

# **Development of Computer Tool and GUI for Performance-Based Earthquake Assessment**

Nicole A. Paul

*Stanford University  
Department of Civil & Environmental Engineering*

*Submitted for final project of CEE 385 Performance-Based Earthquake Engineering*

Fall 2015

## Overview

This report documents the creation of a computer tool and graphical user interface developed in MATLAB to assess the seismic performance of a given structure. The computer tool is called the “Seismic Performance Assessment Tool”, abbreviated as “SPAT”. The inputs and outputs of the program are summarized below, separated by the categories in which they are contained. Throughout this report, an example 4-story building is used with a fundamental period of  $T=2.0\text{s}$  in San Francisco for Site Class D.

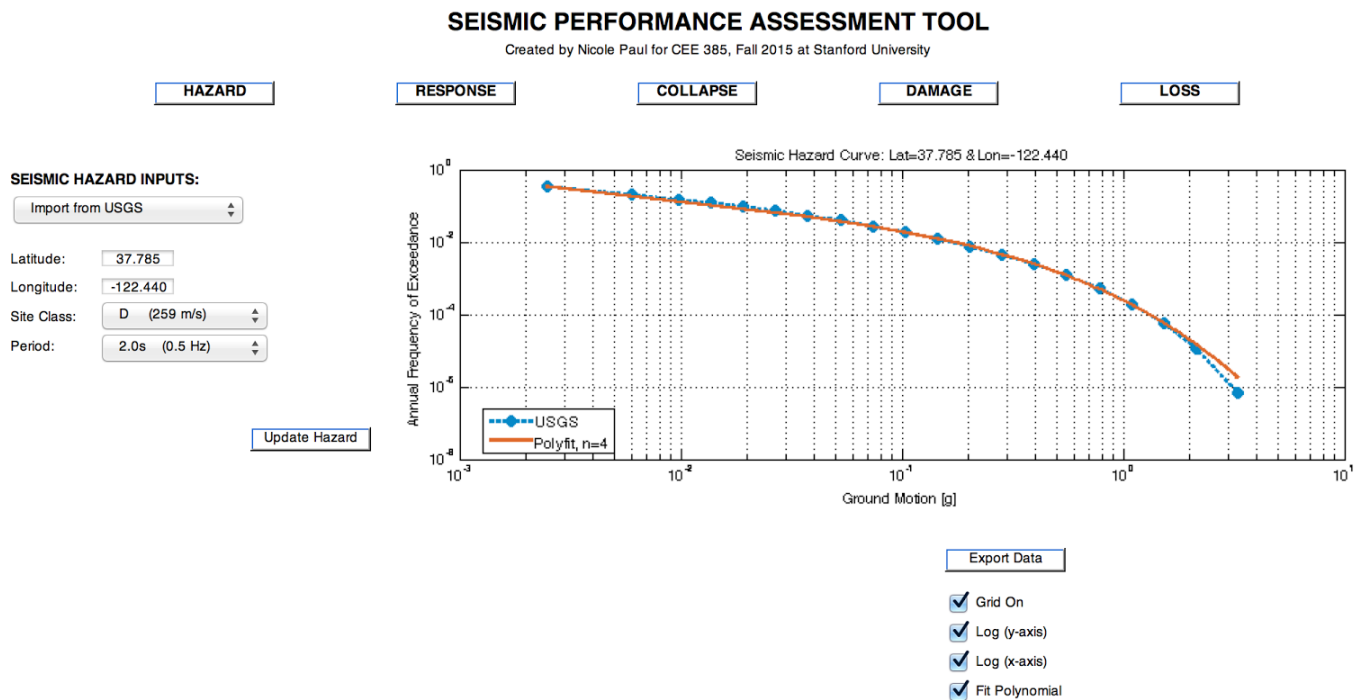
## Hazard

The seismic hazard is provided by the USGS hazard curve tool, and thus internet is required for this portion. In a future update, the program may also take in a user-defined seismic hazard curve.

In order to download and read the hazard curve from USGS, the following inputs are required:

- Longitude
- Latitude
- Site class (based on shear wave velocity of soil)
- Site period

With the above inputs, SPAT writes a csv file that is saved into a subdirectory named “SHC”. SPAT then reads in the csv file and plots the seismic hazard curve. By default, both the x-scale and y-scale are in log and with the grid on. Using the checkboxes shown below the seismic hazard curve, one can toggle the grid, log x-axes, and log y-axes on and off. In addition, a polynomial fit to the seismic hazard curve can be toggled on. This polynomial fit is restricted to be for a 4th order polynomial. The polynomial is not fit on the entire range of data points shown, but a range that has been truncated at a large annual frequency of exceedance.



## Response

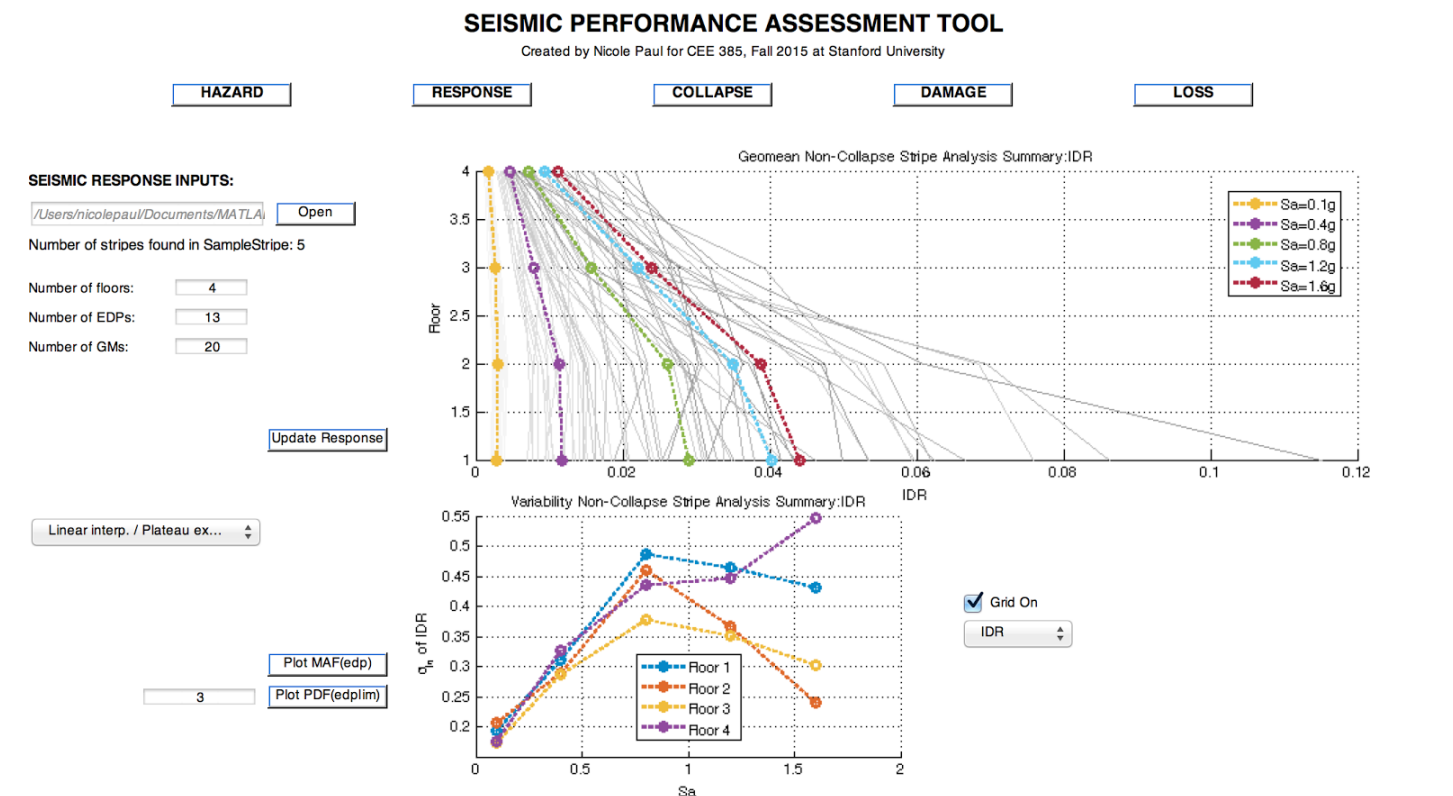
The seismic response expects an input directory, in which there is a csv file for each stripe analysis performed on the structure. Once the directory folder has been located, the number of stripes it contains is automatically computed. All csv files in that directory are assumed to be results of one stripe analysis.

Each stripe analysis csv is expected to have a consistent format, where the IM choice and value are in the top left cell. The IM choice (e.g.  $S_a$  or  $S_{a,ave}$ ) and IM value (e.g. 0.4g) are expected to be separated by an “=”. The rest of the top row and leftmost column are expected to be headers. The row headers are anticipated to be strings of the format “EDP#” where EDP is the engineering demand parameter of interest (e.g. IDR or PFA) and # is the floor number.

In addition to the directory of stripe results, the following inputs are required for successful reading of the stripe analysis csv's. Each stripe analysis csv is expected to have the same values for each of the following inputs:

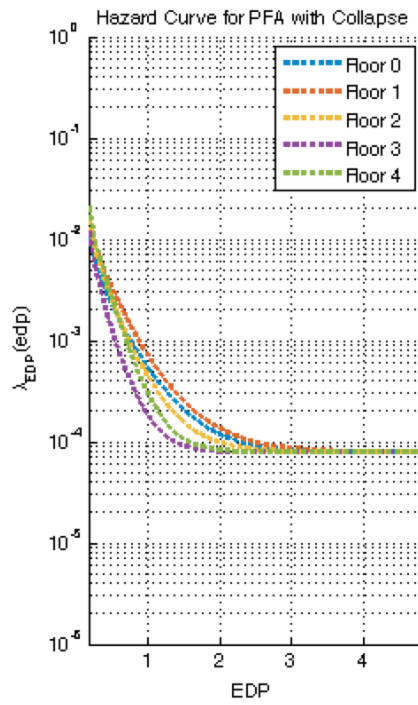
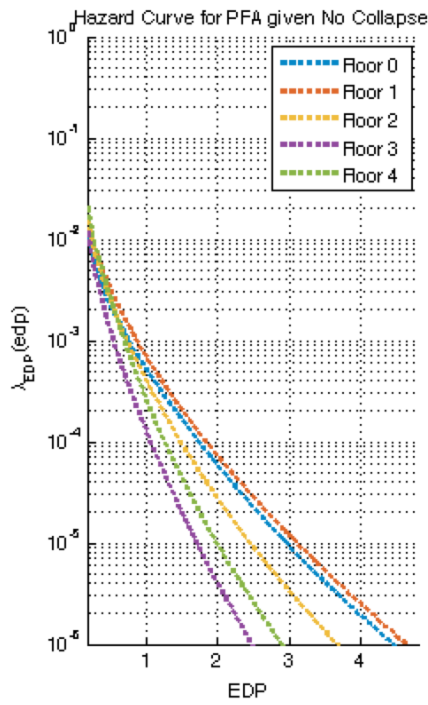
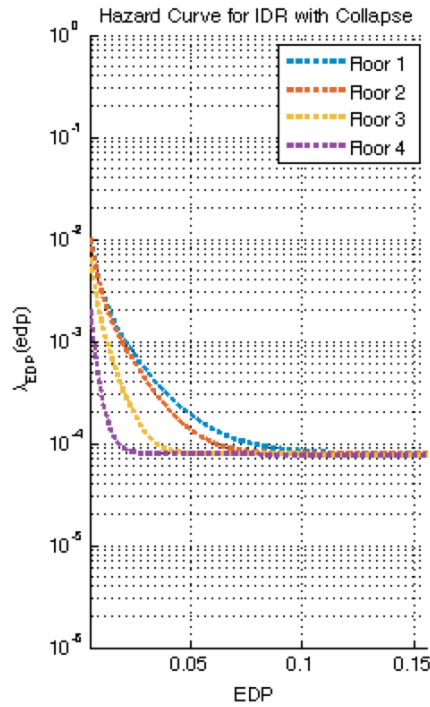
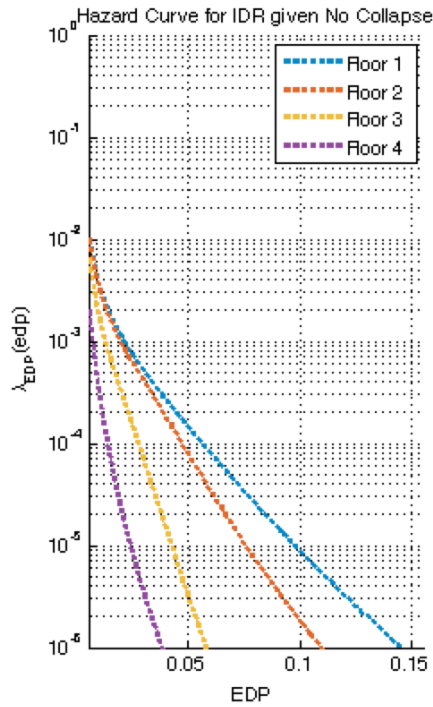
- Number of floors
- Number of EDPs
- Number of GMs

