NERA

Network of European Research Infrastructures for Earthquake Risk Assessment and Mitigation

Report

Methodology and open-source software for strong-motion data processing and estimation of ground-motion parameters

Activity:	*Networking accelerometric networks and SM data users*
Activity number:	*NA3, Task3.4*
Deliverable:	*Methodologies and/or open-source software for reliable estimation of metadata parameters and for strong motion data processing*
Deliverable number:	*D3.4*
Responsible activity leader:	*Sinan Akkar*
Responsible participant:	*KOERI*
Author:	*Sinan Akkar, Lucia Luzi, Aida Azari Sisi*

Seventh Framework Programme
EC project number: 262330

Summary

Strong-motion data processing is a key element in computing ground-motion intensity measures that are distributed by strong-motion databases. One of the major tasks in NERA-NA3 is to establish a prototype strong-motion database (Engineering Strong-Motion Database, ESM_db) for integrated European accelerometric data and this report summarizes the data-processing scheme implemented in ESM_db. The processing scheme explained in the report is based on an extensive study (Boore et al., 2012) that evaluates two data processing schemes implemented by ITACA (Italian Accelerometric Archive) and Pacific Earthquake Engineering Research Center (PEER). Both ITACA and PEER collect and distribute processed accelerograms for engineering needs and their data processing schemes include state-of-art knowledge to obtain reliable and consistent processed data. The open-source software and brief description of ESM_db are also given at the end of the report. The software is available via Orfeus seismological software library (http://www.orfeus-eu.org/software/seismo_softwarelibrary.html)

Table of Contents

I. Introduction

Currently, high-pass and low-pass filtering (or briefly band-pass filtering) is the most common technique used for processing the raw accelerograms recorded after an earthquake. The scientific studies have shown that zero-phase (acausal) filters are essential in the implementation of band-pass filtering as the computed peak ground-motion values as well as spectral quantities (e.g., spectral acceleration and displacement) are less sensitive to the filtering process (Boore and Akkar, 2003; Boore and Bommer, 2005). The zero-phase filters, if applied in time-domain, require zero padding before and after the actual ground motion in order to include the effects of filter transients that play a significant role while integrating the filtered accelerograms for ground velocity and displacement. The lengths of zero pads are proportional to high-pass (low-cut) filter cut-offs (Converse and Brady, 1992). The length of zero pads increases with decreasing low-cut filter value and sometimes it is longer than the actual duration of strong-motion data. In most cases, the end-users who use the processed accelerograms do not appreciate the significance of zero pads and they tend to remove them from the processed data in order to run their analysis with the actual portion of the record. However, removal of zero-pads distorts the entire data processing and results in inconsistent ground velocity and displacement as well as spectral ordinates with respect to original processing that keeps the zero padded and acausally filtered accelerograms (Boore and Bommer, 2005; Boore et al., 2012). It should be noted that zero padding is still required when zero-phase filters are applied in the frequency domain due to fast Fourier transformation techniques as they complete the raw accelerograms data to 2^n data points to convolve filter response function with the frequency components of the accelerograms.

ITACA and PEER data processing impose post-processing schemes after acausally filtering the strong-motion data and distribute the post-processed accelerograms with their original lengths. The post-processed accelerograms are shaped such that they intended to give very similar ground-motion intensity measures with respect to zero padded and acausally filtered accelerograms. Although such post-processing of accelerometric data is not ideal, it prevents the blind removal of zero pads by the end-user.

Following the above argument, ESM_db that is developed as part of NERA-NA3 activity to serve as a prototype database for integrating accelerometric data within and around Europe evaluated the performance of ITACA and PEER data processing to adopt the most convenient methodology as the standard strong-motion data processing scheme. This report describes the performance evaluation of ITACA and PEER data processing procedures and delivers the source of Matlab routines that are used in ESM_db for data processing. The software is available online via Orfeus seismological software library (http://www.orfeus-eu.org/software/seismo_softwarelibrary.html). The report also includes an appendix that shows the major features of ESM_db that currently consist of Turkish and Italian strong-motion data.

II. Evaluation of ITACA and PEER strong-motion data processing

Figures 1 and 2 show the ITACA and PEER strong-motion data processing algorithms. As indicated in the introductory section, both schemes apply a post

processing after acausally filtered accelerograms. The algorithms were developed from Pacor et al. (2011) for ITACA, based on Paolucci et al. (2011). The PEER procedure was implemented by written communications with people who developed and/or are using it (N. Abrahamson, R. Darragh, and W. Silva for PEER scheme). The post-processing steps in each scheme start after the acausal filtering and zero pad removal boxes in the algorithms. The basic difference between the ITACA and the PEER procedures is that the former does linear detrending of the velocity and displacements obtained from the zero pads removed data, whereas PEER scheme fits a sixth-order polynomial (in most cases) to the displacements. ITACA then obtains a corrected acceleration time series by double differentiation of the detrended displacement time series, while PEER NGA subtracts the analytical second derivative of the fitted polynomial from the acceleration time series. It should be noted that the choice of high-pass and low-pass filter cut-offs for acausal filtering is not explained in the report. The suggested rules for the choice filter cut-offs can be found in many relevant studies (e.g., Akkar and Bommer, 2006; Boore and Bommer, 2005; Douglas and Boore, 2010).

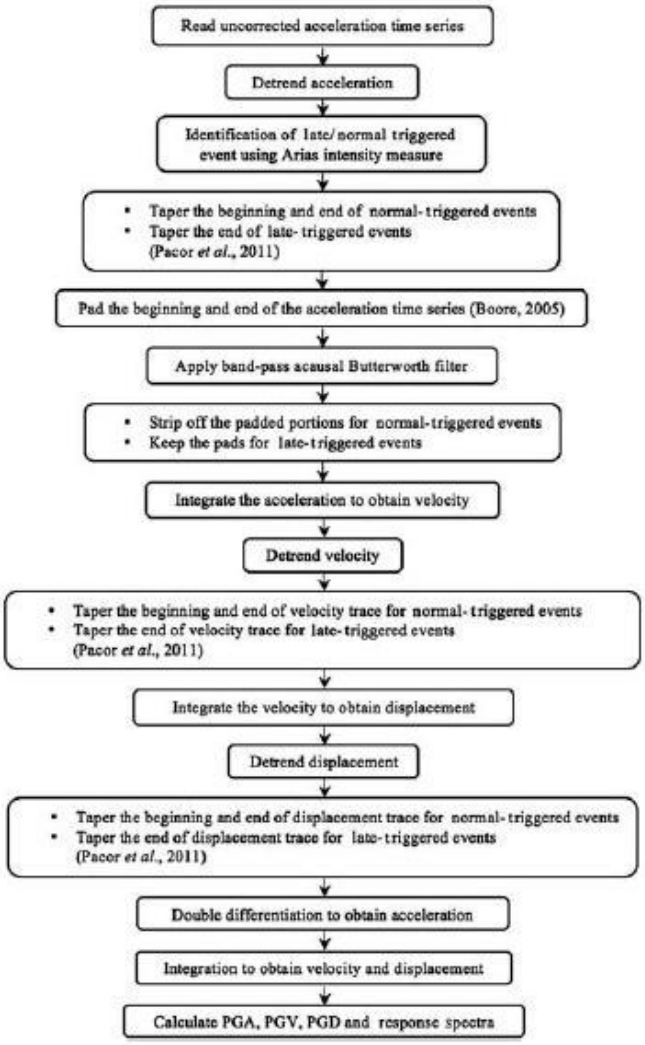

Figure 1. ITACA data processing scheme used in testing of post-processing of acausally filtered accelerograms (modified from Boore et al., 2012)

