Μέρος Β

```
            FuzzyEn(1:fsize4) = zeros(fsize4,1);
            for nFrame=2:NoF
                %sprintf('Remaing execution times: %d\n*',NoF-
nFrame)
                indx1=(nFrame-1)*inc+1;
                indx2=(nFrame-1)*inc+fsize4;

                r =r_var*std(EQ_Data(indx1:indx2));

                % Computation of FuzzyEn per frame

FuzzyEn(indx1:indx2)=FuzzyEntr_v3(EQ_Data(indx1:indx2),M,r,N)
;
            end
            FuzzyEn(1:fsize4) = FuzzyEn(inc+1:fsize4+inc);

            text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {H} Filtered Fuzzy
En.'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,FuzzyEn,0,0,0,0,0,6);
            guidata(hObject,handles);
        end
    end

end;

%% New Zante Fuzzy Computation
if ( get(handles.checkbox30,'Value') == 1 ) %Then NewZante is
checked

    index_of_selected=get(handles.listbox28, 'Value');

    for i=1:length(index_of_selected)

        clear EQ_Data;
        clear M;
        clear N;
        clear r_var;
        clear ovl;
        clear inc;
        clear NoF;
```

```
        clear r;
        clear FuzzyEn;

data_to_plot=['handles.RF_N_ch',num2str(index_of_selected(i))
];
        EQ_Data=eval(data_to_plot);

        fsize4 = handles.fsize4;
        ovl   =   str2double(get(handles.edit9,'String'));   %
percentage of overlapping
        inc=fsize4*(1-ovl/100);%increment in samples

        NoF=floor((length(EQ_Data)-fsize4)/inc)+1;
        %EQ_DataFrame=EQ_Data(1:fsize4);



        M = str2double(get(handles.edit11,'String'));
        N=str2double(get(handles.edit10,'String'));
        r_var=str2double(get(handles.edit8,'String'));

        FuzzyEn(1:fsize4) = zeros(fsize4,1);
        for nFrame=2:NoF
            sprintf('Remaing   execution   times:   %d\n*',NoF-
nFrame)
            indx1=(nFrame-1)*inc+1;
            indx2=(nFrame-1)*inc+fsize4;

            r =r_var*std(EQ_Data(indx1:indx2));

            % Computation of FuzzyEn per frame

FuzzyEn(indx1:indx2)=FuzzyEntr_v3(EQ_Data(indx1:indx2),M,r,N)
;
        end
        FuzzyEn(1:fsize4) = FuzzyEn(inc+1:fsize4+inc);

      text_to_display=['Channel
',num2str(index_of_selected(i)),' {N} Fuzzy En.'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,FuzzyEn,0,0,0,0,0,6);
      guidata(hObject,handles);
%         plot(EQ_Data,'k')
```

```
%         axis tight;
%         title('Input Signal vs. Time');
%
%         subplot(3,1,[2:3]);
%         plot(FuzzyEn,'k')
%         axis tight;
%          title_text=['Fuzzy Entropy (W=',num2str(fsize4),',
Overlap=',num2str(ovl),'%) vs. Time'];
%         title(title_text);


    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox29,
'Value');
        for i=1:length(index_of_selected_filtered)
            clear EQ_Data;
            clear M;
            clear N;
            clear r_var;
            clear ovl;
            clear inc;
            clear NoF;
            clear r;
            clear FuzzyEn;


data_to_plot=['handles.RF_N_ch',num2str(index_of_selected(i))
,'_fil'];
            EQ_Data=eval(data_to_plot);

            fsize4 = handles.fsize4;
            ovl  = str2double(get(handles.edit9,'String')); %
percentage of overlapping
            inc=fsize4*(1-ovl/100);%increment in samples

            NoF=floor((length(EQ_Data)-fsize4)/inc)+1;
            %EQ_DataFrame=EQ_Data(1:fsize4);



            M = str2double(get(handles.edit11,'String'));
```

```
            N=str2double(get(handles.edit10,'String'));
            r_var=str2double(get(handles.edit8,'String'));

            FuzzyEn(1:fsize4) = zeros(fsize4,1);
            for nFrame=2:NoF
                %sprintf('Remaing execution times: %d\n*',NoF-
nFrame)
                indx1=(nFrame-1)*inc+1;
                indx2=(nFrame-1)*inc+fsize4;

                r =r_var*std(EQ_Data(indx1:indx2));

                % Computation of FuzzyEn per frame

FuzzyEn(indx1:indx2)=FuzzyEntr_v3(EQ_Data(indx1:indx2),M,r,N)
;
            end
            FuzzyEn(1:fsize4) = FuzzyEn(inc+1:fsize4+inc);

            text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {N} Filtered Fuzzy
En.'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,FuzzyEn,0,0,0,0,0,6);
            guidata(hObject,handles);
        end
    end

end;

msg4=msgbox('Fuzzy Entropy Computation was finished');
clear selected_indx_NZa_fil;
clear selected_indx_NZa;
clear selected_indx_Ata_fil;
clear selected_indx_Ata;
clear selected_indx_Cor_fil;
clear selected_indx_Cor;
clear selected_indx_Vam_fil;
clear selected_indx_Vam;
clear selected_indx_Nea_fil;
clear selected_indx_Nea;
clear selected_indx_Rod_fil;
clear selected_indx_Rod;
```

```
clear selected_indx_Myt_fil;
clear selected_indx_Myt;
clear selected_indx_Kom_fil;
clear selected_indx_Kom;
clear selected_indx_Koz_fil;
clear selected_indx_Koz;
clear selected_indx_Ioa_fil;
clear selected_indx_Ioa;
clear selected_indx_kef_fil;
clear selected_indx_kef;
clear selected_indx_kal_fil;
clear selected_indx_kal;
clear selected_indx_fil;
clear selected_indx;


%=============================================================
====
%  End of Fuzzy Entropy Compuation Per Channel of Each Station
%=============================================================
====
```

```
function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as
text
%        str2double(get(hObject,'String')) returns contents of
edit8 as a double


% --- Executes during object creation, after setting all
properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as
text
%        str2double(get(hObject,'String')) returns contents of
edit9 as a double
```

```
% --- Executes during object creation, after setting all
properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as
text
%        str2double(get(hObject,'String')) returns contents of
edit10 as a double


% --- Executes during object creation, after setting all
properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end


function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as
text
%        str2double(get(hObject,'String')) returns contents of
edit11 as a double


% --- Executes during object creation, after setting all
properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu9.
function popupmenu9_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu9 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu9
```

```
row=get(handles.popupmenu9,'Value');
string=get(handles.popupmenu9,'String');
handles.fsize4=str2num(strtrim(string{row}));



% --- Executes during object creation, after setting all
properties.
function popupmenu9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu10.
function popupmenu10_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu10 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu10
row=get(handles.popupmenu10,'Value');
string=get(handles.popupmenu10,'String');
handles.fsize3=str2num(strtrim(string{row}));



% --- Executes during object creation, after setting all
properties.
function popupmenu10_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to popupmenu10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as
text
%        str2double(get(hObject,'String')) returns contents of
edit12 as a double


% --- Executes during object creation, after setting all
properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as
text
%        str2double(get(hObject,'String')) returns contents of
edit13 as a double


% --- Executes during object creation, after setting all
properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%
%
% % --- Executes on button press in checkbox16.
% function checkbox16_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox16 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox16


% % --- Executes on button press in checkbox17.
% function checkbox17_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox17 (see GCBO)
```

```
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkbox17

%
% % --- Executes on button press in checkbox18.
% function checkbox18_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox18 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox18
%
%
% % --- Executes on button press in checkbox19.
% function checkbox19_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox19 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox19
%
%
% % --- Executes on button press in checkbox20.
% function checkbox20_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox20 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox20
%
%
% % --- Executes on button press in checkbox21.
% function checkbox21_Callback(hObject, eventdata, handles)
```

```
% % hObject    handle to checkbox21 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox21
%
%
% % --- Executes on button press in checkbox22.
% function checkbox22_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox22 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox22
%
%
% % --- Executes on button press in checkbox23.
% function checkbox23_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox23 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox23
%
%
% % --- Executes on button press in checkbox24.
% function checkbox24_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox24 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox24
%
%
```

```
% % --- Executes on button press in checkbox25.
% function checkbox25_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox25 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox25
%
%
% % --- Executes on button press in checkbox26.
% function checkbox26_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox26 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox26
%
%
% % --- Executes on button press in checkbox27.
% function checkbox27_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox27 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox27
%
%
% % --- Executes on button press in checkbox28.
% function checkbox28_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox28 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox28
```

```
%
%
% % --- Executes on button press in checkbox29.
% function checkbox29_Callback(hObject, eventdata, handles)
% % hObject    handle to checkbox29 (see GCBO)
% % eventdata  reserved - to be defined in a future version of
MATLAB
% % handles      structure with handles and user data (see
GUIDATA)
%
% % Hint: get(hObject,'Value') returns toggle state of
checkbox29



function edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as
text
%        str2double(get(hObject,'String')) returns contents of
edit14 as a double


% --- Executes during object creation, after setting all
properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit16_Callback(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit16 as
text
%        str2double(get(hObject,'String')) returns contents of
edit16 as a double


% --- Executes during object creation, after setting all
properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
radiobutton2


% --- Executes on button press in radiobutton3.
function radiobutton3_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
radiobutton3


% --- Executes on button press in radiobutton4.
function radiobutton4_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
radiobutton4


% --- Executes on button press in radiobutton5.
function radiobutton5_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
radiobutton5



% --- Executes when selected object is changed in
uibuttongroup1.
function uibuttongroup1_SelectionChangeFcn(hObject,
eventdata, handles)
% hObject    handle to the selected object in uibuttongroup1
% eventdata  structure with the following fields (see
UIBUTTONGROUP)
%    EventName: string 'SelectionChanged' (read only)
%    OldValue: handle of the previously selected object or
empty if none was selected
%    NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
if hObject == handles.radiobutton2
    set(handles.text38,'string','1');

else
    set(handles.text38,'string','2');
```

```
end
guidata(hObject,handles);


% --- Executes when selected object is changed in uipanel25.
function   uipanel25_SelectionChangeFcn(hObject,   eventdata,
handles)
% hObject    handle to the selected object in uipanel25
% eventdata    structure with the following fields (see
UIBUTTONGROUP)
%    EventName: string 'SelectionChanged' (read only)
%    OldValue: handle of the previously selected object or
empty if none was selected
%    NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
if hObject == handles.radiobutton6
    set(handles.text39,'string','1');

else
    set(handles.text39,'string','2');

end



function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit17 as
text
%        str2double(get(hObject,'String')) returns contents of
edit17 as a double


% --- Executes during object creation, after setting all
properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called
```

```
% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit18 as
text
%        str2double(get(hObject,'String')) returns contents of
edit18 as a double


% --- Executes during object creation, after setting all
properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit19_Callback(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit19 as
text
%        str2double(get(hObject,'String')) returns contents of
edit19 as a double


% --- Executes during object creation, after setting all
properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit20_Callback(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit20 as
text
%        str2double(get(hObject,'String')) returns contents of
edit20 as a double


% --- Executes during object creation, after setting all
properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
```

```
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit21_Callback(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit21 as
text
%        str2double(get(hObject,'String')) returns contents of
edit21 as a double


% --- Executes during object creation, after setting all
properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton15.
function pushbutton15_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton15 (see GCBO)
% eventdata  reserved - to be defined in a future version of
```

```
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% figure; plot (handles.interv); axis tight;
% title('Intersets Time Intervals in seconds');
% ylabel('Time difference in sec');
% xlabel('Number of Earthquakes occurred');

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str('Seismic Time Intervals')];
guiplot(handles.plot_dropdown_names,handles.interv,0,0,0,0,0,
5);

%handles.plot_fig_commands=[handles.plot_fig_commands;cellstr
('plot(handles.interv);')];
%guiplot(handles.plot_dropdown_names,handles.plot_fig_command
s{:});
%guiplot(handles.plot_dropdown_names,handles.plot_fig_command
s,handles.interv);

guidata(hObject,handles);

%=============Select         Default         data         for
process===================
% --- Executes when selected object is changed in uipanel28.
function   uipanel28_SelectionChangeFcn(hObject,   eventdata,
handles)
% hObject    handle to the selected object in uipanel28
% eventdata    structure with the following fields (see
UIBUTTONGROUP)
%    EventName: string 'SelectionChanged' (read only)
%    OldValue: handle of the previously selected object or
empty if none was selected
%    NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
if hObject == handles.radiobutton8
    set(handles.text50,'string','1');
else
    set(handles.text50,'string','2');
end


% --- Executes on button press in checkbox30.
function checkbox30_Callback(hObject, eventdata, handles)
% hObject    handle to checkbox30 (see GCBO)
% eventdata  reserved - to be defined in a future version of
```

```
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkbox30


%=============================================================
============
%                          Plot Selected Channels
%=============================================================
============

% --- Executes on button press in pushbutton17.
function pushbutton17_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
%% Zante Channels
if ( get(handles.checkbox12,'Value') == 1 ) %Then Zante is
checked
    index_of_selected=get(handles.listbox4, 'Value');
    try
    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_Z_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {Z}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox5,
'Value');
        for i=1:length(index_of_selected_filtered)
```

```
data_to_plot=['handles.RF_Z_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
          text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {Z} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
          guidata(hObject,handles);
        end
    end
    catch exception

    end
end
%% Kalamata Channels
if ( get(handles.checkbox2,'Value') == 1 ) %Then Kalamata is
checked
    index_of_selected=get(handles.listbox14, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_O_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {O}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox15,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_O_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
```

```
            text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {O} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
            guidata(hObject,handles);
        end
    end

end
%% Kefalonia Channels
if ( get(handles.checkbox3,'Value') == 1 ) %Then Kefalonia is
checked
    index_of_selected=get(handles.listbox12, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_F_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {F}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox13,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_F_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
            text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {F} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
```

```
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
         end
    end

end

%% Ioannina Channels
if ( get(handles.checkbox4,'Value') == 1 ) %Then Ioannina is
checked
   index_of_selected=get(handles.listbox8, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_J_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {J}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox9,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_J_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
          text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {J} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
```

```
0,0,7);
           guidata(hObject,handles);
        end
    end

end
%% Kozani Channels
if ( get(handles.checkbox5,'Value') == 1 ) %Then Kozani is
checked
  index_of_selected=get(handles.listbox22, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_K_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {K}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox23,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_K_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
           text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {K} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end
```

```
end
%% Komotini Channels
if ( get(handles.checkbox6,'Value') == 1 ) %Then Komotini is
checked
   index_of_selected=get(handles.listbox10, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_T_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {T}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox11,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_T_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
           text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {T} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end

end
%% Mytilene Channels
if ( get(handles.checkbox7,'Value') == 1 ) %Then Mytilene is
```

```
checked
   index_of_selected=get(handles.listbox6, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_M_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {M}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox7,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_M_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
           text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {M} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end

end
%% Rhode Channels
if ( get(handles.checkbox8,'Value') == 1 ) %Then Rhode is
checked
   index_of_selected=get(handles.listbox18, 'Value');

    for i=1:length(index_of_selected)
```

```
data_to_plot=['handles.RF_A_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {A}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox19,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_A_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
           text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {A} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end

end

%% Neapolis Channels
if ( get(handles.checkbox9,'Value') == 1 ) %Then Neapolis is
checked
    index_of_selected=get(handles.listbox26, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_E_ch',num2str(index_of_selected(i))
];
```

```
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {E}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox27,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_E_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
           text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {E} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end

end
%% Vamos Channels
if ( get(handles.checkbox10,'Value') == 1 ) %Then Vamos is
checked
   index_of_selected=get(handles.listbox16, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_V_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {V}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
```

```
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox17,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_V_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
           text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {V} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end

end
%% Corfu Channels
if ( get(handles.checkbox11,'Value') == 1 ) %Then Corfu is
checked
   index_of_selected=get(handles.listbox24, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_P_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {P}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
```

```
        guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox25,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_P_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
           text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {P} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end

end
%% Atalanti Channels
if ( get(handles.checkbox13,'Value') == 1 ) %Then Atalanti is
checked
    index_of_selected=get(handles.listbox20, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_H_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {H}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
```

```
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox21,
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_H_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
          text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {H} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
           guidata(hObject,handles);
        end
    end

end

%% New Zante Channels
if ( get(handles.checkbox30,'Value') == 1 ) %Then Atalanti is
checked
   index_of_selected=get(handles.listbox28, 'Value');

    for i=1:length(index_of_selected)

data_to_plot=['handles.RF_N_ch',num2str(index_of_selected(i))
];
       text_to_display=['Channel
',num2str(index_of_selected(i)),' {N}'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
       guidata(hObject,handles);
    end

    data_filt = get(handles.checkbox14,'Value');
    if data_filt==1

        index_of_selected_filtered=get(handles.listbox29,
```

```
'Value');
        for i=1:length(index_of_selected_filtered)

data_to_plot=['handles.RF_N_ch',num2str(index_of_selected_fil
tered(i)),'_fil'];
            text_to_display=['Channel
',num2str(index_of_selected_filtered(i)),' {N} Filtered'];

handles.plot_dropdown_names=[handles.plot_dropdown_names;cell
str(text_to_display)];

guiplot(handles.plot_dropdown_names,eval(data_to_plot),0,0,0,
0,0,7);
            guidata(hObject,handles);
        end
    end

end

%% Station Channels Listboxes
%=============================================================
=======
%                     Start of Channels' listboxes
%=============================================================
=======


% --- Executes on selection change in listbox4.
function listbox4_Callback(hObject, eventdata, handles)
% hObject    handle to listbox4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox4 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox4
% list=get(handles.listbox4,'String');

% index_of_selected=get(handles.listbox4, 'Value');
% handles.sl_ch_Z=index_of_selected;

% --- Executes during object creation, after setting all
properties.
function listbox4_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to listbox4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox5.
function listbox5_Callback(hObject, eventdata, handles)
% hObject    handle to listbox5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox5 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox5

% index_of_selected=get(handles.listbox4, 'Value');
% handles.sl_ch_Z_fil=index_of_selected;
% guidata(hObject, handles);

% --- Executes during object creation, after setting all
properties.
function listbox5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


% --- Executes on selection change in listbox6.
function listbox6_Callback(hObject, eventdata, handles)
% hObject    handle to listbox6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox6 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox6


% --- Executes during object creation, after setting all
properties.
function listbox6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox7.
function listbox7_Callback(hObject, eventdata, handles)
% hObject    handle to listbox7 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox7 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox7
```

```
% --- Executes during object creation, after setting all
properties.
function listbox7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox7 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox8.
function listbox8_Callback(hObject, eventdata, handles)
% hObject    handle to listbox8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox8 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox8


% --- Executes during object creation, after setting all
properties.
function listbox8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox9.
function listbox9_Callback(hObject, eventdata, handles)
% hObject    handle to listbox9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox9 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox9


% --- Executes during object creation, after setting all
properties.
function listbox9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox10.
function listbox10_Callback(hObject, eventdata, handles)
% hObject    handle to listbox10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox10 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox10
```

```
% --- Executes during object creation, after setting all
properties.
function listbox10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox11.
function listbox11_Callback(hObject, eventdata, handles)
% hObject    handle to listbox11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox11 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox11


% --- Executes during object creation, after setting all
properties.
function listbox11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
```

```
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox12.
function listbox12_Callback(hObject, eventdata, handles)
% hObject    handle to listbox12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox12 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox12


% --- Executes during object creation, after setting all
properties.
function listbox12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox13.
function listbox13_Callback(hObject, eventdata, handles)
% hObject    handle to listbox13 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox13 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
```

```
from listbox13


% --- Executes during object creation, after setting all
properties.
function listbox13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox13 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox14.
function listbox14_Callback(hObject, eventdata, handles)
% hObject    handle to listbox14 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox14 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox14


% --- Executes during object creation, after setting all
properties.
function listbox14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox14 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
```

```
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox15.
function listbox15_Callback(hObject, eventdata, handles)
% hObject    handle to listbox15 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox15 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox15


% --- Executes during object creation, after setting all
properties.
function listbox15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox15 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox16.
function listbox16_Callback(hObject, eventdata, handles)
% hObject    handle to listbox16 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox16 contents as cell array
```

```
%        contents{get(hObject,'Value')} returns selected item
from listbox16


% --- Executes during object creation, after setting all
properties.
function listbox16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox16 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox17.
function listbox17_Callback(hObject, eventdata, handles)
% hObject    handle to listbox17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox17 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox17


% --- Executes during object creation, after setting all
properties.
function listbox17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
```

```
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox18.
function listbox18_Callback(hObject, eventdata, handles)
% hObject    handle to listbox18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox18 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox18


% --- Executes during object creation, after setting all
properties.
function listbox18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox19.
function listbox19_Callback(hObject, eventdata, handles)
% hObject    handle to listbox19 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
```

```
listbox19 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox19


% --- Executes during object creation, after setting all
properties.
function listbox19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox19 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox20.
function listbox20_Callback(hObject, eventdata, handles)
% hObject    handle to listbox20 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox20 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox20


% --- Executes during object creation, after setting all
properties.
function listbox20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox20 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
```

```
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox21.
function listbox21_Callback(hObject, eventdata, handles)
% hObject    handle to listbox21 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox21 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox21


% --- Executes during object creation, after setting all
properties.
function listbox21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox21 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox22.
function listbox22_Callback(hObject, eventdata, handles)
% hObject    handle to listbox22 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns
listbox22 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox22


% --- Executes during object creation, after setting all
properties.
function listbox22_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox22 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox23.
function listbox23_Callback(hObject, eventdata, handles)
% hObject    handle to listbox23 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox23 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox23


% --- Executes during object creation, after setting all
properties.
function listbox23_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox23 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called
```

```
% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox24.
function listbox24_Callback(hObject, eventdata, handles)
% hObject    handle to listbox24 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox24 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox24


% --- Executes during object creation, after setting all
properties.
function listbox24_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox24 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox25.
function listbox25_Callback(hObject, eventdata, handles)
% hObject    handle to listbox25 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns
listbox25 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox25


% --- Executes during object creation, after setting all
properties.
function listbox25_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox25 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox26.
function listbox26_Callback(hObject, eventdata, handles)
% hObject    handle to listbox26 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox26 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox26


% --- Executes during object creation, after setting all
properties.
function listbox26_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox26 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called
```

```
% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox27.
function listbox27_Callback(hObject, eventdata, handles)
% hObject    handle to listbox27 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox27 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox27


% --- Executes during object creation, after setting all
properties.
function listbox27_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox27 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

%==========================================================
=======
%                    End of Channels' listboxes
%==========================================================
=======
```

```
% --- Executes on selection change in listbox28.
function listbox28_Callback(hObject, eventdata, handles)
% hObject    handle to listbox28 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox28 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox28


% --- Executes during object creation, after setting all
properties.
function listbox28_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox28 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox29.
function listbox29_Callback(hObject, eventdata, handles)
% hObject    handle to listbox29 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox29 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox29


% --- Executes during object creation, after setting all
```

```
properties.
function listbox29_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox29 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu11.
function popupmenu11_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu11 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu11
 selected_indx_methods=get(handles.popupmenu11,'value');
 if selected_indx_methods==1
     set(handles.uipanel12, 'visible', 'on');
     set(handles.uipanel11, 'visible', 'off');
     set(handles.uipanel19, 'visible', 'off');
     set(handles.uipanel17, 'visible', 'off');
 elseif selected_indx_methods==2
     set(handles.uipanel12, 'visible', 'off');
     set(handles.uipanel11, 'visible', 'on');
     set(handles.uipanel19, 'visible', 'off');
     set(handles.uipanel17, 'visible', 'off');
 elseif selected_indx_methods==3
     set(handles.uipanel12, 'visible', 'off');
     set(handles.uipanel11, 'visible', 'off');
     set(handles.uipanel19, 'visible', 'on');
     set(handles.uipanel17, 'visible', 'off');
 else
     set(handles.uipanel12, 'visible', 'off');
```

```
    set(handles.uipanel11, 'visible', 'off');
    set(handles.uipanel19, 'visible', 'off');
    set(handles.uipanel17, 'visible', 'on');
end;


% --- Executes during object creation, after setting all
properties.
function popupmenu11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%=========================================================
%           Start of update channels' names
%=========================================================
% --- Executes on selection change in popupmenu12.
function popupmenu12_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu12 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu12


% --- Executes during object creation, after setting all
properties.
function popupmenu12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
```

```
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit37_Callback(hObject, eventdata, handles)
% hObject    handle to edit37 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit37 as
text
%        str2double(get(hObject,'String')) returns contents of
edit37 as a double


% --- Executes during object creation, after setting all
properties.
function edit37_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit37 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu17.
function popupmenu17_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
```

```
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu17 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu17


% --- Executes during object creation, after setting all
properties.
function popupmenu17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu18.
function popupmenu18_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu18 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu18


% --- Executes during object creation, after setting all
properties.
function popupmenu18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
```

```
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton22.
function pushbutton22_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton22 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)



selected_indx_station=get(handles.popupmenu12,'value');
selected_indx_signal=get(handles.popupmenu17,'value');
selected_indx_channel=get(handles.popupmenu18,'value');
new_ch_name=get(handles.edit37, 'string');

tmp_values=get(handles.listbox4, 'String');
tmp_values(selected_indx_channel)=cellstr(new_ch_name);
set(handles.listbox4,'string',tmp_values);

if selected_indx_station==1
    % then Kalamata {0} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox14, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox14,'string',tmp_values);
    else
        tmp_values=get(handles.listbox15, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox15,'string',tmp_values);
    end
    clear tmp_values;
```

```
elseif selected_indx_station==2
    % then Kefalonia {F} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox12, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox12,'string',tmp_values);
    else
        tmp_values=get(handles.listbox13, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox13,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==3
    % then Komotini {T} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox10, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox10,'string',tmp_values);
    else
        tmp_values=get(handles.listbox11, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox11,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==4
    % then Ioannina {J} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox8, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox8,'string',tmp_values);
    else
        tmp_values=get(handles.listbox9, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox9,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==5
    % then Mytilene {M} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox6, 'String');
```

```
tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox6,'string',tmp_values);
    else
        tmp_values=get(handles.listbox7, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox7,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==6
    % then Zante {Z} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox4, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox4,'string',tmp_values);
    else
        tmp_values=get(handles.listbox5, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox5,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==7
    % then Neapolis {E} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox26, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox26,'string',tmp_values);
    else
        tmp_values=get(handles.listbox27, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox27,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==8
    % then Corfu {P} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox24, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox24,'string',tmp_values);
    else
```

```
        tmp_values=get(handles.listbox25, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox25,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==9
    % then Kozani {K} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox22, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox22,'string',tmp_values);
    else
        tmp_values=get(handles.listbox23, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox23,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==10
    % then Atalanti {H} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox20, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox20,'string',tmp_values);
    else
        tmp_values=get(handles.listbox21, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox21,'string',tmp_values);
    end
    clear tmp_values;
elseif selected_indx_station==11
    % then Rhode {A} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox18, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox18,'string',tmp_values);
    else
        tmp_values=get(handles.listbox19, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox19,'string',tmp_values);
```

```
    end
    clear tmp_values;
elseif selected_indx_station==12
    % then Vamos {V} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox16, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox16,'string',tmp_values);
    else
        tmp_values=get(handles.listbox17, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox17,'string',tmp_values);
    end
    clear tmp_values;
else
    % then New Zante {N} is selected
    if selected_indx_signal==1
        tmp_values=get(handles.listbox28, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox28,'string',tmp_values);
    else
        tmp_values=get(handles.listbox29, 'String');

tmp_values(selected_indx_channel)=cellstr(new_ch_name);
        set(handles.listbox29,'string',tmp_values);
    end
    clear tmp_values;
end
clear selected_indx_station;
clear selected_indx_signal;
clear selected_channel;
clear new_ch_name;
%=======================================================
%           Start of update channels' names
%=======================================================


function edit38_Callback(hObject, eventdata, handles)
% hObject    handle to edit38 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit38 as
text
%        str2double(get(hObject,'String')) returns contents of
edit38 as a double


% --- Executes during object creation, after setting all
properties.
function edit38_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit38 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton31.
function pushbutton31_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton31 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

LAT_TABLE=1;
LONG_TABLE=1;
EQ_M=1;

try
    magn_limit = str2double(get(handles.edit38,'String'));
    for i=1:length(handles.EQ_M)
       if handles.EQ_M(i) > magn_limit
           handles.EQ_LAT(i)
           handles.EQ_LONG(i)
           handles.EQ_M(i)
           sprintf('===============')
             LAT_TABLE=[LAT_TABLE, handles.EQ_LAT(i)];
             LONG_TABLE=[LONG_TABLE, handles.EQ_LONG(i)];
```

```
            EQ_M=[EQ_M,handles.EQ_M(i)];
        end
    end
    GoogleMap(LAT_TABLE,LONG_TABLE,EQ_M)
catch exception
    msgbox('Please type a number');
end
guidata(hObject,handles);
```

15.5 Ανεπτυγμένος κώδικας γραφικής διεπαφής χρήστη για την εκτύπωση γραφημάτων μετρήσεων του ηλεκτομαγνητικού πεδίου απο πολλαπλούς σταθμούς μέτρησης, της πυκνότητας φάσματος ισχύος και της Ασαφούς Εντροπίας.

```
function varargout = guiplot(varargin)
% GUIPLOT MATLAB code for guiplot.fig
%      GUIPLOT, by itself, creates a new GUIPLOT or raises the
existing
%      singleton*.
%
%      H = GUIPLOT returns the handle to a new GUIPLOT or the
handle to
%      the existing singleton*.
%
%      GUIPLOT('CALLBACK',hObject,eventData,handles,...) calls
the local
%       function named CALLBACK in GUIPLOT.M with the given
input arguments.
%
%       GUIPLOT('Property','Value',...) creates a new GUIPLOT
or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%     applied to the GUI before guiplot_OpeningFcn gets called.
An
%        unrecognized property name or invalid value makes
property application
%      stop.  All inputs are passed to guiplot_OpeningFcn via
varargin.
%
%       *See GUI Options on GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run (singleton)".
%
```

```
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help guiplot

% Last Modified by GUIDE v2.5 17-Nov-2012 14:29:36

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @guiplot_OpeningFcn, ...
                   'gui_OutputFcn',  @guiplot_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}]      =      gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before guiplot is made visible.
function   guiplot_OpeningFcn(hObject,   eventdata,   handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to guiplot (see VARARGIN)

% Choose default command line output for guiplot
handles.output = hObject;

% This sets up the initial plot - only do when we are invisible
% so window can get raised using guiplot.
if strcmp(get(hObject,'Visible'),'off')
    plot(0,0);
end

set(hObject, 'color', 'w');
```

```
%handles.plot_commands=[varargin{2}:varargin{length(varargin)
}];
%handles.plot_commands=varargin{2};

tmp_names=varargin{1};
handles.dropdown_names=tmp_names(2:length(tmp_names));

handles.flag=0;

set (handles.popupmenu1, 'String', handles.dropdown_names(:));
set (handles.popupmenu2, 'String', handles.dropdown_names(:));
set (handles.popupmenu3, 'String', handles.dropdown_names(:));
set (handles.popupmenu4, 'String', handles.dropdown_names(:));
set (handles.popupmenu5, 'String', handles.dropdown_names(:));
set (handles.popupmenu6, 'String', handles.dropdown_names(:));
handles.index=length(handles.dropdown_names);
% axes initialization
axes(handles.axes1);
plot(0,0);
cla;
axes(handles.axes2);
plot(0,0);
cla;
axes(handles.axes3);
plot(0,0);
cla;
axes(handles.axes4);
plot(0,0);
cla;
axes(handles.axes5);
plot(0,0);
cla;

%handles.data_for_plot1=varargin{2};
%construct name


%handles.popup_sel_index = get(handles.popupmenu1, 'Value');

%data_name=['handles.data_for_plot',
num2str(length(handles.popup_sel_index)+1)];

%===========
%num_of_iter=evalin('base', var_iter);
%data_name=['handles.data_for_plot', num2str(num_of_iter+1)];
```

```
% tmp_var_in=varargin{2};
% handles.tmp_var_in=tmp_var_in;
% tmp_var_out=num2str(tmp_var_in);
% handles.tmp_var_out=tmp_var_out;
%eval([sprintf(data_name),'= ','[ ', tmp_var_out, '];'])
%==========

try
    clear exist_2_variables;
    handles=rmfield(handles, 'exist_2_var');

catch exception

end
try
    clear exist_3_variables;
    handles=rmfield(handles, 'exist_3_var');
catch exception

end
try
    clear exist_4_variables
    handles=rmfield(handles, 'exist_4_var');
catch exception

end
try
    clear exist_5_variables
    handles=rmfield(handles, 'exist_5_var');
catch exception

end
try
    clear exist_6_variables
    handles=rmfield(handles, 'exist_6_var');
catch exception

end



try
    handles.var1{handles.index}=varargin{2};
catch exeption
    handles.var1{handles.index}=0;
end
```

```
try
    handles.var2{handles.index}=varargin{3};
catch exeption
    handles.var2{handles.index}=0;
end

try
    handles.var3{handles.index}=varargin{4};
catch exeption
    handles.var3{handles.index}=0;
end

try
    handles.var4{handles.index}=varargin{5};
catch exeption
    handles.var4{handles.index}=0;
end

try
    handles.var5{handles.index}=varargin{6};
catch exeption
    handles.var5{handles.index}=0;
end

try
    handles.var6{handles.index}=varargin{7};
catch exeption
    handles.var6{handles.index}=0;
end

%Power Law Flag 0=Not Power Law 1=b(t) 2=log(a) 3=r^2 4=EQ_DATA
5=Intervals
% 6= Fuzzy 7=Channels
try
    handles.var7{handles.index}=varargin{8};
catch exeption
    handles.var7{handles.index}=0;
end
% eval(sprintf(data_name))


% UIWAIT makes guiplot wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% Update handles structure
```

```
guidata(hObject, handles);

% --- Outputs from this function are returned to the command
line.
function varargout = guiplot_OutputFcn(hObject, eventdata,
handles)
% varargout   cell array for returning output args (see
VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)


axes(handles.axes1);
cla;

handles.popup_sel_index = get(handles.popupmenu1, 'Value');
index=handles.popup_sel_index;

start_date=evalin('base', 'startdate');
stop_date=evalin('base', 'stopdate');
xdata=linspace(start_date,stop_date,length(handles.var1{index
}));



%plot EQ_DATA or Intervals or Channels
if   handles.var7{index}==4   ||   handles.var7{index}==5   ||
handles.var7{index}==7
    if handles.var7{index}==4
    % EQ_DATA
        plot(xdata,handles.var1{index}, 'o');axis tight;
    else
        plot(xdata,handles.var1{index});axis tight;
    end
```

```
    datetick('x','mmm-dd','keepticks','keeplimits');
    %Calculate Steps

    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider1,'Min',x_min);
    set(handles.slider1,'Value',x_value);
    set(handles.slider1,'Max',x_max);
    set(handles.slider1,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider2,'Min',x_min);
    set(handles.slider2,'Value',x_value);
    set(handles.slider2,'Max',x_max);
    set(handles.slider2,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider3,'Min',y_min);
    set(handles.slider3,'Value',y_value);
    set(handles.slider3,'Max',y_max);
    set(handles.slider3,'SliderStep',step_y);

    % Y Phase Shifting
```

```
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider4,'Min',y_min);
    set(handles.slider4,'Value',y_value);
    set(handles.slider4,'Max',y_max);
    set(handles.slider4,'SliderStep',step_y);

    guidata(hObject, handles);
end

%plot Fuzzy
if handles.var7{index}==6
    plot(handles.var1{index},'k');axis tight;
    %Calculate Steps
    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider1,'Min',x_min);
    set(handles.slider1,'Value',x_value);
    set(handles.slider1,'Max',x_max);
    set(handles.slider1,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider2,'Min',x_min);
    set(handles.slider2,'Value',x_value);
    set(handles.slider2,'Max',x_max);
```

```
    set(handles.slider2,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider3,'Min',y_min);
    set(handles.slider3,'Value',y_value);
    set(handles.slider3,'Max',y_max);
    set(handles.slider3,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider4,'Min',y_min);
    set(handles.slider4,'Value',y_value);
    set(handles.slider4,'Max',y_max);
    set(handles.slider4,'SliderStep',step_y);

    guidata(hObject, handles);
end


%plot b(t)
if handles.var7{index}==1

plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim',[min((tmp_ylim)) 0]);

    %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];
```

```
    %Calculate Steps Y
    tmp_max=0;
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider1,'Min',x_min);
    set(handles.slider1,'Value',x_value);
    set(handles.slider1,'Max',x_max);
    set(handles.slider1,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider2,'Min',x_min);
    set(handles.slider2,'Value',x_value);
    set(handles.slider2,'Max',x_max);
    set(handles.slider2,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=y_max;
    set(handles.slider3,'Min',y_min);
    set(handles.slider3,'Value',y_value);
    set(handles.slider3,'Max',y_max);
    set(handles.slider3,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider4,'Min',y_min);
    set(handles.slider4,'Value',y_value);
    set(handles.slider4,'Max',y_max);
    set(handles.slider4,'SliderStep',step_y);

    guidata(hObject, handles);
```

```
end
%plot log(a)
if handles.var7{index}==2


plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [tmp_xlim(1) tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim', [min(tmp_ylim) max(tmp_ylim)]);

     %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=max(tmp_ylim);
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider1,'Min',x_min);
    set(handles.slider1,'Value',x_value);
    set(handles.slider1,'Max',x_max);
    set(handles.slider1,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider2,'Min',x_min);
```

```
    set(handles.slider2,'Value',x_value);
    set(handles.slider2,'Max',x_max);
    set(handles.slider2,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=y_max;
    set(handles.slider3,'Min',y_min);
    set(handles.slider3,'Value',y_value);
    set(handles.slider3,'Max',y_max);
    set(handles.slider3,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider4,'Min',y_min);
    set(handles.slider4,'Value',y_value);
    set(handles.slider4,'Max',y_max);
    set(handles.slider4,'SliderStep',step_y);

    guidata(hObject, handles);
end
%plot r^2
if handles.var7{index}==3

plot(handles.var1{index},handles.var2{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var1{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var3{index};
    set(gca, 'ylim',[(tmp_ylim) 1.01]);

      %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=1.01;
    tmp_min=tmp_ylim;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
```

```
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider1,'Min',x_min);
    set(handles.slider1,'Value',x_value);
    set(handles.slider1,'Max',x_max);
    set(handles.slider1,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider2,'Min',x_min);
    set(handles.slider2,'Value',x_value);
    set(handles.slider2,'Max',x_max);
    set(handles.slider2,'SliderStep',step_x);

    % Y Zoom
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=y_max;
    set(handles.slider3,'Min',y_min);
    set(handles.slider3,'Value',y_value);
    set(handles.slider3,'Max',y_max);
    set(handles.slider3,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider4,'Min',y_min);
    set(handles.slider4,'Value',y_value);
    set(handles.slider4,'Max',y_max);
    set(handles.slider4,'SliderStep',step_y);

    guidata(hObject, handles);
end


data_of_dropdown=get(handles.popupmenu1, 'string');
```

```
data_of_dropdown(index);

y=ylabel(data_of_dropdown(index),'Rotation', 90);
pos= get(y, 'pos');
% set(y,'pos',pos+[1.075 0 0]);

% axes_size=get(handles.axes1,'OuterPosition');
% axes_width=axes_size(3);



clear handles.popup_sel_index;
guidata(hObject, handles);

% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider


% tmp_max=get(handles.slider1,'Max');
% tmp_max=tmp_max-get(hObject,'SliderStep');
% set(handles.slider1,'Max',tmp_max);
axes(handles.axes1);

start_point=get(handles.slider2,'Value');
range=abs(get(handles.slider1,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);



% --- Executes on slider movement.
function slider2_Callback(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
```

```
%       get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes1);

start_point=get(handles.slider2,'Value');
range=abs(get(handles.slider1,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);

% --- Executes on slider movement.
function slider3_Callback(hObject, eventdata, handles)
% hObject    handle to slider3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%       get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
% --- Executes on slider movement.
start_point=get(handles.slider4,'Value');
range=abs(get(handles.slider3,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);

function slider4_Callback(hObject, eventdata, handles)
% hObject    handle to slider4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%       get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
start_point=get(handles.slider4,'Value');
range=abs(get(handles.slider3,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);

% ------------------------------------------------------------
---------
function FileMenu_Callback(hObject, eventdata, handles)
% hObject    handle to FileMenu (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --------------------------------------------------------------------
function OpenMenuItem_Callback(hObject, eventdata, handles)
% hObject    handle to OpenMenuItem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
file = uigetfile('*.fig');
if ~isequal(file, 0)
    open(file);
end

% --------------------------------------------------------------------
function PrintMenuItem_Callback(hObject, eventdata, handles)
% hObject    handle to PrintMenuItem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
printdlg(handles.figure1)

% --------------------------------------------------------------------
function CloseMenuItem_Callback(hObject, eventdata, handles)
% hObject    handle to CloseMenuItem (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
selection = questdlg(['Close ' get(handles.figure1,'Name') '?'],...
                     ['Close ' get(handles.figure1,'Name') '...'],...
                     'Yes','No','Yes');
if strcmp(selection,'No')
    return;
end

delete(handles.figure1)


% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
```

```
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu1


%handles.popup_sel_index = get(handles.popupmenu1, 'Value');
% for i=1:length(get(handles.popupmenu1, 'string'))
%     if handles.popup_sel_index==i
%                          %data_name=['handles.data_for_plot',
num2str(handles.popup_sel_index+1)];
%         plot(eval(sprintf(handles.data_name)));axis tight;
%     end
% end



% --- Executes during object creation, after setting all
properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
     set(hObject,'BackgroundColor','white');
end
%set(hObject,           'String',           {'plot(rand(5))',
'plot(sin(1:0.01:25))',    'bar(1:.5:10)',    'plot(membrane)',
'surf(peaks)'});
```

```
% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

%print -f -dtiff 'Figure1'
% load kostas
% figure;
% subplot(3,1,1)
% plot(kostas);axis tight;
% subplot(3,1,2)
% plot(kostas);axis tight;
% subplot(3,1,3)
% plot(kostas);axis tight;
set(handles.pushbutton4, 'Visible', 'off');
set(handles.popupmenu1, 'visible', 'off');
set(handles.popupmenu2, 'visible', 'off');
set(handles.popupmenu3, 'visible', 'off');
set(handles.pushbutton5, 'Visible', 'off');
set(handles.pushbutton6, 'Visible', 'off');
set(handles.pushbutton1, 'Visible', 'off');
set(handles.pushbutton8, 'Visible', 'off');
set(handles.pushbutton7, 'Visible', 'off');
set(handles.popupmenu4, 'visible', 'off');
set(handles.popupmenu5, 'visible', 'off');

set(handles.text3, 'visible', 'off');
set(handles.text8, 'visible', 'off');
set(handles.text13, 'visible', 'off');
set(handles.text18, 'visible', 'off');
set(handles.text4, 'visible', 'off');
set(handles.text9, 'visible', 'off');
set(handles.text14, 'visible', 'off');
set(handles.text19, 'visible', 'off');
set(handles.text10, 'visible', 'off');
set(handles.text15, 'visible', 'off');
set(handles.text20, 'visible', 'off');
set(handles.text6, 'visible', 'off');
set(handles.text11, 'visible', 'off');
set(handles.text16, 'visible', 'off');
set(handles.text21, 'visible', 'off');
set(handles.text7, 'visible', 'off');
set(handles.text12, 'visible', 'off');
set(handles.text17, 'visible', 'off');
```

```
set(handles.text3, 'visible', 'off');
set(handles.text22, 'visible', 'off');
set(handles.text5, 'visible', 'off');

set(handles.slider1, 'visible', 'off');
set(handles.slider2, 'visible', 'off');
set(handles.slider3, 'visible', 'off');
set(handles.slider4, 'visible', 'off');
set(handles.slider5, 'visible', 'off');
set(handles.slider6, 'visible', 'off');
set(handles.slider7, 'visible', 'off');
set(handles.slider8, 'visible', 'off');
set(handles.slider9, 'visible', 'off');
set(handles.slider10, 'visible', 'off');
set(handles.slider11, 'visible', 'off');
set(handles.slider12, 'visible', 'off');
set(handles.slider13, 'visible', 'off');
set(handles.slider14, 'visible', 'off');
set(handles.slider15, 'visible', 'off');
set(handles.slider16, 'visible', 'off');
set(handles.slider17, 'visible', 'off');
set(handles.slider18, 'visible', 'off');
set(handles.slider19, 'visible', 'off');
set(handles.slider20, 'visible', 'off');


set(handles.uipanel1, 'visible', 'off');
set(handles.text24, 'visible', 'off');
set(handles.text23, 'visible', 'off');
set(handles.popupmenu6, 'visible', 'off');
set(handles.edit1, 'visible', 'off');
set(handles.pushbutton9, 'visible', 'off');
tmp_title=get(handles.uipanel1, 'Title');
set(handles.uipanel1,'Title',' ');

%figure('color', 'w');
% set(h, 'box', 'off');
% set(h(1:4),'xcolor','w');

% set(handles.axes1, 'box', 'off')
% set(handles.axes2, 'box', 'off')
% set(handles.axes3, 'box', 'off')

% set(handles.axes1,'xcolor','w');
% set(handles.axes2,'xcolor','w');
set(handles.axes1,'XTick',[]);
```

```
set(handles.axes2,'XTick',[]);
set(handles.axes3,'XTick',[]);
set(handles.axes4,'XTick',[]);

set(handles.axes2,'YAxisLocation', 'right')
set(handles.axes4,'YAxisLocation', 'right')



F = getframe(gcf);
figure(123);image(F.cdata);close figure 123;
format shortg
c=clock;
fix(c);
datename=['Figure_',num2str(c(3)),'_',num2str(c(2)),'_',num2s
tr(c(1)),'_',num2str(c(4)),'_',num2str(c(5)),'_',num2str(c(6)
),'.tiff'];

imwrite(F.cdata, datename);
format;

set(handles.axes1, 'box', 'on')
set(handles.axes2, 'box', 'on')
set(handles.axes3, 'box', 'on')

set(handles.axes1,'xcolor','k');
set(handles.axes2,'xcolor','k');

set(handles.pushbutton4, 'Visible', 'on');
set(handles.popupmenu1, 'visible', 'on');
set(handles.popupmenu2, 'visible', 'on');
set(handles.popupmenu3, 'visible', 'on');
set(handles.pushbutton5, 'Visible', 'on');
set(handles.pushbutton6, 'Visible', 'on');
set(handles.pushbutton1, 'Visible', 'on');
set(handles.pushbutton8, 'Visible', 'on');
set(handles.pushbutton7, 'Visible', 'on');
set(handles.popupmenu4, 'visible', 'on');
set(handles.popupmenu5, 'visible', 'on');

set(handles.text3, 'visible', 'on');
set(handles.text8, 'visible', 'on');
set(handles.text13, 'visible', 'on');
set(handles.text18, 'visible', 'on');
set(handles.text4, 'visible', 'on');
```

```
set(handles.text9, 'visible', 'on');
set(handles.text14, 'visible', 'on');
set(handles.text19, 'visible', 'on');
set(handles.text10, 'visible', 'on');
set(handles.text15, 'visible', 'on');
set(handles.text20, 'visible', 'on');
set(handles.text6, 'visible', 'on');
set(handles.text11, 'visible', 'on');
set(handles.text16, 'visible', 'on');
set(handles.text21, 'visible', 'on');
set(handles.text7, 'visible', 'on');
set(handles.text12, 'visible', 'on');
set(handles.text17, 'visible', 'on');
set(handles.text3, 'visible', 'on');
set(handles.text22, 'visible', 'on');
set(handles.text5, 'visible', 'on');


set(handles.slider1, 'visible', 'on');
set(handles.slider2, 'visible', 'on');
set(handles.slider3, 'visible', 'on');
set(handles.slider4, 'visible', 'on');
set(handles.slider5, 'visible', 'on');
set(handles.slider6, 'visible', 'on');
set(handles.slider7, 'visible', 'on');
set(handles.slider8, 'visible', 'on');
set(handles.slider9, 'visible', 'on');
set(handles.slider10, 'visible', 'on');
set(handles.slider11, 'visible', 'on');
set(handles.slider12, 'visible', 'on');
set(handles.slider13, 'visible', 'on');
set(handles.slider14, 'visible', 'on');
set(handles.slider15, 'visible', 'on');
set(handles.slider16, 'visible', 'on');
set(handles.slider17, 'visible', 'on');
set(handles.slider18, 'visible', 'on');
set(handles.slider19, 'visible', 'on');
set(handles.slider20, 'visible', 'on');

set(handles.uipanel1, 'visible', 'on');
set(handles.text24, 'visible', 'on');
set(handles.text23, 'visible', 'on');
set(handles.popupmenu6, 'visible', 'on');
set(handles.edit1, 'visible', 'on');
set(handles.pushbutton9, 'visible', 'on');
```

```
set(handles.uipanel1,'Title',tmp_title);



% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu2 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu2


% --- Executes during object creation, after setting all
properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,            'String',            {'plot(rand(5))',
'plot(sin(1:0.01:25))',    'bar(1:.5:10)',    'plot(membrane)',
'surf(peaks)'});

% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
```

```
popupmenu3 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu3


% --- Executes during object creation, after setting all
properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,            'String',            {'plot(rand(5))',
'plot(sin(1:0.01:25))',    'bar(1:.5:10)',    'plot(membrane)',
'surf(peaks)'});

% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
axes(handles.axes2);
cla;

%  ===========ylabel==============
% handles.popup_sel_index = get(handles.popupmenu2, 'Value');
% index=handles.popup_sel_index;
%
% y=ylabel(data_of_dropdown(index),'Rotation', 90);
% pos= get(y, 'pos');
% set(y,'pos',pos+[1.075 0 0]);

handles.popup_sel_index = get(handles.popupmenu2, 'Value');
index=handles.popup_sel_index;

start_date=evalin('base', 'startdate');
```

```
stop_date=evalin('base', 'stopdate');
xdata=linspace(start_date,stop_date,length(handles.var1{index
}));


%plot EQ_DATA or Intervals or Channels
if   handles.var7{index}==4   ||   handles.var7{index}==5   ||
handles.var7{index}==7
    if handles.var7{index}==4
    % EQ_DATA
        plot(xdata,handles.var1{index}, 'o');axis tight;
    else
        plot(xdata,handles.var1{index});axis tight;
    end

    datetick('x','mmm-dd','keepticks','keeplimits');
    %Calculate Steps

    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider5,'Min',x_min);
    set(handles.slider5,'Value',x_value);
    set(handles.slider5,'Max',x_max);
    set(handles.slider5,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider6,'Min',x_min);
```

```
    set(handles.slider6,'Value',x_value);
    set(handles.slider6,'Max',x_max);
    set(handles.slider6,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider7,'Min',y_min);
    set(handles.slider7,'Value',y_value);
    set(handles.slider7,'Max',y_max);
    set(handles.slider7,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider8,'Min',y_min);
    set(handles.slider8,'Value',y_value);
    set(handles.slider8,'Max',y_max);
    set(handles.slider8,'SliderStep',step_y);

    guidata(hObject, handles);
end

%plot Fuzzy
if handles.var7{index}==6
    plot(handles.var1{index},'k');axis tight;
    %Calculate Steps
    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
```

```
    x_value= x_max;
    set(handles.slider5,'Min',x_min);
    set(handles.slider5,'Value',x_value);
    set(handles.slider5,'Max',x_max);
    set(handles.slider5,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider6,'Min',x_min);
    set(handles.slider6,'Value',x_value);
    set(handles.slider6,'Max',x_max);
    set(handles.slider6,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider7,'Min',y_min);
    set(handles.slider7,'Value',y_value);
    set(handles.slider7,'Max',y_max);
    set(handles.slider7,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider8,'Min',y_min);
    set(handles.slider8,'Value',y_value);
    set(handles.slider8,'Max',y_max);
    set(handles.slider8,'SliderStep',step_y);

    guidata(hObject, handles);
end


%plot b(t)
if handles.var7{index}==1

plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on
```

```
plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim',[min((tmp_ylim)) 0]);

    %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=0;
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider5,'Min',x_min);
    set(handles.slider5,'Value',x_value);
    set(handles.slider5,'Max',x_max);
    set(handles.slider5,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider6,'Min',x_min);
    set(handles.slider6,'Value',x_value);
    set(handles.slider6,'Max',x_max);
    set(handles.slider6,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=y_max;
    set(handles.slider7,'Min',y_min);
    set(handles.slider7,'Value',y_value);
```

```
    set(handles.slider7,'Max',y_max);
    set(handles.slider7,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider8,'Min',y_min);
    set(handles.slider8,'Value',y_value);
    set(handles.slider8,'Max',y_max);
    set(handles.slider8,'SliderStep',step_y);

    guidata(hObject, handles);

end
%plot log(a)
if handles.var7{index}==2


plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [tmp_xlim(1) tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim', [min(tmp_ylim) max(tmp_ylim)]);

     %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=max(tmp_ylim);
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
```

```
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider5,'Min',x_min);
    set(handles.slider5,'Value',x_value);
    set(handles.slider5,'Max',x_max);
    set(handles.slider5,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider6,'Min',x_min);
    set(handles.slider6,'Value',x_value);
    set(handles.slider6,'Max',x_max);
    set(handles.slider6,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=y_max;
    set(handles.slider7,'Min',y_min);
    set(handles.slider7,'Value',y_value);
    set(handles.slider7,'Max',y_max);
    set(handles.slider7,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider8,'Min',y_min);
    set(handles.slider8,'Value',y_value);
    set(handles.slider8,'Max',y_max);
    set(handles.slider8,'SliderStep',step_y);

    guidata(hObject, handles);
end
%plot r^2
if handles.var7{index}==3

plot(handles.var1{index},handles.var2{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var1{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
```

```
    tmp_ylim=handles.var3{index};
    set(gca, 'ylim',[(tmp_ylim) 1.01]);

      %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=1.01;
    tmp_min=tmp_ylim;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider5,'Min',x_min);
    set(handles.slider5,'Value',x_value);
    set(handles.slider5,'Max',x_max);
    set(handles.slider5,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider6,'Min',x_min);
    set(handles.slider6,'Value',x_value);
    set(handles.slider6,'Max',x_max);
    set(handles.slider6,'SliderStep',step_x);

    % Y Zoom
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=y_max;
    set(handles.slider7,'Min',y_min);
    set(handles.slider7,'Value',y_value);
    set(handles.slider7,'Max',y_max);
    set(handles.slider7,'SliderStep',step_y);

    % Y Phase Shifting
```

```
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider8,'Min',y_min);
    set(handles.slider8,'Value',y_value);
    set(handles.slider8,'Max',y_max);
    set(handles.slider8,'SliderStep',step_y);

    guidata(hObject, handles);
end

data_of_dropdown=get(handles.popupmenu1, 'string');
data_of_dropdown(index);

y=ylabel(data_of_dropdown(index),'Rotation', 90);

clear handles.popup_sel_index;
guidata(hObject, handles);
% --- Executes on slider movement.
function slider5_Callback(hObject, eventdata, handles)
% hObject    handle to slider5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
% --- Executes on slider movement.
axes(handles.axes2);

start_point=get(handles.slider6,'Value');
range=abs(get(handles.slider5,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);

function slider6_Callback(hObject, eventdata, handles)
% hObject    handle to slider6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
```

```
% --- Executes on slider movement.
axes(handles.axes2);

start_point=get(handles.slider6,'Value');
range=abs(get(handles.slider5,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);
function slider7_Callback(hObject, eventdata, handles)
% hObject    handle to slider7 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes1);

start_point=get(handles.slider8,'Value');
range=abs(get(handles.slider7,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);
% --- Executes on slider movement.
function slider8_Callback(hObject, eventdata, handles)
% hObject    handle to slider8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes2);

start_point=get(handles.slider8,'Value');
range=abs(get(handles.slider7,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
axes(handles.axes3);
cla;

handles.popup_sel_index = get(handles.popupmenu3, 'Value');
index=handles.popup_sel_index;

start_date=evalin('base', 'startdate');
stop_date=evalin('base', 'stopdate');
xdata=linspace(start_date,stop_date,length(handles.var1{index
}));


%plot EQ_DATA or Intervals or Channels
if   handles.var7{index}==4   ||   handles.var7{index}==5   ||
handles.var7{index}==7
    if handles.var7{index}==4
    % EQ_DATA
        plot(xdata,handles.var1{index}, 'o');axis tight;
    else
        plot(xdata,handles.var1{index});axis tight;
    end

    datetick('x','mmm-dd','keepticks','keeplimits');
    %Calculate Steps

    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider9,'Min',x_min);
    set(handles.slider9,'Value',x_value);
    set(handles.slider9,'Max',x_max);
    set(handles.slider9,'SliderStep',step_x);
```

```
     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider10,'Min',x_min);
    set(handles.slider10,'Value',x_value);
    set(handles.slider10,'Max',x_max);
    set(handles.slider10,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider11,'Min',y_min);
    set(handles.slider11,'Value',y_value);
    set(handles.slider11,'Max',y_max);
    set(handles.slider11,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider12,'Min',y_min);
    set(handles.slider12,'Value',y_value);
    set(handles.slider12,'Max',y_max);
    set(handles.slider12,'SliderStep',step_y);

    guidata(hObject, handles);
end

%plot Fuzzy
if handles.var7{index}==6
    plot(handles.var1{index},'k');axis tight;
    %Calculate Steps
    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
```

```
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider9,'Min',x_min);
    set(handles.slider9,'Value',x_value);
    set(handles.slider9,'Max',x_max);
    set(handles.slider9,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider10,'Min',x_min);
    set(handles.slider10,'Value',x_value);
    set(handles.slider10,'Max',x_max);
    set(handles.slider10,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider11,'Min',y_min);
    set(handles.slider11,'Value',y_value);
    set(handles.slider11,'Max',y_max);
    set(handles.slider11,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider12,'Min',y_min);
    set(handles.slider12,'Value',y_value);
    set(handles.slider12,'Max',y_max);
    set(handles.slider12,'SliderStep',step_y);

    guidata(hObject, handles);
end
```

```
%plot b(t)
if handles.var7{index}==1

plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    try
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim',[min((tmp_ylim)) 0]);
    catch exception

    end

    %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=0;
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider9,'Min',x_min);
    set(handles.slider9,'Value',x_value);
    set(handles.slider9,'Max',x_max);
    set(handles.slider9,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
```

```
    set(handles.slider10,'Min',x_min);
    set(handles.slider10,'Value',x_value);
    set(handles.slider10,'Max',x_max);
    set(handles.slider10,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=y_max;
    set(handles.slider11,'Min',y_min);
    set(handles.slider11,'Value',y_value);
    set(handles.slider11,'Max',y_max);
    set(handles.slider11,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider12,'Min',y_min);
    set(handles.slider12,'Value',y_value);
    set(handles.slider12,'Max',y_max);
    set(handles.slider12,'SliderStep',step_y);

    guidata(hObject, handles);

end
%plot log(a)
if handles.var7{index}==2


plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [tmp_xlim(1) tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim', [min(tmp_ylim) max(tmp_ylim)]);

     %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
```

```
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=max(tmp_ylim);
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider9,'Min',x_min);
    set(handles.slider9,'Value',x_value);
    set(handles.slider9,'Max',x_max);
    set(handles.slider9,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider10,'Min',x_min);
    set(handles.slider10,'Value',x_value);
    set(handles.slider10,'Max',x_max);
    set(handles.slider10,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    if y_min > y_max
        tmp=y_min;
        y_min=y_max;
        y_max=tmp;
        clear tmp
    end
    y_value=y_max;
    set(handles.slider11,'Min',y_min);
    set(handles.slider11,'Value',y_value);
    set(handles.slider11,'Max',y_max);
    set(handles.slider11,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
```

```
    y_max=max(tmp_ylim);
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider12,'Min',y_min);
    set(handles.slider12,'Value',y_value);
    set(handles.slider12,'Max',y_max);
    set(handles.slider12,'SliderStep',step_y);

    guidata(hObject, handles);
end
%plot r^2
if handles.var7{index}==3

plot(handles.var1{index},handles.var2{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var1{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var3{index};
    set(gca, 'ylim',[(tmp_ylim) 1.01]);

      %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=1.01;
    tmp_min=tmp_ylim;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider9,'Min',x_min);
    set(handles.slider9,'Value',x_value);
    set(handles.slider9,'Max',x_max);
    set(handles.slider9,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
```

```
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider10,'Min',x_min);
    set(handles.slider10,'Value',x_value);
    set(handles.slider10,'Max',x_max);
    set(handles.slider10,'SliderStep',step_x);

    % Y Zoom
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=y_max;
    set(handles.slider11,'Min',y_min);
    set(handles.slider11,'Value',y_value);
    set(handles.slider11,'Max',y_max);
    set(handles.slider11,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider12,'Min',y_min);
    set(handles.slider12,'Value',y_value);
    set(handles.slider12,'Max',y_max);
    set(handles.slider12,'SliderStep',step_y);

    guidata(hObject, handles);
end

data_of_dropdown=get(handles.popupmenu1, 'string');
data_of_dropdown(index);

y=ylabel(data_of_dropdown(index),'Rotation', 90);

clear handles.popup_sel_index;
guidata(hObject, handles);
% --- Executes on slider movement.
function slider9_Callback(hObject, eventdata, handles)
% hObject    handle to slider9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
```

```
% --- Executes on slider movement.
axes(handles.axes3);

start_point=get(handles.slider10,'Value');
range=abs(get(handles.slider9,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);

function slider10_Callback(hObject, eventdata, handles)
% hObject    handle to slider10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
% --- Executes on slider movement.
axes(handles.axes3);

start_point=get(handles.slider10,'Value');
range=abs(get(handles.slider9,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);
function slider11_Callback(hObject, eventdata, handles)
% hObject    handle to slider11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
% --- Executes on slider movement.
axes(handles.axes3);

start_point=get(handles.slider12,'Value');
range=abs(get(handles.slider11,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);
function slider12_Callback(hObject, eventdata, handles)
% hObject    handle to slider12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'Value') returns position of slider
%

axes(handles.axes3);

start_point=get(handles.slider12,'Value');
range=abs(get(handles.slider11,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);
% --- Executes on selection change in popupmenu4.
function popupmenu4_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu4 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu4


% --- Executes during object creation, after setting all
properties.
function popupmenu4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,            'String',            {'plot(rand(5))',
'plot(sin(1:0.01:25))',    'bar(1:.5:10)',    'plot(membrane)',
'surf(peaks)'});

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of
```

```
MATLAB
% handles    structure with handles and user data (see GUIDATA)
axes(handles.axes4);
cla;

handles.popup_sel_index = get(handles.popupmenu4, 'Value');
index=handles.popup_sel_index;

start_date=evalin('base', 'startdate');
stop_date=evalin('base', 'stopdate');
xdata=linspace(start_date,stop_date,length(handles.var1{index
}));


%plot EQ_DATA or Intervals or Channels
if   handles.var7{index}==4   ||   handles.var7{index}==5   ||
handles.var7{index}==7
    if handles.var7{index}==4
    % EQ_DATA
        plot(xdata,handles.var1{index}, 'o');axis tight;
    else
        plot(xdata,handles.var1{index});axis tight;
    end

    datetick('x','mmm-dd','keepticks','keeplimits');
    %Calculate Steps

    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider13,'Min',x_min);
    set(handles.slider13,'Value',x_value);
```

```
    set(handles.slider13,'Max',x_max);
    set(handles.slider13,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider14,'Min',x_min);
    set(handles.slider14,'Value',x_value);
    set(handles.slider14,'Max',x_max);
    set(handles.slider14,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider15,'Min',y_min);
    set(handles.slider15,'Value',y_value);
    set(handles.slider15,'Max',y_max);
    set(handles.slider15,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider16,'Min',y_min);
    set(handles.slider16,'Value',y_value);
    set(handles.slider16,'Max',y_max);
    set(handles.slider16,'SliderStep',step_y);

    guidata(hObject, handles);
end

%plot Fuzzy
if handles.var7{index}==6
    plot(handles.var1{index},'k');axis tight;
    %Calculate Steps
    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
```

```
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider13,'Min',x_min);
    set(handles.slider13,'Value',x_value);
    set(handles.slider13,'Max',x_max);
    set(handles.slider13,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider14,'Min',x_min);
    set(handles.slider14,'Value',x_value);
    set(handles.slider14,'Max',x_max);
    set(handles.slider14,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider15,'Min',y_min);
    set(handles.slider15,'Value',y_value);
    set(handles.slider15,'Max',y_max);
    set(handles.slider15,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider16,'Min',y_min);
    set(handles.slider16,'Value',y_value);
    set(handles.slider16,'Max',y_max);
    set(handles.slider16,'SliderStep',step_y);

    guidata(hObject, handles);
end
```

```
%plot b(t)
if handles.var7{index}==1

plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim',[min((tmp_ylim)) 0]);

    %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=0;
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider13,'Min',x_min);
    set(handles.slider13,'Value',x_value);
    set(handles.slider13,'Max',x_max);
    set(handles.slider13,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider14,'Min',x_min);
    set(handles.slider14,'Value',x_value);
```

```
    set(handles.slider14,'Max',x_max);
    set(handles.slider14,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=y_max;
    set(handles.slider15,'Min',y_min);
    set(handles.slider15,'Value',y_value);
    set(handles.slider15,'Max',y_max);
    set(handles.slider15,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider16,'Min',y_min);
    set(handles.slider16,'Value',y_value);
    set(handles.slider16,'Max',y_max);
    set(handles.slider16,'SliderStep',step_y);

    guidata(hObject, handles);

end
%plot log(a)
if handles.var7{index}==2


plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [tmp_xlim(1) tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim', [min(tmp_ylim) max(tmp_ylim)]);

     %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];
```

```
    %Calculate Steps Y
    tmp_max=max(tmp_ylim);
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider13,'Min',x_min);
    set(handles.slider13,'Value',x_value);
    set(handles.slider13,'Max',x_max);
    set(handles.slider13,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider14,'Min',x_min);
    set(handles.slider14,'Value',x_value);
    set(handles.slider14,'Max',x_max);
    set(handles.slider14,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=y_max;
    set(handles.slider15,'Min',y_min);
    set(handles.slider15,'Value',y_value);
    set(handles.slider15,'Max',y_max);
    set(handles.slider15,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider16,'Min',y_min);
    set(handles.slider16,'Value',y_value);
    set(handles.slider16,'Max',y_max);
    set(handles.slider16,'SliderStep',step_y);
```

```
    guidata(hObject, handles);
end
%plot r^2
if handles.var7{index}==3

plot(handles.var1{index},handles.var2{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var1{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var3{index};
    set(gca, 'ylim',[(tmp_ylim) 1.01]);

      %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=1.01;
    tmp_min=tmp_ylim;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider13,'Min',x_min);
    set(handles.slider13,'Value',x_value);
    set(handles.slider13,'Max',x_max);
    set(handles.slider13,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider14,'Min',x_min);
    set(handles.slider14,'Value',x_value);
    set(handles.slider14,'Max',x_max);
    set(handles.slider14,'SliderStep',step_x);

    % Y Zoom
```

```
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=y_max;
    set(handles.slider15,'Min',y_min);
    set(handles.slider15,'Value',y_value);
    set(handles.slider15,'Max',y_max);
    set(handles.slider15,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider16,'Min',y_min);
    set(handles.slider16,'Value',y_value);
    set(handles.slider16,'Max',y_max);
    set(handles.slider16,'SliderStep',step_y);

    guidata(hObject, handles);
end

data_of_dropdown=get(handles.popupmenu1, 'string');
data_of_dropdown(index);

y=ylabel(data_of_dropdown(index),'Rotation', 90);

clear handles.popup_sel_index;
guidata(hObject, handles);

% --- Executes on slider movement.
function slider13_Callback(hObject, eventdata, handles)
% hObject    handle to slider13 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes4);

start_point=get(handles.slider14,'Value');
range=abs(get(handles.slider13,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);
```

```
% --- Executes on slider movement.
function slider14_Callback(hObject, eventdata, handles)
% hObject    handle to slider14 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes4);

start_point=get(handles.slider14,'Value');
range=abs(get(handles.slider13,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);
% --- Executes on slider movement.
function slider15_Callback(hObject, eventdata, handles)
% hObject    handle to slider15 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes4);

start_point=get(handles.slider16,'Value');
range=abs(get(handles.slider15,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);

% --- Executes on slider movement.
function slider16_Callback(hObject, eventdata, handles)
% hObject    handle to slider16 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes4);

start_point=get(handles.slider16,'Value');
```

```
range=abs(get(handles.slider15,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);

% --- Executes on selection change in popupmenu5.
function popupmenu5_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu5 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu5


% --- Executes during object creation, after setting all
properties.
function popupmenu5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,            'String',            {'plot(rand(5))',
'plot(sin(1:0.01:25))',    'bar(1:.5:10)',    'plot(membrane)',
'surf(peaks)'});

% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
axes(handles.axes5);
cla;
```

```
handles.popup_sel_index = get(handles.popupmenu5, 'Value');
index=handles.popup_sel_index;

start_date=evalin('base', 'startdate');
stop_date=evalin('base', 'stopdate');
xdata=linspace(start_date,stop_date,length(handles.var1{index
}));

%plot EQ_DATA or Intervals or Channels
if   handles.var7{index}==4   ||   handles.var7{index}==5   ||
handles.var7{index}==7

    if handles.var7{index}==4
    % EQ_DATA
        plot(xdata,handles.var1{index}, 'o');axis tight;
    else
        plot(xdata,handles.var1{index});axis tight;
    end


    datetick('x','mmm-dd','keepticks','keeplimits');
    %Calculate Steps

    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number)  (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider17,'Min',x_min);
    set(handles.slider17,'Value',x_value);
    set(handles.slider17,'Max',x_max);
    set(handles.slider17,'SliderStep',step_x);

     % X Phase Shifting
```

```
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider18,'Min',x_min);
    set(handles.slider18,'Value',x_value);
    set(handles.slider18,'Max',x_max);
    set(handles.slider18,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider19,'Min',y_min);
    set(handles.slider19,'Value',y_value);
    set(handles.slider19,'Max',y_max);
    set(handles.slider19,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider20,'Min',y_min);
    set(handles.slider20,'Value',y_value);
    set(handles.slider20,'Max',y_max);
    set(handles.slider20,'SliderStep',step_y);

    guidata(hObject, handles);
end

%plot Fuzzy
if handles.var7{index}==6
    plot(handles.var1{index},'k');axis tight;
    %Calculate Steps
    tmp_max=length(handles.var1{index});
    tmp_min=1;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    tmp_max=max(handles.var1{index});
    tmp_min=min(handles.var1{index});
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];
```

```
    % X Zoom
    x_min=1;
    x_max=length(handles.var1{index}-1)/2; % 1 is the minimum
value of x-axis
    x_value= x_max;
    set(handles.slider17,'Min',x_min);
    set(handles.slider17,'Value',x_value);
    set(handles.slider17,'Max',x_max);
    set(handles.slider17,'SliderStep',step_x);

     % X Phase Shifting
    x_min=1;
    x_max=length(handles.var1{index});
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider18,'Min',x_min);
    set(handles.slider18,'Value',x_value);
    set(handles.slider18,'Max',x_max);
    set(handles.slider18,'SliderStep',step_x);

    % Y Zoom
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=y_max;
    set(handles.slider19,'Min',y_min);
    set(handles.slider19,'Value',y_value);
    set(handles.slider19,'Max',y_max);
    set(handles.slider19,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min(handles.var1{index});
    y_max=max(handles.var1{index});
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider20,'Min',y_min);
    set(handles.slider20,'Value',y_value);
    set(handles.slider20,'Max',y_max);
    set(handles.slider20,'SliderStep',step_y);

    guidata(hObject, handles);
end


%plot b(t)
if handles.var7{index}==1
```

```
plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim',[min((tmp_ylim)) 0]);

    %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=0;
    tmp_min=min((tmp_ylim));
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider17,'Min',x_min);
    set(handles.slider17,'Value',x_value);
    set(handles.slider17,'Max',x_max);
    set(handles.slider17,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider18,'Min',x_min);
    set(handles.slider18,'Value',x_value);
    set(handles.slider18,'Max',x_max);
    set(handles.slider18,'SliderStep',step_x);

    % Y Zoom
```

```
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=y_max;
    set(handles.slider19,'Min',y_min);
    set(handles.slider19,'Value',y_value);
    set(handles.slider19,'Max',y_max);
    set(handles.slider19,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=0-eps;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider20,'Min',y_min);
    set(handles.slider20,'Value',y_value);
    set(handles.slider20,'Max',y_max);
    set(handles.slider20,'SliderStep',step_y);

    guidata(hObject, handles);

end
%plot log(a)
if handles.var7{index}==2


plot(handles.var1{index},handles.var2{index},'r^','markersize
',4);%axis tight;
    hold on

plot(handles.var3{index},handles.var4{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var5{index};
    set(gca,'xlim', [tmp_xlim(1) tmp_xlim(end)]);
    tmp_ylim=handles.var6{index};
    set(gca, 'ylim', [min(tmp_ylim) max(tmp_ylim)]);

     %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=max(tmp_ylim);
    tmp_min=min((tmp_ylim));
```

```
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider17,'Min',x_min);
    set(handles.slider17,'Value',x_value);
    set(handles.slider17,'Max',x_max);
    set(handles.slider17,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider18,'Min',x_min);
    set(handles.slider18,'Value',x_value);
    set(handles.slider18,'Max',x_max);
    set(handles.slider18,'SliderStep',step_x);

    % Y Zoom
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=y_max;
    set(handles.slider19,'Min',y_min);
    set(handles.slider19,'Value',y_value);
    set(handles.slider19,'Max',y_max);
    set(handles.slider19,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=min((tmp_ylim));
    y_max=max(tmp_ylim);
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider20,'Min',y_min);
    set(handles.slider20,'Value',y_value);
    set(handles.slider20,'Max',y_max);
    set(handles.slider20,'SliderStep',step_y);

    guidata(hObject, handles);
end
%plot r^2
if handles.var7{index}==3
```

```
plot(handles.var1{index},handles.var2{index},'bo','markersize
',4);%axis tight;
    tmp_xlim=handles.var1{index};
    set(gca,'xlim', [ tmp_xlim(1)  tmp_xlim(end)]);
    tmp_ylim=handles.var3{index};
    set(gca, 'ylim',[(tmp_ylim) 1.01]);

      %Calculate Steps X
    tmp_max=tmp_xlim(end);
    tmp_min=tmp_xlim(1);
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_x=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    %Calculate Steps Y
    tmp_max=1.01;
    tmp_min=tmp_ylim;
    tmp_zeros_number=numel(num2str(ceil(tmp_max)))+2;
    step_y=[(tmp_max-tmp_min)/(10^tmp_zeros_number) (tmp_max-
tmp_min)/(10^tmp_zeros_number)];

    % X Zoom
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end)/2; % 1 is the minimum value of x-axis
    x_value= x_max;
    set(handles.slider17,'Min',x_min);
    set(handles.slider17,'Value',x_value);
    set(handles.slider17,'Max',x_max);
    set(handles.slider17,'SliderStep',step_x);

     % X Phase Shifting
    x_min=tmp_xlim(1);
    x_max=tmp_xlim(end);
    x_value=(x_max-x_min)/2 +x_min;   % X Center Point Index
Position
    set(handles.slider18,'Min',x_min);
    set(handles.slider18,'Value',x_value);
    set(handles.slider18,'Max',x_max);
    set(handles.slider18,'SliderStep',step_x);

    % Y Zoom
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=y_max;
    set(handles.slider19,'Min',y_min);
```

```
    set(handles.slider19,'Value',y_value);
    set(handles.slider19,'Max',y_max);
    set(handles.slider19,'SliderStep',step_y);

    % Y Phase Shifting
    y_min=tmp_ylim;
    y_max=1.01;
    y_value=(y_max-y_min)/2+y_min;   % Y Center Point Index
Position
    set(handles.slider20,'Min',y_min);
    set(handles.slider20,'Value',y_value);
    set(handles.slider20,'Max',y_max);
    set(handles.slider20,'SliderStep',step_y);

    guidata(hObject, handles);
end

data_of_dropdown=get(handles.popupmenu1, 'string');
data_of_dropdown(index);

ylabel(data_of_dropdown(index),'Rotation', 90);


clear handles.popup_sel_index;
guidata(hObject, handles);
% --- Executes on slider movement.
function slider17_Callback(hObject, eventdata, handles)
% hObject    handle to slider17 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
% --- Executes on slider movement.
axes(handles.axes5);

start_point=get(handles.slider18,'Value');
range=abs(get(handles.slider17,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);
function slider18_Callback(hObject, eventdata, handles)
% hObject    handle to slider18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
% --- Executes on slider movement.
axes(handles.axes5);

start_point=get(handles.slider18,'Value');
range=abs(get(handles.slider17,'Value'))/2;
set(gca,'xlim', [start_point-range  start_point+range]);
guidata(hObject, handles);
function slider19_Callback(hObject, eventdata, handles)
% hObject    handle to slider19 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes5);

start_point=get(handles.slider20,'Value');
range=abs(get(handles.slider19,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);
% --- Executes on slider movement.
function slider20_Callback(hObject, eventdata, handles)
% hObject    handle to slider20 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider
axes(handles.axes5);

start_point=get(handles.slider20,'Value');
range=abs(get(handles.slider19,'Value'))/2;
set(gca,'ylim', [start_point-range  start_point+range]);
guidata(hObject, handles);

% --- Executes during object creation, after setting all
properties.
```

```
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
% --- Executes during object creation, after setting all
properties.
function slider2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes during object creation, after setting all
properties.
function slider3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes during object creation, after setting all
properties.
function slider4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes during object creation, after setting all
properties.
function slider5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes during object creation, after setting all
properties.
function slider6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
```

```
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes during object creation, after setting all
properties.
function slider7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider7 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes during object creation, after setting all
properties.
function slider8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes during object creation, after setting all
properties.
function slider9_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to slider9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes during object creation, after setting all
properties.
function slider10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes during object creation, after setting all
properties.
function slider11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
```

```
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes during object creation, after setting all
properties.
function slider12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes during object creation, after setting all
properties.
function slider13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider13 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes during object creation, after setting all
properties.
function slider14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider14 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called
```

```
% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes during object creation, after setting all
properties.
function slider15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider15 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes during object creation, after setting all
properties.
function slider16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider16 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes during object creation, after setting all
properties.
function slider17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider17 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes during object creation, after setting all
properties.
function slider18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider18 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes during object creation, after setting all
properties.
function slider19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider19 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes during object creation, after setting all
properties.
function slider20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider20 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as
text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double


% --- Executes during object creation, after setting all
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu6.
function popupmenu6_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenu6 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from popupmenu6


% --- Executes during object creation, after setting all
properties.
function popupmenu6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
selected_index=get(handles.popupmenu6,'value');
new_name=get(handles.edit1, 'String');
handles.dropdown_names(selected_index)=cellstr(new_name);

set (handles.popupmenu1, 'String', handles.dropdown_names(:));
```

```
set (handles.popupmenu2, 'String', handles.dropdown_names(:));
set (handles.popupmenu3, 'String', handles.dropdown_names(:));
set (handles.popupmenu4, 'String', handles.dropdown_names(:));
set (handles.popupmenu5, 'String', handles.dropdown_names(:));
set (handles.popupmenu6, 'String', handles.dropdown_names(:));
guidata(hObject, handles);
```

