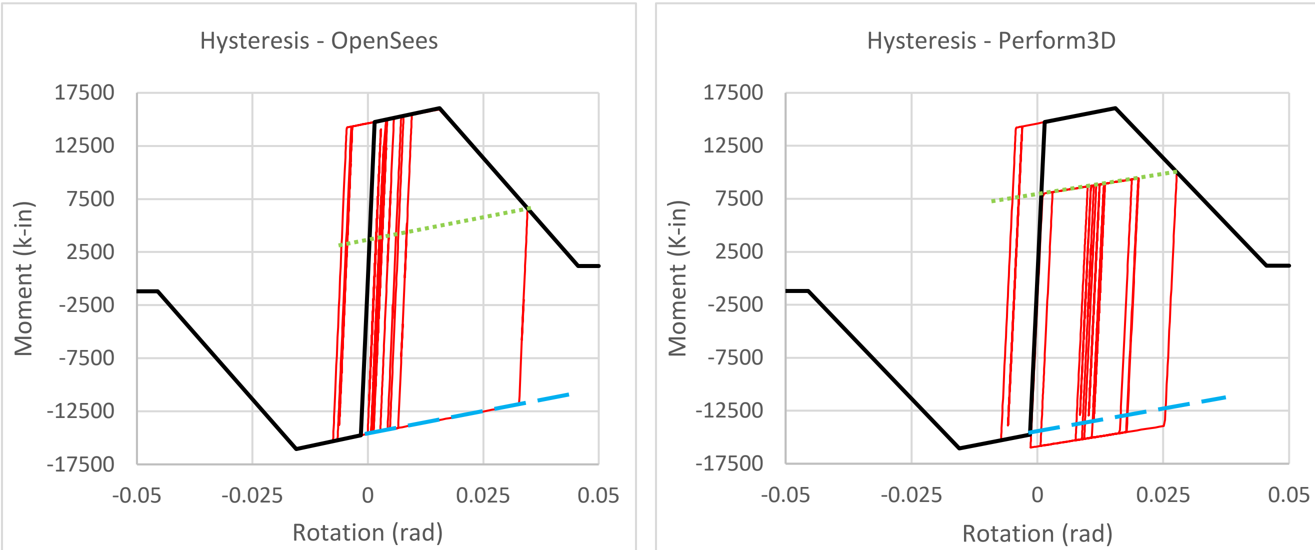

Figure 17: Hysteresis of Nonlinear Hinge #20283 (GMSF - 5.0)

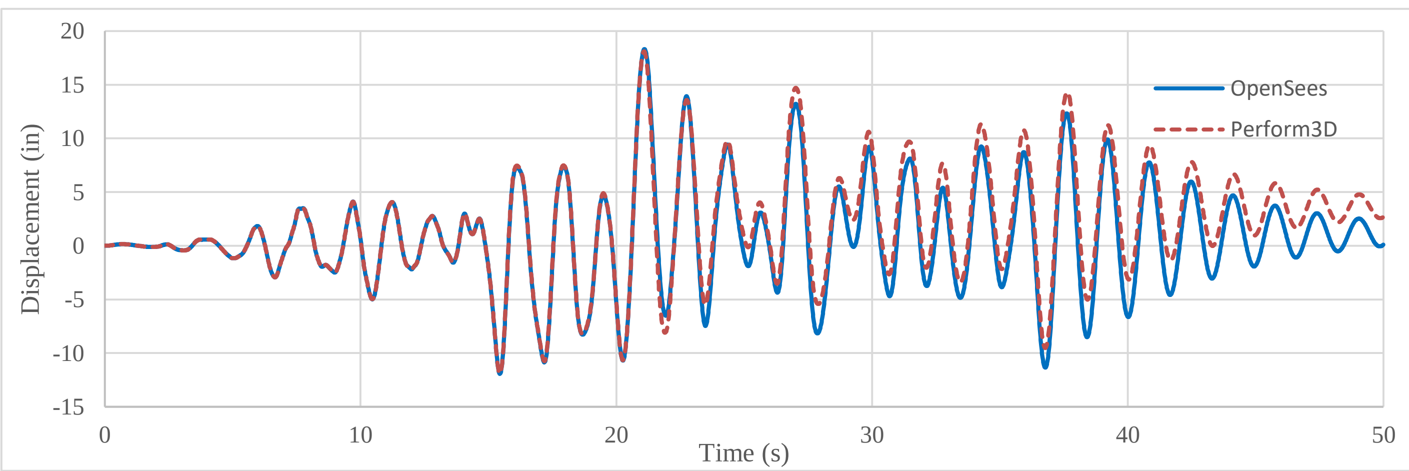

Figure 18: Roof COM Node Displacement (GMSF – 5.0)

difference is that OpenSees follows the Bauschinger effect and limits the capacity by extrapolating the backbone as shown with blue dashed line in Figure 17. Perform3D does not have limitation and goes beyond the blue dashed line.

