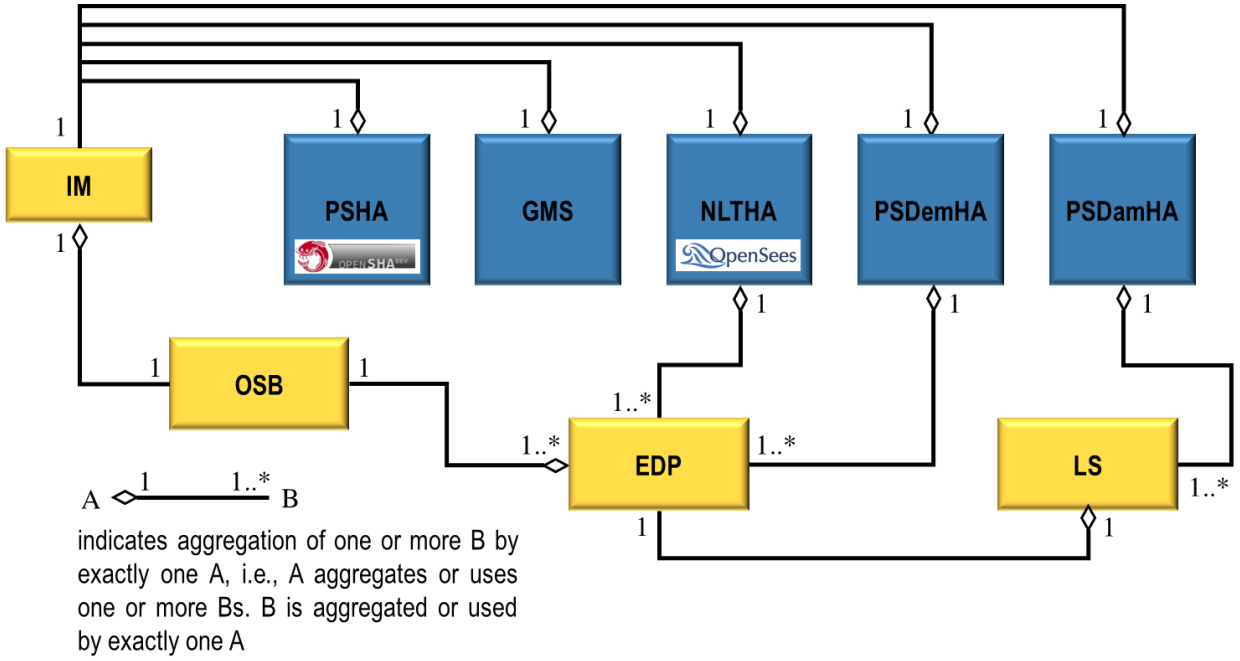
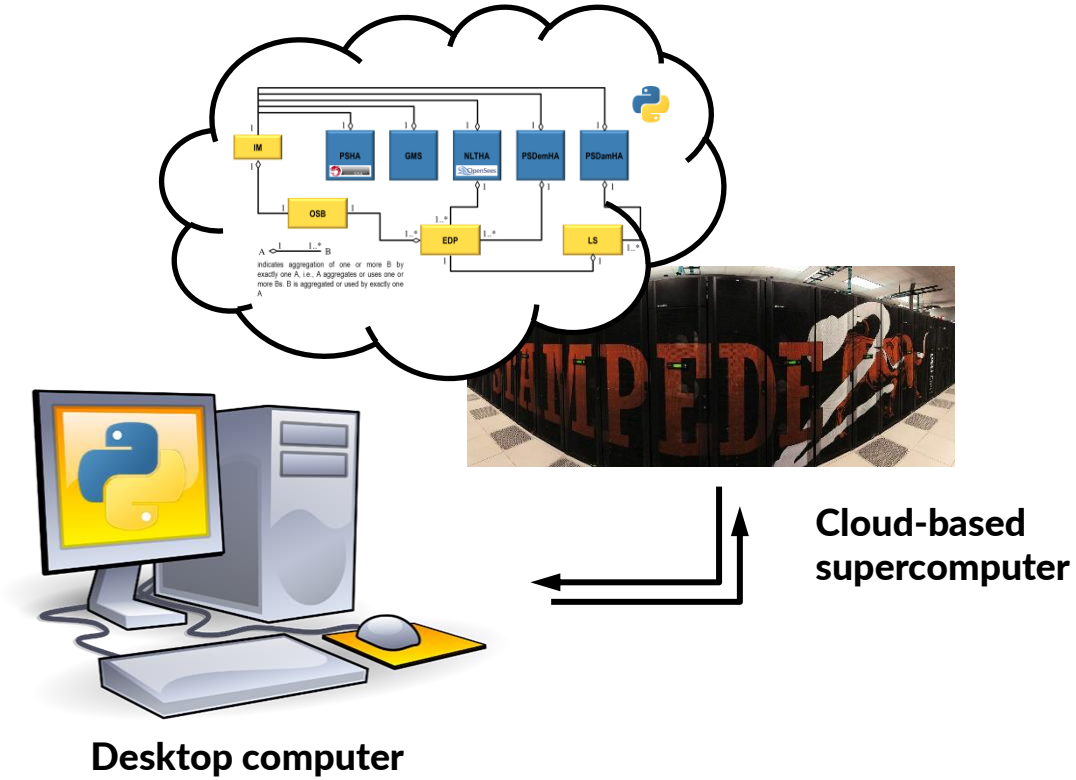
PyPBEE (PBEE for Python) @ UC San Diego