15.6 Ανεπτυγμένος κώδικας για την αναπαράσταση σεισμών σε χάρτη

```
function varargout = GoogleMap(varargin)

gui_Singleton = 1;


gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',
@GoogleMap_OpeningFcn, ...
                   'gui_OutputFcn',  @GoogleMap_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}]      =      gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State, varar
end

function  GoogleMap_OpeningFcn(hObject,  eventdata,  handles,
varargin)

set(hObject, 'color', 'w');
axis(handles.axes1);

handles.key='86aa5dcc5a22d6c74676e00679e68f058d6a2b98';

EQ_LAT=varargin{1};
handles.EQ_LAT=EQ_LAT(2:length(EQ_LAT));
```

```
EQ_LONG=varargin{2};
handles.EQ_LONG=EQ_LONG(2:length(EQ_LONG));

EQ_M=varargin{3};
handles.EQ_M=EQ_M(2:length(EQ_M));

link_str='http://maps.google.com/maps/api/staticmap?center=Gr
eece&zoom=5&size=512x512&maptype=hybrid';
link_str_end='&sensor=false';


tmp_values=get(handles.listbox1, 'String');

for i=1:length(handles.EQ_LAT)

link_str=[link_str,'&markers=color:blue|label:',num2str(i),'|
',num2str(handles.EQ_LAT(i)),',',num2str(handles.EQ_LONG(i)),
link_str_end];
   list_str=['Point                          No.',num2str(i),'
M=',num2str(handles.EQ_M(i))];
   tmp_values=[get(handles.listbox1, 'String'); list_str];
   set(handles.listbox1, 'String',tmp_values);
end

[I map]=imread(link_str,'png');
RGB=ind2rgb(I,map);
imshow(RGB);


% Choose default command line output for GoogleMap
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

function varargout = GoogleMap_OutputFcn(hObject, eventdata,
handles)
% varargout   cell array for returning output args (see
VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
```

```
varargout{1} = handles.output;

% --- Executes on button press in btn_up.
function btn_up_Callback(hObject, eventdata, handles)
% hObject    handle to btn_up (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
axis(handles.axes1);
handles.key='86aa5dcc5a22d6c74676e00679e68f058d6a2b98';



cx=str2num(get(handles.edit_x,'String'));
cy=str2num(get(handles.edit_y,'String'));
cx=cx+0.1;
set(handles.edit_x,'String',cx);

zoom=int2str(get(handles.slider2,'Value'));

show_map(num2str(cx),num2str(cy),zoom,handles.key);

%address=strcat('http://maps.google.com/staticmap?center=',cx
,...
%                                                    ',',cy,...
%                                                '&zoom=',zoom,...
%                                      '&size=512x512&key=',key);
%[I map]=imread(address,'gif');
%RGB=ind2rgb(I,map);
%imshow(RGB);

guidata(hObject, handles);



function show_map(x,y,zoom,key)

address=strcat('http://maps.google.com/staticmap?center=',x,.
..

',',y,...

'&zoom=',zoom,...
                                      '&size=512x512&key=',key);
[I map]=imread(address,'gif');
```

```
RGB=ind2rgb(I,map);
imshow(RGB);



% --- Executes on button press in btn_left.
function btn_left_Callback(hObject, eventdata, handles)
% hObject    handle to btn_left (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
axis(handles.axes1);



cx=str2num(get(handles.edit_x,'String'));
cy=str2num(get(handles.edit_y,'String'));
cy=cy-0.1;
set(handles.edit_y,'String',cy);

zoom=int2str(get(handles.slider2,'Value'));

show_map(num2str(cx),num2str(cy),zoom,handles.key);
guidata(hObject, handles);



% --- Executes on button press in btn_right.
function btn_right_Callback(hObject, eventdata, handles)
% hObject    handle to btn_right (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
axis(handles.axes1);


handles.key='86aa5dcc5a22d6c74676e00679e68f058d6a2b98';

cx=str2num(get(handles.edit_x,'String'));
cy=str2num(get(handles.edit_y,'String'));
cy=cy+0.1;
set(handles.edit_y,'String',cy);
```

```
zoom=int2str(get(handles.slider2,'Value'));

show_map(num2str(cx),num2str(cy),zoom,handles.key);
guidata(hObject, handles);



% --- Executes on button press in btn_down.
function btn_down_Callback(hObject, eventdata, handles)
% hObject    handle to btn_down (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

axis(handles.axes1);

handles.key='86aa5dcc5a22d6c74676e00679e68f058d6a2b98';

cx=str2num(get(handles.edit_x,'String'));
cy=str2num(get(handles.edit_y,'String'));
cx=cx-0.1;
set(handles.edit_x,'String',cx);

zoom=int2str(get(handles.slider2,'Value'));

show_map(num2str(cx),num2str(cy),zoom,handles.key);
guidata(hObject, handles);



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as
text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double


% --- Executes during object creation, after setting all
```

```
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as
text
%        str2double(get(hObject,'String')) returns contents of
edit2 as a double


% --- Executes during object creation, after setting all
properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider


% --- Executes during object creation, after setting all
properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider2_Callback(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider

axis(handles.axes1);

handles.key='86aa5dcc5a22d6c74676e00679e68f058d6a2b98';
clear link_str
```

```
zoom=int2str(get(handles.slider2,'Value'));

link_str='http://maps.google.com/maps/api/staticmap?center=Gr
eece&zoom=';
link_str2='&size=512x512&maptype=hybrid';
link_str_end='&sensor=false';

link_str=[link_str,zoom,link_str2];

for i=1:length(handles.EQ_LAT)

link_str=[link_str,'&markers=color:blue|label:',num2str(i),'|
',num2str(handles.EQ_LAT(i)),',',num2str(handles.EQ_LONG(i)),
link_str_end];
end

[I map]=imread(link_str,'png');
RGB=ind2rgb(I,map);
imshow(RGB);


guidata(hObject, handles);




% --- Executes during object creation, after setting all
properties.
function slider2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if                    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
function edit_x_Callback(hObject, eventdata, handles)
% hObject    handle to edit_x (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit_x as
text
%        str2double(get(hObject,'String')) returns contents of
edit_x as a double


% --- Executes during object creation, after setting all
properties.
function edit_x_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit_x (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit_y_Callback(hObject, eventdata, handles)
% hObject    handle to edit_y (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit_y as
text
%        str2double(get(hObject,'String')) returns contents of
edit_y as a double


% --- Executes during object creation, after setting all
properties.
```

```
function edit_y_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit_y (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles      empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in btn_init.
function btn_init_Callback(hObject, eventdata, handles)
% hObject    handle to btn_init (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
listbox1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from listbox1


% --- Executes during object creation, after setting all
properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
```

```
% handles      empty - handles not created until after all
CreateFcns called

% Hint: listbox controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if     ispc     &&     isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.slider2, 'visible', 'off');
set(handles.text4, 'visible', 'off');
set(handles.edit_x, 'visible', 'off');
set(handles.edit_y, 'visible', 'off');
set(handles.btn_up, 'visible', 'off');
set(handles.btn_down, 'visible', 'off');
set(handles.btn_left, 'visible', 'off');
set(handles.btn_right, 'visible', 'off');
set(handles.listbox1, 'visible', 'off');
set(handles.pushbutton6, 'visible', 'off');

F = getframe(gcf);
figure(1234);image(F.cdata);close figure 1234;
figname='GoogleMap_of_Magnitudes.tiff';
imwrite(F.cdata, figname);

set(handles.slider2, 'visible', 'on');
set(handles.text4, 'visible', 'on');
set(handles.edit_x, 'visible', 'on');
set(handles.edit_y, 'visible', 'on');
set(handles.btn_up, 'visible', 'on');
set(handles.btn_down, 'visible', 'on');
set(handles.btn_left, 'visible', 'on');
set(handles.btn_right, 'visible', 'on');
set(handles.listbox1, 'visible', 'on');
set(handles.pushbutton6, 'visible', 'on');
```

15. 7 Παράθεση χρησιμοποιούμενου κώδικα πυκνότητας φάσματος ισχύος του Πανεπιστημίου του Κολοράντο

```
function        [f,log_a,b,rr,tt,yy,wave_j,lin_fit]        =
powerlawPAOS(y,dt,step,...

window_size,sp_ty,nr_sc,sc_sp,mother,r_critical,plot_flag,...
    print_shift,octave_flag,fast_flag)

%function        [f,log_a,b,rr,tt,yy,wave_j,lin_fit]        =
powerlawPAOSwt_m1(y,dt,...
%   step,window_size,sp_ty,nr_sc,sc_sp,mother,r_critical,...
%   plot_flag,print_shift,octave_flag,fast_flag)
%
% Power Law fitting function, using the wavelet.m function
found at
%  http://paos.colorado.edu/research/wavelets/  for  wavelet
transform.
% It is designed to be flexible in use, providing different
possibilities
% of application and can be used either on GNU-Octave or on
% Matlab
% It is property of the ????? team.
%
% INPUT ARGUMENTS:
%
% y = measured time-series [vector of real numbers]
% dt = sampling period [in seconds]
% step = rolling step for the estimation of power law fitting
[in samples]
% window_size = window_size for wavelet transform calculation
[in samples]
% sp_ty = spacing type, 'log', or 'linear'
% nr_sc = number of scales, default = 25
% sc_sp = scale spacing, default = 0.25 for 'log' spacing, use
integer for
%         linear spacing
% mother = wavelet name, use any of the wave_bases.m valid
wavelet names,
%          like 'MORLET' or 'PAUL' or 'DOG'
% plot_flag = flag to check plotting inside the function.
%            If = 0 do not plot
%               If >1 plot (plot_flag) number of figures (if
available)
```

```
% print_shift = how many iterations to shift before checking
to print %
% octave_flag = flag to check Octave run. If equals 1 then run
is in Octave
%                                          If equals 0 then run
is in MATLAB
% fast_flag = flag to indicate fast version of the routine,
where memory
%             consuming computations (wave_j, lin_fit and yy)
are ommited
%
% OUTPUT ARGUMENTS:
%
% f = frequencies corresponding to central frequencies of
scales
% log_a = vector of the log10 of factor "a" of the fitted power
law
%           "S(f)= a * f^b" through the time-series scanning
with "step"
%          and "window_size"
% b = vector of the exponents "b" of the fitted power law
"S(f)= a * f^b"
%         through the time-series scanning with "step" and
"window_size"
% rr = square (r^2) of the Spearman correlation coefficients
(showing the
%      quality of the linear -on log-log scale- curve fitting)
% tt = table of all time vectors corresponding to the processed
%      time-series windows [hours]
% yy = table of all signal vectors corresponding to the
processed
%      time-series windows
% wave_j = table of the wavelet transform coefficients for all
iterations
% lin_fit = linear fit parameters P for each iteration
(contains log_a and
%           b, but useful for applying "polyfit.m")


if (nargin < 13), fast_flag = -1; end
if (nargin < 12), octave_flag = -1; end
if (nargin < 11), print_shift = -1; end
if (nargin < 10), plot_flag = -1; end
if (nargin < 9), r_critical= -1;end
if (nargin < 8), mother = -1; end
if (nargin < 7), sc_sp = -1; end
```

```
if (nargin < 6), nr_sc = -1; end
if (nargin < 5), sp_ty = 'log'; end
if (nargin < 4), window_size = -1; end
if (nargin < 3), step = -1; end
if (nargin < 2)
    error('At least an input vector y and sampling time dt MUST
be provided')
end

if (fast_flag == -1), fast_flag = 1; end
if (octave_flag == -1), octave_flag = 1; end
if (print_shift == -1), print_shift = 50; end
if (plot_flag == -1), plot_flag = 0; end
if (r_critical == -1), r_critical = 0.95; end
if (mother == -1), mother = 'MORLET'; end
if (sc_sp == -1), sc_sp = 0.25; end
if (nr_sc == -1), nr_sc = 25; end
if (window_size == -1), window_size = 128; end
if (step == -1), step = 1; end

pad = 0;
segment_size=length(y);
t = [1:segment_size];
iters = (segment_size-window_size)/step;
iterations = iters
range = 60;

N_of_passes = 0;
fs=1/dt;
if (octave_flag == 0)
    wvname = 'morl';
end
colormp = pink; %jet;
%t = (1:Nr)*dt/3600; % construct time vector in hours
range = 50;
bw = 10;

tic;

for j=1:iters


    if (fast_flag == 0)

        yy(j,:) = y((j-1)*step+[1:window_size]);
```

```
        [WAVE,PERIOD,SCALE,COI]                                    =
wavelet2(yy(j,:),dt,pad,sc_sp,2*dt,nr_sc-1,mother,sp_ty);
        wave_j(j,:,:) = WAVE;
        LOG_Power_Spectrum=log10((sum(abs(WAVE').^2)))';
        f = 1./PERIOD;
        log_f=log10(1./PERIOD');
        [P]=polyfit(log_f,LOG_Power_Spectrum,1);
        lin_fit(j,:) = P;

    else

        [WAVE,PERIOD,SCALE,COI]        =        wavelet2(y((j-
1)*step+[1:window_size]),dt,pad,sc_sp,2*dt,nr_sc-
1,mother,sp_ty);
        LOG_Power_Spectrum=log10((sum(abs(WAVE').^2)))';
%         f = 1./PERIOD(1:end-0);
        f                  =                  1./PERIOD(1:end-
8);                                                          %Rn
%         log_f=log10(1./PERIOD(1:end-0)');
        log_f=log10(1./PERIOD(1:end-
8)');                                          %Rn
%         [P]=polyfit(log_f,LOG_Power_Spectrum(1:end-0),1);
        [P]=polyfit(log_f,LOG_Power_Spectrum(1:end-
8),1);          %Rn

        papap=1;

    end

    if (octave_flag == 1)

%         r=spearman(log_f,LOG_Power_Spectrum(1:end-0));
        r=spearman(log_f,LOG_Power_Spectrum(1:end-
8));        %Rn

    else

%                       r=corr(log_f,LOG_Power_Spectrum(1:end-
0),'type','Spearman');
        r=corr(log_f,LOG_Power_Spectrum(1:end-
8),'type','Spearman');        %Rn

    end

    b(j) = P(1);
    log_a(j) = P(2);
```

```
    rr(j) = r.^2;
    tt(j)=t(window_size+j*step)*dt/3600;


    %%
%     figure
%             fit = polyval(P,log_f);
%         plot(log_f,LOG_Power_Spectrum(1:end-25),'r*',...
%             log_f,fit,'b-');
%         xlabel('Log(Frequency)');
%         ylabel('Log(S(f))');
%         grid;

    %%

    %     construct (plot_flag) WAVE vectors for plotting
purposes
    if (plot_flag >= 1 && N_of_passes <= plot_flag &&
j>1+print_shift)
        if    (abs(b(j-print_shift))    >    1    &&    rr(j-
print_shift)>=r_critical)
            N_of_passes=N_of_passes+1;
            initial_j(N_of_passes)=j;
            polynomial_in_window(N_of_passes,:)=P;
            beta_in_window(N_of_passes,:)=b(j);
            log_a_in_window(N_of_passes,:)=log_a(j);
            r_square_in_window(N_of_passes,:)=rr(j);

log_ps_in_window(N_of_passes,:)=LOG_Power_Spectrum;
            log_f_in_window(N_of_passes,:)=log_f;
            wavelet_transform(N_of_passes,:,:)= WAVE;
        end
    end
    %memory

end
% plot (plot_iflag) examples of powerlaw fit and scalograms if
plot_flag >=1

if (plot_flag >= 1 && N_of_passes ~= 0)

    for i=1:plot_flag
        % for plotting purpose
        for k = 1:N_of_passes
            wave_tran = squeeze(wavelet_transform(k,:,:));
        end
```

```
        for l = 1:window_size
            t_in_window(l)=t(initial_j(N_of_passes)+(l-
1))*dt/3600;  % in hours
            y_in_window(l)=y(initial_j(N_of_passes)+(l-1));
        end
        % end for plotting purpose

        figure(i+1000);
        subplot(3,1,1);
%                                                       fit      =
polyval(polynomial_in_window(i,:),log_f_in_window(i,:));
%
plot(log_f_in_window(i,:),log_ps_in_window(i,:),'r*',...
%             log_f_in_window(i,:),fit,'b-');

                fit                                                =
polyval(polynomial_in_window(i,:),log_f_in_window(i,:));     %
Rn

plot(log_f_in_window(i,:),log_ps_in_window(i,:),'r*',...
                    %  Rn
            log_f_in_window(i,:),fit,'b-
');                                                           %
Rn

        xlabel('Log(Frequency)');
        ylabel('Log(S(f))');
        grid;
        if (octave_flag == 0)% Create textbox in Matlab
            annotation('textbox',[0.6635     0.7857     0.2044
0.09762],...
                'String',{['log(a)=',
num2str(log_a_in_window(i,:),3),...

sprintf('\n'),'b=',num2str(abs(beta_in_window(i,:)),3),...
                ' \pm ',num2str(std(beta_in_window),2),...

sprintf('\n'),'r^2=',num2str(r_square_in_window(i,:),2),...

sprintf('\n'),'t=',num2str(t_in_window(1),2),' - ',...
                num2str(t_in_window(end),2),
'h']},'FitBoxToText','on');
        else % Create legent in Octave
            legend(strcat('log_a=',
num2str(log_a_in_window(i,:)),...
                ', b=',num2str(beta_in_window(i,:)),...
```

```
                ', r^2=',num2str(r_square_in_window(i,:))));
        end
        title(strcat('Power    Law    Fit    and    Scalogram,
',num2str(window_size),...
            ' Window size, ', num2str(step),' Step'));

        subplot(3,1,2);
        plot (t_in_window,y_in_window,'r');
        set(gca,'xlim',[t_in_window(1) t_in_window(end)]);
        xlabel('Time (h)');
        ylabel('Amplitude')
        grid;

        S=wave_tran.*wave_tran;
        maxS = max(max(abs(S))); minS = 10^-8*maxS;
        dBS = 10*log10(abs(S));
        %dBS = 20*log10(max(abs(S), minS));
        maxdBS = max(max(dBS));
        mindBS = min(min(dBS));
        %dBS = min(maxdBS, max(maxdBS-range, dBS));
        %dBS = max(mindBS, min(mindBS+range, dBS));

        subplot(3,1,3)
        yscale='log';
        mesh(t_in_window,f,dBS);
        set(gca,'YScale',yscale);
        colormap(colormp);
        ylim([min(f) max(f)]);
        ylabel('f(Hz)');
        xlabel('Time (h)');
        set(gca,'TickDir','out')
        view(2);
        axis tight;

        if (octave_flag == 0)
            title(['Signal ' wvname ' Scalogram (',...
                num2str(range),'dB,   ',yscale,'   scale,   bw
parameter=',...
                num2str(bw),')',sprintf('\n'),...
                'dB of Absolute Value of Ca,b coefficients']);
        end

        set(gca,'TickDir','out')
        axis tight;

        if (octave_flag == 1),
```

```
            colorbar('EastOutside');
            print(num2str(i+1000),'-dgif')
        else
            colorbar([0.93 0.11 0.02 0.7742],'FontSize',8);
            figure_name_1000i = ['Figure',num2str(1000+i)];
            print('-dtiff', figure_name_1000i);
        end

        figure(i+2000);
        yscale='log';
        mesh(t_in_window,f,dBS);
        set(gca,'YScale',yscale);
        colormp=pink;
        colormap(colormp);
        ylim([min(f) fs/2]);
        ylabel('f(Hz)');
        xlabel('Time (h)');
        set(gca,'TickDir','out')
        axis tight;
        title ('3D Wavelet Tranform');

        if (octave_flag == 1),
            print(num2str(i+2000),'-dgif')
        else
            figure_name_2000i = ['Figure',num2str(2000+i)];
            print('-dtiff', figure_name_2000i);

            figure(i+3000)

            COEFS                                                    =
cwtext(y_in_window,(1:1:25),'morl','PlotMode','glbabs');
            figure_name_3000i = ['Figure',num2str(3000+i)];
            print('-dtiff', figure_name_3000i);
        end

    end
end

save powerlawPAOSwt_m1_temp

%================

%WAVE_BASES  1D Wavelet functions Morlet, Paul, or DOG
%
%  [DAUGHTER,FOURIER_FACTOR,COI,DOFMIN] = ...
%       wave_bases(MOTHER,K,SCALE,PARAM);
```

```
%
%    Computes the wavelet function as a function of Fourier
frequency,
%   used for the wavelet transform in Fourier space.
%   (This program is called automatically by WAVELET)
%
% INPUTS:
%
%    MOTHER = a string, equal to 'MORLET' or 'PAUL' or 'DOG'
%   K = a vector, the Fourier frequencies at which to calculate
the wavelet
%    SCALE = a number, the wavelet scale
%      PARAM = the nondimensional parameter for the wavelet
function
%
% OUTPUTS:
%
%    DAUGHTER = a vector, the wavelet function
%    FOURIER_FACTOR = the ratio of Fourier period to scale
%    COI = a number, the cone-of-influence size at the scale
%     DOFMIN = a number, degrees of freedom for each point in
the wavelet power
%             (either 2 for Morlet and Paul, or 1 for the DOG)
%
%--------------------------------------------------------------
---------------
%   Copyright (C) 1995-1998, Christopher Torrence and Gilbert
P. Compo
%   University of Colorado, Program in Atmospheric and Oceanic
Sciences.
%   This software may be used, copied, or redistributed as long
as it is not
%   sold and this copyright notice is reproduced on each copy
made.  This
%    routine is provided as is without any express or implied
warranties
%   whatsoever.
%--------------------------------------------------------------
---------------
function [daughter,fourier_factor,coi,dofmin] = ...
     wave_bases(mother,k,scale,param);

mother = upper(mother);
n = length(k);

if (strcmp(mother,'MORLET'))  %----------------------------
```

```
-----  Morlet
      if (param == -1), param = 6.;, end
      k0 = param;
      expnt = -(scale.*k - k0).^2/2.*(k > 0.);
      norm = sqrt(scale*k(2))*(pi^(-0.25))*sqrt(n);    % total
energy=N   [Eqn(7)]
      daughter = norm*exp(expnt);
      daughter = daughter.*(k > 0.);       % Heaviside step
function
      fourier_factor = (4*pi)/(k0 + sqrt(2 + k0^2)); % Scale-
->Fourier [Sec.3h]
      coi = fourier_factor/sqrt(2);                  % Cone-of-
influence [Sec.3g]
      dofmin = 2;                                     % Degrees
of freedom
elseif (strcmp(mother,'PAUL'))  %---------------------------
----  Paul
      if (param == -1), param = 4.;, end
      m = param;
      expnt = -(scale.*k).*(k > 0.);
      norm     =     sqrt(scale*k(2))*(2^m/sqrt(m*prod(2:(2*m-
1))))*sqrt(n);
      daughter = norm*((scale.*k).^m).*exp(expnt);
      daughter = daughter.*(k > 0.);       % Heaviside step
function
      fourier_factor = 4*pi/(2*m+1);
      coi = fourier_factor*sqrt(2);
      dofmin = 2;
elseif (strcmp(mother,'DOG'))  %---------------------------
---  DOG
      if (param == -1), param = 2.;, end
      m = param;
      expnt = -(scale.*k).^2 ./ 2.0;
      norm = sqrt(scale*k(2)/gamma(m+0.5))*sqrt(n);
      daughter = -norm*(i^m)*((scale.*k).^m).*exp(expnt);
      fourier_factor = 2*pi*sqrt(2./(2*m+1));
      coi = fourier_factor/sqrt(2);
      dofmin = 1;
else
      error('Mother must be one of MORLET,PAUL,DOG')
end

return

%==================
```

```
%WAVELET  1D Wavelet transform with optional singificance
testing
%
%                          [WAVE,PERIOD,SCALE,COI]          =
wavelet2(Y,DT,PAD,DJ,S0,J1,mother,sp_ty,param)
%
%
%  Custom version of wavelet.m that permits either 'log' or
'linear' scales
%  spacing.
%
%  In case of "inear" scales spacing, it is proposed to use
DJ=1,
%  S0=irrelevant %  (e.g. = 1) and J1 = number of scales
%
%
%   Computes the wavelet transform of the vector Y (length N),
%   with sampling rate DT.
%
%   By default, the Morlet wavelet (k0=6) is used.
%   The wavelet basis is normalized to have total energy=1 at
all scales.
%
%
% INPUTS:
%
%    Y = the time series of length N.
%   DT = amount of time between each Y value, i.e. the sampling
time.
%
% OUTPUTS:
%
%     WAVE is the WAVELET transform of Y. This is a complex
array
%     of dimensions (N,J1+1). FLOAT(WAVE) gives the WAVELET
amplitude,
%    ATAN(IMAGINARY(WAVE),FLOAT(WAVE) gives the WAVELET phase.
%    The WAVELET power spectrum is ABS(WAVE)^2.
%    Its units are sigma^2 (the time series variance).
%
%
% OPTIONAL INPUTS:
%
% *** Note *** setting any of the following to -1 will cause
the default
%              value to be used.
```

```
%
%     PAD = if set to 1 (default is 0), pad time series with
enough zeroes to get
%           N up to the next higher power of 2. This prevents
wraparound
%           from the end of the time series to the beginning,
and also
%         speeds up the FFT's used to do the wavelet transform.
%           This will not eliminate all edge effects (see COI
below).
%
%    DJ = the spacing between discrete scales. Default is 0.25.
%           A smaller # will give better scale resolution, but
be slower to plot.
%
%    S0 = the smallest scale of the wavelet.  Default is 2*DT.
%
%     J1 = the # of scales minus one. Scales range from S0 up
to S0*2^(J1*DJ),
%           to give a total of (J1+1) scales. Default is J1 =
(LOG2(N DT/S0))/DJ.
%
%
% OPTIONAL OUTPUTS:
%
%     PERIOD = the vector of "Fourier" periods (in time units)
that corresponds
%            to the SCALEs.
%
%    SCALE = the vector of scale indices, given by S0*2^(j*DJ),
j=0...J1
%             where J1+1 is the total # of scales.
%
%      COI = if specified, then return the Cone-of-Influence,
which is a vector
%        of N points that contains the maximum period of useful
information
%         at that particular time.
%        Periods greater than this are subject to edge effects.
%         This can be used to plot COI lines on a contour plot
by doing:
%             IDL>  CONTOUR,wavelet,time,period
%             IDL>  PLOTS,time,coi,NOCLIP=0
%
%
%-------------------------------------------------------------
```

```
---------------
%   Copyright (C) 1995-1998, Christopher Torrence and Gilbert
P. Compo
%   University of Colorado, Program in Atmospheric and Oceanic
Sciences.
%   This software may be used, copied, or redistributed as long
as it is not
%   sold and this copyright notice is reproduced on each copy
made.  This
%   routine is provided as is without any express or implied
warranties
%   whatsoever.
%
% Notice: Please acknowledge the use of this program in any
publications:
%    ``Wavelet software was provided by C. Torrence and G.
Compo,
%                    and    is    available    at    URL:
http://paos.colorado.edu/research/wavelets/''.
%
% Reference: Torrence, C. and G. P. Compo, 1998: A Practical
Guide to
%           Wavelet Analysis. <I>Bull. Amer. Meteor. Soc.</I>,
79, 61-78.
%
% Please send a copy of such publications to either C. Torrence
or G. Compo:
%  Dr. Christopher Torrence               Dr. Gilbert P. Compo
%  Advanced Study Program                   NOAA/CIRES Climate
Diagnostics Center
%  National Center for Atmos. Research    Campus Box 449
%  P.O. Box 3000                         University of Colorado
at Boulder
%  Boulder CO 80307--3000, USA.             Boulder CO 80309-
0449, USA.
%    E-mail: torrence@ucar.edu                          E-mail:
gpc@cdc.noaa.gov
%--------------------------------------------------------------
---------------
function [wave,period,scale,coi] = ...
     wavelet2(Y,dt,pad,dj,s0,J1,mother,sp_ty,param);

if (nargin < 9), param = -1;, end
if (nargin < 8), sp_ty = -1;, end
if (nargin < 7), mother = -1;, end
if (nargin < 6), J1 = -1;, end
```

```
if (nargin < 5), s0 = -1;, end
if (nargin < 4), dj = -1;, end
if (nargin < 3), pad = 0;, end
if (nargin < 2)
     error('Must input a vector Y and sampling time DT')
end

n1 = length(Y);

if (s0 == -1), s0=2*dt;, end
if (dj == -1), dj = 1./4.;, end
if (J1 == -1), J1=fix((log(n1*dt/s0)/log(2))/dj);, end
if (mother == -1), mother = 'MORLET';, end
if (mother == -1), sp_ty = 'log';, end

%....construct time series to analyze, pad if necessary
x(1:n1) = Y - mean(Y);
if (pad == 1)
     base2 = fix(log(n1)/log(2) + 0.4999);    % power of 2
nearest to N
     x = [x,zeros(1,2^(base2+1)-n1)];
end
n = length(x);

%....construct wavenumber array used in transform [Eqn(5)]
k = [1:fix(n/2)];
k = k.*((2.*pi)/(n*dt));
k = [0., k, -k(fix((n-1)/2):-1:1)];

%....compute FFT of the (padded) time series
f = fft(x);    % [Eqn(3)]

%....construct SCALE array & empty PERIOD & WAVE arrays

  if  strcmp(sp_ty,'log') ==1
      scale = s0*2.^((0:J1)*dj);
  else
      scale = (1:dj:J1+1)*dt;
  end

period = scale;
wave = zeros(J1+1,n);  % define the wavelet array
wave = wave + i*wave;  % make it complex

% loop through all scales and compute transform
for a1 = 1:J1+1
```

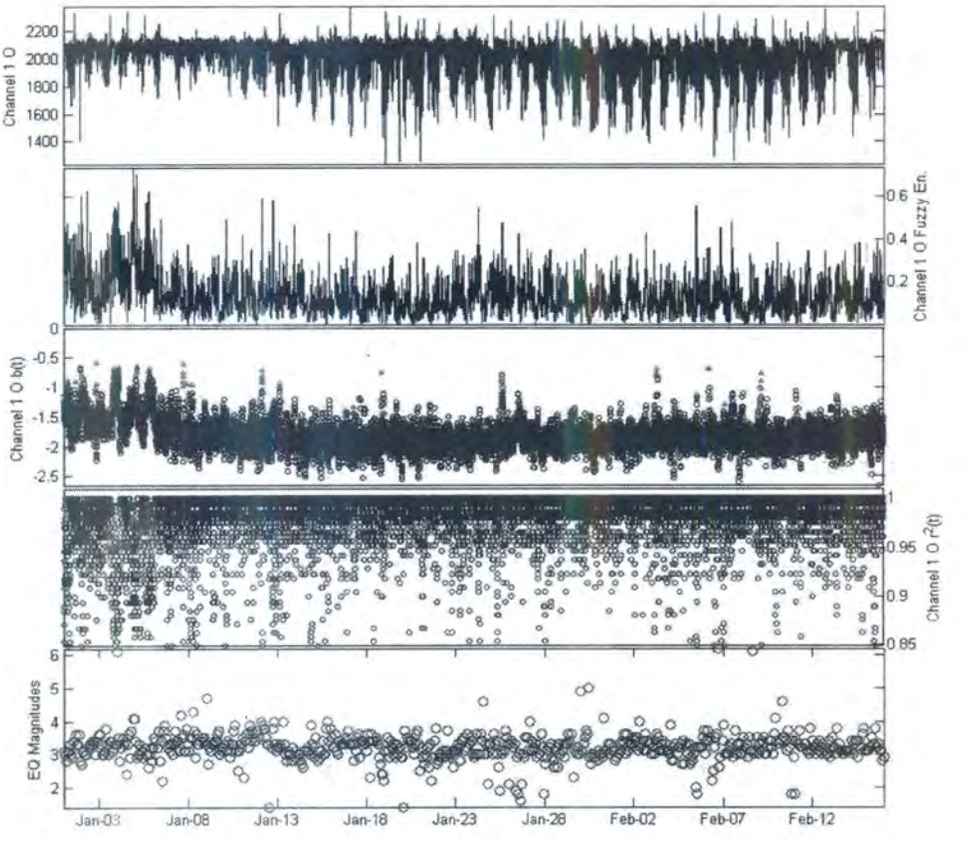
```
    [daughter,fourier_factor,coi,dofmin]=wave_bases(mother,k
,scale(a1),param);
    wave(a1,:)   =   ifft(f.*daughter);       %   wavelet
transform[Eqn(4)]
end

period = fourier_factor*scale;
coi   =   coi*dt*[1E-5,1:((n1+1)/2-1),fliplr((1:(n1/2-1))),1E-
5];  % COI [Sec.3g]
wave = wave(:,1:n1);  % get rid of padding before returning

return
```

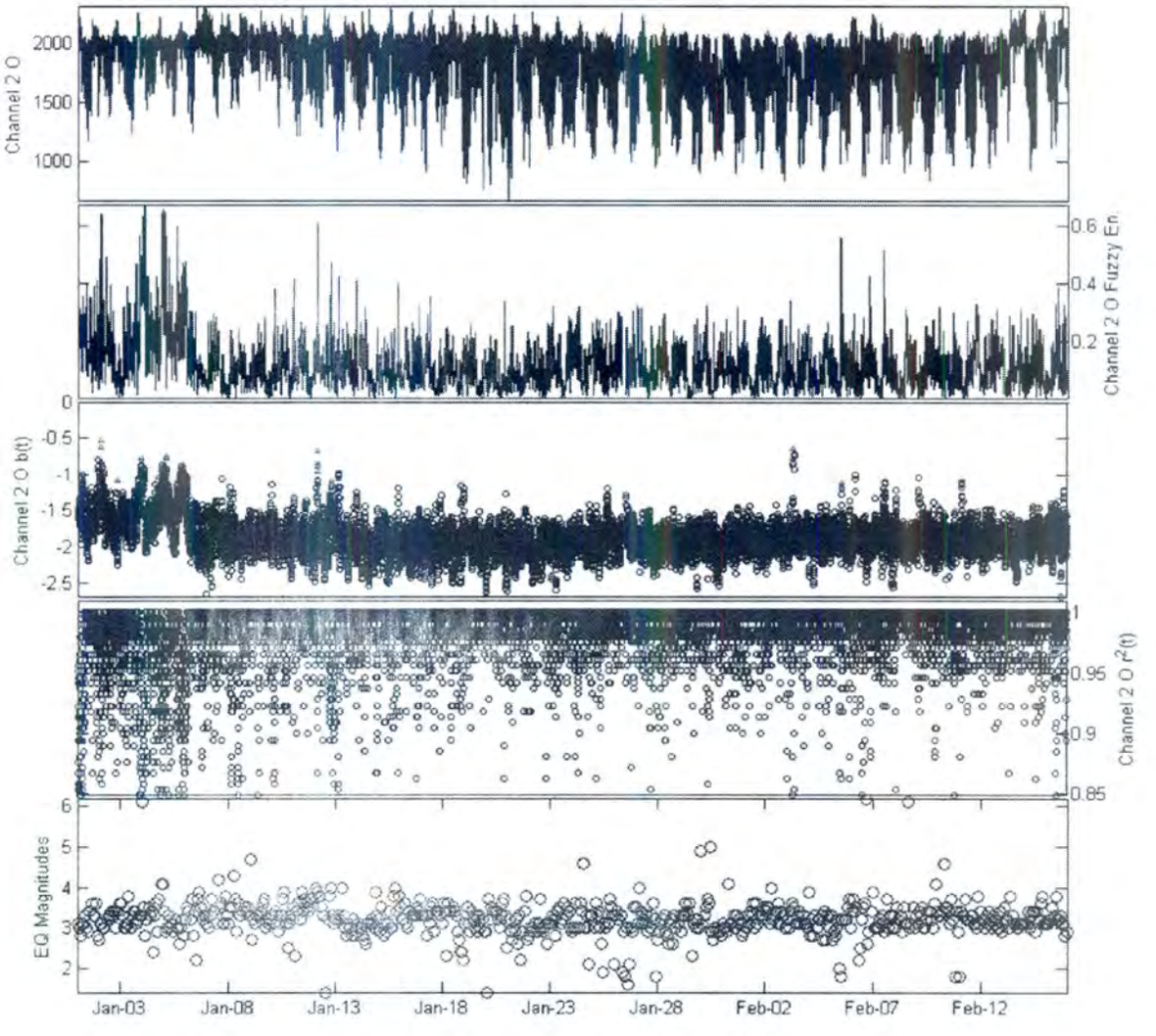
16. Παράρτημα Γ – Διαγράμματα Μετρήσεων

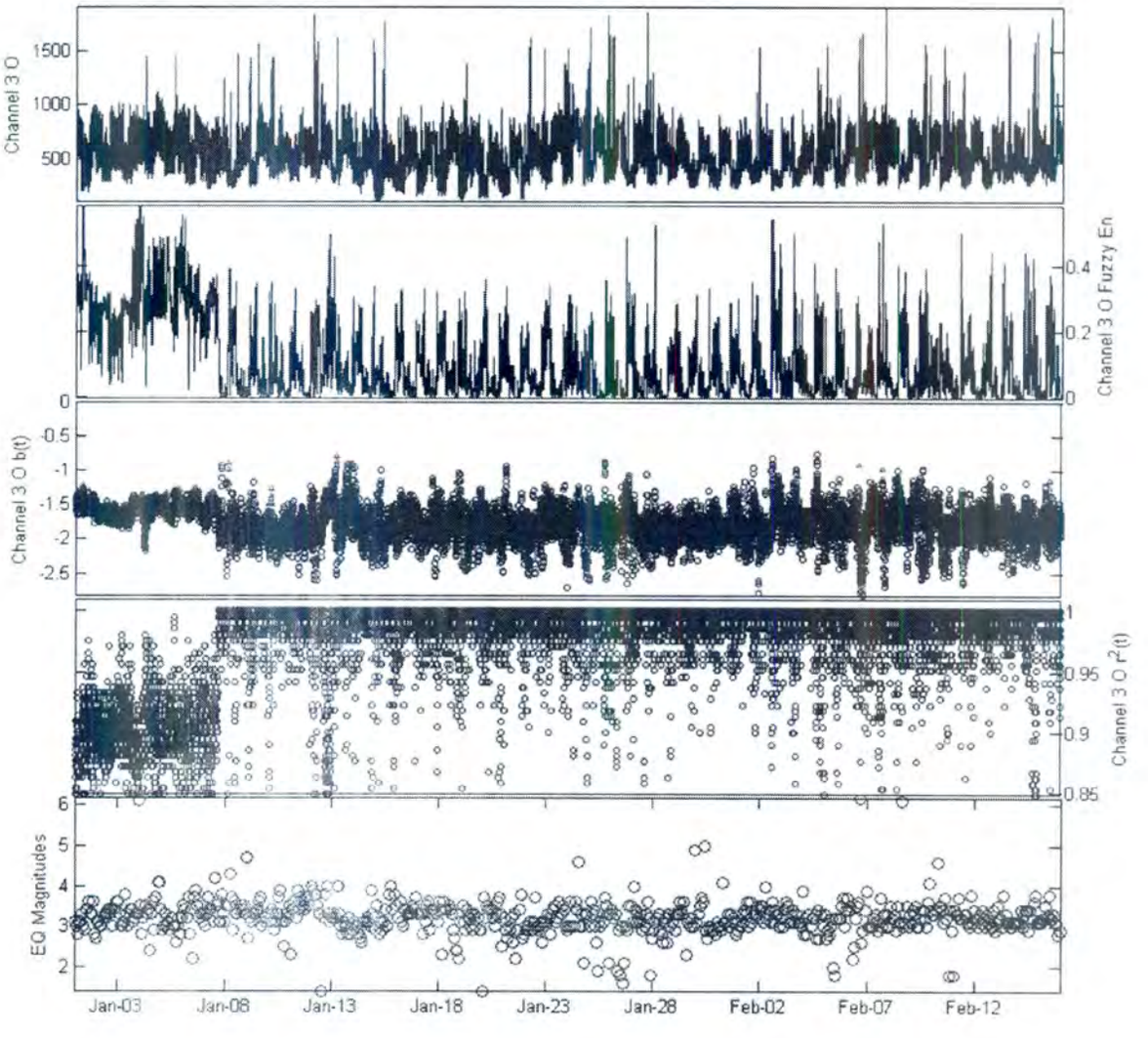
Ακολουθούν οι εκτενής μετρήσεις πειραματικών αποτελεσμάτων των προς ανάλυση μεθόδων της ασαφούς εντροπίας και της πυκνότητας φάσματος ισχύος σε αντιπαραβολή με τα MHz/KHz σήματα μέτρησης γεωηλεκτρισμού και την αντίστοιχη ισχύ των σεισμών. Υπενθυμίζονται οι σταθμοί με το κωδικό μονόγραμμα ανα περιοχή:

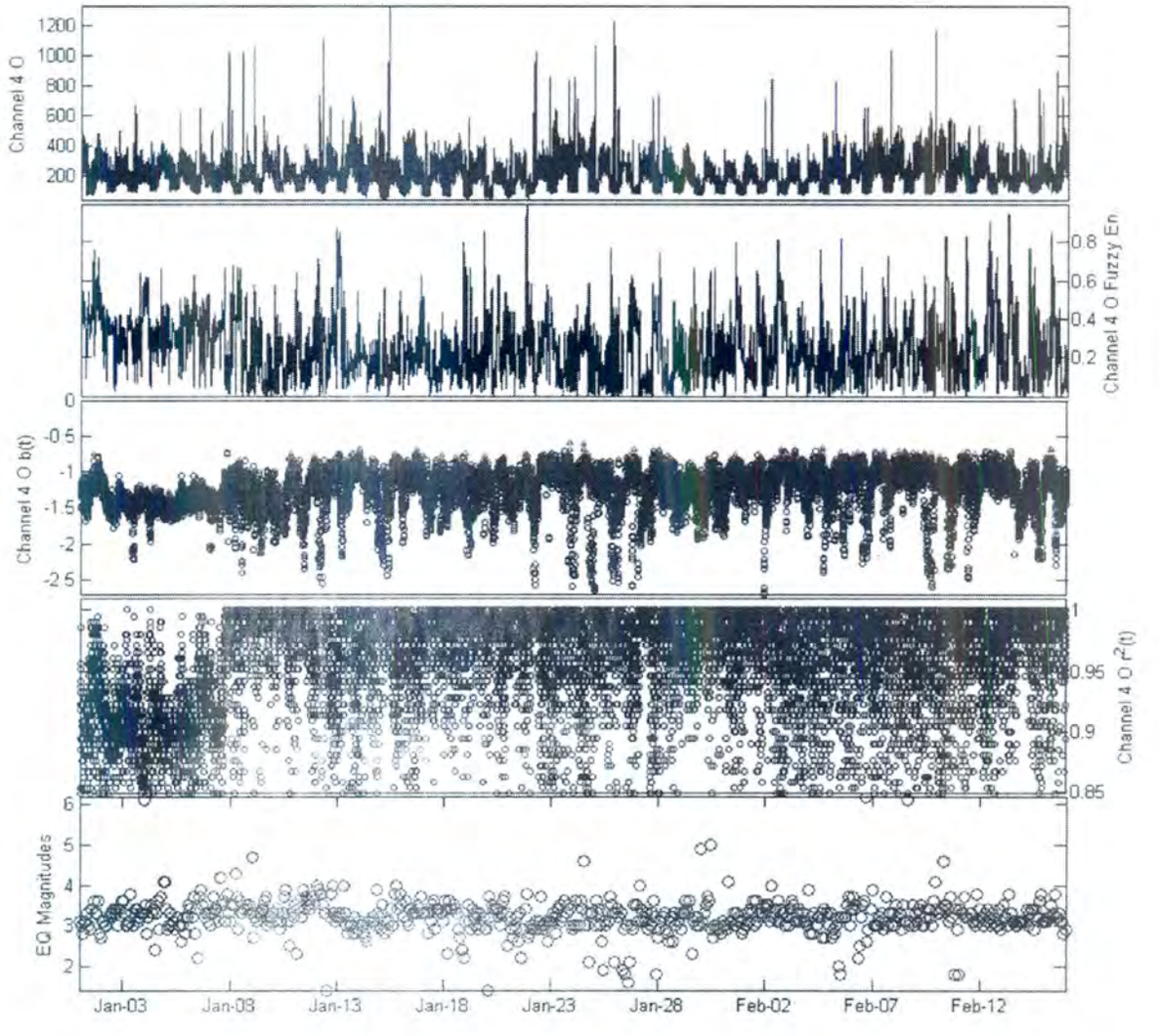
Όνομα Σταθμού	Κωδ. Σταθμού	Όνομα Σταθμού	Κωδ. Σταθμού
Καλαμάτα	O	Νεάπολις	E
Κεφαλονιά	F	Κέρκυρα	P
Κομοτηνή	T	Κοζάνη	K
Ιωάννινα	J	Αταλάντη	H
Μυτιλήνη	M	Ρόδος	A
Ζάκυνθος	Z	Βάμος	V

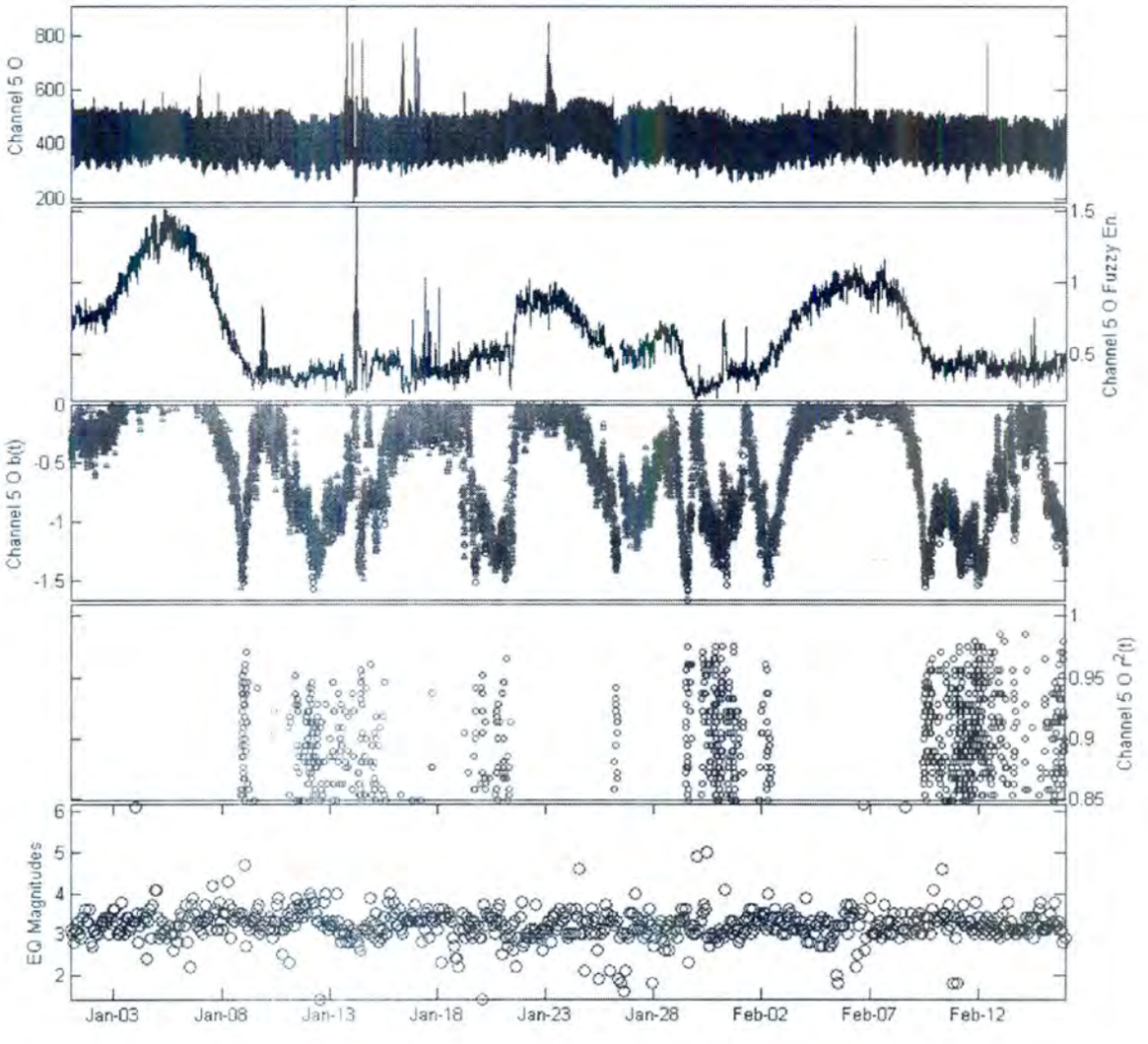
Κάθε γράφημα στο δεξιό ή στο αριστερό μέρος του φέρει την σχετική ονομασία για την ιδιότητα του. Επίσης στον άξονα των x φαίνεται και η κοινή περίοδος μέτρησης.

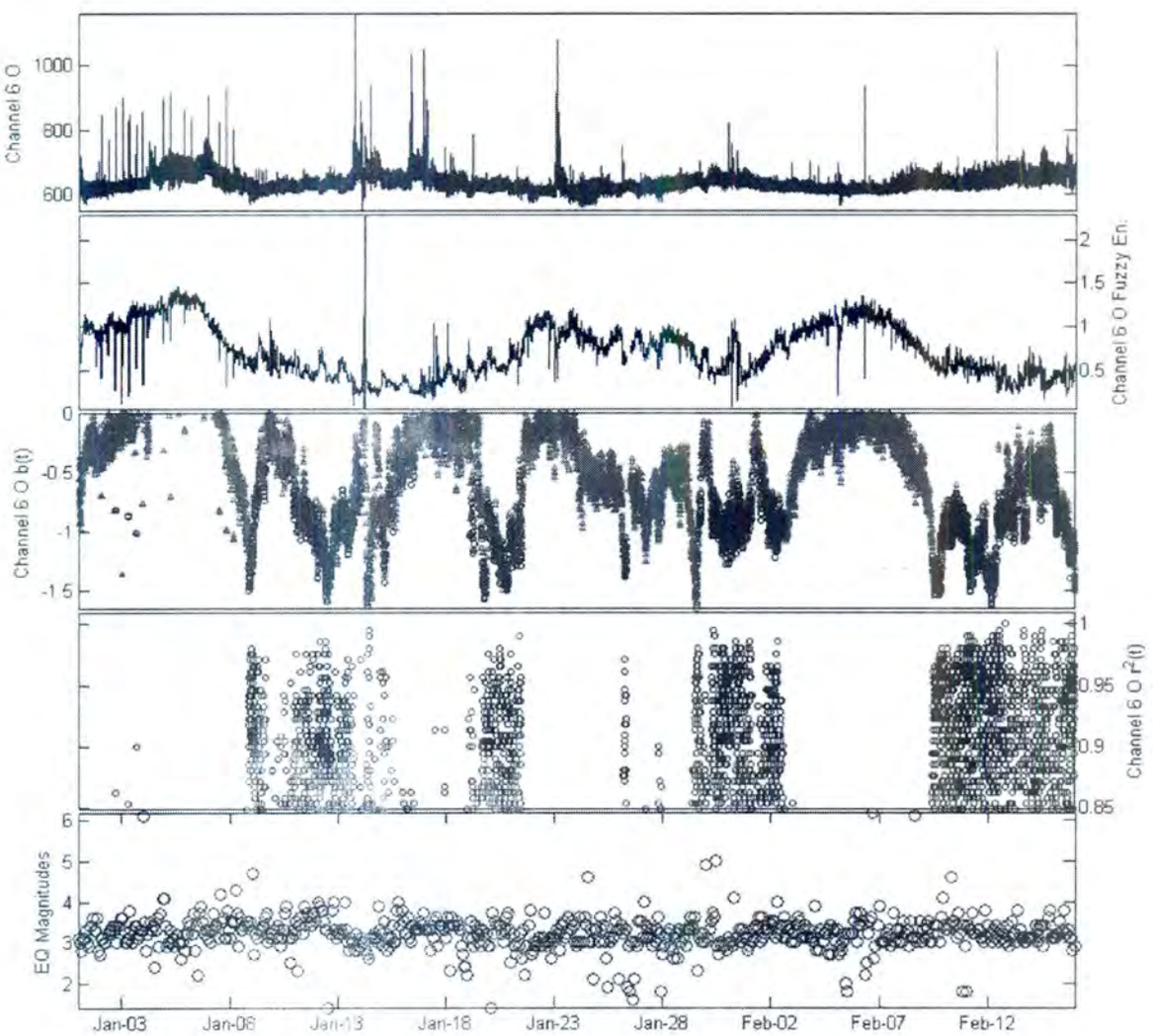


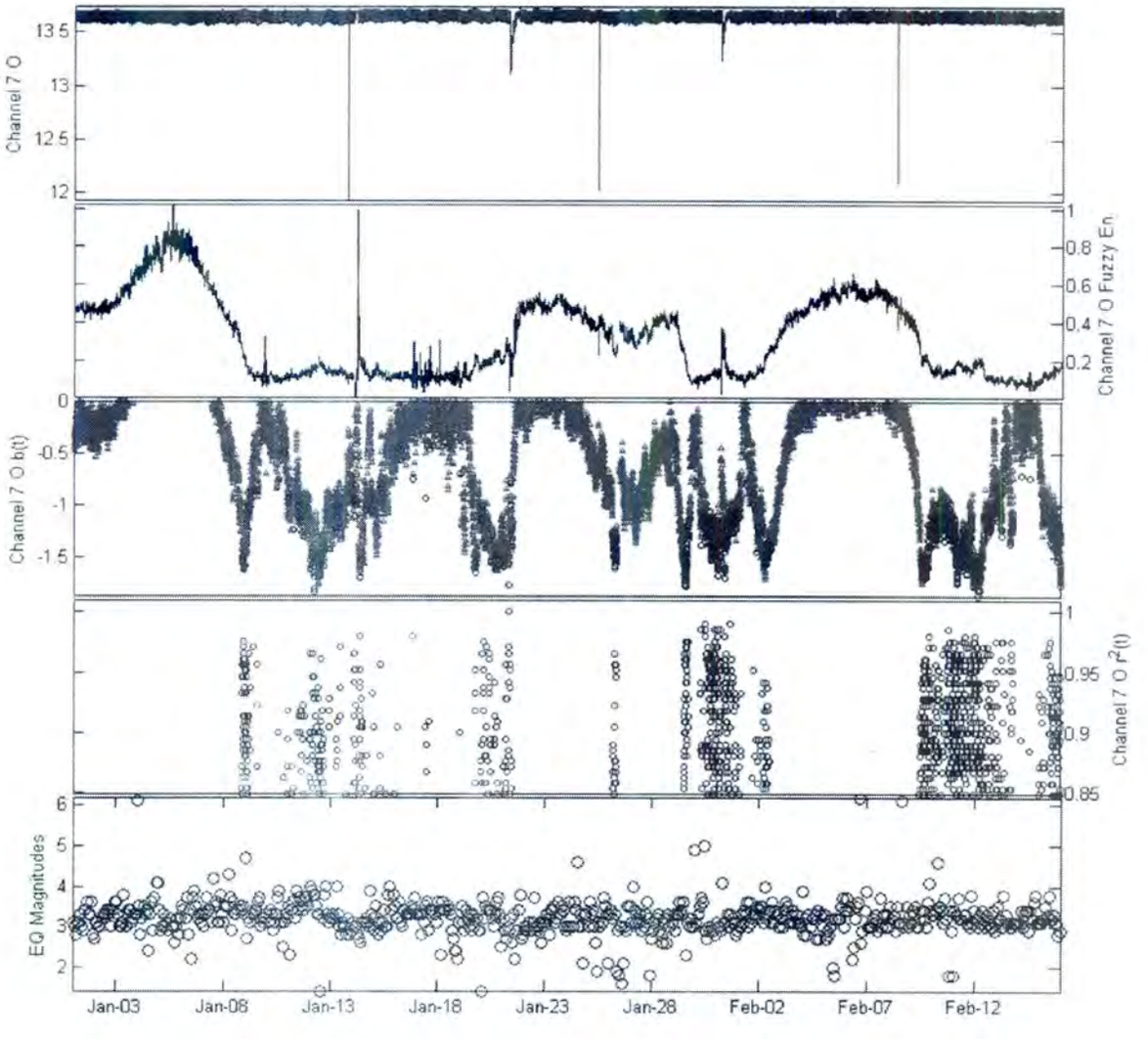

Channel 2 O
2000
1500
1000
0.6
0.4
0.2
0
Channel 2 O Fuzzy En
Channel 2 O b(t)
-0.5
-1
-1.5
-2
-2.5
1
0.95
0.9
0.85
Channel 2 O r²(t)
6
5
4
3
2
EQ Magnitudes
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28
Feb-02
Feb-07
Feb-12

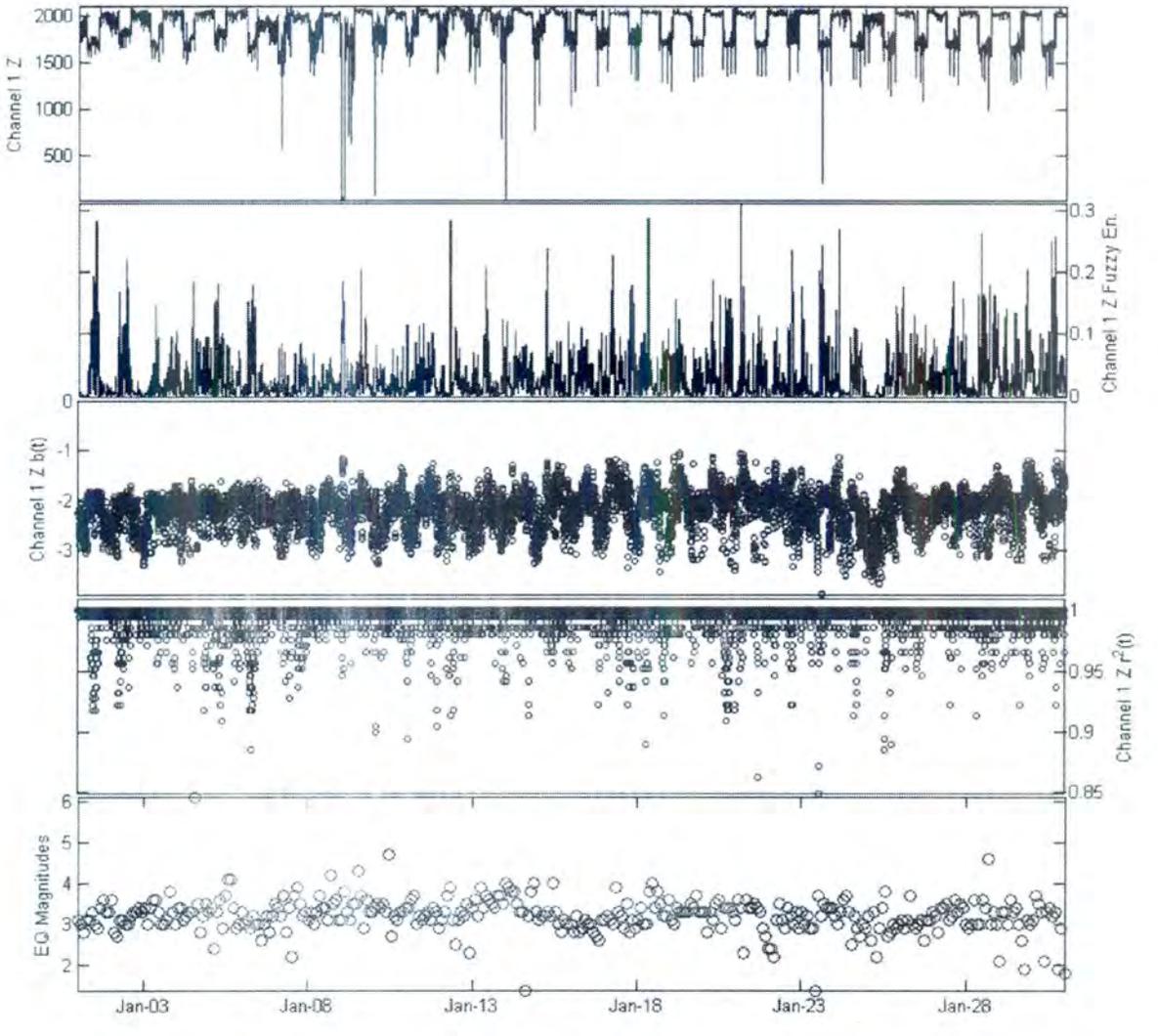

Channel 3 O
1500
1000
500
Channel 3 O Fuzzy En
0.4
0.2
0
0
Channel 3 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 3 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28
Feb-02
Feb-07
Feb-12

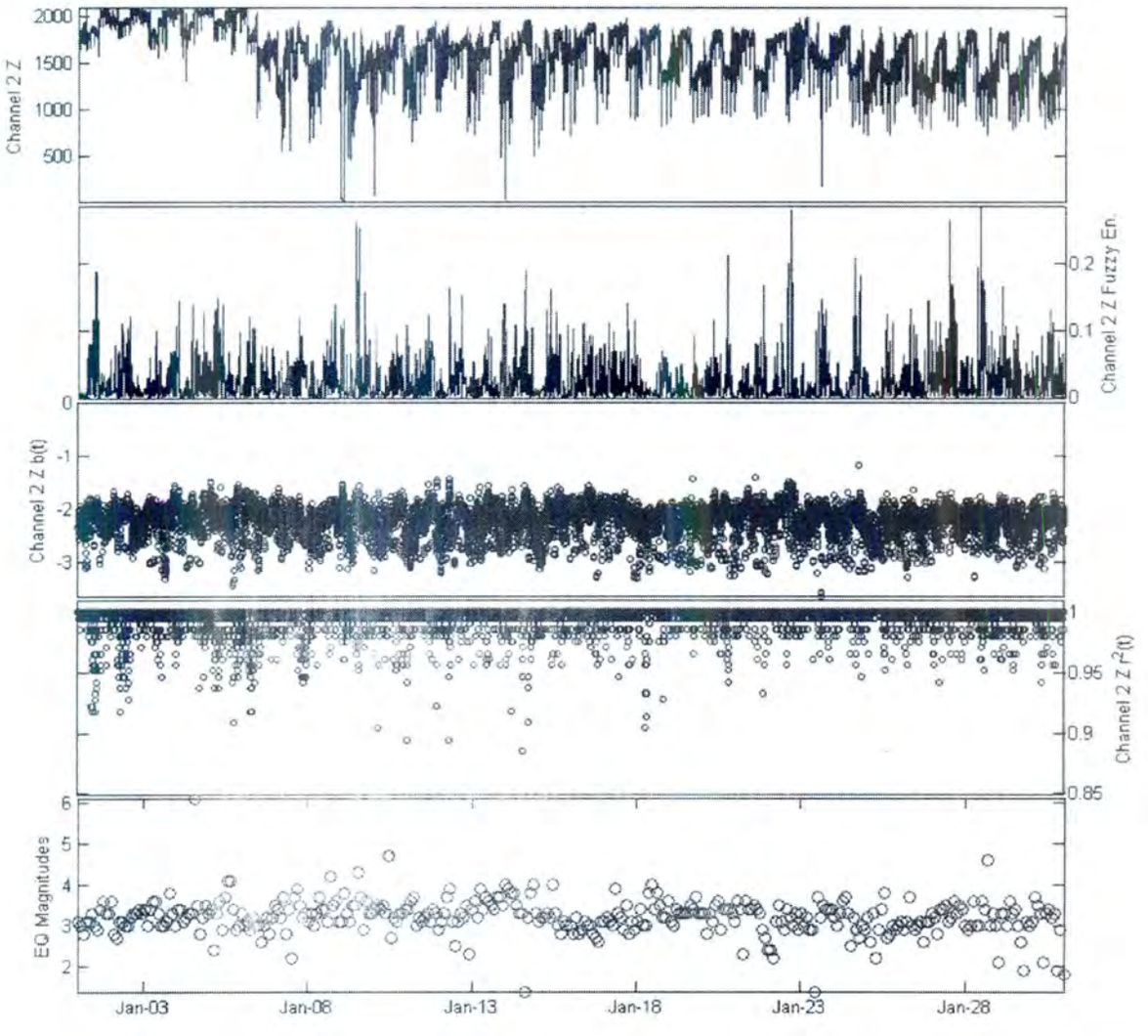

Channel 4 O
1200
1000
800
600
400
200
Channel 4 O Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 4 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28
Feb-02
Feb-07
Feb-12

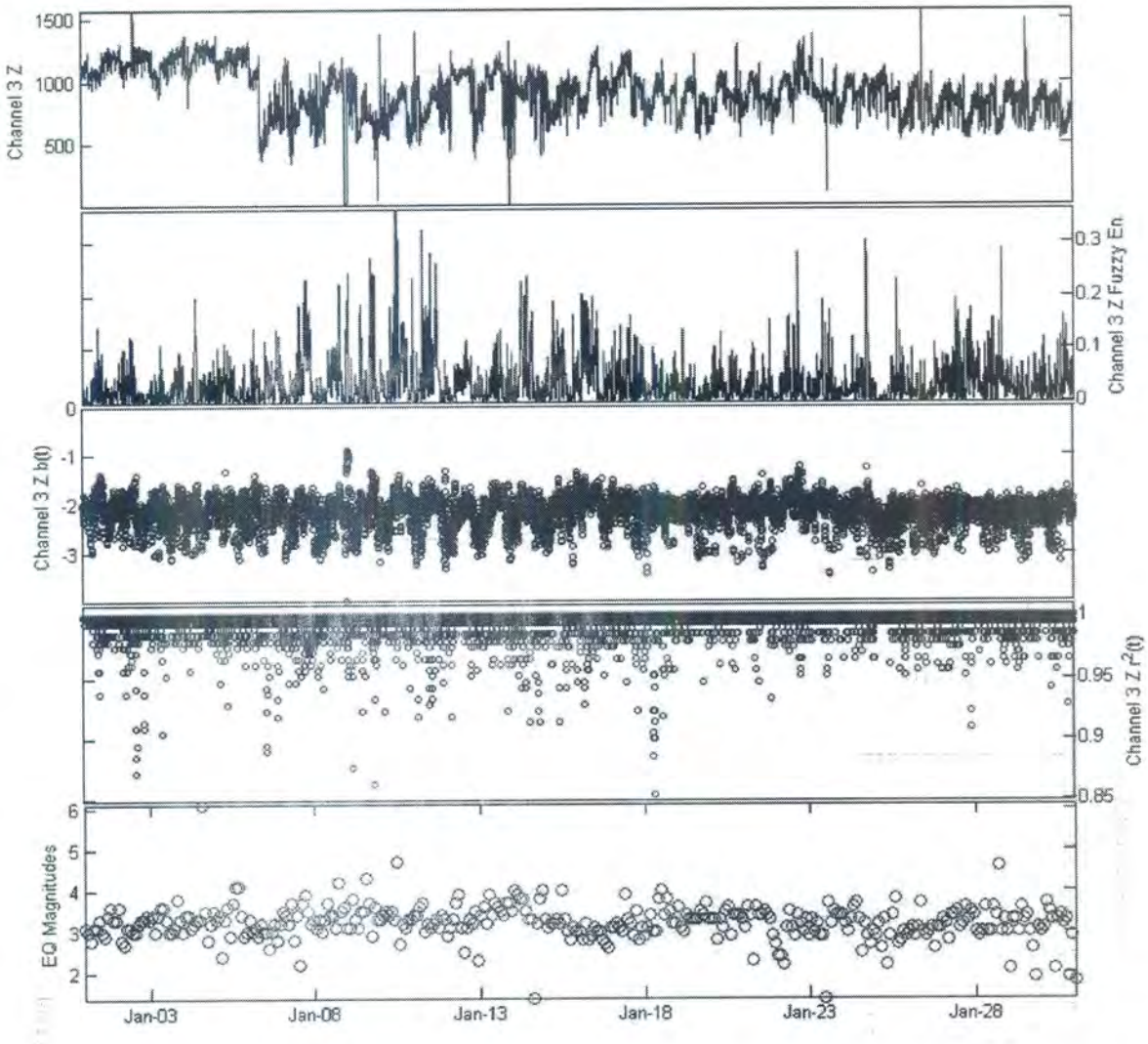

Channel 5 O
800
600
400
200
1.5
1
0.5
Channel 5 O Fuzzy En.
0
-0.5
-1
-1.5
Channel 5 O b(t)
1
0.95
0.9
0.85
Channel 5 O r²(t)
6
5
4
3
2
EQ Magnitudes
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28
Feb-02
Feb-07
Feb-12

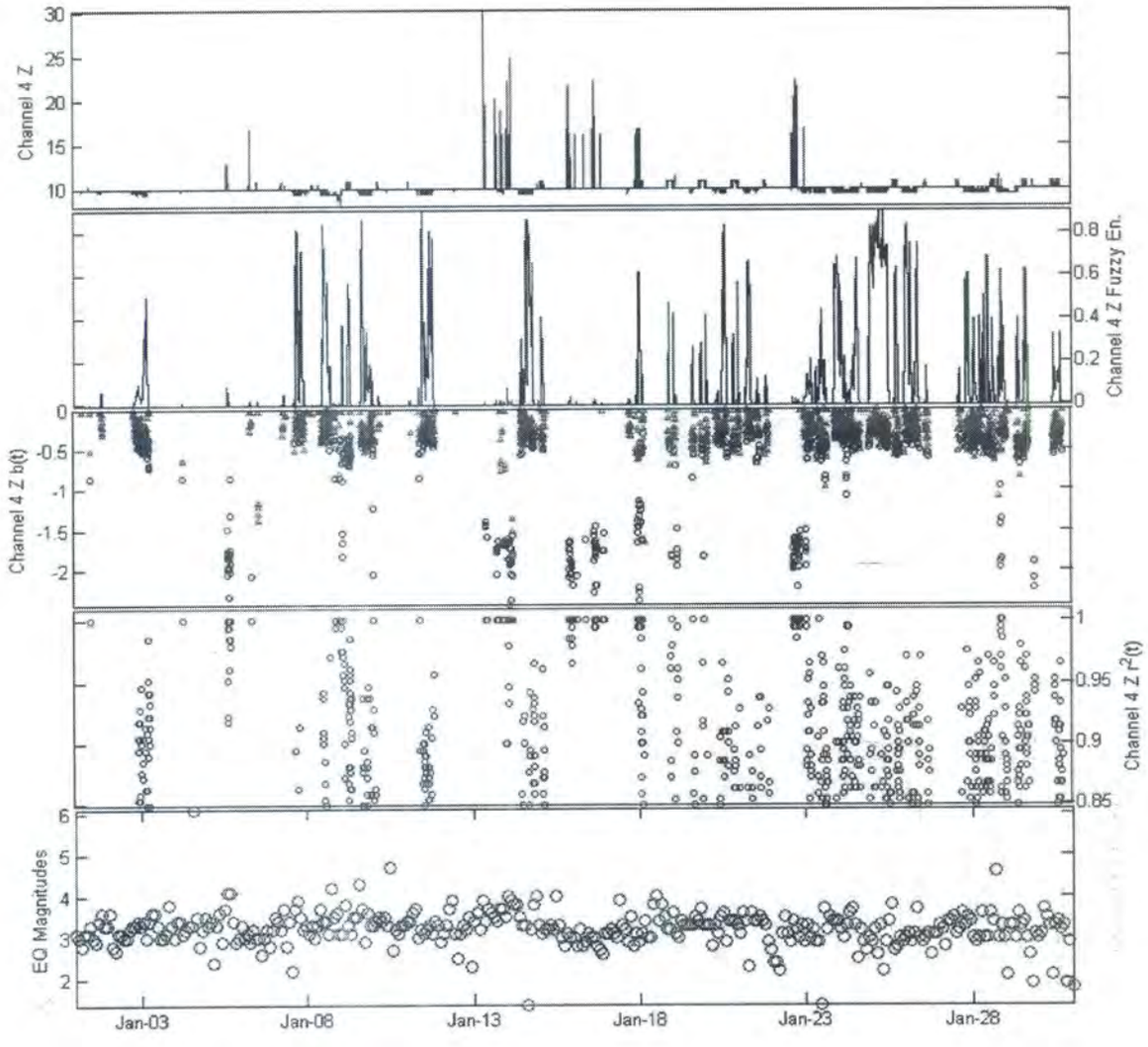

Channel 6 O
1000
800
600
Channel 6 O Fuzzy En.
2
1.5
1
0.5
Channel 6 O b(t)
0
-0.5
-1
-1.5
Channel 6 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28
Feb-02
Feb-07
Feb-12