Therefore with no cyclic degradation in the mod.IK model the behavior is the same as the Perform3D hysteretic model as long as hinge rotation is within strain-hardening region. However, once the hinge rotation goes beyond the initiation of strength loss, the behavior of mod.IK is different than Perform3D. There is no perfect replacement in OpenSees for the hysteretic model used in Perform3D. Hence, we could not obtain a good match. Figure 17 shows the hysteresis of the hinge #20283 under GMSF of 5.0 on the Modified Borrego Mtn ground motion in OpenSees and Perform3D. Figure 18 shows the roof center of mass node displacement comparison of OpenSees results and perform3D results. The base shear comparison is shown in the Figure 19, and the story drift comparison is shown in Figure 20. Although the hysteretic model were different in the two software, we found relatively small effect on global behavior. We note that the mod.IK hysteretic model was developed to represent ductile wide flanged steel beams and not brittle connections. Hence, in a real-world scenario, it is advised to calibrate the hysteresis with the brittle connection test data.

GitHub Repository

We made the E2O library used in this case available to public. It can be found at the link below along with the ETABS and Perform3D models: https://github.com/OpenSeesPro/E2O-SEAOC2020 .

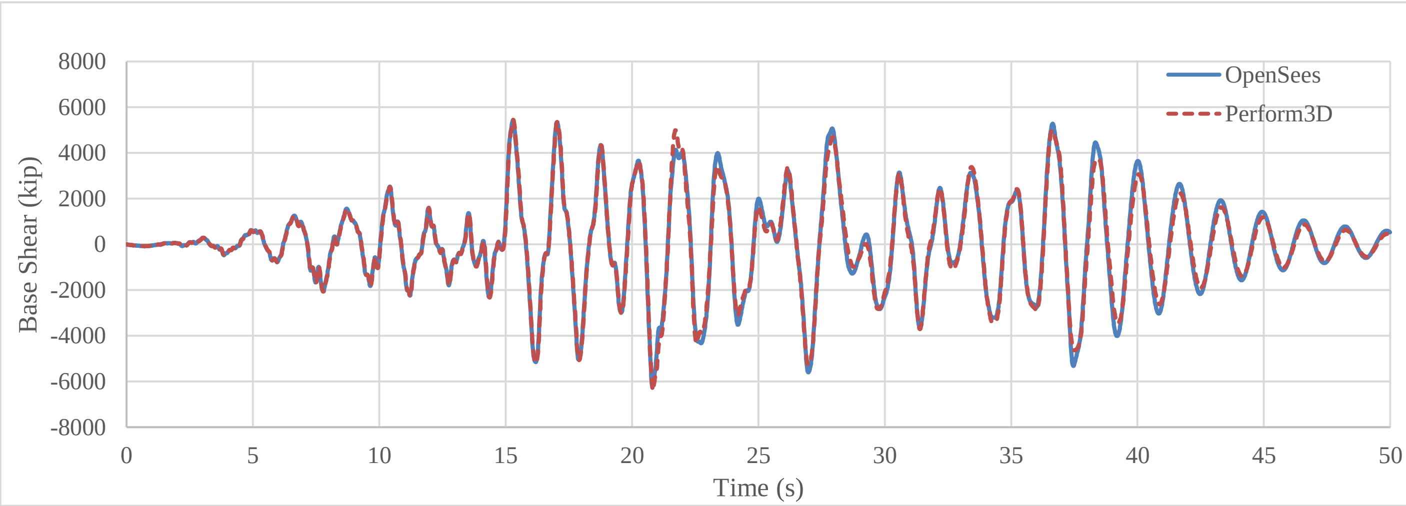

Figure 19: Base Shear (GMSF – 5.0)

Conclusions

OpenSees, so far, has only been deemed suitable for academic research and has neither been tested nor benchmarked against prevailing industry practices and standards. This paper presents a Python library, E2O, as a proof of concept that shows promising results in OpenSees for conducting professional analysis. Using this Python library, we performed nonlinear response history analyses in OpenSees and Perform3D and compared their results. We found good agreement between the two software for modal, linear response history, nonlinear pushover, and nonlinear response history analyses. However, very large magnitude earthquake ground motion produced differing post-peak responses for NLRHA performed in OpenSees and Perform3D. Such post-peak differences are attributable to different hysteretic models used in the two software as the larger ground motion produced peak beam hinge rotations in excess of rotation corresponding to initiation of strength loss, which was also the point of divergence for the two hysteretic models.