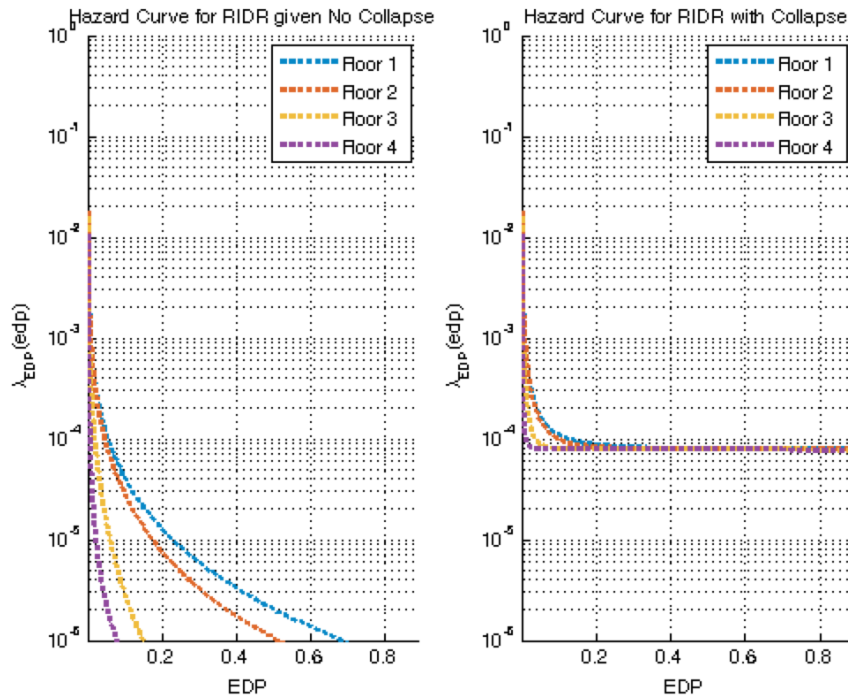
Once all of the above is input, the “Update Response” button can be pressed in order to plot the stripe results for different EDPs. The geomean result for each floor is superimposed over individual results for each motion. In addition, a smaller plot shows the variability of the given EDP for each floor as a function of IM.



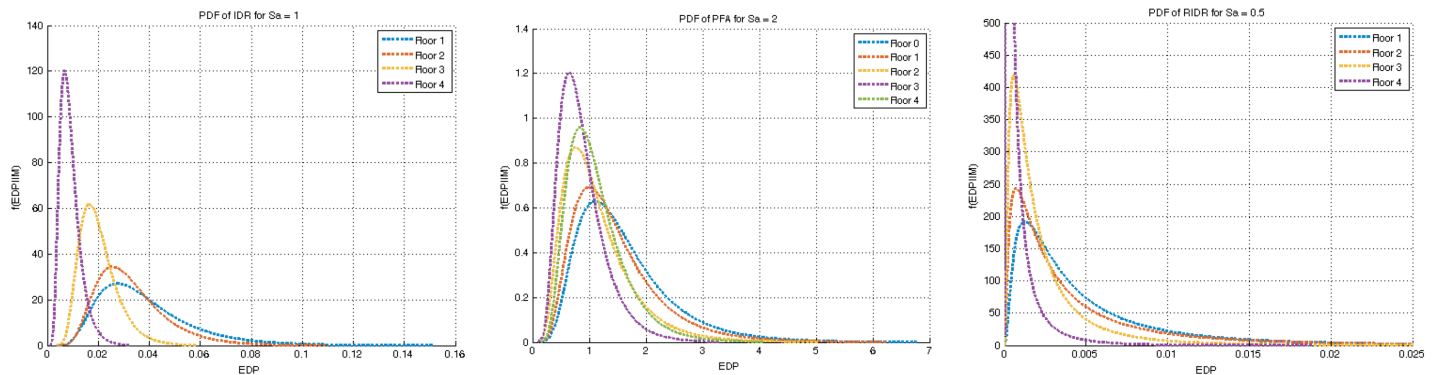
The EDP of interest can be toggled using the dropdown menu that appears to the right of the smaller axes. For this EDP, the MAF(EDP) curves and P(EDP|IM) curves for each floor can be plotted using the buttons to the left of the smaller axes. A few examples of these are shown throughout the next few pages.

The MAF(EDP) curves are shown both for non-collapse only and with collapse considered. These results demonstrate the significant floor-to-floor variation of EDP. When collapse is considered, we see that the EDP values all plateau to a value equal to MAF(C) as collapse dominates at those IM values. It is also apparent that the annual frequency of EDP goes towards infinity as EDP decreases to zero.



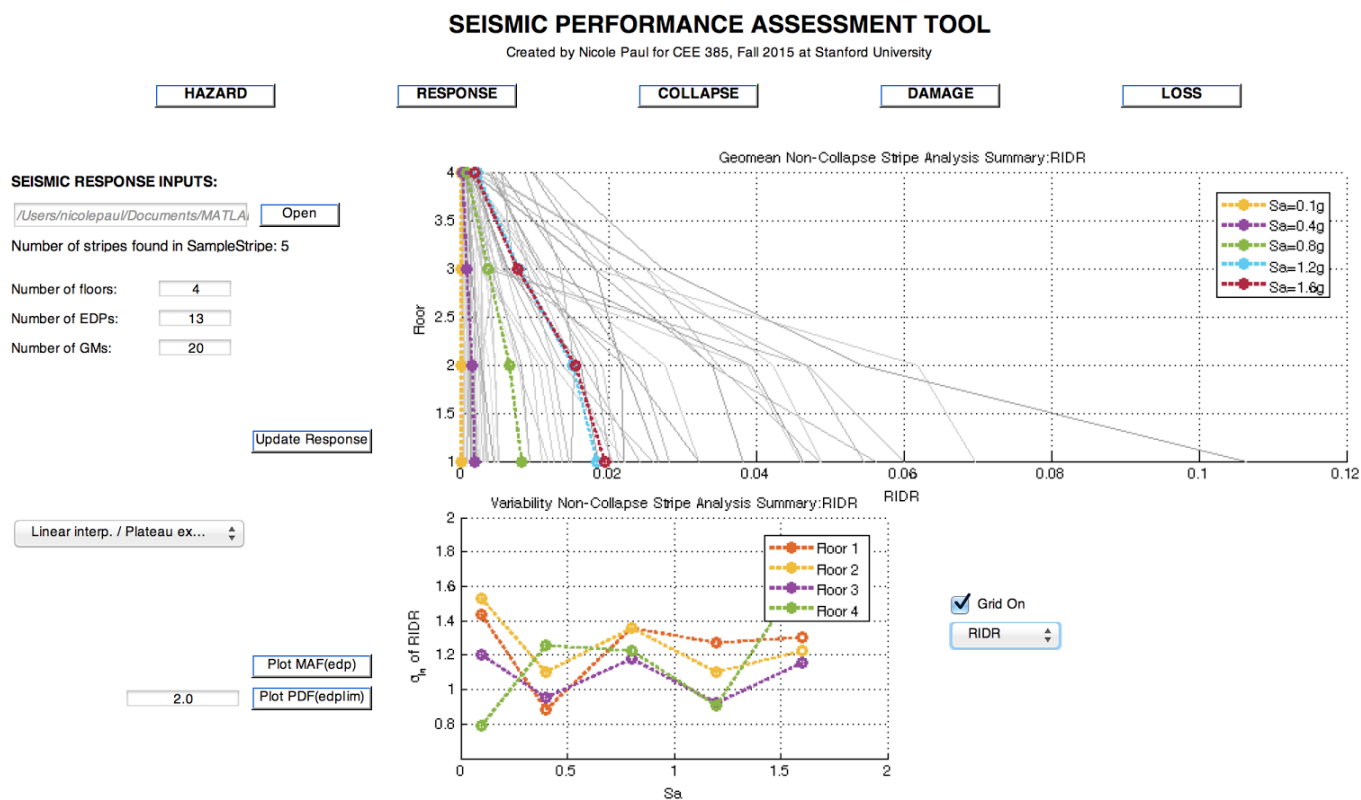
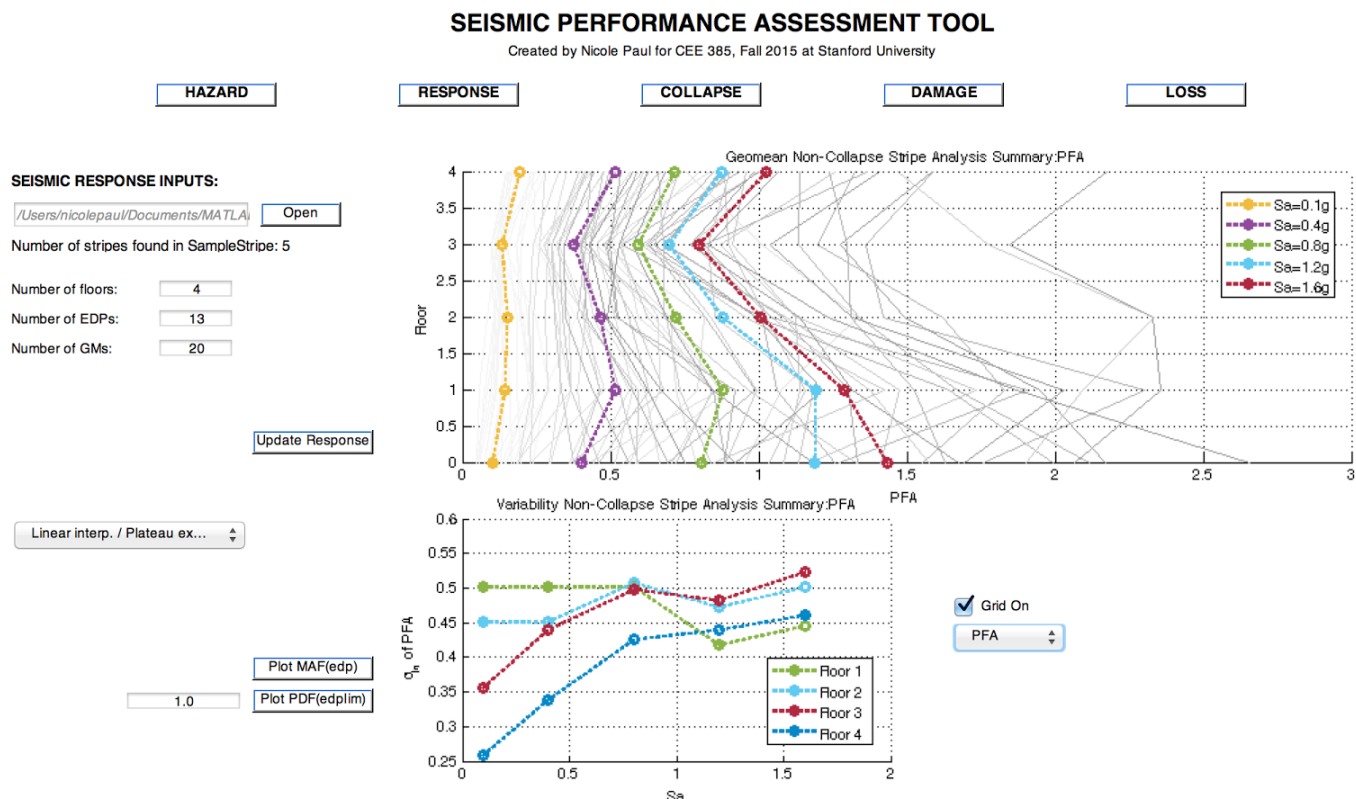


In order to calculate the above curves, as well as the  $P(\text{EDP}|\text{IM})$  curves shown later, interpolation between the stripe analyses and extrapolation outside is performed. By default, it is assumed that there will be linear interpolation between the stripe analysis IM values and a plateau beyond those IM values for both geomean and dispersion. The plateau value is equal to the maximum EDP value from the stripe analyses. In a future version of SPAT, different interpolation and extrapolation methods could be included. Note that for linear interpolation, it was assumed that each EDP was 0 at an IM value of 0.