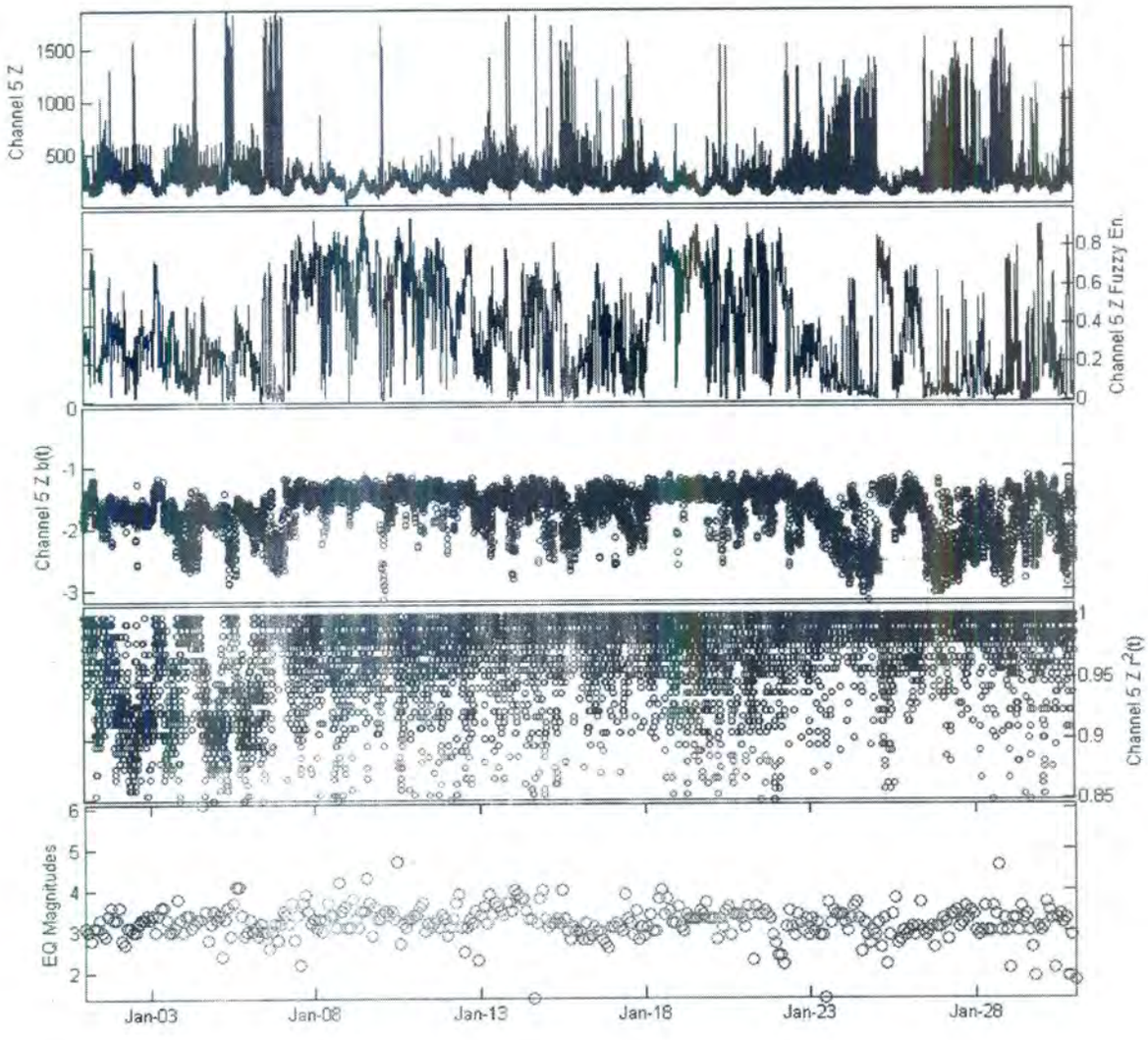

Channel 7 O
13.5
13
12.5
12
Channel 7 O Fuzzy En
1
0.8
0.6
0.4
0.2
0
Channel 7 O b(t)
-0.5
-1
-1.5
Channel 7 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28
Feb-02
Feb-07
Feb-12

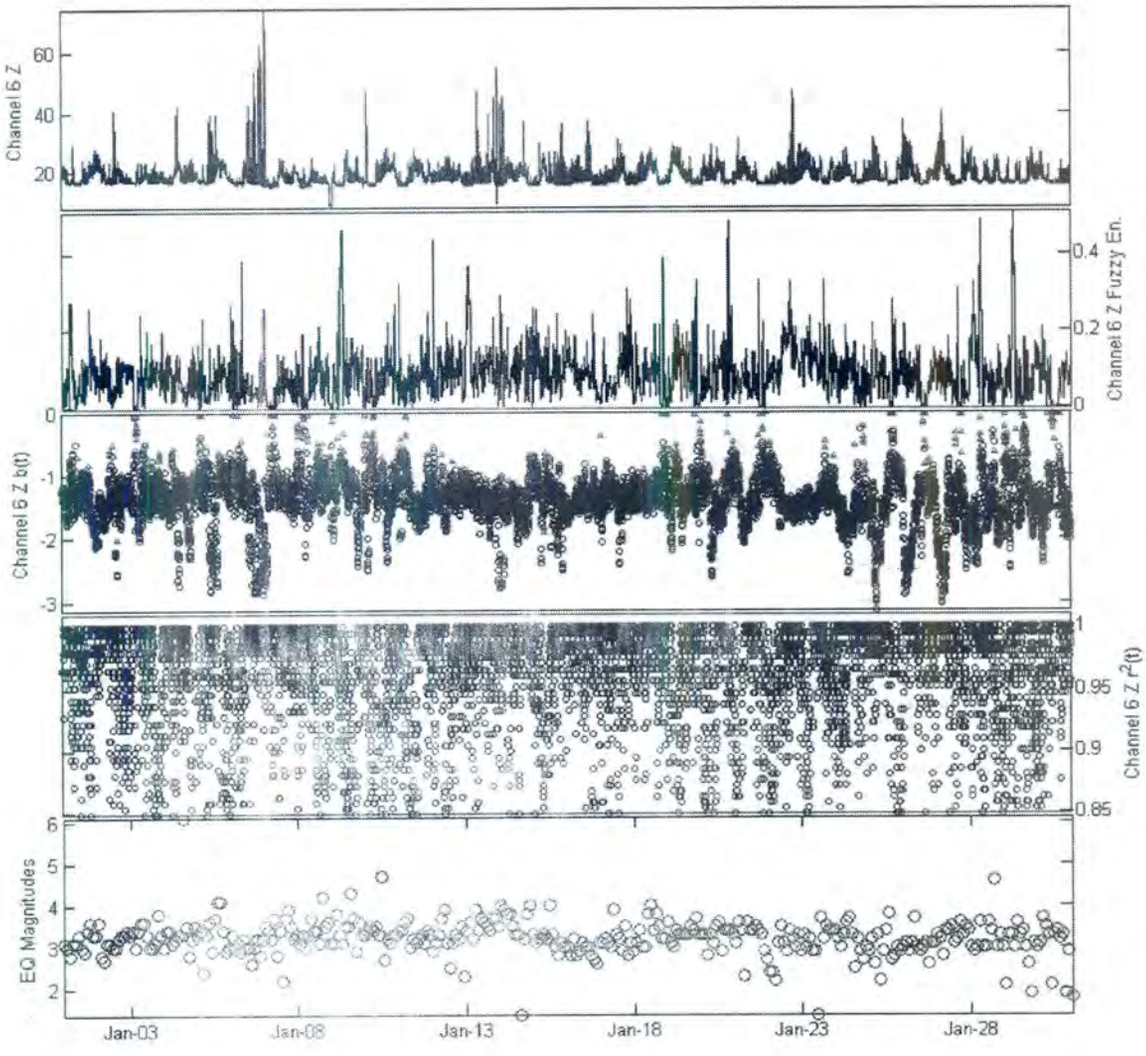

Channel 1 Z
2000
1500
1000
500
Channel 1 Z Fuzzy En.
0.3
0.2
0.1
0
Channel 1 Z b(t)
-1
-2
-3
Channel 1 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

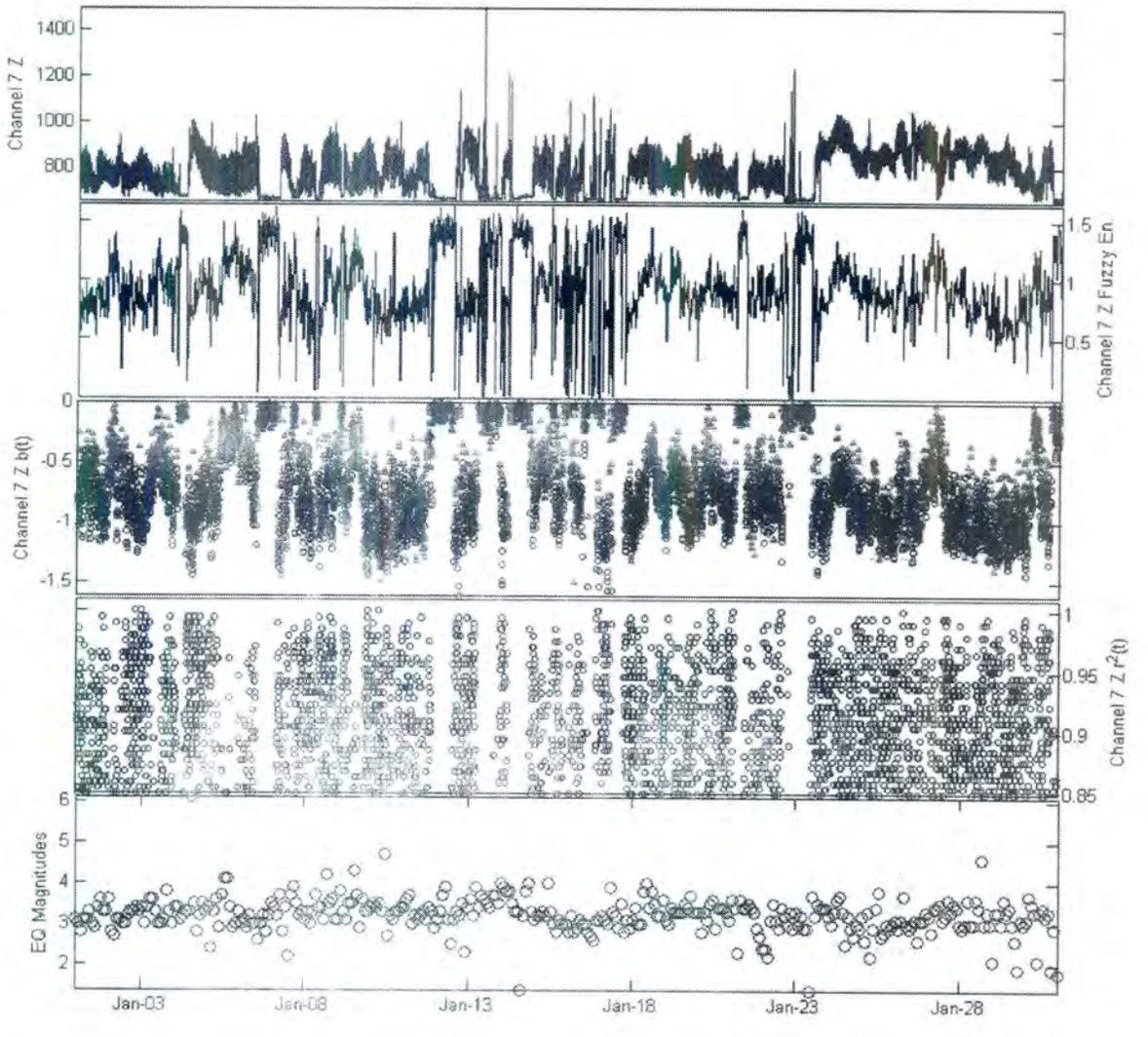

Channel 2 Z
2000
1500
1000
500
Channel 2 Z Fuzzy En.
0.2
0.1
0
Channel 2 Z b(t)
0
-1
-2
-3
Channel 2 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

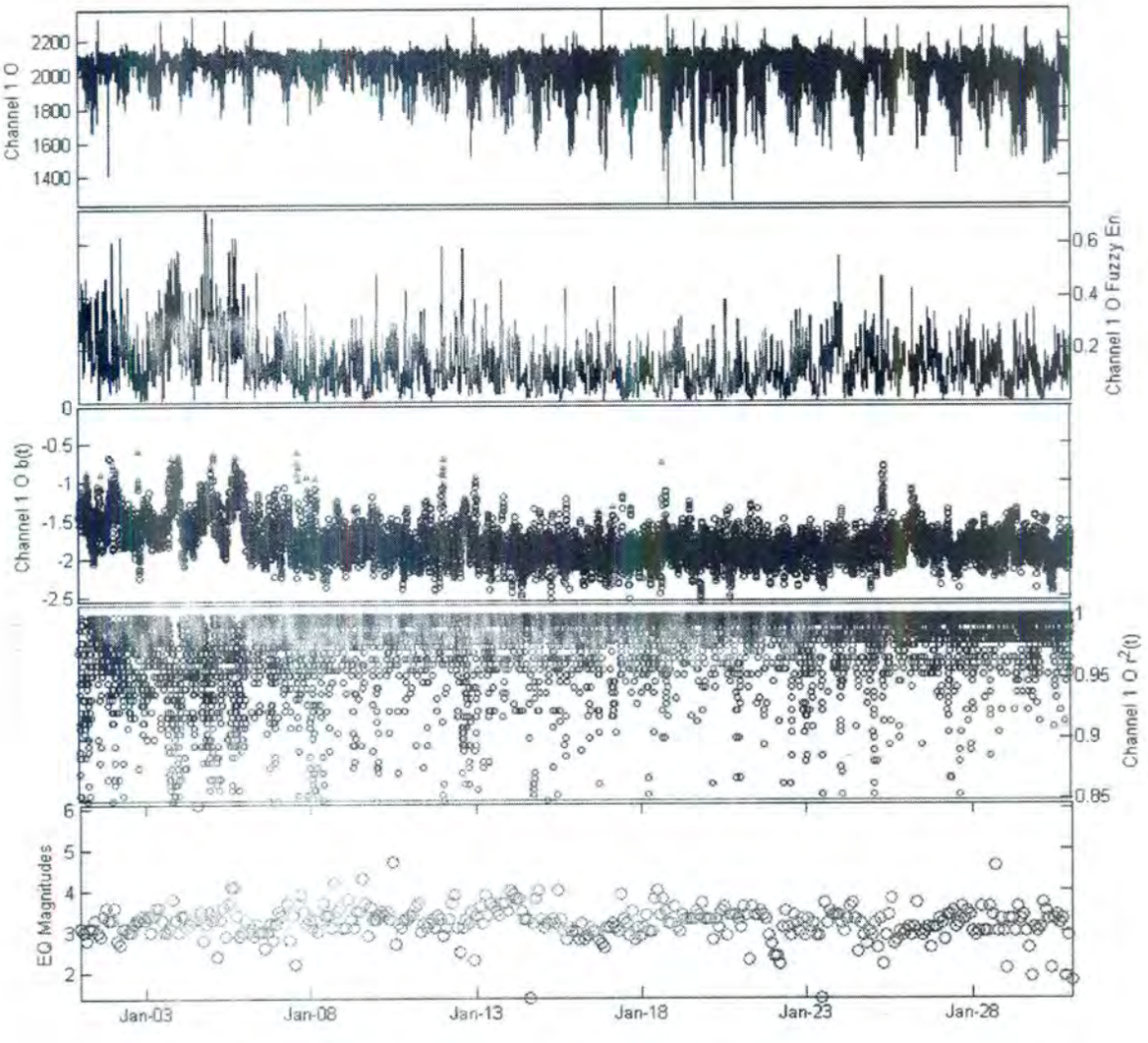

Channel 3 Z
1500
1000
500
Channel 3 Z Fuzzy En
0.3
0.2
0.1
0
Channel 3 Z b(t)
-1
-2
-3
Channel 3 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

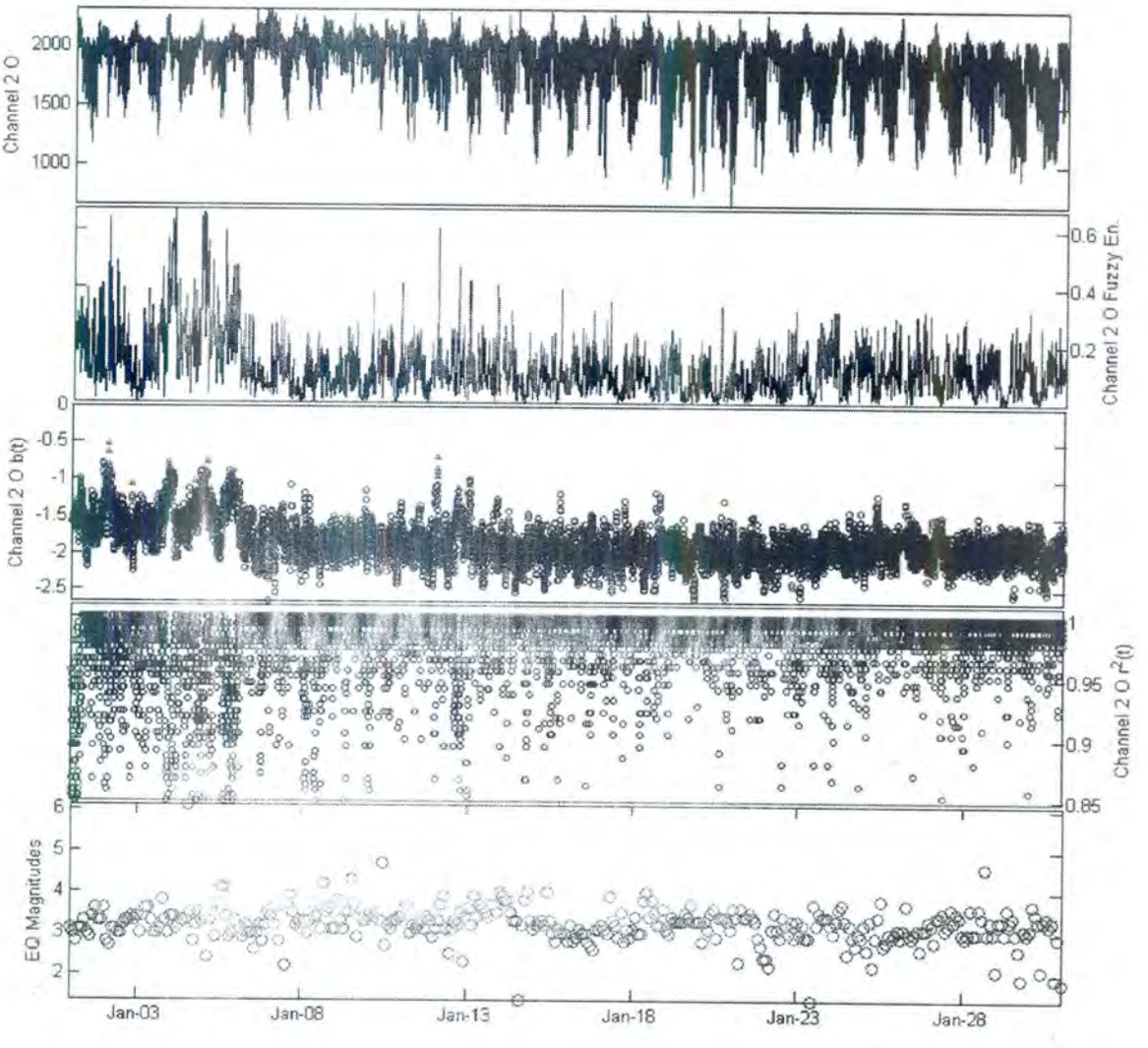

Channel 4 Z
Channel 4 Z Fuzzy En.
Channel 4 Z b(t)
Channel 4 Z r²(t)
EQ Magnitudes
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

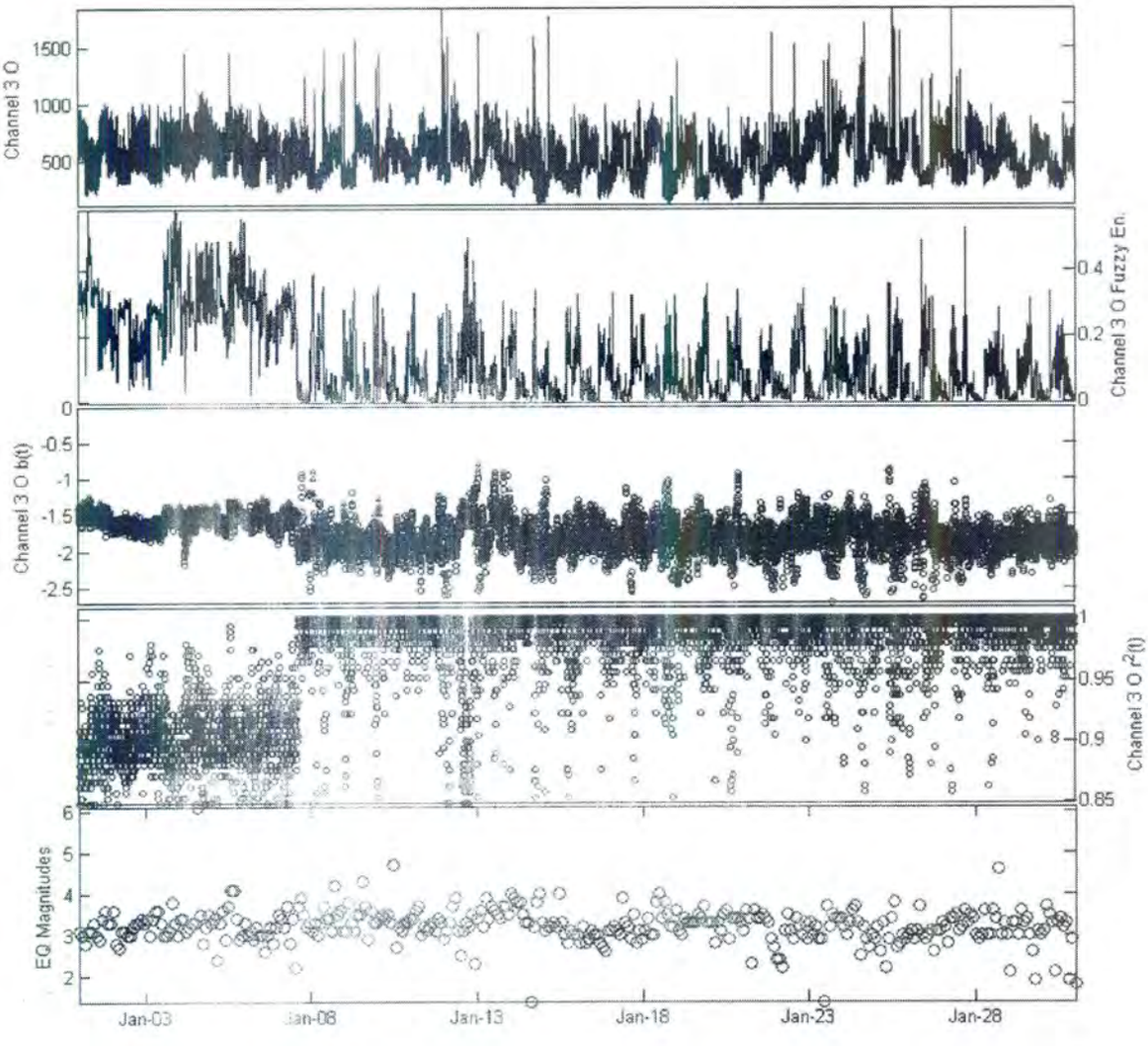

Channel 5 Z
1500
1000
500
Channel 5 Z Fuzzy En
0.8
0.6
0.4
0.2
0
Channel 5 Z b(t)
0
-1
-2
-3
Channel 5 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

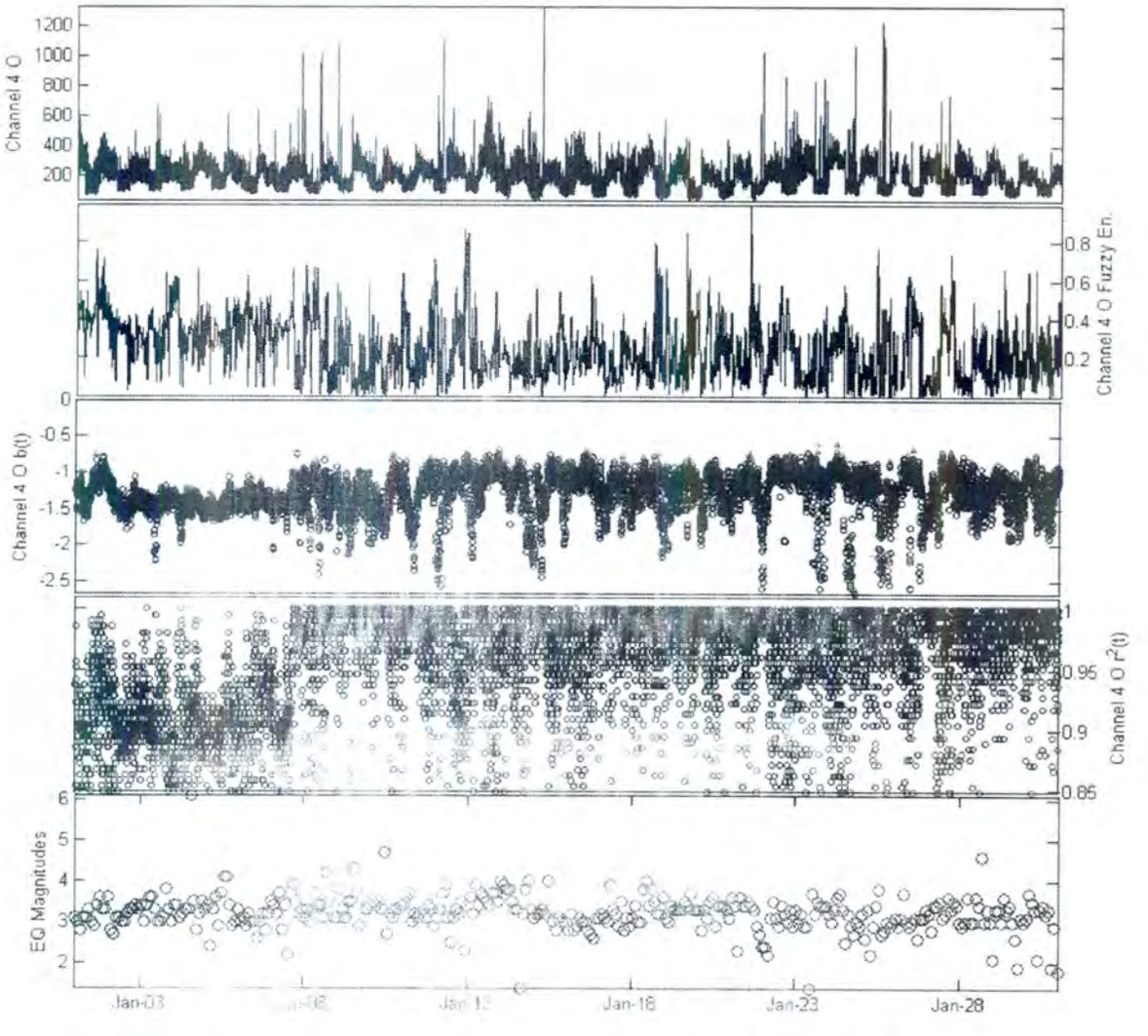

Channel 6 Z
60
40
20
Channel 6 Z Fuzzy En.
0.4
0.2
0
Channel 6 Z b(t)
0
-1
-2
-3
Channel 6 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

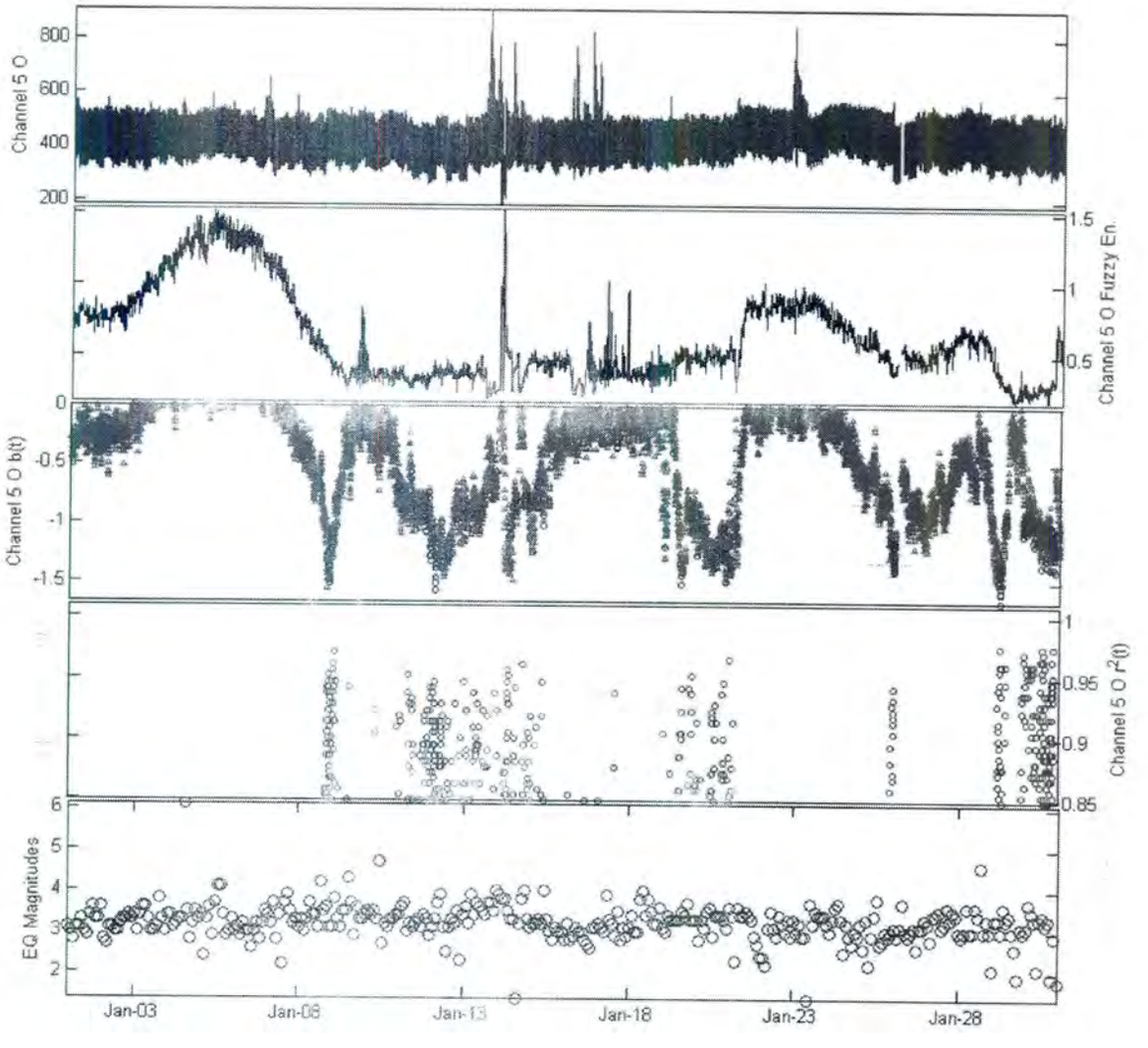

Channel 7 Z
1400
1200
1000
800
Channel 7 Z Fuzzy En.
1.5
1
0.5
0
Channel 7 Z b(t)
0
-0.5
-1
-1.5
Channel 7 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

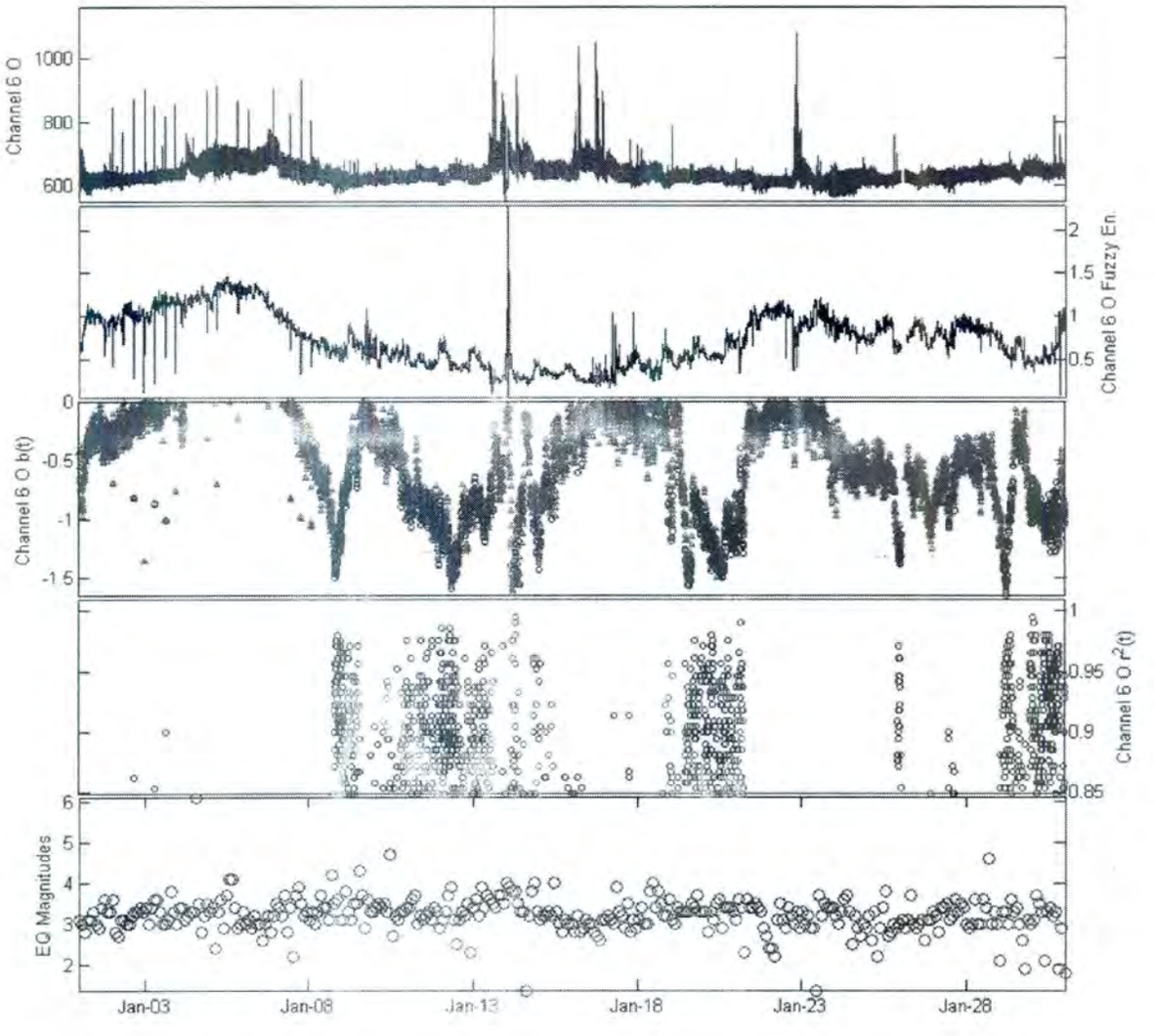

Channel 1 O
2200
2000
1800
1600
1400
Channel 1 O Fuzzy En
0.6
0.4
0.2
0
Channel 1 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 O r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

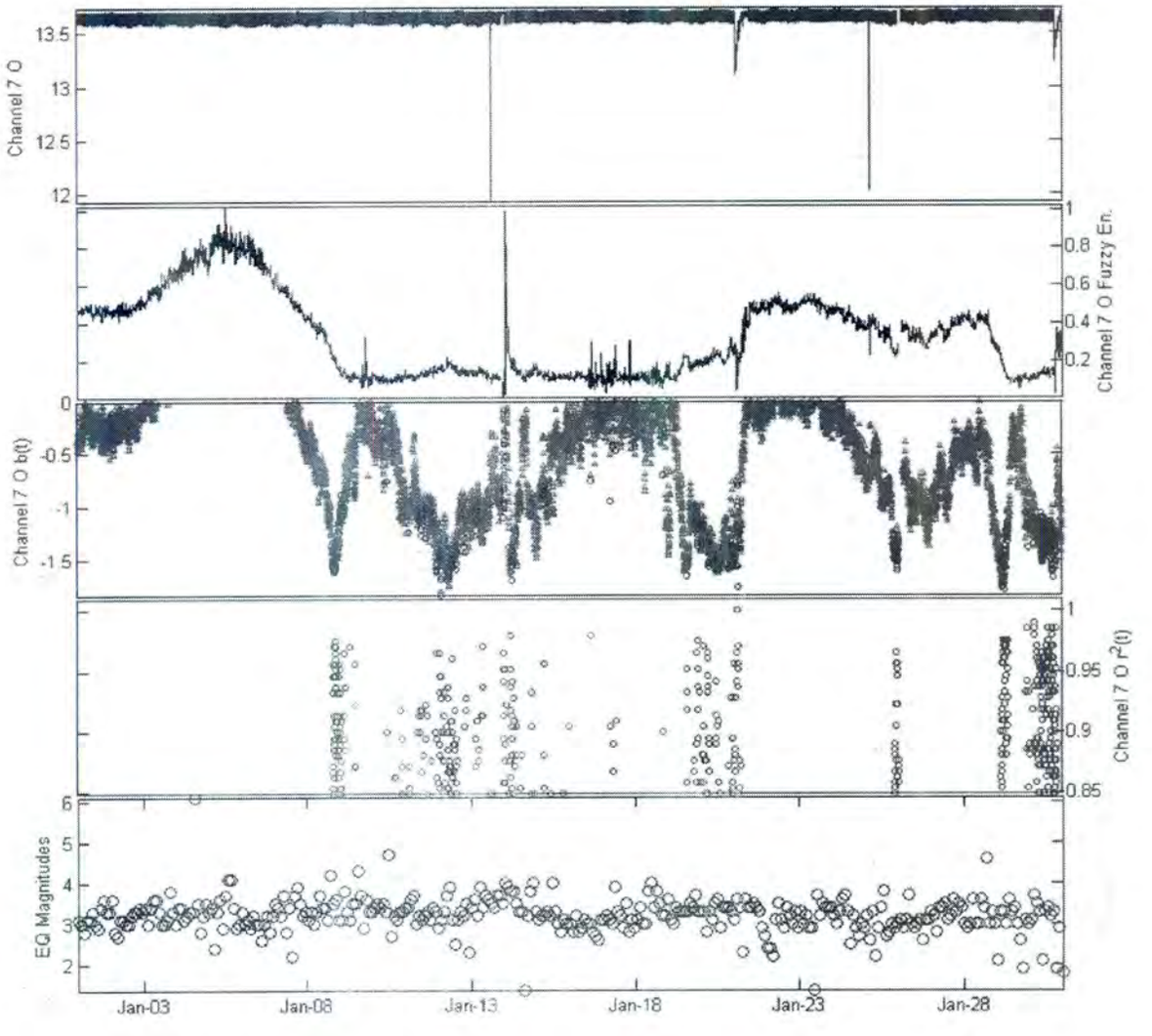

Channel 2 O
2000
1500
1000
0.6
0.4
0.2
0
Channel 2 O Fuzzy En.
Channel 2 O b(t)
-0.5
-1
-1.5
-2
-2.5
1
0.95
0.9
0.85
Channel 2 O r²(t)
6
5
4
3
2
EQ Magnitudes
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

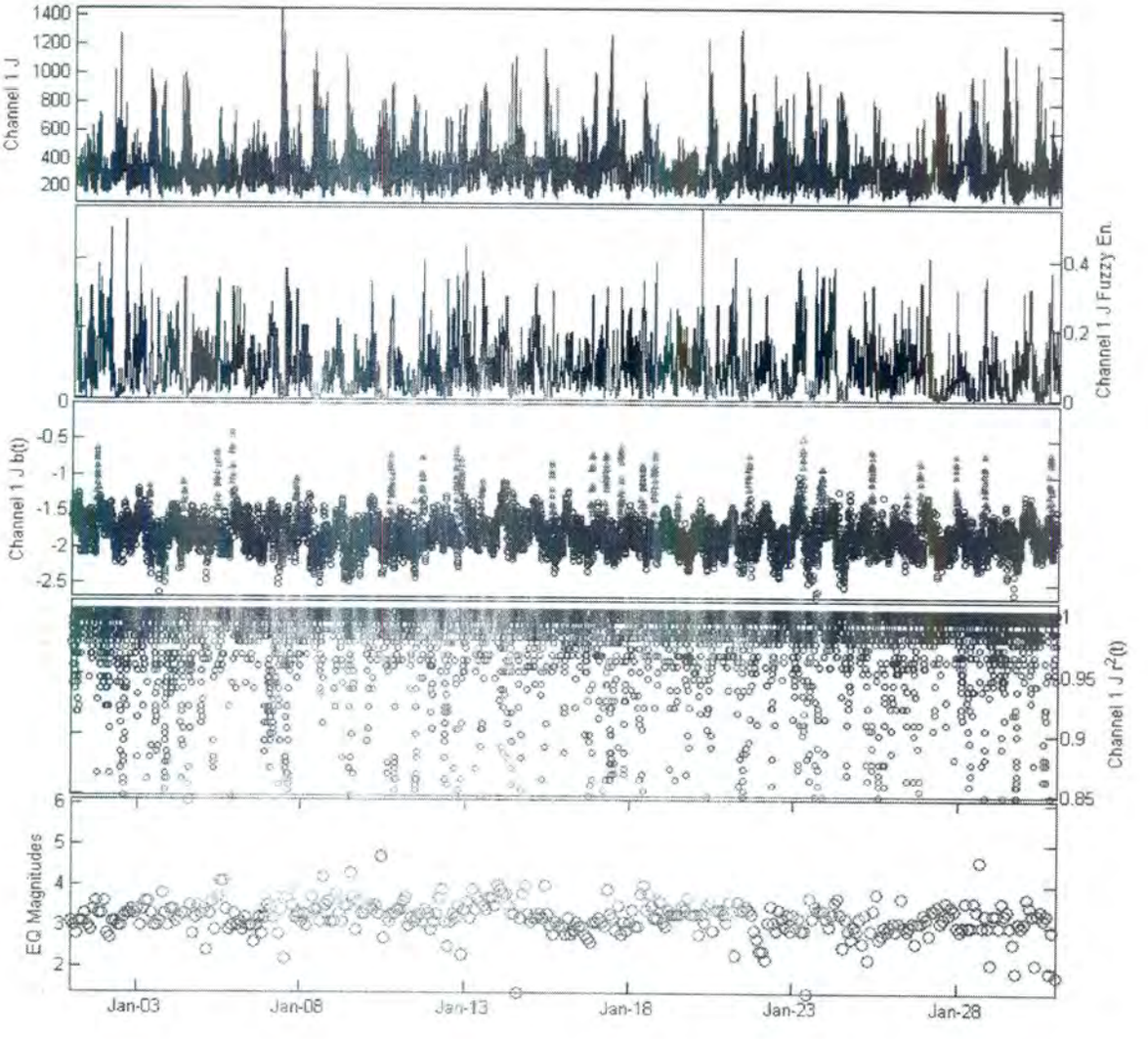

Channel 3 O
1500
1000
500
Channel 3 O Fuzzy En.
0.4
0.2
0
Channel 3 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 3 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

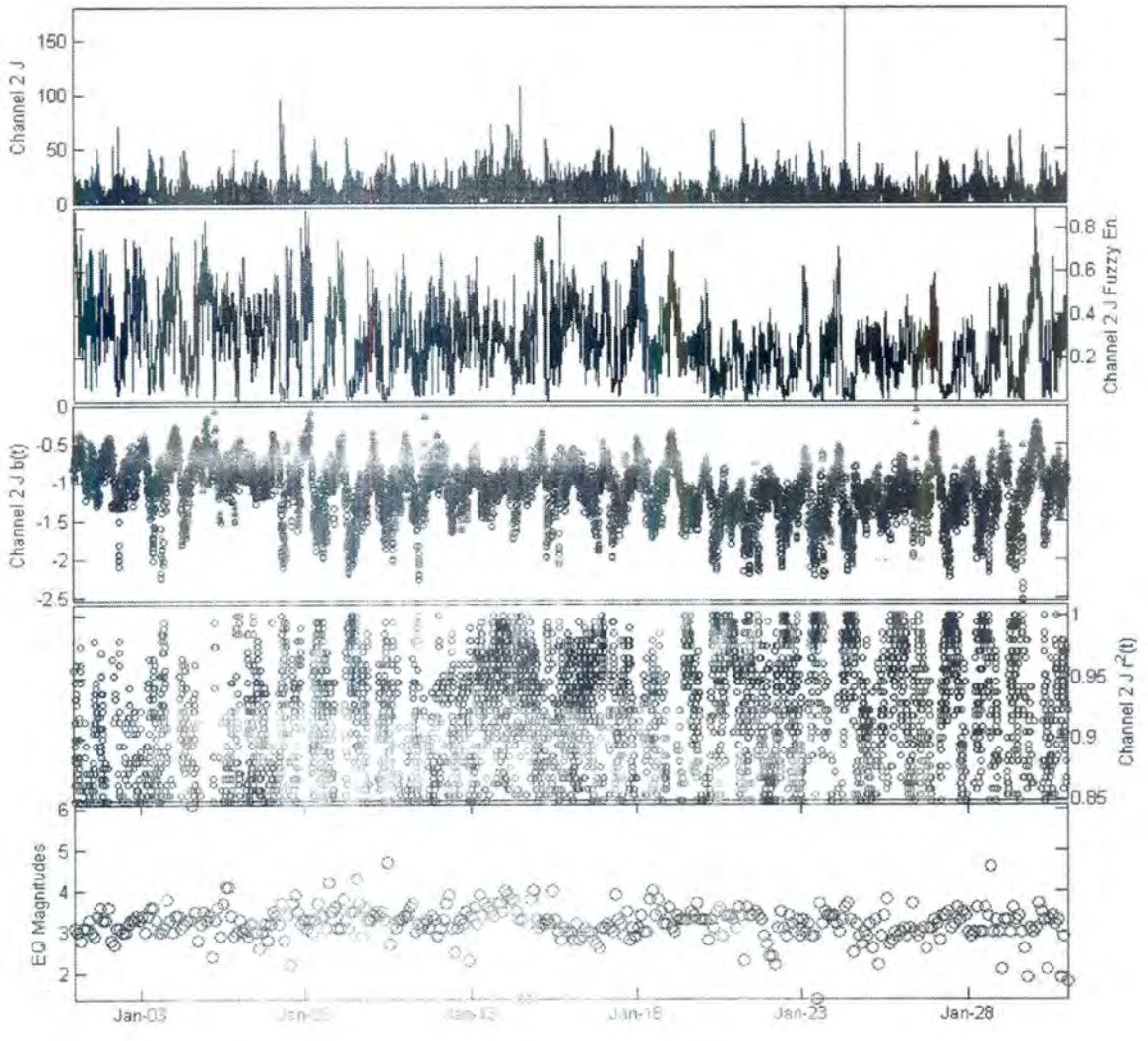

Channel 4 O
1200
1000
800
600
400
200
Channel 4 O Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 4 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-18
Jan-23
Jan-28

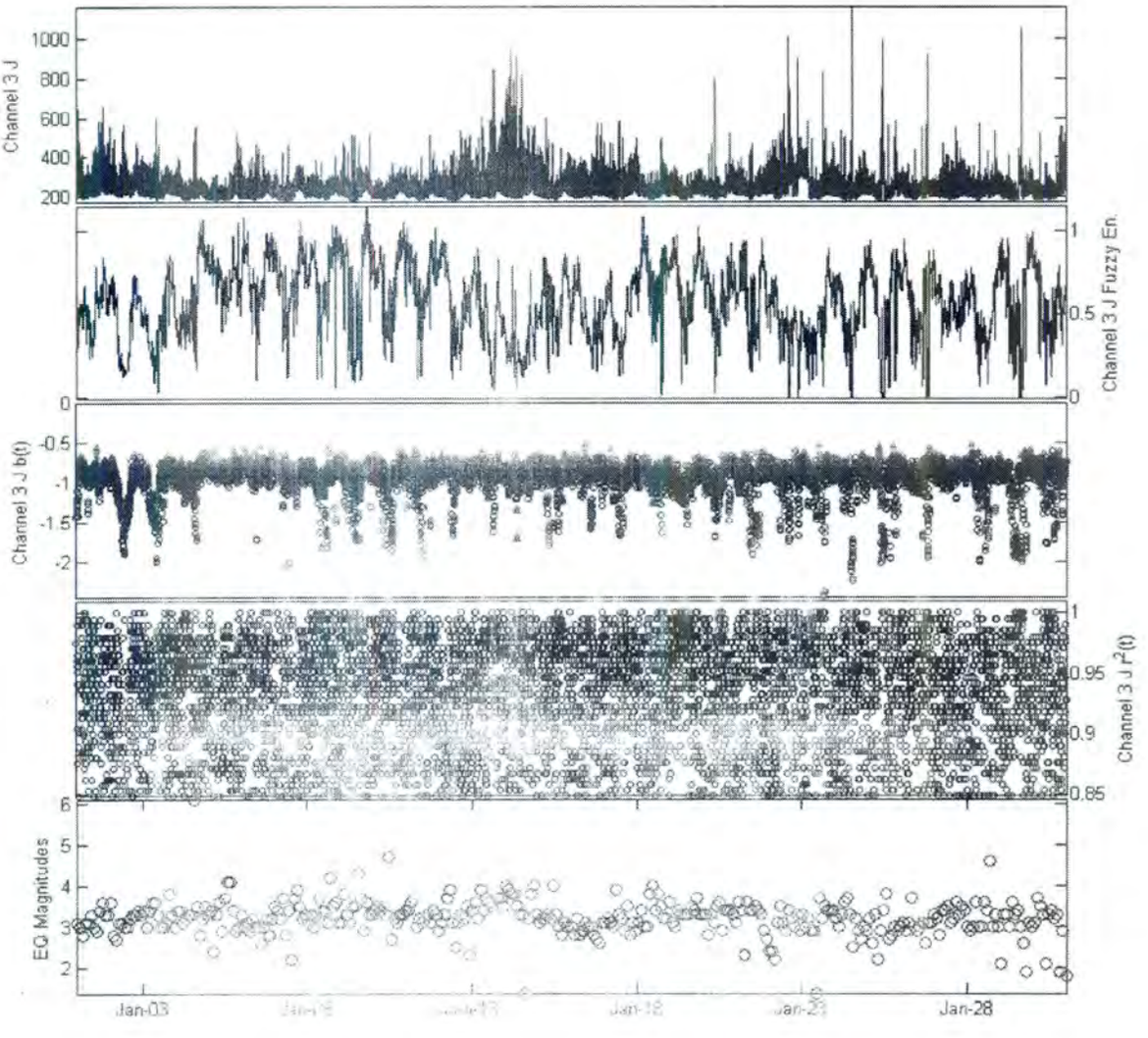

Channel 5 O
800
600
400
200
Channel 5 O Fuzzy En.
1.5
1
0.5
0
Channel 5 O b(t)
-0.5
-1
-1.5
Channel 5 O r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

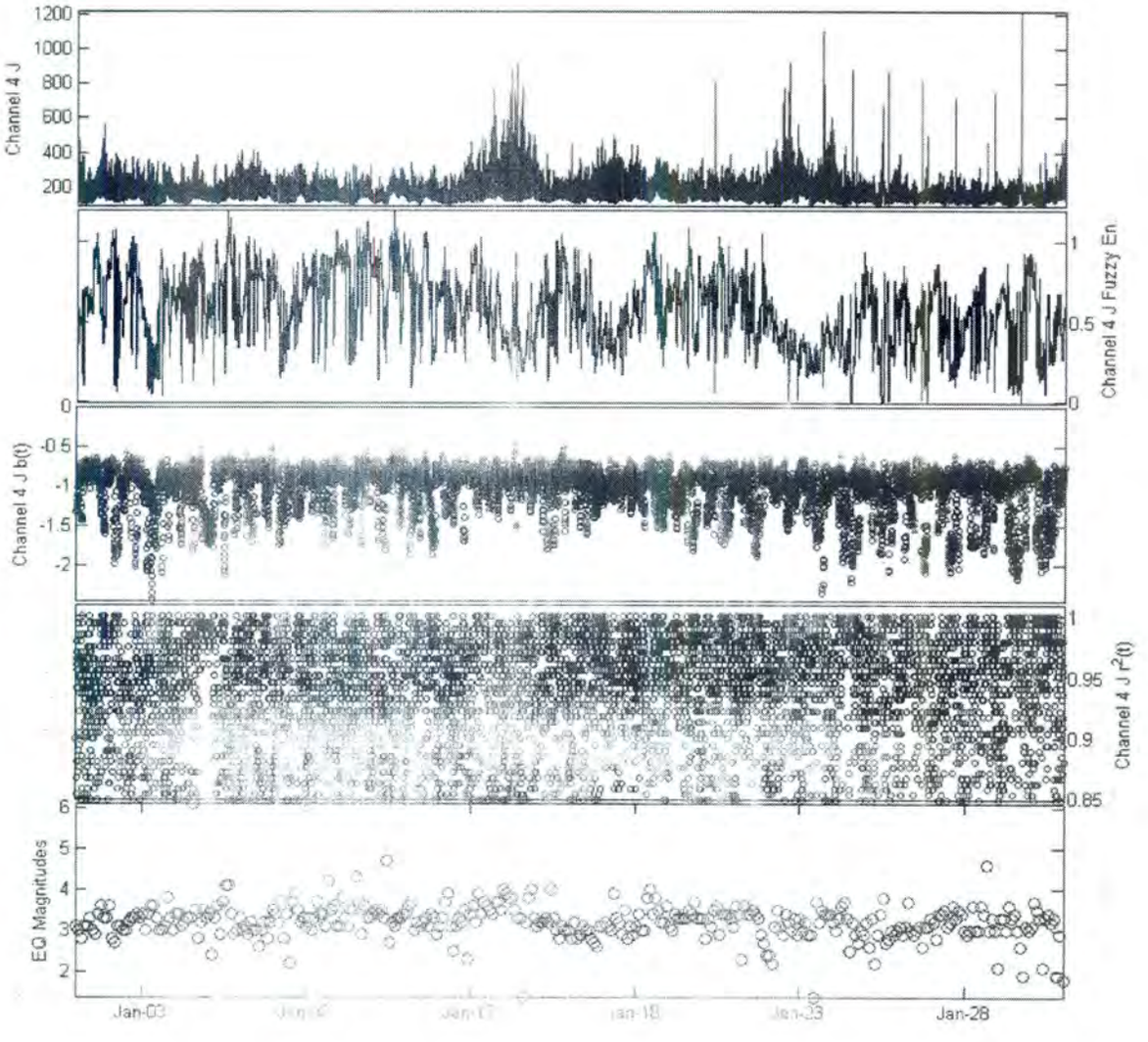

Channel 6 O
1000
800
600
Channel 6 O Fuzzy En.
2
1.5
1
0.5
0
Channel 6 O b(t)
-0.5
-1
-1.5
Channel 6 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

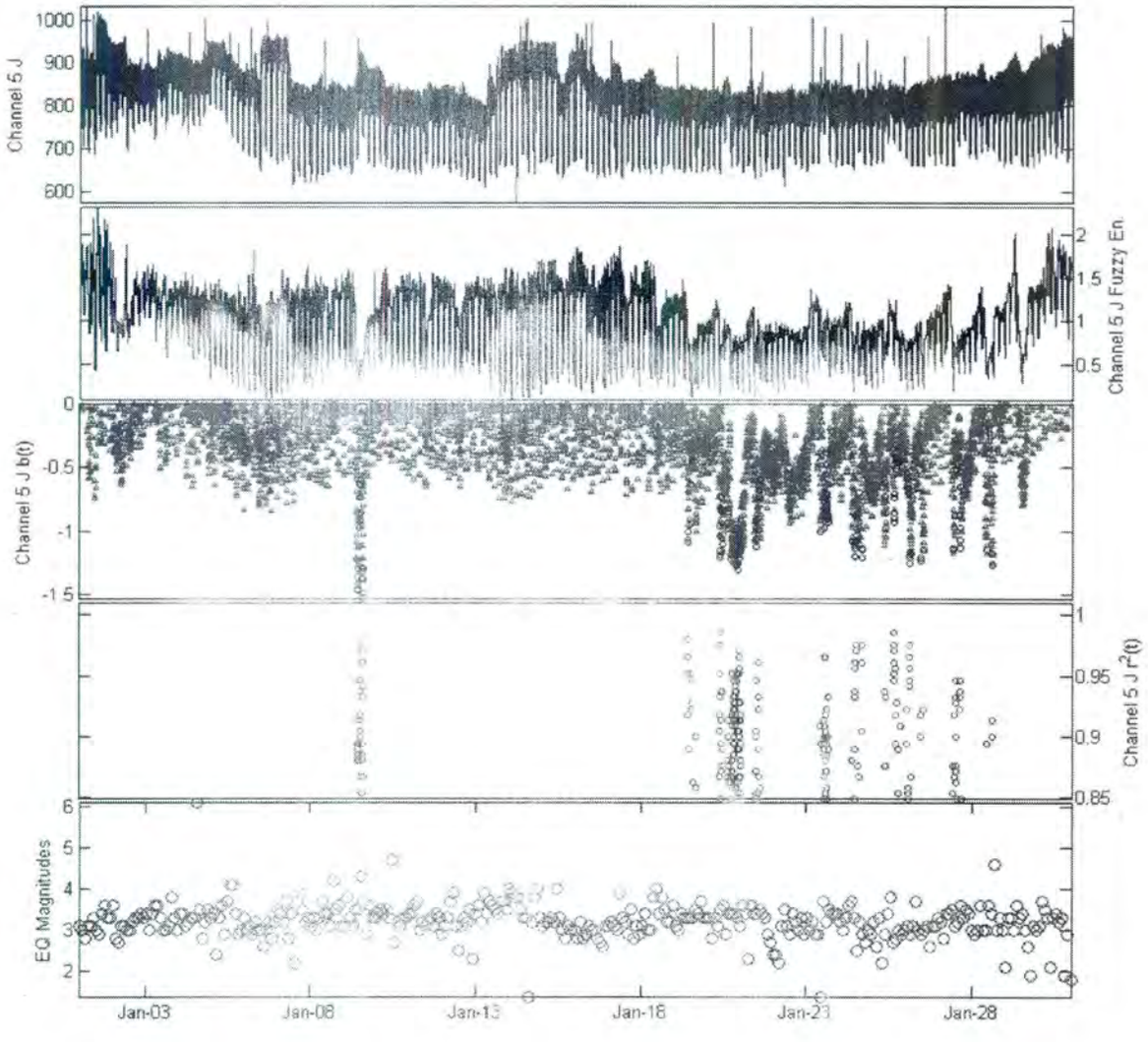

Channel 7 O
13.5
13
12.5
12
Channel 7 O Fuzzy En.
1
0.8
0.6
0.4
0.2
Channel 7 O b(t)
0
-0.5
-1
-1.5
Channel 7 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

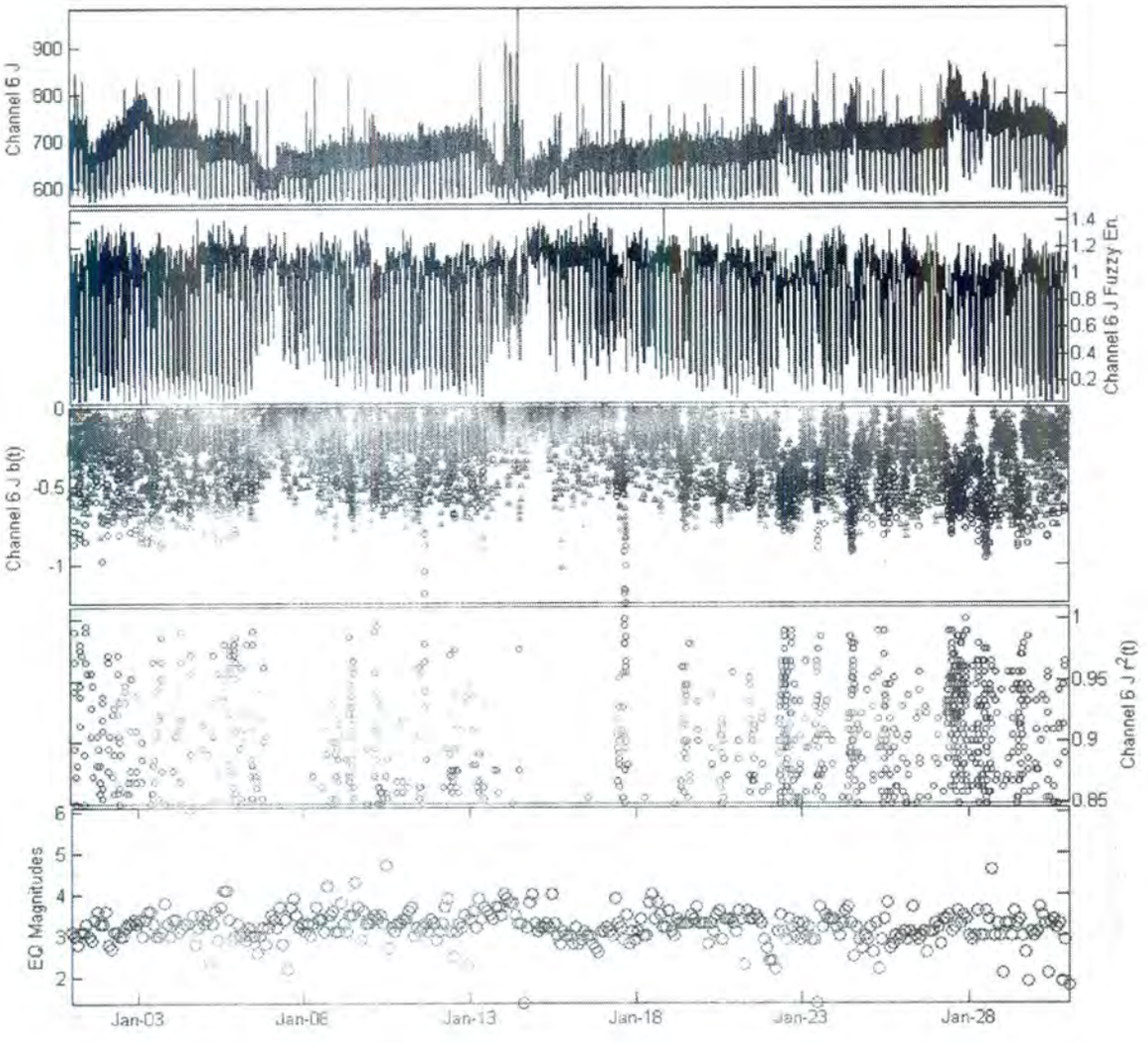

Channel 1 J
1400
1200
1000
800
600
400
200
Channel 1 J Fuzzy En.
0.4
0.2
0
Channel 1 J b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

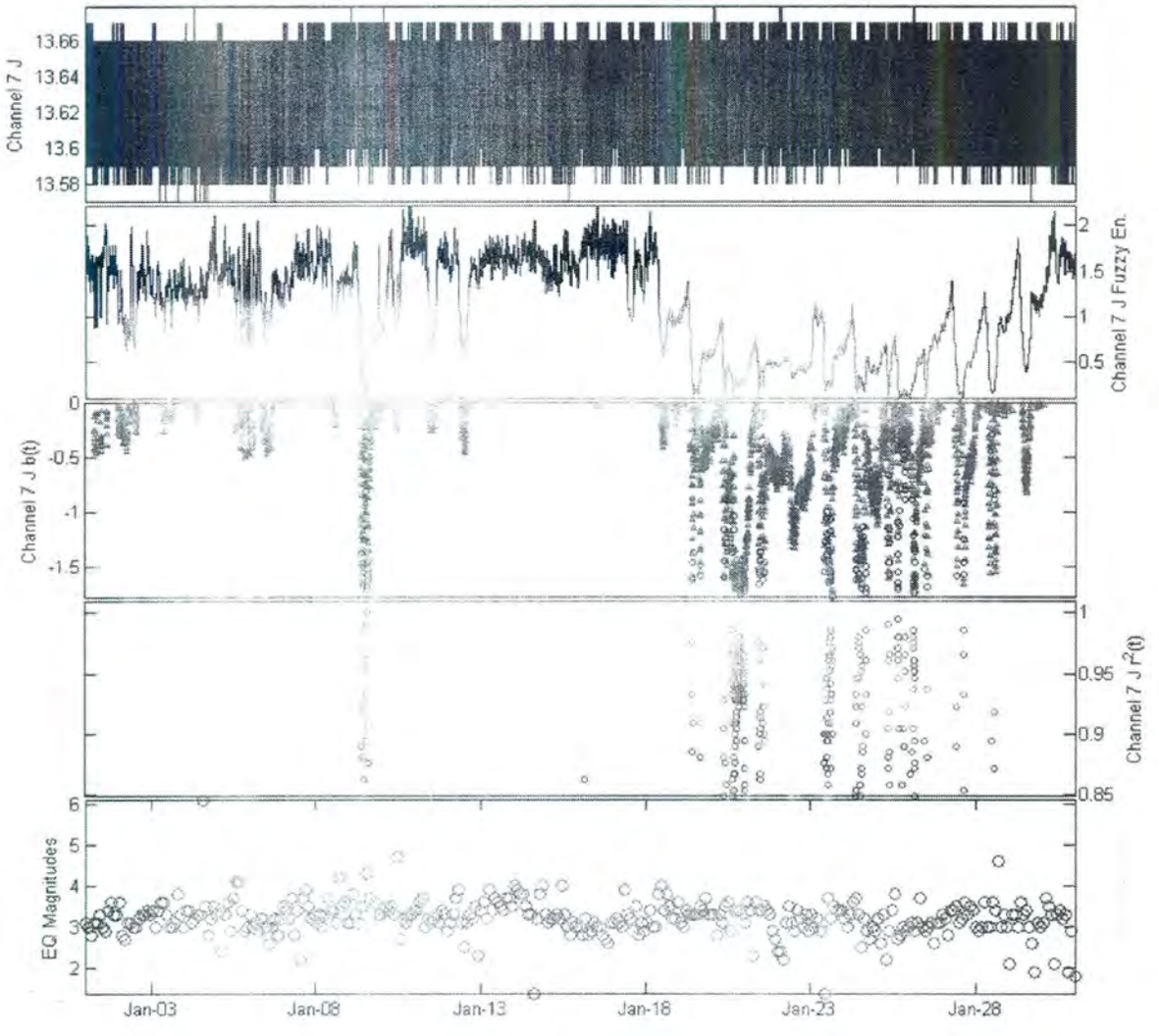

Channel 2 J
150
100
50
0
Channel 2 J Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 2 J b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-18
Jan-23
Jan-28

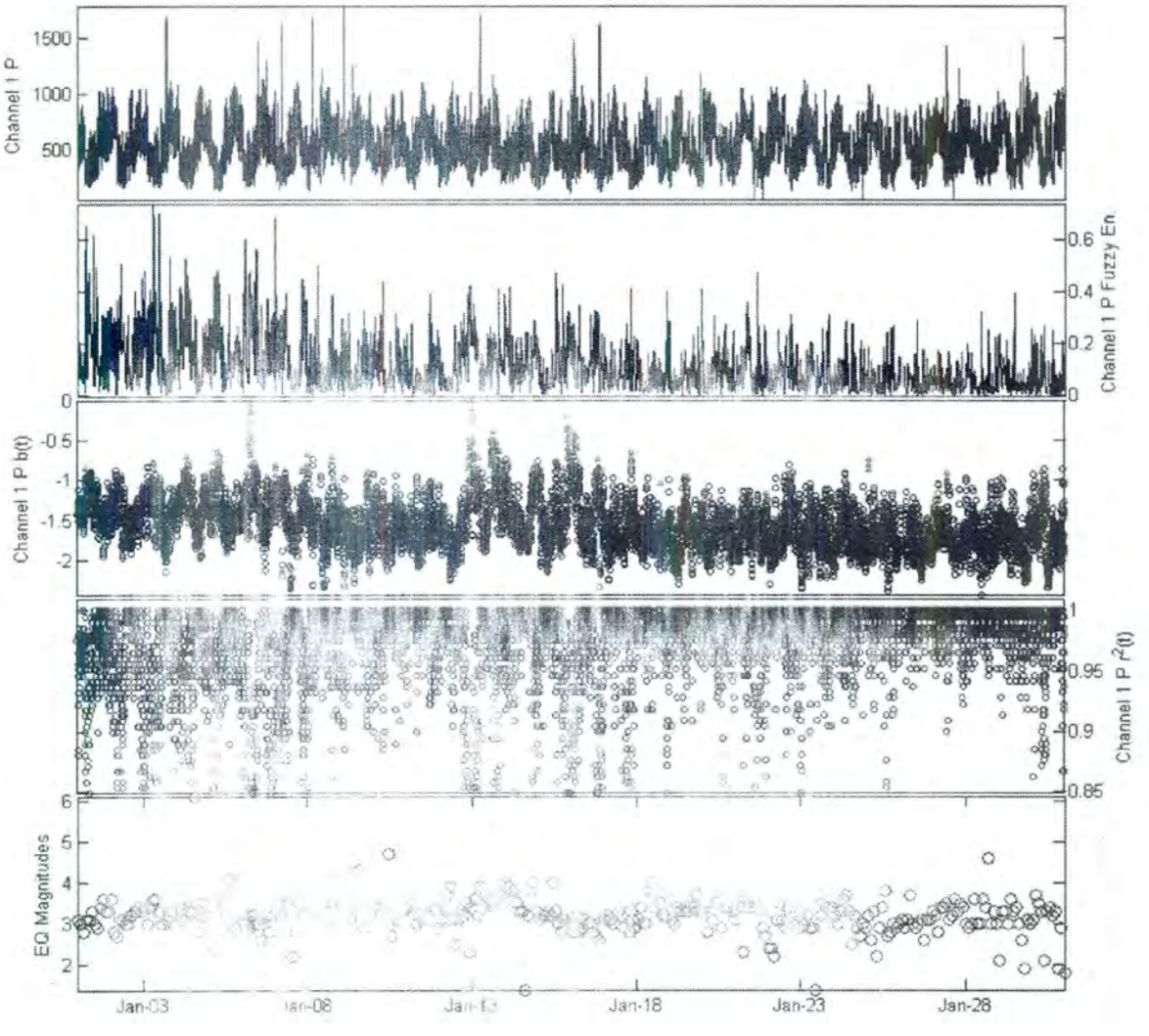

Channel 3 J
1000
800
600
400
200
Channel 3 J Fuzzy En.
1
0.5
0
Channel 3 J b(t)
-0.5
-1
-1.5
-2
Channel 3 J F²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-28

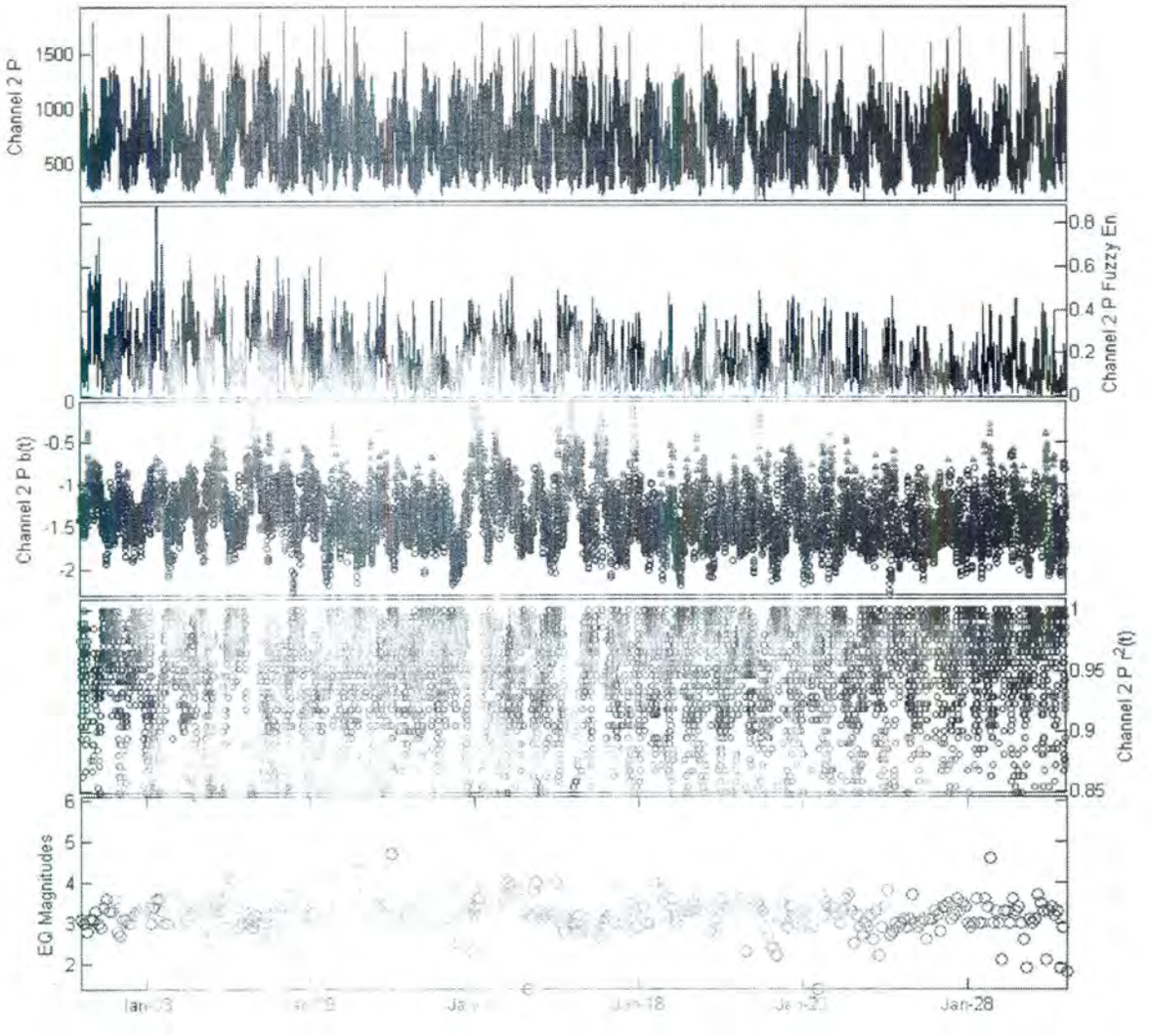

Channel 4 J
1200
1000
800
600
400
200
Channel 4 J Fuzzy En
1
0.5
0
Channel 4 J b(t)
-0.5
-1
-1.5
-2
Channel 4 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-28