- **A software framework admitting modular and flexible class definitions of the interface variables (in yellow) of the PBEE framework (i.e., OSB, IM, EDP, and LS) to be used with the analysis classes (in blue) providing generic functionality that can be selectively changed by an analyst thus providing application-specific software**

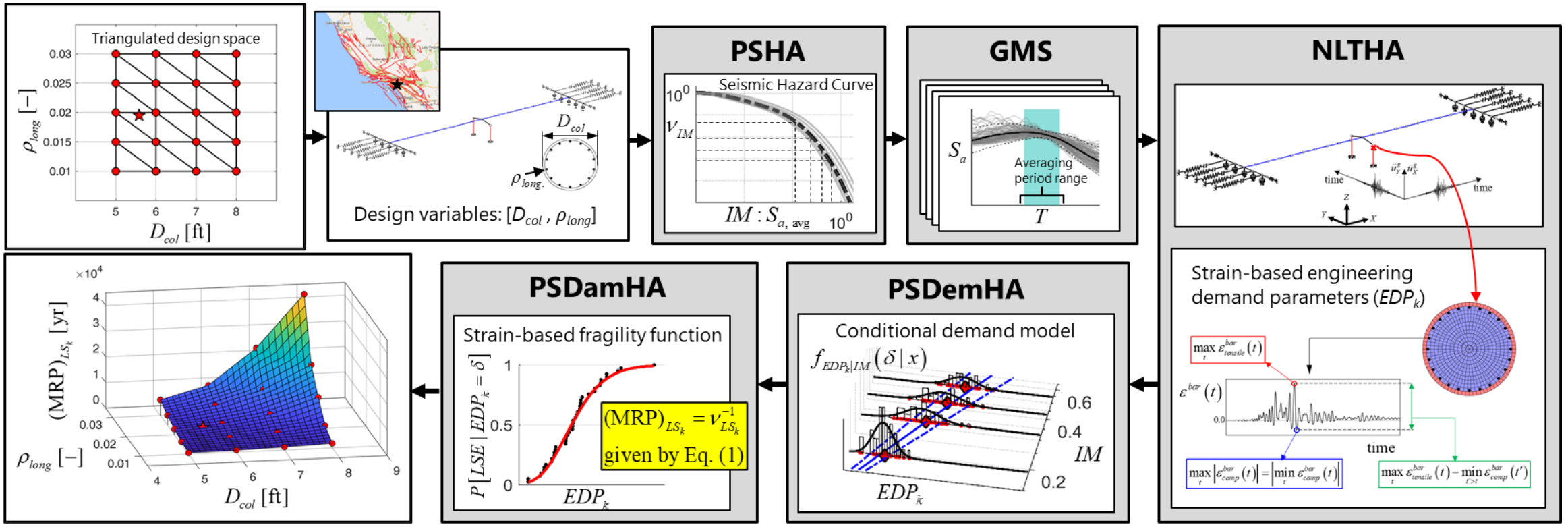


Unified modeling language (UML) diagram describing relationship between objects of different classes involved in PyPBEE