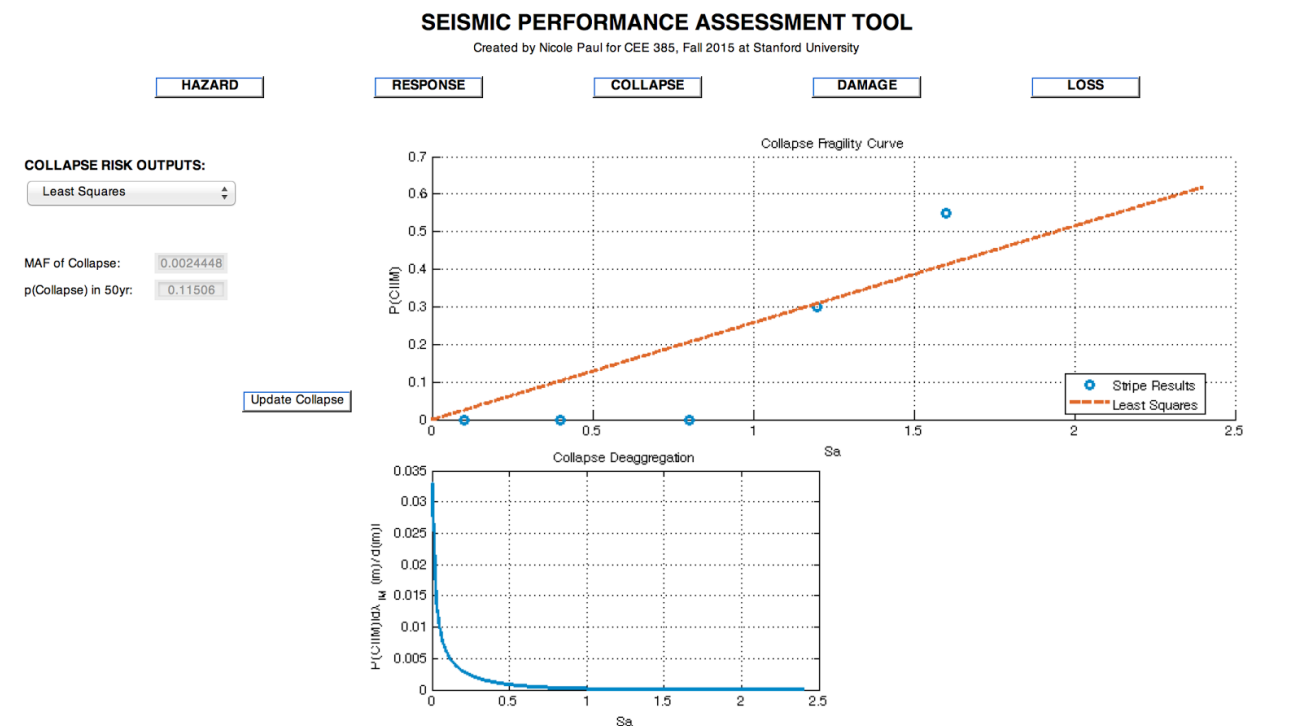
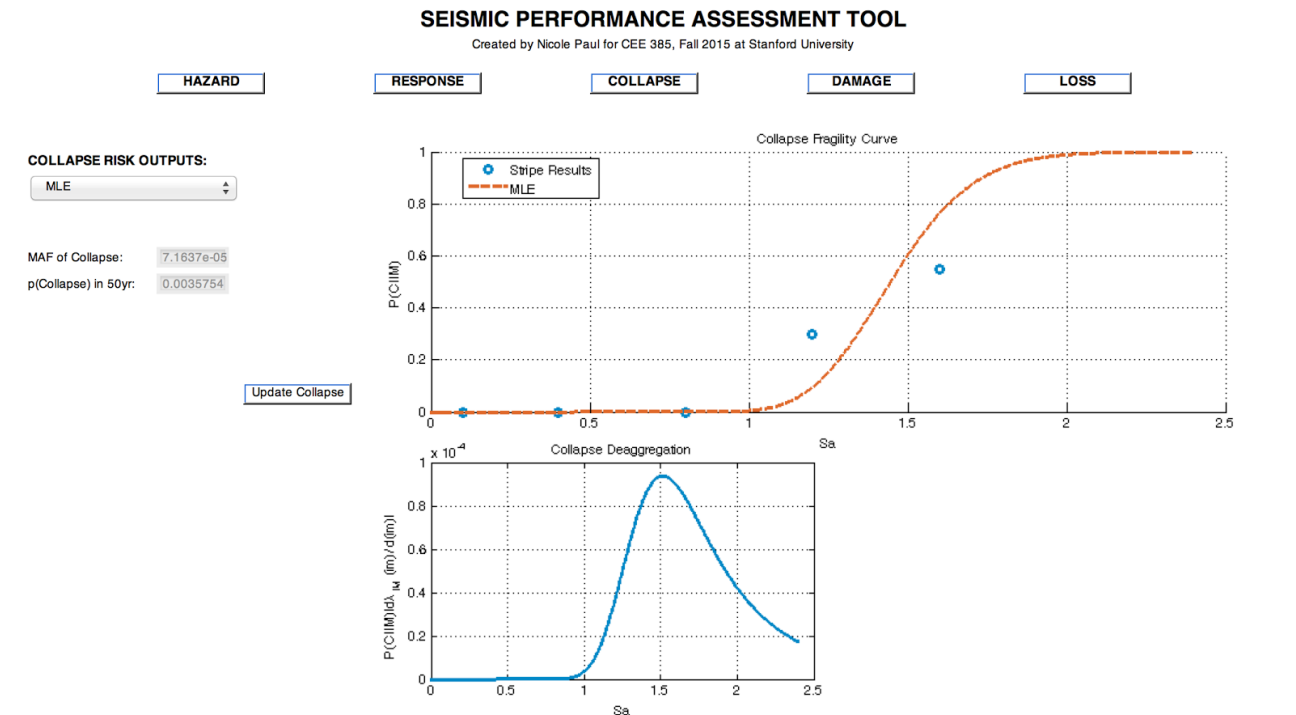
Above are the computed PDF's of each EDP, conditioned on an IM value that is input by the user. The IM value chosen by the user is shown in the title of the plot so that it is apparent. Again, we see the significant floor-to-floor variability. For example, the floor drifts tend to be much larger and with larger variability in the first two floors.

Below are examples of the GUI for different EDP choices.



## Collapse

Using the “Hazard” and “Response” inputs from earlier, the “Collapse” window can be enabled. The user can choose a fit method in order to get a collapse fragility based off of limited stripe analyses. The default fit is “MLE” assuming a lognormal distribution, however “Least Squares” can also be used. In future updates to SPAT, other fit methods may be incorporated.



This window also automatically calculates the mean annual frequency of collapse and the probability of collapse in 50 years. These values vary significantly with the chosen fit method.

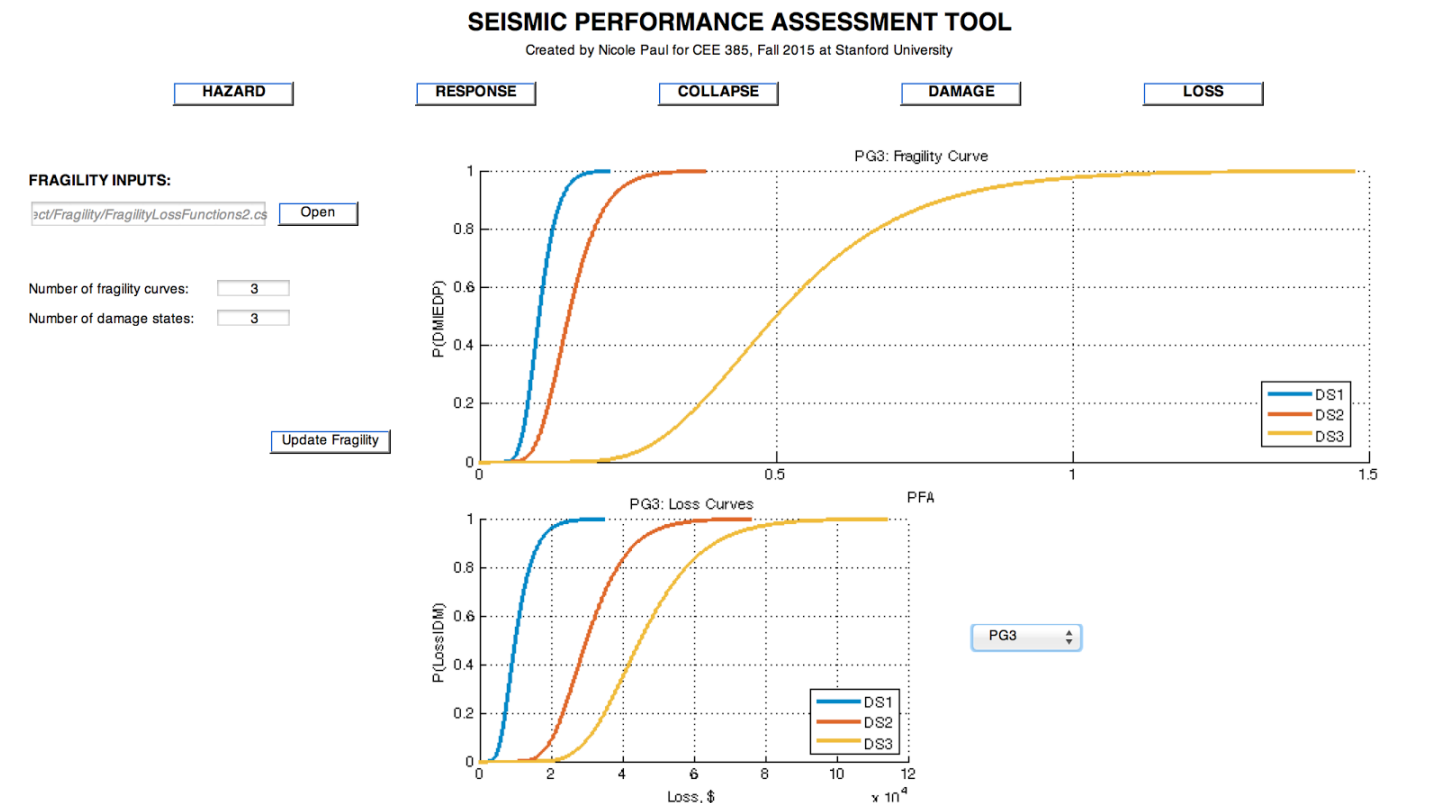


## Damage

For the “Damage” window, it is expected that the user will provide a csv file of a certain format. This csv file will contain the lognormal parameters for damage and loss for each performance group in the building. At this time, it is assumed that there will be one of each of those components on each floor. In addition to the csv file of fragility curves, the user is also expected to input:

- The total number of fragility curves (i.e. performance groups)
- The number of damage states
- Demolition median RIDR and dispersion

At this time, it is assumed that the number of damage states is the same for each performance group. If one wanted to specify a performance group that less damage states than another, one could assign it the same damage and loss parameters for all subsequent damage states.



SPAT reads each performance group, the EDP it is sensitive to, and its parameters for damage and loss. Once the EDP is known, SPAT can cross-reference the EDP to the stripe results which show the number of floors or storeys that EDP is sensitive to. For example, PFA may refer to the ground in addition each floor of the building whereas IDR may not. The user can toggle through performance groups using the dropdown menu to the right of the smaller axes.