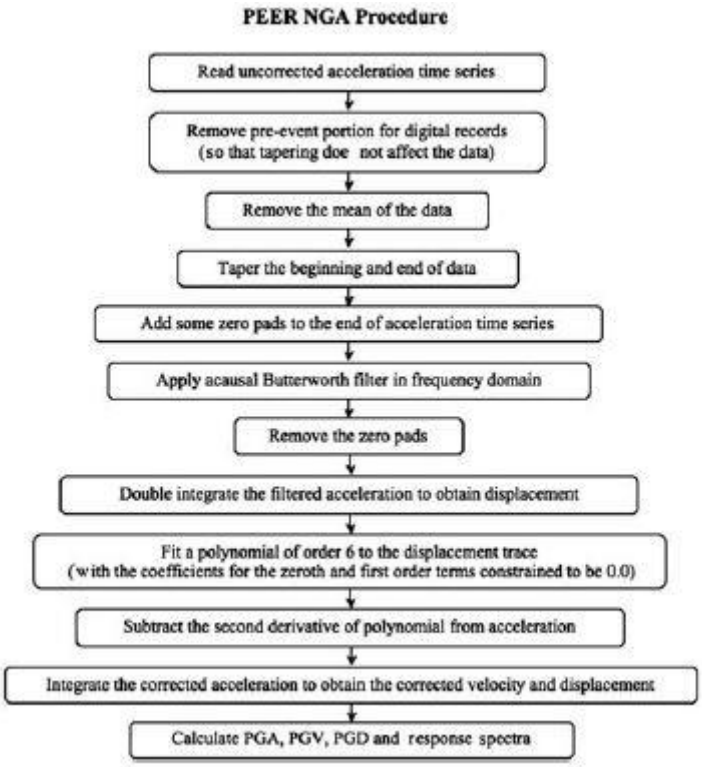

Figure 2. PEER data processing scheme used in testing of post-processing of acausally filtered accelerograms (modified from Boore et al., 2012)

The accelerometric data from the recently compiled Turkish strong-motion database (Akkar et al., 2010; Sandıkkaya et al., 2010) were used to evaluate the ITACA and PEER post-processing schemes. These data are almost entirely digital data for earthquakes from M 2.8 to 7.6, recorded at distances of 0–200 km. A total of 2714 horizontal component acceleration time series and 1357 vertical component acceleration time series were used. Only 28 analog three-component recordings were included in the evaluations. Most of the time series are sampled at 200 points per second. None of the records were classified as late triggers, as defined by the ITACA scheme. Each record was processed using the procedure given in Figure 3 that only implements acausal filtering without removing the zero pads. The filter cut-offs of these records were already identified on a record-by-record basis as part of the uniform processing of the Turkish database (Akkar et al., 2010). The zero pad lengths were taken from Converse and Brady (1992). The processed Turkish data in this manner yielded the reference measures of ground-motion intensity measures, including PGA, PGV, PGD, and 5%-damped pseudo–absolute response spectral acceleration (PSA). The row acceleration time series were then processed using the ITACA and the PEER procedures.

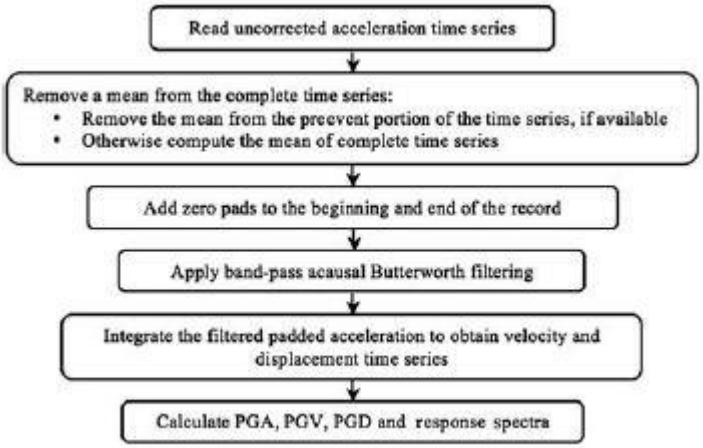

Figure 3. Reference data processing procedure used in the evaluation of PEER and ITACA post-processing evaluation (modified from Boore et al., 2012)

Comparisons of ITACA and PEER processing are provided in Figures 4 and 5 by computing the ratios of PGA, PGV, PGD, and 5%-damped PSA computed by ITACA and PEER procedures to the reference measures. For each figure the ITACA/Reference results are given in the left column and the PEER NGA/Reference results are given in the right column. The ratios are plotted against spectral periods in the top row and against spectral periods divided by the low-cut filter period used in the processing of each record in the bottom row of graphs. No evaluation was done with respect to low-pass (high-cut) filter periods as their effects are minimal on the processed ground-motion intensity measures even at very short spectral periods (Douglas and Boore, 2010; Akkar et al., 2011).

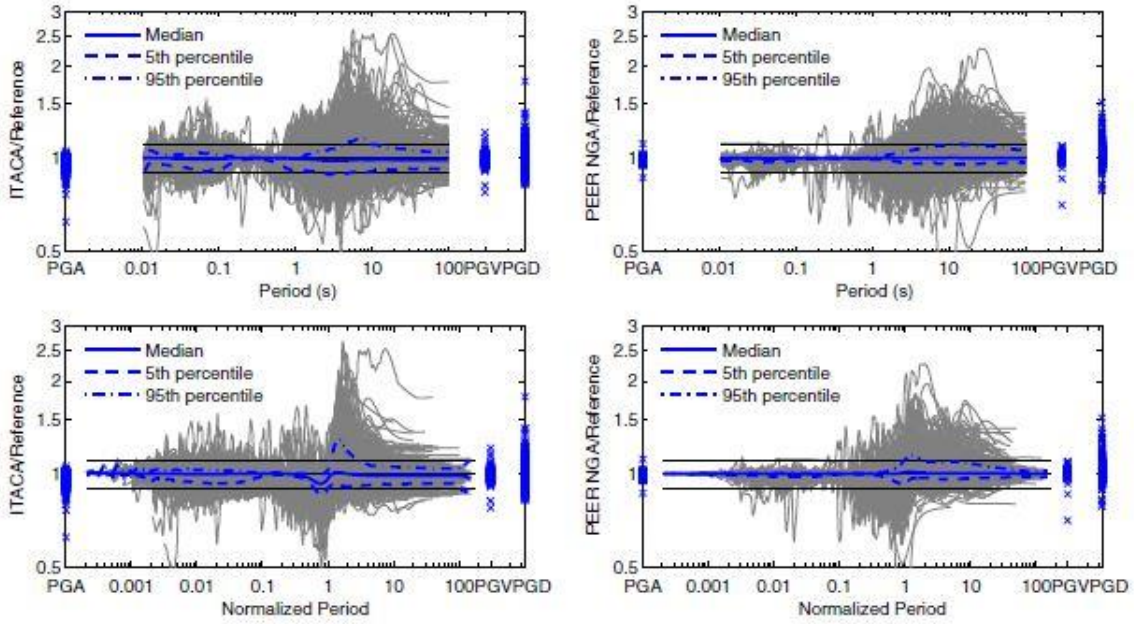

Figure 4. Performances of ITACA (first column) and PEER (second column) processing schemes with respect to reference processing for horizontal component intensity measures (modified from Boore et al., 2012)

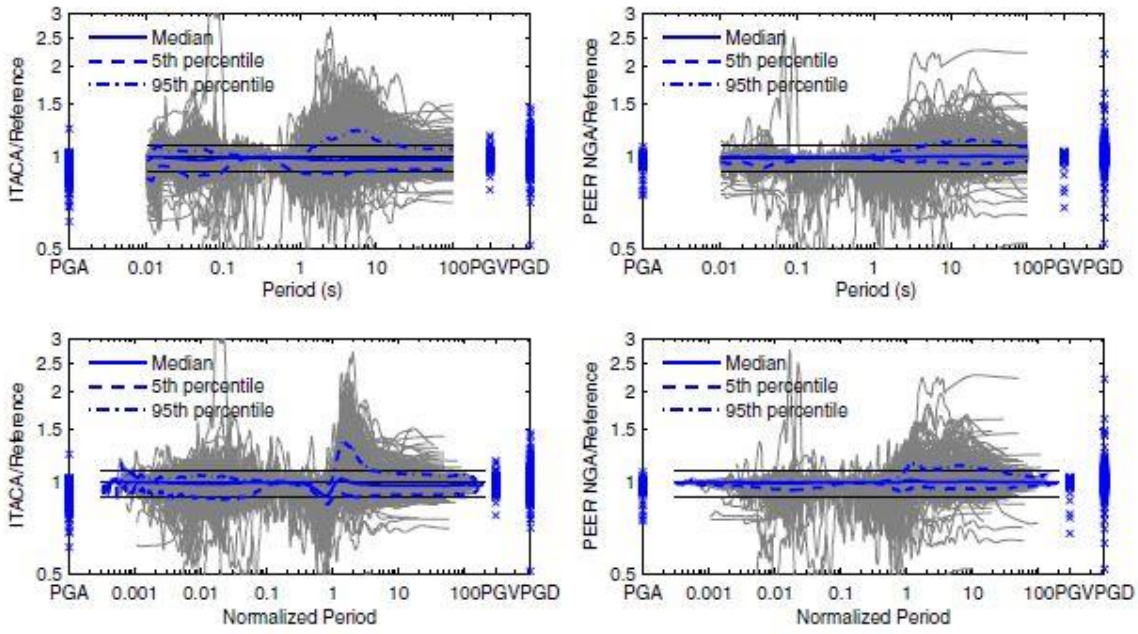

Figure 5. Performances of ITACA (first column) and PEER (second column) processing schemes with respect to reference processing for vertical component intensity measures (modified from Boore et al., 2012)

The fifth, fiftieth, and ninety-fifth percentile curves are included in each graph in Figures 4 and 5 to see better where the bulk of the ratios lie. These curves show that ratios are quite close to unity but with some period dependence. The ratios become particularly close to unity for periods between about 0.1 and 1 s. When the normalized period increases beyond about 0.5, the fluctuations in ratios increases, emphasizing the fact that the recordings should only be used up to a fraction of their low-cut filter cut-off values (Akkar and Bommer, 2006). The ratios are somewhat closer to unity for the PEER NGA processing than the ITACA processing. The outliers in Figures 4 and 5 are shown separately in Figure 6 as a histogram. The outlier data are associated with very low PGA values as can be inferred from this plot. Most of the analog accelerograms fall into this group.