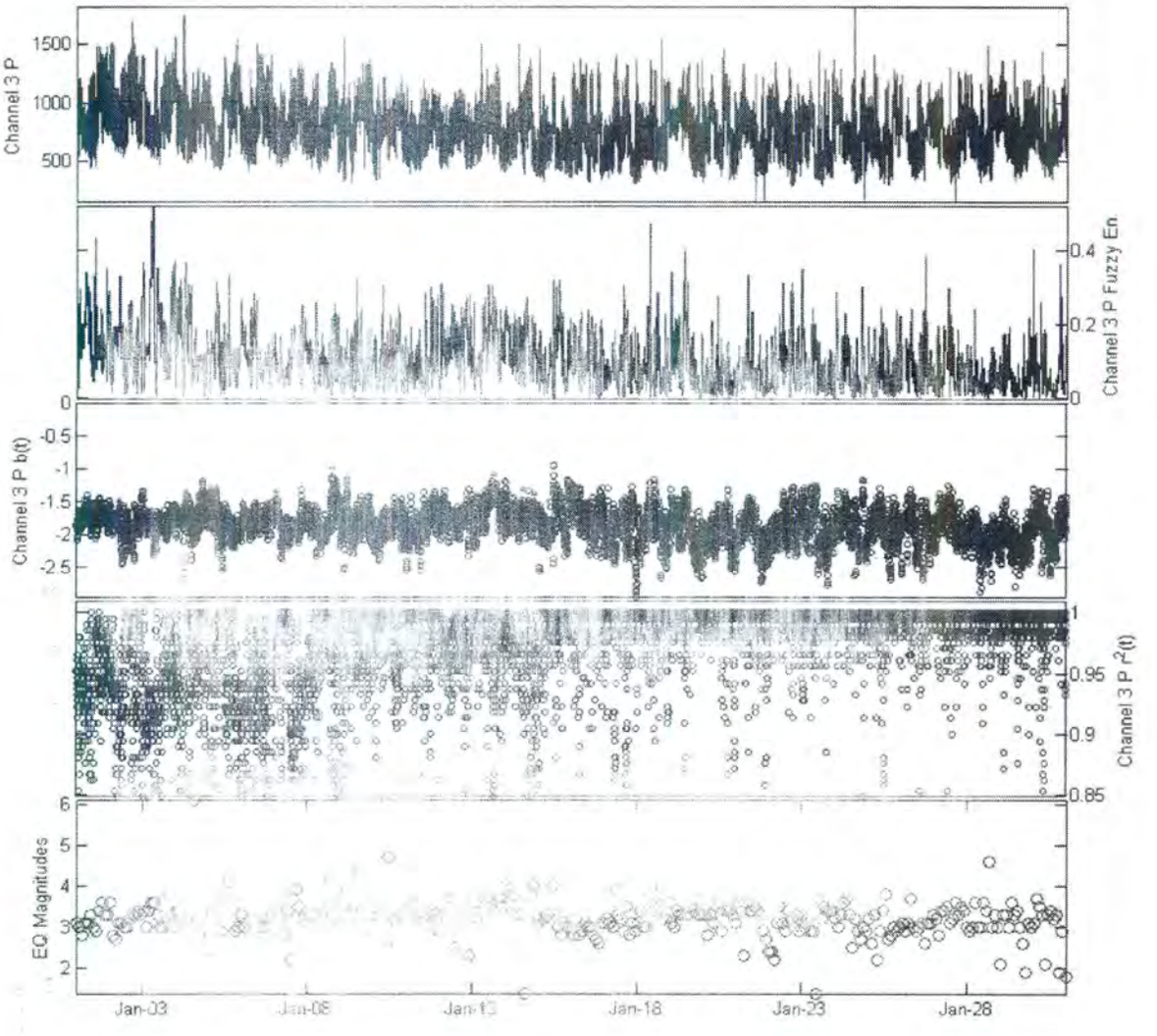

Channel 5 J
1000
900
800
700
600
Channel 5 J Fuzzy En.
2
1.5
1
0.5
Channel 5 J b(t)
0
-0.5
-1
-1.5
Channel 5 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

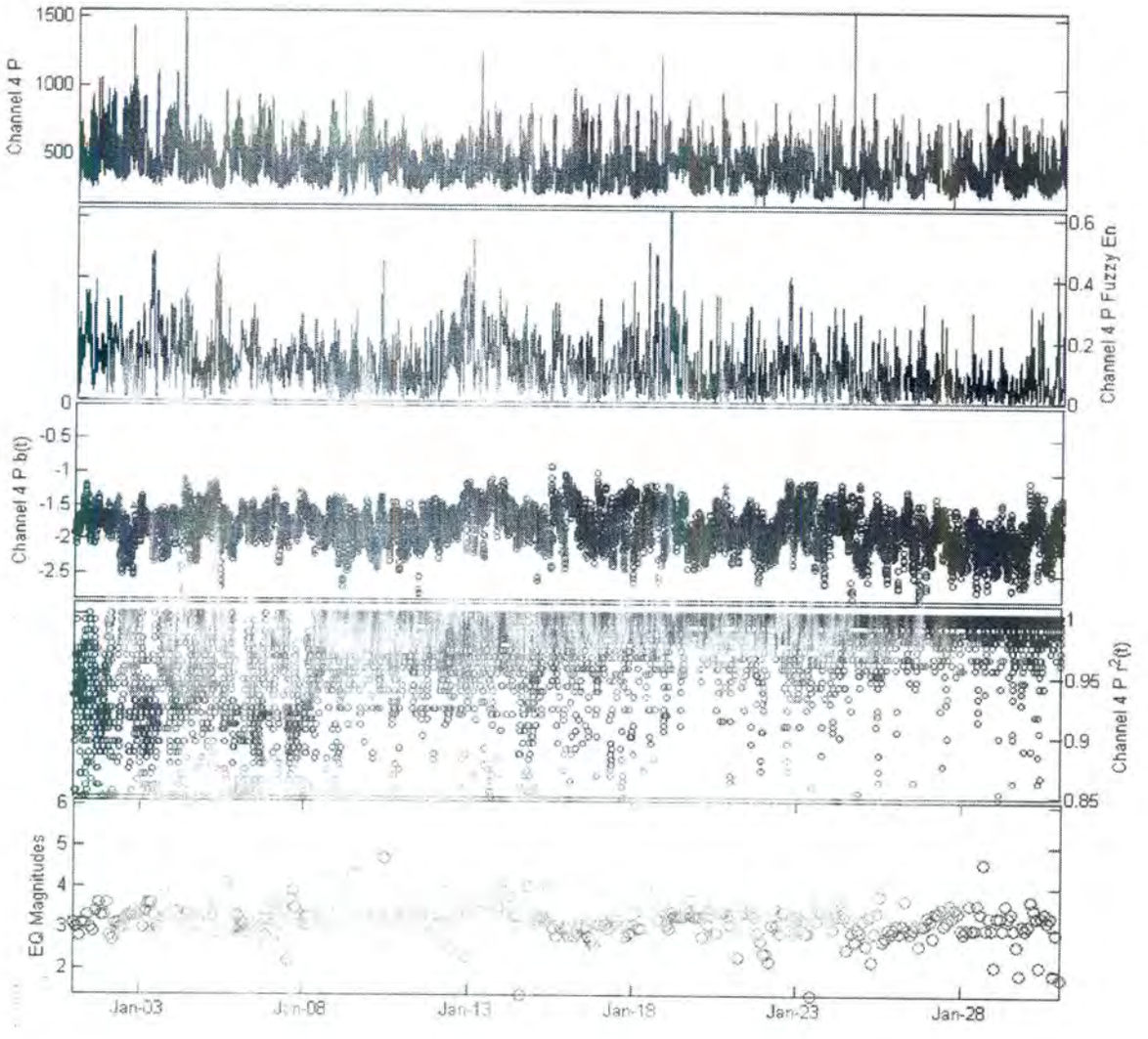

Channel 6 J
900
800
700
600
Channel 6 J Fuzzy En.
1.4
1.2
1
0.8
0.6
0.4
0.2
Channel 6 J b(t)
0
-0.5
-1
Channel 6 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

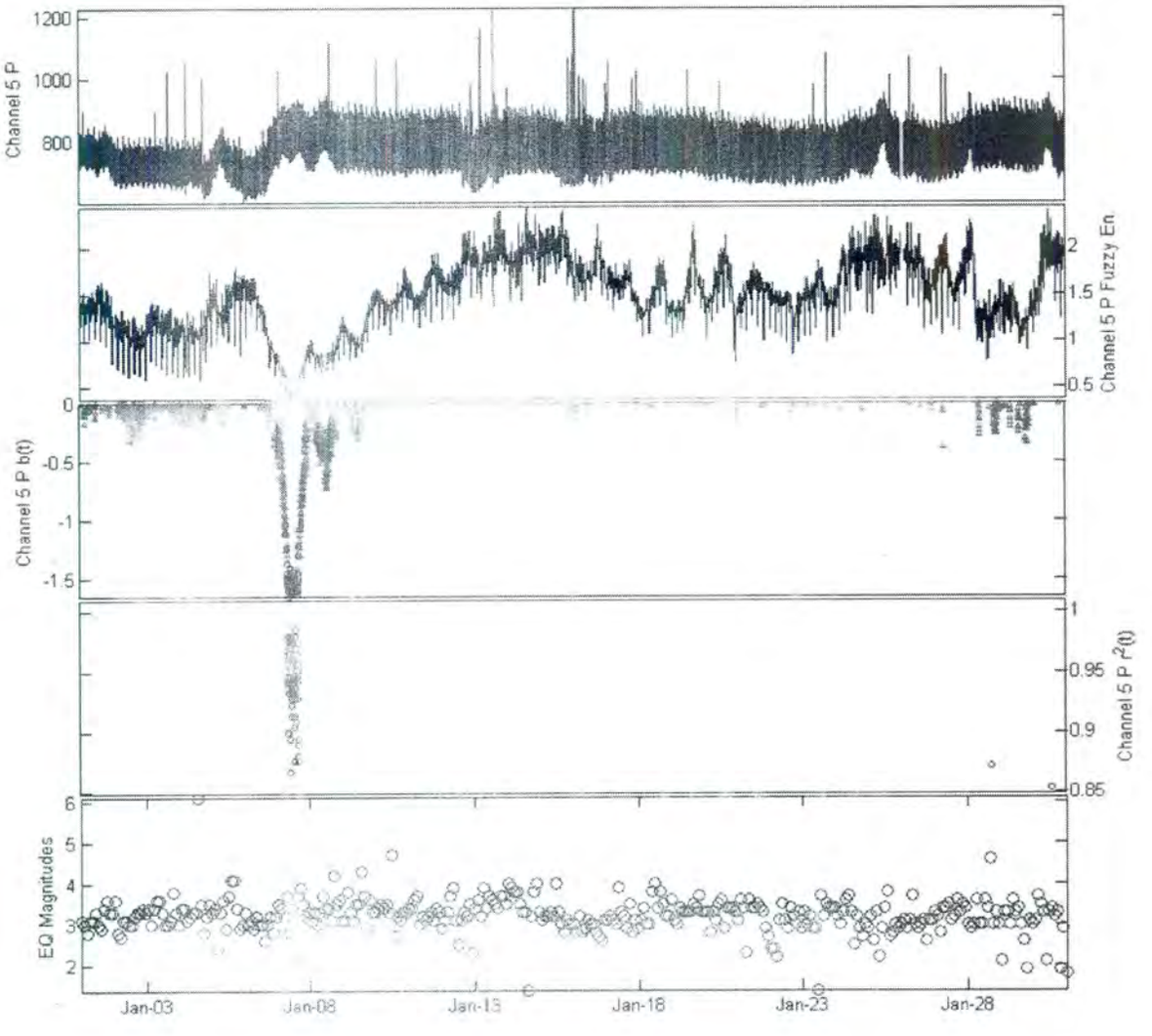

Channel 7 J
13.66
13.64
13.62
13.6
13.58
Channel 7 J Fuzzy En.
2
1.5
1
0.5
0
Channel 7 J b(t)
-0.5
-1
-1.5
Channel 7 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

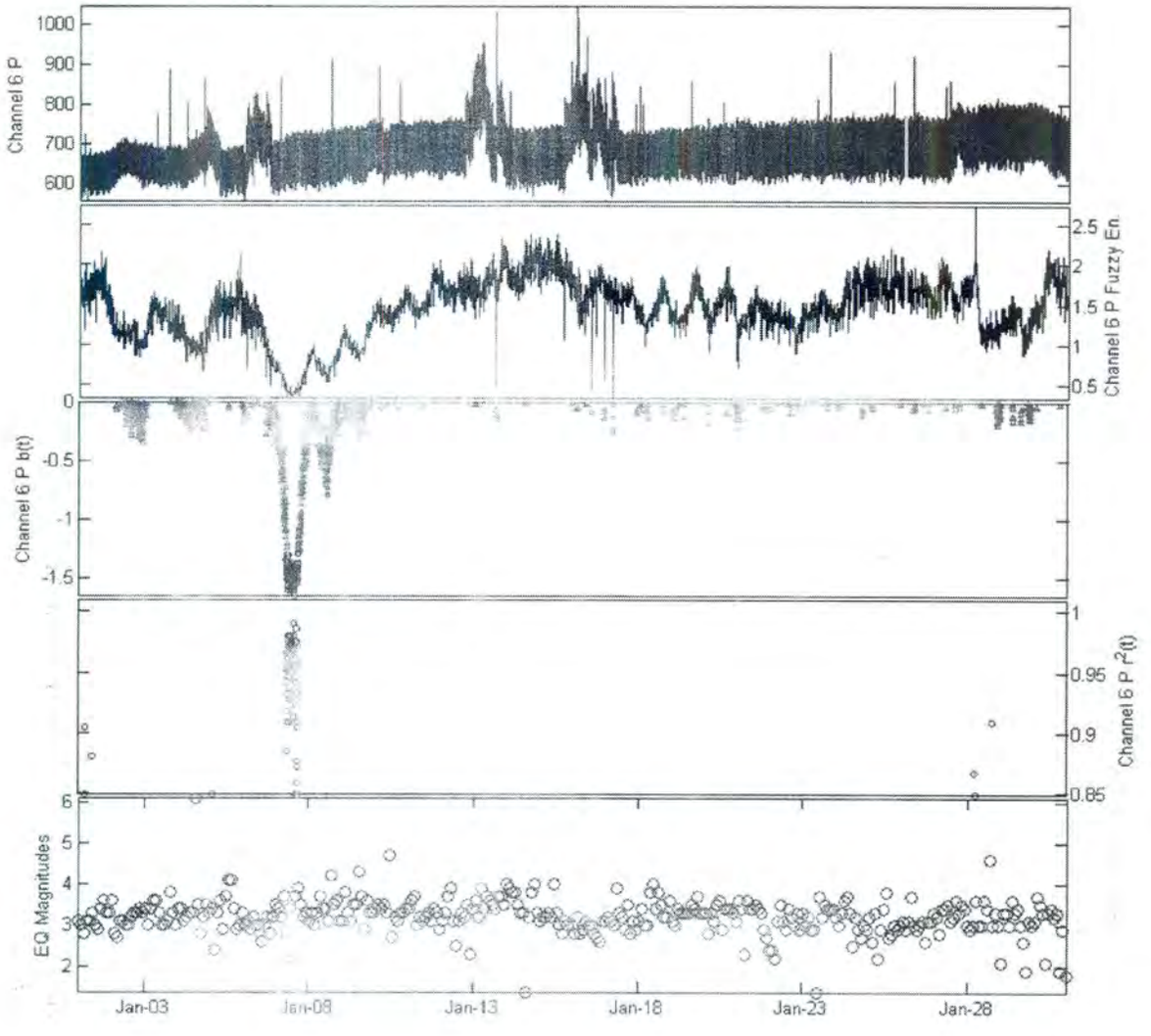

Channel 1 P
1500
1000
500
Channel 1 P Fuzzy En
0.6
0.4
0.2
0
Channel 1 P b(t)
-0.5
-1
-1.5
-2
Channel 1 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

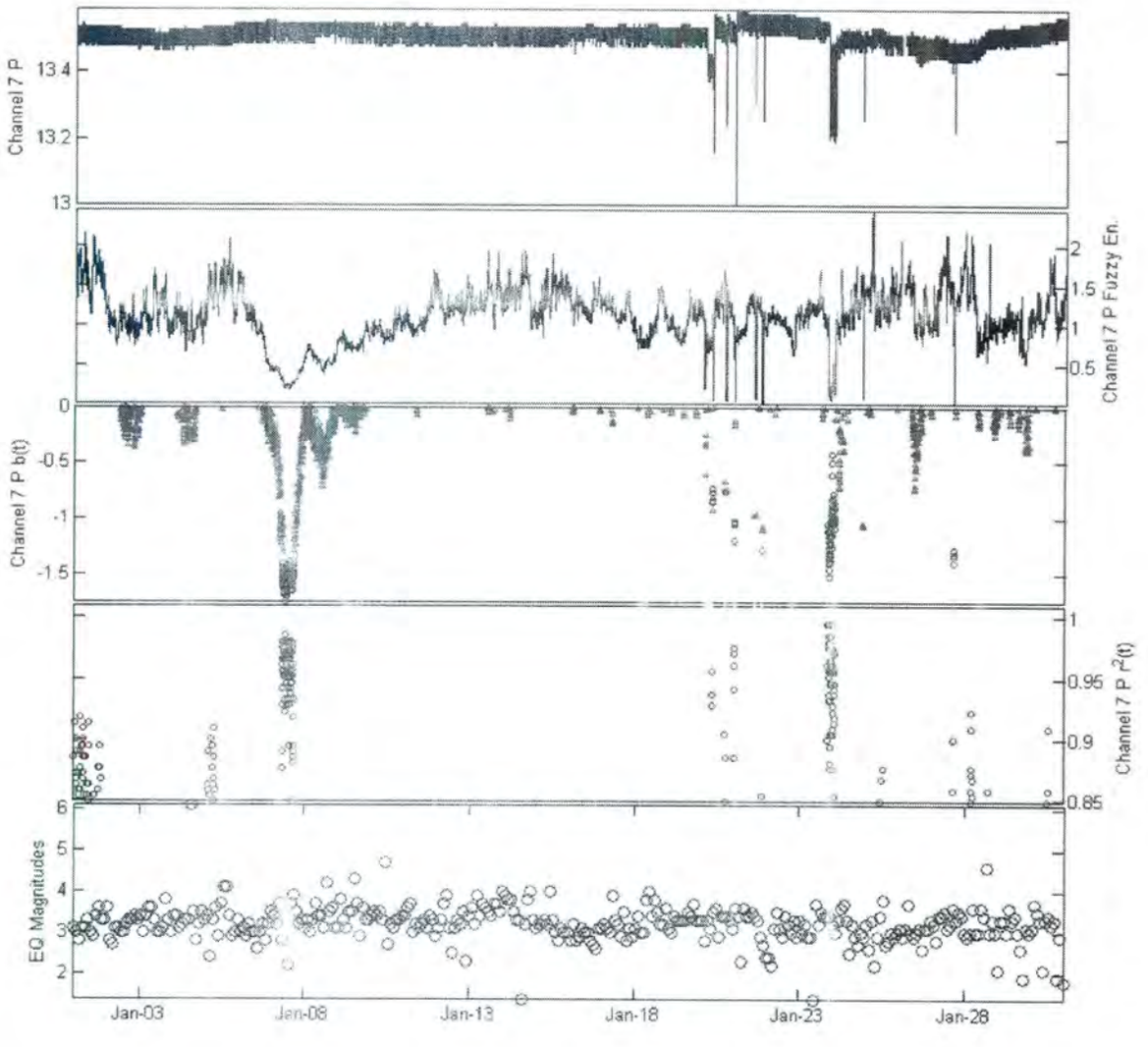

Channel 2 P
1500
1000
500
0
Channel 2 P Fuzzy En
0.8
0.6
0.4
0.2
0
Channel 2 P b(t)
-0.5
-1
-1.5
-2
Channel 2 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-3
Jan-18
Jan-28

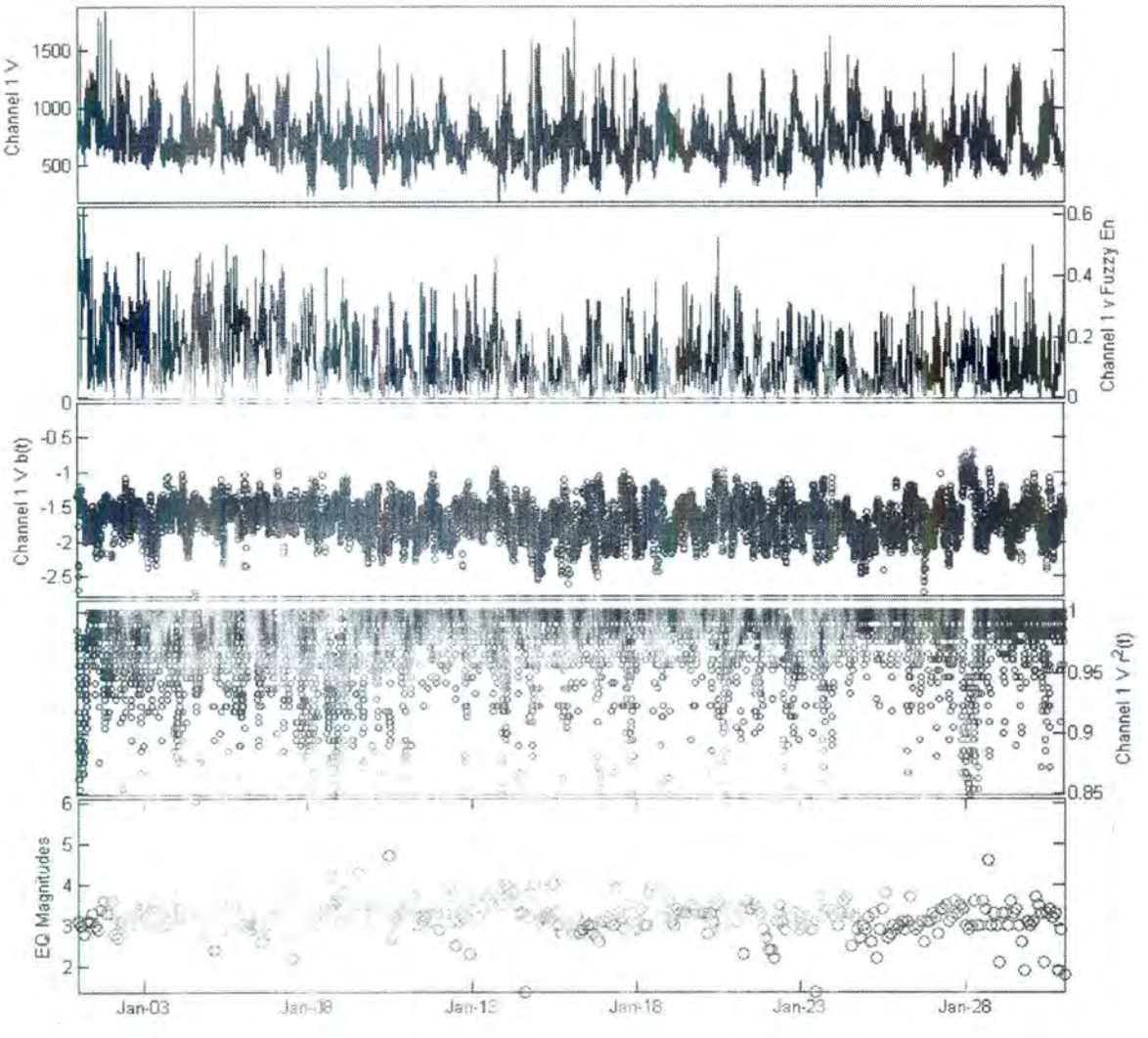

Channel 3 P
1500
1000
500
Channel 3 P Fuzzy En
0.4
0.2
0
Channel 3 P b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 3 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

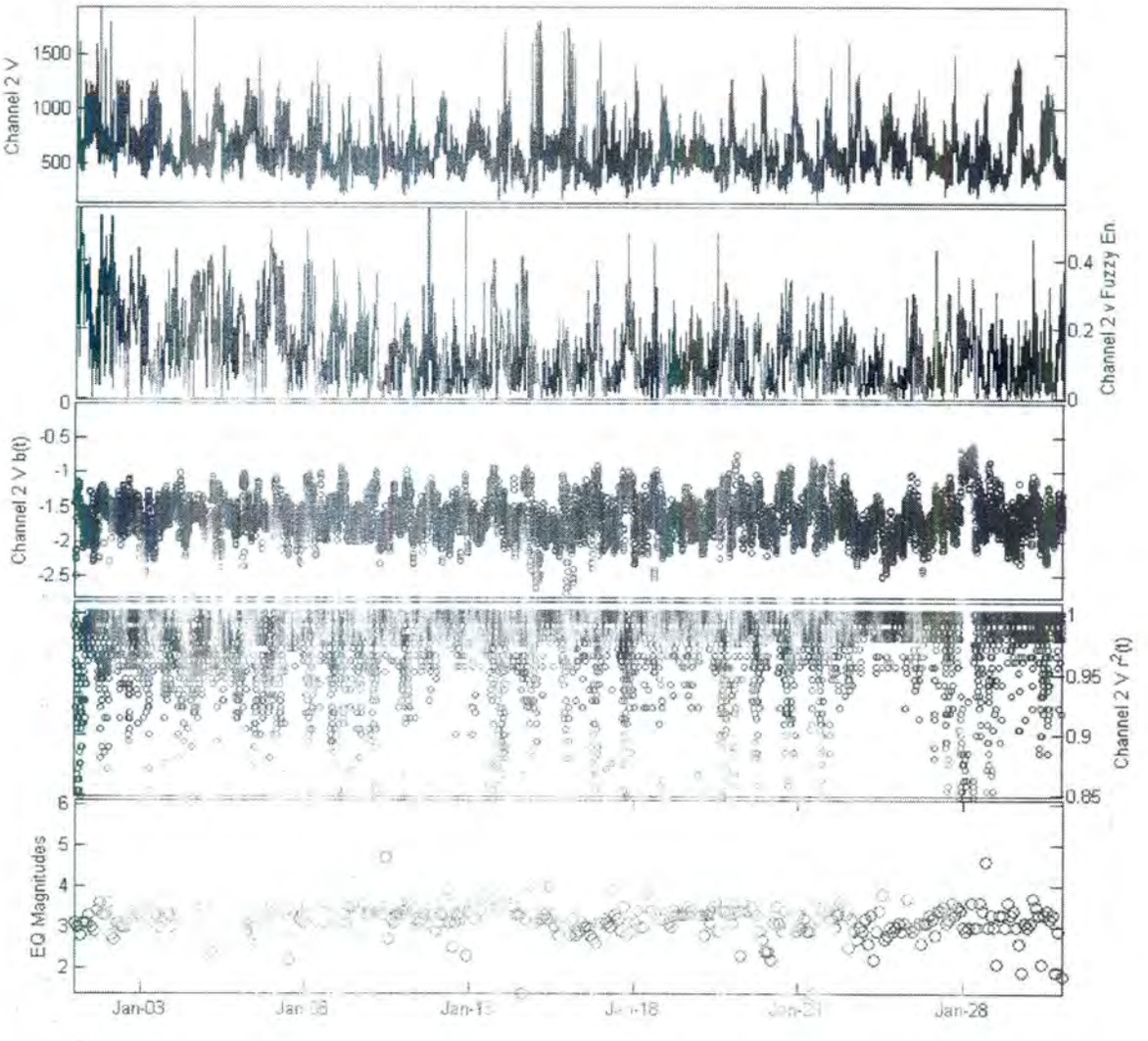

Channel 4 P
1500
1000
500
Channel 4 P Fuzzy En
0.6
0.4
0.2
0
Channel 4 P b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

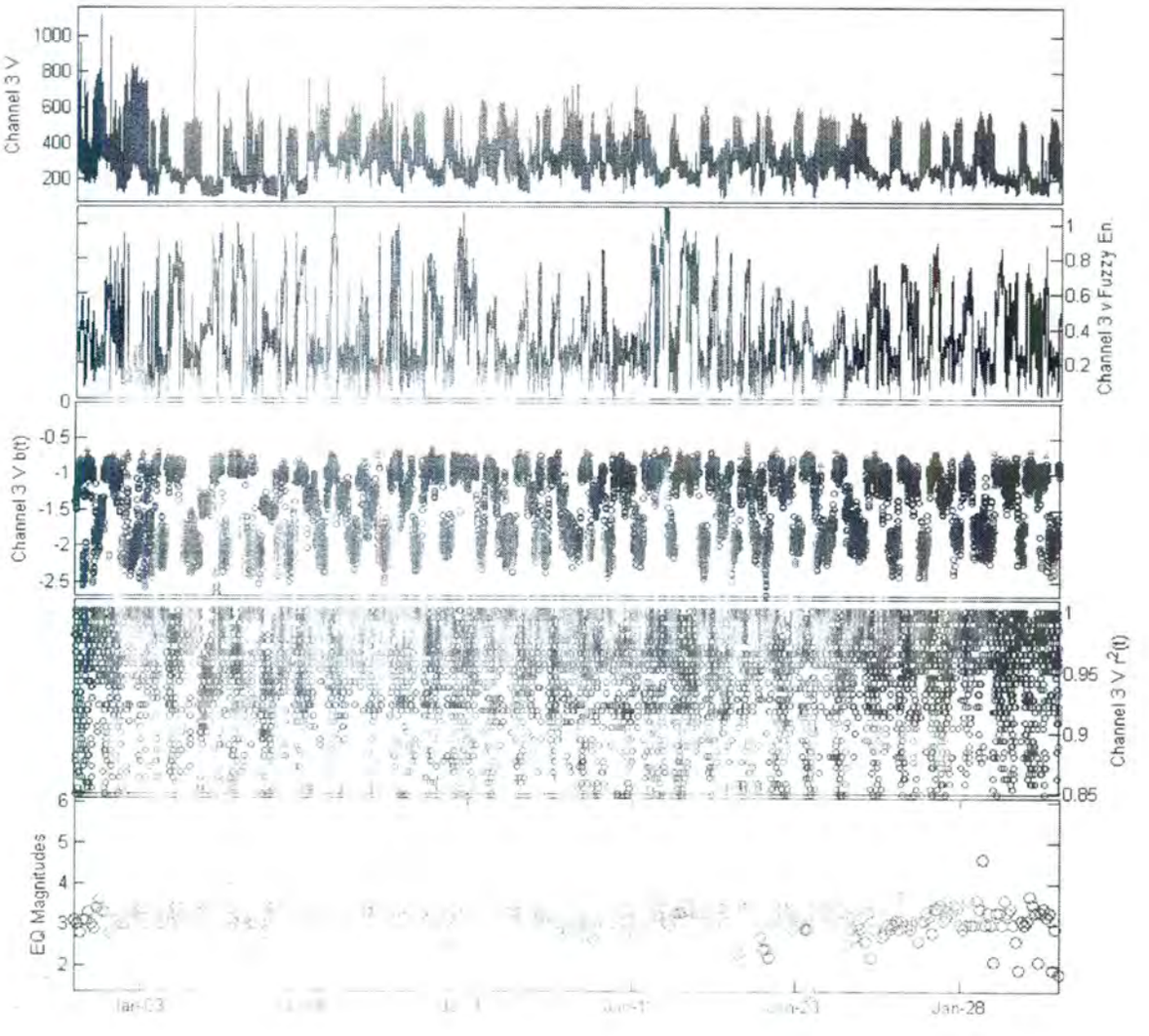

Channel 5 P
1200
1000
800
Channel 5 P Fuzzy En
2
1.5
1
0.5
Channel 5 P b(t)
0
-0.5
-1
-1.5
Channel 5 P r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

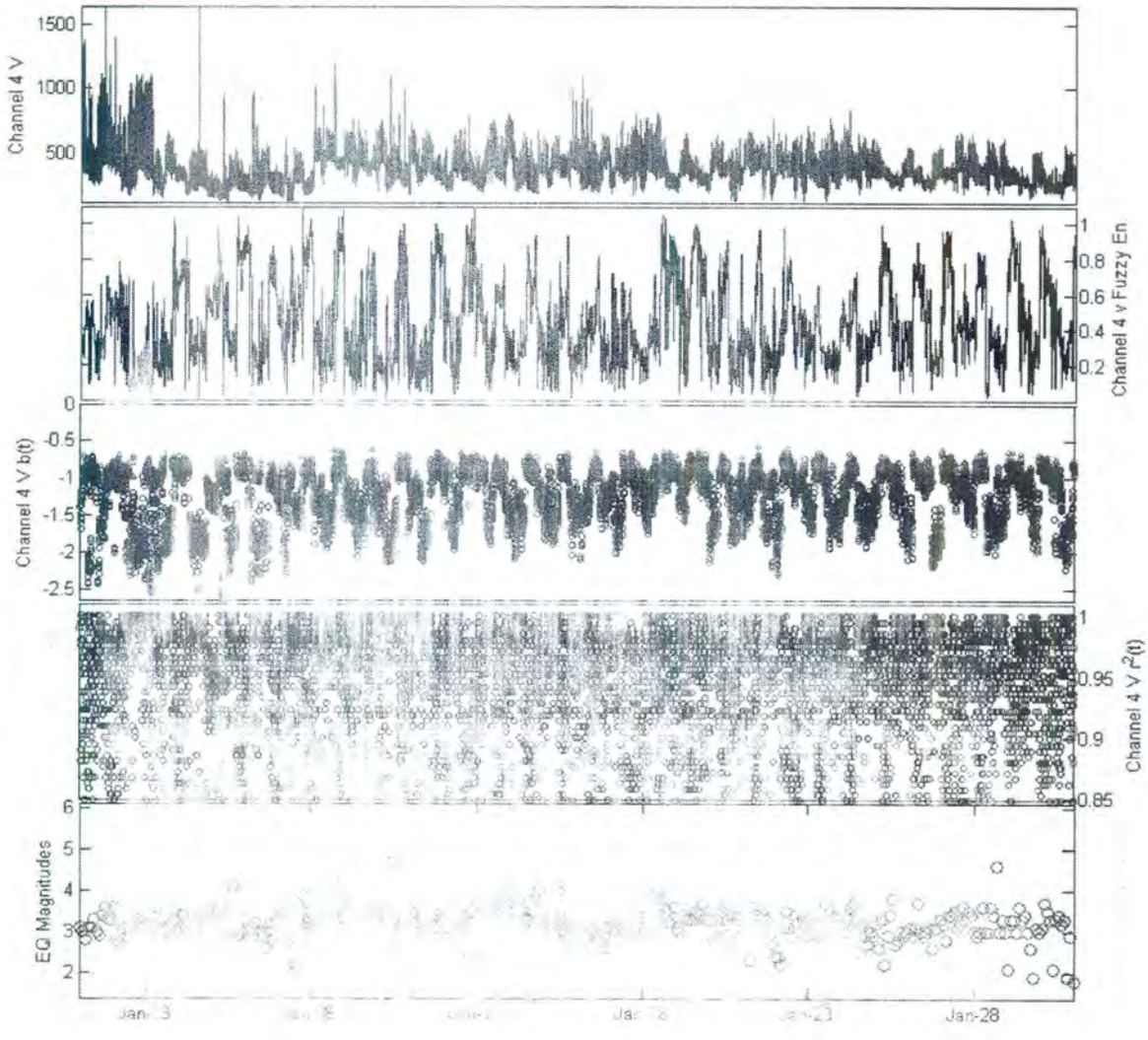

Channel 6 P
1000
900
800
700
600
Channel 6 P Fuzzy En
2.5
2
1.5
1
0.5
Channel 6 P b(t)
0
-0.5
-1
-1.5
Channel 6 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

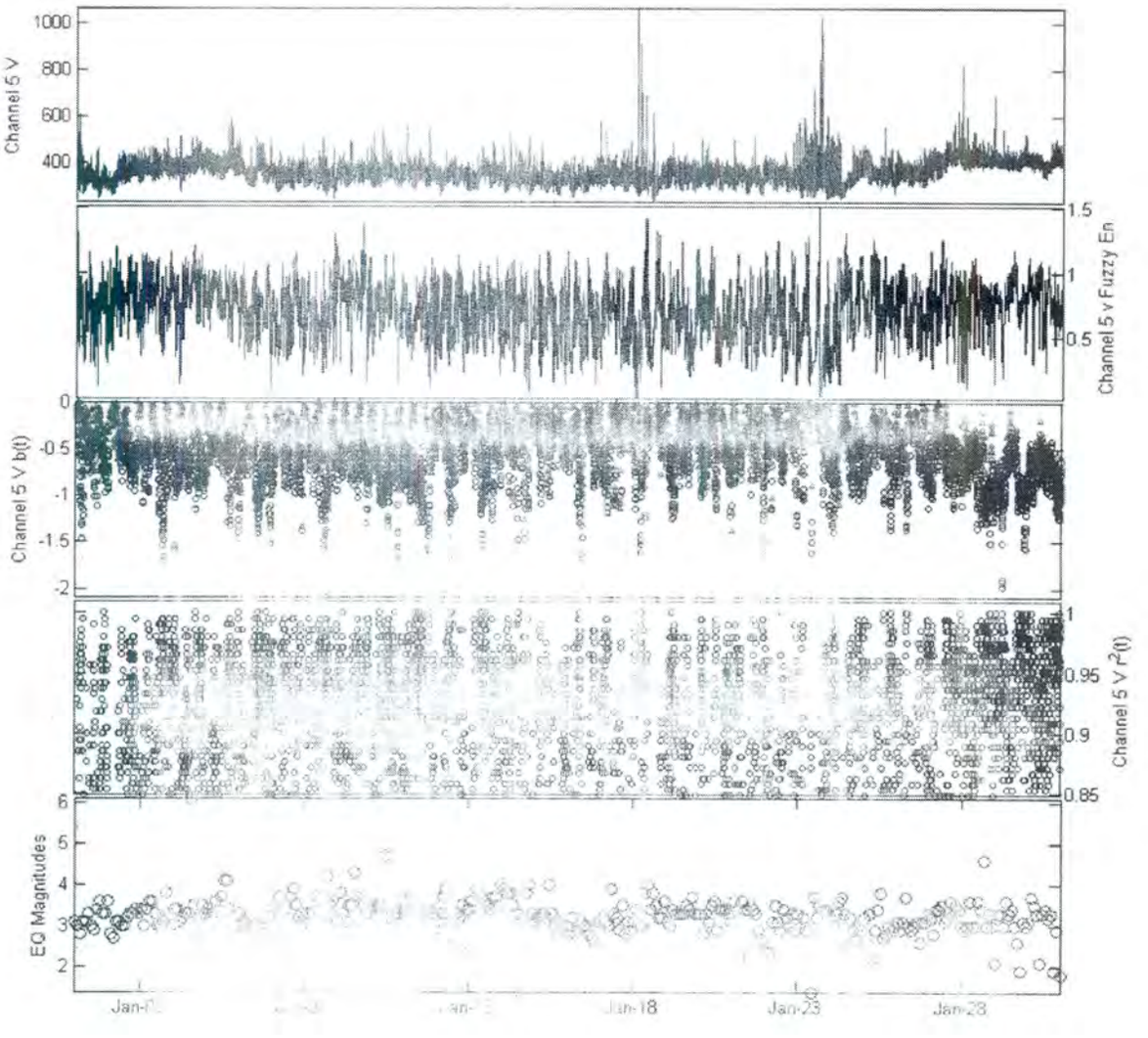

Channel 7 P
13.4
13.2
13
Channel 7 P Fuzzy En.
2
1.5
1
0.5
0
Channel 7 P b(t)
-0.5
-1
-1.5
Channel 7 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

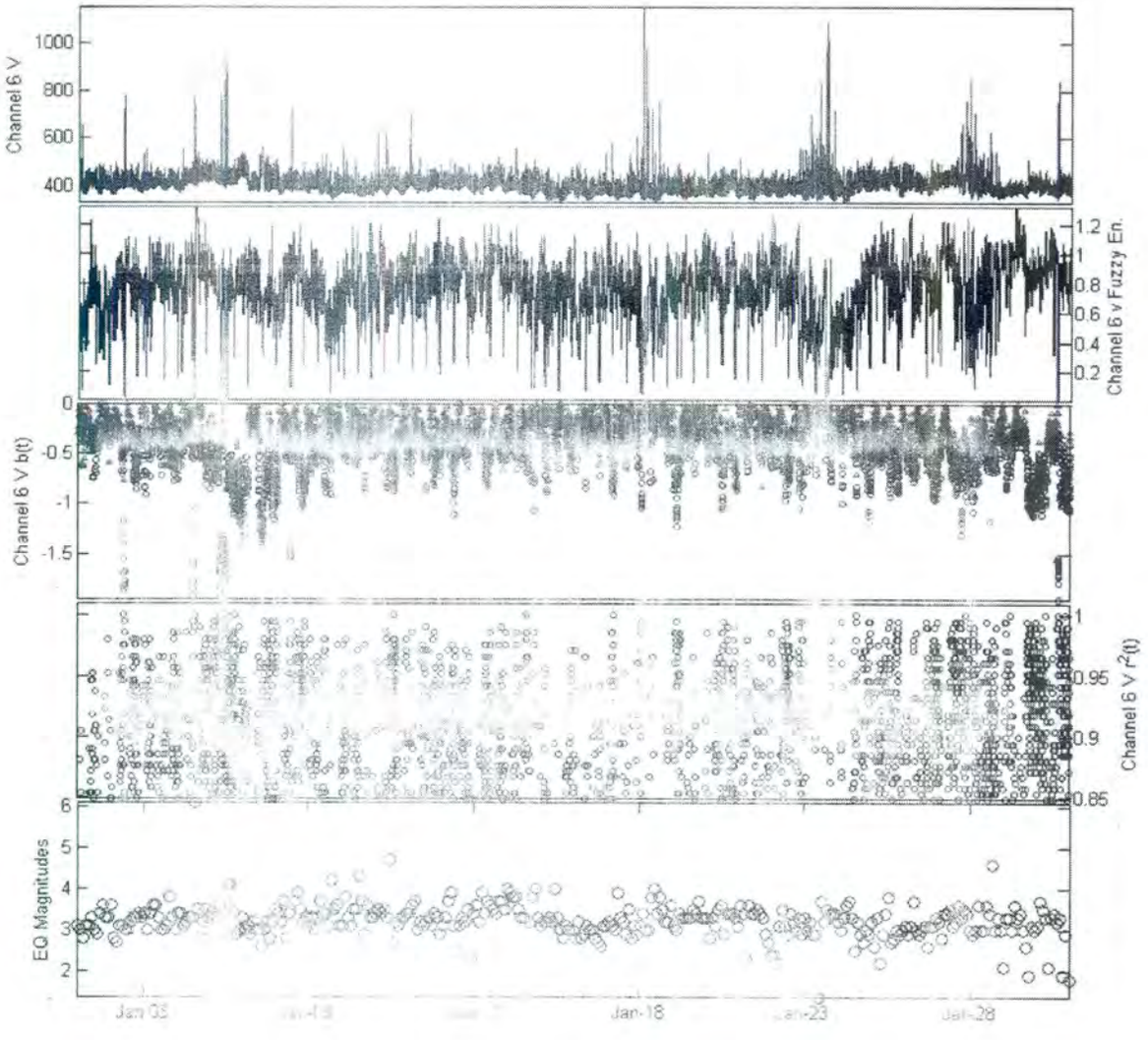

Channel 1 V
1500
1000
500
0.6
0.4
0.2
0
Channel 1 v Fuzzy En
Channel 1 V b(t)
-0.5
-1
-1.5
-2
-2.5
1
0.95
0.9
0.85
Channel 1 V r^2(t)
6
5
4
3
2
EQ Magnitudes
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

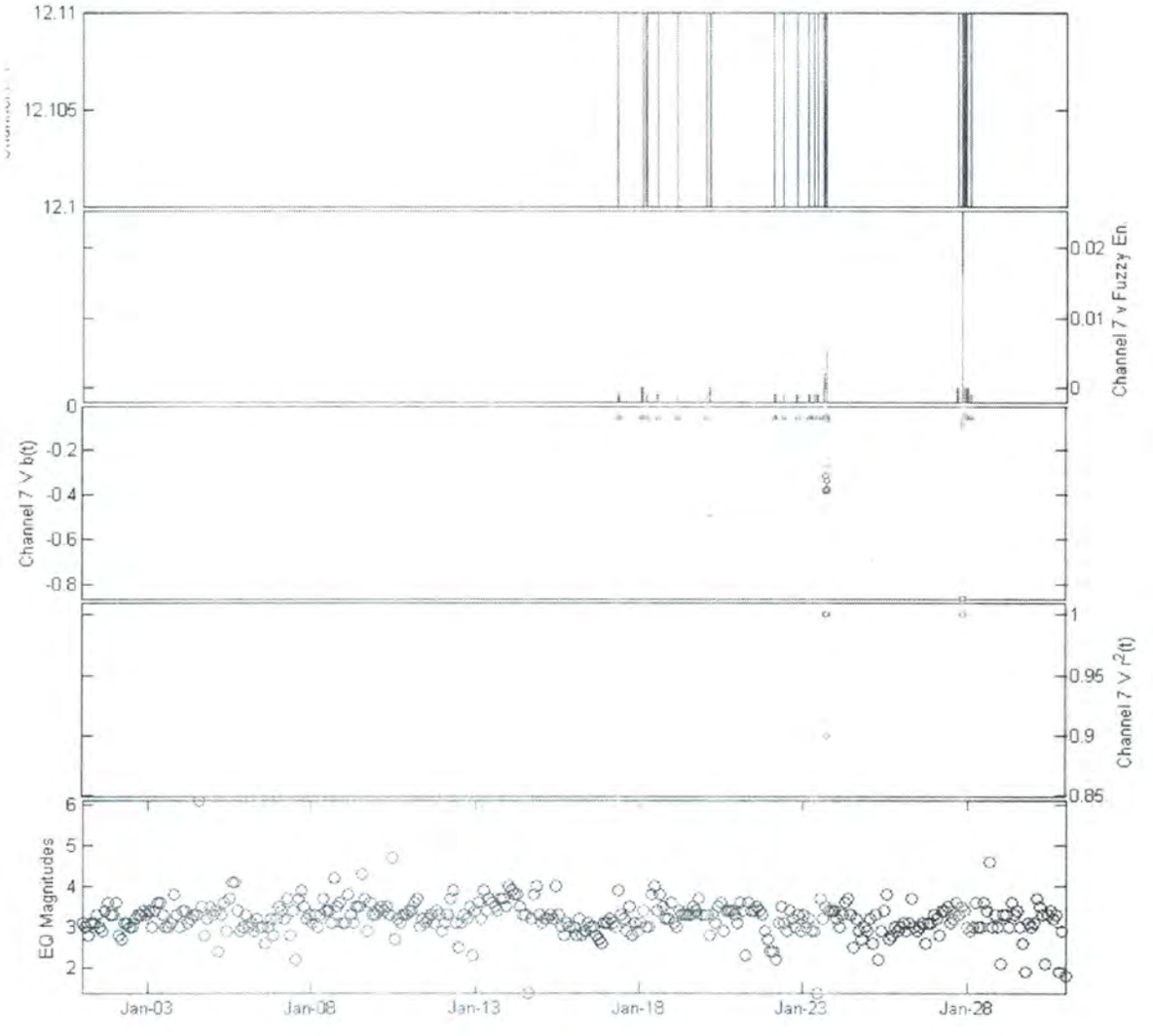

Channel 2 V
1500
1000
500
0.4
0.2
0
Channel 2 v Fuzzy En.
Channel 2 V b(t)
-0.5
-1
-1.5
-2
-2.5
1
0.95
0.9
0.85
Channel 2 V r^2(t)
6
5
4
3
2
EQ Magnitudes
Jan-03
Jan-28

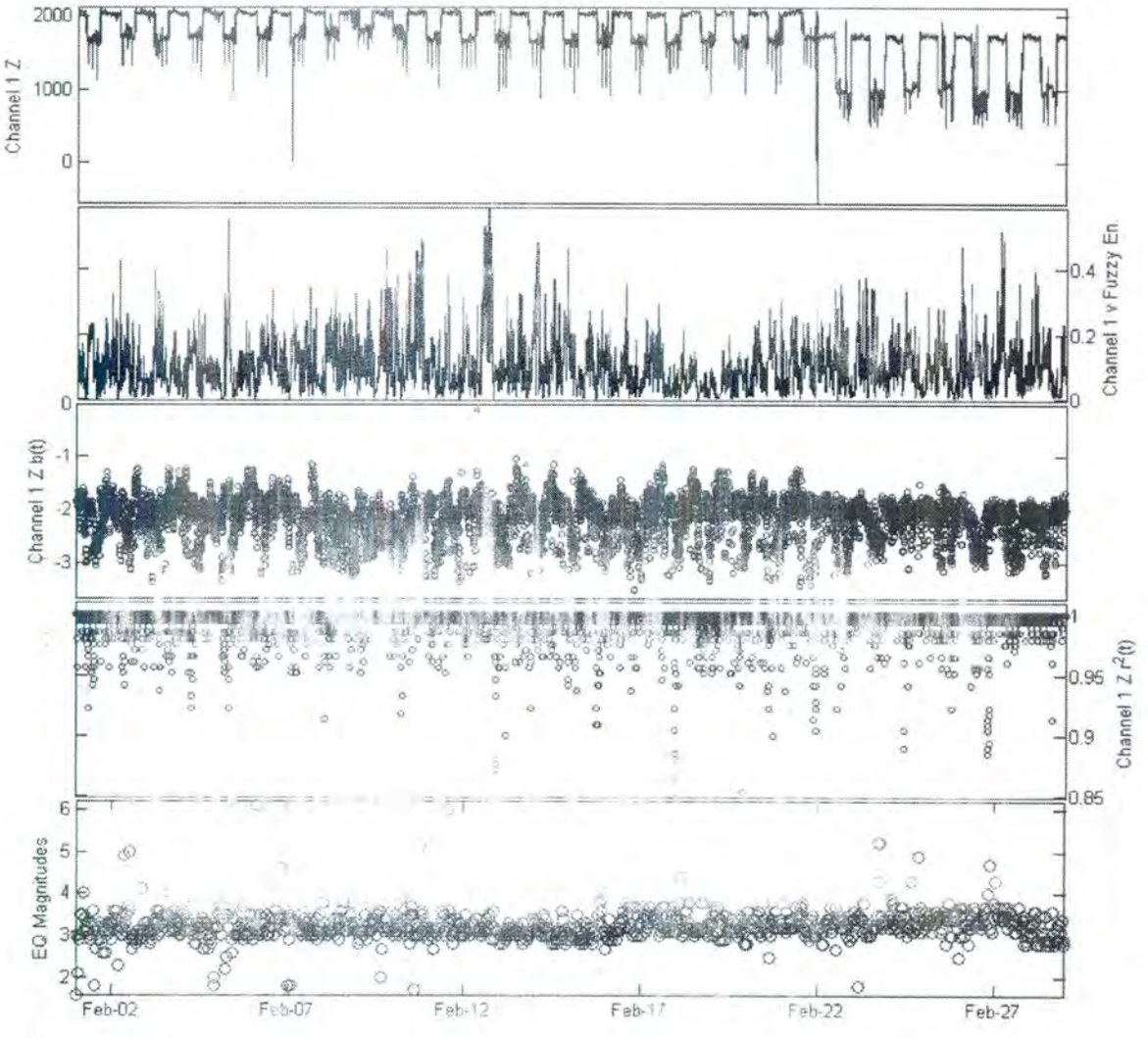

Channel 3 V
1000
800
600
400
200
Channel 3 v Fuzzy En.
1
0.8
0.6
0.4
0.2
0
Channel 3 V b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 3 V r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-28

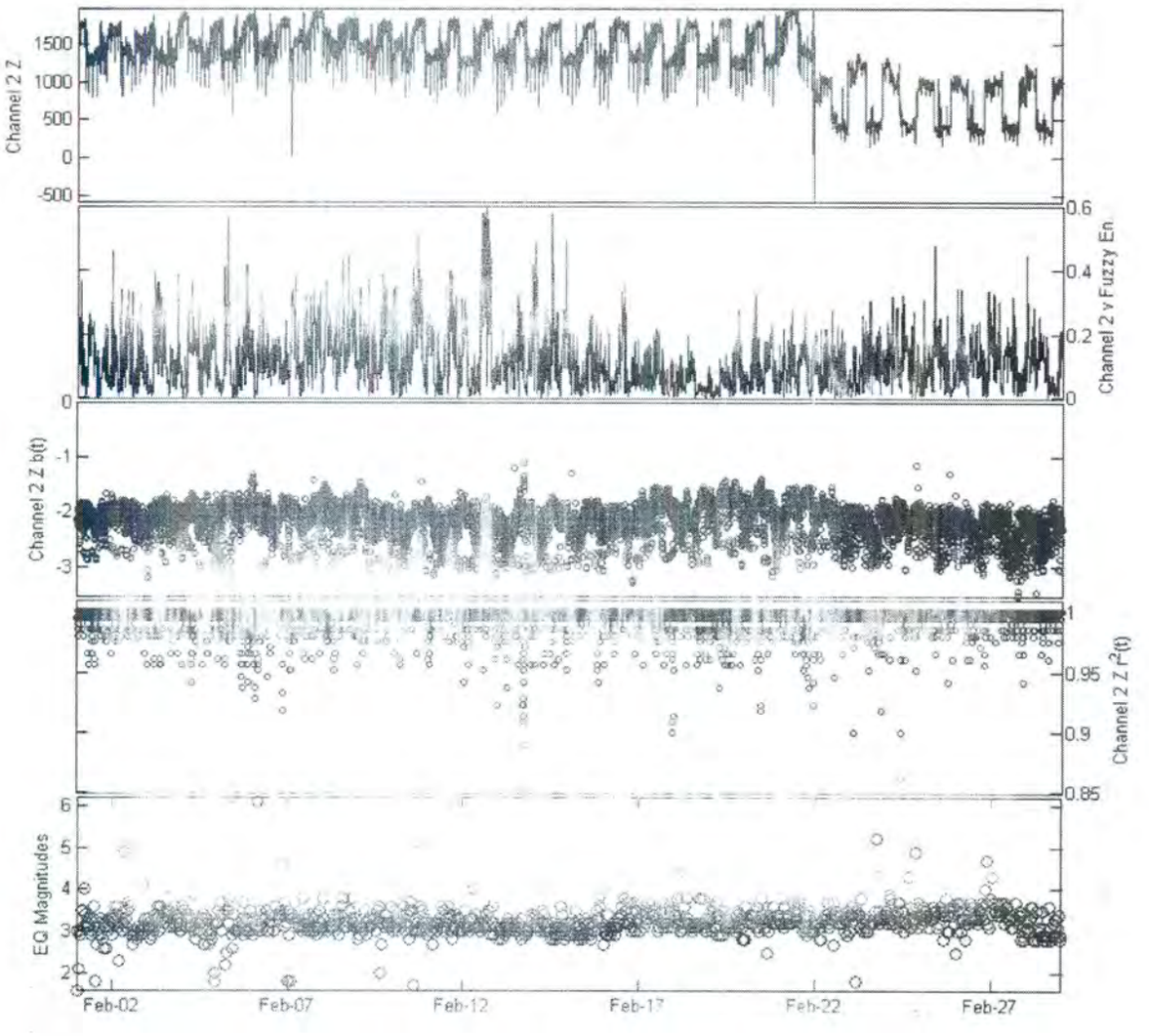

Channel 4 V
1500
1000
500
Channel 4 v Fuzzy En
1
0.8
0.6
0.4
0.2
0
Channel 4 V b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-3
Jan-28

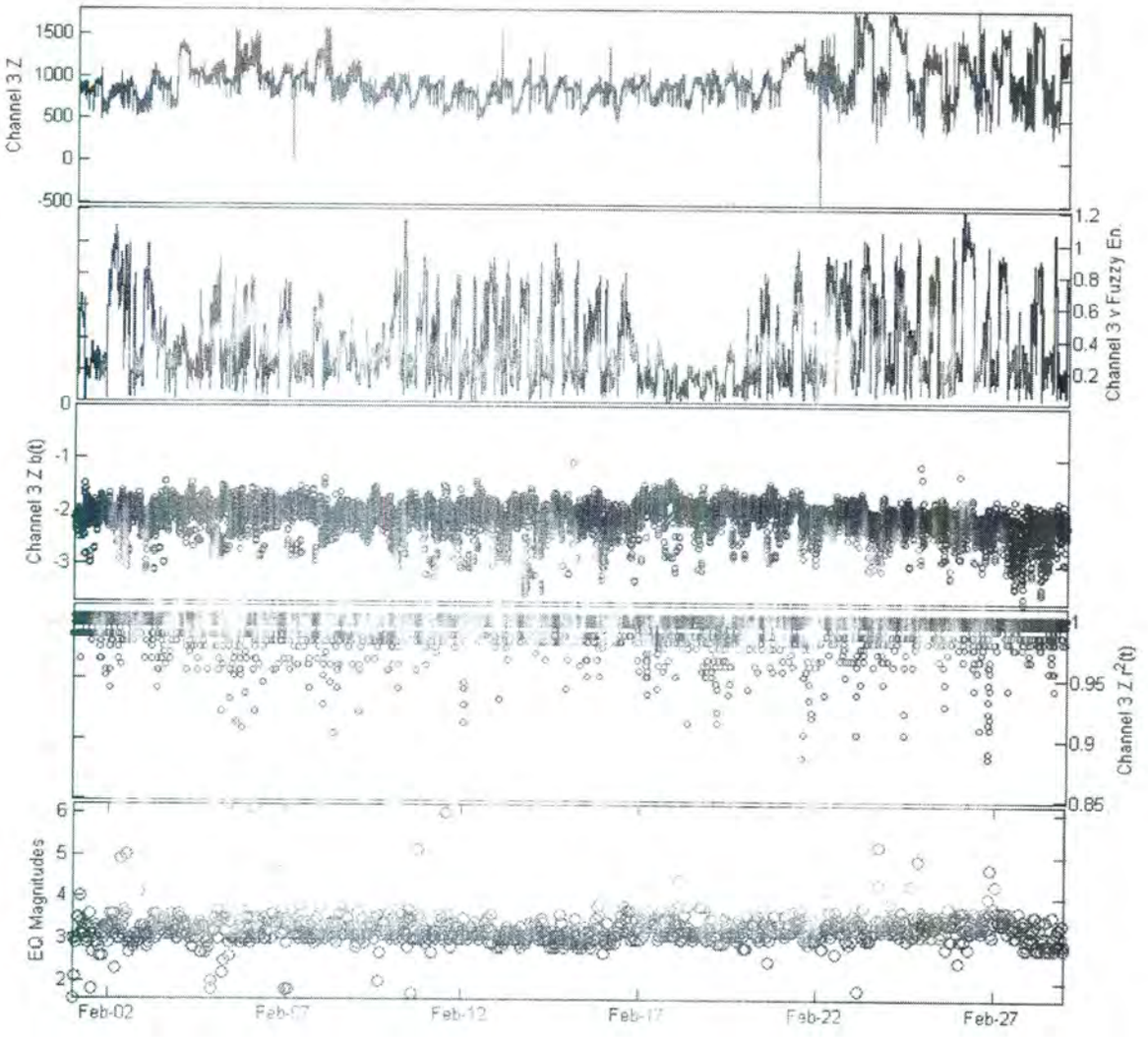

Channel 5 V
1000
800
600
400
Channel 5 v Fuzzy En
1.5
1
0.5
Channel 5 V b(t)
0
-0.5
-1
-1.5
-2
Channel 5 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-18
Jan-23
Jan-28

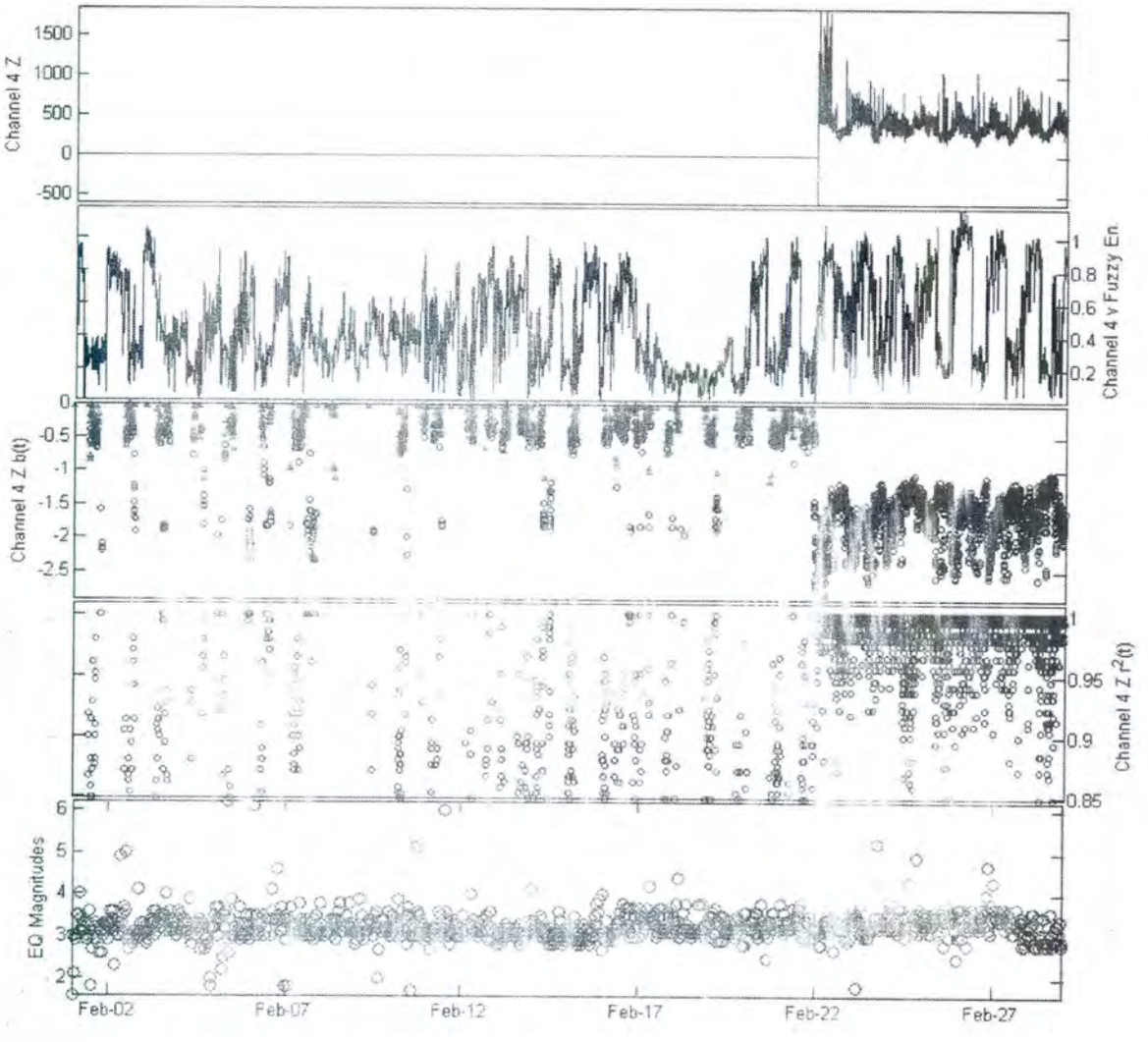

Channel 6 V
1000
800
600
400
Channel 6 v Fuzzy En.
1.2
1
0.8
0.6
0.4
0.2
Channel 6 V b(t)
0
-0.5
-1
-1.5
Channel 6 V r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Jan-03
Jan-18
Jan-23
Jan-28

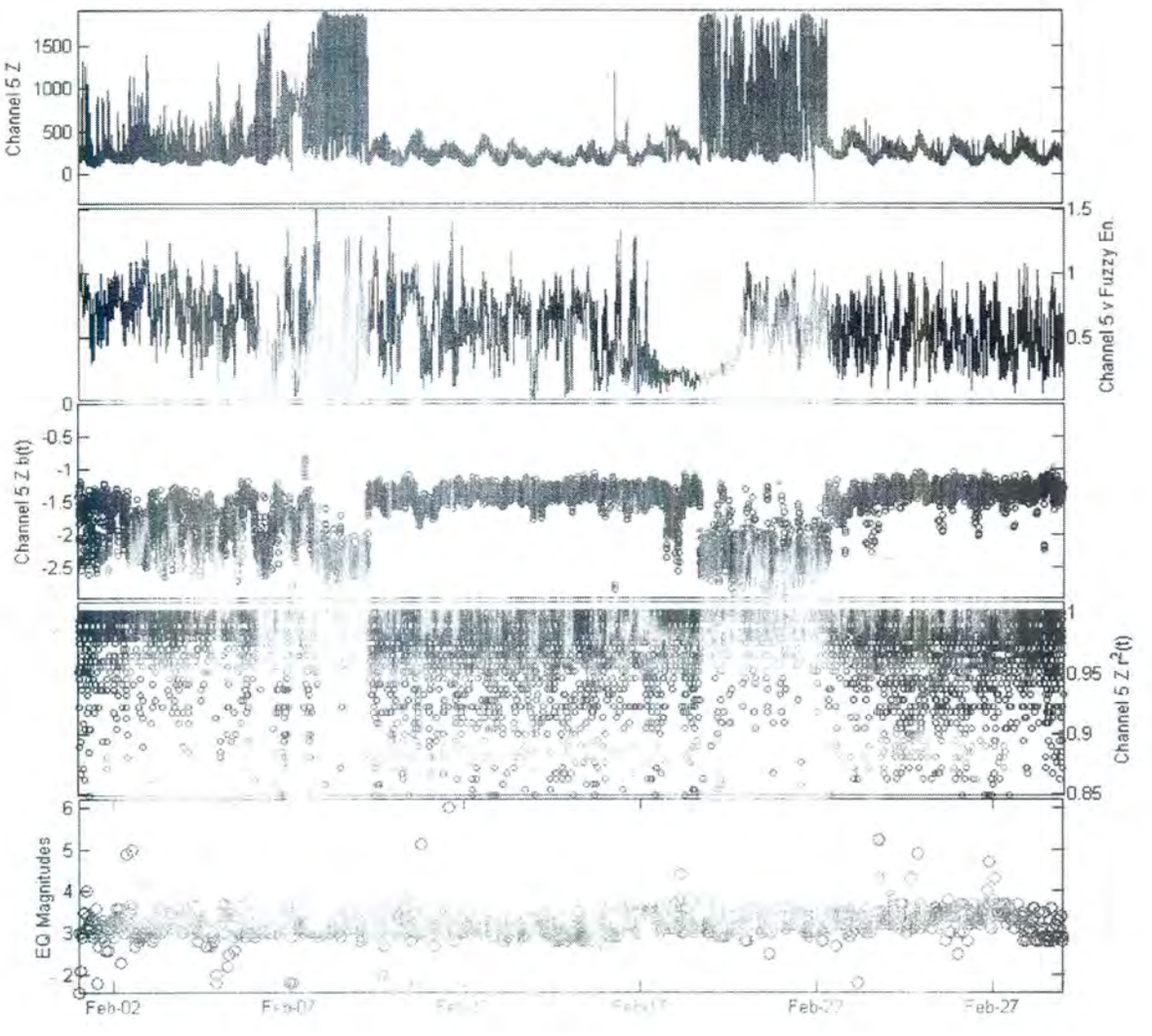

12.11
12.105
12.1
0.02
0.01
0
Channel 7 v Fuzzy En.
0
-0.2
-0.4
-0.6
-0.8
Channel 7 V b(t)
1
0.95
0.9
0.85
Channel 7 V r^2(t)
6
5
4
3
2
EQ Magnitudes
Jan-03
Jan-08
Jan-13
Jan-18
Jan-23
Jan-28

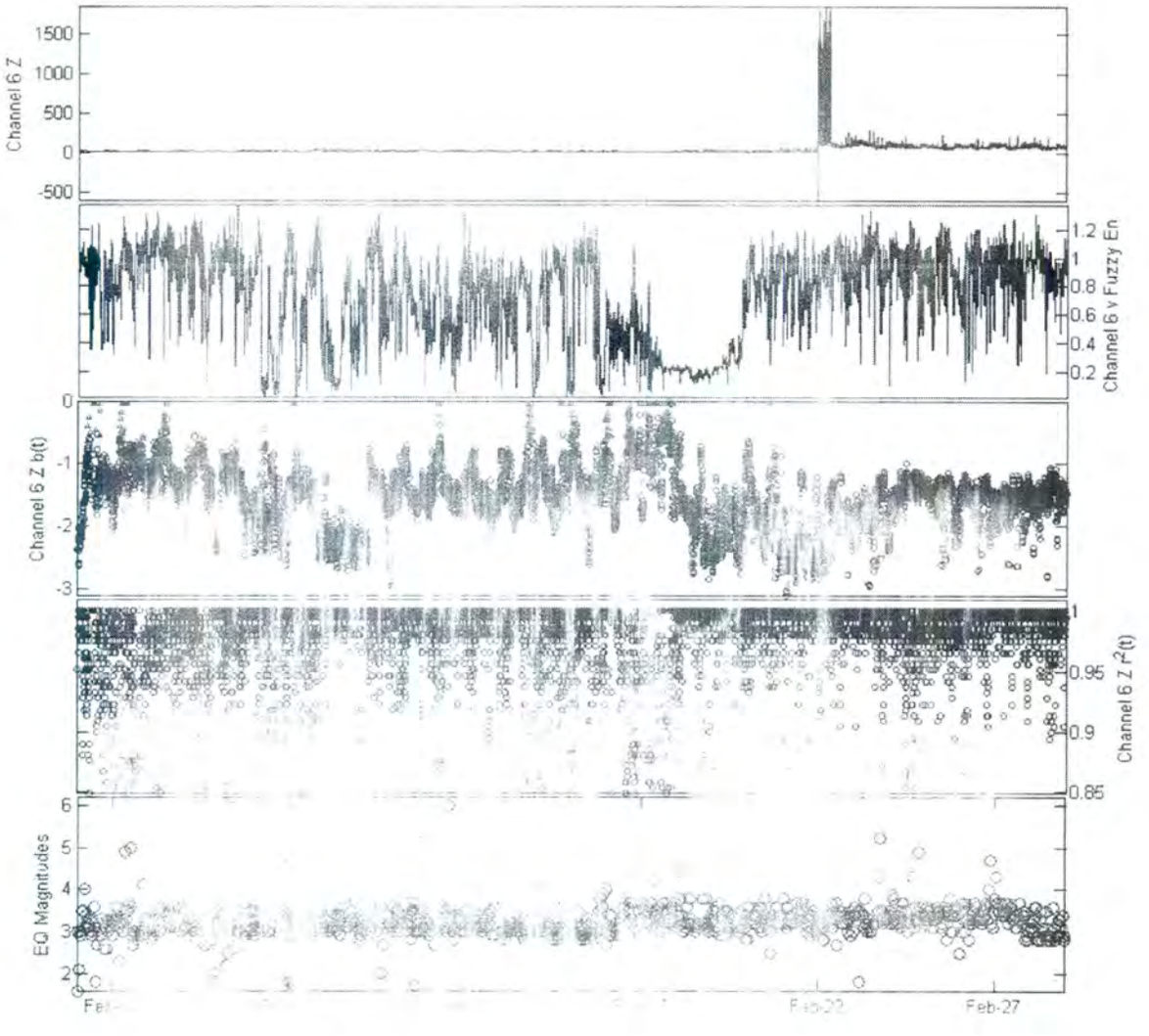

Channel 1 Z
2000
1000
0
Channel 1 γ Fuzzy En
0.4
0.2
0
Channel 1 Z b(t)
-1
-2
-3
Channel 1 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

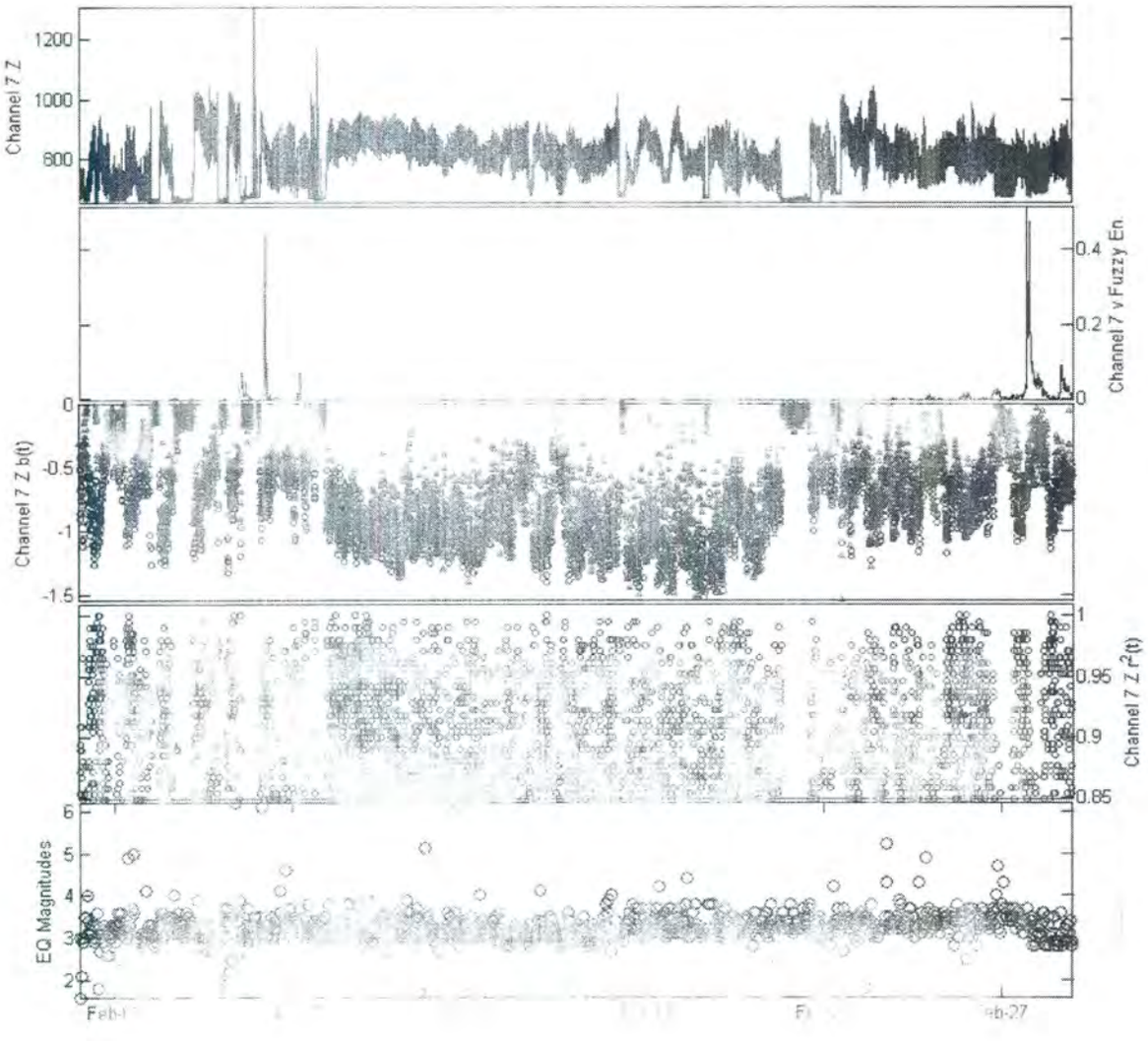

Channel 2 Z
1500
1000
500
0
-500
0.6
0.4
0.2
0
Channel 2 v Fuzzy En
Channel 2 Z b(t)
-1
-2
-3
1
0.95
0.9
0.85
Channel 2 Z r²(t)
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