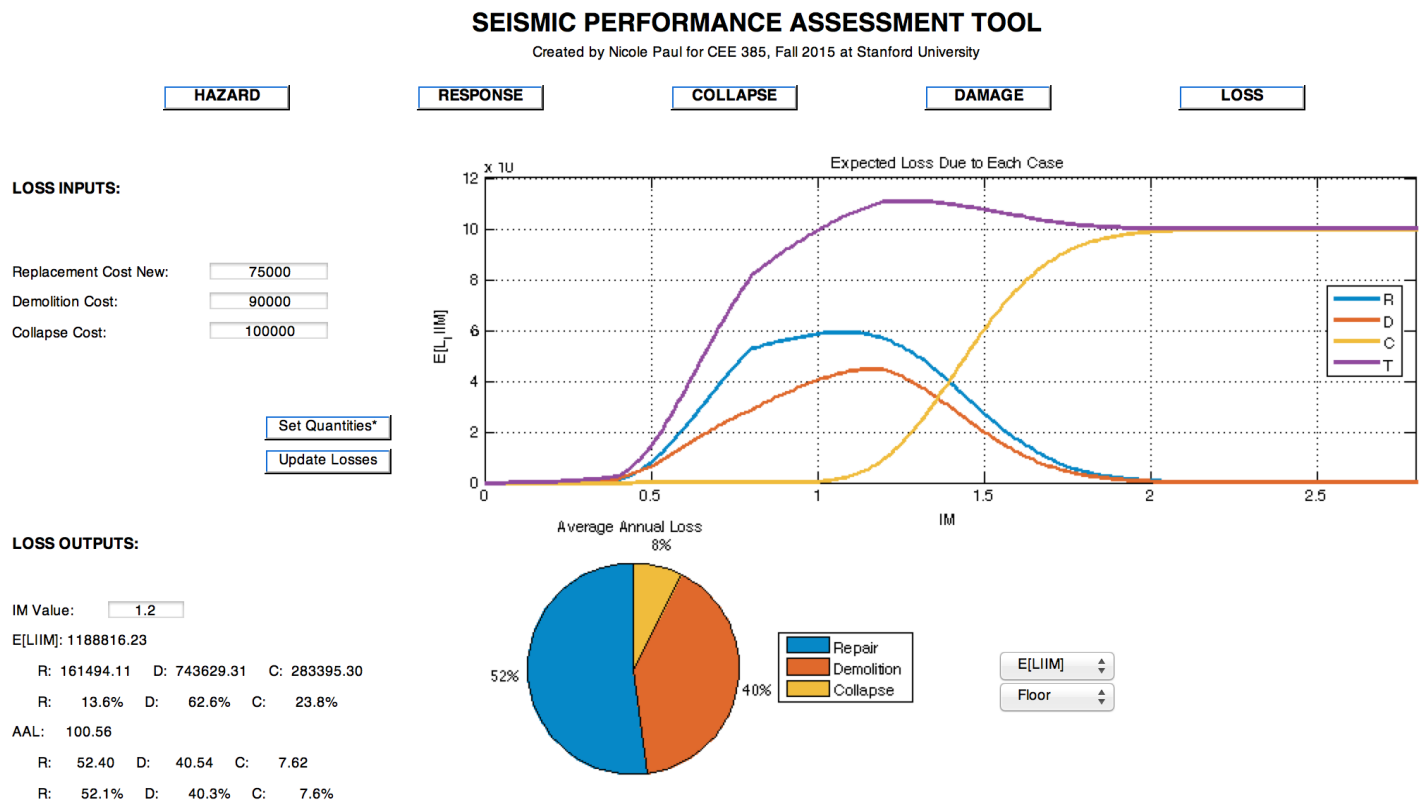
In a future update to the program, a user may input the quantity of each performance group on each floor. It could also be helpful to distinguish between structural and nonstructural components. It should also be noted that the “\$” symbols which occasionally were used in the provided fragility csv were removed

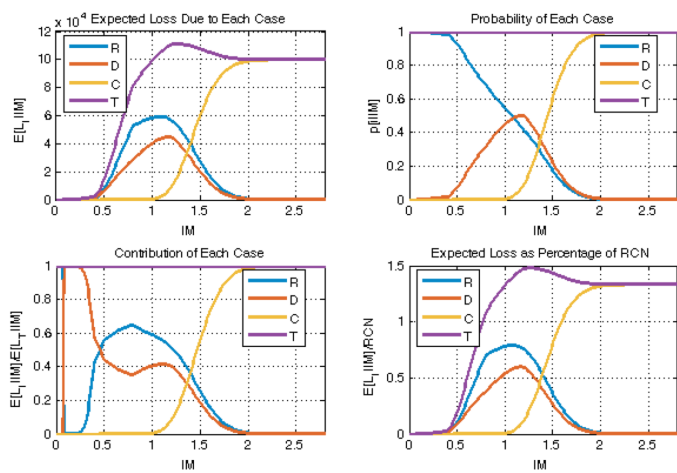
## Loss

For the “Loss” window and calculations, the user is expected to input the following values:

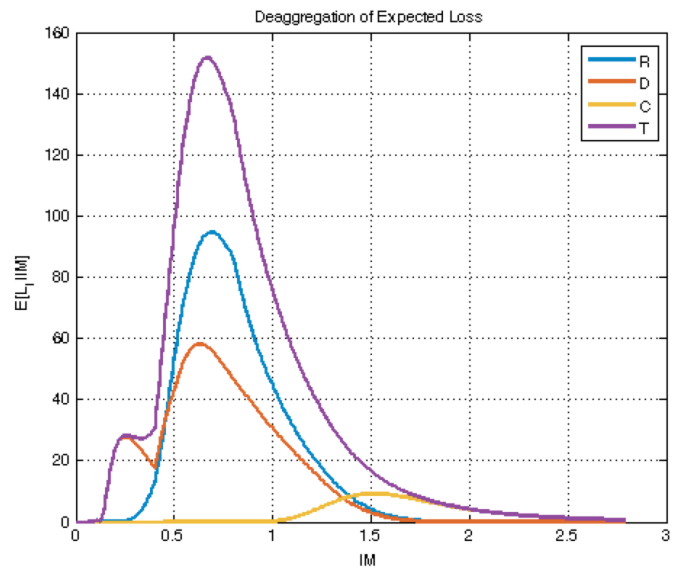
- Replacement cost new (RCN)
- Demolition cost
- Collapse cost
- Quantities of each performance group

When the “Updated Losses” button is pressed, a figure pops up with subplots showing the expected loss due to each case (R, D, C or All), probability of each case, contribution of each case to the total expected loss, and the expected loss as percentage of RCN. It should be noted that the demolition fragility was assumed to have a median of 1.5% RIDR and a dispersion of 0.3.





Annual expected loss = \$100.56  
 due to Repairs = \$52.40  
 due to Demolition = \$40.54  
 due to Collapse = \$7.62



Percentage of total annual expected loss  
 Repairs = 52.11%  
 Demolition = 40.31%  
 Collapse = 7.58%

## Code Organization

The computer tool is organized into a GUI function, subdirectories for inputs, and a series of MATLAB functions. For easier customization of the GUI, the GUI figure is produced entirely from an m-file as opposed to a .fig using GUIDE. As much as possible, separate functions are used for calculations. All Callbacks within the GUI function handle plotting or call separate functions. These separate functions have the prefix “get\_” or “calc\_” depending on whether they are reading/writing/organizing or performing calculations. Below is documentation on those functions. Some comments have been redacted to save space.

*get\_usgsHazard* -- writes and reads USGS seismic hazard curve data based off of user inputs

```
%% Main function
function shcdat = get_usgsHazard(lat, long, siteclass,
period)

% Processing inputs
[site, per] = get_usgsFormat(siteclass, period);

% Forming url for data access
str1 =
'http://geohazards.usgs.gov/hazardtool/curves.php?format
=2&lat=';
str2 = '&lon=';
str3 = '&site=';
str4 = '&period=';
url = strcat(str1,lat,str2,long,str3,site,str4,per);

% Saving data locally
filename =
strcat('LAT',lat,'LON',long,'SC',site,'T',per,'.csv');
shcFile = fullfile('SHC',filename);
urlwrite(url,shcFile);

% Importing data
try dlmread(shcFile, ',', 1, 5);
    shcdat = dlmread(shcFile, ',', 1, 5)';
catch
    shcdat = dlmread(shcFile, ',', 1, 6)';
end

end
```

```
%% Subfunction
function [site, per] =
get_usgsFormat(siteclass, period)

% Switching site type to match usgs
expected string
switch siteclass
    case 1
        site = '2000';
    case 2
        site = '1150';
    case 3
        site = '760';
    case 4
        site = '537';
    case 5
        site = '360';
    case 6
        site = '259';
    case 7
        site = '180';
end

% Switching period to match usgs
expected string
switch period
    case 1
        per = '0p00';
    case 2
        per = '0p10';
    case 3
        per = '0p20';
    case 4
        per = '0p30';
    case 5
        per = '0p50';
    case 6
        per = '0p75';
    case 7
        per = '1p00';
    case 8
        per = '2p00';
    case 9
        per = '3p00';
    case 10
        per = '4p00';
    case 11
        per = '5p00';
end
```

### ***get\_idaResponse*** -- reads and organizes stripe analysis data given user inputs

```
function [imval, stripeDat, imchoice, unitstr, edptext] = get_idaResponse(dirpath, nstripe, nedps, ngms)

% Opening directory with IDA results
FileList = dir(dirpath);
FileList = FileList(arrayfun(@(x) x.name(1), FileList) ~= '.');
N = size(FileList, 1);
% Display warning if nstripe does not match number of stripes found
if nstripe ~= N
    warning('Number of stripes input does not match number of stripes found');
end

% Initialization of variables
imchoice = cell(nstripe, 1);
unitstr = cell(nstripe, 1);
imval = NaN(nstripe, 1);
edptext = cell(nstripe, 1);
stripeDat = cell(nstripe, 1);

% Reading each data file
for i = 1:nstripe

    % Reading IM case
    fid = fullfile(dirpath, FileList(i).name);
    imstr = textscan(fopen(fid), '%s', 'Delimiter', ',');
    [tok, ~] = strsplit(imstr{1}{1}, '=');
    imchoice{i} = tok{1};
    unitstr{i} = tok{2}(end); % Assuming last character is units
    imval(i) = str2double(tok{2}(1:end-1));

    % Reading each EDP case
    headers = transpose(reshape(imstr{1}, ngms+1, nedps+1));
    edptext{i} = headers(2:end, 1);

    % Reading data
    stripeDat{i} = dlmread(fid, ',', 1, 1);

end
fclose('all');
end
```

### ***get\_fragilityLoss*** -- reads and organizes fragility data provided by user

```
function [compnames, compedp, fragDat] = get_fragilityLoss(nfragility, nds, fragfile)

% Import fragility csv
nparam = 2; % central value and uncertainty value
ninfo = 2; % damage and loss
textdat = textscan(fopen(fragfile), '%s', 'Delimiter', ',');
numdat = dlmread(fragfile, ',', [1 2 nfragility 1+nds*nparam*ninfo]);
fclose('all');

% Looping through fragility info
ncol = nds*nparam*ninfo + 2;
fragDat = cell(nfragility, 1);
compnames = cell(nfragility, 1);
compedp = cell(nfragility, 1);
inds_theta = 1:nparam:nds*nparam;
inds_beta = 2:nparam:nds*nparam;
inds_ctheta = nds*nparam+1:nparam:nds*nparam+nds*nparam;
inds_cbeta = nds*nparam+2:nparam:nds*nparam+nds*nparam;
m = 1000;
```

```

p = linspace(0,1,m);
for i = 1:nfragility
    % Text data
    compnames{i} = textdat{1}{ncol*i+1};
    compedp{i} = textdat{1}{ncol*i+2};
    % Numeric data
    fragDat{i}.theta = numdat(i, inds_theta);
    fragDat{i}.beta = numdat(i, inds_beta);
    fragDat{i}.ctheta = numdat(i, inds_ctheta);
    fragDat{i}.cbeta = numdat(i, inds_cbeta);
    fragDat{i}.nds = nds;
    % Solving at multiple points for fragility curve plottings
    fragDat{i}.x = NaN(m, nds);
    fragDat{i}.c = NaN(m, nds);
    fragDat{i}.p = p;
    for j = 1:nds
        fragDat{i}.x(:,j) = logninv(p, log(fragDat{i}.theta(j)), fragDat{i}.beta(j));
        fragDat{i}.c(:,j) = logninv(p, log(fragDat{i}.ctheta(j)), fragDat{i}.cbeta(j));
    end
end
end
end