One of the biggest hurdles for engineers in using OpenSees is the absence of a graphical user interface. The E2O Python library presented in this paper solves this problem by using a commercially available software, like ETABS, that engineers are already familiar with as front-end to an OpenSees-powered backend. We show that the use of OpenSees in professional nonlinear analysis projects can be simplified when supported by GUI of commercial software, ETABS in this case. As shown in this paper, once the ETABS model is generated, the OpenSees model can be created in one click using the E2O converter. Based on the modeling rules in ETABS, all linear and nonlinear properties can be directly imported to OpenSees.

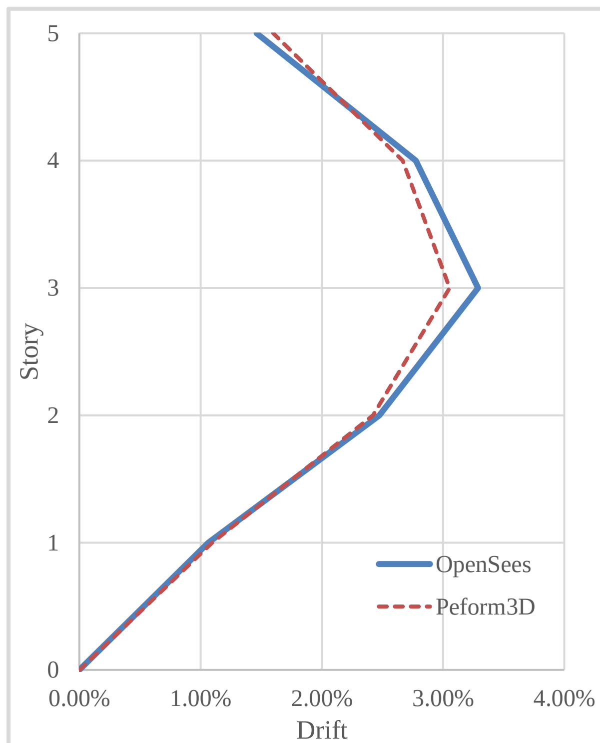

Figure 20: Story Drift (GMSF – 5.0)

We highlighted in this paper that OpenSees provides more flexibility in modeling hysteretic behavior than what is available in industry standard software. OpenSees also provides state-of-the-art elements, material models and the ability to create custom elements and materials. We note that this paper is a proof of concept and the E2O Python library we developed currently only has three material model options – “Bilin”, “Mod. IMK Peak Oriented”, and “Mod. IMK

Pinching" material. We intend to add that more material models as a part of future work.

Computational expense is another major factor that governs the decision of choosing any software tool by professionals. We showed in this paper that the current library has the capabilities of achieving the computational speed comparable to Perform3D. We highlighted in this paper that OpenSees provides the option of running an analyses parallelly in multiple CPU cores by discretizing the structure model, this is a part of the future work for this library. In most commercial software it is only possible to run an analyses per CPU core. This parallel processing in OpenSees could be very powerful for very large and complex structures in reducing the runtime. In addition to this, OpenSees also provides the option of using GPU based solvers which has proven to reduce computational time by orders of magnitude relative to CPU based solvers. OpenSees also provides the ability to use different iterative solution strategies, which can be utilized to further boost computational speed.

The current implementation of the E2O library has shown promising results and provided a springboard for an OpenSees tool that can, in the future, be used by professional to solve complicated problem while leveraging state-of-the-art research.

Future Studies

This paper includes some literature review that formulates several problems to be solved in the future. The intent of this paper was to provide a proof of concept while starting a library which can be continuously under development to add new capabilities. Below is a list of tasks that will be the immediate focus of the authors after the effort presented in this paper.