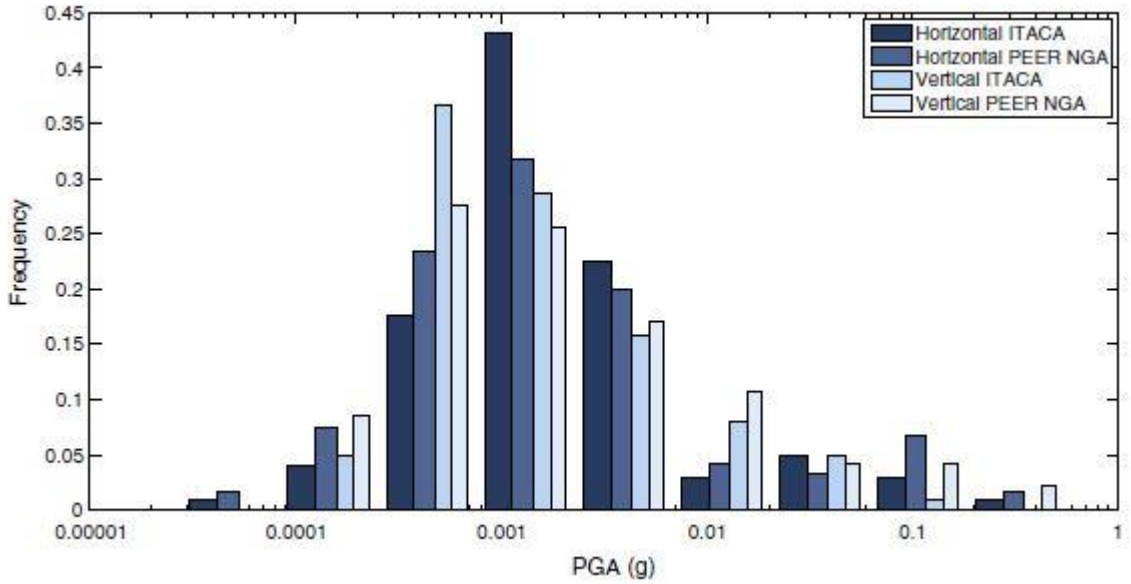

Figure 6. Distribution of outliers given in Figures 4 and 5 as histogram plots (modified from Boore et al., 2012)

Figures 7 and 8 plot histograms of the ratios at some selected normalized spectral periods. One can infer from these figures that the vertical components are more sensitive to the verified post-processing procedures. The distribution of the ITACA ratios being more skewed and somewhat larger than those from the PEER processing. Overall, however, the bulk of the ratios are close to unity for both procedures.

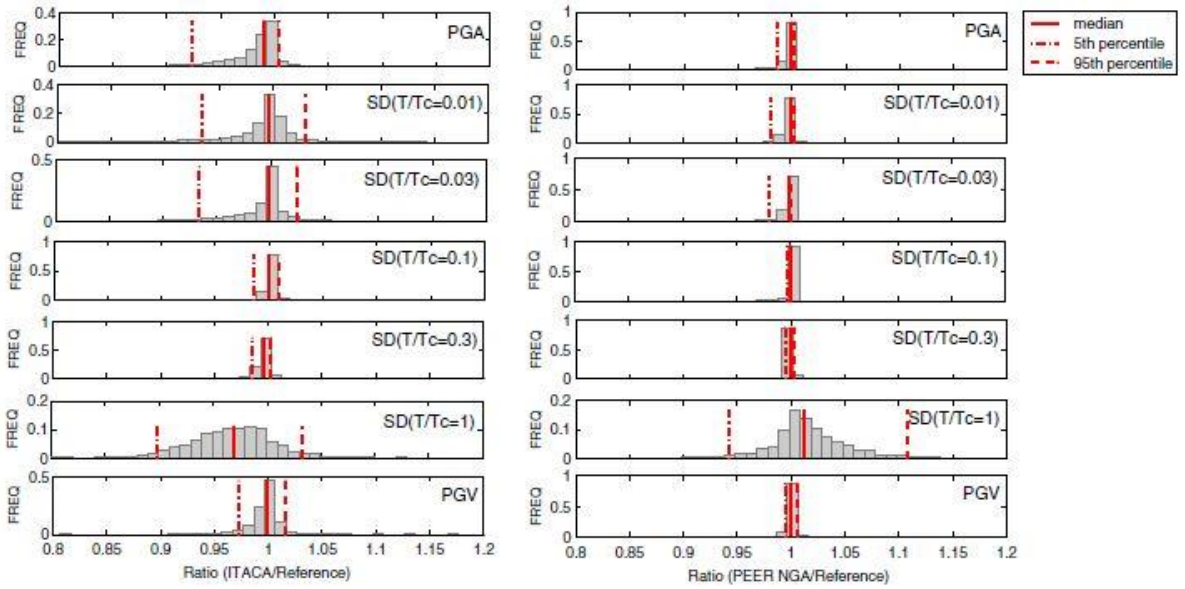

Figure 7. Statistics of some selected normalized periods for ITACA (first column) and PEER (second column) post-processing schemes for horizontal components (modified from Boore et al., 2012)

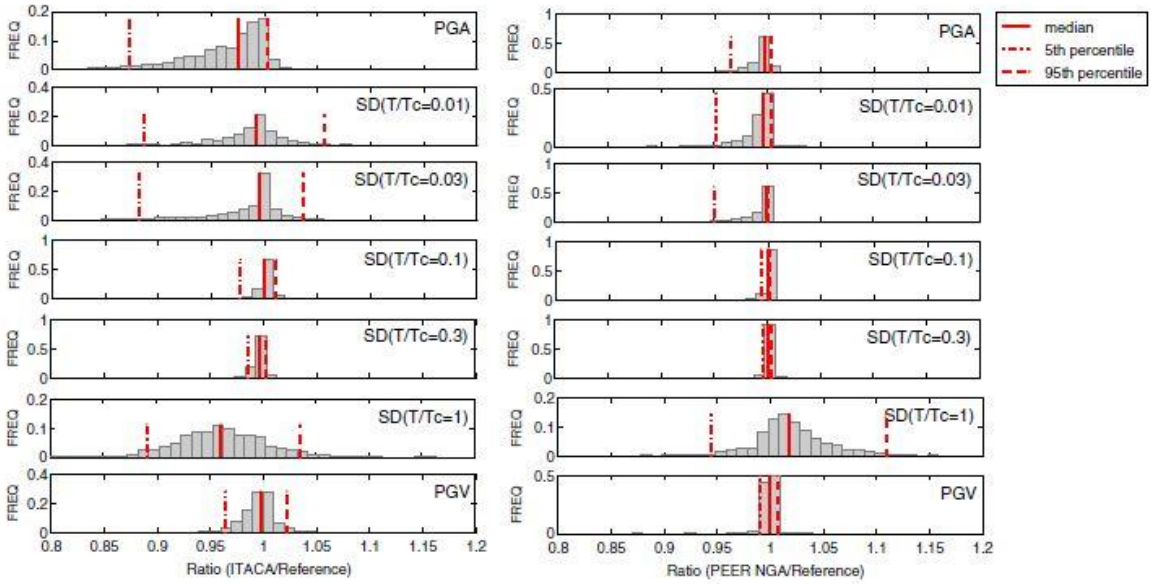

Figure 8. Statistics of some selected normalized periods for ITACA (first column) and PEER (second column) post-processing schemes for vertical components (modified from Boore et al., 2012)

III. Main Conclusions

Acausal (zero-phase) filtering is the state-of-art in strong-motion data processing. It requires zero pads to accommodate filter response transients. Removal of zero pads will distort the specific features of acausal filtering and will result in incompatible ground-motion intensity measure with respect to zero-padded and acausally filtered data. The ideal situation is to keep the zero pads as well as the filter transients of acausally filtered accelerograms. This crucial point is generally overlooked by the end users. In order to prevent the incompatibility in the computed ground-motion intensity measures by blindly removing zero pads, some data providers have developed some practical post-processing procedures. These procedures remove the zero pads of acausally filtered accelerograms in a controlled manner and compute the intensity measures as well as velocity and displacement ground motions that are close to original processing. This type of post-processing is not ideal but in many cases implemented by data providers in order to minimize the distortions in acausally filtered and zero-padded accelerograms.