```

**calc\_collapseRisk** -- calculates collapse fragility, deaggregation curve, MAF(C), and  $P(C, t=50yr)$

<pre> %% Main function function [IM, pCIM, deagg, fittedFC, MAF, pc50] = calc_collapseRisk(pcoeff, imval, stripeDat, fitttype, nGM)  % Calculation will be done with below range of IM values m = 500; im = linspace(1e-5, max(imval)*1.5, m)'; % Full range IM = (im(2:end)+im(1:end-1))/2; % At midpoints  % Determining probability of collapse given IM, p(C IM) % Note: Defining "NaN" as collapse nIM = numel(imval); pCIM = NaN(size(imval)); for i = 1:nIM     pCIM(i) = sum(isnan(stripeDat{i}(1,:)))/nGM; end  % Performing calculations using fit type requested fc = get_collapseFit(fitttype, imval, pCIM, nGM); fittedFC = fc(IM);  % Performing numerical integration using closed form solution of seismic % hazard curve pcim = NaN(m-1,1); pIMdiff = NaN(m-1,1); deagg = NaN(m-1,1); for i = 1:m-1     pcim(i) = fc(IM(i));     lnx = log(IM(i));     p = fliplr(pcoeff);     pIMdiff(i) = abs(((p(2)+2*p(3)*lnx+3*p(4)*lnx^2+4*p(5)*lnx^3)/ IM(i))*exp(p(1)+p(2)*lnx+p(3)*lnx^2+p(4)*lnx^3+p(5)*lnx^4));     deagg(i) = pcim(i)*pIMdiff(i); end  % Mean annual frequency of collapse MAF = trapz(IM, deagg); % Probability of collapse in 50 years pc50 = 1 - exp(-MAF*50); </pre>	<pre> %% Subfunction function [fc, FC] = get_collapseFit(fitttype, imval, pCIM, nGM)  % Fitting MLE parameters using lognormal distribution FC.mle = lognfit(imval, 0.5, [], pCIM*nGM); % Fitting least square parameterers FC.lsq = lsqnonneg(imval, pCIM);  % Writing fragility curve equation based off of fit type if strcmpi(fitttype, 'Least Squares')     fc = @(x) FC.lsq.*x; else     fc = @(x) logncdf(x, FC.mle(1), FC.mle(2)); end  end </pre>
---	--

***calc\_geoMean*** -- calculates geomean, ignoring rows with NaN values

```
function M = calc_geoMean(X, dim)

% Default dimension of 1
if nargin == 1
    dim = 1;
end

% Find all lines which have NaN and removing them
if dim == 1
    realX = X(~any(isnan(X),2),:);
else
    realX = X(:,~any(isnan(X),1));
end

% Calculating geomean of result above
M = geomean(realX, dim);

end
```

***calc\_edpParam*** -- calculates geomean and dispersion of EDP for floor given IM

```
function [thetaPoint, betaPoint] = calc_edpParam(stripeDat, imval, imPoint, nEDP)

% Processing inputs
nstripe = numel(stripeDat);

% Finding the geomean and dispersion of each EDP value, for each stripe
thetaStripe = NaN(nEDP, nstripe);
betaStripe = NaN(nEDP, nstripe);
for i = 1:nstripe
    thetaStripe(:, i) = calc_geoMean(stripeDat{i},2);
    betaStripe(:, i) = nanstd(log(stripeDat{i}),[],2) ;
end

% Figuring out whether needs to interpolate or extrapolate
interpPoint = imPoint <= max(imval);
extrapPoint = imPoint > max(imval);

% Initialization
thetaPoint = NaN(nEDP, numel(imPoint));
betaPoint = NaN(nEDP, numel(imPoint));

% Setting all extrapolated points to be a plateau at the maximum stripe
% result and all
thetaPoint(:, extrapPoint) = repmat(max(thetaStripe, [], 2),1,sum(extrapPoint));
betaPoint(:, extrapPoint) = repmat(max(betaStripe, [], 2),1,sum(extrapPoint));

% Setting all interpolated points to be linear between given stripe values
for j = 1:nEDP
    thetaPoint(j, interpPoint) = interp1([0; imval], [0; thetaStripe(j,:)'],
    imPoint(interpPoint))';
    betaPoint(j, interpPoint) = interp1([0; imval], [0; betaStripe(j,:)'],
    imPoint(interpPoint))';
end
```

### ***calc\_edpHazard*** -- Calculates the EDP hazard curve for all EDP types on all floors

```
function [lambda_edp_c, lambda_edp_nc, midedp_im] = calc_edpHazard(imval, stripeDat, nEDP,
pcoeff, typeEDP)

% Discretization
% nIM = 100;
% imPoint = linspace(1e-6,max(imval)*2, nIM)';
n = 1000;
pval = linspace(1e-8, 0.99, n)';

% Calculating collapse
% Determining probability of collapse fragility
fitttype = 'MLE';
ngms = size(stripeDat{1},2);
[IM, ~, ~, fittedFC, ~, ~] = get_collapseRisk(pcoeff, imval, stripeDat, fitttype, ngms);

nIM = numel(IM); imPoint = IM';
dIM = IM(2) - IM(1);

% Getting interpolated or extrapolated values
[thetaPoint, betaPoint] = calc_edpParam(stripeDat, imval, imPoint, nEDP);

% Initialization
edp_im = NaN(n, nEDP); pIMdiff = NaN(nIM, 1);
midedp_im = NaN(n-1, nEDP); binedp_im = NaN(n-1, nEDP);
pEDP_im = NaN(n-1, nEDP, nIM); lambda_edp_im = NaN(n-1, nEDP, nIM);
lambda_edp_im_c = NaN(n-1, nEDP, nIM); lambda_edp_nc = NaN(n-1, nEDP);
lambda_edp_c = NaN(n-1, nEDP);

% Calculating deaggregation curve for EDP
uniqEDP = unique(typeEDP);
for i = 1:numel(uniqEDP)
    indEDP = find(typeEDP == uniqEDP(i));
    edp_im(:,indEDP) = repmat(logninv(pval, log(max(thetaPoint(indEDP,round(nIM/2)),[],1))),
max(betaPoint(indEDP,round(nIM/2)),[],1),1,numel(indEDP));
    binedp_im(:,indEDP) = edp_im(2:end,indEDP) - edp_im(1:end-1,indEDP);
    midedp_im(:,indEDP) = edp_im(1:end-1,indEDP) + binedp_im(:,indEDP)/2;
end
for i = 1:nEDP
    for j = 1:nIM
        % Finding probability exceedance of that edp value
        pEDP_im(:,i,j) = 1 - logncdf(midedp_im(:,i), log(thetaPoint(i,j)), betaPoint(i,j));
    end
end
for i = 1:nIM
    % Interpolating collapse fragility curve
    % Seismic hazard curve differentiation
    lnx = log(imPoint(i));
    p = fliplr(pcoeff);
    pIMdiff(i) =
abs(((p(2)+2*p(3)*lnx+3*p(4)*lnx^2+4*p(5)*lnx^3)/imPoint(i))*exp(p(1)+p(2)*lnx+p(3)*lnx^2+p(4)*lnx
^3+p(5)*lnx^4));
    % EDP hazard curve
    for j = 1:nEDP
        lambda_edp_im(:,j,i) = pEDP_im(:,j,i)*pIMdiff(i)*dIM;
        lambda_edp_im_c(:,j,i) = (pEDP_im(:,j,i).*(1 - fittedFC(i)) +
fittedFC(i))*pIMdiff(i)*dIM;
    end
end

% Summing for all edp values
for i = 1:nEDP
    lambda_edp_nc(:,i) = sum(squeeze(lambda_edp_im(:,i,:)),2);
    lambda_edp_c(:,i) = sum(squeeze(lambda_edp_im_c(:,i,:)),2);
end
```



### *calc\_lossIntensity* -- calculates expected loss for a given IM value

```
function [bldgEL_im, floorEL_im, compqEL_im, pDS_edp_im, midedp_im, p_fordemo] =  
calc_lossIntensity(edp, frag, cqty, n)  
  
% Counting inputs  
nstory = size(edp, 2);  
ncomp = numel(frag);  
  
% Choosing number of points for EDP discretization  
p = linspace(1e-5,0.99999,n)';  
  
% Initialization  
edp_im = NaN(n, nstory); midedp_im = NaN(n-1, nstory);  
binedp_im = NaN(n-1, nstory); pDS_edp_im = cell(ncomp,1);  
pEDP_im = NaN(n-1,nstory); cEDP_im = NaN(n-1,nstory);  
qtyEL_im = NaN(n-1,nstory); cEL_im = NaN(n-1,nstory);  
compEL_im = NaN(n-1, nstory, ncomp); compqEL_im = NaN(nstory, ncomp);  
floorEL_im = NaN(nstory,1); p_fordemo = NaN(1, nstory);  
  
% Calculating loss for each component  
for i = 1:nstory  
    % Finding range of edp's of interest  
    edp_im(:,i) = logninv(p, log(edp(1,i)), edp(2,i));  
    binedp_im(:,i) = edp_im(2:end,i) - edp_im(1:end-1,i);  
    midedp_im(:,i) = edp_im(1:end-1,i) + binedp_im(:,i);  
    % Finding probability of that edp value for that story  
    pEDP_im(:,i) = lognpdf(midedp_im(:,i), log(edp(1,i)), edp(2,i)).*binedp_im(:,i);  
    cEDP_im(:,i) = logncdf(midedp_im(:,i), log(edp(1,i)), edp(2,i)).*binedp_im(:,i);  
    % Determining losses for each component  
    for j = 1:ncomp  
        pDS_edp_im{j} = NaN(n-1, frag{j}.nds, nstory);  
        % Determining probability of each damage state  
        for k = frag{j}.nds:-1:1  
            if k == frag{j}.nds  
                pDS_edp_im{j}(:,k,i) = logncdf(midedp_im(:,i), log(frag{j}.theta(k)),  
frag{j}.beta(k));  
            else  
                pDS_edp_im{j}(:,k,i) = logncdf(midedp_im(:,i), log(frag{j}.theta(k)),  
frag{j}.beta(k)) - pDS_edp_im{j}(:,k+1,i);  
            end  
        end  
        % Multiplying probability of each damage state by expected loss  
        compEL_im(:,i,j) = sum(repmat(frag{j}.ctheta,n-1,1).*pDS_edp_im{j}(:, :, i), 2);  
        compqEL_im(i,j) = sum(compEL_im(:,i,j)*cqty(i,j),1);  
    end  
    % Multiplying component loss by quantity at that floor  
    qtyEL_im(:,i) = sum(squeeze(compEL_im(:,i,:)).*repmat(cqty(i,:),n-1,1),2);  
    % Multiplying loss by probability of seeing that edp  
    cEL_im(:,i) = qtyEL_im(:,i).*pEDP_im(:,i);  
    % Calculating probability for demolition  
    p_fordemo(i) = sum(repmat(cqty(i,1),n-1,1).*pDS_edp_im{1}(:,1,i).*pEDP_im(:,i),1);  
    % Summing expected losses for all floors  
    floorEL_im(i) = sum(cEL_im(:,i));  
end  
  
% Summing expected losses across entire building  
bldgEL_im = sum(floorEL_im);  
  
end
```