1. Current library only uses the "Bilin", "Mod.IMK Peak Oriented" and "Mod.IMK Pinching" material, which can be extended to other material models.
2. Develop the library to extend the capabilities to other element nonlinearity like columns, panel zones, etc.
3. Extend the E2O library to generate OpenSees model for braced frame, shear wall and reinforced concrete moment frame.
4. Auto-generate material properties with cyclic degradation using distributed plasticity or different mod.IMK models in OpenSees to match performance with code defined backbones.
5. Develop the option to use modal damping for dynamic analysis.
6. Develop library to unable parallel computing that will run the same analysis in multiple CPU cores by dividing the analysis model. This capability currently does not exist in most commercially available software, they only provide the option of running one analysis per CPU core.
7. Implement GPU based fast solvers to exceed the performance of the parallelization in CPU when the hardware exists.

Acknowledgement

We would like to acknowledge Simpson Gumpertz & Heger Inc. and Degenkolb Engineers for supporting the work done during the writing of this paper. We would also like to thank Prof. Dimitrios Lignos for providing guidance on the use of OpenSees' implementation of modified IMK model.

References

ASCE, 2017, *Seismic Evaluation and Retrofit of Existing Buildings*, ASCE 41, 2017, American Society of Civil Engineers, Reston, VA

Chopra, A., 2016, "Dynamics of structures: Theory and application to earthquake engineering, Fifth Edition", Pearson, August 2016, Prentice Hall, Upper Saddle River, NJ

Chopra, A., McKenna, F. & Qian, X., "Modeling viscous damping in nonlinear response history analysis of buildings for earthquake excitation," *PEER Report No 2020/01*, June 2020 Pacific Earthquake Engineering Research Center, Berkeley, CA

CSI, 2020, *ETABS Software*, version 18.1.1, Walnut Creek, CA

CSI, 2018a, *Perform3D Software*, version 7, Walnut Creek, CA

CSI, 2018b, *Perform3D: User Manual*, Walnut Creek, CA

FEMA, 2018, *FEMA P-2006, Example Application Guide for ASCE/SEI 41-13 Seismic Evaluation and Retrofit of Existing Buildings*, Federal Emergency Management Agency, Washington, D.C.

Ibarra, L.F., Medina, R.A., and Krawinkler, H., 2005, "Hysteretic models that incorporate strength and stiffness deterioration", Earthquake Engineering and Structural Dynamics, 34: 1489-1511. doi:10.1002/eqe.495

Lignos, D.G., and Krawinkler, H., 2011, "Deterioration modeling of steel components in support of collapse prediction of steel moment frames under earthquake loading", Journal of Structural Engineering, ASCE, Vol. 137 (11), 1291-1302.

McKinney, W. & others, 2010, “Data structures for statistical computing in python”, In Proceedings of the 9th Python in Science Conference. pp. 51–56.

Mondkar, D.P., Powell, G.H., 1978, “Evaluation of Solution Schemes for Nonlinear Structures”, Computers & Structures, Volume 9, No. 3, pp. 223-236.

Moore, K. S., Ranchal, P., Singhania, A., 2019, “Using Computer Automation to Simplify Seismic Evaluation of Existing Buildings,” Proceedings of 2019 SEAOC Convention, SEAOC, Squaw Creek, CA

OpenSees Wiki, 2010, “Modified Borrego Mtn ground motion – RSN#36”, https://opensees.berkeley.edu/OpenSees/manuals/ExamplesManual/ExamplesFiles/example1.intro/BM68elc.acc

PEER/ATC (2010). “Modeling and acceptance criteria for seismic design and analysis of tall buildings”, PEER/ATC-72-1, prepared by the Applied Technology Council in cooperation with the Pacific Earthquake Engineering Research Center, Redwood City, CA

Prakash, V., 1992, “Dynamic Response Analysis of Inelastic Building Structures: The DRAIN Series of Computer Programs,” (Doctoral dissertation, University of California, Berkeley), Retrieved from Earthquake Engineering Online Archive.

PSF, 2019, *Python: A dynamic, open source programming language*, Python Software Foundation. URL https://www.python.org/

Scott, M.H., Fenves, G.L., 2010, “Krylov Subspace Accelerated Newton Algorithm: Application to Dynamic Progressive Collapse Simulation of Frames”, Journal of Structural Engineering, ASCE, Vol. 136 (5), 473-480.

Tian, Y., Xie, L., Zhen, X., and Lu, X., 2015, “GPU-Powered High-Performance Computing for the Analysis of Large-Scale Structures Based on OpenSees,” Computing in Civil Engineering 2015, June 21-23, 2015, Austin, TX

Zhu, M., McKenna, F., and Scott M.H., 2018, “OpenSeesPy: Python library for the OpenSees finite element framework”, *Software X, Volume 7*, Elsevier B.V.