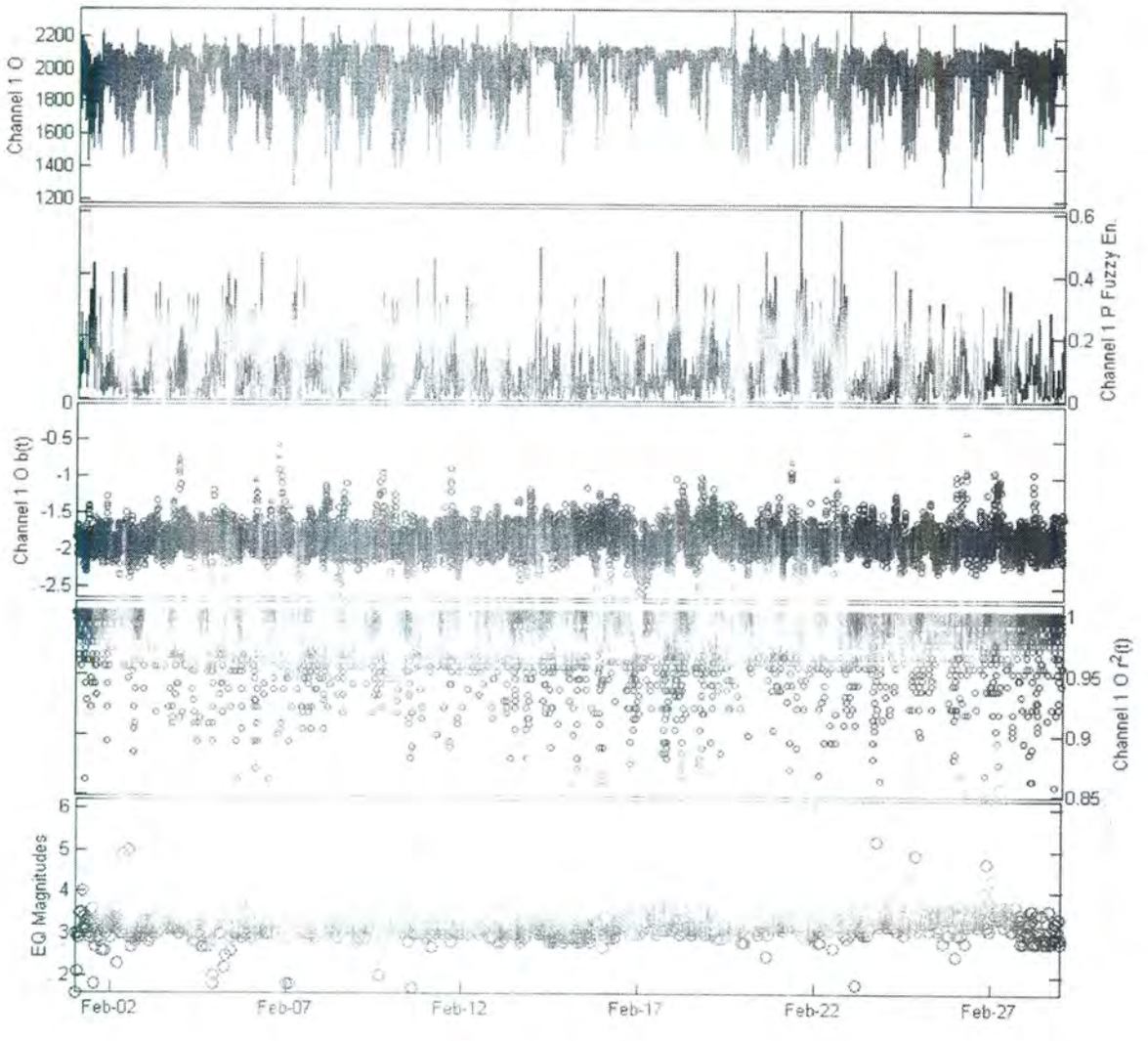

Channel 3 Z
1500
1000
500
0
-500
Channel 3 γ Fuzzy En.
1.2
1
0.8
0.6
0.4
0.2
0
Channel 3 Z b(t)
-1
-2
-3
Channel 3 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

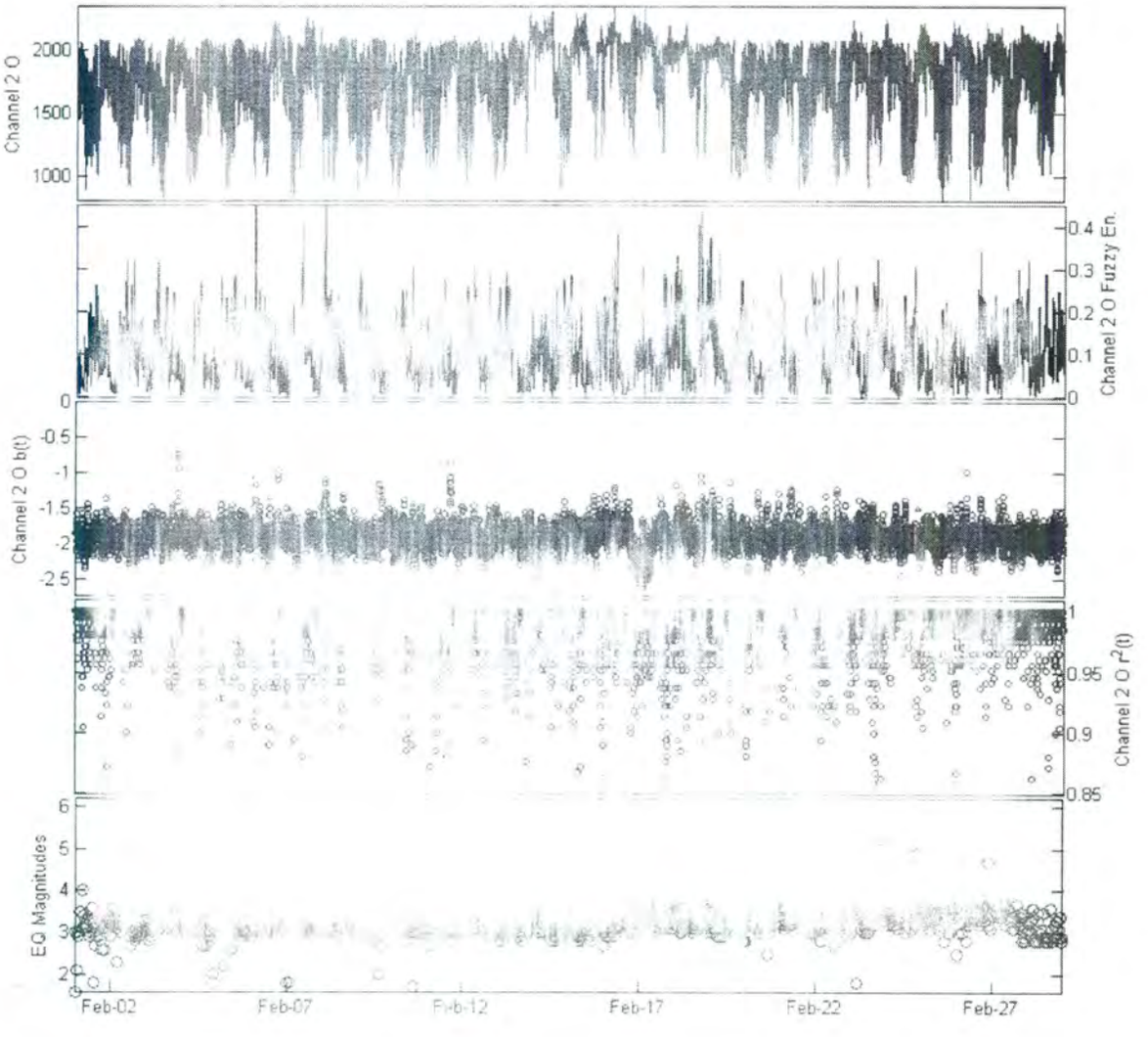

Channel 4 Z
1500
1000
500
0
-500
Channel 4 v Fuzzy En
1
0.8
0.6
0.4
0.2
0
Channel 4 Z b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

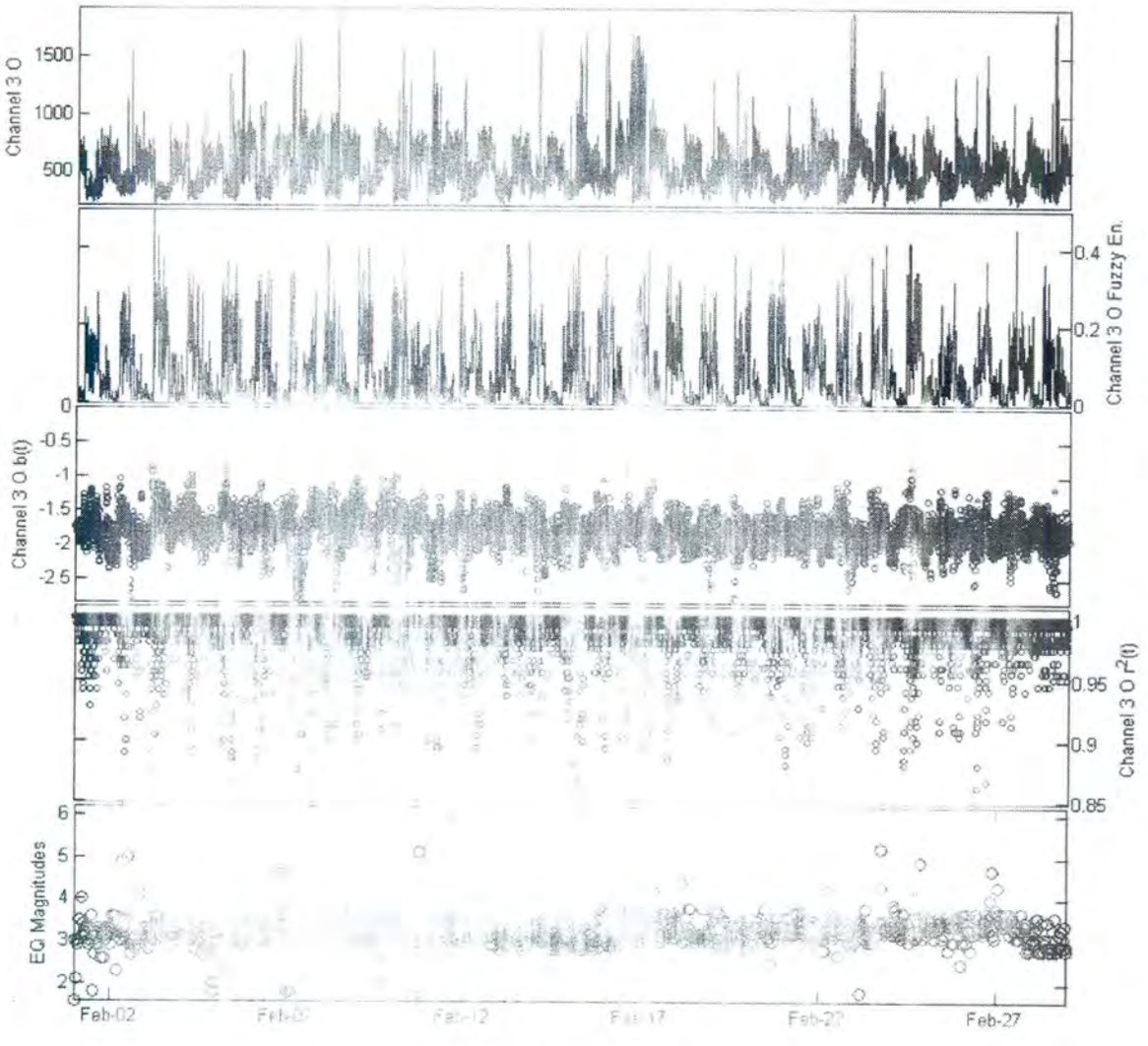

Channel 5 Z
1500
1000
500
0
Channel 5 v Fuzzy En.
1.5
1
0.5
0
Channel 5 Z b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 5 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-27

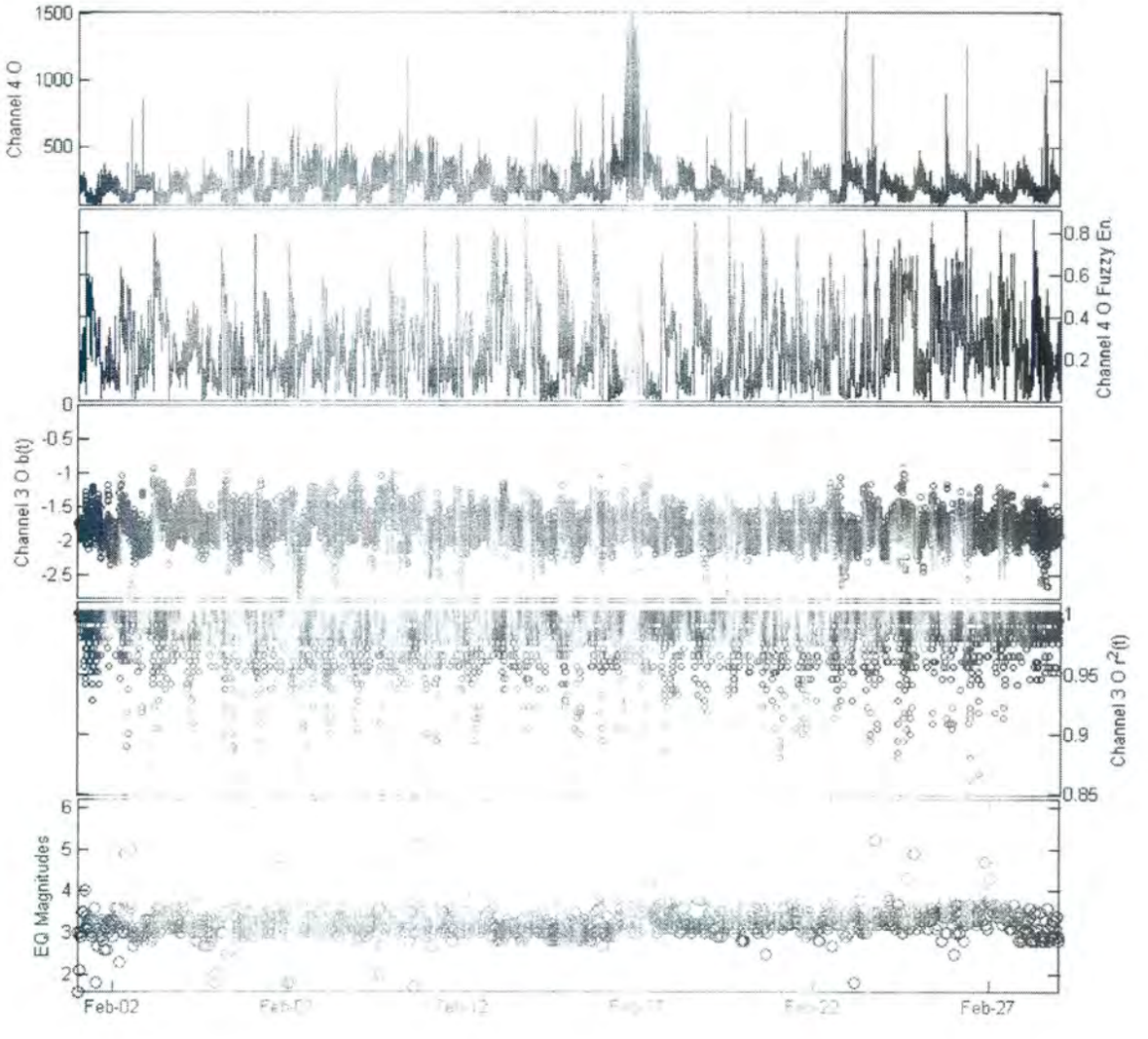

Channel 6 Z
1500
1000
500
0
-500
Channel 6 v Fuzzy En
1.2
1
0.8
0.6
0.4
0.2
Channel 6 Z b(t)
0
-1
-2
-3
Channel 6 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-22
Feb-27

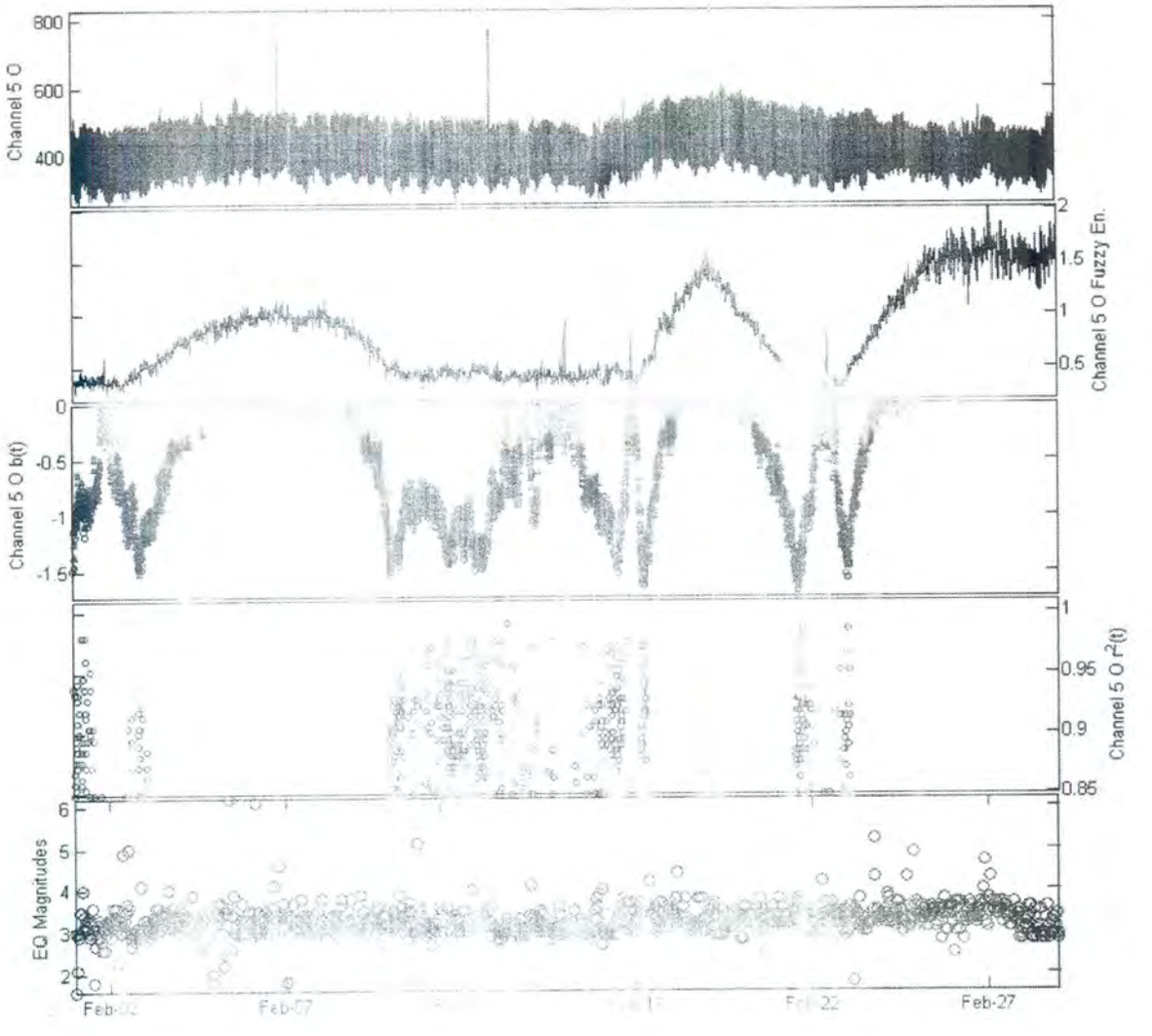

Channel 7 Z
1200
1000
800
0.4
0.2
0
Channel 7 γ Fuzzy En
0
-0.5
-1
-1.5
Channel 7 Z b(t)
1
0.95
0.9
0.85
Channel 7 Z r²(t)
6
5
4
3
2
EQ Magnitudes
Feb-1
Feb-27

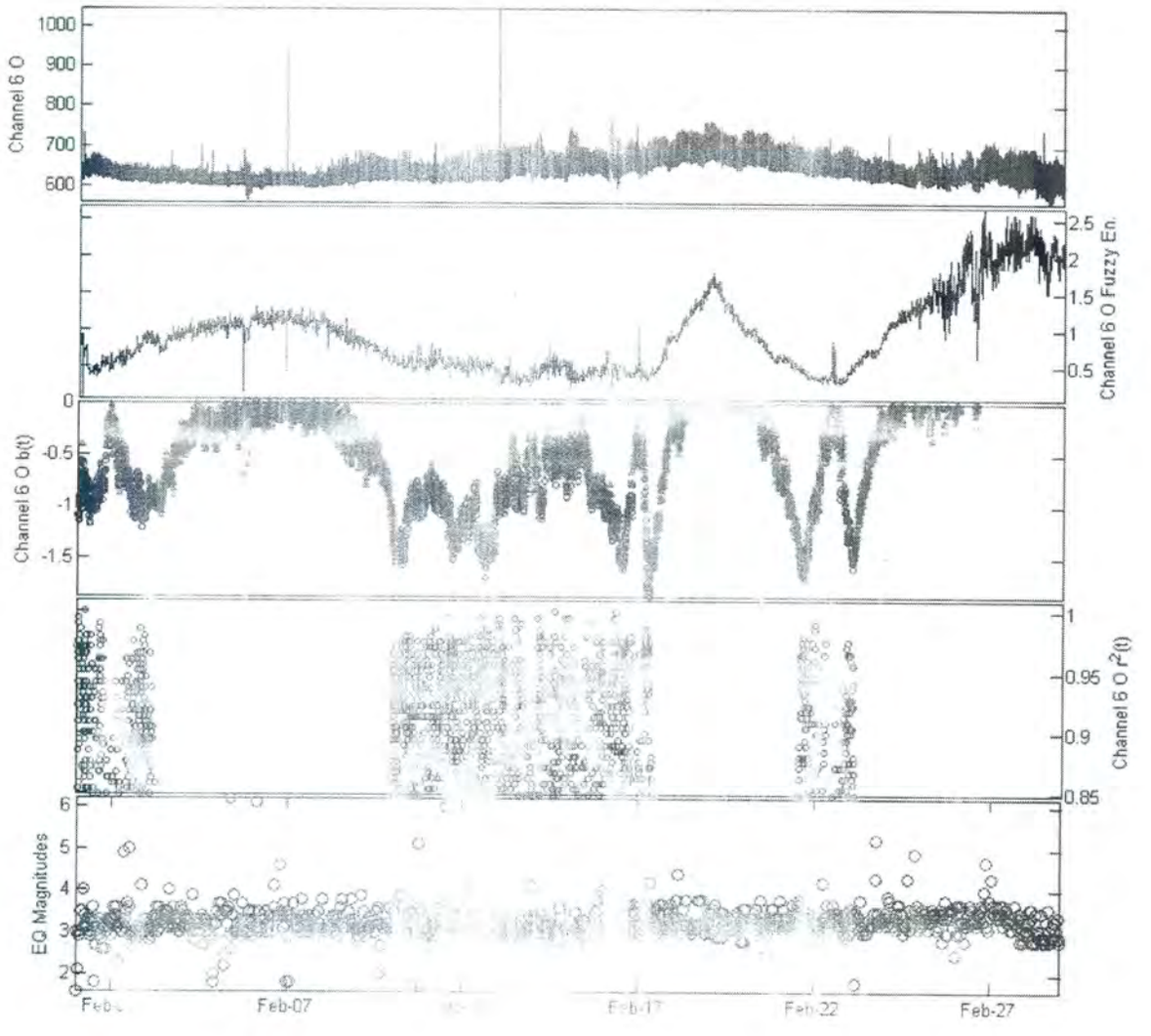

Channel 1 O
2200
2000
1800
1600
1400
1200
Channel 1 P Fuzzy En.
0.6
0.4
0.2
0
Channel 1 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 O r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

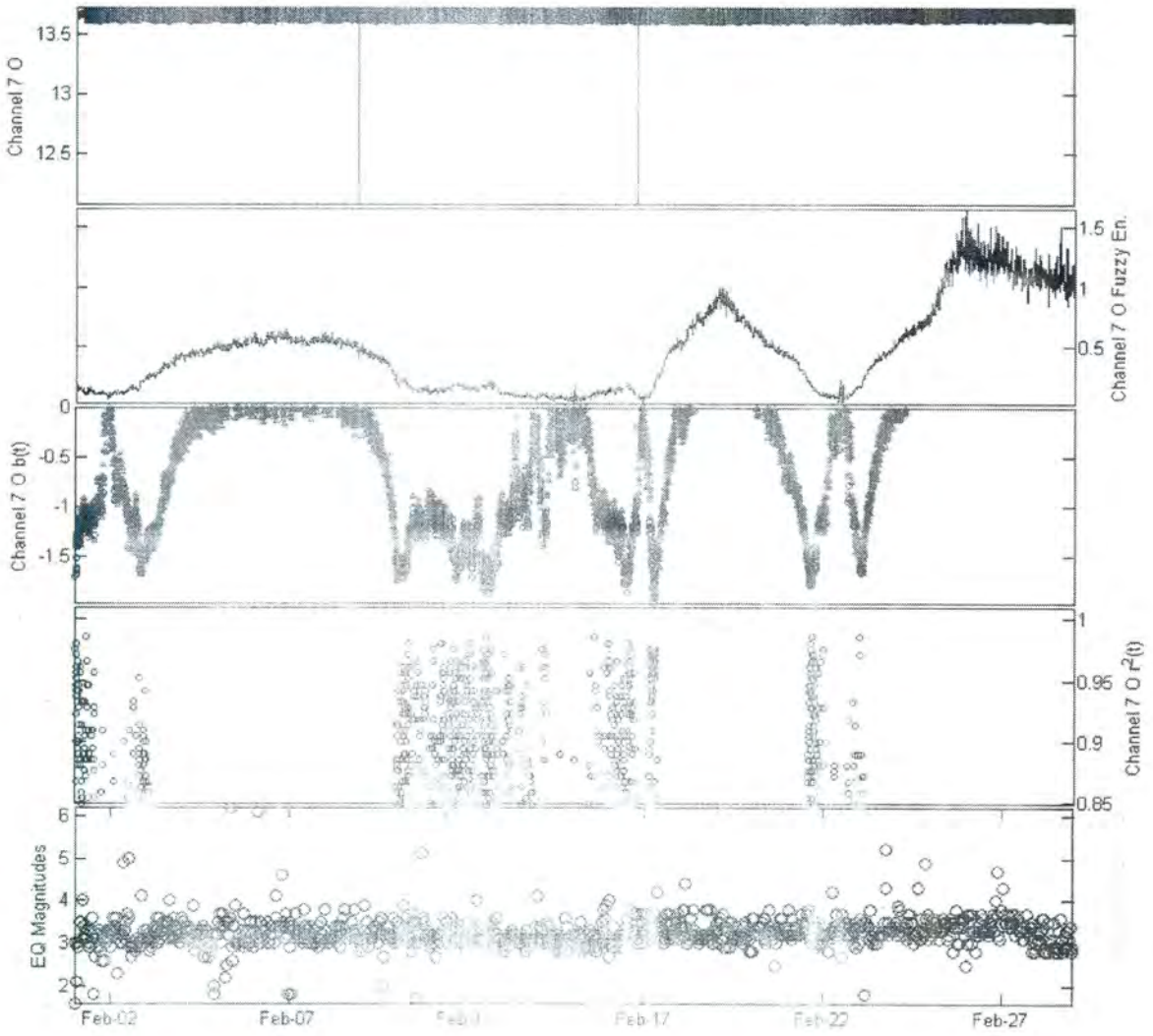

Channel 2 O
2000
1500
1000
0.4
0.3
0.2
0.1
0
Channel 2 O Fuzzy En.
Channel 2 O b(t)
-0.5
-1
-1.5
-2
-2.5
1
0.95
0.9
0.85
Channel 2 O r^2(t)
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

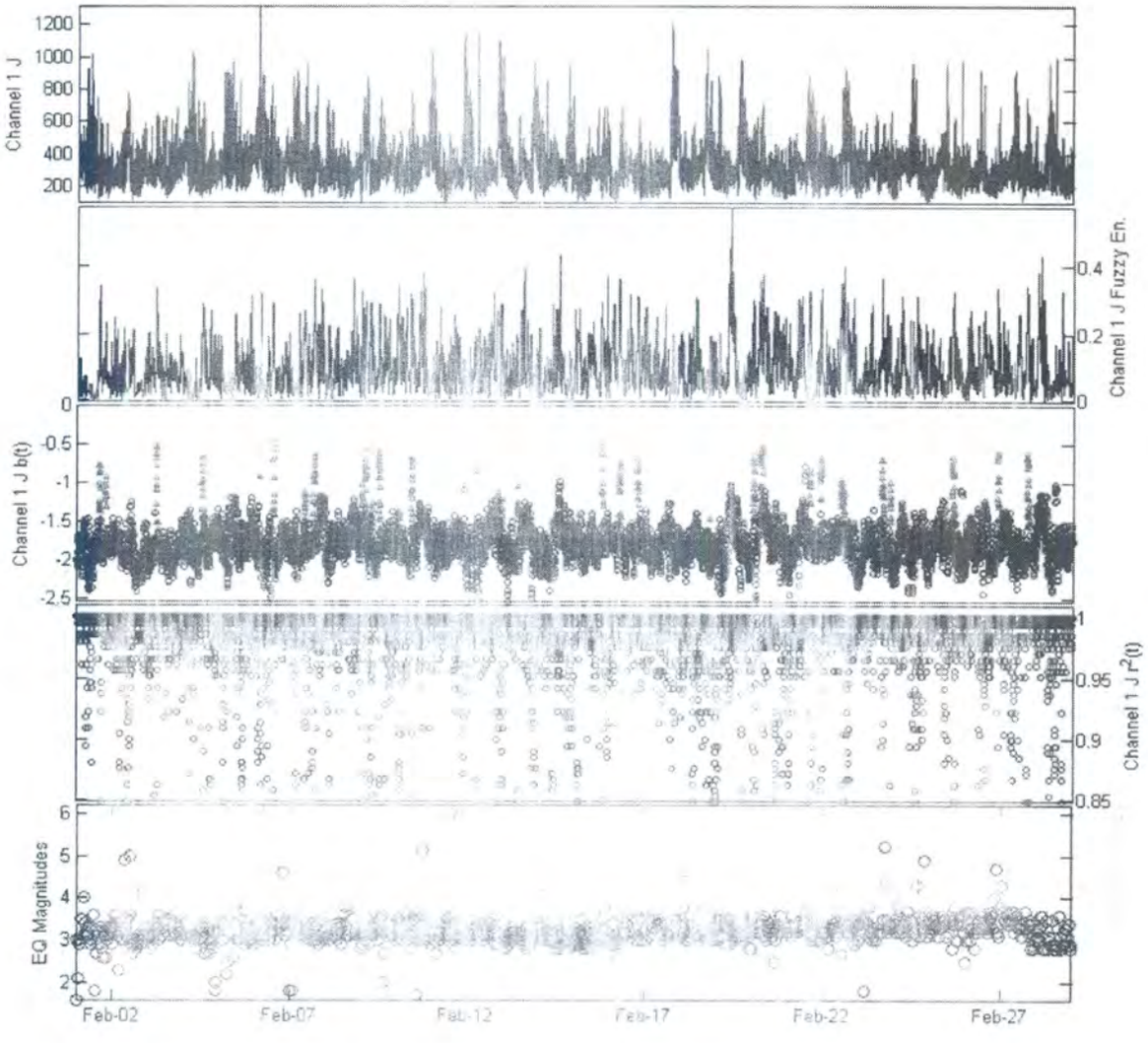

Channel 3 O
1500
1000
500
0.4
0.2
0
Channel 3 O Fuzzy En.
Channel 3 O b(t)
-0.5
-1
-1.5
-2
-2.5
1
0.95
0.9
0.85
Channel 3 O r²(t)
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

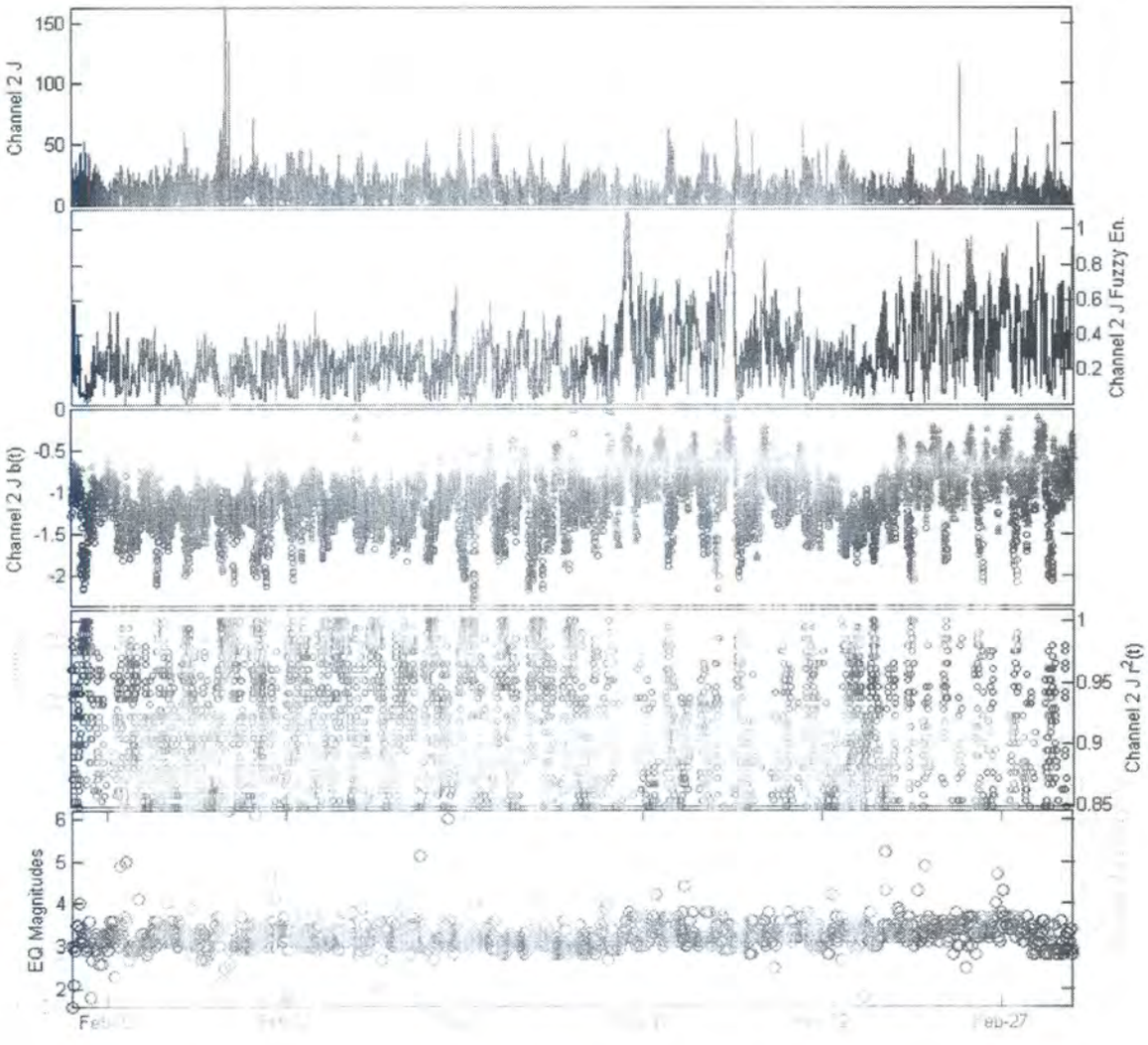

Channel 4 O
1500
1000
500
0.8
0.6
0.4
0.2
Channel 4 O Fuzzy En.
0
Channel 3 O b(t)
-0.5
-1
-1.5
-2
-2.5
1
0.95
0.9
0.85
Channel 3 O r²(t)
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-27

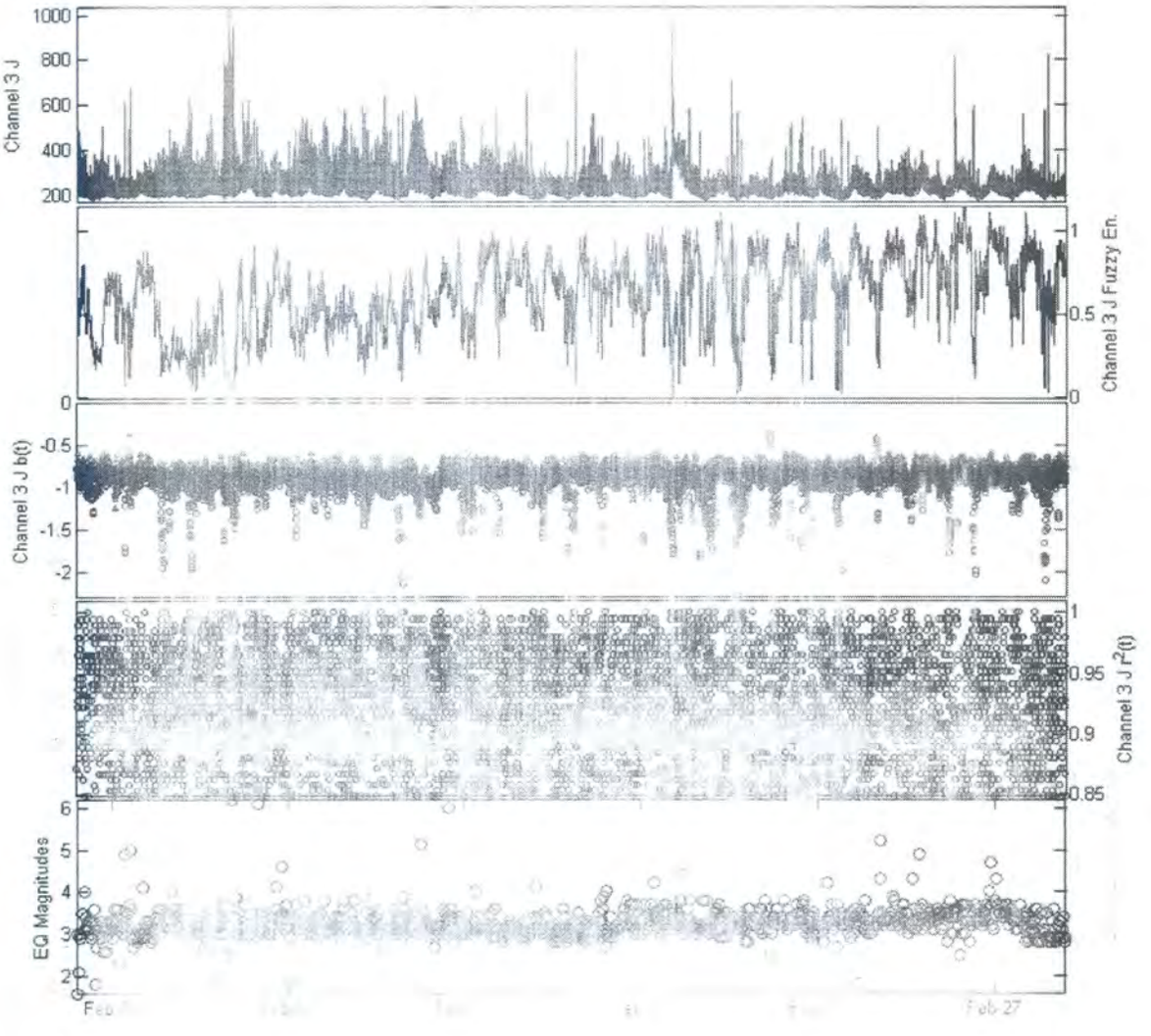

Channel 5 O
800
600
400
Channel 5 O Fuzzy En.
2
1.5
1
0.5
Channel 5 O b(t)
0
-0.5
-1
-1.5
Channel 5 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-22
Feb-27

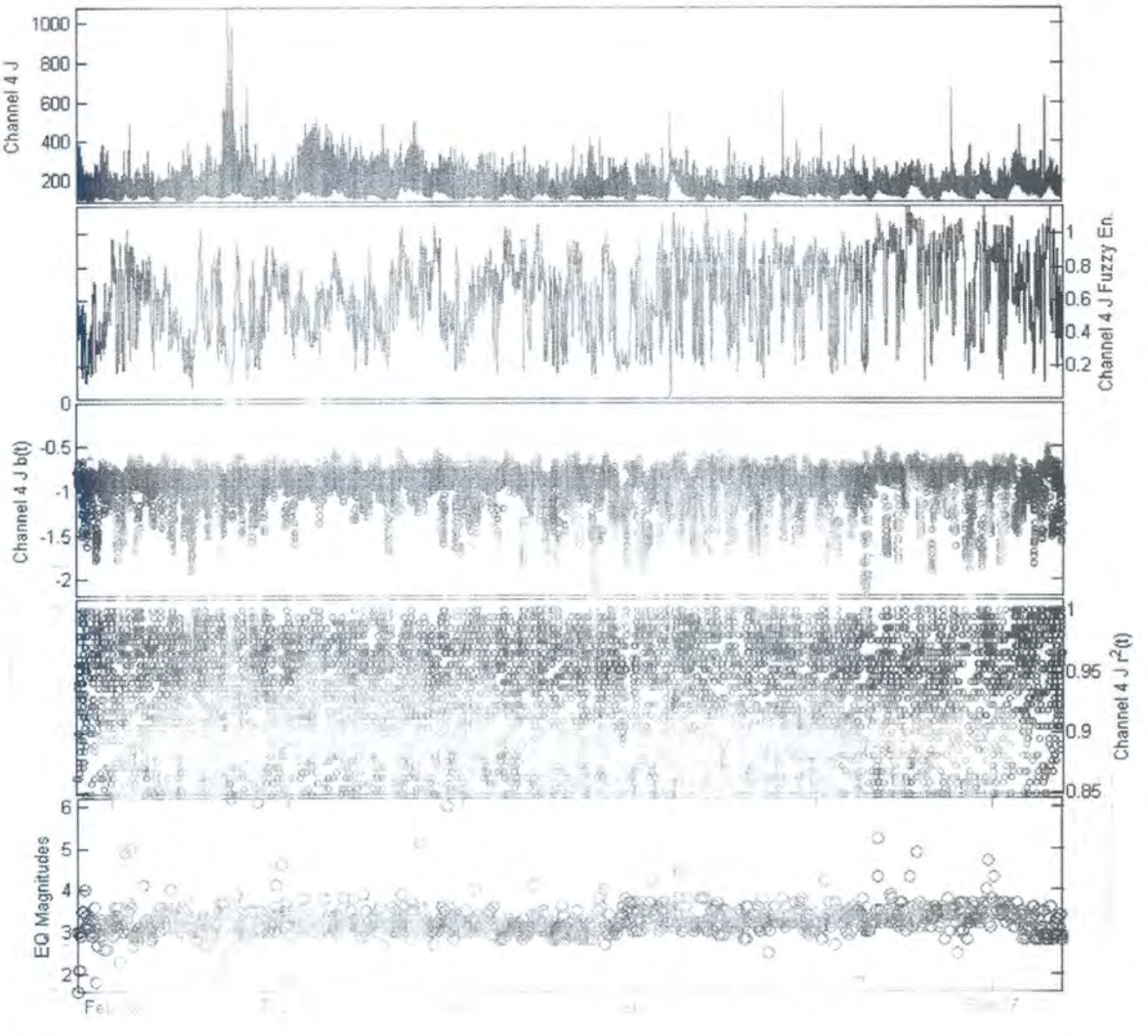


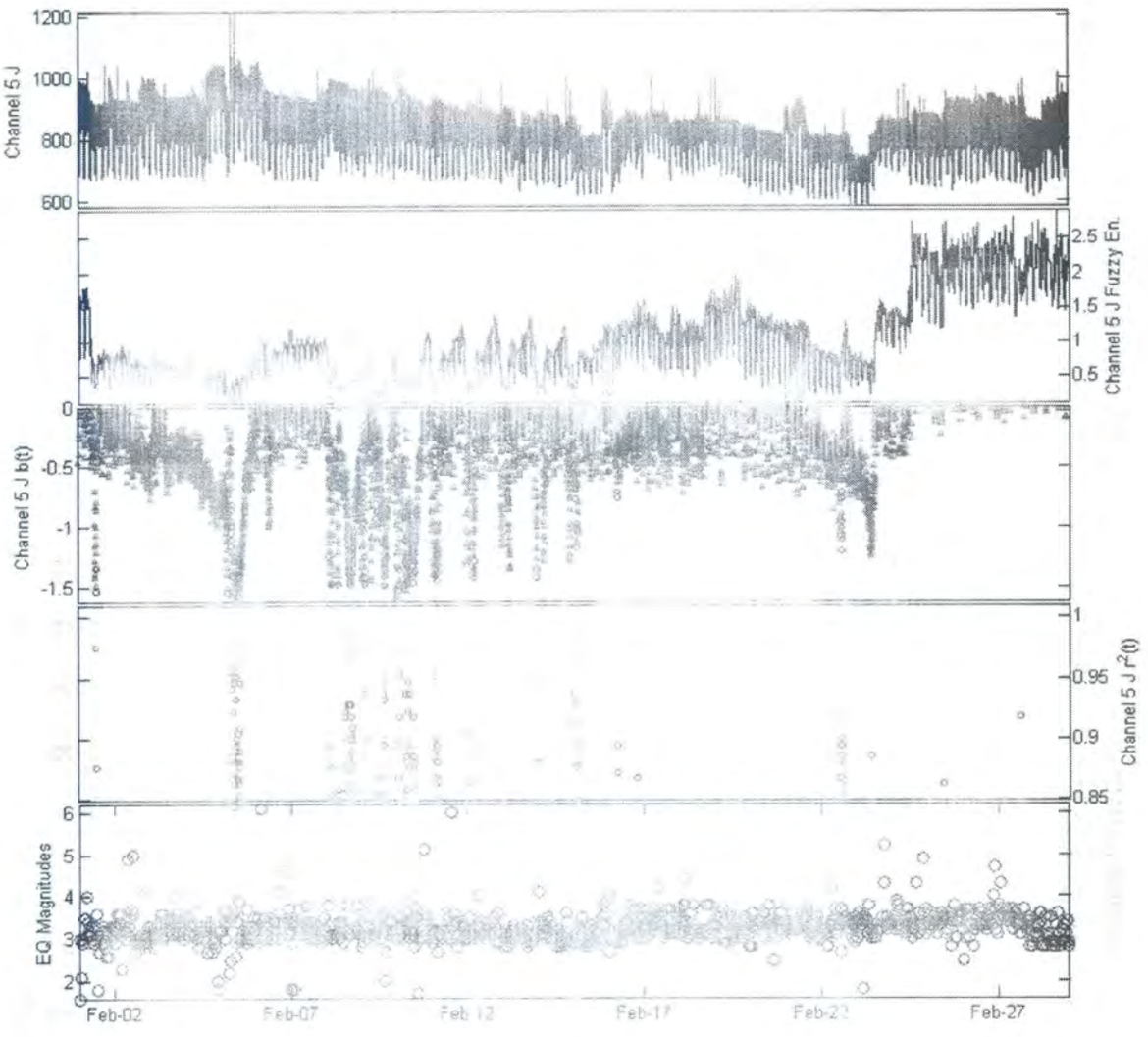

Channel 7 O
13.5
13
12.5
Channel 7 O Fuzzy En.
1.5
1
0.5
0
Channel 7 O b(t)
-0.5
-1
-1.5
Channel 7 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-17
Feb-22
Feb-27

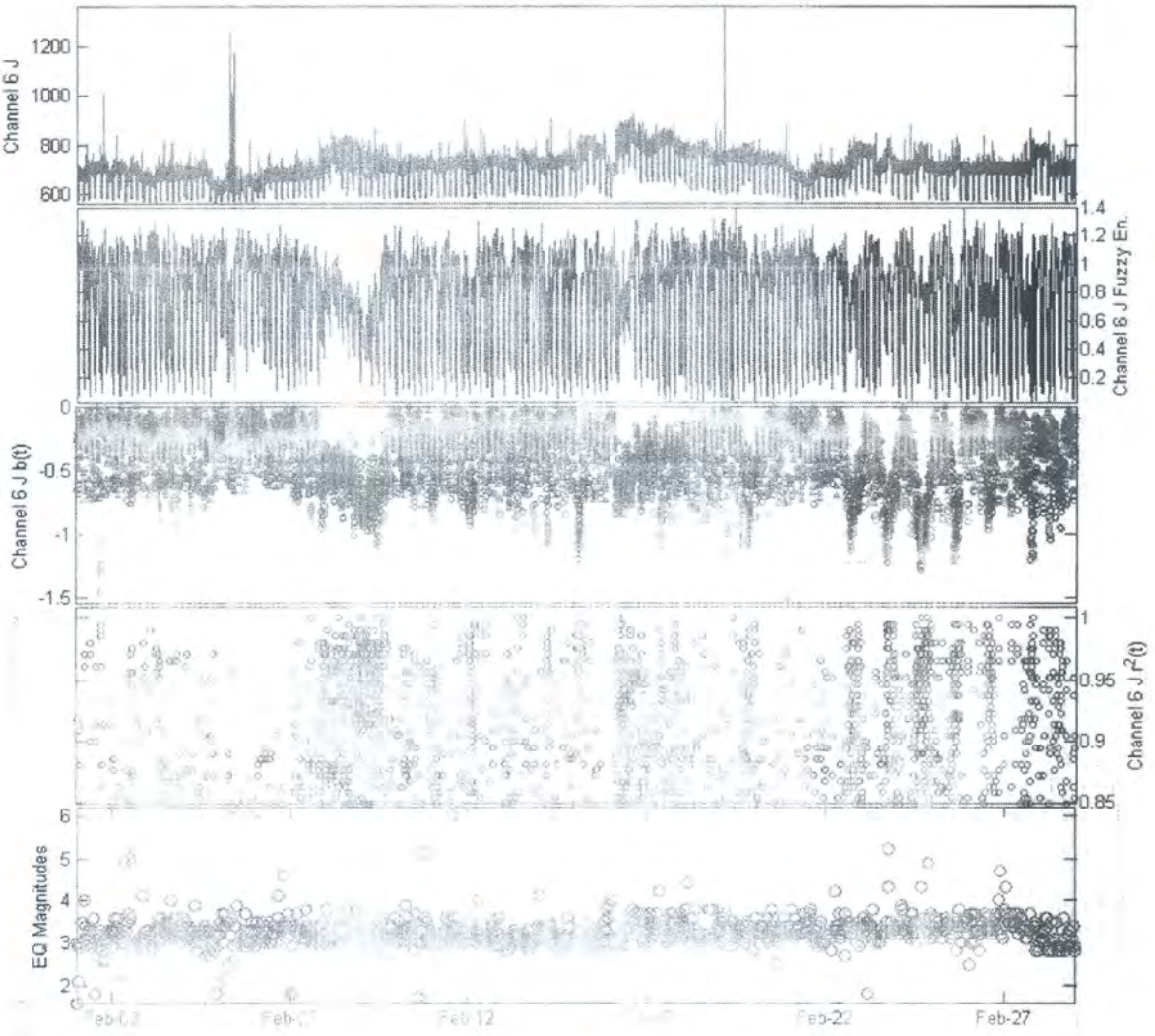

Channel 1 J
1200
1000
800
600
400
200
Channel 1 J Fuzzy En.
0.4
0.2
0
Channel 1 J b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

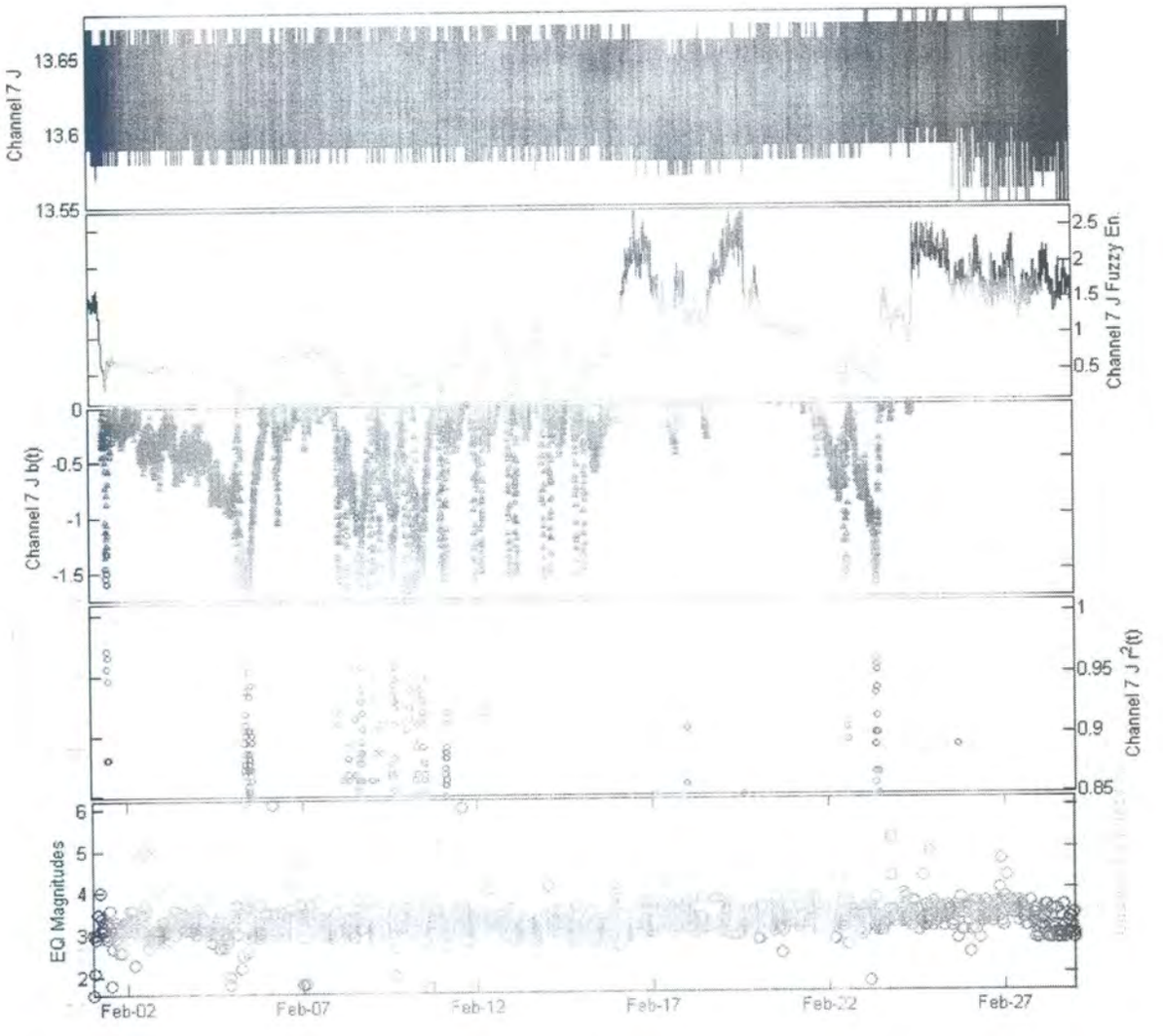

Channel 2 J
150
100
50
0
Channel 2 J Fuzzy En.
1
0.8
0.6
0.4
0.2
0
Channel 2 J b(t)
-0.5
-1
-1.5
-2
Channel 2 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-27

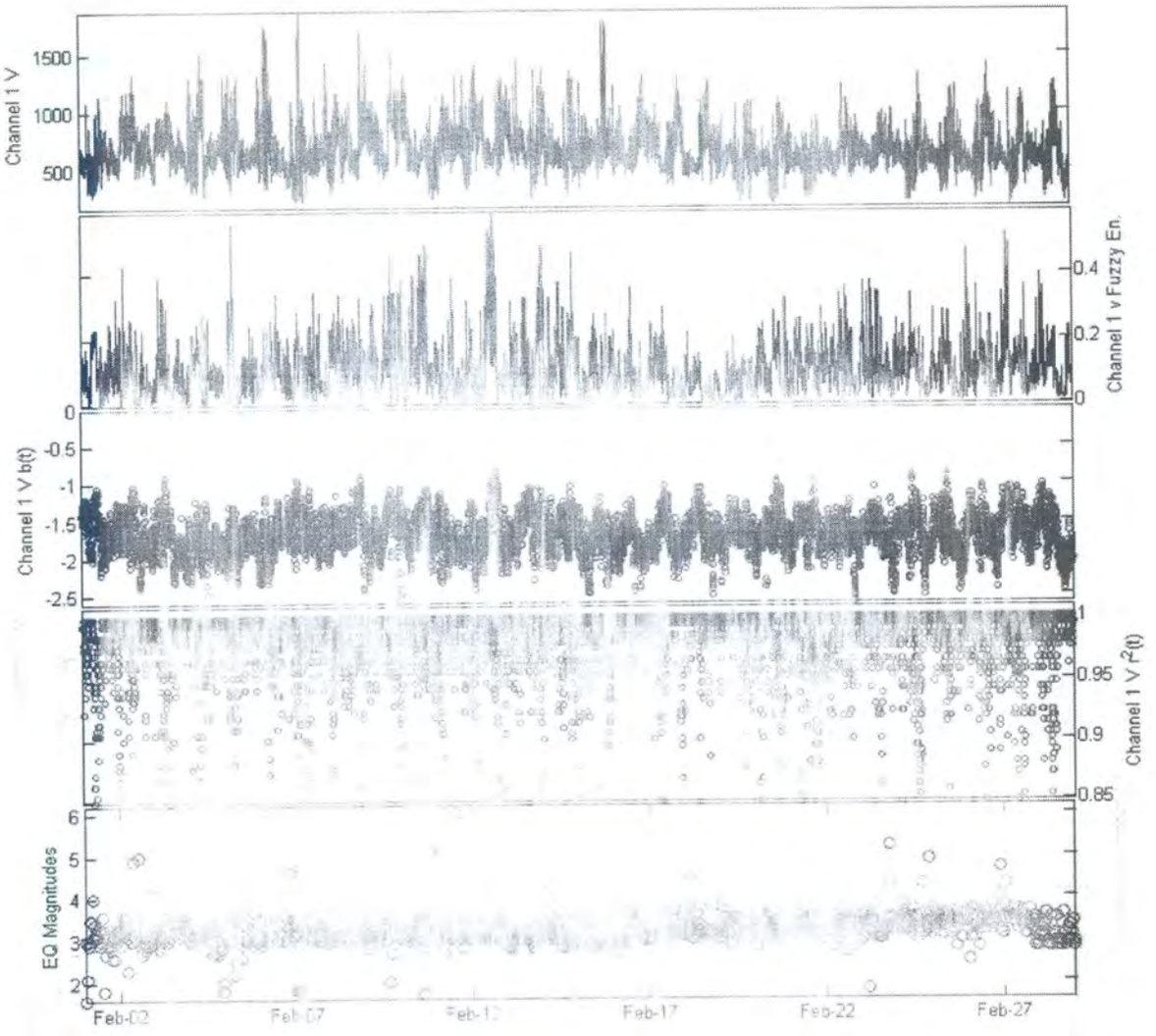

Channel 3 J
1000
800
600
400
200
Channel 3 J Fuzzy En.
1
0.5
0
Channel 3 J b(t)
-0.5
-1
-1.5
-2
Channel 3 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb
Feb 27

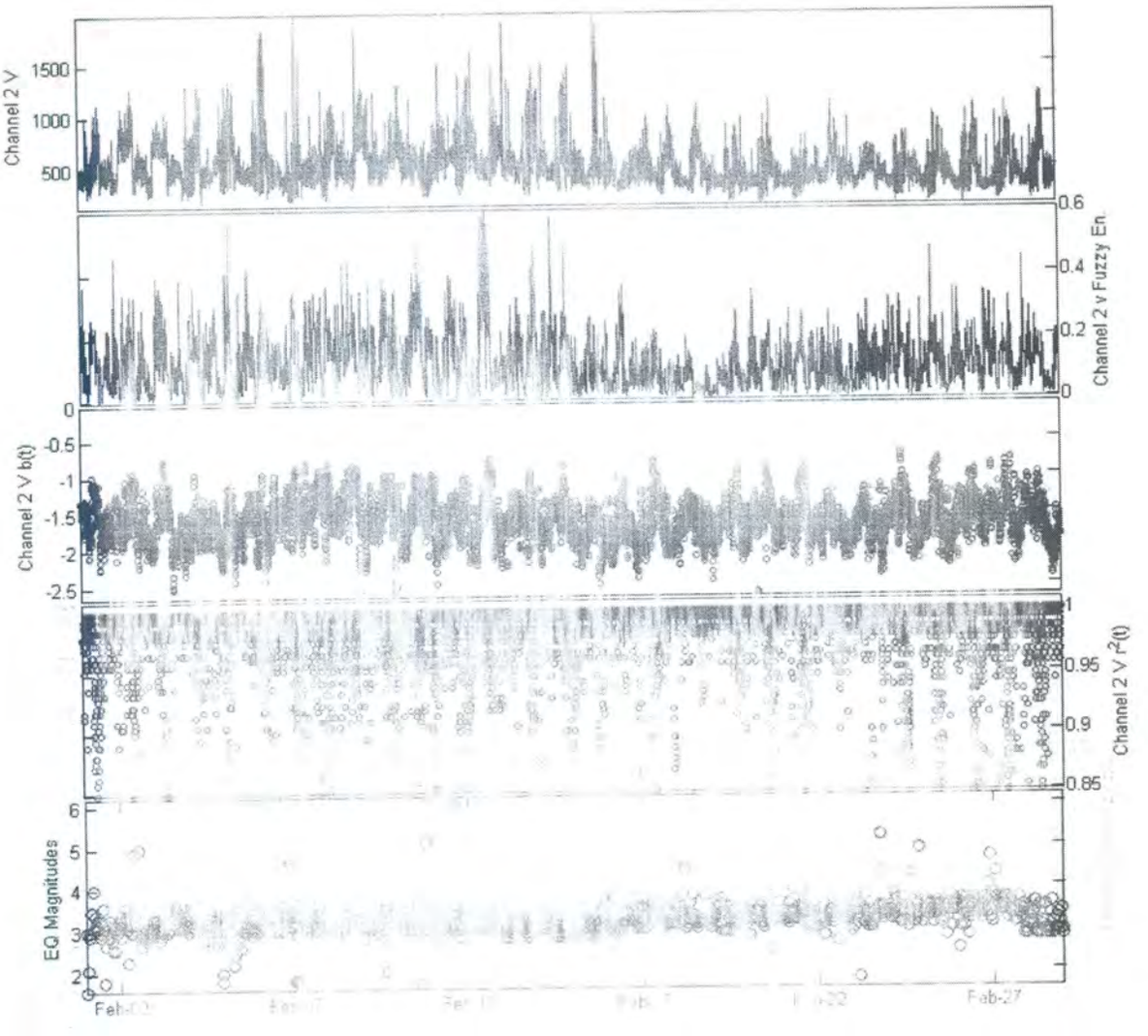

Channel 4 J
1000
800
600
400
200
Channel 4 J Fuzzy En.
1
0.8
0.6
0.4
0.2
0
Channel 4 J b(t)
-0.5
-1
-1.5
-2
Channel 4 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2

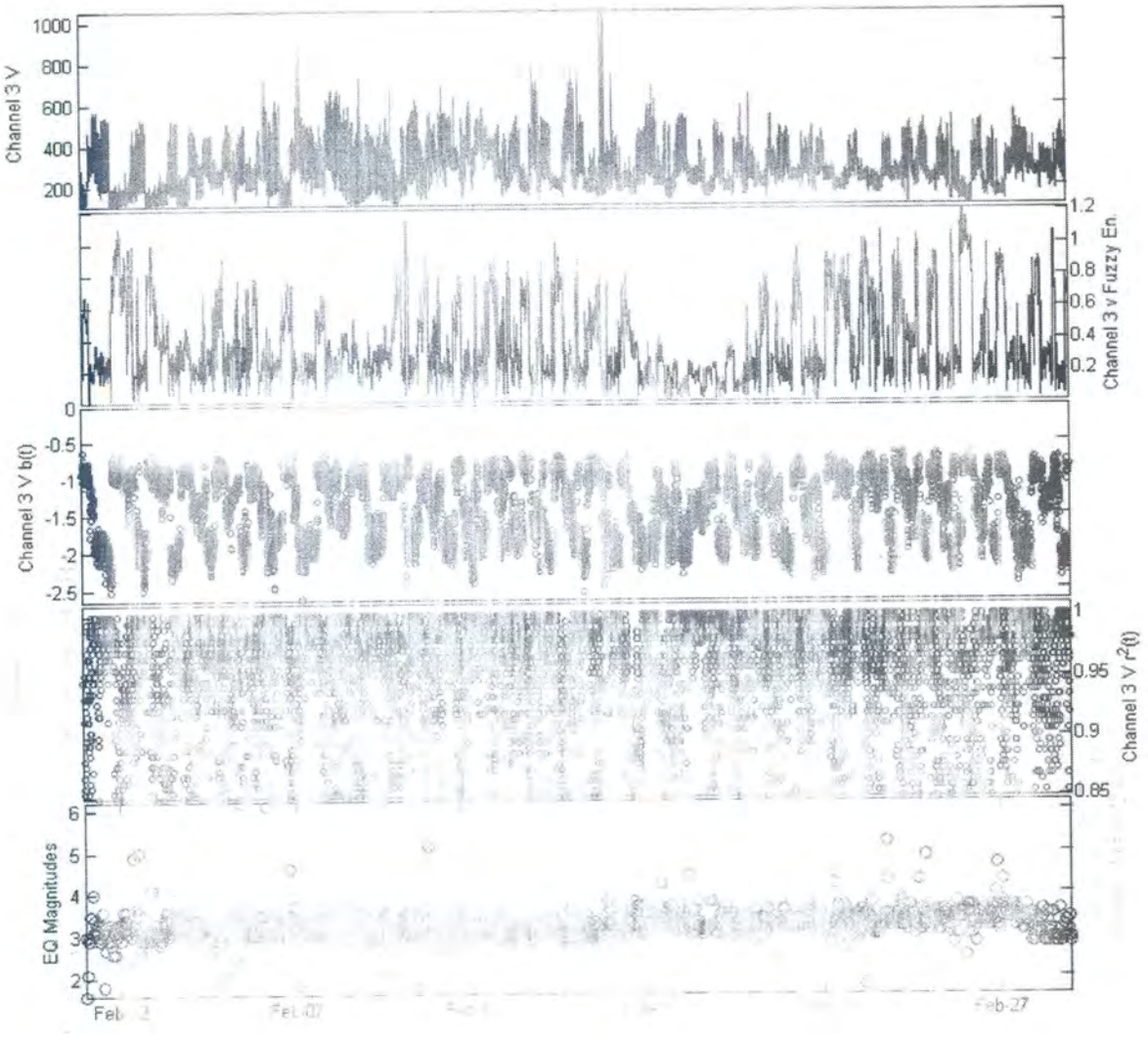

Channel 5 J
1200
1000
800
600
Channel 5 J Fuzzy En.
2.5
2
1.5
1
0.5
Channel 5 J b(t)
0
-0.5
-1
-1.5
Channel 5 J F^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

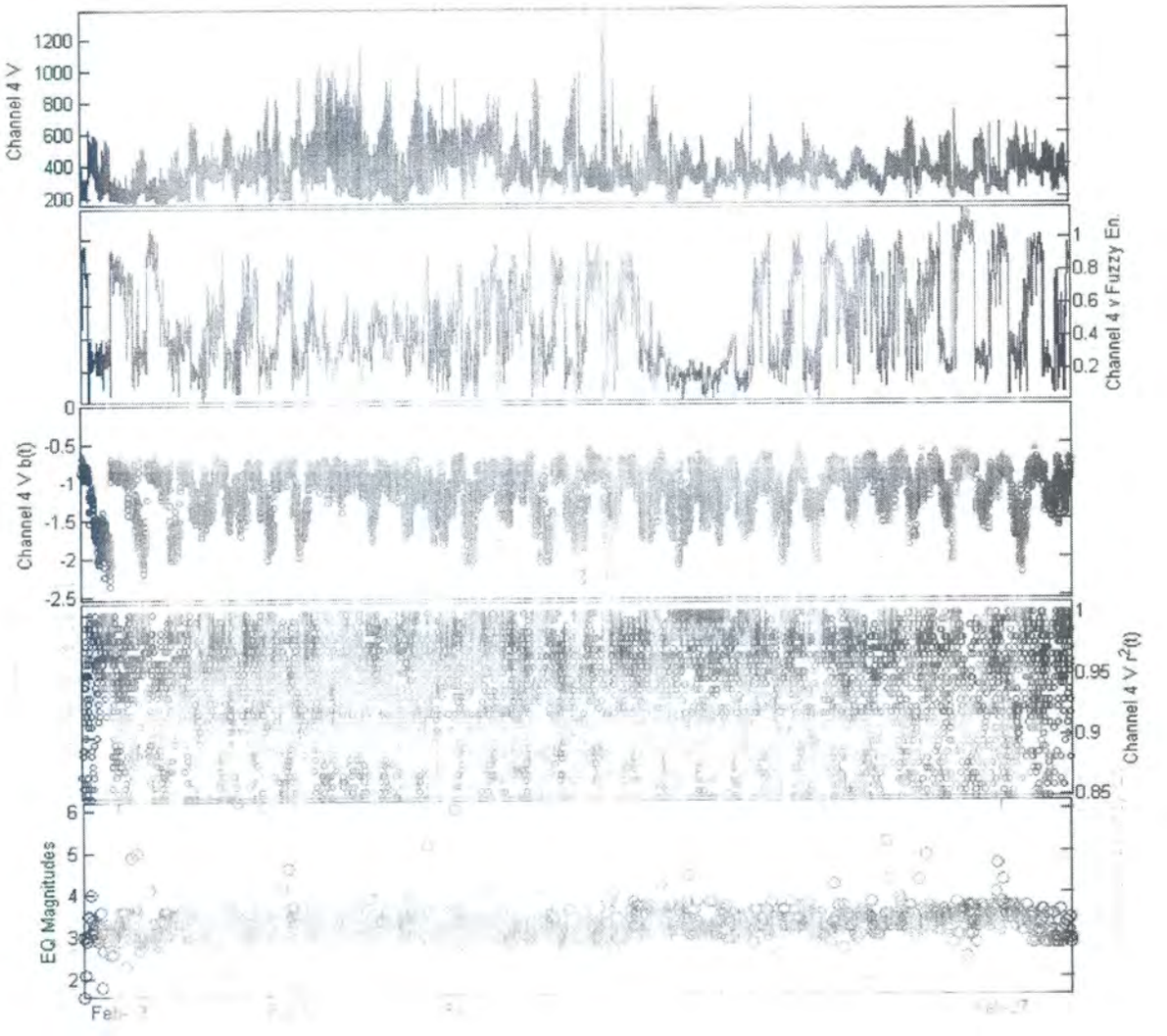

Channel 6 J
1200
1000
800
600
Channel 6 J Fuzzy En.
1.4
1.2
1
0.8
0.6
0.4
0.2
Channel 6 J b(t)
0
-0.5
-1
-1.5
Channel 6 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-22
Feb-27

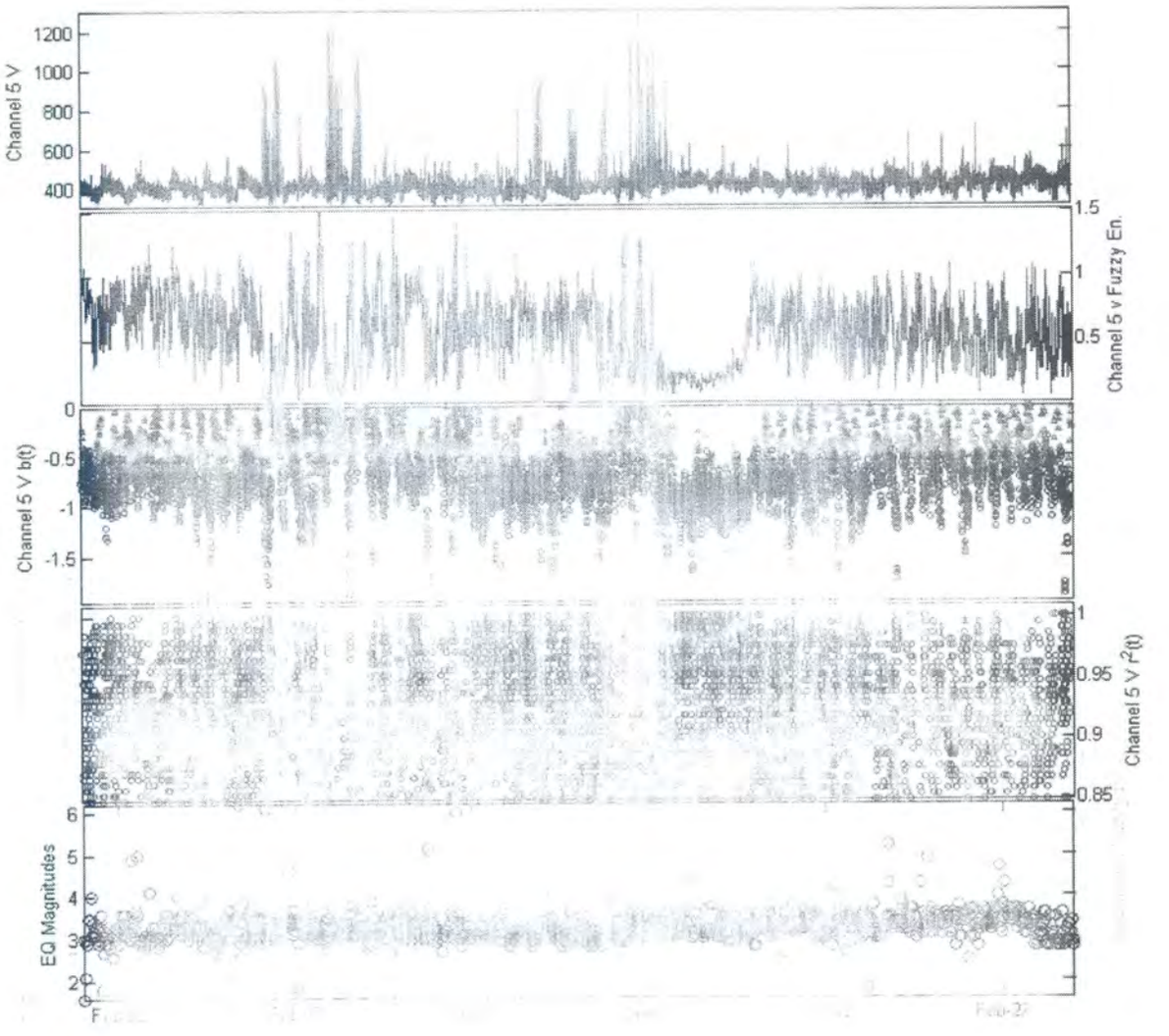

Channel 7 J
13.65
13.6
13.55
Channel 7 J Fuzzy En.
2.5
2
1.5
1
0.5
Channel 7 J b(t)
0
-0.5
-1
-1.5
Channel 7 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