## **calc\_lossCurve** -- calculates losses from repair, demolition, and collapse

```
function [EL, P, bldgEL, floorEL, compEL, im_ofint, IM, AAL] = calc_lossCurve(frag, imval,
stripeDat, nEDP, nfloors, pcoeff, EC, cqty, dqty, dfrag)

%% Collapse Case

% Getting collapse fragility
ngms = size(stripeDat{1},2);
[im_ofint, ~, ~, fittedFC, ~, ~] = get_collapseRisk(pcoeff, imval, stripeDat, fitttype, ngms);
n = numel(im_ofint);

% Calculating expected loss for each IM
P.C = fittedFC;
EL.C = P.C*EC.C;
P.NC = 1 - P.C;

%% Demolition Case

% Initialization
dbldgEL = NaN(n, 1);
dfloorEL = NaN(nEDP,n);
pd = NaN(n,1);

% Deaggregation curve of p(D)
pIMdiff = NaN(n,1);
for i = 1:n-1
    lnx = log(im_ofint(i));
    p = fliplr(pcoeff);
    pIMdiff(i) =
abs((p(2)+2*p(3)*lnx+3*p(4)*lnx^2+4*p(5)*lnx^3)/im_ofint(i))*exp(p(1)+p(2)*lnx+p(3)*lnx^2+p(4)*ln
x^3+p(5)*lnx^4));
end

% Getting median and dispersion values of EDPs
[thetaPoint, betaPoint] = calc_edpParam(stripeDat, imval, im_ofint, nEDP);

% Looping over all im requested
for i = 1:n
    edp = [thetaPoint(:,i)'; betaPoint(:,i)'];
    [bldgEL_im, floorEL_im, ~, ~, ~, p_fordemo] = calc_lossIntensity(edp, dfrag, dqty, n);
    dbldgEL(i) = bldgEL_im;
    dfloorEL(:,i) = floorEL_im;
    % Calculating total probability of demolition (given no collapse)
    pd(i) = nanmax(p_fordemo,[],2); % Assume max pd across all floors is p(D|IM,NC)
end

% Calculating expected loss for each IM given NC for R
P.D = (pd).*(1-P.C);
EL.D = P.D*EC.D;

%% Non-collapse Case

% Initialization
bldgEL = NaN(n, 1);
floorEL = NaN(nEDP,n);
compEL = NaN(nEDP,numel(frag),n);

% Looping over all im requested
for i = 1:n
    edp = [thetaPoint(:,i)'; betaPoint(:,i)'];
    [bldgEL_im, floorEL_im, compEL_im] = calc_lossIntensity(edp, frag, cqty, n);
    bldgEL(i) = bldgEL_im;
    floorEL(:,i) = floorEL_im;
    compEL(:, :, i) = compEL_im;
end
```

```

% Calculating expected loss for each IM given NC for R
P.R = P.NC - P.D;
EL.R = bldgEL.*P.R;

%% Combination

EL.T = EL.D + EL.C + EL.R;
P.T = P.D + P.C + P.R;

%% Plotting
figure;
subplot(2,2,1);
plot(im_ofint, [EL.R EL.D EL.C EL.T], 'LineWidth', 1.2);
grid on;
legend('R','D','C','T','Location','best');
xlim([0 max(im_ofint)]);
ylabel('E[L_i|IM]'); xlabel('IM');
title('Expected Loss Due to Each Case');

subplot(2,2,2);
plot(im_ofint, [P.R P.D P.C P.T], 'LineWidth', 1.2);
grid on;
legend('R','D','C','T','Location','best');
xlim([0 max(im_ofint)]);
ylabel('p[i|IM]'); xlabel('IM');
title('Probability of Each Case');

subplot(2,2,3);
plot(im_ofint, [EL.R EL.D EL.C EL.T]./(repmat(EL.T,1,4)), 'LineWidth', 1.2);
grid on;
legend('R','D','C','T','Location','best');
xlim([0 max(im_ofint)]);
ylabel('E[L_i|IM]/E[L_T|IM]'); xlabel('IM');
title('Contribution of Each Case');

subplot(2,2,4);
plot(im_ofint, [EL.R EL.D EL.C EL.T]./EC.RCN, 'LineWidth', 1.2);
grid on;
legend('R','D','C','T','Location','best');
xlim([0 max(im_ofint)]);
ylabel('E[L_i|IM]/RCN'); xlabel('IM');
title('Expected Loss as Percentage of RCN');

figure;
plot(im_ofint, [EL.R EL.D EL.C EL.T].*repmat(pIMdiff,1,4), 'Linewidth', 1.2); grid on;
legend('R','D','C','T','Location','best');
xlabel('IM'); ylabel('E[L_i|IM]');
title('Deaggregation of Expected Loss');

%%
% Display AAL results

xval = im_ofint;
yval = [EL.R EL.D EL.C EL.T].*repmat(pIMdiff,1,4);
for i = 1
    indsnan = isnan(xval(:,i)) | isnan(yval(:,i));
    xval(indsnan,:) = [];
    yval(indsnan,:) = [];
end

disp('AAL for each Case (R, D, C, T)');
format bank;
AAL.val = trapz(xval, yval);

```

```

disp(AAL.val);
disp('AAL for each Case % of Total');
format short;
AAL.perc = trapz(xval,yval)./trapz(xval(:,end),yval(:,end));
disp(AAL.perc);

IM = im_ofint;
end

```

In addition to the functions above, there is a function to set new default figure properties. It is shown below.

### *defaultFigureProperties* -- adjust default figure properties

```

function defaultFigureProperties

set(0,'defaultFigureColor',[1 1 1]);
set(0,'DefaultAxesFontSize',11)
set(0,'DefaultTextColor',[0, 0, 0]);
set(0,'DefaultAxesColorOrder',[
    0      0.4470    0.7410
    0.8500    0.3250    0.0980
    0.9290    0.6940    0.1250
    0.4940    0.1840    0.5560
    0.4660    0.6740    0.1880
    0.3010    0.7450    0.9330
    0.6350    0.0780    0.1840]);
set(0,'defaultAxesLineStyleOrder','-|--|:');

set(0,'DefaultLineLineSmoothing','off')
set(0,'DefaultPatchLineSmoothing','off')
end

```

The GUI code is over a thousand lines long and will not be included in this report. Instead, it will be submitted online.

## Planned Updates

By the time of the presentation, some diagnostic tools are planned in order to better and more easily assess the contribution of different factors to the expected loss. For example, pie charts that breakdown the contribution of case (repair vs. demolition vs. collapse), EDP (drift-sensitive vs. acceleration-sensitive), performance group, etc. may be beneficial to the user in order to identify significant contributors to loss. In addition, it may be helpful to explicitly identify values for common hazard levels (e.g. MCE or DBE). An ideal format for these outputs would be a PDF that neatly summarizes useful quantities and plots.

Other useful updates include improved error handling and user interface feedback. Try-catch blocks could be used to identify incorrect or missing inputs and better inform the user of those. Also, more effort could be made to ensure that the user is aware of the order of inputs (e.g. hazard input is required before collapse calculations).

The `calc_lossCurve` function can also be made more efficient by calculating values for all IM points at once. At the moment, it calculates values for each IM in a for-loop. This causes countable computation time and will be adjusted by the time of the presentation.

Lastly, better code organization and commenting will be included so that the tool is accessible and editable for users.

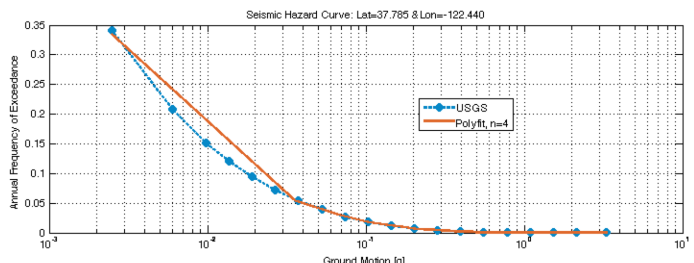
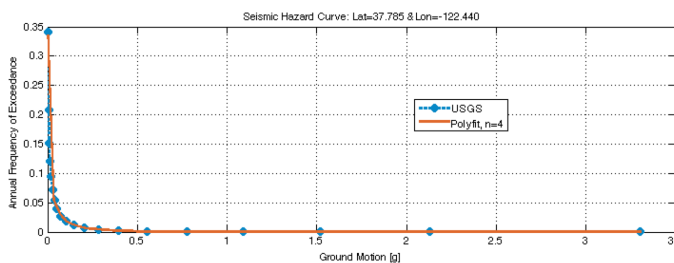
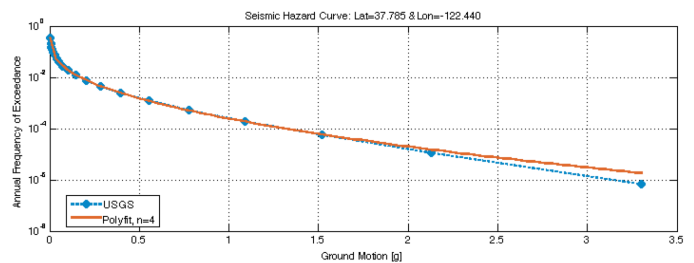
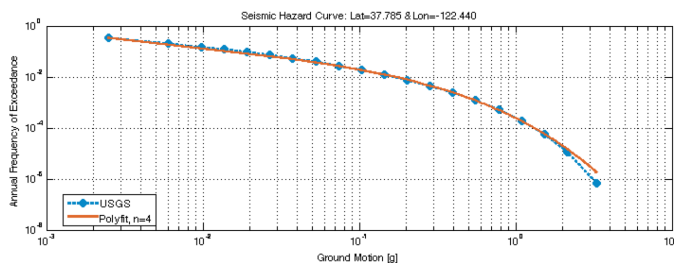
All updated code will be submitted again at the time of the presentation.

## Results of Example Building

Throughout the report, plots have been shown for the same 4-story,  $T=2.0s$  building located in San Francisco, CA on Site Class D. Additional values and plots are shown here for reference.

### Hazard

The seismic hazard curves in different scales are shown below along with polynomial fit.



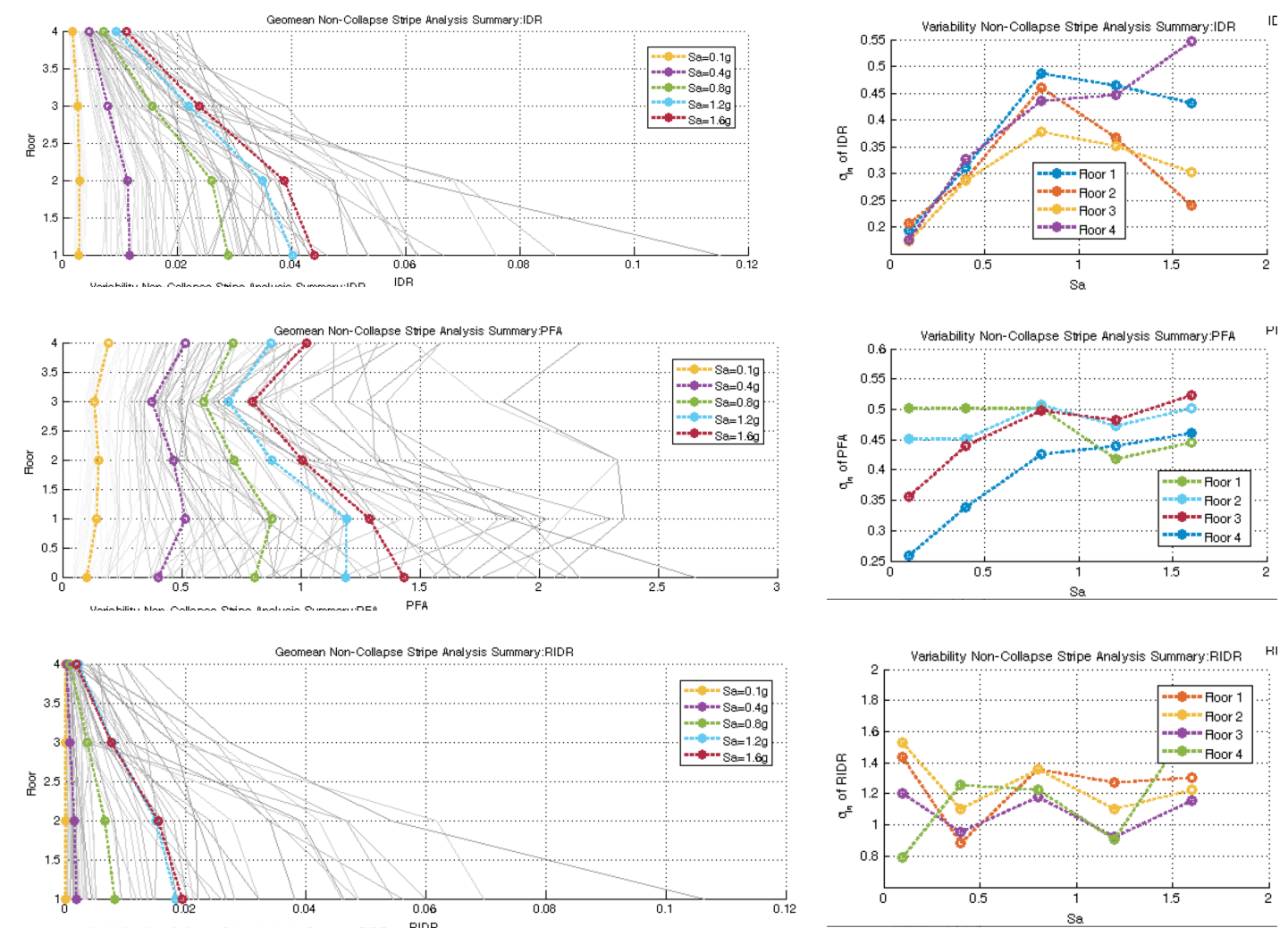
A summary of the spectral accelerations at different hazard levels are shown in the table below.