The ITACA and PEER post-processing schemes evaluated in the report show similar performances but the PEER procedure yields slightly better results for the reference Turkish strong-motion database used in the evaluation process. Thus, the Engineering Strong-Motion database (ESM_db) established as a prototype under NERA-NA3 use the PEER data processing as the standard tool for removing the long-period and short-period noise from the raw accelerometric data. The Matlab codes developed under this task are included in Appendix A. The basic content of ESM_db is given in Appendix B.

References

Akkar, S., and J. J. Bommer (2006). Influence of long-period filter cut-off on elastic spectral displacements, Earthq. Eng. Struct. Dynam. 35, 1145–1165.

Akkar, S., Z. Çağnan, E. Yenier, Ö. Erdoğan, M. A. Sandıkkaya, and P. Gülkan (2010). The recently compiled Turkish strong-motion database: Preliminary investigation for seismological parameters, J. Seismol. 14, 457–479.

Akkar, S., O. Kale, E. Yenier and J.J. Bommer (2011). The high-frequency limit of usable response spectral ordinates from filtered analogue and digital strong-motion accelerograms, Earthq. Eng. Struct. Dynam. 40, 1387-1401.

Boore, D. M., and S. Akkar (2003). Effect of causal and acausal filters on elastic and inelastic response spectra, Earthq. Eng. Struct. Dynam. 32, 1729–1748.

Boore, D. M., and J. J. Bommer (2005). Processing of strong-motion accelerograms: Needs, options and consequences, Soil Dynam. Earthq. Eng. 25, 93–115.

Boore, D. M., A. Azari Sisi, and S. Akkar (2012). Using pad-stripped acausally filtered strong-motion data, Bull. Seismol. Soc. Am. 102 751-760.

Converse, A. M., and A. G. Brady (1992). BAP—Basic strong-motion accelerograms processing software; Version 1.0, U. S. Geological Survey Open-File Report 92-296A 174 pp.

Douglas, J. and D. M. Boore (2011). High-frequency filtering of strong-motion records, Bull. Earthquake Engineering 9 395-409.

Pacor, F., R. Paolucci, G. Ameri, M. Massa, and R. Puglia (2011). Italian strong motion records in ITACA: Overview and record processing, Bull. Earthq. Eng. 9, 1741–1759.

Paolucci, R., F. Pacor, R. Puglia, G. Ameri, C. Cauzzi, and M. Massa (2011). Record processing in ITACA, the newItalian strong-motion database, in Earthquake Data in Engineering Seismology: Predictive Models, Data Management, and Networks Akkar, Sinan, Polat Gülkan, and Torild Van Eck (Editors), Springer, Dordrecht, the Netherlands, 99–113.

Appendix A: Matlab codes for data processing of accelerograms in ESM_db

The codes given in the report are devised for batch processing of raw strong-motion data. The inputs are read from an Excel file that contains the file names of raw accelerograms, low-pass and high-pass filter cut of values. The excel file also contains information about the non-standard errors of the accelerograms (raw acceleration time series with spikes or records with multiple events). These non-standard errors are removed first. Then each accelerogram is acausally filtered in the frequency domain and PEER post-processing scheme is implemented.

The code is written in a modular format. The following Matlab files are given in the report:

- Batchprocess.m: Reads the input Excel file and calls the routines for removing non-standard errors as well as PEER post-processing. It also outputs the post-processed accelerograms as text files together with their velocity and displacement ground motions. This Matlab code should be executed for running the entire process.
- ClearSpike.m: Removes high-frequency and unusual spikes from the raw accelerograms
- Taper.m: Tapers the beginning and end of the accelerograms before band-pass acausal filtering in frequency domain
- BP_FD.m: Band-pass acausal filtering in frequency domain
- bcPEA.m: PEER post-processing

```
% This program applies PEER strong-motion data processing scheme in a batch mode.
% It reads the necessary inputs of a set of raw accelerograms from an excel
% file to apply the procedure.
% The code is developed for NERA-NA3 and is used in the paper entitled
% "Using pad-stripped acausally filtered strong-motion data"
% by Boore, Sisi and Akkar (2012). The paper is published in the Bulletin
% of the Seismological Socitey of America, Vol. 102, No. 2, pp. 751-760.
% The user is referred to this article for details of PEER strong-motion
% data processing
%
% Initial coding: Aida Azari Sisi
% Last modified: Sinan Akkar
function [] = Batchprocess_mod(Folder1, Folder2, xl_fle, sprdsht)


% Read the Excel file and open the relevant folders for input and output
% post-processed accelerograms and consisten velocity and displacements
% Folder1: A text string designating the location of input files
% Example 'D:\Processing\PEER\Input'
% Folder2: A text string designating the location of output files
% Example 'D:\Processing\PEER\Output'
% xl_fle: Name of the excel file. Example 'processing.xlsx'
% sprdsht: Name of the spreadsheet in the Excel file. Example 'Metadata'



clear all

[A,B]=xlsread(xl_fle,sprdsht); % "A" stores the numeric information and "B" stores text
information from the Excel file


% Start processing the relevant information from Excel file for PEER scheme

for j=1:9
    disp(['WFID=',(B(j+1,1))]); % Read waveform id. Use it for outputting the post proccesed
accelerogram
    WFID=B{j+1,1};
    OutputFileName=(WFID);
    mkdir([Folder2,'\',OutputFileName]);
    Acc1=importdata([Folder1,'\',B{j+1,3}]); % Raw acceleration data of Comp. 1
    Acc2=importdata([Folder1,'\',B{j+1,5}]); % Raw acceleration data of Comp. 2
    Acc3=importdata([Folder1,'\',B{j+1,7}]); % Raw acceleration data of Comp. 3
    dt1=Acc1(2,1); % time increment of Comp. 1
    dt2=Acc2(2,1); % time increment of Comp. 2
    dt3=Acc3(2,1); % time increment of Comp. 3


    % Correction of non-standard errors


    % Remove spikes if data are good quality
    if strcmp(B{j+1,13},'Y')==1  % check if the accelerogram (with 3 components) contains
spike
        if strcmp(B{j+1,8},'Y')~=1;  % check if Comp. 1 is a bad quality record (if so, don't
do anything)
            [acc1]=ClearSpike(Acc1(:,1)',Acc1(:,2)'); % If Comp. 1 contains spike and if it is
not bad quality, clear the spikes
           Acc1 = acc1;
        end
        if strcmp(B{j+1,9},'Y')~=1;  % check if Comp. 2 is a bad quality record (if so, don't
do anything)
            [acc2]=ClearSpike(Acc2(:,1)',Acc2(:,2)');  % If Comp. 2 contains spike and if it is
not a bad quality record, clear the spike
            Acc2 = acc2;
        end
        if strcmp(B{j+1,10},'Y')~=1;  % check if Comp. 3 is a bad quality record (if so, don't
do anything)
            [acc3]=ClearSpike(Acc3(:,1)',Acc3(:,2)'); % If Comp. 3 contains spike and if it is
not a bad quality record, clear the spike
            Acc3 = acc3;
        end
    end


    % Reduce multiple shocks to main shock for Comp. 1 and proceed with PEER
    % post processing scheme
```

```
    if strcmp(B{j+1,8},'Y')~=1; % check if Comp. 1 is a bad quality record (if so, don't do
anything)
        dt=dt1;
        if strcmp(B{j+1,13},'Y')==1  % check if Comp. 1 contains pre-event buffer
            acc=Acc1;
        end
        if strcmp(B{j+1,12},'Y')==1 % check if Comp. 1 contains multiple events
            st=A(j,2);  % isolate the main event from the entire record
            et=A(j,3);  % (st: start time of main event, et: end time of main event)
            [acc]=Acc1(floor(st/dt)+1:floor(et/dt)+1,2);
        end
        if (strcmp(B{j+1,13},'Y')~=1 && strcmp(B{j+1,12},'Y')~=1)
            acc=Acc1(:,2);
        end
        pe=A(j,1);              % read pre-event time from Excel file (it is assumed to be the
same for all 3 components)
        du=(length(acc)-1)*dt;  % total duration of Comp. 1
        pr=pe-(du-pe)*0.05;     % time length for tapering (5% of actual ground-motion length
- excluding pre-event buffer)
        r=floor(pr/dt)+1;       % number of data for tapering
        if r<=0,
            r=1;  % If r is negative, there is no pre-event buffer in Comp.1
        end
        Acc1=acc(r:end);       % Comp. 1 to be tapered and band-pass filtered
        Acc1=Acc1-mean(Acc1);  % Subtract mean of Comp. 1 from the entire record
        flc=A(j,4);      % Read low-cut and
        fhc=A(j,5);      % hi-cut filter values of Comp. 1 from Excel file
        np=4;            % number of poles for acausal filtering
        if strcmp(B{j+1,11},'Y')==1  % If Comp. 1 is S-wave triggered
        [Acc1]=taper(Acc1,0,5);      % taper the end
        else                         % If Comp. 1 is not S-wave triggered
        [Acc1]=taper(Acc1,5,5);      % taper the beginning and end
        end
        [acc_filt,acc_filt_pads_removed]=BP_FD(Acc1,dt,flc,fhc,np);  %frequency domain acausal
filtering
        % PEER post processing
        order=6;
        taper_frac=10;
        [acc_bc]=bcPEA(acc_filt_pads_removed,dt,order,taper_frac); % acc_bc is the post
processed acausally filtered accelerogram
        t=0:dt:(length(acc_bc)-1)*dt;
        % Integrate post-processed accelerogram to obtain consistent displacement and velocity
        vel=cumtrapz(t,acc_bc);
        disp=cumtrapz(t,vel);
        PGA=max(abs(acc_bc));
        PGV=max(abs(vel));
        PGD=max(abs(disp));
        output=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp1.a.COR.col'],'wt');
        Accnew=[t' acc_bc];
        nData=length(acc_bc);
        fprintf(output,'%g\n',nData);
        fprintf(output,'%7.3f %20.12f\n',Accnew');
        fclose(output);

output1=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp1.v.COR.col'],'wt');
        Velnew=[t' vel];
        nData=length(vel);
        fprintf(output1,'%g\n',nData);
        fprintf(output1,'%7.3f %20.12f\n',Velnew');
        fclose(output1);

output2=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp1.d.COR.col'],'wt');
        Dispnew=[t' vel];
        nData=length(disp);
        fprintf(output2,'%g\n',nData);
        fprintf(output2,'%7.3f %20.12f\n',Dispnew');
        fclose(output2);
        Peak=[PGA PGV PGD];
        xlswrite('saed.xlsx',Peak,'Flatfile',['Y',num2str(j+1)]);
    end
    clear acc acc_bc st et pe dis vel t dt pr r du Accnew Velnew Dispnew PGA PGV PGD Peak disp

    % Reduce multiple shocks to main shock for Comp. 2 and proceed with PEER
    % post processing scheme
    if strcmp(B{j+1,9},'Y')~=1;  % check if Comp. 2 is a bad quality record (if so, don't do
anything)
```

```
        dt=dt2;
        if strcmp(B{j+1,13},'Y')==1  % check if Comp. 2 contains pre-event buffer
            acc=Acc2;
        end
        if strcmp(B{j+1,12},'Y')==1  % check if Comp. 2 contains multiple events
            st=A(j,2);  % isolate the main event from the entire record
            et=A(j,3);  % (st: start time of main event, et: end time of main event)
            [acc]=Acc2(floor(st/dt)+1:floor(et/dt)+1,2);
        end
        if (strcmp(B{j+1,13},'Y')~=1 && strcmp(B{j+1,12},'Y')~=1)
            acc=Acc2(:,2);
        end
        pe=A(j,1);   % read pre-event time from Excel file (it is assumed to be the same for
all 3 components)
        du=(length(acc)-1)*dt;  % total duration of Comp. 2
        pr=pe-(du-pe)*0.05;     % time length for tapering (5% of actual ground-motion length
- excluding pre-event buffer)
        r=floor(pr/dt)+1;       % number of data for tapering
        if r<=0
            r=1;  % If r is negative, there is no pre-event buffer in Comp.2
        end
        Acc2=acc(r:end);       % Comp. 2 to be tapered and band-pass filtered
        Acc2=Acc2-mean(Acc2);  % Subtract mean of Comp. 2 from the entire record
        flc=A(j,6);     % Read low-cut and
        fhc=A(j,7);     % hi-cut filter values of Comp. 2 from Excel file
        np=4;           % number of poles for acausal filtering
        if strcmp(B{j+1,11},'Y')==1  % If Comp. 2 is S-wave triggered
            [Acc2]=taper(Acc2,0,5);  % taper the end
        else                         % If Comp. 2 is not S-wave triggered
            [Acc2]=taper(Acc2,5,5);  % taper the beginning and end
        end
        [acc_filt,acc_filt_pads_removed]=BP_FD(Acc2,dt,flc,fhc,np);  %frequency domain acausal
filtering
        % PEER post processing
        order=6;
        taper_frac=10;
        [acc_bc]=bcPEA(acc_filt_pads_removed,dt,order,taper_frac); % acc_bc is the post
processed acausally filtered accelerogram
        t=0:dt:(length(acc_bc)-1)*dt;
        % Integrate post-processed accelerogram to obtain consistent displacement and velocity
        vel=cumtrapz(t,acc_bc);
        disp=cumtrapz(t,vel);
        PGA=max(abs(acc_bc));
        PGV=max(abs(vel));
        PGD=max(abs(disp));
        output=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp2.a.COR.col'],'wt');
        Accnew=[t' acc_bc];
        nData=length(acc_bc);
        fprintf(output,'%g\n',nData);
        fprintf(output,'%7.3f %20.12f\n',Accnew');
        fclose(output);

output1=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp2.v.COR.col'],'wt');
        Velnew=[t' vel];
        nData=length(vel);
        fprintf(output1,'%g\n',nData);
        fprintf(output1,'%7.3f %20.12f\n',Velnew');
        fclose(output1);

output2=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp2.d.COR.col'],'wt');
        Dispnew=[t' vel];
        nData=length(disp);
        fprintf(output2,'%g\n',nData);
        fprintf(output2,'%7.3f %20.12f\n',Dispnew');
        fclose(output2);
        Peak=[PGA PGV PGD];
        xlswrite('saed.xlsx',Peak,'Flatfile',['AB',num2str(j+1)]);
    end

    clear acc acc_bc st et pe dis vel t dt pr r du Accnew Velnew Dispnew PGA PGV PGD Peak disp

    % Reduce multiple shocks to main shock for Comp. 3 and proceed with PEER
    % post processing scheme
```

```
    if strcmp(B{j+1,10},'Y')~=1;  % check if Comp. 3 is a bad quality record (if so, don't do
anything)
        dt=dt3;
        if strcmp(B{j+1,13},'Y')==1   % check if Comp. 3 contains pre-event buffer
            acc=Acc3;
        end
        if strcmp(B{j+1,12},'Y')==1   % check if Comp. 3 contains multiple events
            st=A(j,2);   % isolate the main event from the entire record