Parametric Probabilistic Risk-targeted Seismic Performance Assessment

$$\nu_{LS_k} = \int_{IM} \int_{EDP_k} P[C_k < EDP_k | EDP_k = \delta] \,.\, f_{EDP_k|IM}(\delta|x) \,.\, d\delta \,.\, |d\nu_{IM}(x)| \qquad \text{Eq. (1)}$$



PSHA: Prob. Seismic Hazard Analysis
GMS: Ground Motion Selection
NLTHA: Nonlinear Time Hist. Analysis

PSDemHA: Prob. Seismic Demand Hazard Analysis
PSDamHA: Prob. Seismic Damage Hazard Analysis

Publication-ready Plotting Capabilities

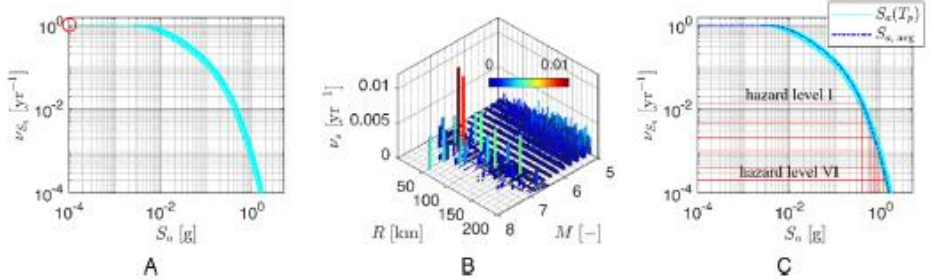


FIGURE 6 (a) SHCs in terms of $S_a(T)$ for each of the 10 discrete periods used in the averaging period range, (b) MARs of occurrences of all M-R scenarios, and (c) SHC in terms of $S_{a,\,avg}$

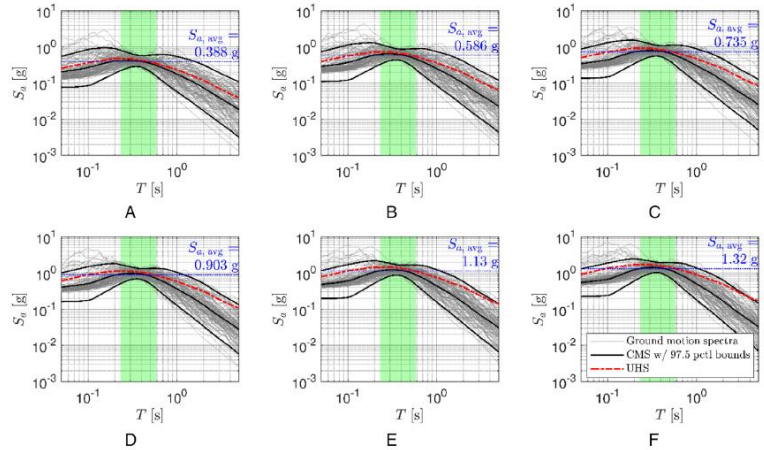


FIGURE 7 Site-specific risk-consistent ground motion ensembles at (a) hazard level I, (b) hazard level II, (c) hazard level III, (d) hazard level IV, (e) hazard level V, and (f) hazard level VI