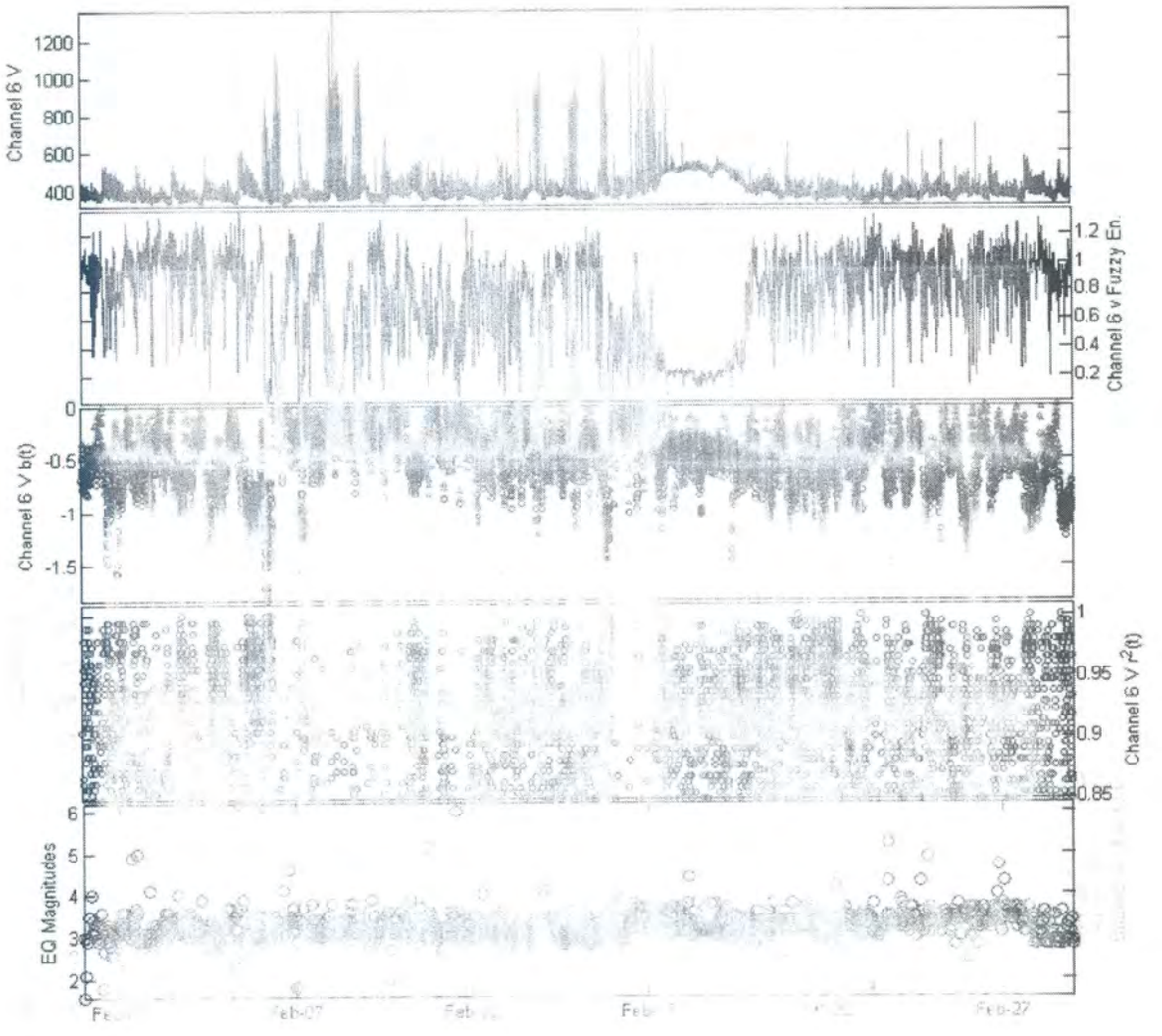

Channel 1 V
1500
1000
500
Channel 1 v Fuzzy En.
0.4
0.2
0
Channel 1 V b(t)
0
-0.5
-1
-1.5
-2
-2.5
Channel 1 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

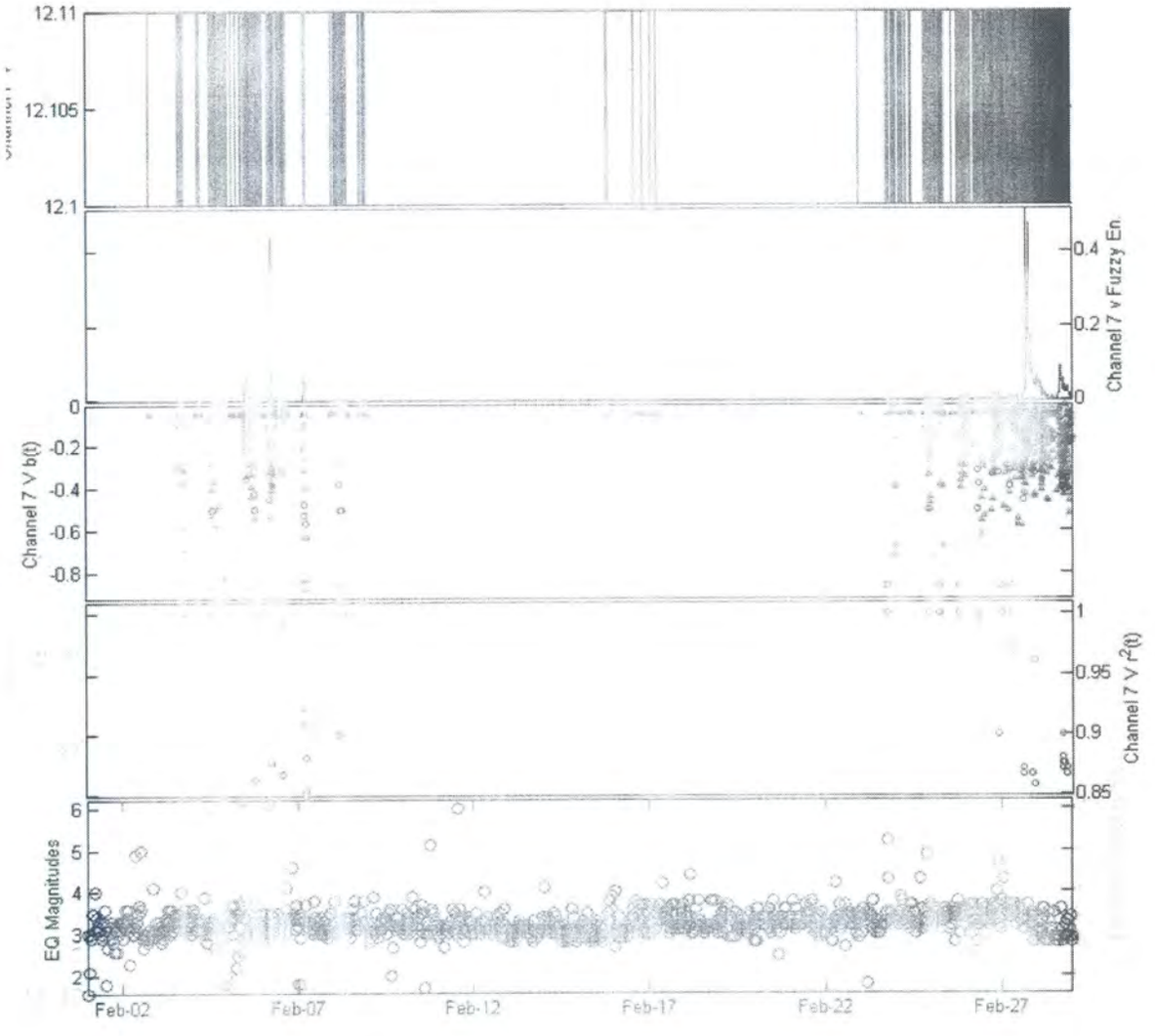

Channel 2 V
1500
1000
500
0.6
0.4
0.2
0
Channel 2 v Fuzzy En
0
-0.5
-1
-1.5
-2
-2.5
Channel 2 V b(t)
1
0.95
0.9
0.85
Channel 2 V r^2(t)
6
5
4
3
2
EQ Magnitudes
Feb-22
Feb-27

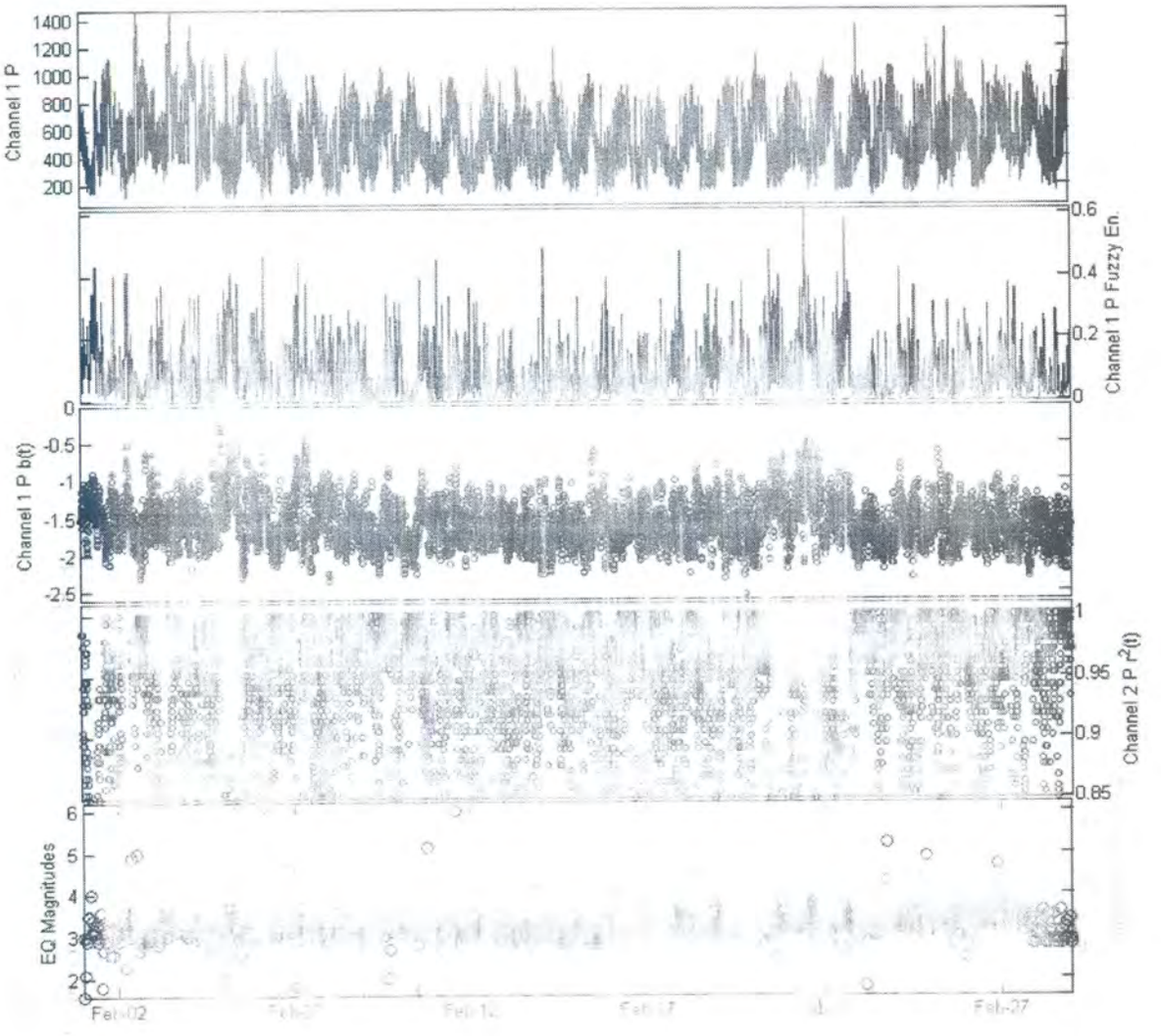

Channel 3 V
1000
800
600
400
200
Channel 3 v Fuzzy En.
1.2
1
0.8
0.6
0.4
0.2
0
Channel 3 V b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 3 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb
Feb-27

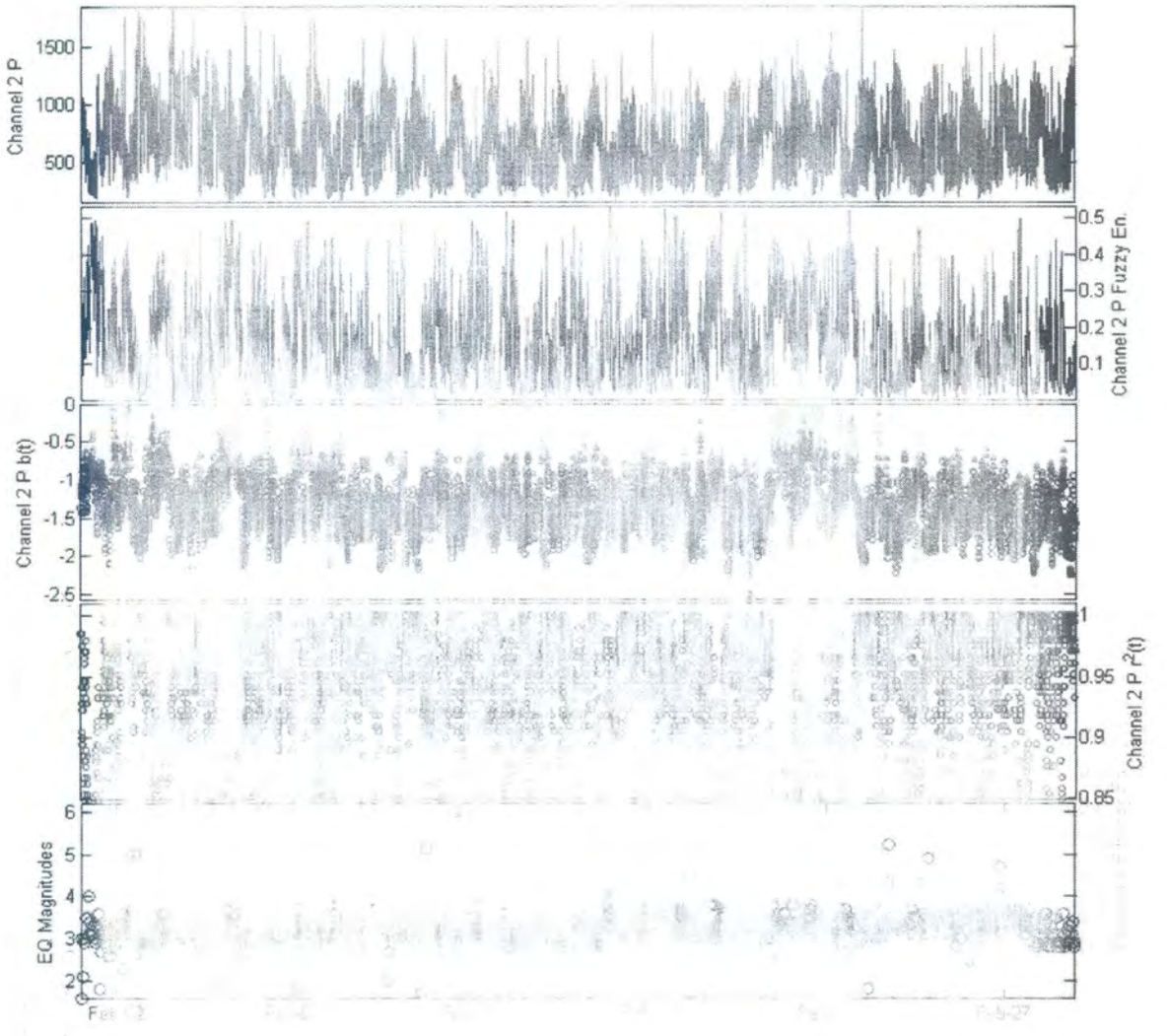

Channel 4 V
1200
1000
800
600
400
200
0
Channel 4 v Fuzzy En.
1
0.8
0.6
0.4
0.2
Channel 4 V b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb

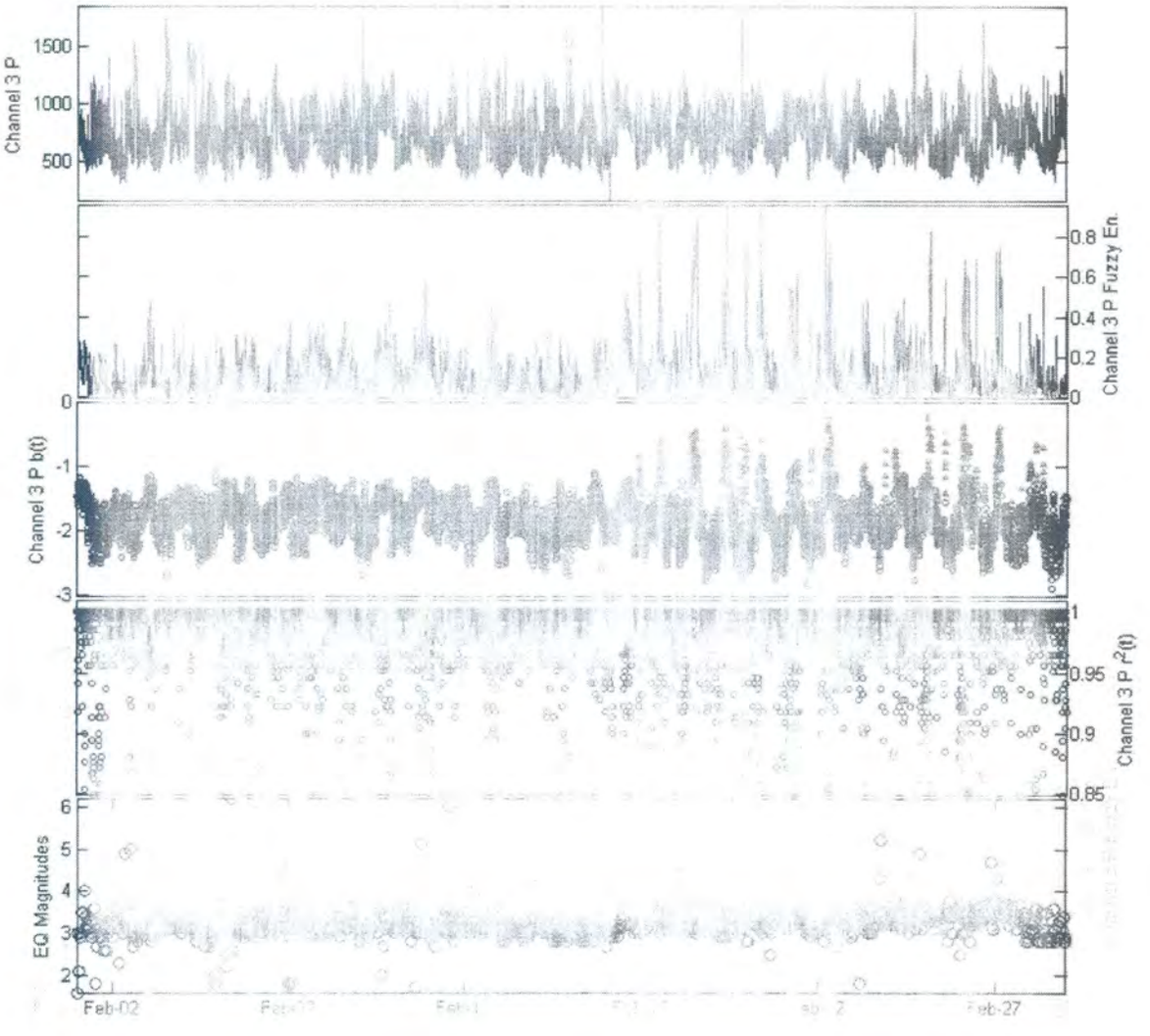

Channel 5 V
1200
1000
800
600
400
Channel 5 v Fuzzy En.
1.5
1
0.5
Channel 5 V b(t)
0
-0.5
-1
-1.5
Channel 5 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2

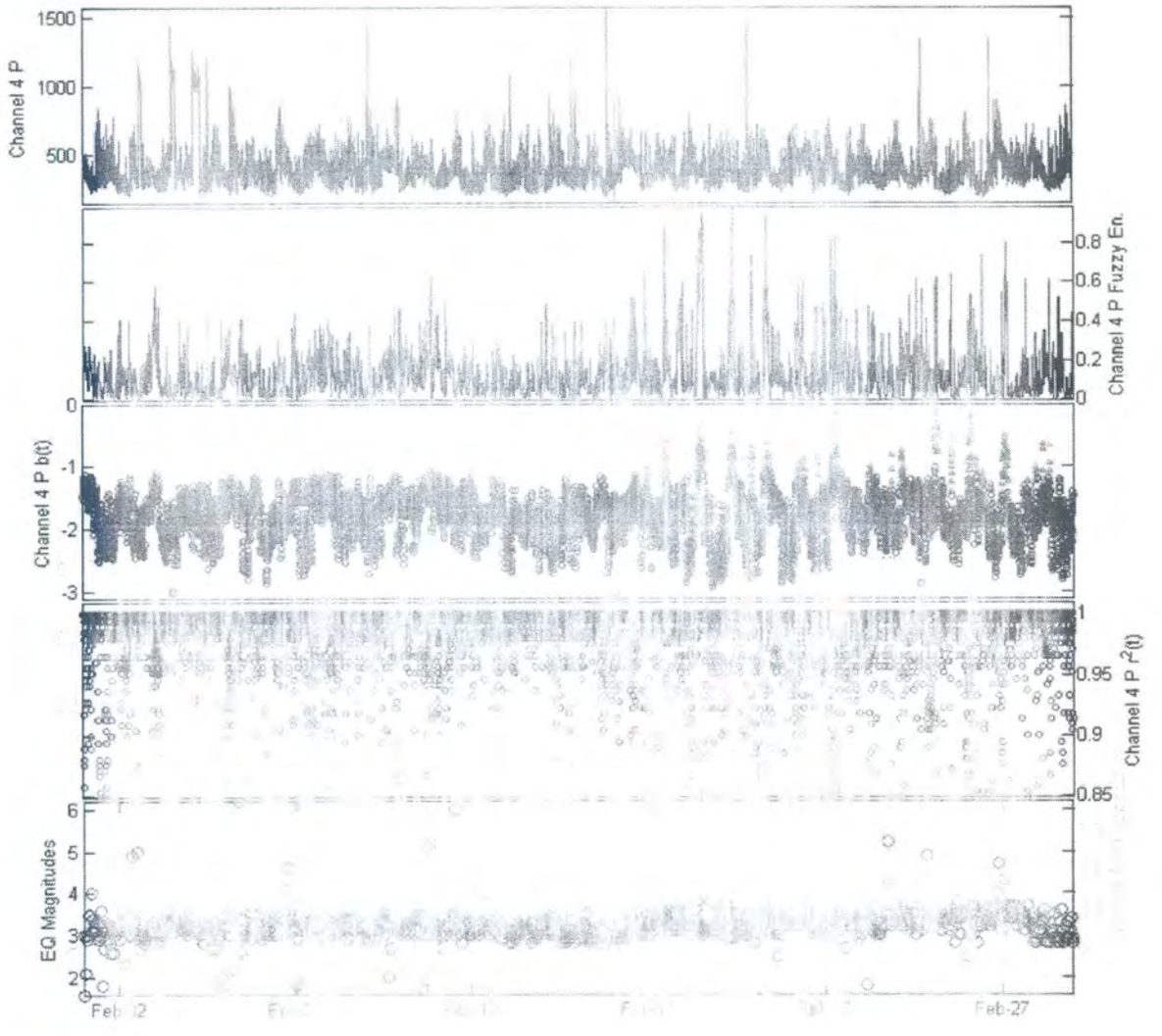

Channel 6 V
1200
1000
800
600
400
Channel 6 v Fuzzy En.
1.2
1
0.8
0.6
0.4
0.2
Channel 6 V b(t)
0
-0.5
-1
-1.5
Channel 6 V r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-07
Feb-27

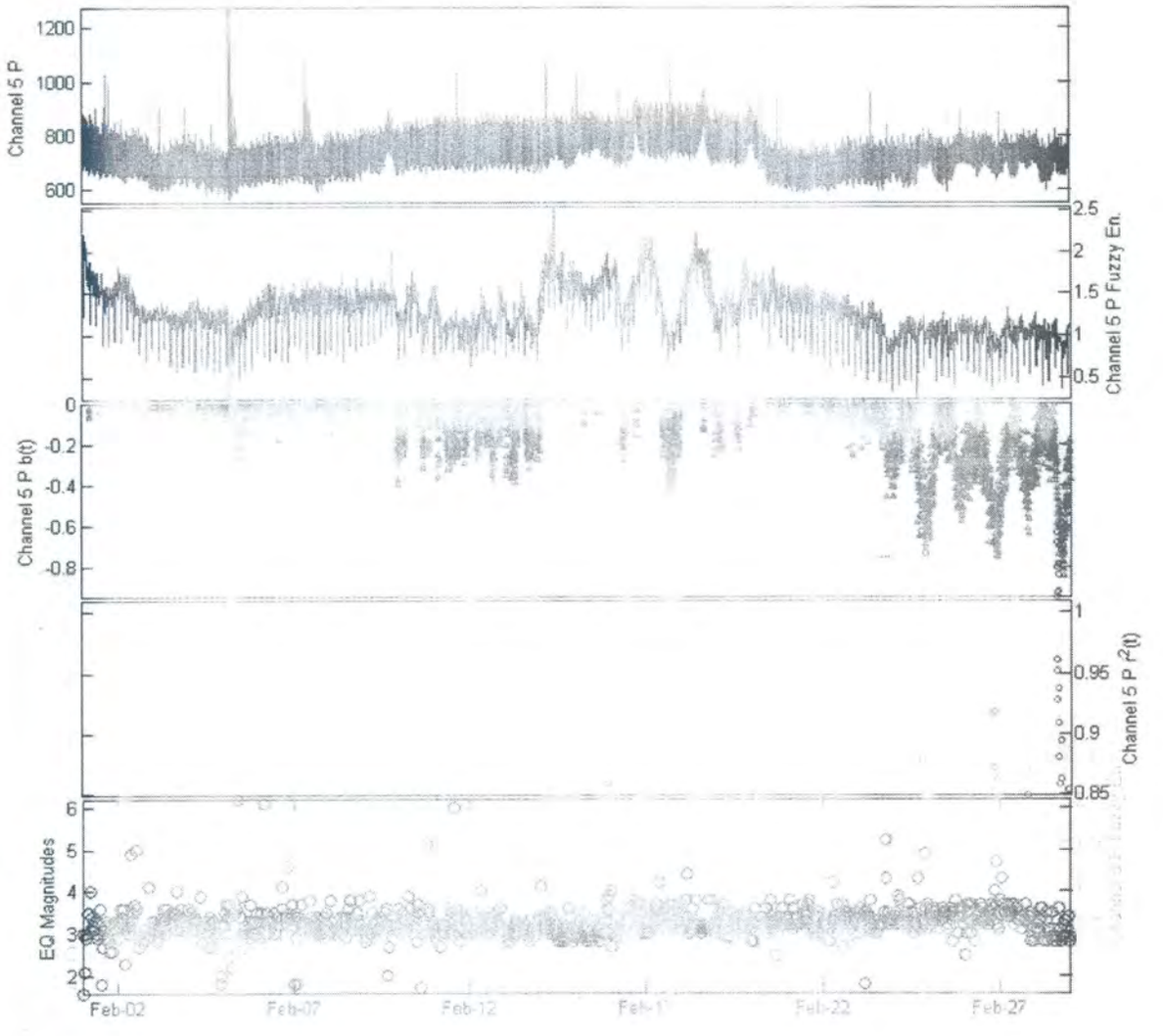

12.11
12.105
12.1
0.4
0.2
0
Channel 7 v Fuzzy En
0
-0.2
-0.4
-0.6
-0.8
Channel 7 V b(t)
1
0.95
0.9
0.85
Channel 7 V r^2(t)
6
5
4
3
2
EQ Magnitudes
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

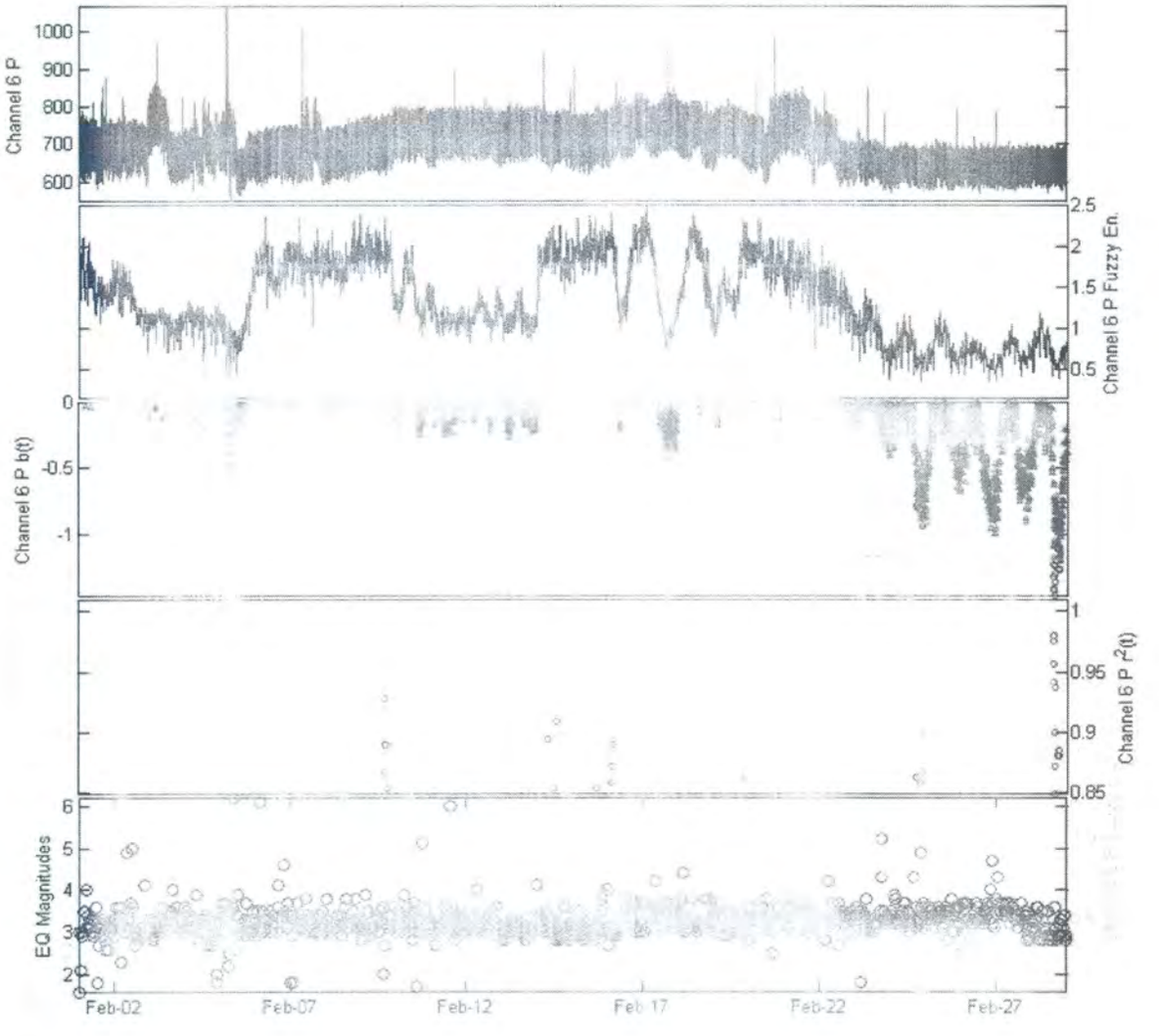

Channel 1 P
1400
1200
1000
800
600
400
200
Channel 1 P Fuzzy En.
0.6
0.4
0.2
0
Channel 1 P b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 P r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07

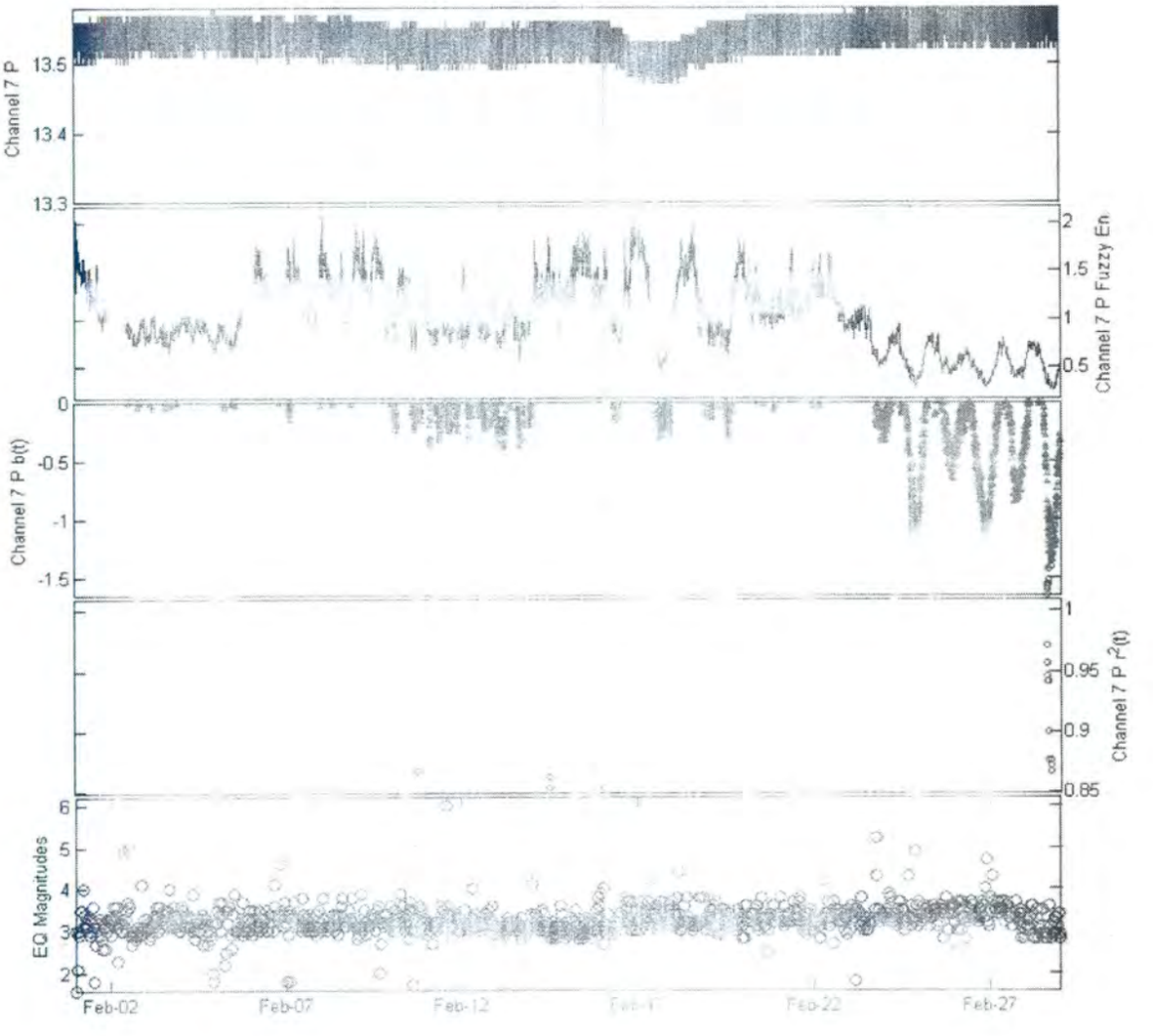

Channel 2 P
1500
1000
500
0
Channel 2 P Fuzzy En.
0.5
0.4
0.3
0.2
0.1
Channel 2 P b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2

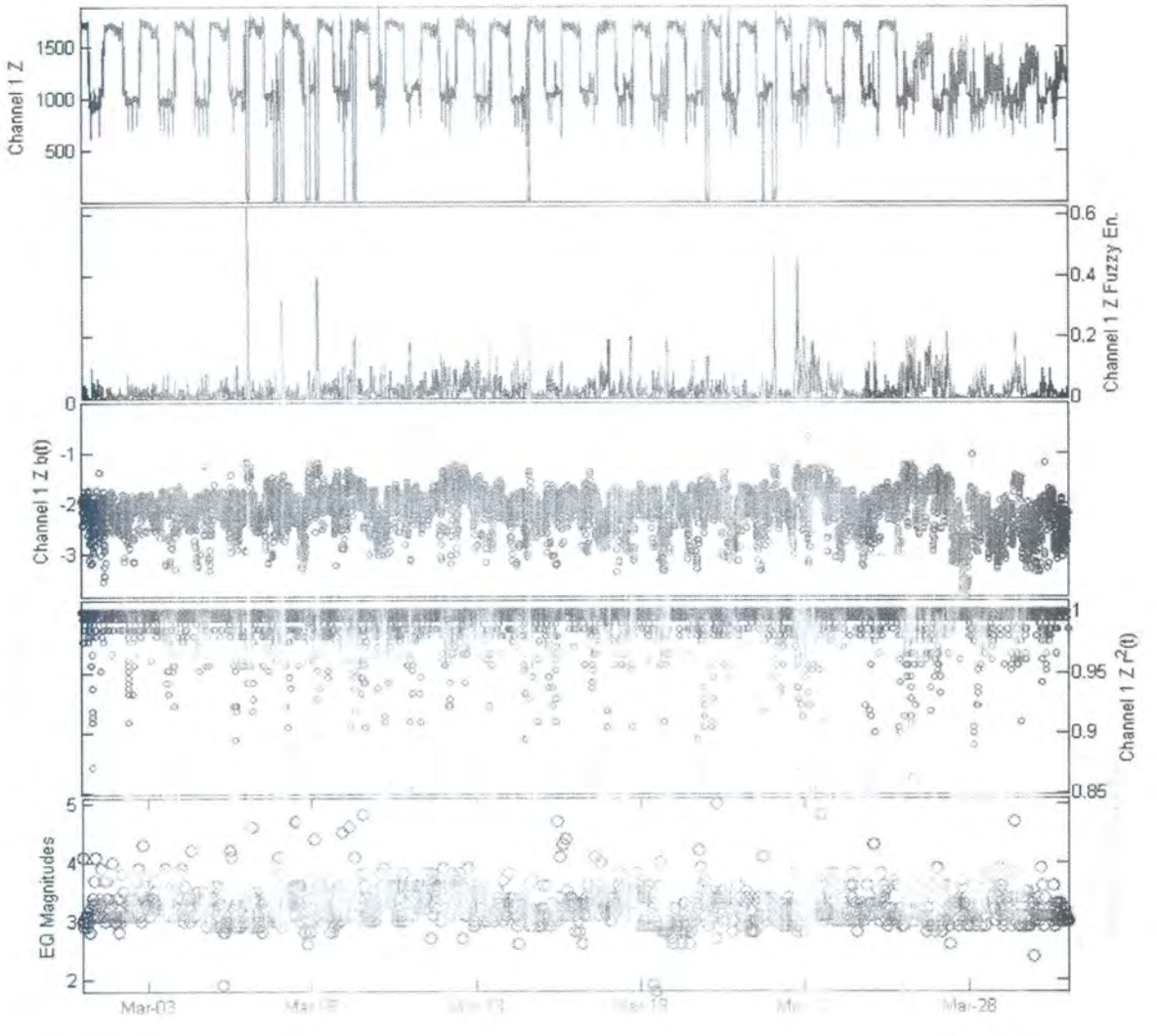

Channel 3 P
1500
1000
500
Channel 3 P Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 3 P b(t)
-1
-2
-3
Channel 3 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-27

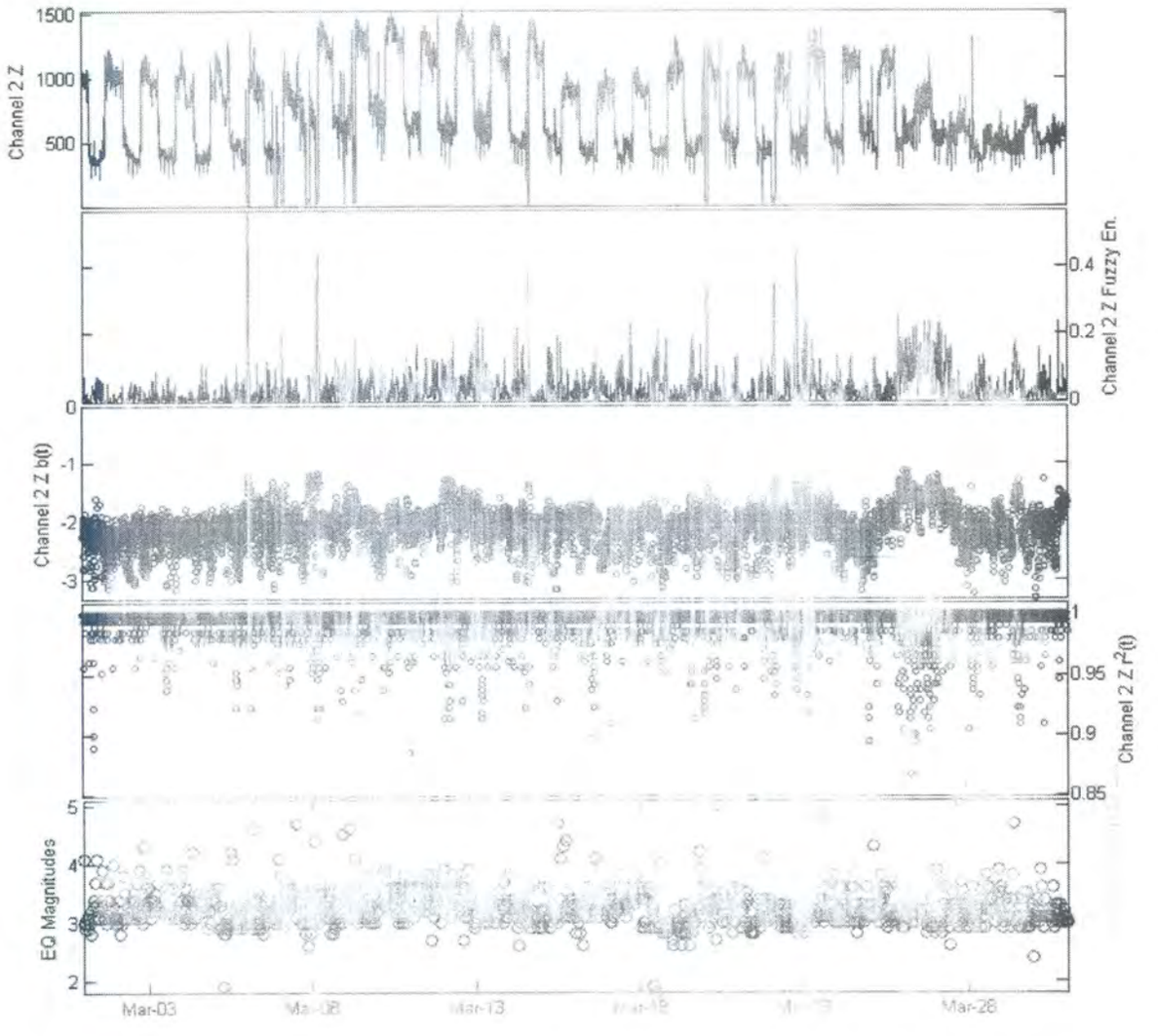

Channel 4 P
1500
1000
500
Channel 4 P Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 4 P b(t)
-1
-2
-3
Channel 4 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-27

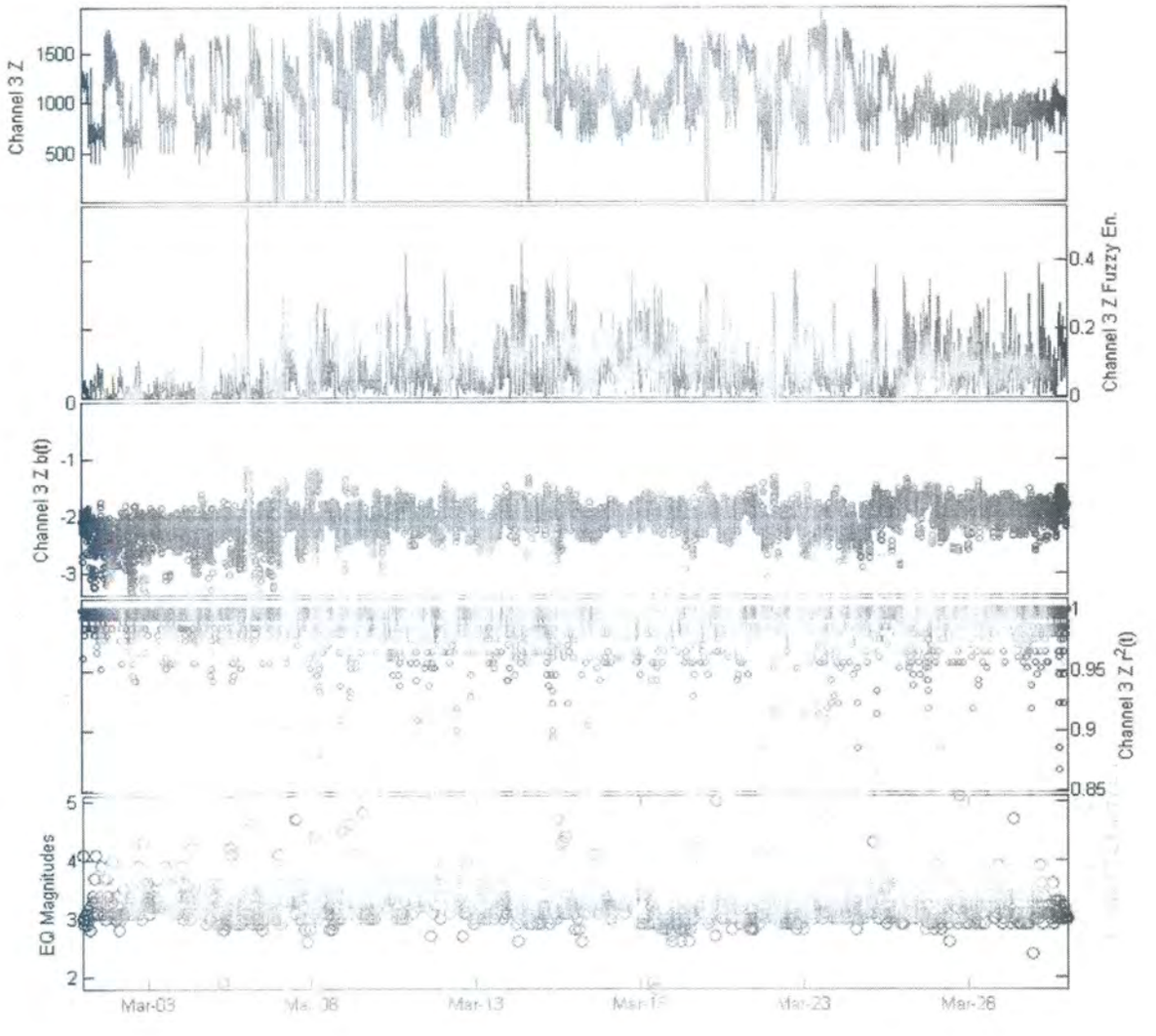

Channel 5 P
1200
1000
800
600
Channel 5 P Fuzzy En.
2.5
2
1.5
1
0.5
Channel 5 P b(t)
0
-0.2
-0.4
-0.6
-0.8
Channel 5 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

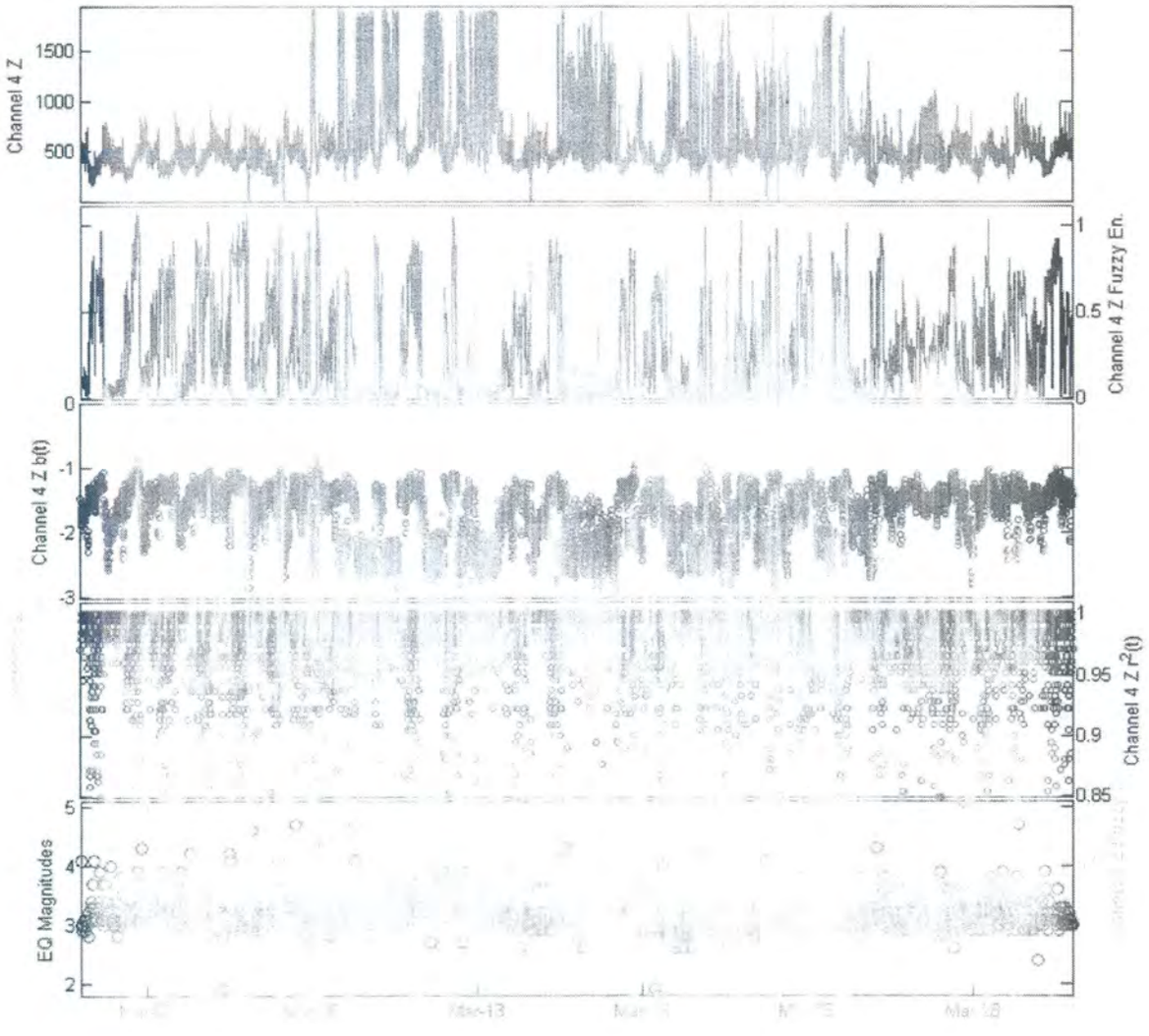

Channel 6 P
1000
900
800
700
600
Channel 6 P Fuzzy En.
2.5
2
1.5
1
0.5
Channel 6 P b(t)
0
-0.5
-1
Channel 6 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

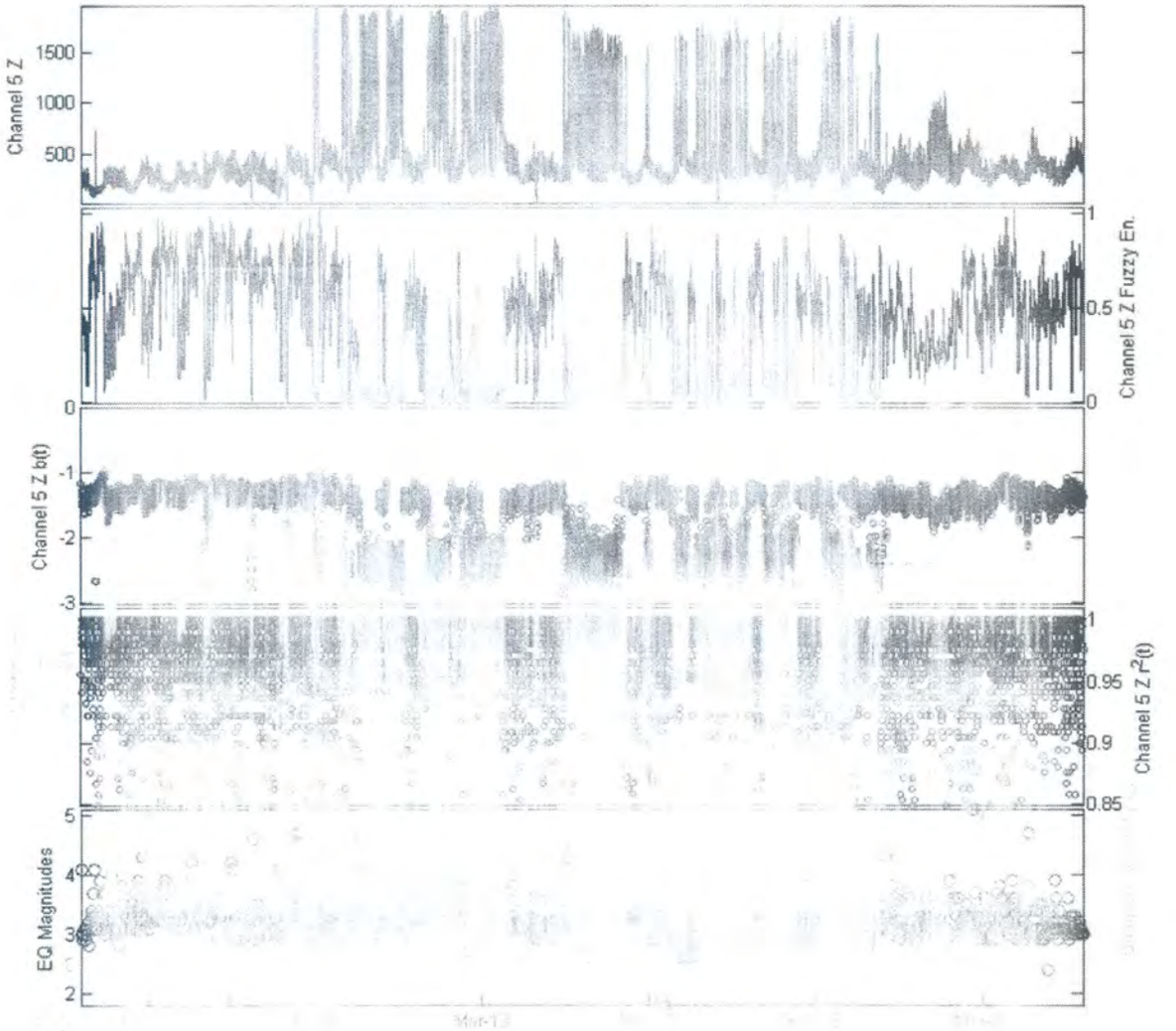

Channel 7 P
13.5
13.4
13.3
Channel 7 P Fuzzy En.
2
1.5
1
0.5
Channel 7 P b(t)
0
-0.5
-1
-1.5
Channel 7 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
6
5
4
3
2
Feb-02
Feb-07
Feb-12
Feb-17
Feb-22
Feb-27

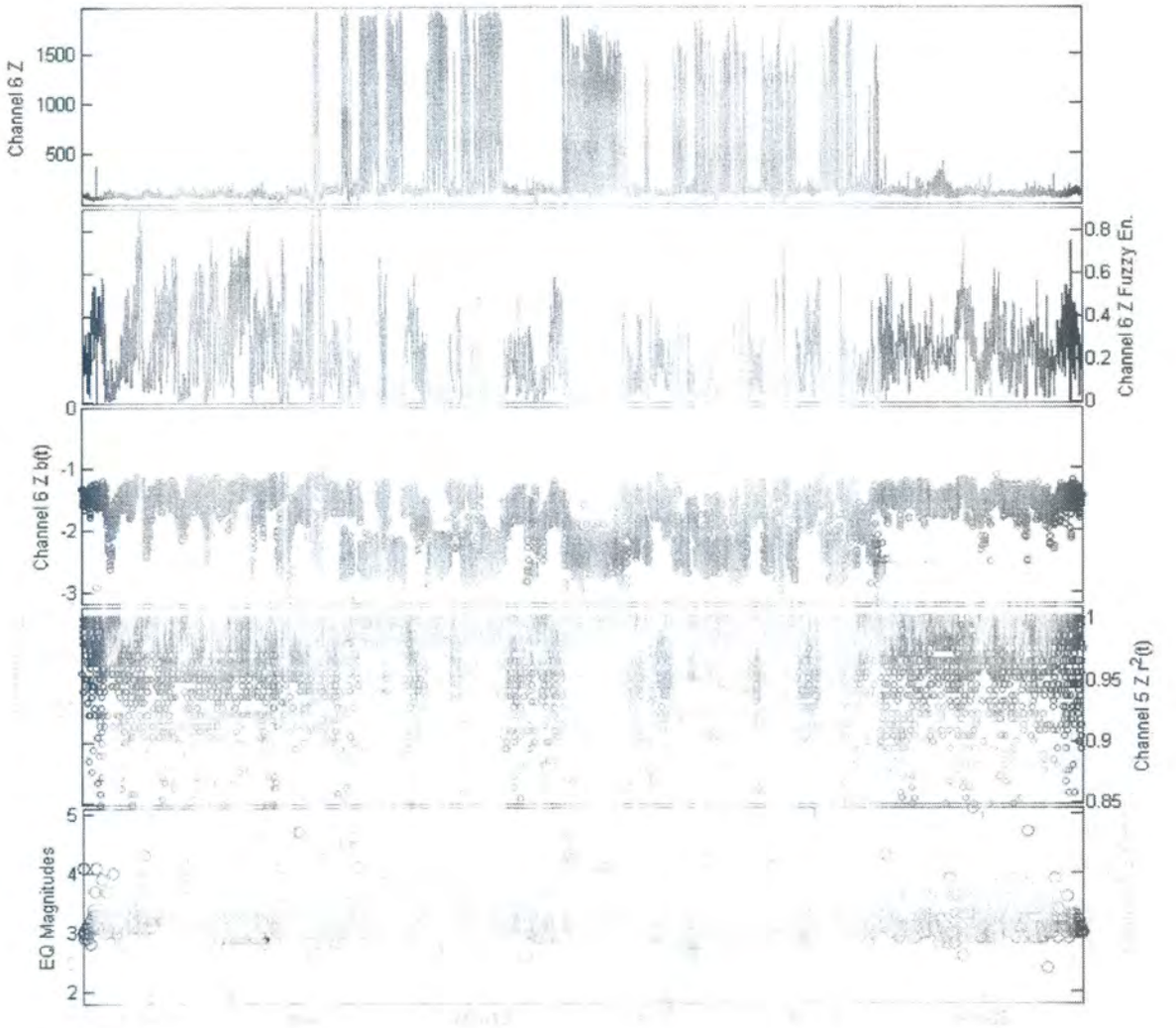

Channel 1 Z
1500
1000
500
Channel 1 Z Fuzzy En.
0.6
0.4
0.2
0
Channel 1 Z b(t)
-1
-2
-3
Channel 1 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar-28

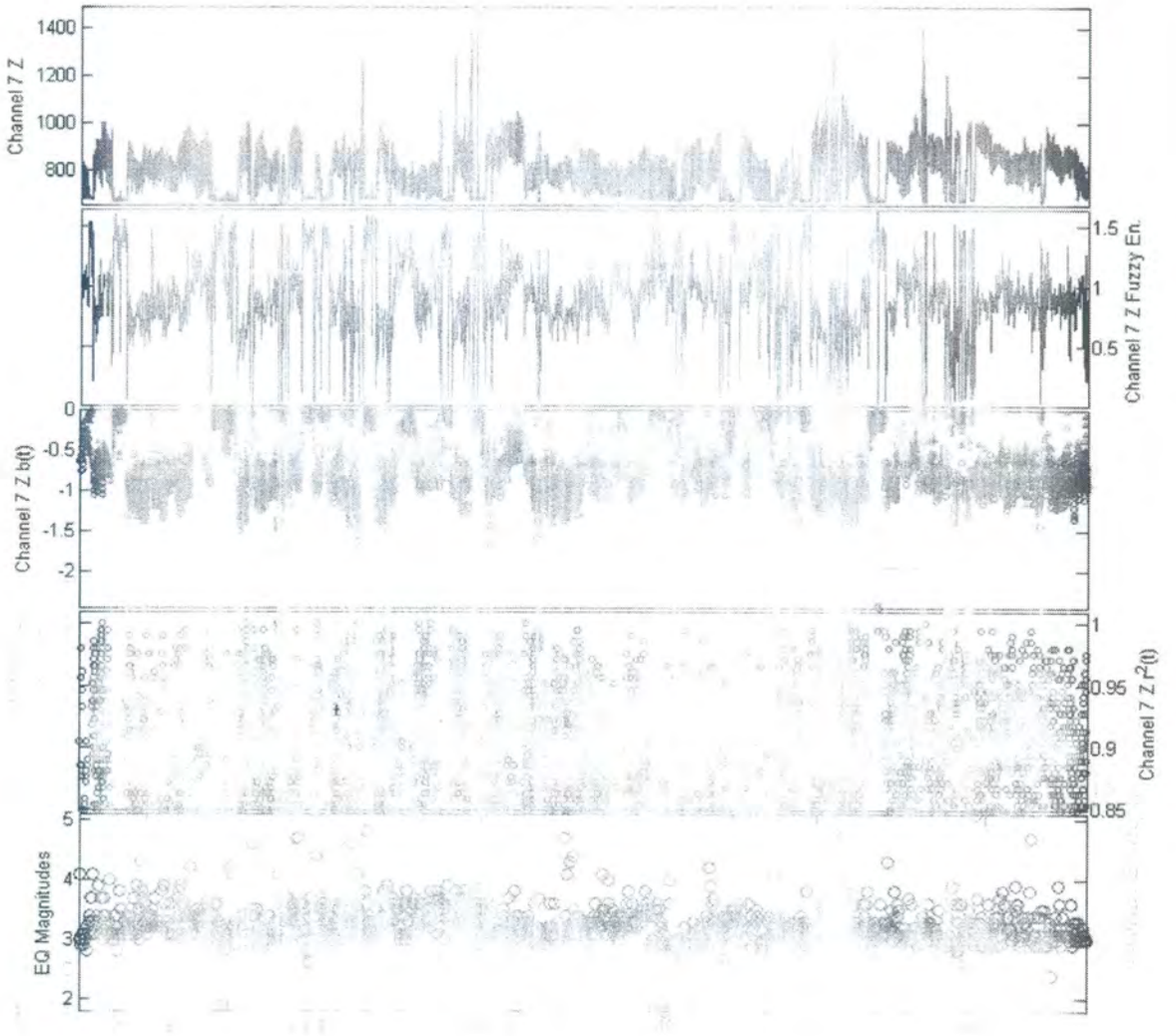

Channel 2 Z
1500
1000
500
Channel 2 Z Fuzzy En.
0.4
0.2
0
Channel 2 Z b(t)
-1
-2
-3
Channel 2 Z F²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar-28

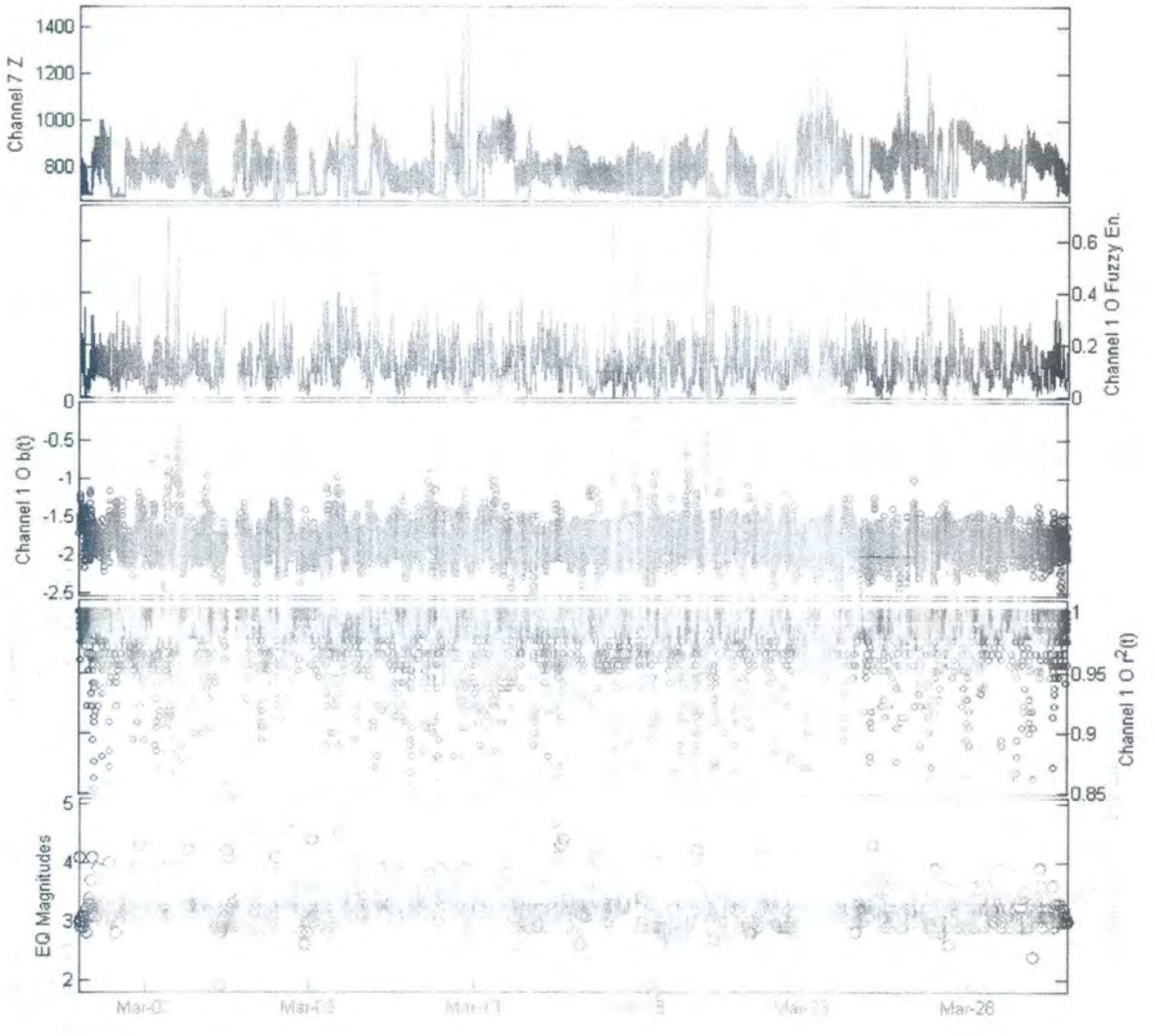

Channel 3 Z
1500
1000
500
Channel 3 Z Fuzzy En.
0.4
0.2
0
Channel 3 Z b(t)
0
-1
-2
-3
Channel 3 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar-13
Mar-23
Mar-28

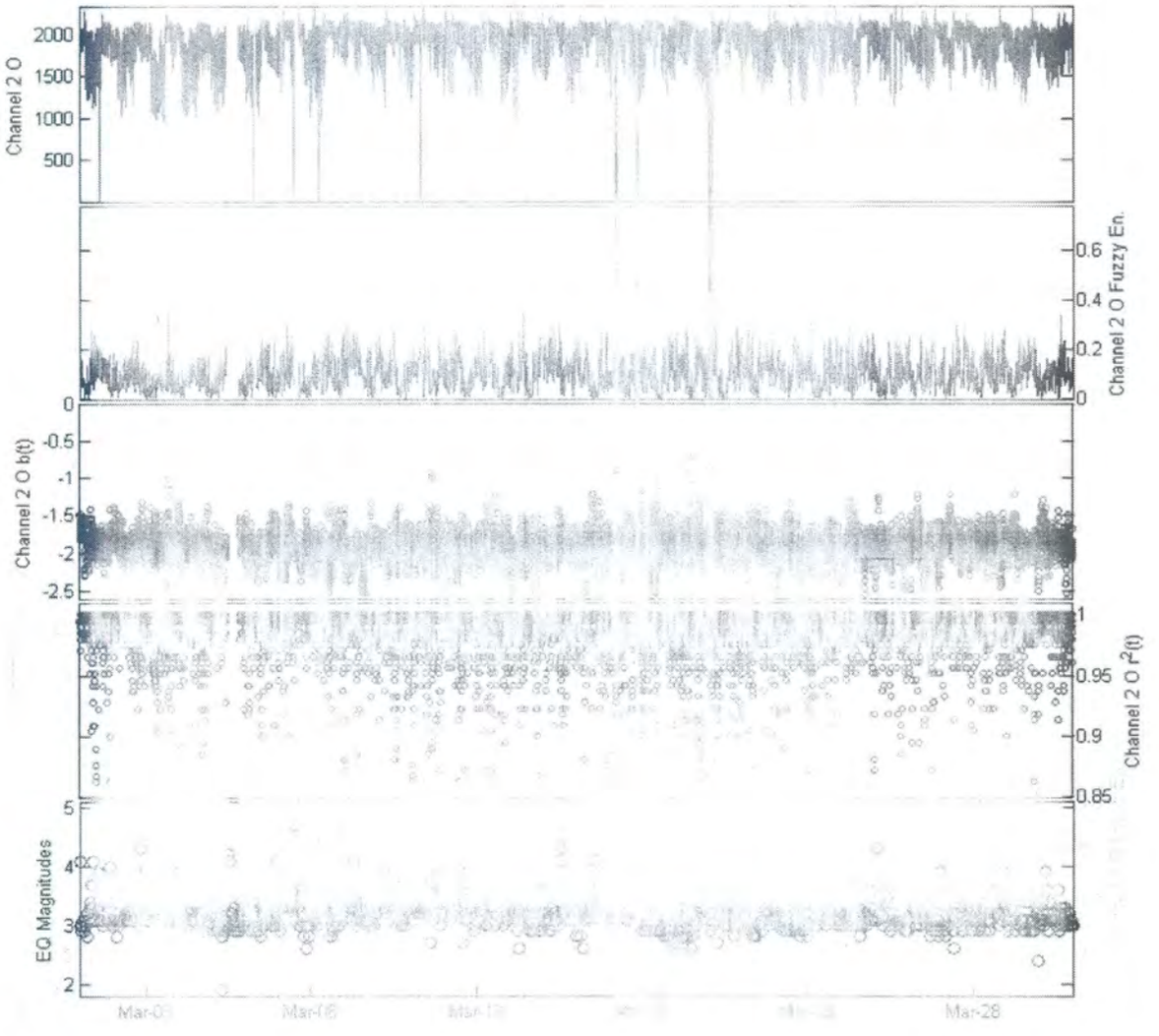

Channel 4 Z
1500
1000
500
0
1
0.5
0
Channel 4 Z Fuzzy En.
Channel 4 Z b(t)
-1
-2
-3
1
0.95
0.9
0.85
Channel 4 Z r^2(t)
5
4
3
2
EQ Magnitudes

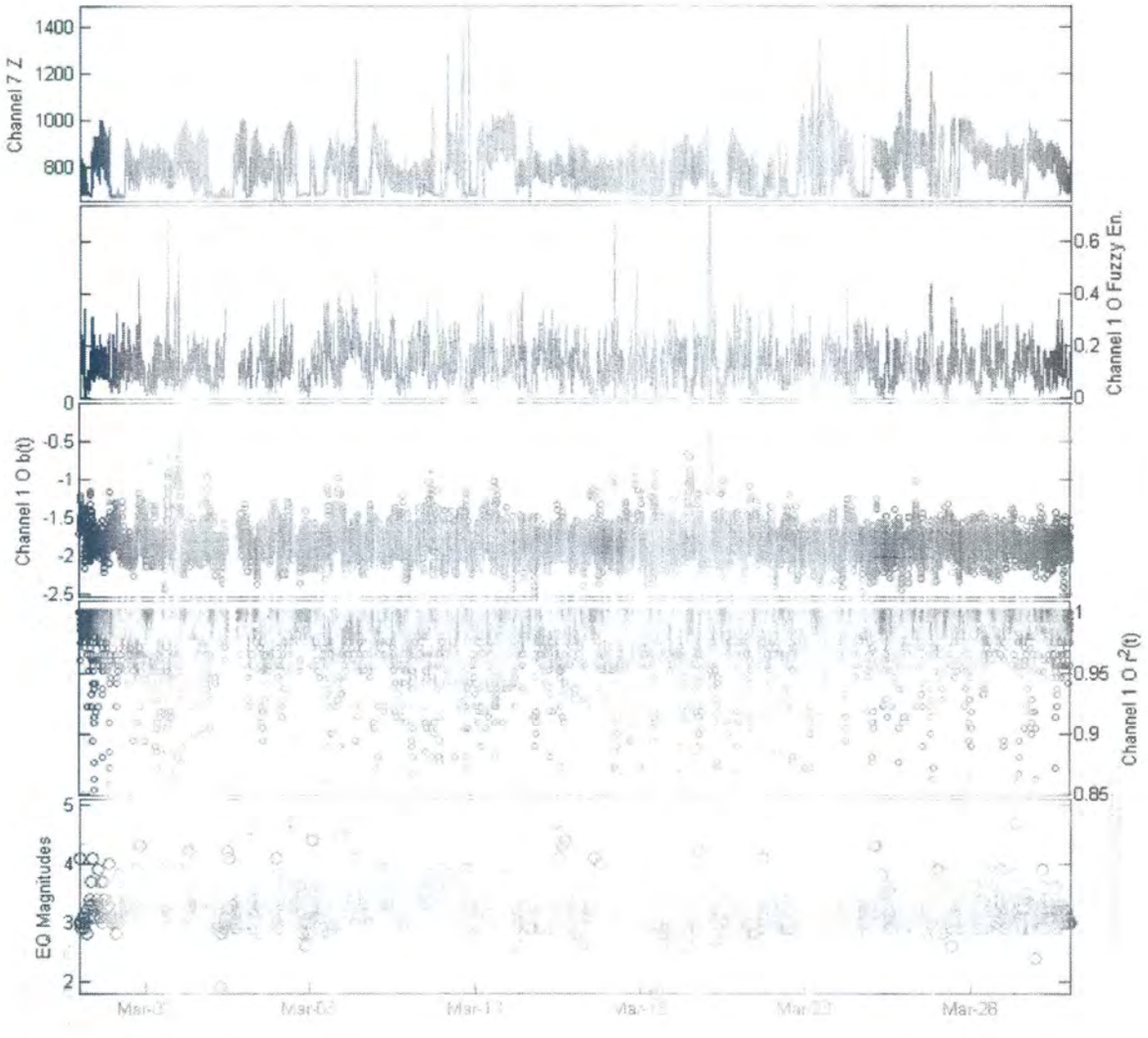

Channel 5 Z
1500
1000
500
Channel 5 Z Fuzzy En.
1
0.5
0
Channel 5 Z b(t)
-1
-2
-3
Channel 5 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