            et=A(j,3);   % (st: start time of main event, et: end time of main event)
            [acc]=Acc3(floor(st/dt)+1:floor(et/dt)+1,2);
        end
        if (strcmp(B{j+1,13},'Y')~=1 && strcmp(B{j+1,12},'Y')~=1)
            acc=Acc3(:,2);
        end
        pe=A(j,1);   % read pre-event time from Excel file (it is assumed to be the same for
all 3 components)
        du=(length(acc)-1)*dt;   % total duration of Comp. 3
        pr=pe-(du-pe)*0.05;      % time length for tapering (5% of actual ground-motion length
- excluding pre-event buffer)
        r=floor(pr/dt)+1;        % number of data for tapering
        if r<=0
            r=1;   % If r is negative, there is no pre-event buffer in Comp.3
        end
        Acc3=acc(r:end);       % Comp. 3 to be tapered and band-pass filtered
        Acc3=Acc3-mean(Acc3);  % Subtract mean of Comp. 3 from the entire record
        flc=A(j,8);    % Read low-cut and
        fhc=A(j,9);    % hi-cut filter values of Comp. 3 from Excel file
        np=4;          % number of poles for acausal filtering
        if strcmp(B{j+1,11},'Y')==1  % If Comp. 3 is S-wave triggered
            [Acc3]=taper(Acc3,0,5);  % taper the end
        else                         % If Comp. 3 is not S-wave triggered
            [Acc3]=taper(Acc3,5,5);  % taper the beginning and end
        end
        [acc_filt,acc_filt_pads_removed]=BP_FD(acc3,dt,flc,fhc,np);  %frequency domain acausal
filtering
        % PEER post processing
        order=6;
        taper_frac=10;
        [acc_bc]=bcPEA(acc_filt_pads_removed,dt,order,taper_frac); % acc_bc is the post
processed acausally filtered accelerogram
        t=0:dt:(length(acc_bc)-1)*dt;
        % Integrate post-processed accelerogram to obtain consistent displacement and velocity
        vel=cumtrapz(t,acc_bc);
        disp=cumtrapz(t,vel);
        PGA=max(abs(acc_bc));
        PGV=max(abs(vel));
        PGD=max(abs(disp));
        output=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp3.a.COR.col'],'wt');
        Accnew=[t' acc_bc];
        nData=length(acc_bc);
        fprintf(output,'%g\n',nData);
        fprintf(output,'%7.3f %20.12f\n',Accnew');
        fclose(output);

output1=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp3.v.COR.col'],'wt');
        Velnew=[t' vel];
        nData=length(vel);
        fprintf(output1,'%g\n',nData);
        fprintf(output1,'%7.3f %20.12f\n',Velnew');
        fclose(output1);

output2=fopen([Folder2,'\',OutputFileName,'\',OutputFileName,'_Comp3.d.COR.col'],'wt');
        Dispnew=[t' vel];
        nData=length(disp);
        fprintf(output2,'%g\n',nData);
        fprintf(output2,'%7.3f %20.12f\n',Dispnew');
        fclose(output2);
        Peak=[PGA PGV PGD];
        xlswrite('saed.xlsx',Peak,'Flatfile',['AE',num2str(j+1)]);
    end

    clear acc acc_bc st et pe dis vel t dt pr r du Accnew Velnew Dispnew PGA PGV PGD Peak disp
end
```

```
% A subroutine to remove spikes from the accelerograms
% Initially coded by Aida Azari Sisi
% Last modified by Sinan Akkar
function [AccelData_comp_spike_cleared]=ClearSpike(TimeData,AccelData_comp)


clf;
plot(TimeData,AccelData_comp);ylabel('Acceleration, cm/s^2','FontSize',8);xlabel('Time
(s)','FontSize',8);
disp('> Accelerogram of the record is ploted.');
disp(' ');

SampleInt=TimeData(1,2)-TimeData(1,1);

comp_spike_index=input('- Does the record include any Spikes? (Y,N): ','s');
if length(comp_spike_index)<1
    while upper(comp_spike_index)=='Y' | upper(comp_spike_index)=='N'
        comp_spike_index=input('- Does the record include any Spikes? (Please enter Y or N):
','s');
    end
end
disp(' ');
comp_spike_index=upper(comp_spike_index);




if comp_spike_index=='Y'

    while comp_spike_index=='Y'
        start_time=input('- Enter the starting time of the spike. t_start= ');
        end_time=input('- Enter the ending time of the spike. t_end= ');

        start_index=start_time/SampleInt+1;
        end_index=end_time/SampleInt+1;

        max_acc=max(AccelData_comp(1,start_index:end_index));
        max_index=find(AccelData_comp(1,start_index:end_index)==max_acc);

        min_acc=min(AccelData_comp(1,start_index:end_index));
        min_index=find(AccelData_comp(1,start_index:end_index)==min_acc);

        if abs(max_acc)>abs(min_acc)
            pga_range=max_acc;
            s_index=max_index+start_index-1;
        else
            pga_range=min_acc;
            s_index=min_index+start_index-1;
        end
        t_spike=(s_index-1)*SampleInt;

        confirm_index=input(['> Spike: ',num2str(pga_range),'cm/sec^2 @
t=',num2str(t_spike),'sec. Do you want to delete? (Y/N): '],'s');
        if length(confirm_index)<1
            while upper(confirm_index)=='Y' | upper(confirm_index)=='N'
                confirm_index=input(['> Spike: ',num2str(pga_range),'cm/sec^2 @
t=',num2str(t_spike),'sec. Do you want to delete? (Please enter Y or N): '],'s');
            end
        end
        disp(' ');
        confirm_index=upper(confirm_index);

        if confirm_index=='Y'
            mean_acc=(AccelData_comp(1,(s_index-1))+AccelData_comp(1,(s_index+1)))/2;
            AccelData_comp=[AccelData_comp(1,1:(s_index-
1)),mean_acc,AccelData_comp(1,(s_index+1):end)];

            plot(TimeData,AccelData_comp);ylabel('Acceleration,
cm/s^2','FontSize',8);xlabel('Time (s)','FontSize',8);
            disp('> Accelerogram of the record is updated.');
            disp(' ');
```

```
            comp_spike_index=input('- Does the record include any Spikes? (Y,N): ','s');
            if length(comp_spike_index)<1
                while upper(comp_spike_index)=='Y' | upper(comp_spike_index)=='N'
                    comp_spike_index=input('- Does the record include any Spikes? (Please
enter Y or N): ','s');
                end
            end
            disp(' ');
            comp_spike_index=upper(comp_spike_index);
        end


    end


end
close all;
AccelData_comp_spike_cleared_1=AccelData_comp;
AccelData_comp_spike_cleared=AccelData_comp_spike_cleared_1';
```

```
% Matlab subroutine to taper the ends of accelerograms before applying
% band-pass acausal filtering
% Initially coded by Aida Azari Sisi
% Last modified by Sinan Akkar
function [x]=taper (x,tb,te)

n1 = (length(x)*tb)/100;

for i=1:n1
arg = pi*(i-1)/n1 + pi;
x(i) = x(i)*(1+cos(arg))/2;
end

n1 = (length(x)*te)/100;

for i=1:n1
arg = pi*(i-1)/n1 + pi;
x(length(x)-i+1) = x(length(x)-i+1)*(1+cos(arg))/2;
end
```

```
% Band-pass acausal filtering in frequency domain (from David M.Boore's Fortran codes)
% Initial coding: Aida Azari Sisi
% Last modified: Sinan Akkar
function [acc_filt,acc_filt_pads_removed]=BP_FD(acc,dt,flc,fhc,np)

% acc is acceleration data
% np is number of poles
% flc is low-cut filter frequency
% fhc is high-cut filter frequency
% npf is number of poles in forward direction
% npr is number of poles in reverse direction

n=nextpow2(length(acc));
N=2^n;
if length(acc)>=N/2
    N=2*N;
end

accFD=fft(acc,N); % Fourier transform of the accelerogram
df=1/(N*dt);
f=df:df:(N/2)*df;
acc_filt_FD=zeros(1,N);

% Low-cut (high-pass) filtering
if flc==0  % No low-cut filtering
    acc_filt_FD_l=accFD(1:N/2)';
else
    npf=-np; % npf and npr are negative
    npr=-np; % if high-pass (low-cut) filtering
    fc=flc;
    [filter_response_f]=FR(f,fc,npf);  % forward filtering response
    [filter_response_r1]=FR(f,fc,npr); % reverse filtering response
    filter_response_r=conj(filter_response_r1);
    filter_response_t= filter_response_f.*filter_response_r; % total filter response
    acc_filt_FD_l=accFD(1:N/2)'.*filter_response_t; % multipy FT of acceleration data by
filter response in FD
end
clear filter_response_f filter_response_r1 filter_response_r filter_response_t

% High-cut (low-pass) filtering
if fhc==999  % No hi-cut filtering
    acc_filt_FD(1:N/2)=acc_filt_FD_l;
else
    npf=np; % npf and npr are positive
    npr=np; % if low-pass (hi-cut) filtering
    fc=fhc;
    [filter_response_f]=FR(f,fc,npf);  % forward filtering response
    [filter_response_r1]=FR(f,fc,npr); % reverse filtering response
    filter_response_r=conj(filter_response_r1);
    filter_response_t= filter_response_f.*filter_response_r; % total filter response
    acc_filt_FD(1:N/2)=acc_filt_FD_l.*filter_response_t; % multipy raw data by filter response
in FD
end

acc_filt_FD(N/2+1)=0; %Set Nyquist value to zero

for j=2:N/2
    acc_filt_FD(N-j+2)=conj(acc_filt_FD(j)); % Fill the other half of the array
end

acc_filt=ifft(acc_filt_FD'); % Back to time domain
acc_filt=real(acc_filt);
acc_filt_pads_removed=acc_filt(1:length(acc));  % zero pads at the end of the record are
removed for PEER post-processing

return
```

```
% This program applies the PEER post-processing procedure to the pads
% stripped off acausally filtered acceleration time series. It compares the
% outputs with the originally padded and acausally filtered time series.
% Initially coded by Aida Azari Sisi 1/4/2011
%
% Input parameters:
% acc: pads stripped-off acceleration time series
% dt: equal time interval (time sampling)
% order: order of polynominal fitted to displacement(a1*t^2+a2*t^3+...+an-1*t^n)
% taper_frc: tapering percent


function [acc1]=bcPEA(acc,dt,order,taper_frac)



% Integration of pads stripped-off acceleration data for velocity and
% displacement
t=0:dt:(length(acc)-1)*dt;
vel=cumtrapz(t,acc);
disp=cumtrapz(t,vel);

% Apply pads to the end of displacement with values equal to the last displacement
% data.(Pad length is 10% of total record duration)
disp1=disp;
for i=length(disp)+1:length(disp)+0.1*length(disp)
    disp1(i)=disp1(length(disp));
end
t1=0:dt:(length(disp1)-1)*dt;

% Fit nth order polynomial to the padded displacement.
% The first and second terms of the polynomial are constrained to zero
% (a1*t^2+a2*t^3+a3*t^4+...+an-1*t^n)

n=order;
for i=1:length(disp1)
    for j=2:n
        A(i,j-1)=t1(i)^j;
    end
end
c=(A'*A)\(A'*disp1); % c is the vector of coefficients of the polynomial fitted to the
displacement

% Compute the baseline fitted to displacement and its first and second derivatives using the
coefficients in c

for i=1:length(disp)
    sum0=0;
    sum1=0;
    sum2=0;
    for j=2:n
        sum0=sum0+c(j-1)*t(i)^j;
        sum1=sum1+j*c(j-1)*t(i)^(j-1);
        sum2=sum2+j*(j-1)*c(j-1)*t(i)^(j-2);
    end
    b=sum0;
    db=sum1;
    ddb=sum2;

    %Compute the taper coefficients to be applied at the end of the trace
    n1=length(disp)-(length(disp)*taper_frac/100);
    if i<=n1
        w0=1;
        w1=0;
        w2=0;
    else
        w0=0.5*(cos(pi*(i-n1)/(length(disp)-n1))+1);
        w1= -0.5 * pi/((length(disp)-n1)*dt) * sin( pi*(i-n1)/(length(disp)-n1) );
        w2=-0.5 * (pi/((length(disp)-n1)*dt))^2 * cos( pi*(i-n1)/(length(disp)-n1) );
    end

    % Compute the post-processed acceleration by subtracting the 2nd time
```

```
    % derivative of displacement baseline and apply the taper to the end of the trace
       acc1(i,1)=(acc(i)-ddb)*w0+2*(vel(i)-db)*w1+(disp(i)-b)*w2;
end

return
```