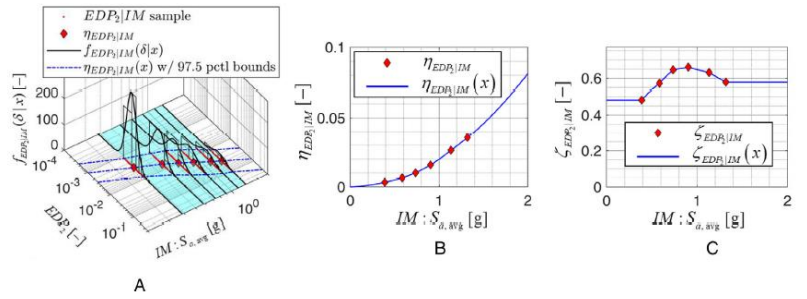


FIGURE 9 (a) Conditional probability distributions of EDP_2 given IM, Interpolation/extrapolation model for (b) $\eta_{EDP_2|IM}$ versus IM, and (c) $\zeta_{EDP_2|IM}$ versus IM

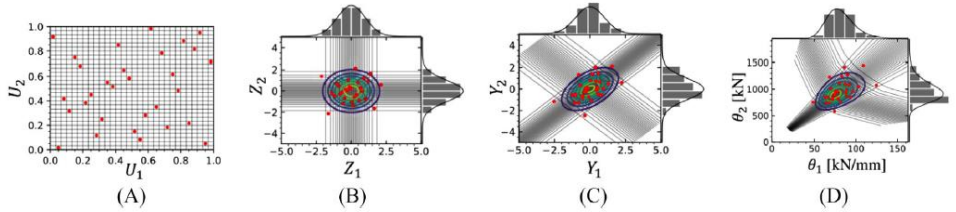
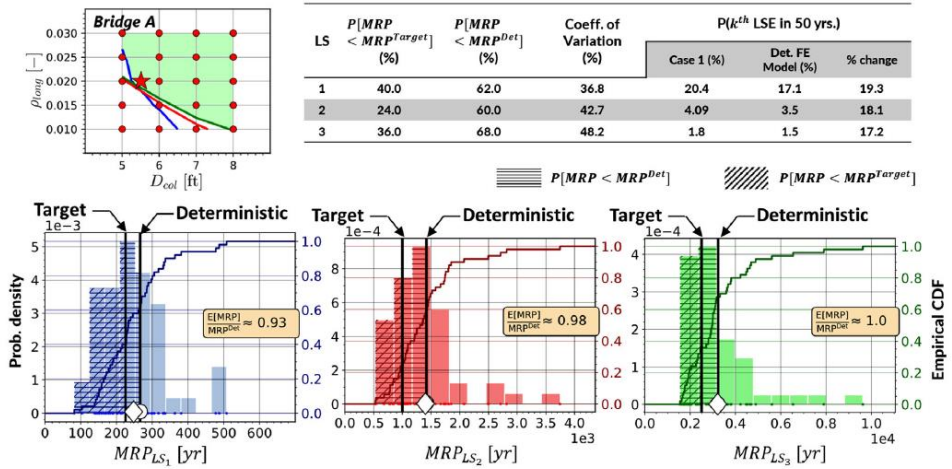