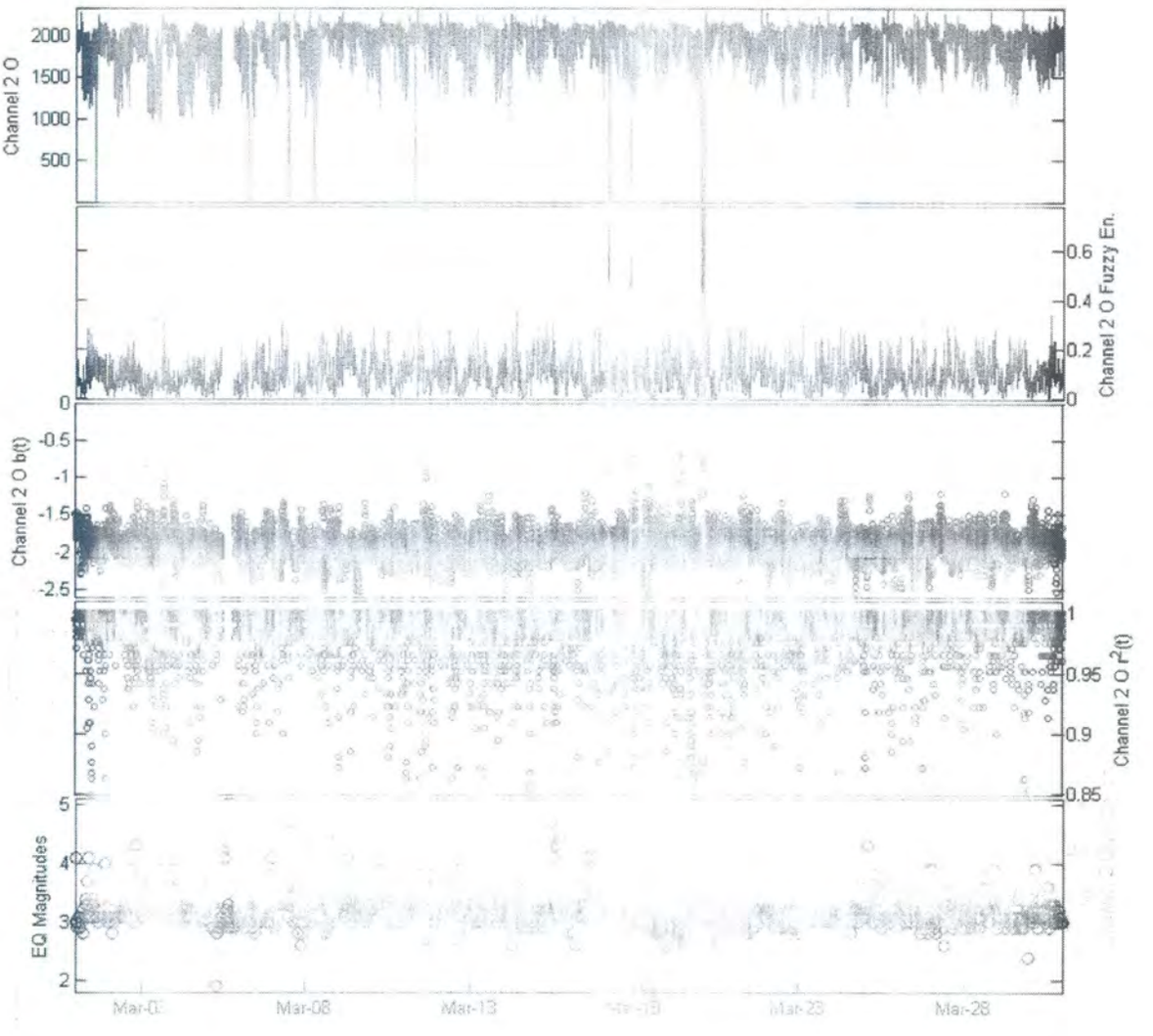

Channel 6 Z
1500
1000
500
0
Channel 6 Z Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 6 Z b(t)
-1
-2
-3
Channel 5 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

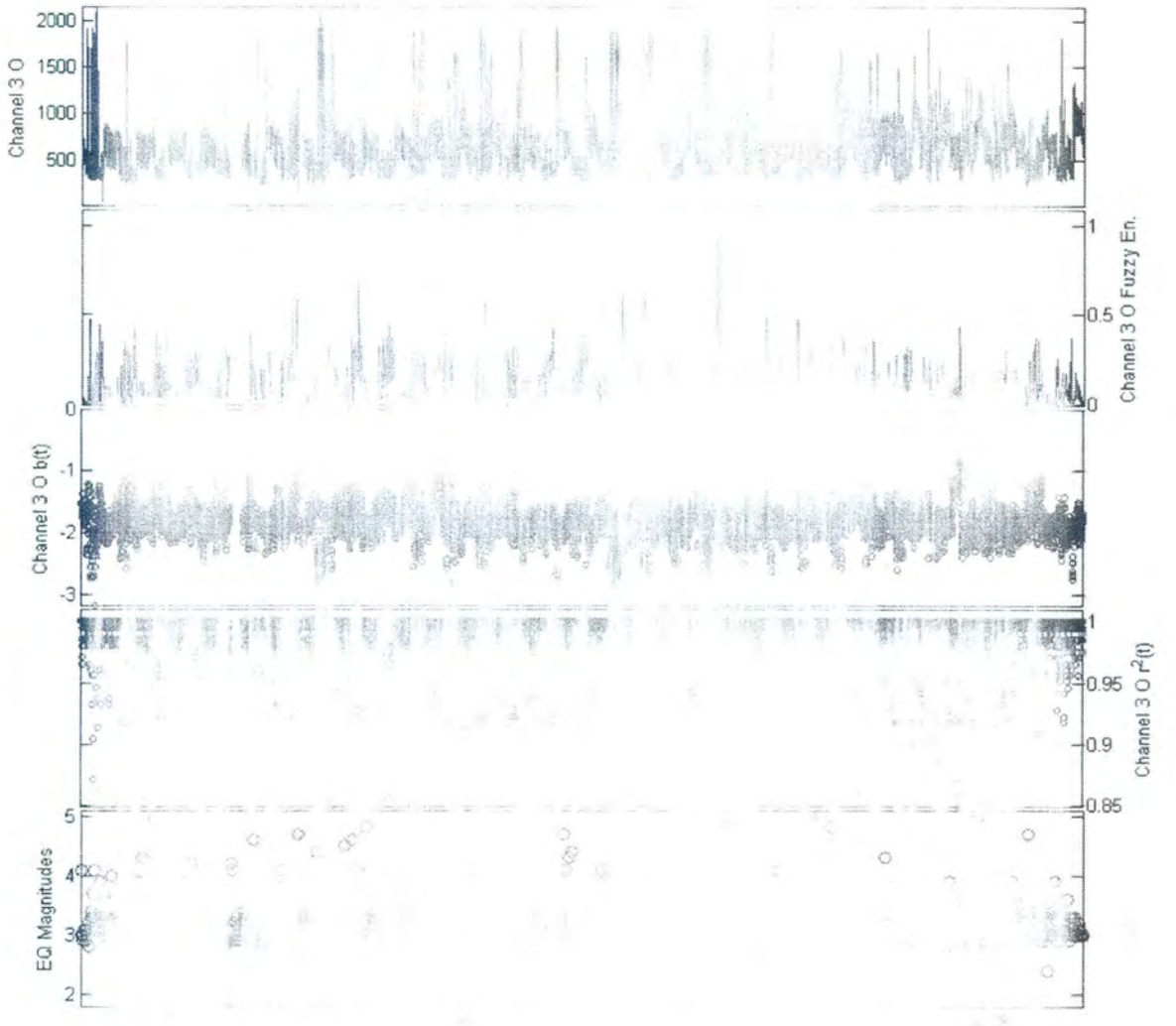

Channel 7 Z
1400
1200
1000
800
Channel 7 Z Fuzzy En.
1.5
1
0.5
0
Channel 7 Z b(t)
-0.5
-1
-1.5
-2
Channel 7 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
t

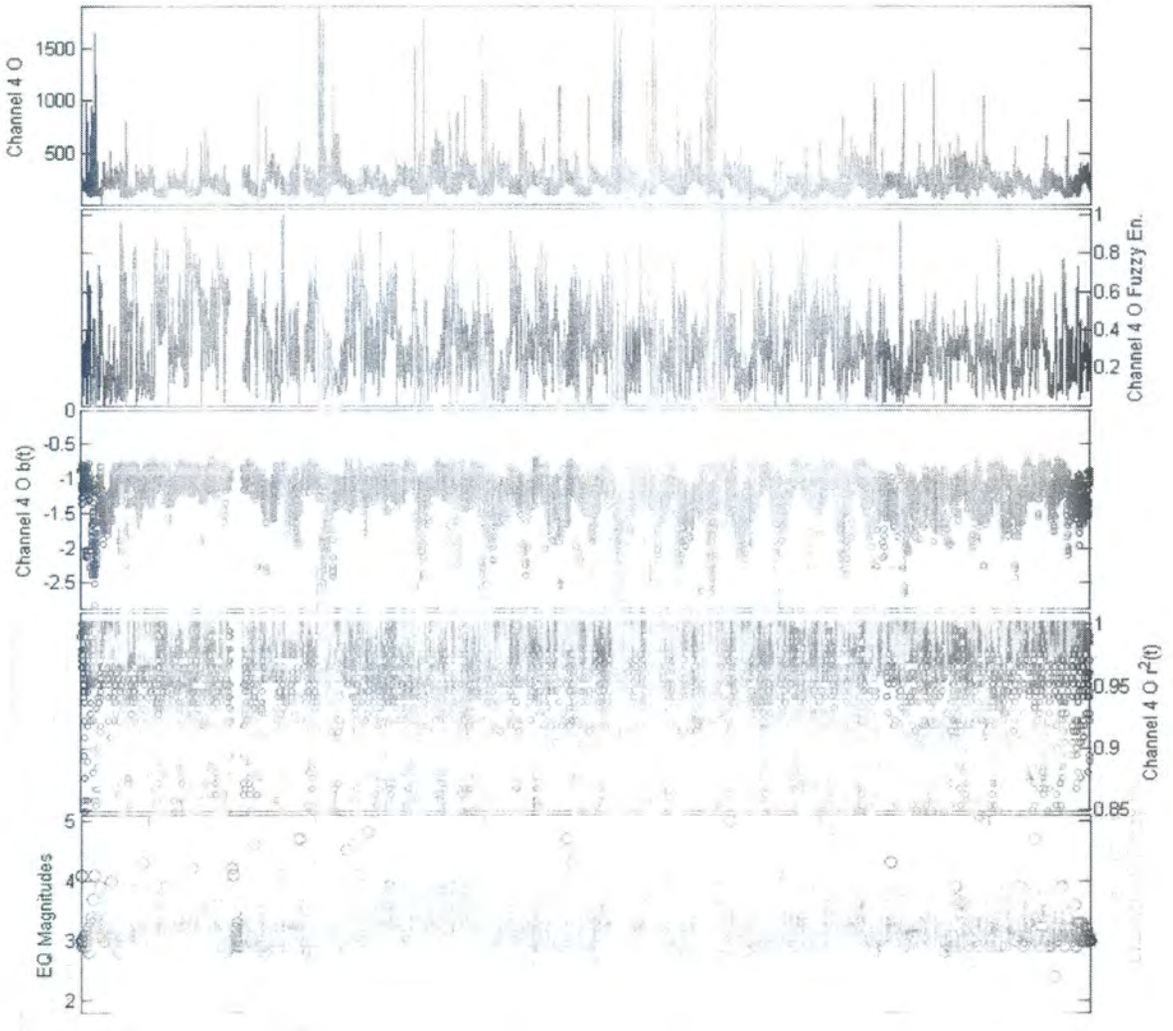

Channel 7 Z
1400
1200
1000
800
Channel 1 O Fuzzy En.
0.6
0.4
0.2
0
Channel 1 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-28

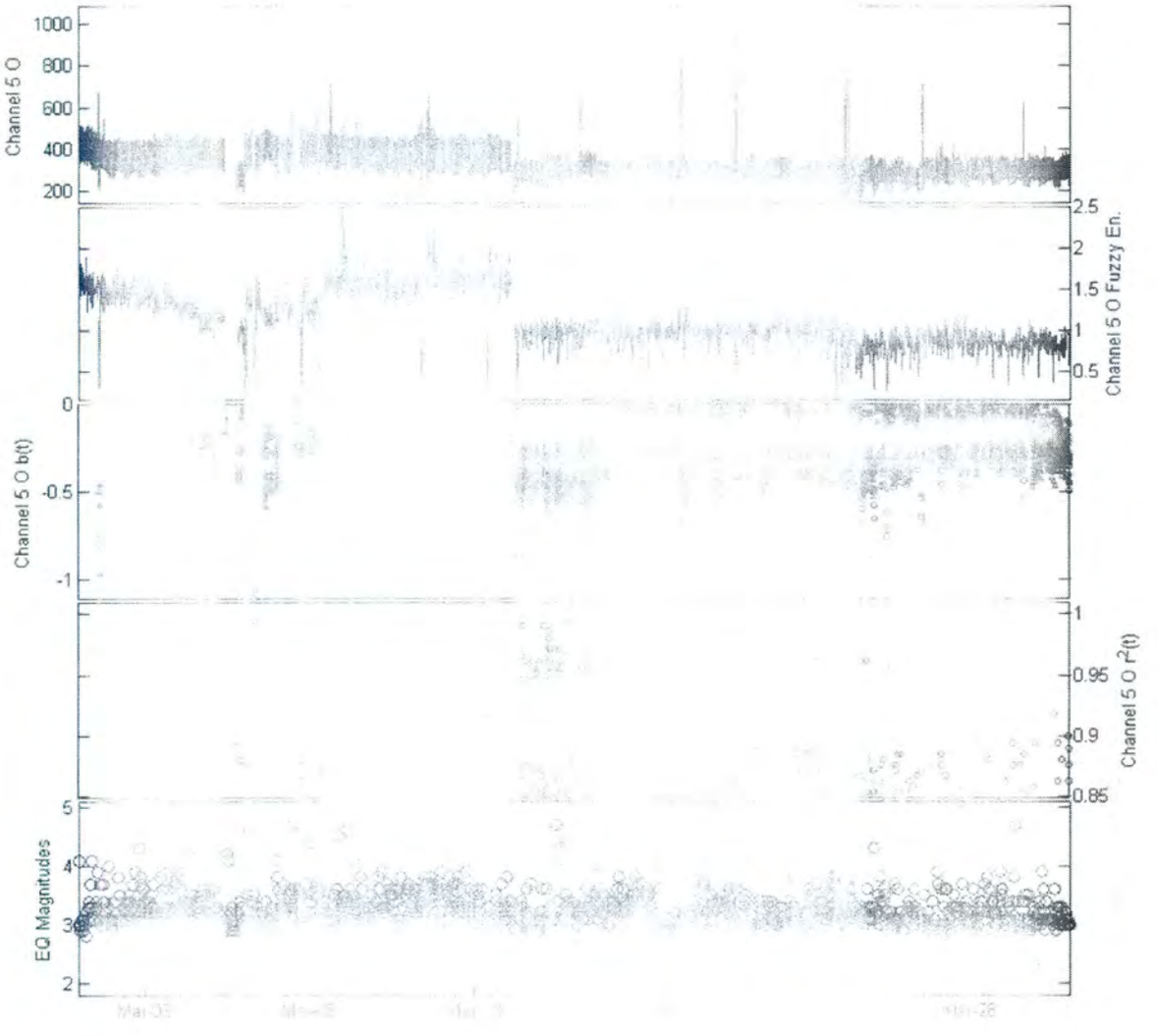

Channel 2 O
2000
1500
1000
500
Channel 2 O Fuzzy En.
0.6
0.4
0.2
0
Channel 2 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-28

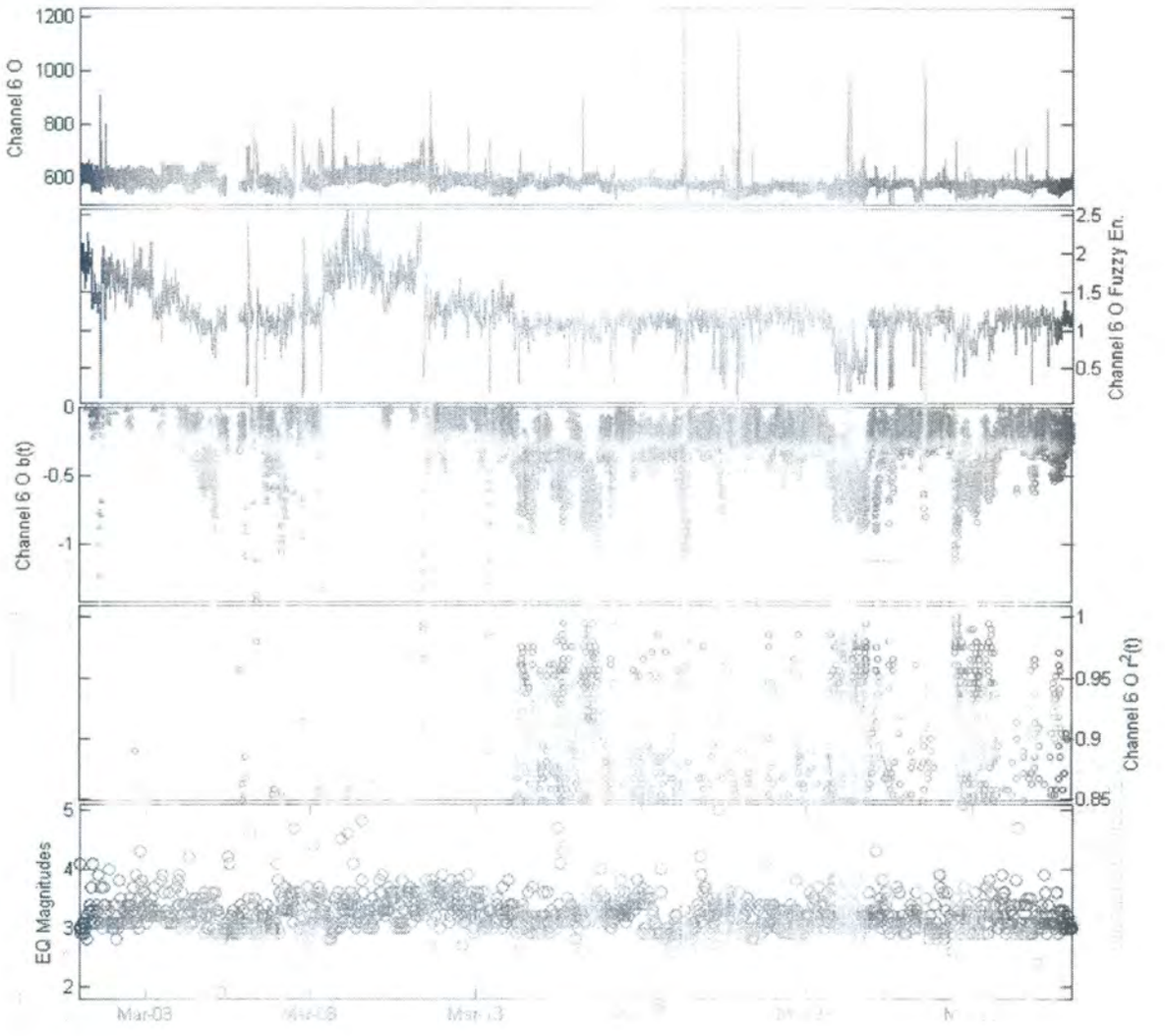

Channel 7 Z
1400
1200
1000
800
Channel 1 O Fuzzy En.
0.6
0.4
0.2
0
Channel 1 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 O r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar-08
Mar-13
Mar-18
Mar-23
Mar-28

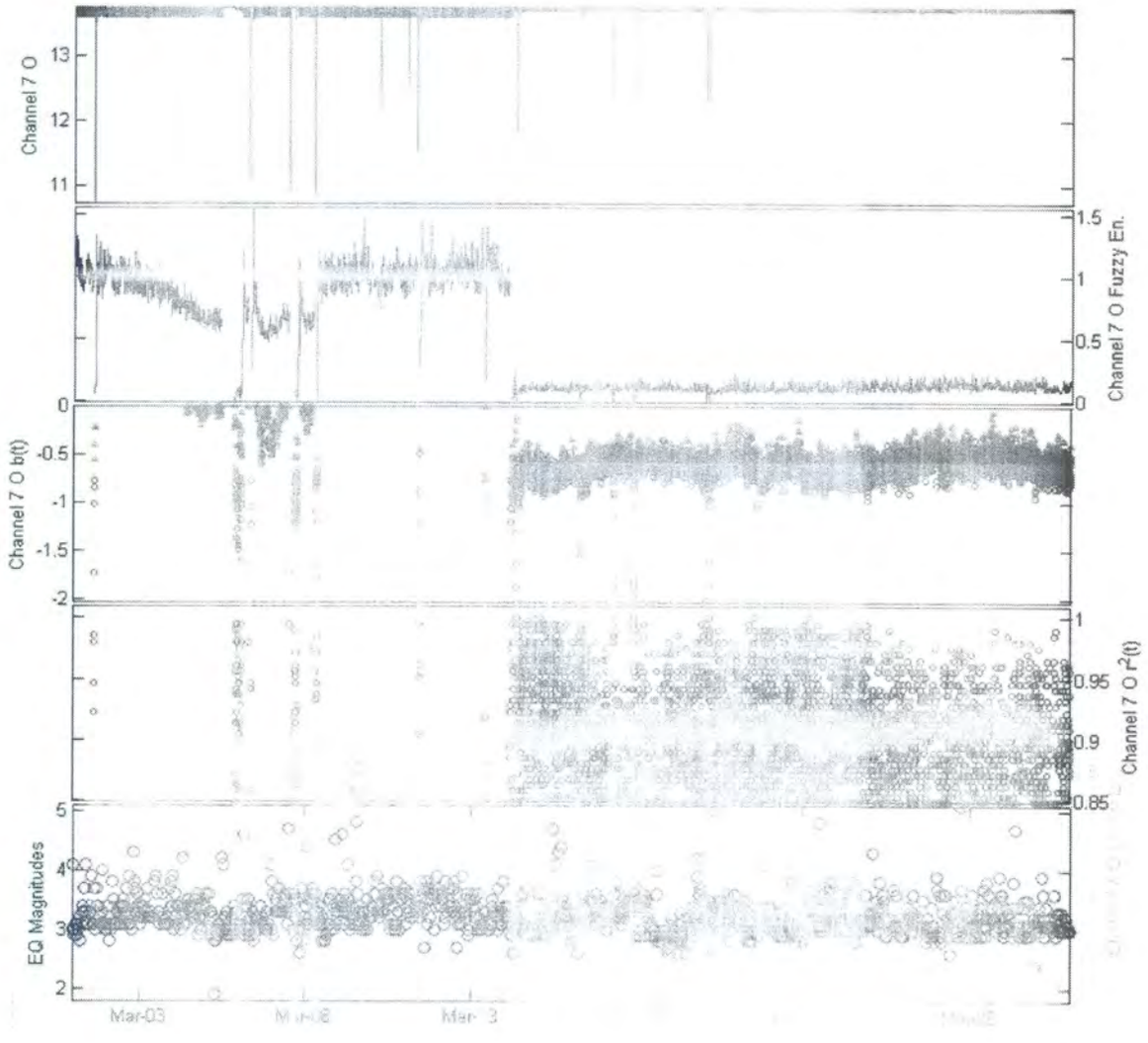

Channel 2 O
2000
1500
1000
500
0
Channel 2 O Fuzzy En.
0.6
0.4
0.2
0
Channel 2 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar-08
Mar-13
Mar-18
Mar-23
Mar-28

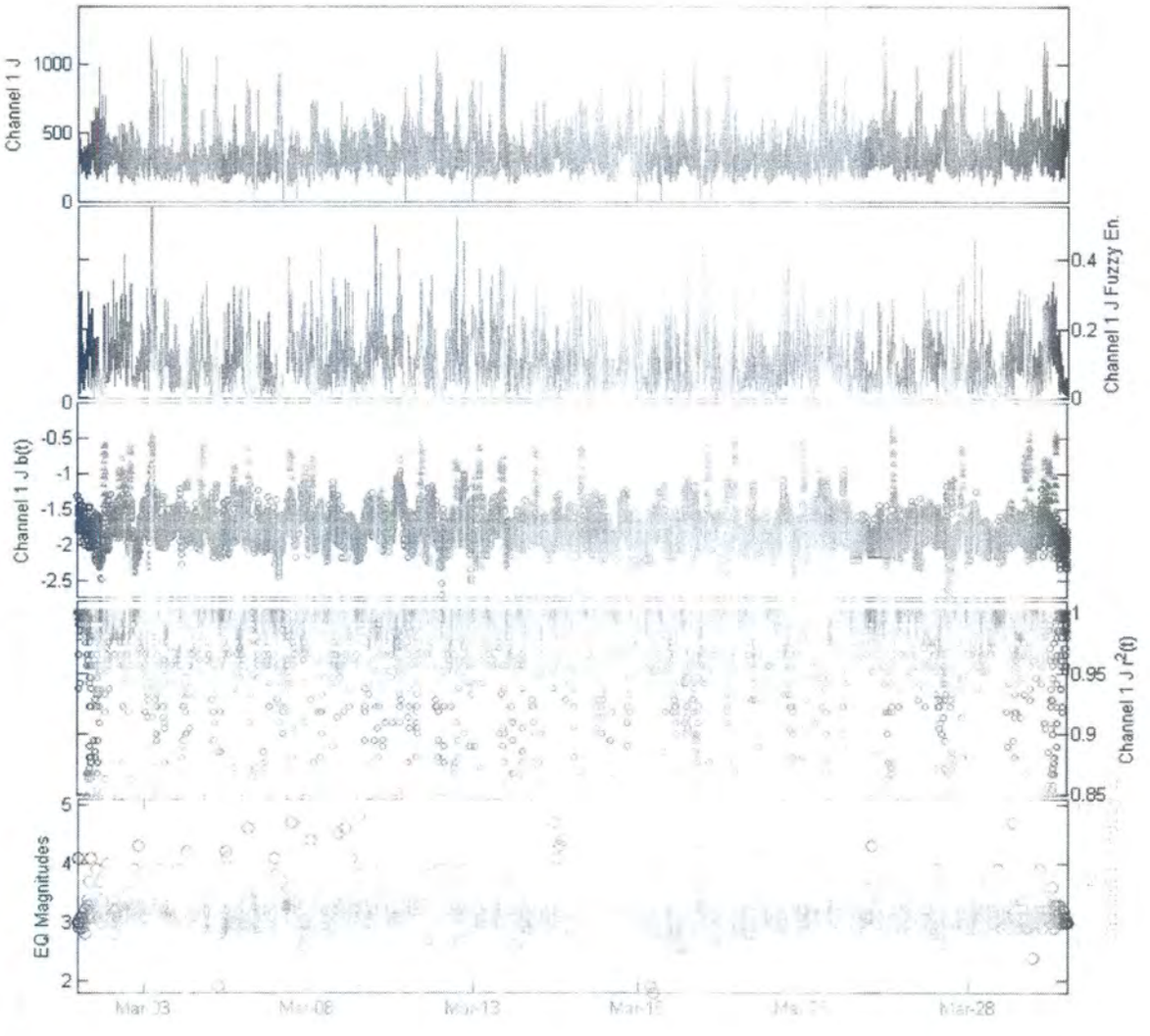

Channel 3 O
2000
1500
1000
500
Channel 3 O Fuzzy En.
1
0.5
0
Channel 3 O b(t)
-1
-2
-3
Channel 3 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

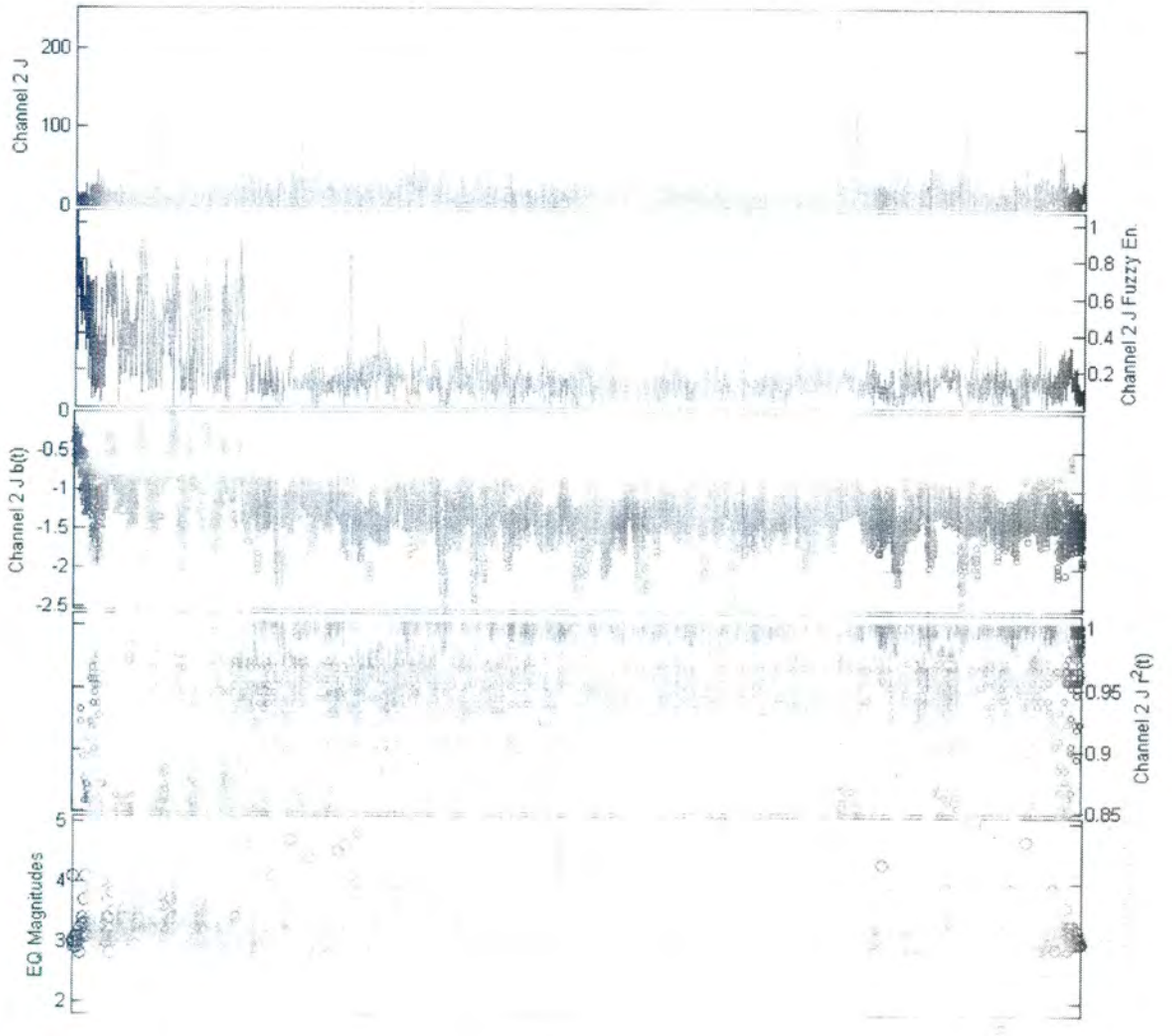

Channel 4 O
1500
1000
500
Channel 4 O Fuzzy En.
1
0.8
0.6
0.4
0.2
0
Channel 4 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

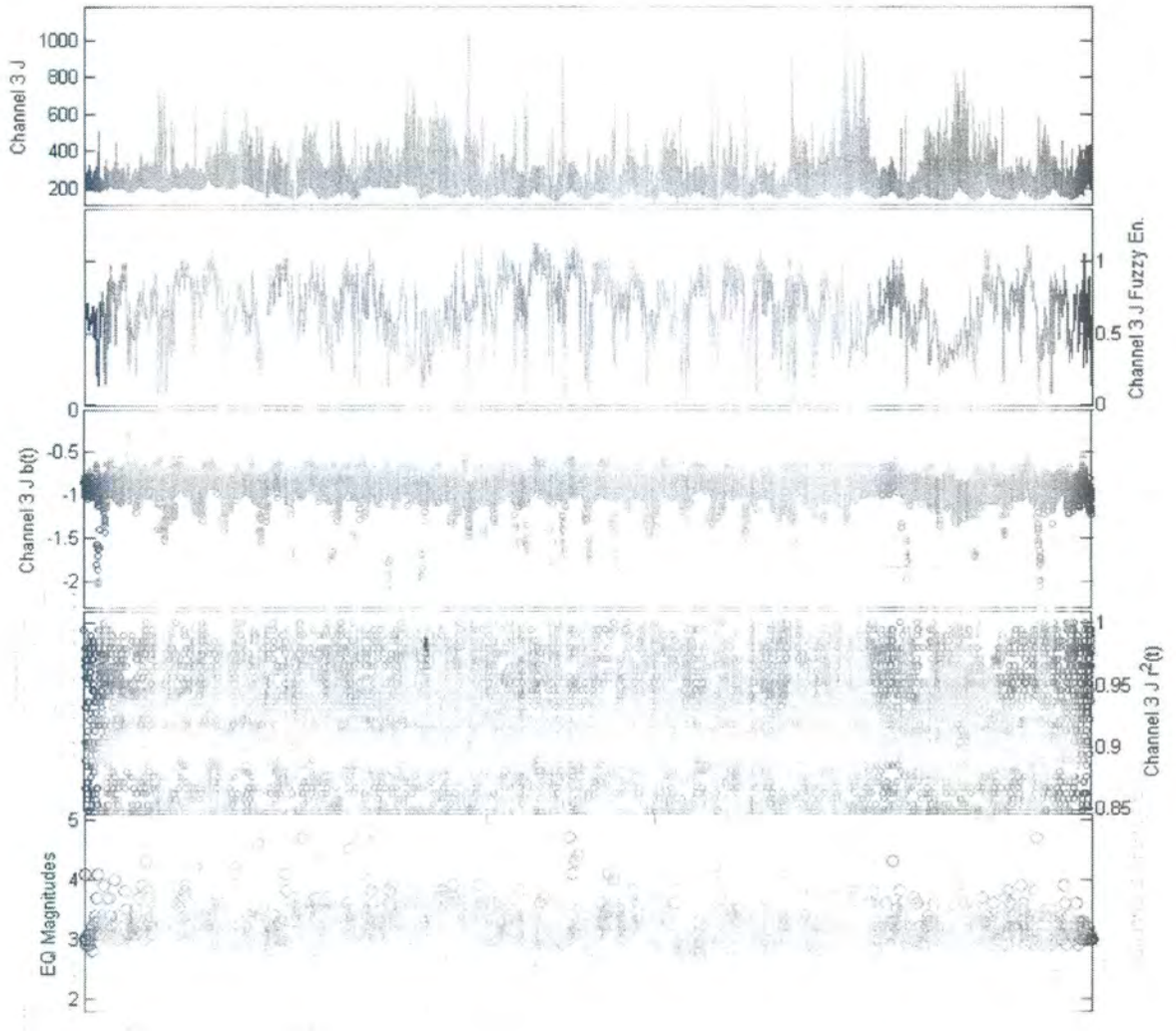

Channel 5 O
1000
800
600
400
200
Channel 5 O Fuzzy En.
2.5
2
1.5
1
0.5
Channel 5 O b(t)
0
-0.5
-1
Channel 5 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-05
Mar-15
Mar-25
Mar-28

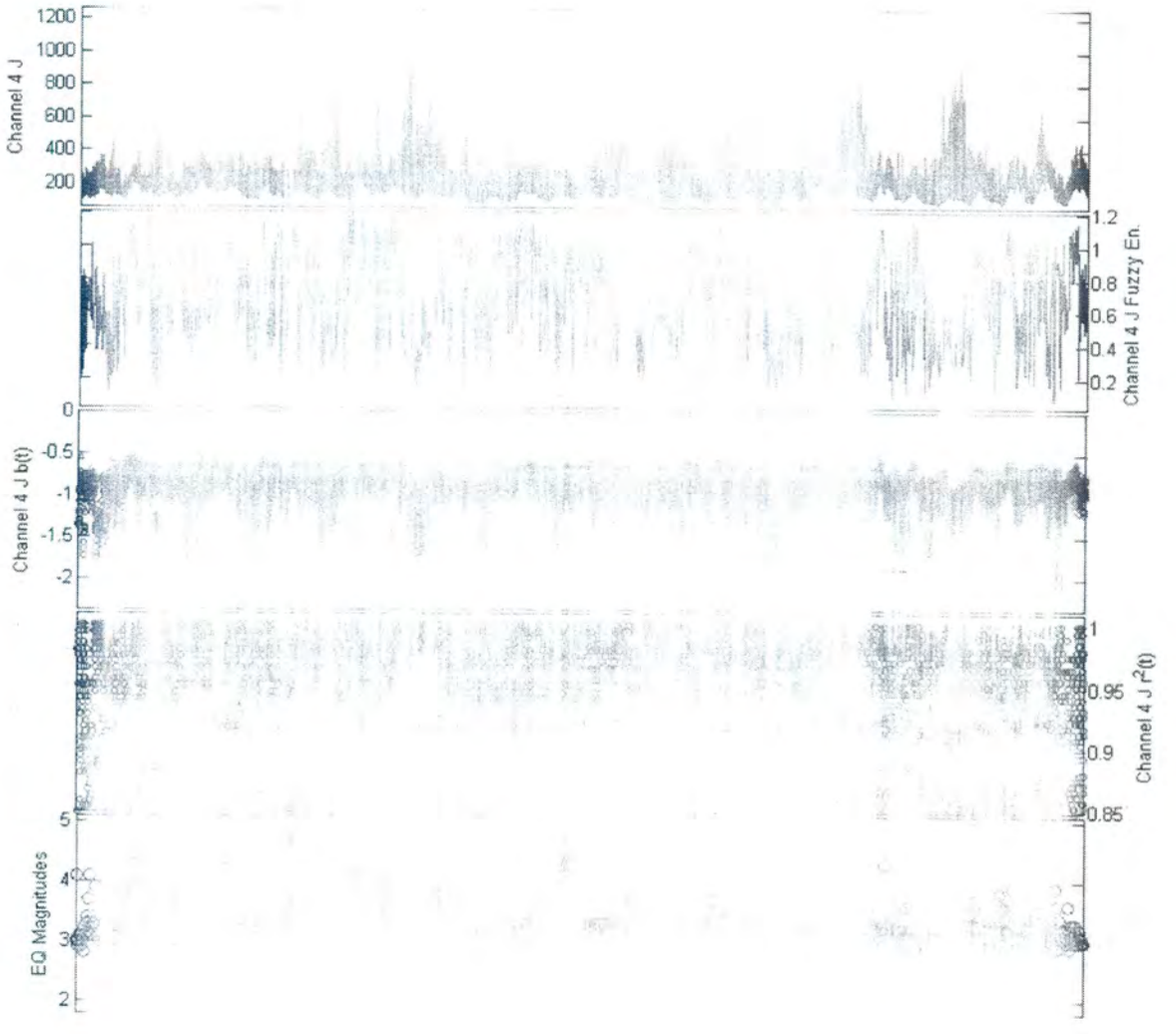

Channel 6 O
1200
1000
800
600
Channel 6 O Fuzzy En.
2.5
2
1.5
1
0.5
Channel 6 O b(t)
0
-0.5
-1
Channel 6 O r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar-03
Mar-03

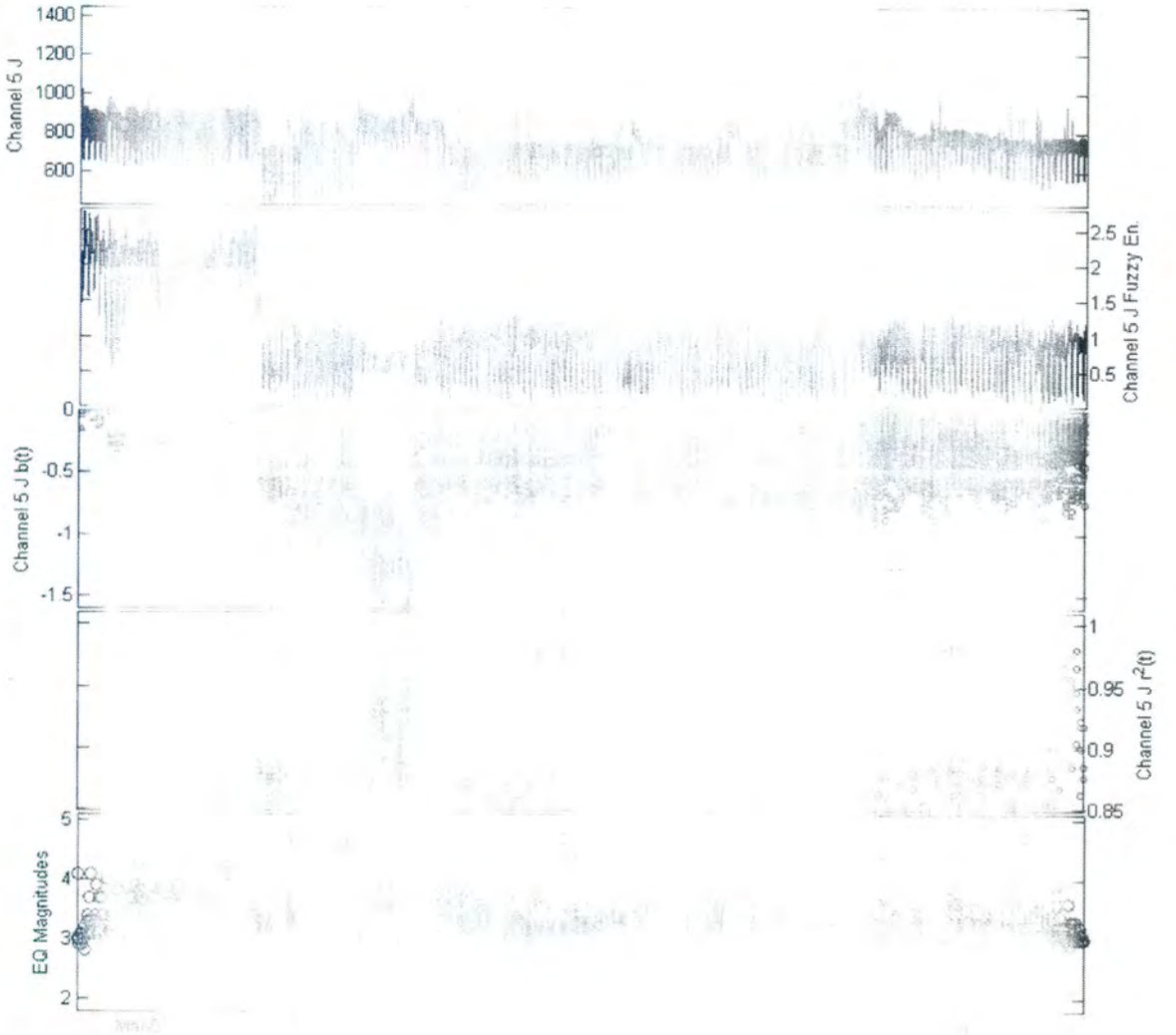

Channel 7 O
13
12
11
Channel 7 O Fuzzy En.
1.5
1
0.5
0
Channel 7 O b(t)
-0.5
-1
-1.5
-2
Channel 7 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar 3

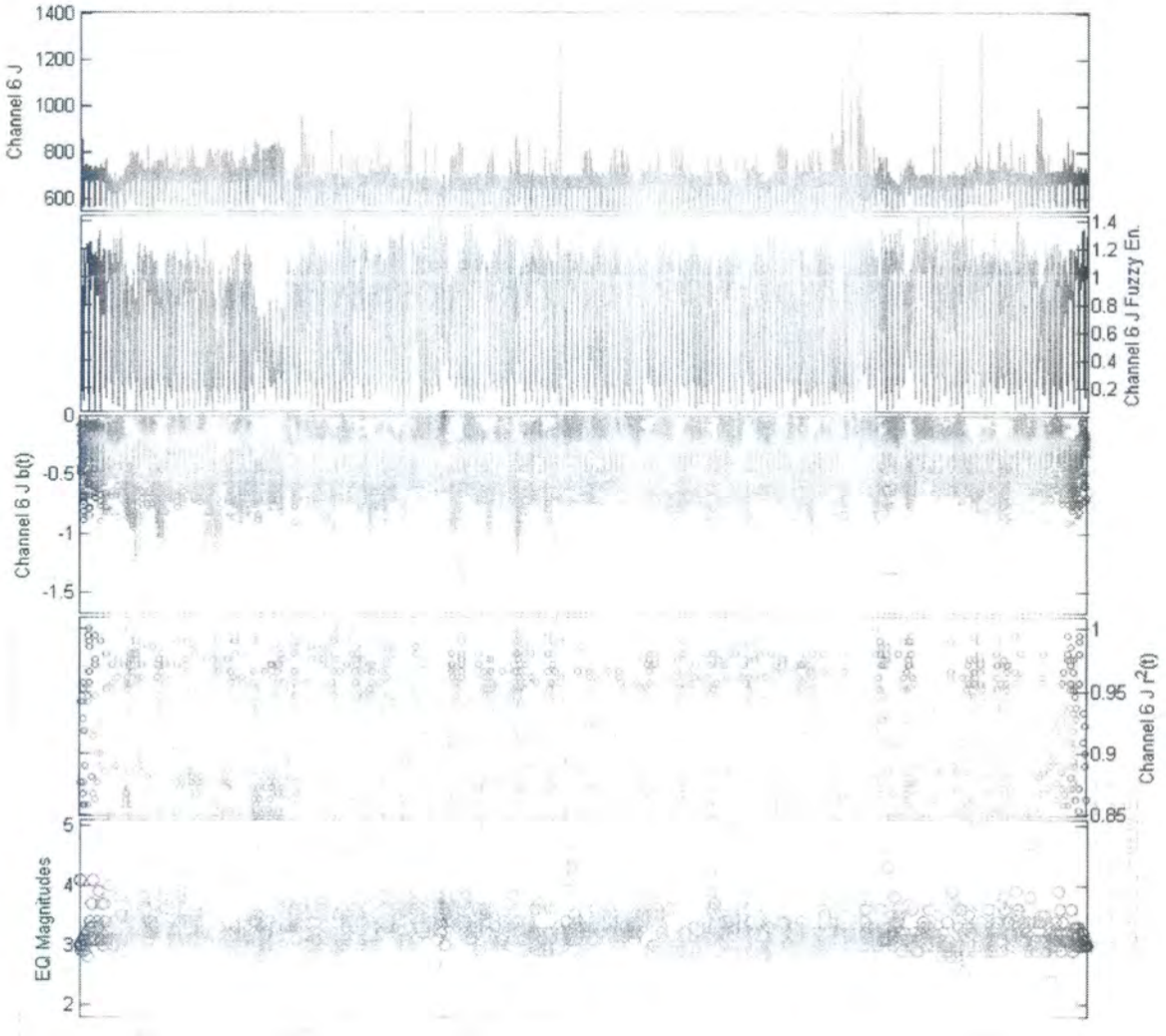

Channel 1 J
1000
500
0
Channel 1 J Fuzzy En.
0.4
0.2
0
Channel 1 J b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03
Mar-08
Mar-13
Mar-18
Mar-28

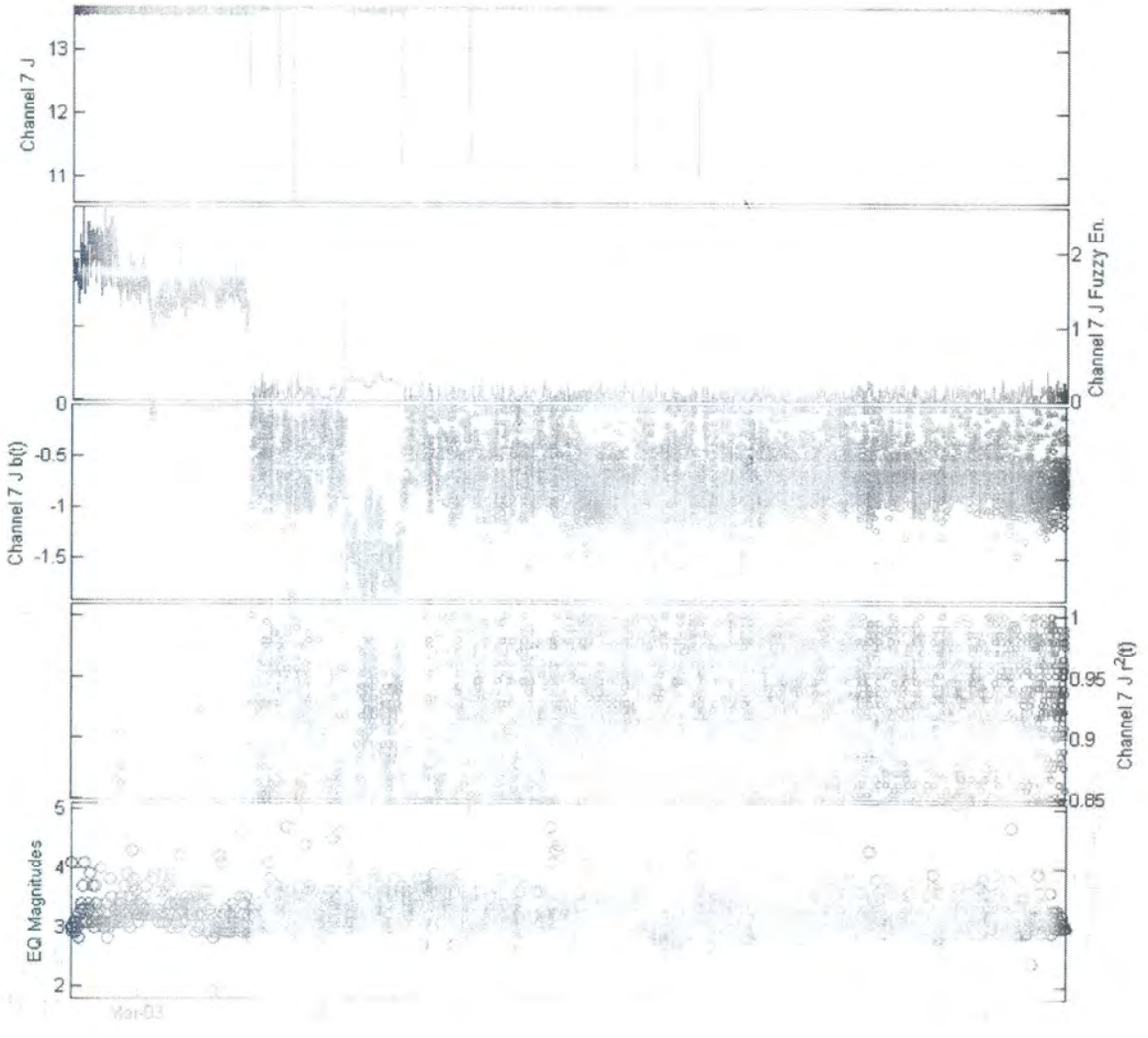

Channel 2 J
200
100
0
Channel 2 J Fuzzy En.
1
0.8
0.6
0.4
0.2
0
Channel 2 J b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

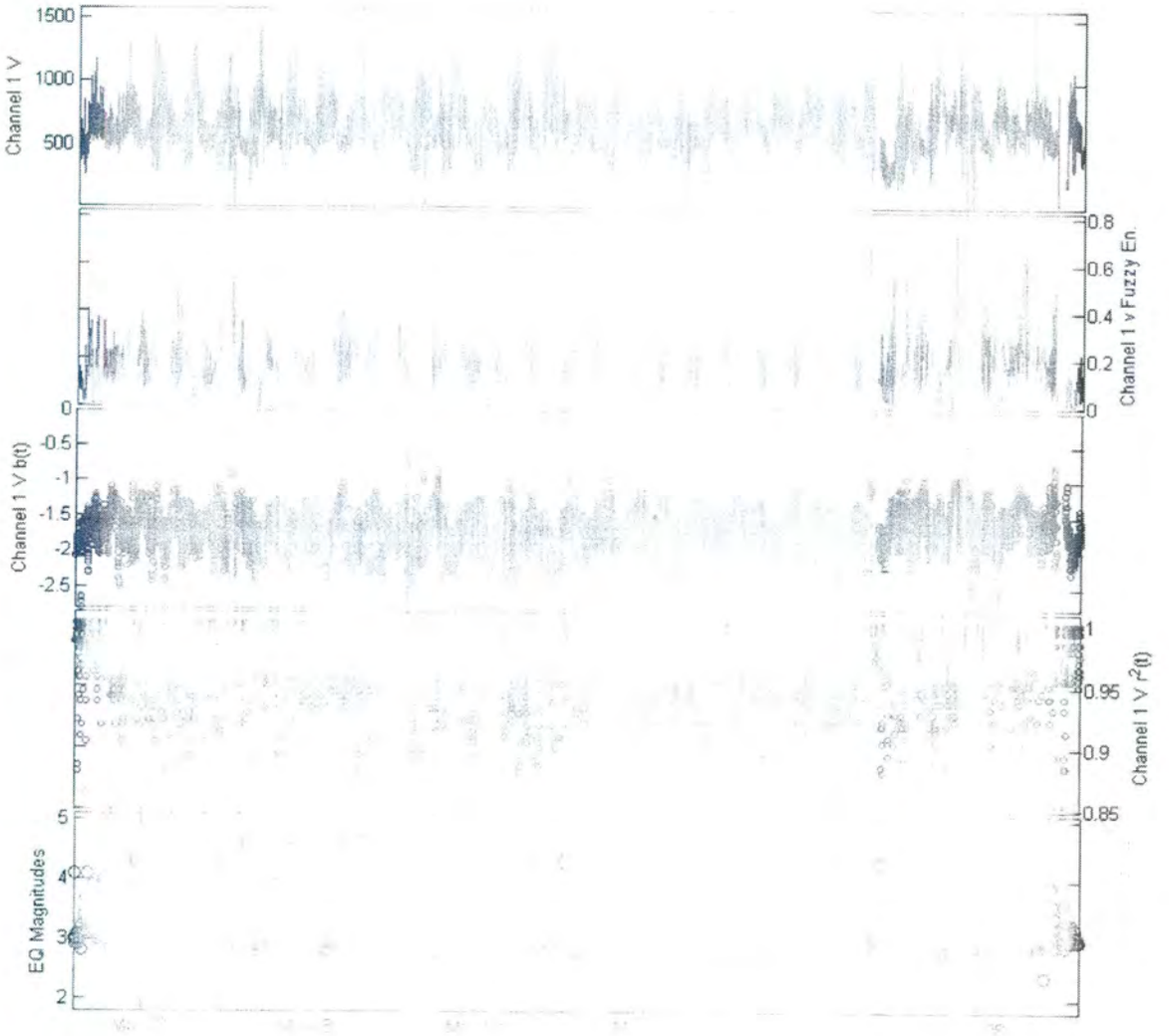

Channel 3 J
1000
800
600
400
200
Channel 3 J Fuzzy En.
1
0.5
0
Channel 3 J b(t)
-0.5
-1
-1.5
-2
Channel 3 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

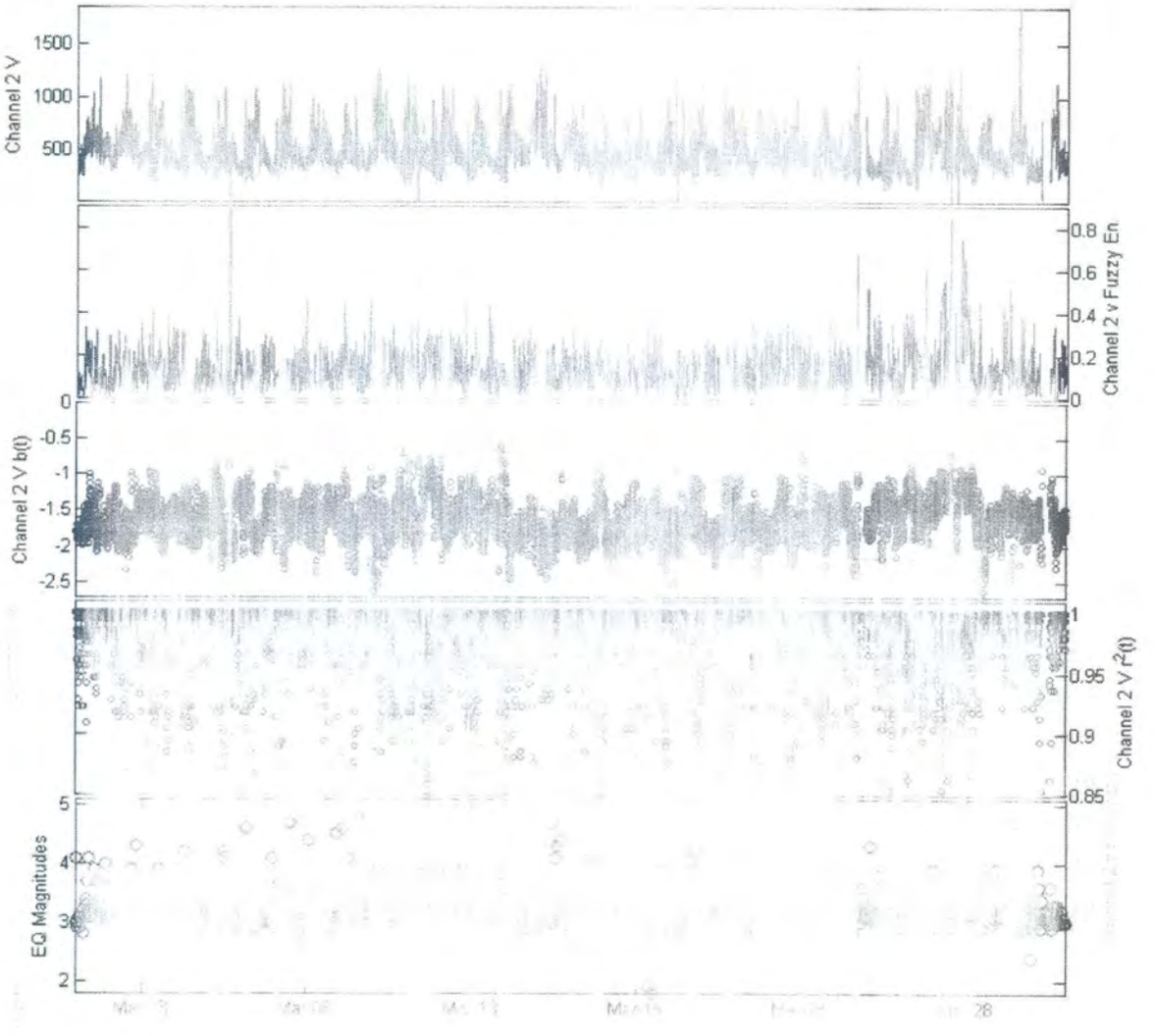

Channel 4 J
1200
1000
800
600
400
200
0
Channel 4 J Fuzzy En.
1.2
1
0.8
0.6
0.4
0.2
Channel 4 J b(t)
-0.5
-1
-1.5
-2
Channel 4 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

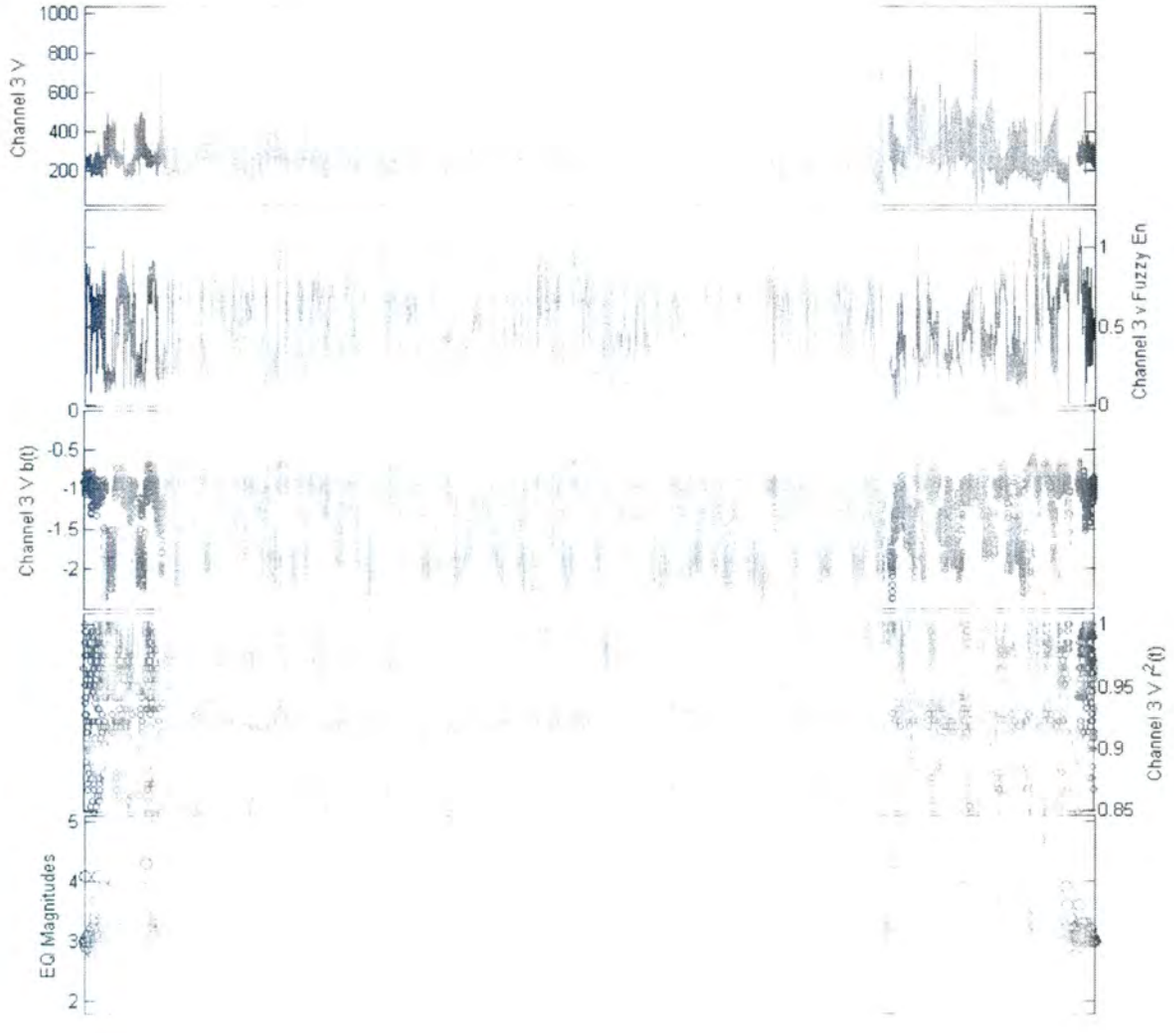

Channel 5 J
1400
1200
1000
800
600
Channel 5 J b(t)
0
-0.5
-1
-1.5
Channel 5 J Fuzzy En.
2.5
2
1.5
1
0.5
Channel 5 J r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

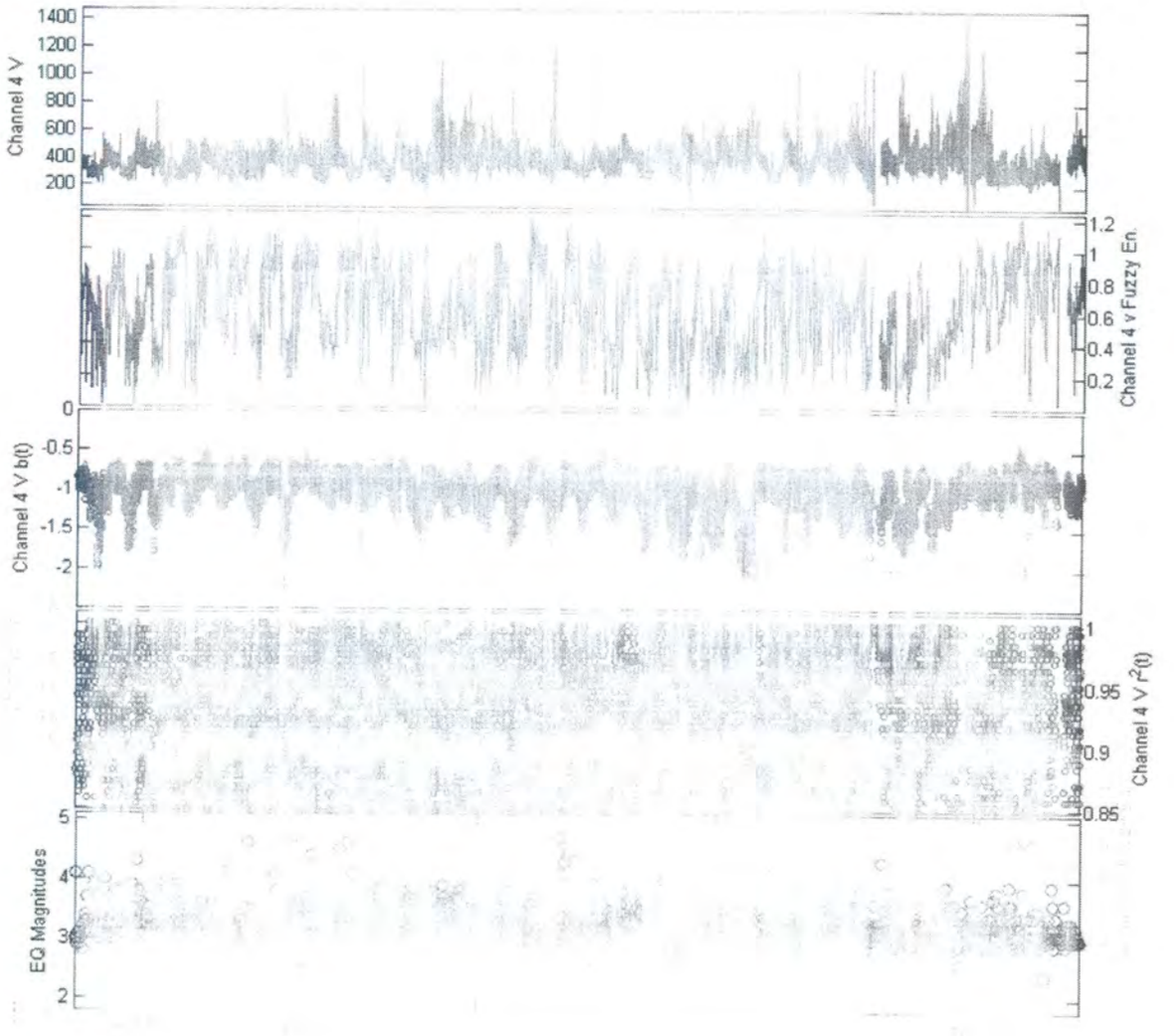

Channel 6 J
1400
1200
1000
800
600
Channel 6 J Fuzzy En.
1.4
1.2
1
0.8
0.6
0.4
0.2
Channel 6 J b(t)
0
-0.5
-1
-1.5
Channel 6 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

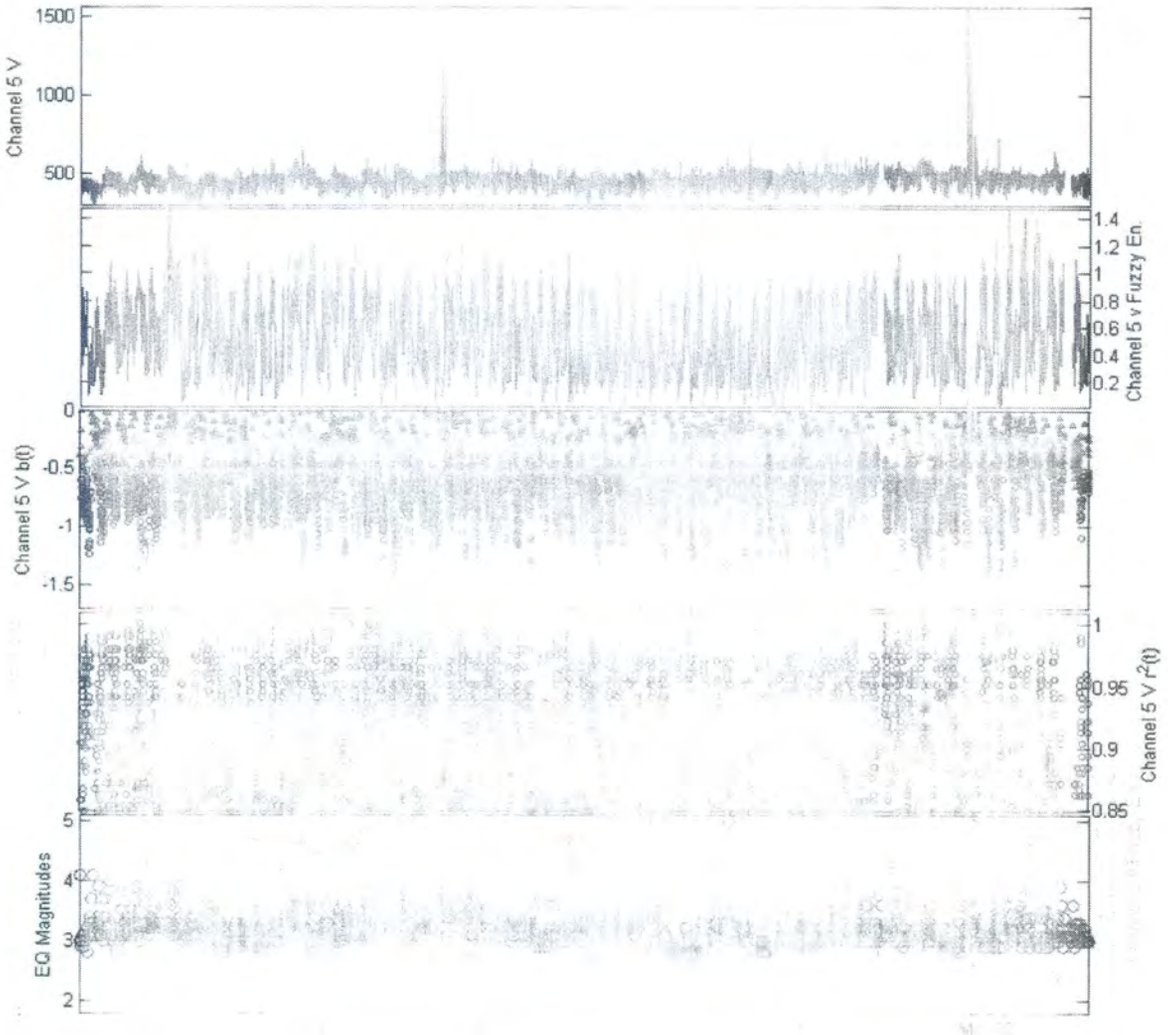

Channel 7 J
13
12
11
Channel 7 J Fuzzy En.
2
1
0
Channel 7 J b(t)
0
-0.5
-1
-1.5
Channel 7 J r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
Mar-03

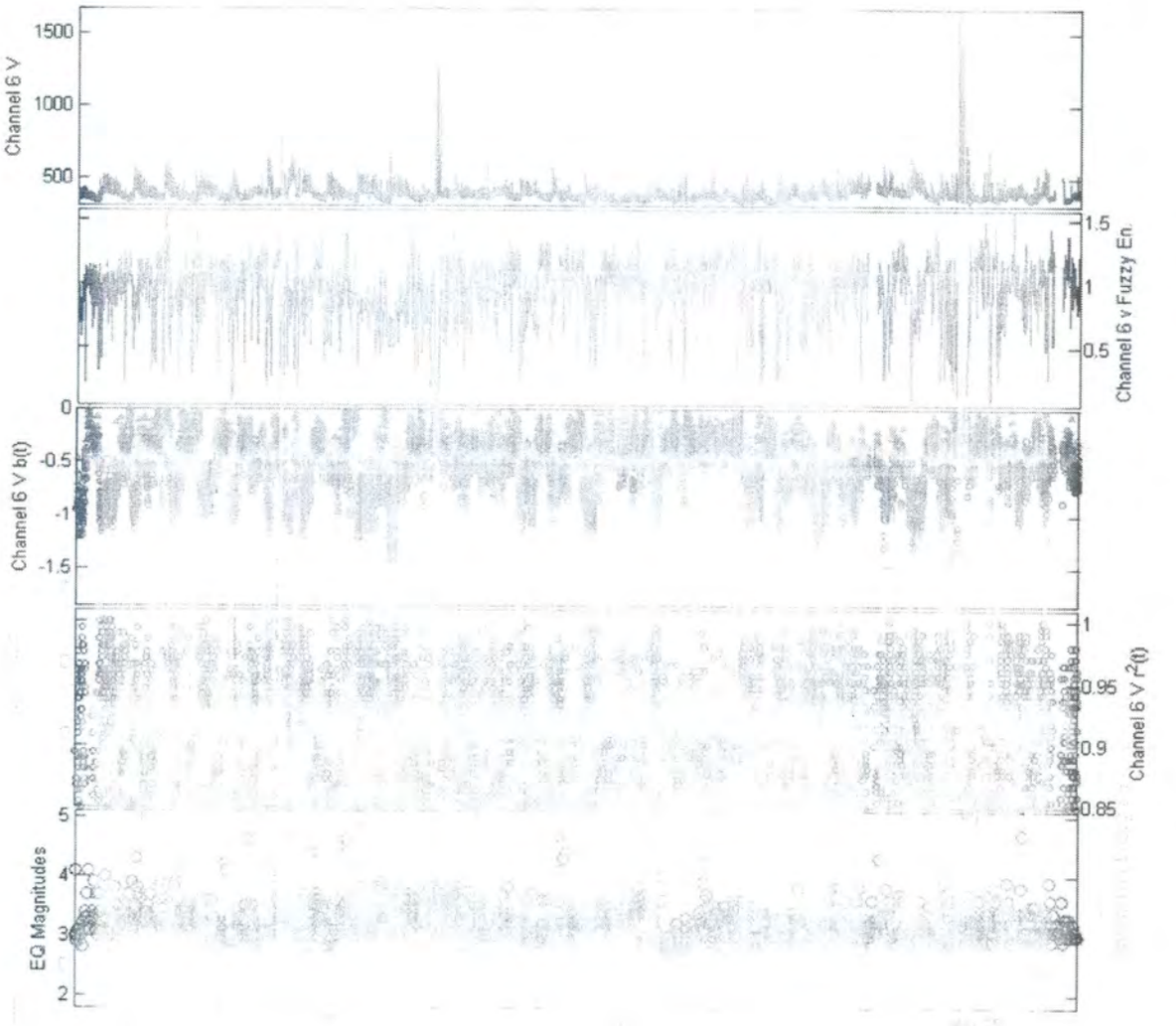

Channel 1 V
1500
1000
500
0
Channel 1 v Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 1 V b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 V ^{2}(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