Appendix B: Content of Engineering Strong Motion database (ESM_db)

The Engineering Strong Motion database (ESMdb), combines the expertise and datasets gained within several European projects, such as the Internet Site for European Strong Motion data (FP4 and FP5), NERIES (FP6), SHARE (FP7) and project funded by private companies (SIGMA) and national agencies (Italian accelerometric archive - http://itaca.mi.ingv.it/; Turkish national strong-motion network - http://kyh.deprem.gov.tr/).

ESMdb is a centralized database that distributes strong-motion recordings available in Europe and surroundings since the 1970s. The current ESMdb prototype for disseminating strong-motion data contains Turkish and Italian strong-motion recordings and it is actually password protected (Figure B.1 shows the homepage). The European data contained in the SHARE database will be included before the end of the project.

Different from the previous databases, ESMdb takes advantage of modern seismological services that provide rapid access to strong motion data and allow to populate the database automatically. Nevertheless, a manual interaction is still required as the accelerometric data in ESMdb are manually processed and quality checked before being distributed. In the perspective of engineering applications, the database is designed to host as many information as possible contained in 80 related tables. They are relevant to:

- recording sites: station information (network code, station code, latitude, longitude, elevation, site morphology, housing, stratigraphy logs, Vs logs, NSPT logs, dispersion curves, HVSR curves, fundamental frequency, etc.) and station documents (photos, maps, papers and reports);
- seismic events: latitude, longitude, depth, magnitude, focal mechanism, fault geometry, macroseismic intensity, etc.;
- waveforms: sampling rate, number of points and strong motion parameters (PGA, PGV, PGD, filter type and corners, spectral ordinates, Arias Intensity, Housner Intensity, duration, etc.);
- networks: international codes, data provider information, logos and contacts;
- data dictionary and references.

and displacement time series and acceleration and displacement response spectra (5% damping). The ASCII files contains a 55 header lines containing all the relevant information for the correct use of the waveform. Additional tools are provided to convert the ASCII files in binary format used for seismological applications (SAC or Miniseed).

About 50 fields can be specified for event, station and waveform selection and mapping tools are provided to display seismic events and recording stations on Google maps (Figure B.2 and B.3) as well as waveform visualization tools (Figure B.4).

The database contains the following strong motion parameters, which can be used to select waveforms from the web site:

PGA of unprocessed waveform
PGA of processed waveform
PGV of processed waveform
PGD of processed waveform
Significant duration
Housner Intensity
Arias intensity

Moreover, in the database and in the file header (Table B.1) the low-pass and high-pass filter corners, which have been used to filter each component are secured in order to provide the user the usable frequency range for each accelerogram.