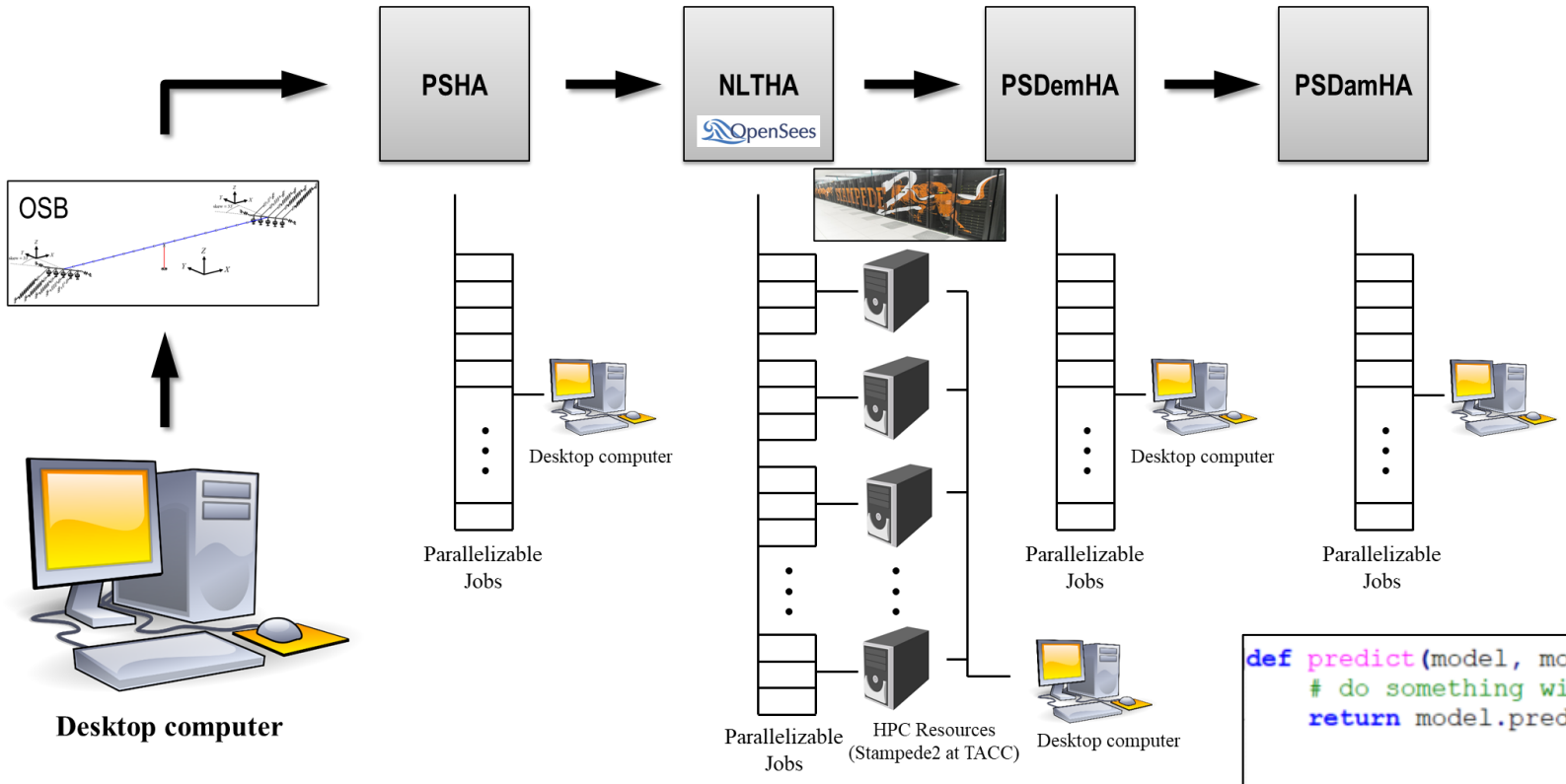
FIGURE 8 LH sampling of correlated RVs $\boldsymbol{\Theta} = [\tilde{K}_{50},\ \tilde{F}_{ult}]^T$ for Bridge A (contour lines in (D) represent the Nataf joint PDF of $\boldsymbol{\Theta}$ and the regions between grid lines represent equiprobable strata)



LS	$P[MRP < MRP^{Target}]$ (%)	$P[MRP < MRP^{Det}]$ (%)	Coeff. of Variation (%)	P(k^{th} LSE in 50 yrs.) Case 1 (%)	Det. FE Model (%)	% change
1	40.0	62.0	36.8	20.4	17.1	19.3
2	24.0	60.0	42.7	4.09	3.5	18.1
3	36.0	68.0	48.2	1.8	1.5	17.2

FIGURE 10 Results of analysis Case 1 for Bridge A

PyPBEE Multiprocessing



Python Multiprocessing

```
def predict(model, model_args, x):
    # do something with model_args
    return model.predict(x)


def predict_parallel(x, model, model_args, my_pool, n_part):
    predictor = partial(predict, model, model_args)
    if my_pool is not None:
        x = np.array_split(x, n_part)
        pred = np.concatenate(my_pool.starmap(predictor, list(zip(x)))).ravel()
    else:
        pred = predictor(x)
    return pred
```