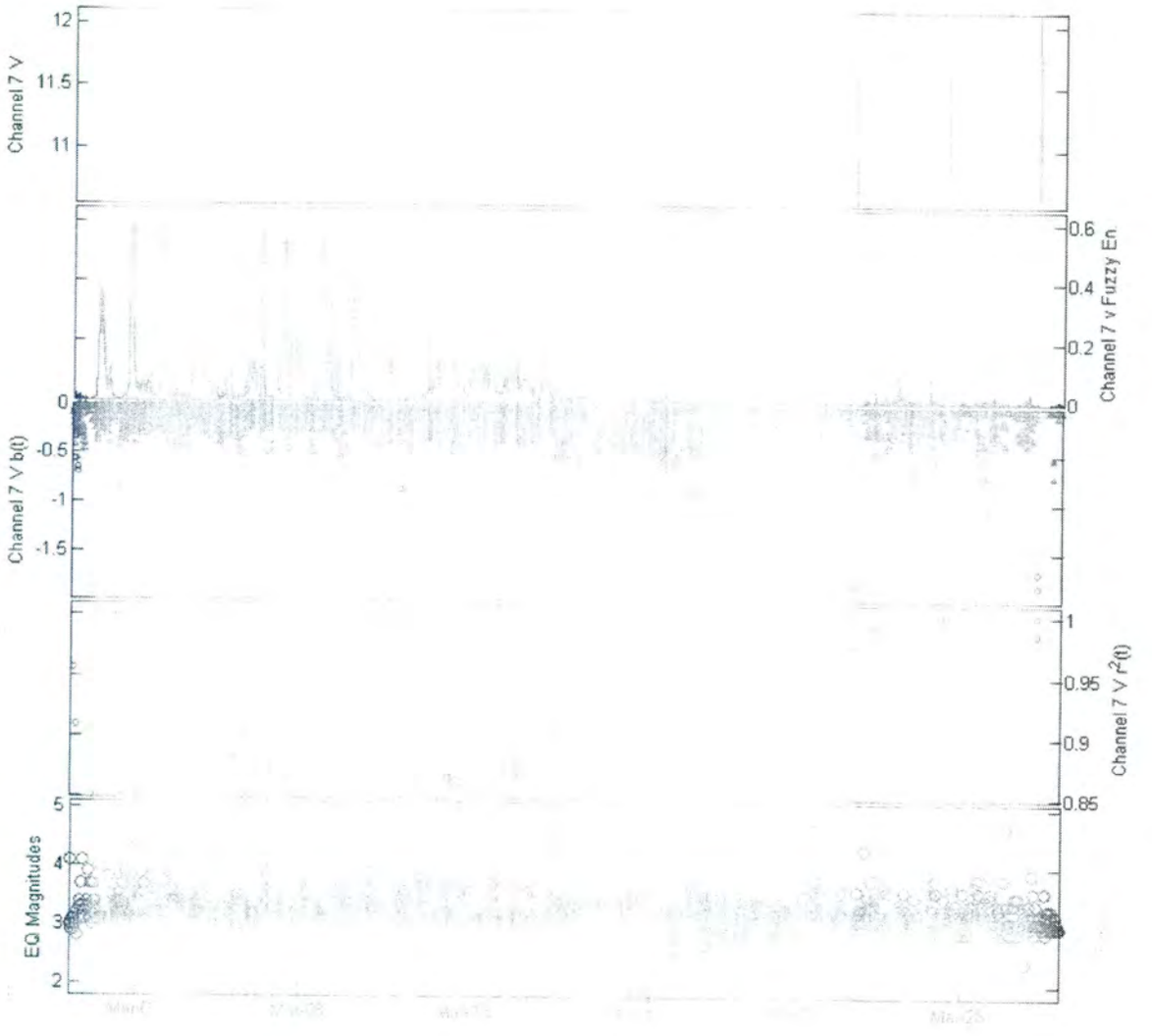

Channel 2 V
1500
1000
500
Channel 2 v Fuzzy En
0.8
0.6
0.4
0.2
0
Channel 2 V b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 V r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2
28

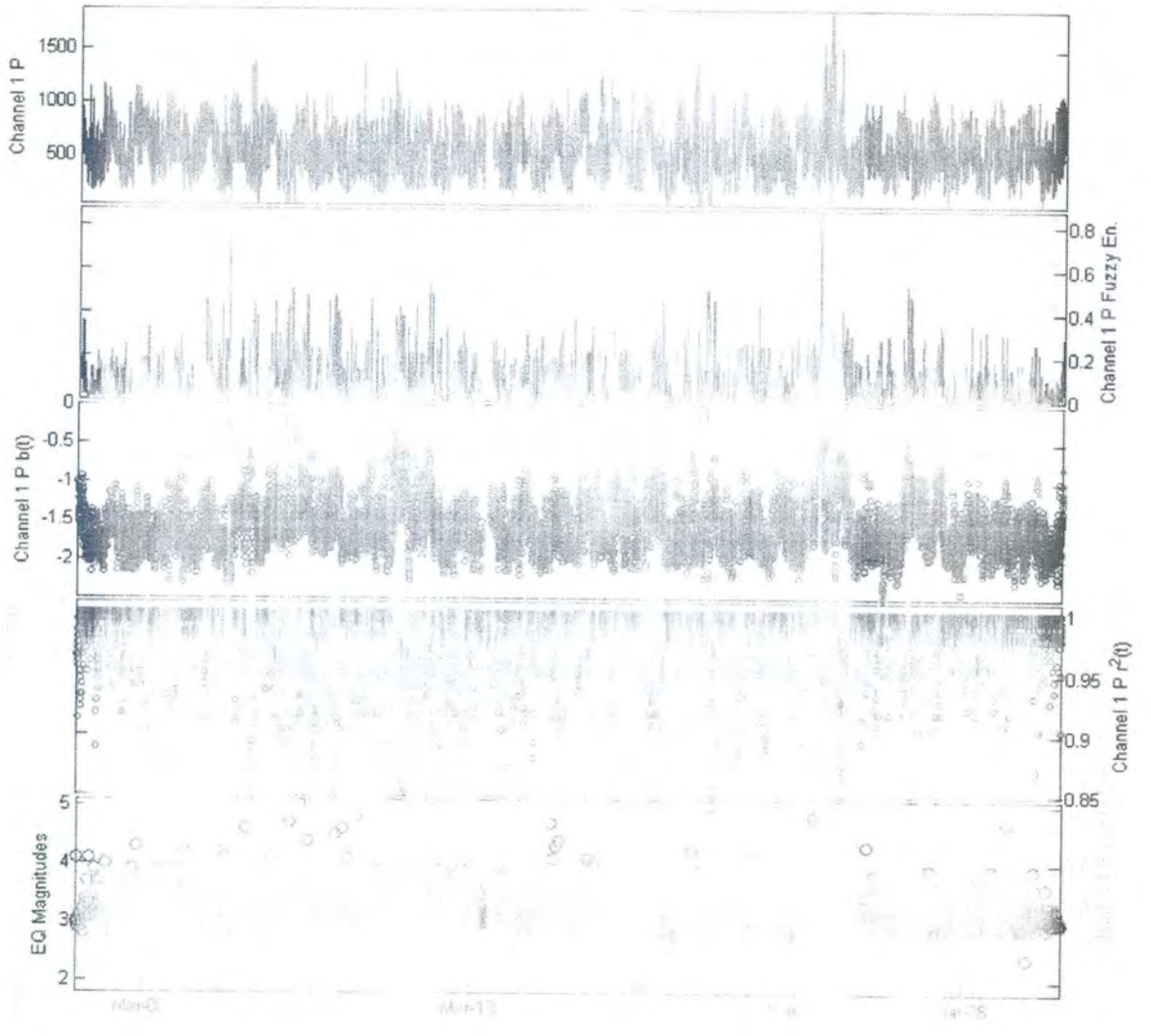

Channel 3 V
1000
800
600
400
200
Channel 3 v Fuzzy En
1
0.5
0
Channel 3 V b(t)
-0.5
-1
-1.5
-2
Channel 3 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

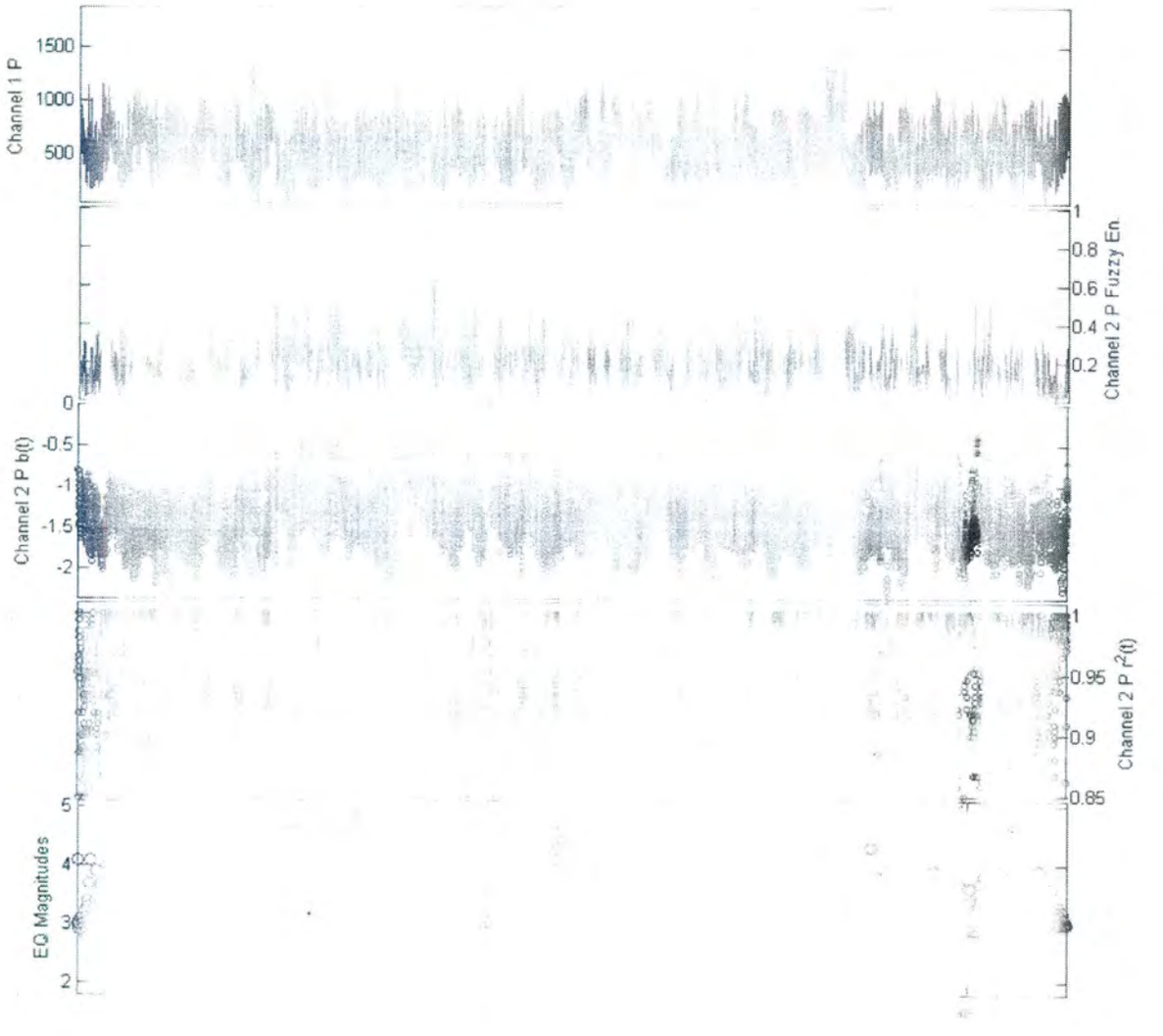

Channel 4 V
1400
1200
1000
800
600
400
200
0
Channel 4 v Fuzzy En.
1.2
1
0.8
0.6
0.4
0.2
Channel 4 V b(t)
-0.5
-1
-1.5
-2
Channel 4 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

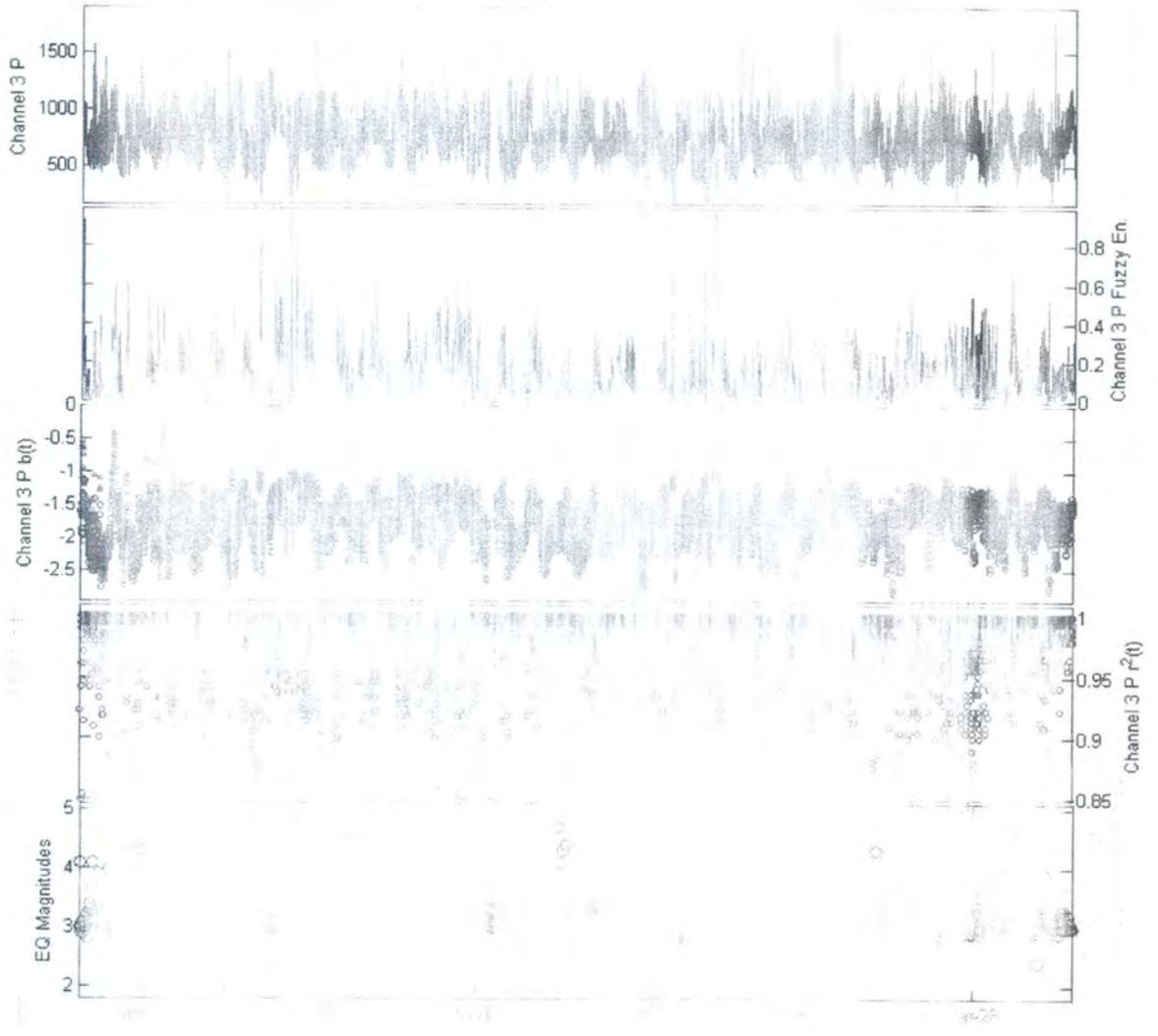

Channel 5 V
1500
1000
500
Channel 5 v Fuzzy En.
1.4
1.2
1
0.8
0.6
0.4
0.2
0
Channel 5 V b(t)
-0.5
-1
-1.5
Channel 5 V r^2(t)
1
0.95
0.9
0.85
5
EQ Magnitudes
4
3
2

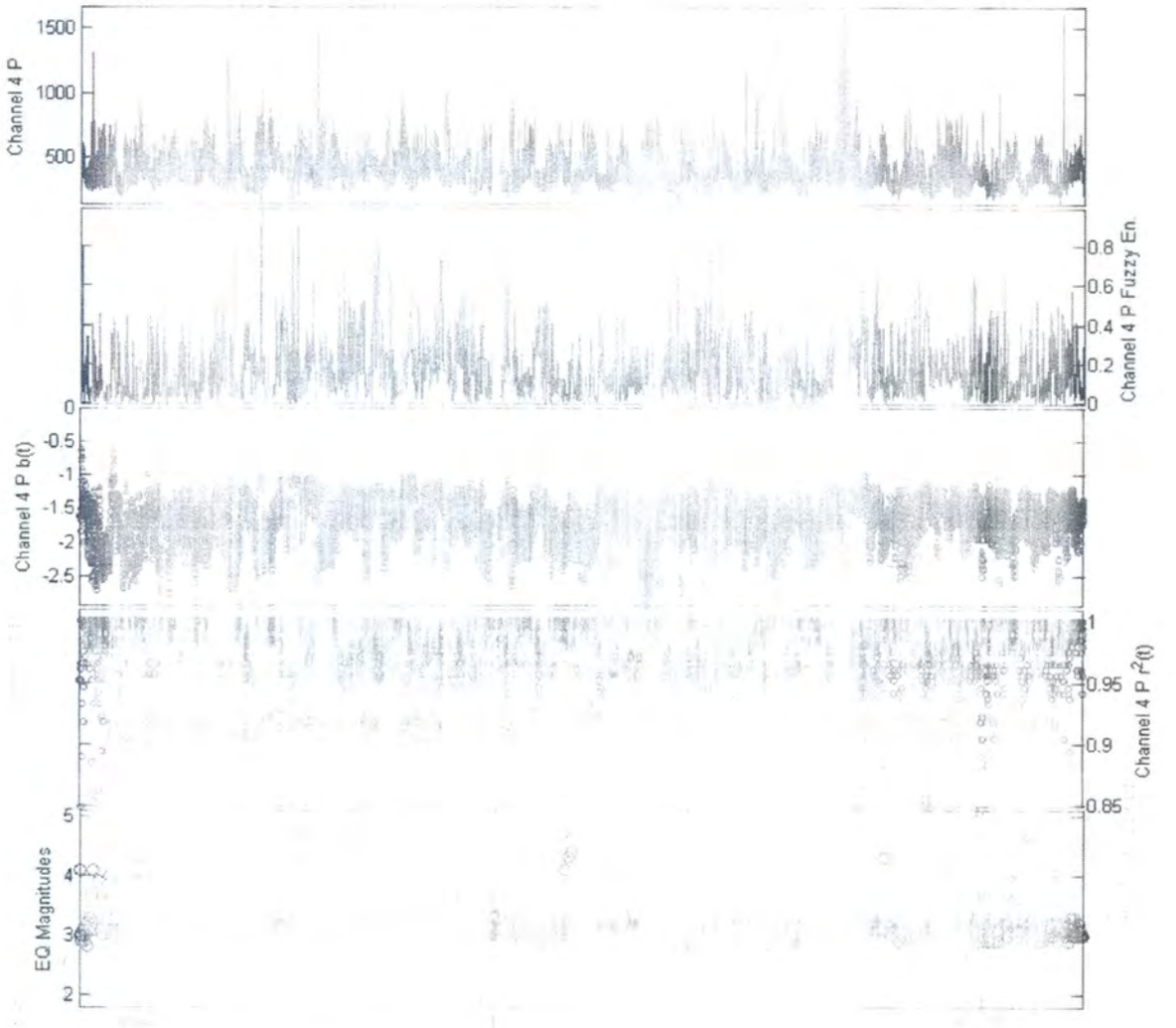

Channel 6 V
1500
1000
500
Channel 6 v Fuzzy En.
1.5
1
0.5
0
Channel 6 V b(t)
-0.5
-1
-1.5
Channel 6 V r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

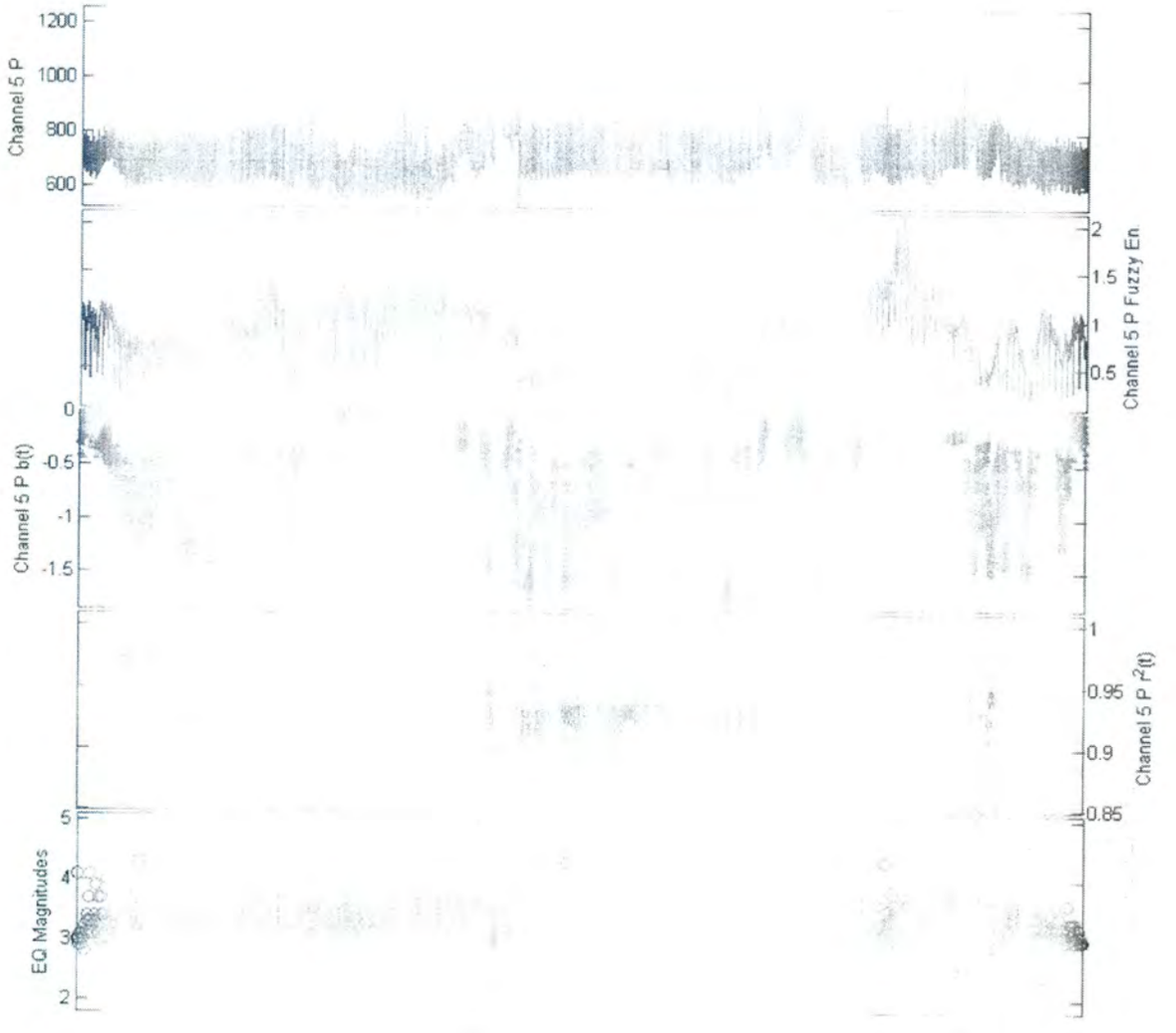

Channel 7 V
12
11.5
11
Channel 7 V b(t)
0
-0.5
-1
-1.5
Channel 7 v Fuzzy En
0.6
0.4
0.2
0
Channel 7 V r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

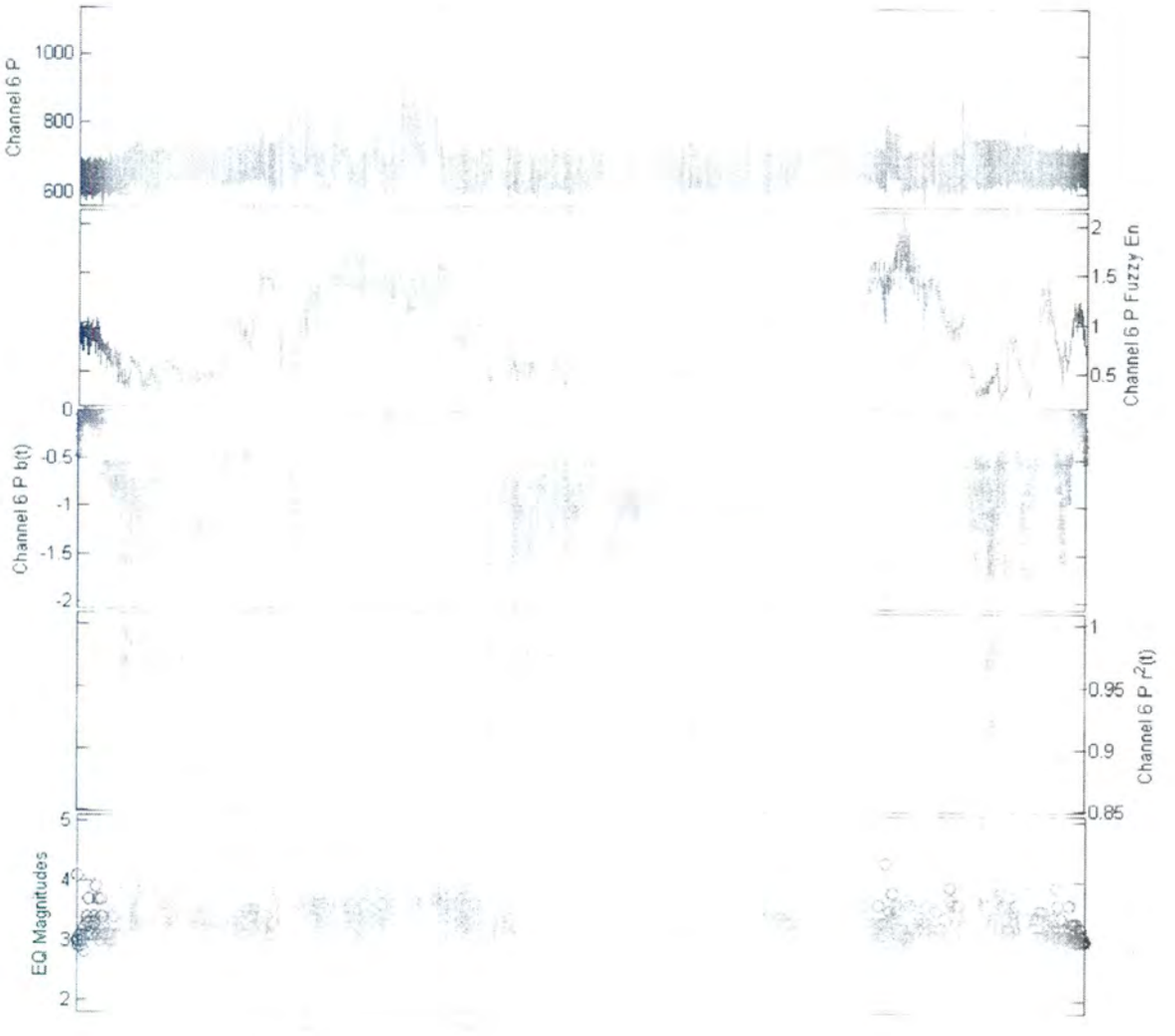

Channel 1 P
1500
1000
500
0
0.8
0.6
0.4
0.2
0
Channel 1 P Fuzzy En.
Channel 1 P b(t)
-0.5
-1
-1.5
-2
1
0.95
0.9
0.85
Channel 1 P r²(t)
EQ Magnitudes
5
4
3
2
Mar-02
Mar-13
Mar-25

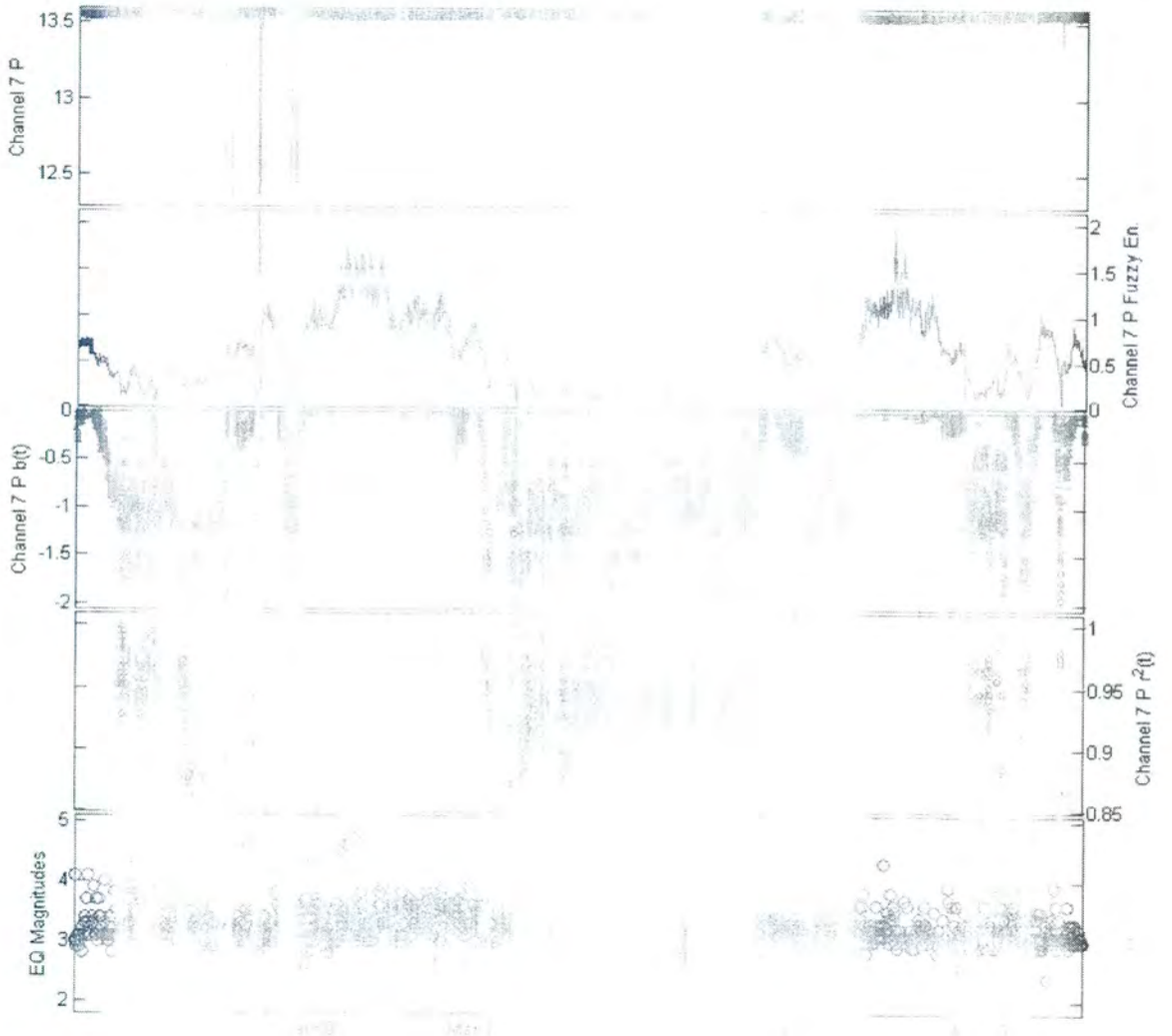

Channel 1 P
1500
1000
500
Channel 2 P Fuzzy En.
1
0.8
0.6
0.4
0.2
0
Channel 2 P b(t)
-0.5
-1
-1.5
-2
Channel 2 P r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

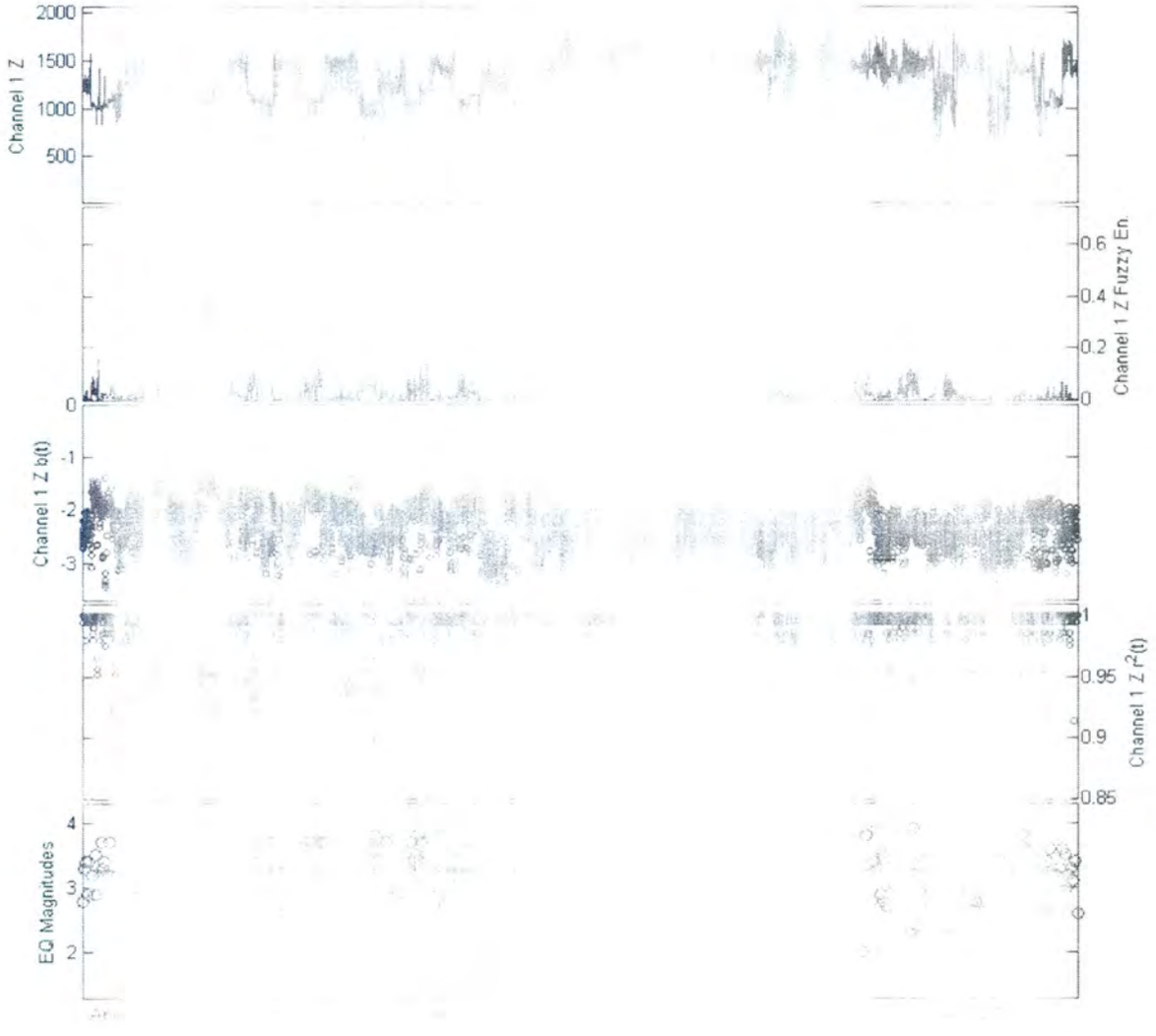

Channel 3 P
1500
1000
500
0
Channel 3 P Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 3 P b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 3 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

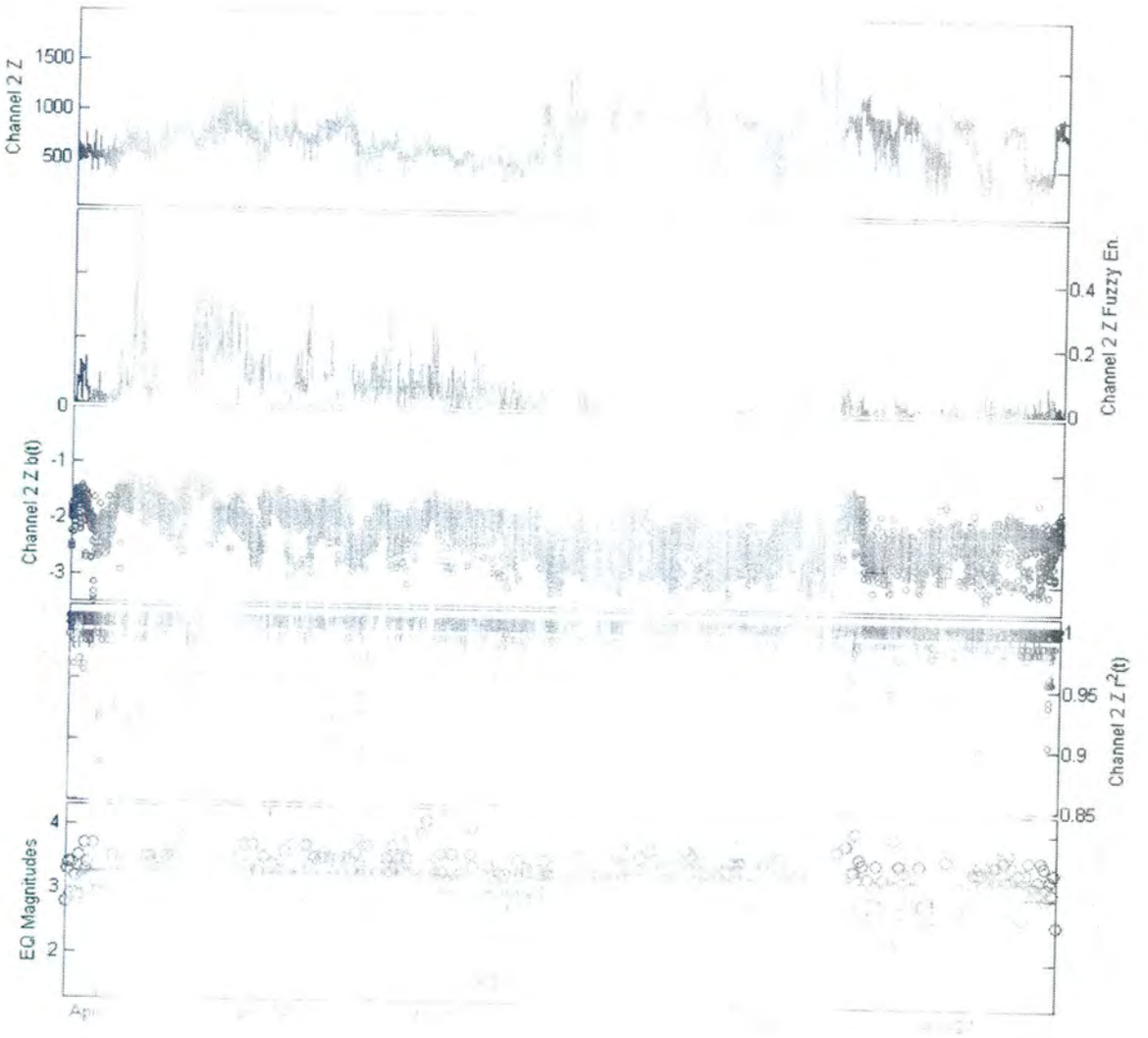

Channel 4 P
1500
1000
500
0
Channel 4 P Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 4 P b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

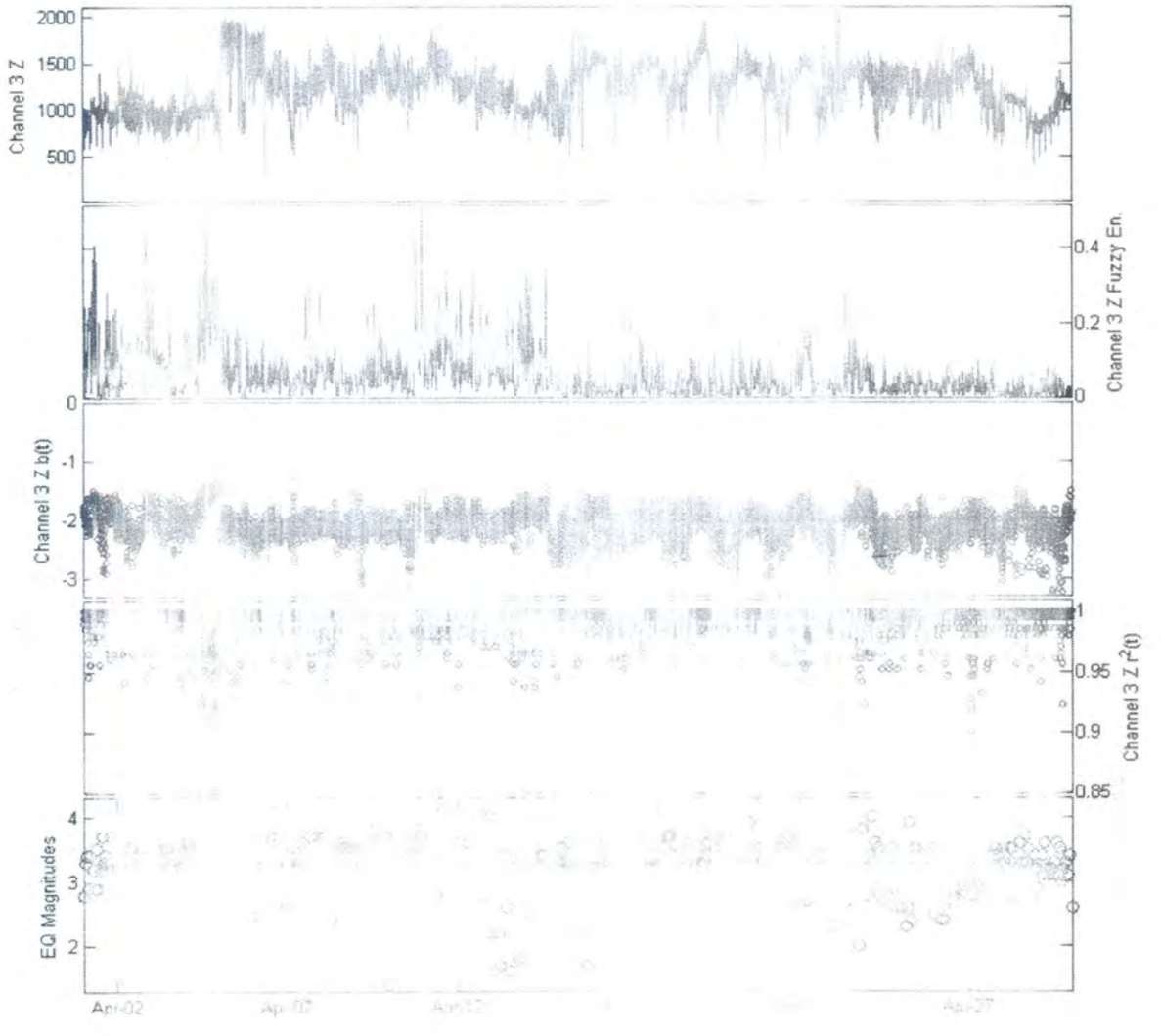

Channel 5 P
1200
1000
800
600
Channel 5 P Fuzzy En
2
1.5
1
0.5
Channel 5 P b(t)
0
-0.5
-1
-1.5
Channel 5 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

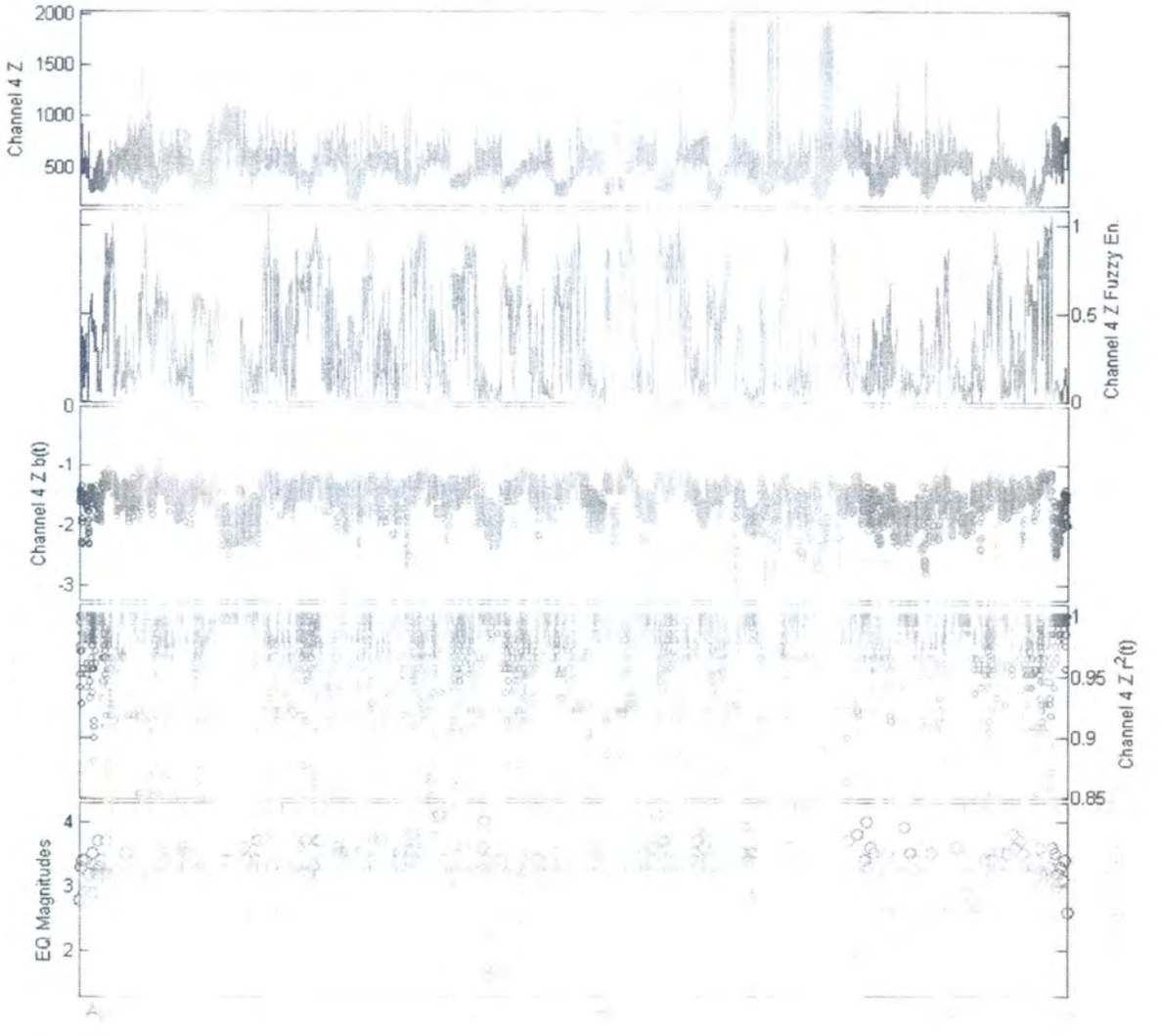

Channel 6 P
1000
800
600
Channel 6 P Fuzzy En
2
1.5
1
0.5
Channel 6 P b(t)
0
-0.5
-1
-1.5
-2
Channel 6 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

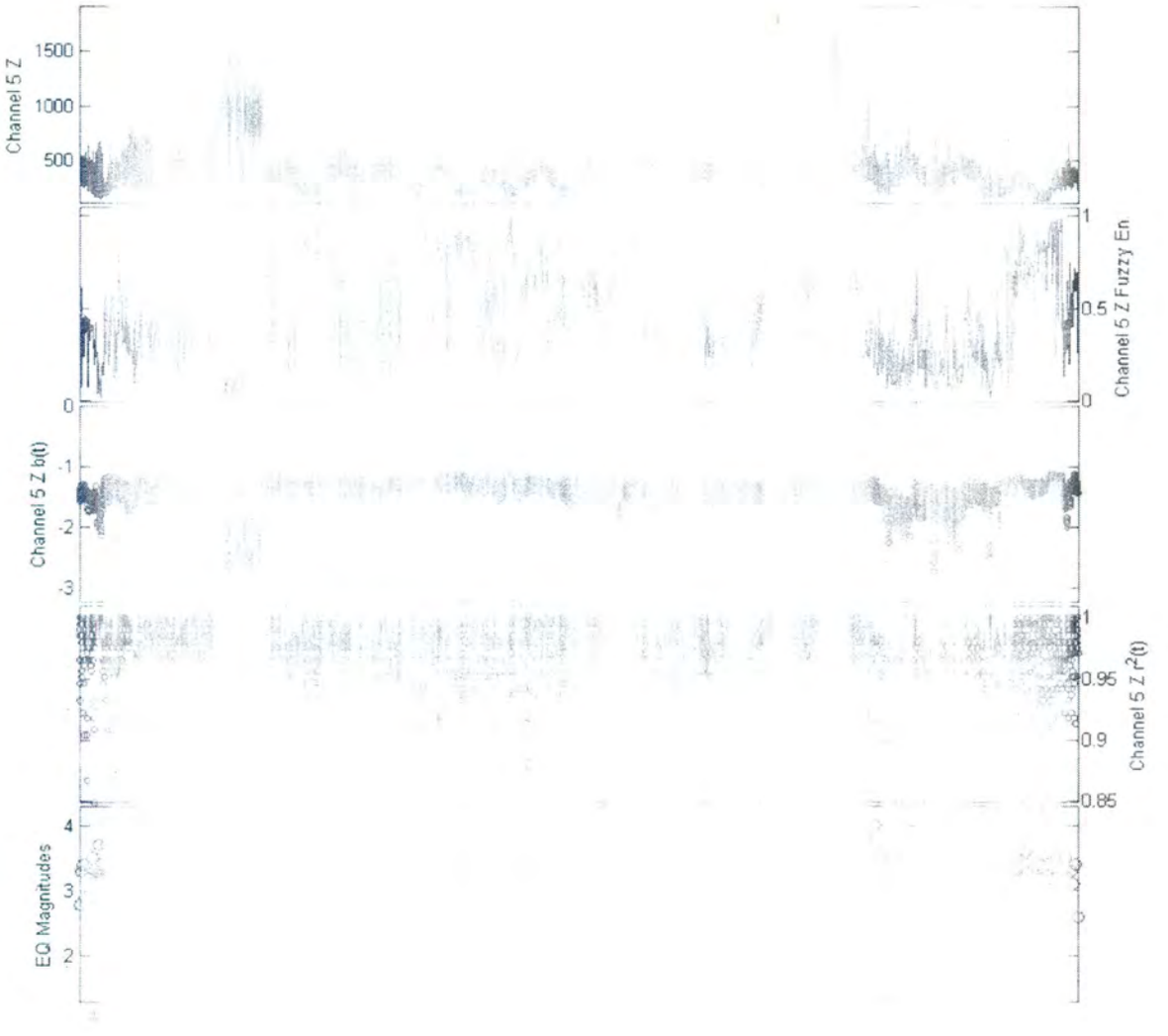

Channel 7 P
13.5
13
12.5
Channel 7 P b(t)
0
-0.5
-1
-1.5
-2
Channel 7 P Fuzzy En.
2
1.5
1
0.5
0
Channel 7 P r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
5
4
3
2

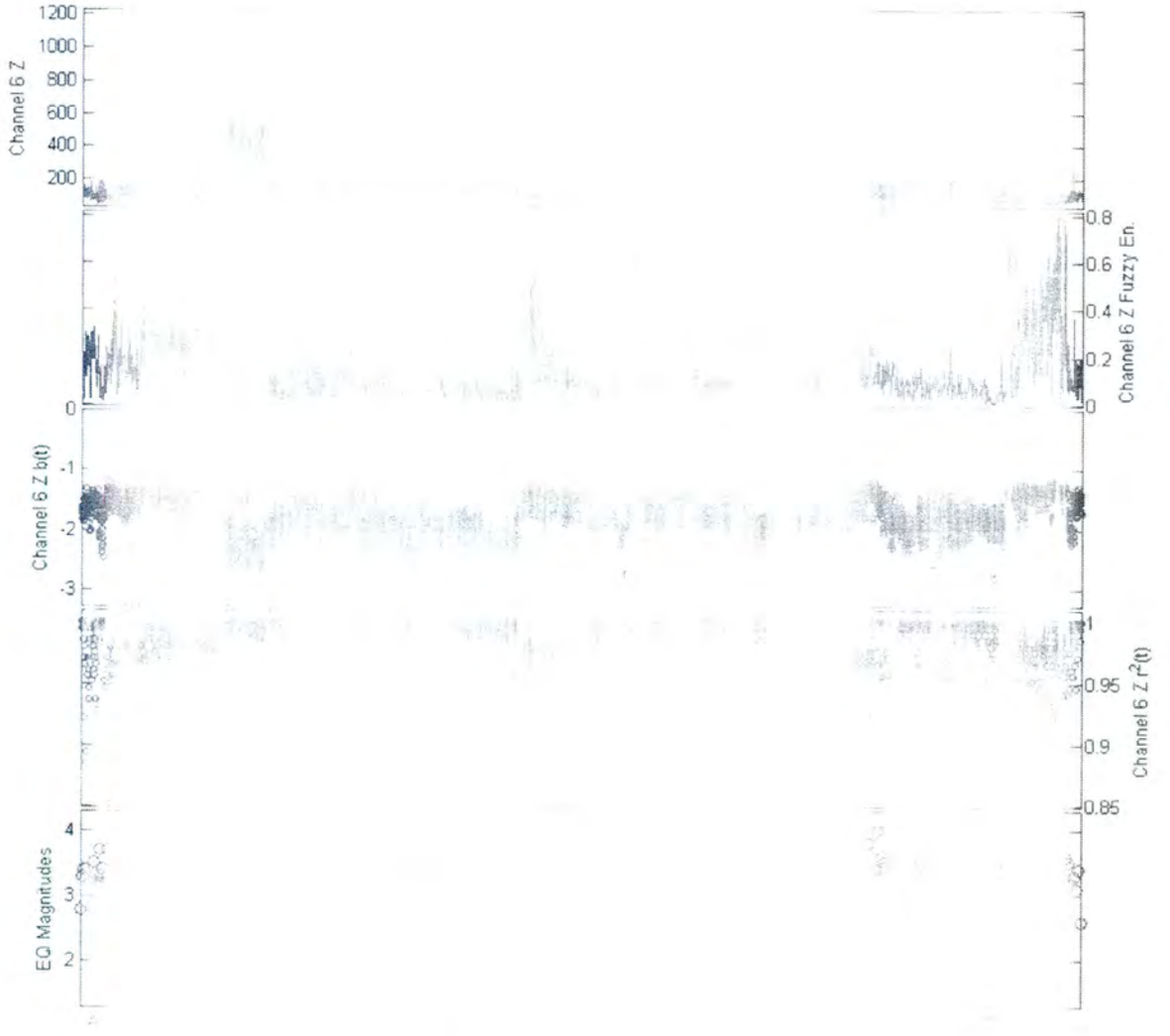

Channel 1 Z
2000
1500
1000
500
Channel 1 Z Fuzzy En
0.6
0.4
0.2
0
Channel 1 Z b(t)
-1
-2
-3
Channel 1 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2

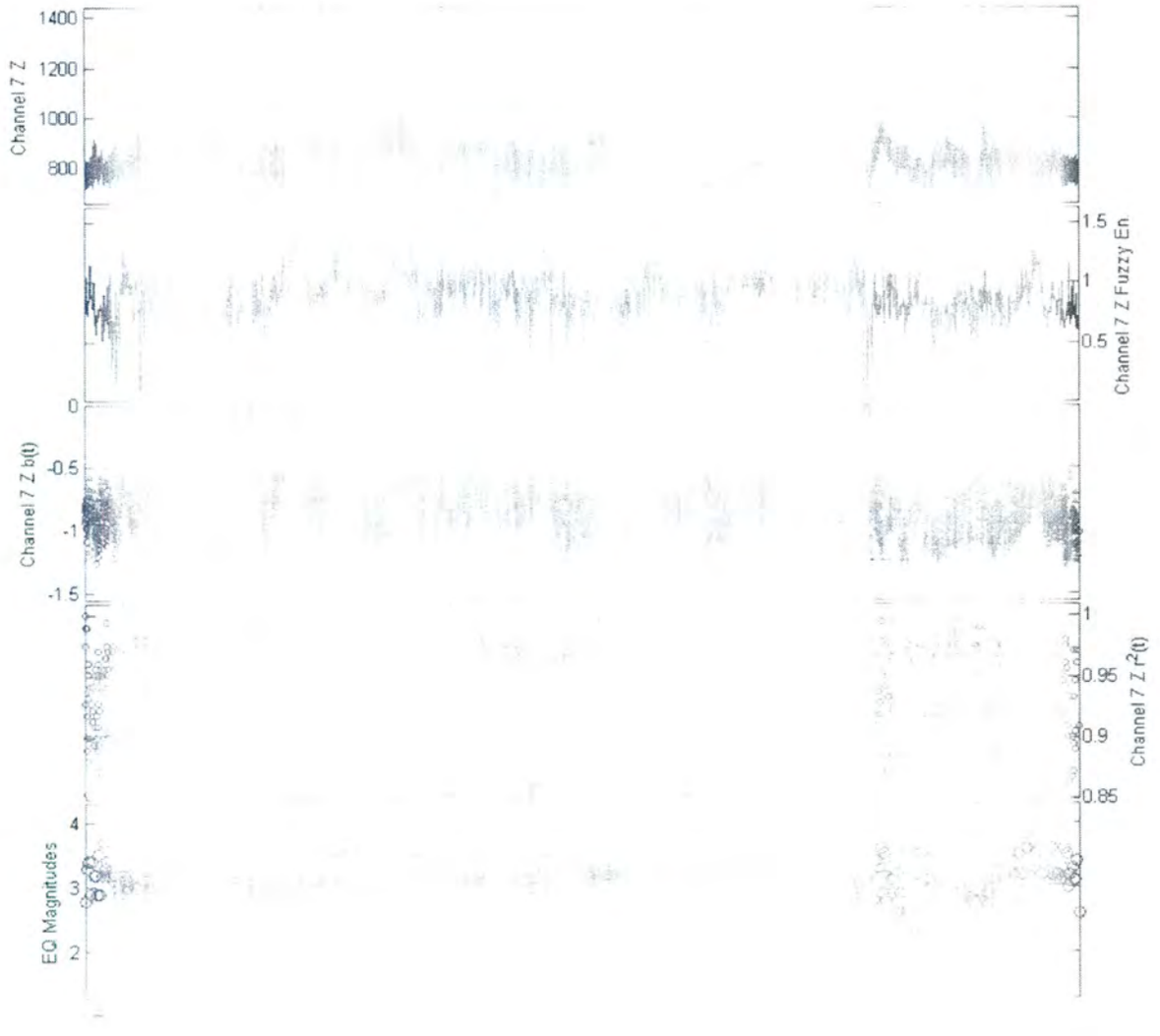

Channel 2 Z
1500
1000
500
0
Channel 2 Z Fuzzy En.
0.4
0.2
0
Channel 2 Z b(t)
-1
-2
-3
Channel 2 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2
Apr

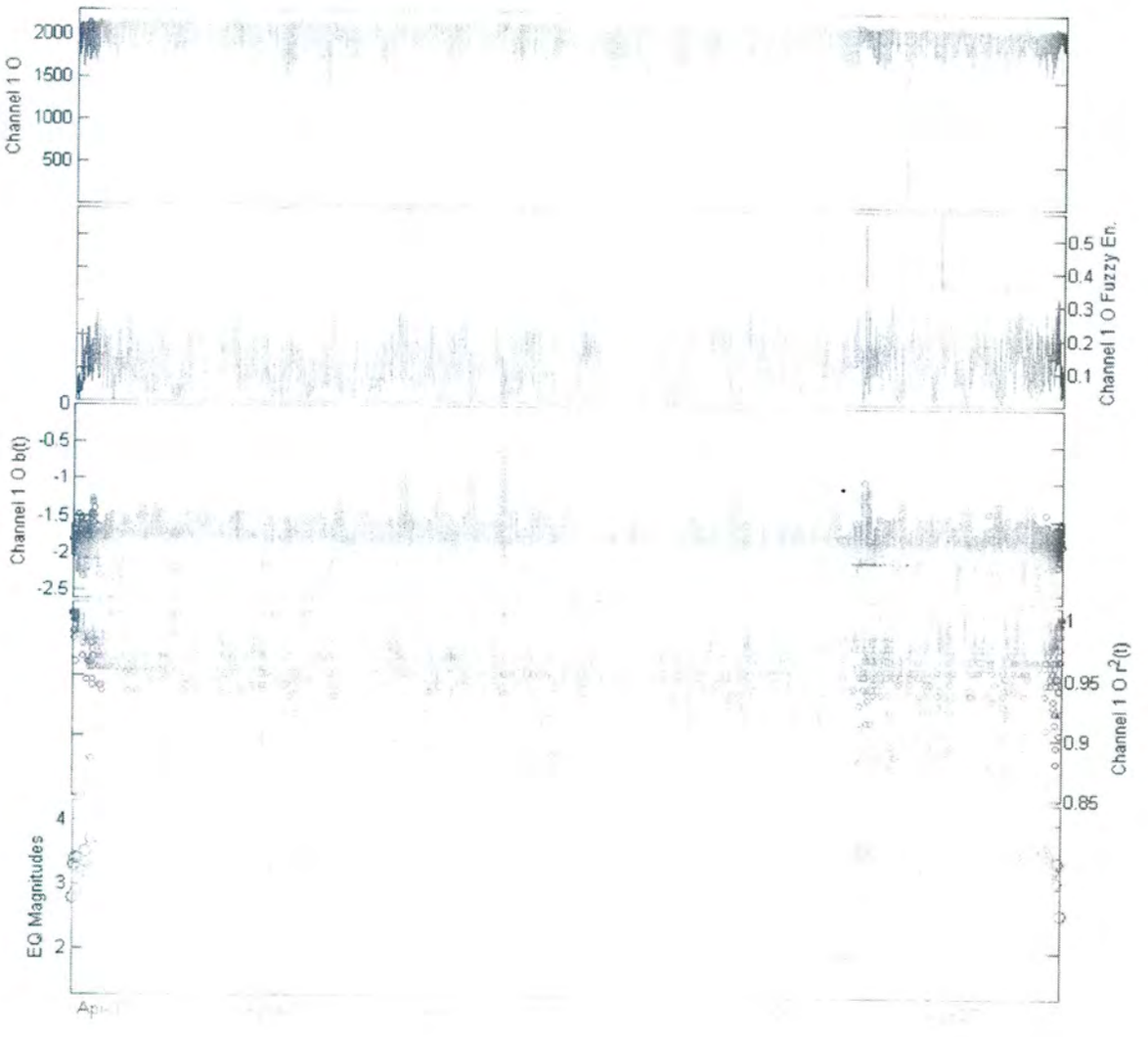

Channel 3 Z
2000
1500
1000
500
Channel 3 Z Fuzzy En.
0.4
0.2
0
Channel 3 Z b(t)
-1
-2
-3
Channel 3 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2
Apr-02
Apr-07
Apr-12
Apr-27

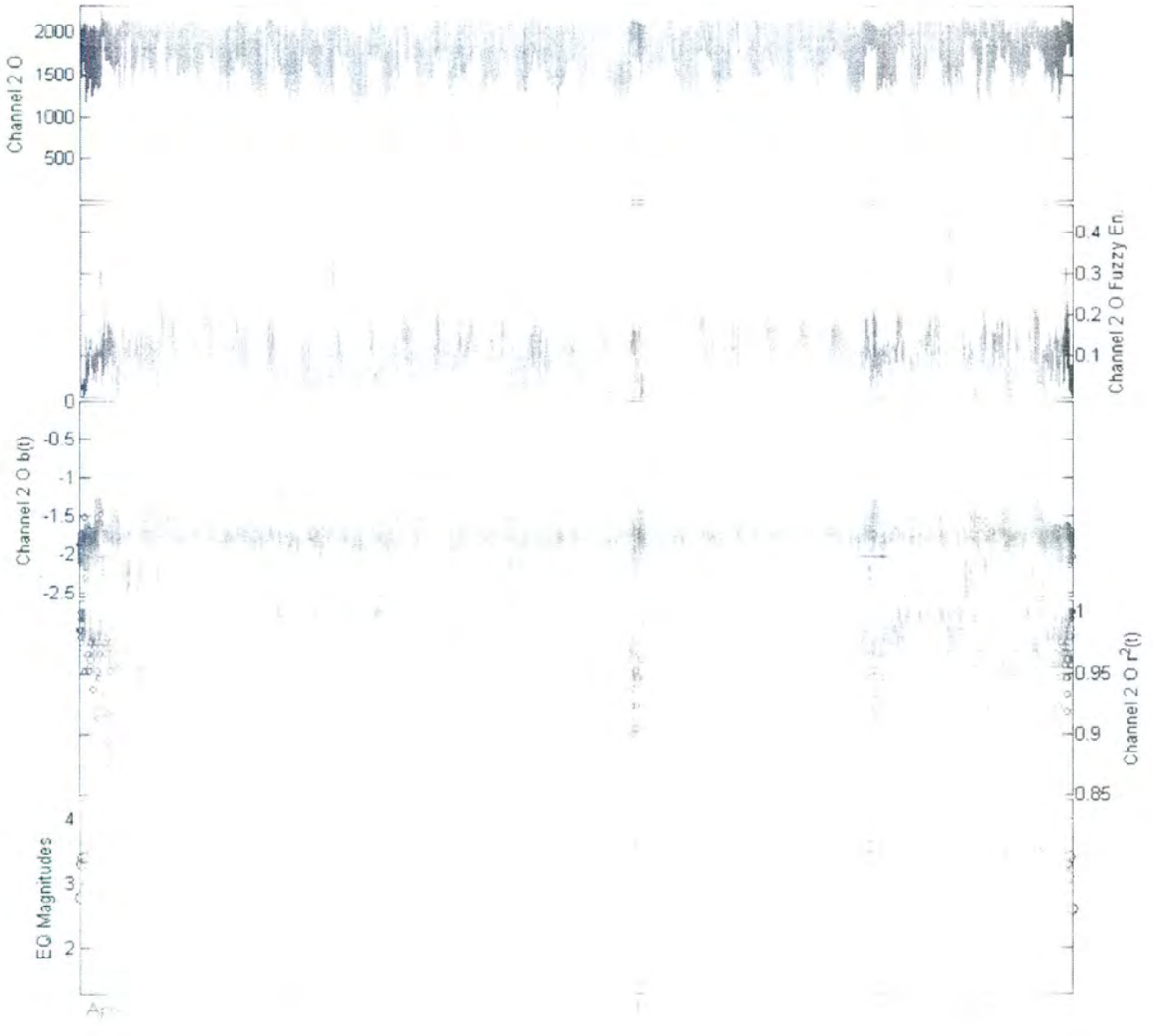

Channel 4 Z
2000
1500
1000
500
Channel 4 Z Fuzzy En.
1
0.5
0
0
Channel 4 Z b(t)
-1
-2
-3
Channel 4 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2

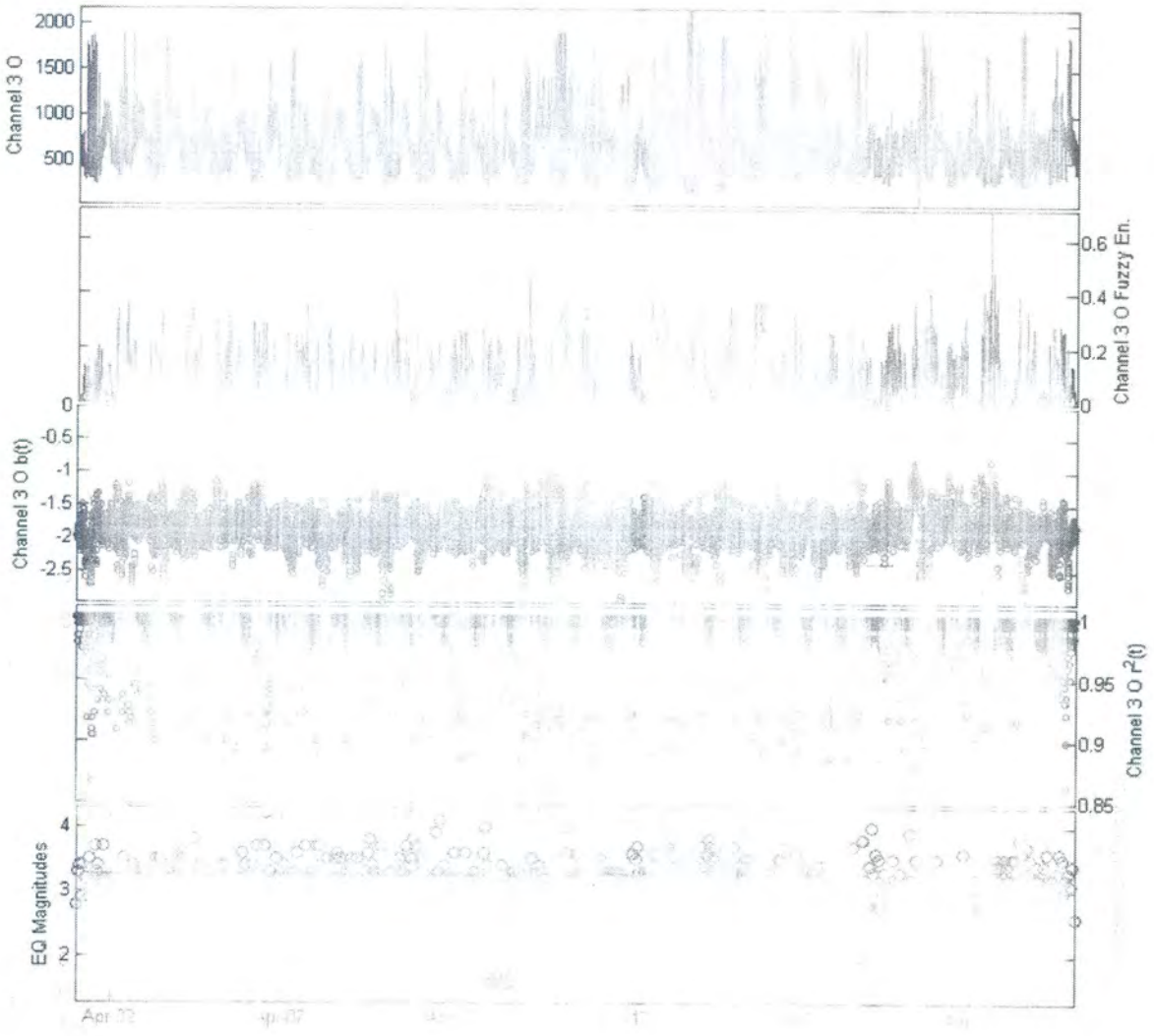

Channel 5 Z
1500
1000
500
Channel 5 Z Fuzzy En
1
0.5
0
Channel 5 Z b(t)
-1
-2
-3
Channel 5 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2

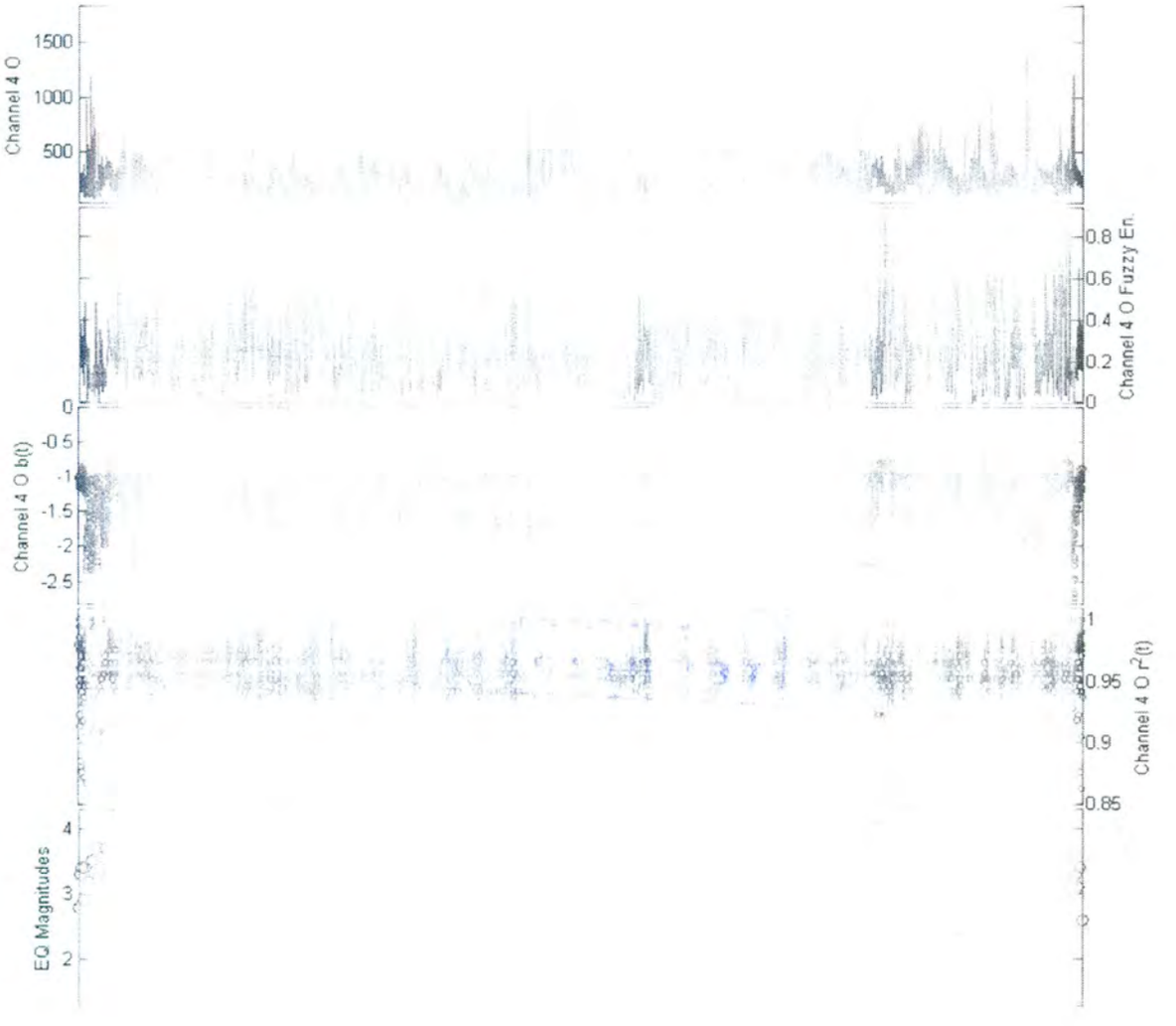

Channel 6 Z
1200
1000
800
600
400
200
0
Channel 6 Z Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 6 Z b(t)
-1
-2
-3
Channel 6 Z r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2


Channel 7 Z
1400
1200
1000
800
Channel 7 Z Fuzzy En
1.5
1
0.5
0
Channel 7 Z b(t)
-0.5
-1
-1.5
Channel 7 Z r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2


Channel 1 O
2000
1500
1000
500
0
Channel 1 O Fuzzy En.
0.5
0.4
0.3
0.2
0.1
Channel 1 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 1 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2
Ap


Channel 2 O
2000
1500
1000
500
0
Channel 2 O Fuzzy En
0.4
0.3
0.2
0.1
Channel 2 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 2 O r^2(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2


Channel 3 O
2000
1500
1000
500
Channel 3 O Fuzzy En.
0.6
0.4
0.2
0
Channel 3 O b(t)
0
-0.5
-1
-1.5
-2
-2.5
Channel 3 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2
Apr 02


Channel 4 O
1500
1000
500
Channel 4 O Fuzzy En.
0.8
0.6
0.4
0.2
0
Channel 4 O b(t)
-0.5
-1
-1.5
-2
-2.5
Channel 4 O r²(t)
1
0.95
0.9
0.85
EQ Magnitudes
4
3
2