Probability of Exc.	50% in 50 years	10% in 50 years	2% in 50 years
$S_a(T_1)$	0.1288g	0.4283g	0.8530g

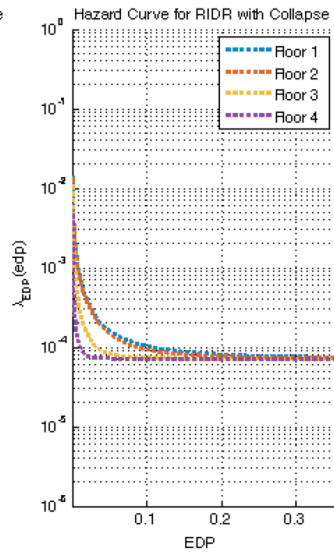
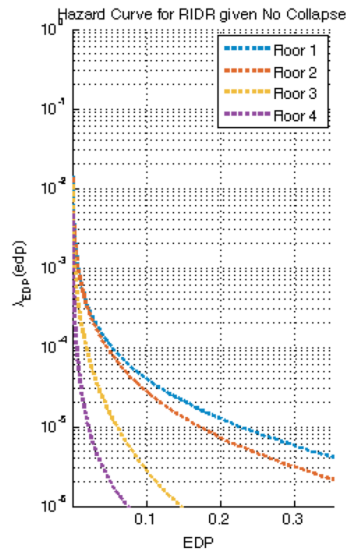
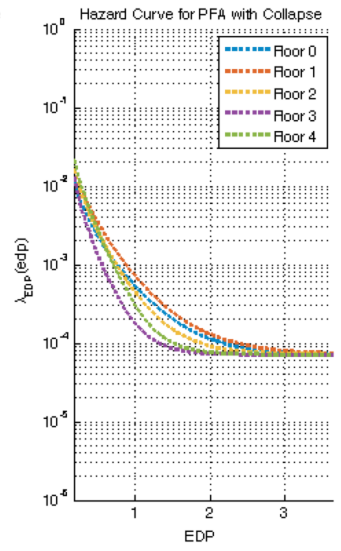
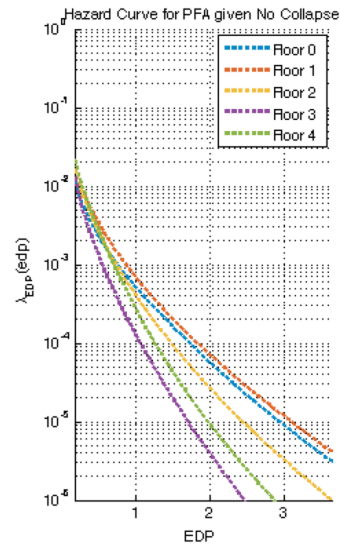
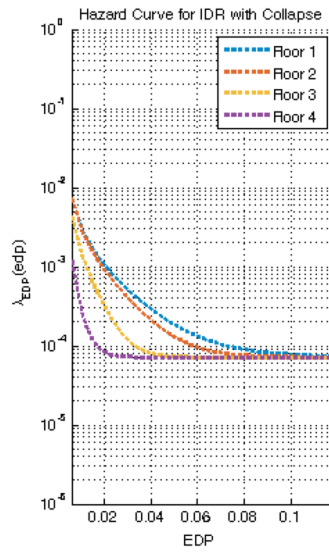
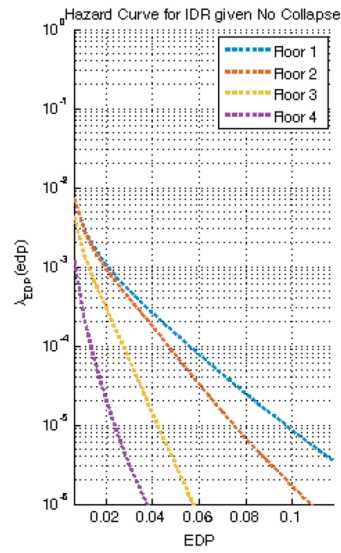
Response

The sample stripe analyses provided on Coursework were used for these. That data, along with the seismic hazard curve, produce to results shown below.

Summary of stripe analysis central results and variabilities.

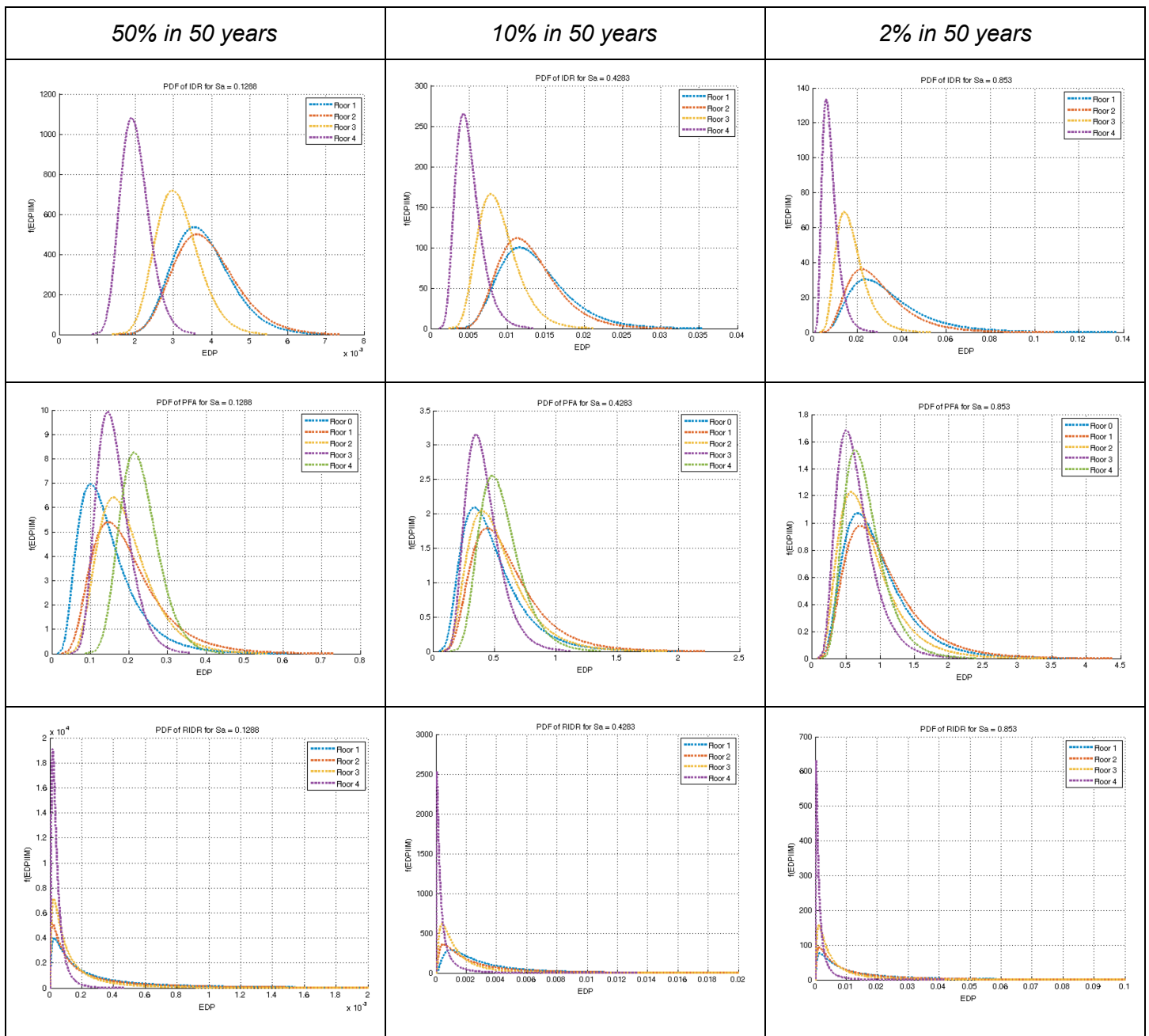


## EDP hazard curves.



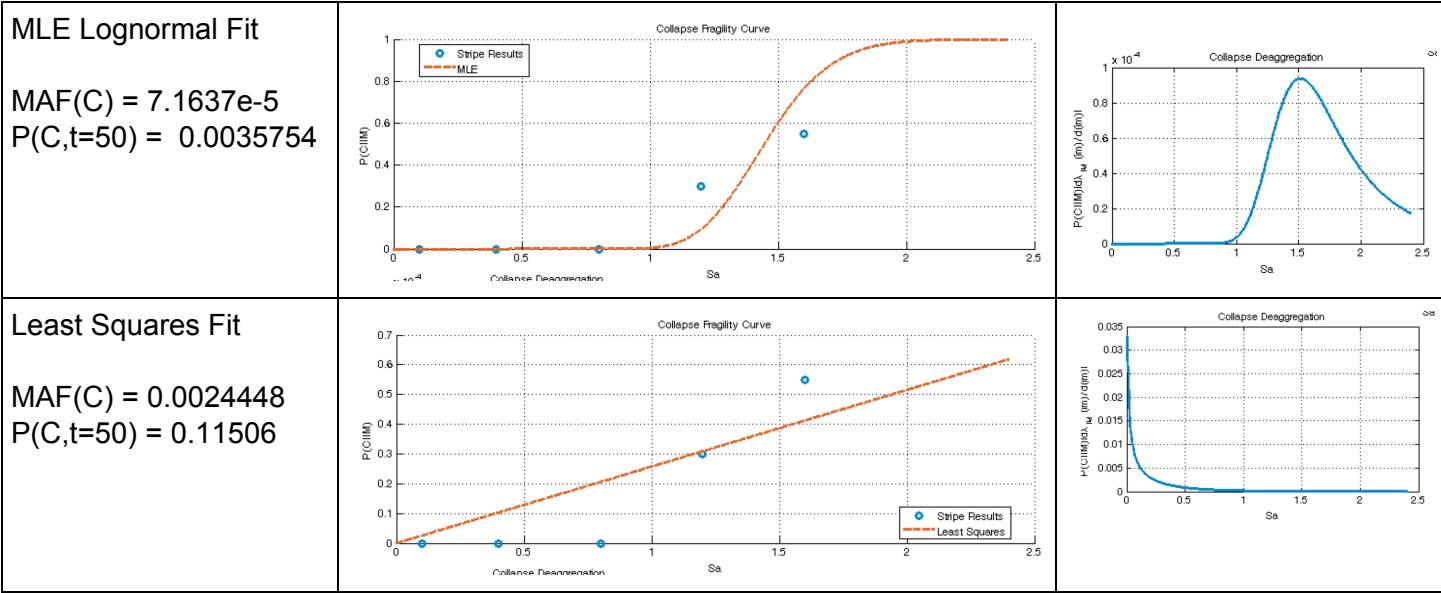


PDF of each EDP at the 50%50, 10%50, and 2%50 yr exceedance levels.