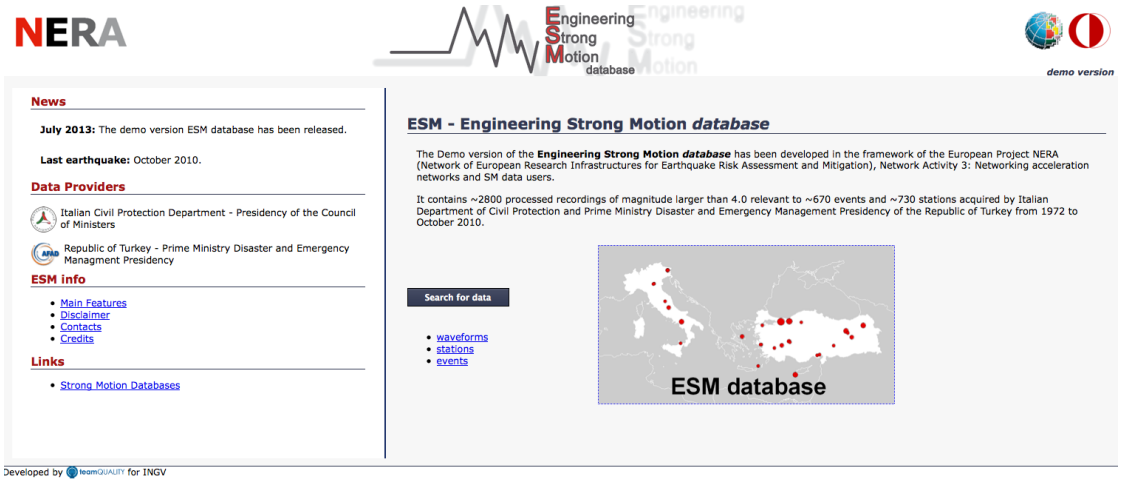

Figure B.1: Engineering Strong motion database home page

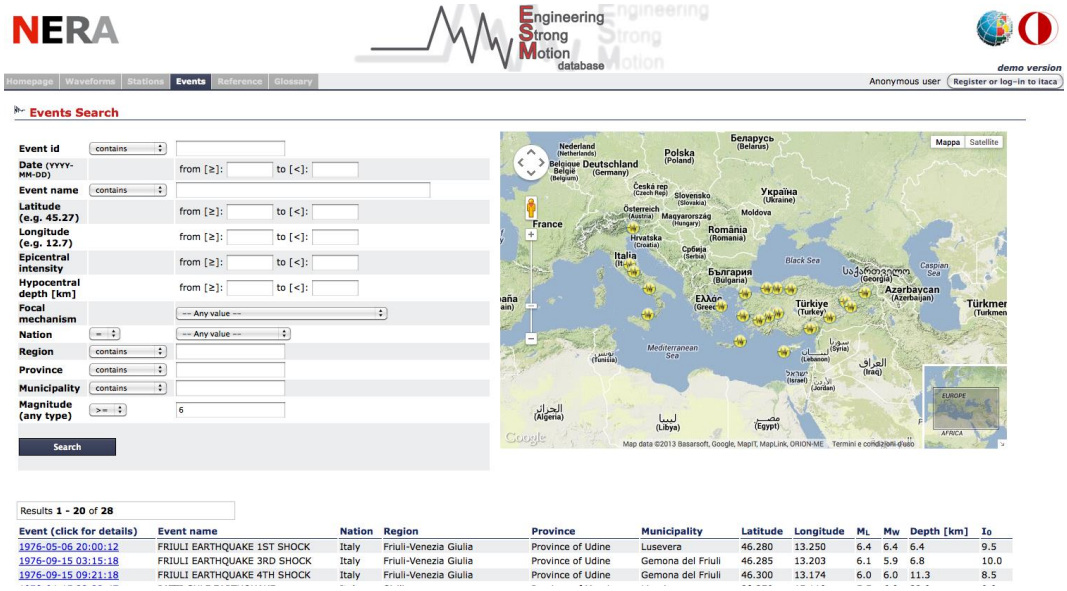

Figure B.2: Event search

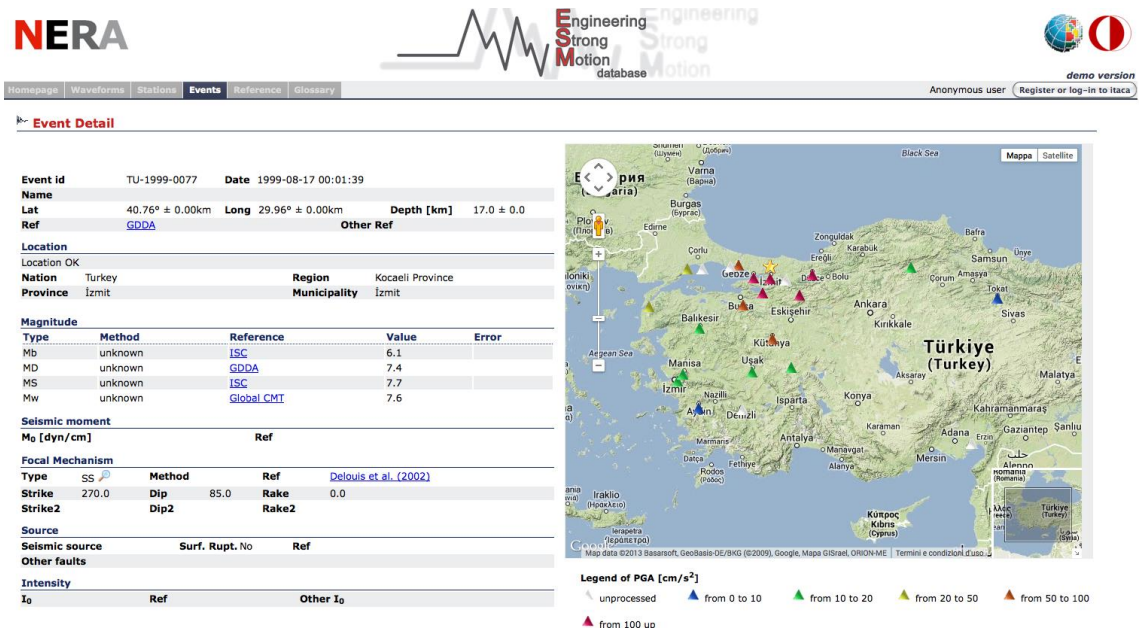

Figure B.3: Single event page

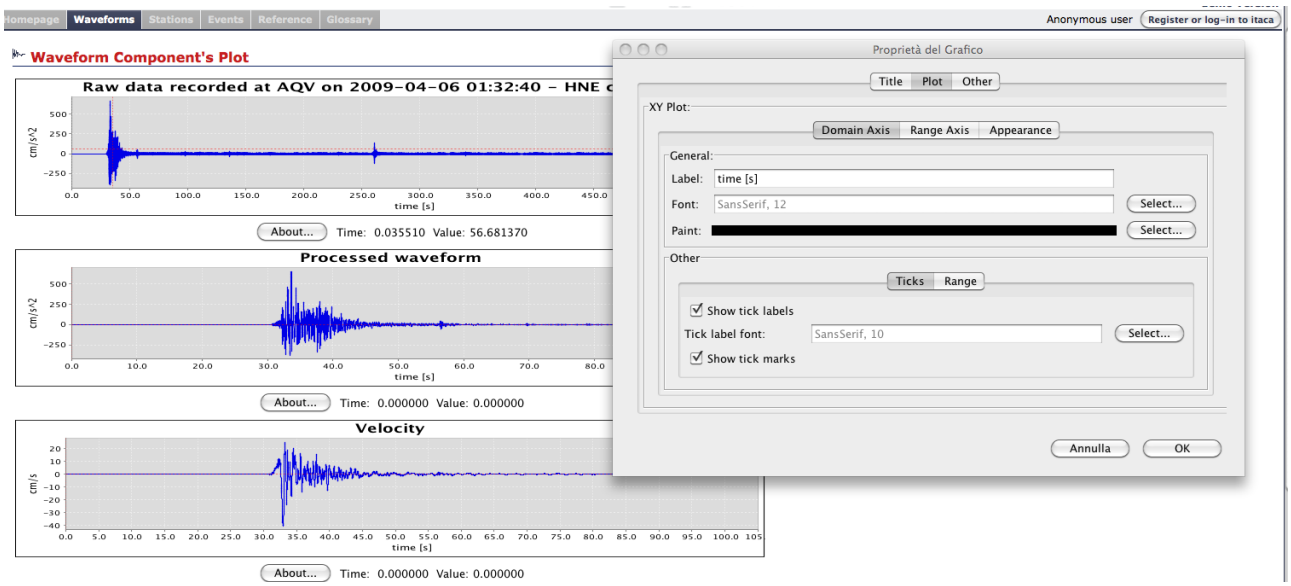

Figure B.4 Tool for data visualization and plot

Table B.1. File header

EVENT_NAME: EMILIA_2ND_MAINSHOCK
EVENT_ID: IT-2012-0011
EVENT_DATE_YYYYMMDD: 20120529
EVENT_TIME_HHMMSS: 070003
EVENT_LATITUDE_DEGREE: 44.8510
EVENT_LONGITUDE_DEGREE: 11.0860
EVENT_DEPTH_KM: 10.2
HYPOCENTER_REFERENCE: ISIDe
MAGNITUDE_W: 6.0

MAGNITUDE_W_REFERENCE: RCMT-INGV
MAGNITUDE_L: 5.8
MAGNITUDE_L_REFERENCE: ISIDe
FOCAL_MECHANISM: TF
NETWORK: IT
STATION_CODE: MRN
STATION_NAME: MIRANDOLA (NAPOLI)
STATION_LATITUDE_DEGREE: 44.886389
STATION_LONGITUDE_DEGREE: 11.072778
STATION_ELEVATION_M: 18
LOCATION:
VS30_M/S:
SITE_CLASSIFICATION_EC8: C*
MORPHOLOGIC_CLASSIFICATION:
EPICENTRAL_DISTANCE_KM: 4.1
EARTHQUAKE_BACKAZIMUTH_DEGREE: 165.1
DATE_TIME_FIRST_SAMPLE_YYYYMMDD_HHMMSS: 20120529_065933.000
DATE_TIME_FIRST_SAMPLE_PRECISION: seconds
SAMPLING_INTERVAL_S: 0.005000
NDATA: 20001
DURATION_S: 100.000000
STREAM: HNE
UNITS: cm/s^2
INSTRUMENT: sensor = MS2007+ [Syscom] | digitizer = 130-01/3 [Reftek]
INSTRUMENT_ANALOG/DIGITAL: D
INSTRUMENTAL_FREQUENCY_HZ:
INSTRUMENTAL_DAMPING:
FULL_SCALE_G:
N_BIT_DIGITAL_CONVERTER: 24
PGA_CM/S^2: 218.643365
TIME_PGA_S: 35.520000
BASELINE_CORRECTION: BASELINE REMOVED
FILTER_TYPE: BUTTERWORTH
FILTER_ORDER: 2
LOW_CUT_FREQUENCY_HZ: 0.070
HIGH_CUT_FREQUENCY_HZ: 40.000
LATE/NORMAL_TRIGGERED: NT
DATABASE_VERSION: DYNA 1.0
HEADER_FORMAT: DYNA 1.0
DATA_TYPE: ACCELERATION
PROCESSING: manual
DATA_TIMESTAMP_YYYYMMDD_HHMMSS:
USER1:
USER2:
USER3:
USER4: