ANEXO A

CÓDIGOS MATLAB

En la Figura A.1 se presenta el código MATLAB que se ha utilizado para calcular el periodo de retorno de cada intensidad sísmica a partir de las curvas de peligrosidad sísmica para la ciudad de Barcelona.

```
%%  Definición del escenario
f_name_0='Peligrosidad Sísmica.xlsx';
fid_0=fopen(f_name_0,'r');
Intensidad=5:0.5:7;
[peligro_1,txt_1,raw1]=xlsread(f_name_0,'Hoja1','A1:B6');
[peligro_2,txt_2,raw2]=xlsread(f_name_0,'Hoja1','D1:E6');
[peligro_3,txt_3,raw3]=xlsread(f_name_0,'Hoja1','G1:H6');
escenario_1=interp1(peligro_1(:,1),log(peligro_1(:,2)),Intensidad,...
        'linear','extrap');
escenario_2=interp1(peligro_2(:,1),log(peligro_2(:,2)),Intensidad,...
        'linear','extrap');
escenario_3=interp1(peligro_3(:,1),log(peligro_3(:,2)),Intensidad,...
        'linear','extrap');
escenario=[escenario_1
           escenario_2
           escenario_3]';
ret_per=1./exp(escenario);
ann_rat=1./ret_per;

%% salidad de resultados
inten=['   V    '
       '  V-VI  '
       '   VI   '
       ' VI-VII '
       '  VII   '];
forma=' %s   %6.0f %6.0f %6.0f          %6.2e  %6.2e  %6.2e  \n';
fprintf('\n Escenarios para intensidades V, V-VI, VI, VI-VII y VII');
fprintf('\n
*****************************************************\n');
fprintf('   Inten           Ret Per                          Ann. Rate
\n');
fprintf('            mu-sigma  mu   mu+sigma       mu-sigma    mu
mu+sigma \n');
for i=1: length(Intensidad);
    fprintf(forma, inten(i,:), ret_per(i,:), ann_rat(i,:));
end
```

Figura A.1: Código MATLAB para calcular el periodo de retorno de cada intensidad sísmica a partir de las curvas de peligrosidad sísmica para la ciudad de Barcelona.

En la Figura A.2, que se muestra a continuación, se presenta el código MATLAB que ha sido utilizado para el cálculo de los grados de daño de cada edificio y las probabilidades de los estados de daño de cada edificio para el cálculo del coste.

```
clear
close all
clc
% Para una intensidad dada
% este programa calcula el grado de daño medio o grado de daño más
% probable teniendo en cuenta la incertidumbre
% en el índice de vulnerabilidad.

%% LECTURA DE DATOS
f_name_1='Curvas_Vulnerabilidad.xls';       % Vulnerabilidad
[vulne_tot,txt1,raw1]=xlsread(f_name_1,'Fiab_diez','A2:Y5503');

%% DEFINIMOS LAS CURVAS DE VULNERABILIDAD
%
v_limits=([vulne_tot(:,12) vulne_tot(:,13)]); % limites inf. y sup.
del Iv
best_alpha=vulne_tot(:,18);                   % Parámetro alpha
best_beta=vulne_tot(:,19);                    % Parámetro beta
% best_alpha=vulne_tot(:,22);
% best_beta=vulne_tot(:,23);
n_edif=length(vulne_tot(:,1));                % Número de edificios
d_f_1(1:n_edif)=0.0;
iv_m=d_f_1;
sigma=iv_m;
%% DEFINIMOS LA INTENSIDAD
I=7;
% I=6;
%I=7;
%% PROCESO ITERATIVO
%
delta_d=0.0001;          % paso fijo para la interpolación de la curva
x_d=0:delta_d:5;         % índice de daño medio a paso fijo ([0 5])
pasos=n_edif;            % número de iteraciones
hwb=waitbar(0,'PACIENCIA');
paso=0;
for i=1:n_edif;
    v_ini=v_limits(i,1);            % Iv inicial
    v_fin=v_limits(i,2);            % Iv final
    delta_v=(v_fin-v_ini)/999;      % paso fijo para el Iv
    x_v=v_ini:delta_v:v_fin;        % Abscisas del Iv a paso fijo
(1000 puntos)

%     %% FUNCIÓN BETA (definida en Aguilar (2011)
%     xnum=((x_v-v_ini).^(best_alpha(i)-1)).*((v_fin-
x_v).^(best_beta(i)-1));
%     xdenom=(v_fin-v_ini)^(best_alpha(i)+best_beta(i)-1);
%     p1_v=xnum/(xdenom*beta(best_alpha(i),best_beta(i))); % pdf del
Iv
%     p1p_v=cumtrapz(p_v)*delta_v;                        % cdf del Iv

    %% FUNCIÓN BETA ( Definida en MatLab)
    xx_v=(x_v-v_ini)/(v_fin-v_ini);  % abscisas escaladas entre 0 y 1)
    p2_v=betapdf(xx_v,best_alpha(i),best_beta(i)); % cpdf del Iv
    p2p_v=betacdf(xx_v,best_alpha(i),best_beta(i)); % cpdf del Iv
    %     figure(1)
    %     plot(x_v,p_v,'-b',x_v,pp_v,'-.r',x_v,ppp_v,'-g');

    %% VALOR MEDIO Y DESVIACION ESTANDAR
    iv_m(i)=v_ini+best_alpha(i)/(best_alpha(i)+best_beta(i))*(v_fin-
v_ini);
```

```
sigma(i)=(best_alpha(i)*best_beta(i)/((best_alpha(i)+best_beta(i))^2*(
best_alpha(i)+best_beta(i)+1)))*(v_fin-v_ini)^2;    %% calculo del
índice de daño
    sigma(i)=sqrt(sigma(i));
    %% DAÑO
    d=2.5*(1+tanh((I+6.25*x_v-13.1)/2.3));  % damage function
    pd=p2_v;                                % pdf of the damage function
(before normalization)
    %     figure(2)
    %     plot(x_v,d,'-r');
        %     pause
    %    calculo 1
    %    interpolamos d a paso fijo;
    y_d=interp1(d,pd,x_d,'linear','extrap');
    %     figure(3)
    %     plot(d,pd,'-r',x_d,y_d,'-.k');
    ydd=cumtrapz(y_d)*delta_d;      % Integration of the pdf function
    ydd=ydd/max(ydd);               % cdf of the damage function
(normalized)
    %     figure(4)
    %     plot(x_d,ydd,'-.k');
    % buscamos el valor medio
    %     p_med=0.5;
    %     d_med=interp1(ydd,x_d,p_med);
    percentil=0.16;  % valor median
    for jj=1:length(x_d)
        if ydd(jj)<=percentil;
            d_med=x_d(jj);
            p_med=ydd(jj);
        else
            break
        end
    end
    d_f_1(i)=d_med;
        paso=paso+1;
    waitbar(paso/pasos);
    clc
end
gis=d_f_1';
%cálculo probabilidades para cada grado de daño, DPM.
d(1:5502)=0.0;
dpm(1:5502,1:7)=0.0;
x=1:5;
xx=0:5;
for i=1:5502;
    d(i)=d_f_1(i);
    dpm(i,1:6)=binopdf(xx,5,d(i)/5);
    dpm(i,7)=sum(dpm(i,:));
end
delete(hwb);
fclose all;
clc
caso='Best'
Inensidad=I
percentil=percentil
Median_d=median(d_f_1)
Sigma=std(d_f_1)
```

Figura A.2: Código MATLAB para el cálculo de los grados de daño de cada edificio y las probabilidades de los estados de daño de cada edificio para el cálculo del coste.