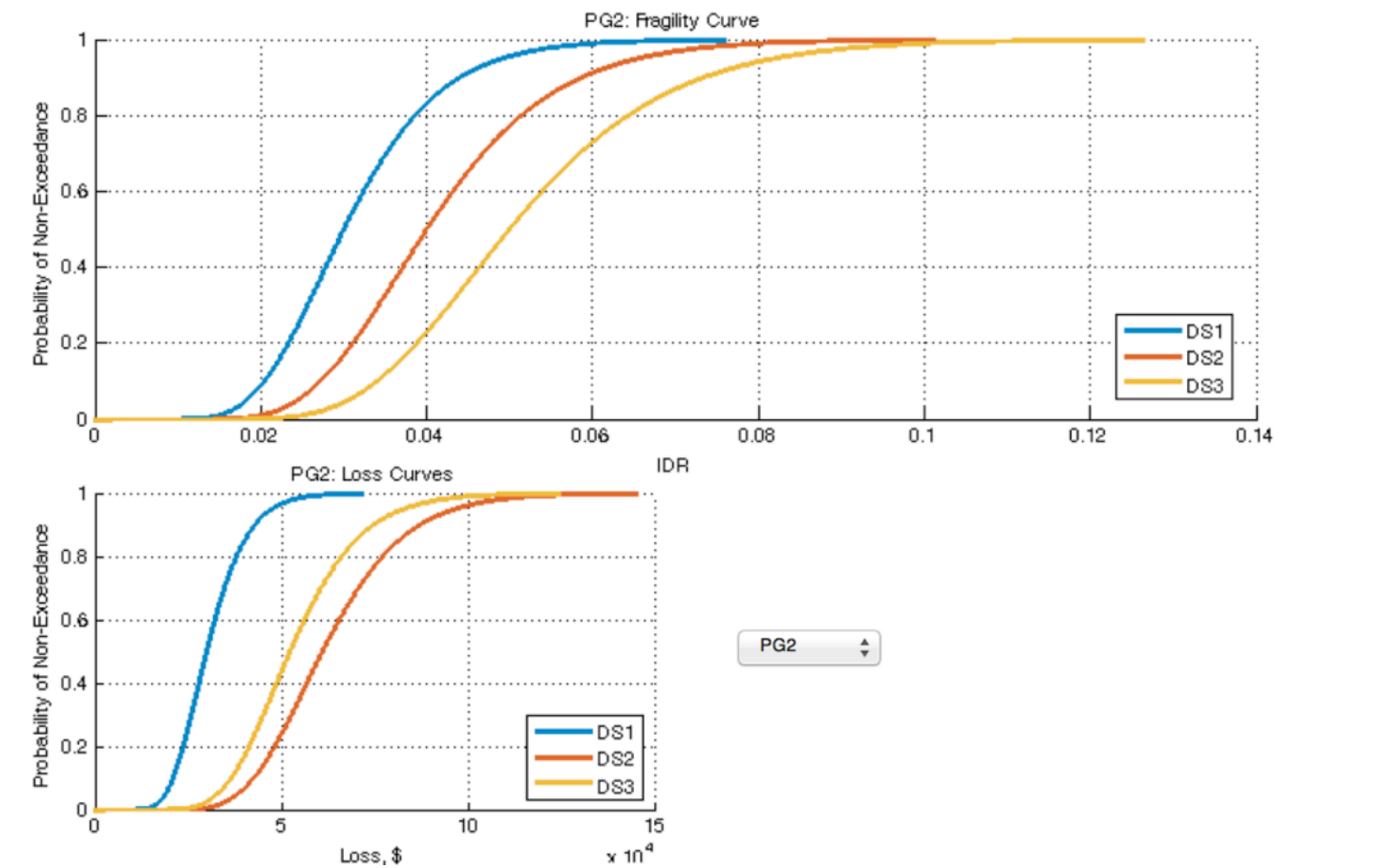


Note the x-axes shifts with increasing IM. The modes of the PDF's shift significantly to the right with increasing IM. In addition, the RIDR (particularly for floor 4) appears to be largely at near-zero values. This will affect the demolition results.

Collapse

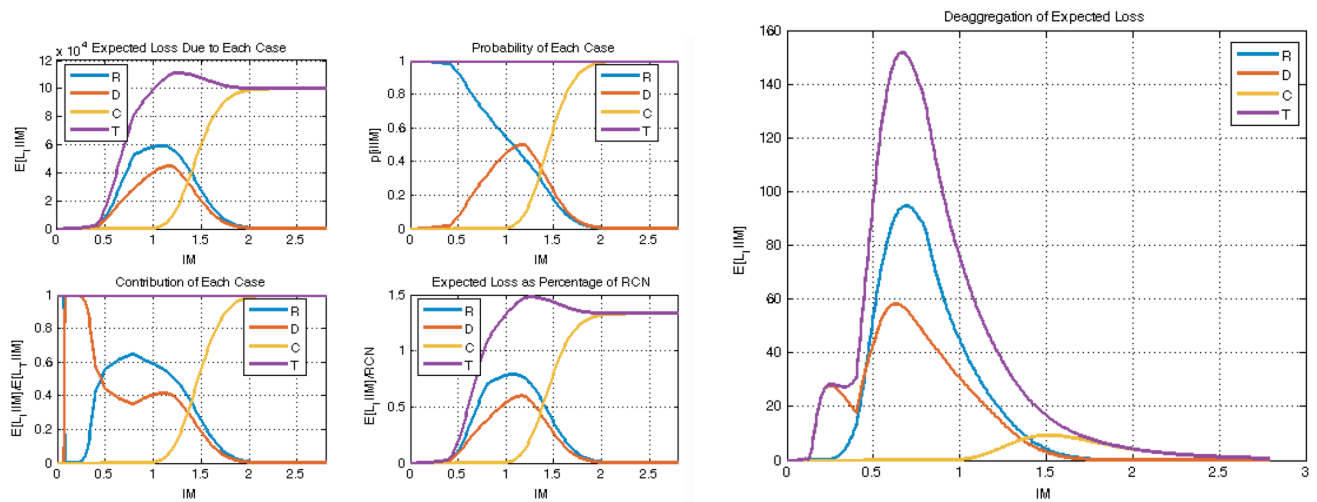


Damage



## Loss

Overall losses and the contributions of each case as a function of IM are shown below.



Annual expected loss = \$100.56

due to Repairs = \$52.40

due to Demolition = \$40.54

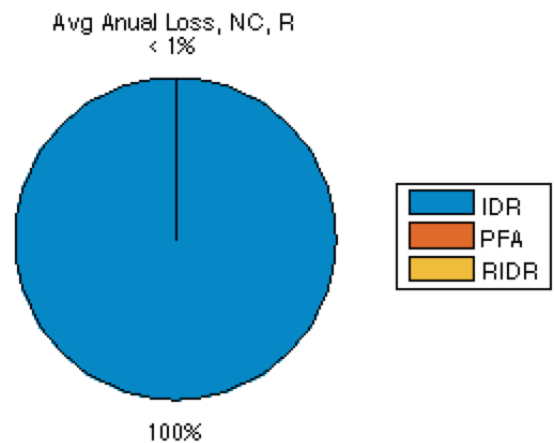
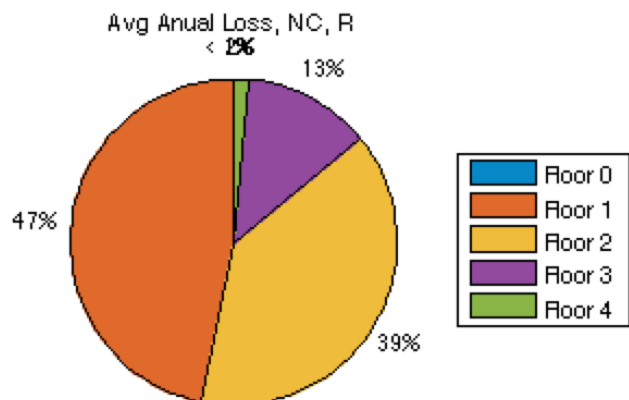
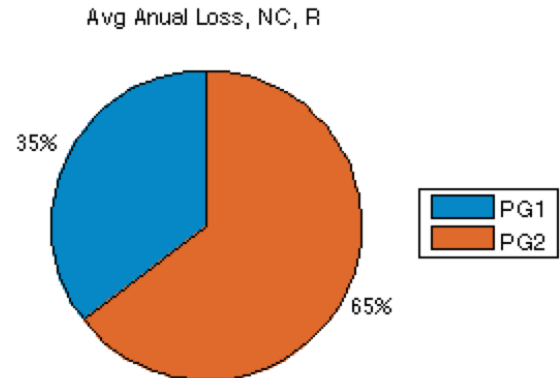
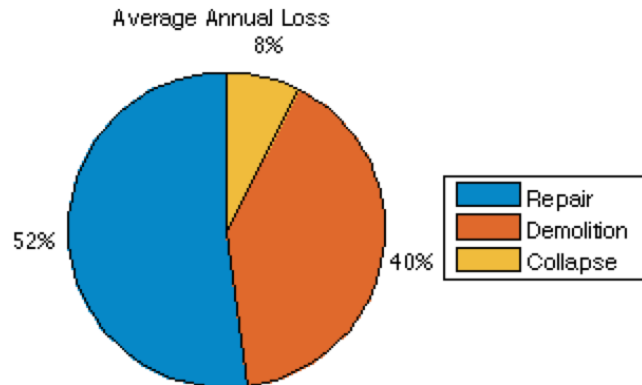
due to Collapse = \$7.62

Percentage of total annual expected loss

Repairs = 52.11%

Demolition = 40.31%

Collapse = 7.58%



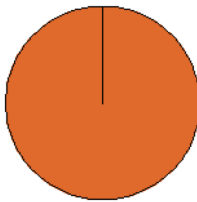
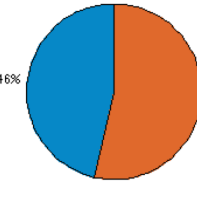
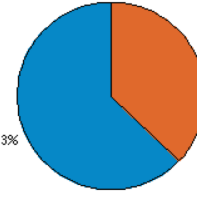
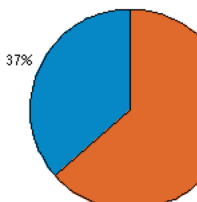
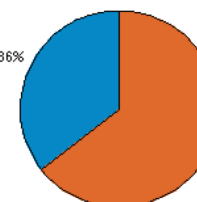
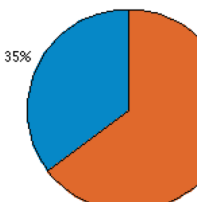
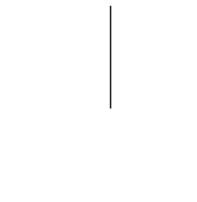
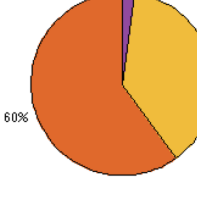
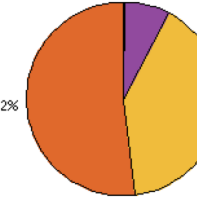

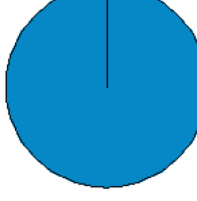
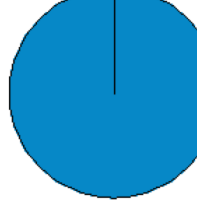
Expected values and contributions of each case at each hazard level.

<i>Probability of Exc.</i>	<i>50% in 50 years</i>	<i>10% in 50 years</i>	<i>2% in 50 years</i>
$E[L_T IM]$	\$7.14	\$4851.51	\$86,800.31
$E[L_R IM]$	\$0.00	\$2248.58	\$54,623.56
$E[L_D IM]$	\$7.14	\$2602.47	\$32,166.17
$E[L_C IM]$	\$0.00	\$0.00	\$9.87

<i>Probability of Exc.</i>	<i>50% in 50 years</i>	<i>10% in 50 years</i>	<i>2% in 50 years</i>
$E[L_R IM] / E[L_T IM]$	0%	46.3%	62.9%
$E[L_D IM] / E[L_T IM]$	100%	53.6%	37.1%
$E[L_C IM] / E[L_T IM]$	0%	0%	<0.1%

From the above tables, we see how demolition governs at frequent events, repair at a design level and MCE event. As seen in the curves, collapse dominates at much larger intensities (>1g).

Below is the contribution of each case, performance group, floor, and EDP to the  $E[L|IM]$  at the same hazard levels. Note that since only IDR-sensitive performance groups were included, there is no contribution of PFA or RIDR to the  $E[L|IM, NC, R]$ . Also, since demolition governs  $E[L|IM]$  so significantly for the 50%50 case, the floor and EDP breakdown are essentially zero.

Probability of Exc.	50% in 50 years	10% in 50 years	2% in 50 years
<b>Case</b>	$E[L, IM=0.1288]$ $< 1\%$  100%	$E[L, IM=0.4283]$ $< 1\%$  46% 54%	$E[L, IM=0.853]$ $< 1\%$  63% 37%
<b>Component</b>	$E[L, IM=0.1288, NC, R]$  37% 63%	$E[L, IM=0.4283, NC, R]$  36% 64%	$E[L, IM=0.853, NC, R]$  35% 65%
<b>Floor</b>	$E[L, IM=0.1288, NC, R]$ $< 1\%$  0%	$E[L, IM=0.4283, NC, R]$ $< 1\%$  60% 38% 2%	$E[L, IM=0.853, NC, R]$ $< 1\%$  52% 40% 8%
<b>EDP</b>	$E[L, IM=0.1288, NC, R]$ $< 1\%$  0%	$E[L, IM=0.4283, NC, R]$ $< 1\%$  100%	$E[L, IM=0.853, NC, R]$ $< 1\%$  100%