LTPDA Training Session
Topic 5

Luigi Ferraioli

Data Fitting in LTPDA

Function	Description
curvefit	Non-linear least square fit to data
lisovfit	LISO to fit a pole/zero model to the input frequency-series
lscov	Overloads lscov function of MATLAB for Analysis Objects
polyfit	Overloads polyfit function of MATLAB for Analysis Objects
sDomainFit	Fit a partial fraction model to frequency series data
zDomainFit	Fit a partial fraction z-domain model to frequency series data

Correlated functions

Function	Description
noisegen1D	Generate colored noise with given power spectral density
noisegen2D	Generate cross correlated colored noise with given cross spectral density
whiten1D	Noise whitening tool
whiten2D	Noise whitening tool for two cross-correlated time series

Scheduled Changes for the next release

Function	Status	Actions
curvefit	✓	
lisovfit	✓	
lscov	✓	
polyfit	✓	
sDomainFit	←	• Fit Objects with a delay • Fit correctly real objects
zDomainFit	←	
noisegenND	+	Multichannel noise generator
whitenND	+	Multichannel noise whitening tool

Inspected Functions

Function	Description
curvefit	Non-linear least square fit to data
polyfit	Overloads polyfit function of MATLAB for Analysis Objects
zDomainFit	Fit a partial fraction z-domain model to frequency series data

Correlated functions

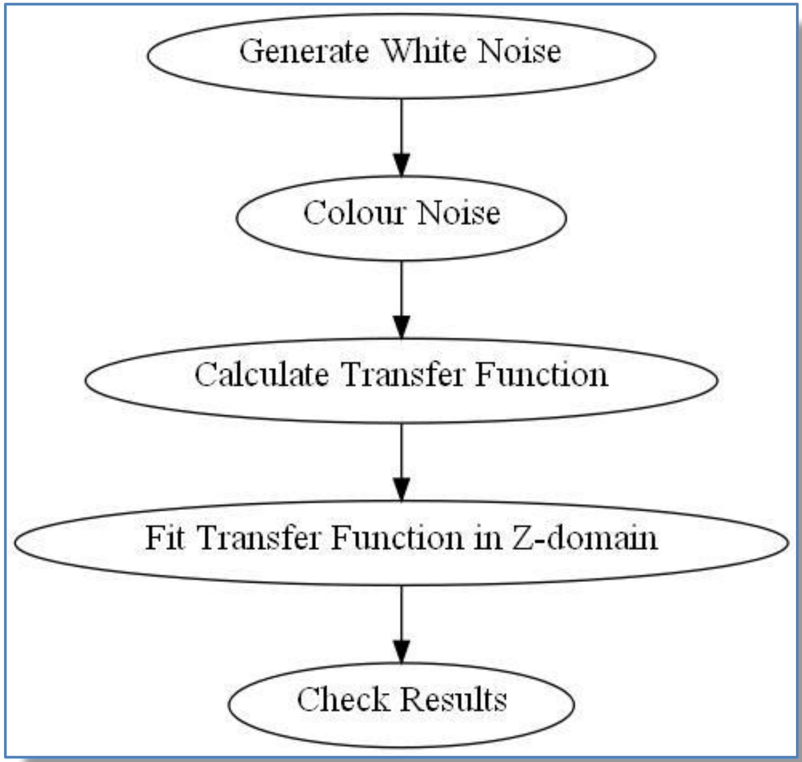
Function	Description
noisegen1D	Generate colored noise with given power spectral density

Topic 5 – Exercise 1

- Go to help section
  - LTPDA Toolbox
    - LTPDA Training Session 1
      - Topic 5 - Model fitting
      - Open the page of the first exercise
        - » System identification in z-domain
      - Open a new editor window
        - » In Matlab command window type » edit

Topic 5 – Exercise 1
System Identification in Z-domain

Relevant functions
zDomainFit
It is used to run a system identification in z-domain on a transfer function calculated from data. Model order is automatically selected on the basis of input tolerance settings



Topic 5 – Exercise 1

Generate random noise

```
a = ao(plist('tsfcn', 'randn(size(t))', 'fs', 1, 'nsecs', 10000,'yunits','m'));
```

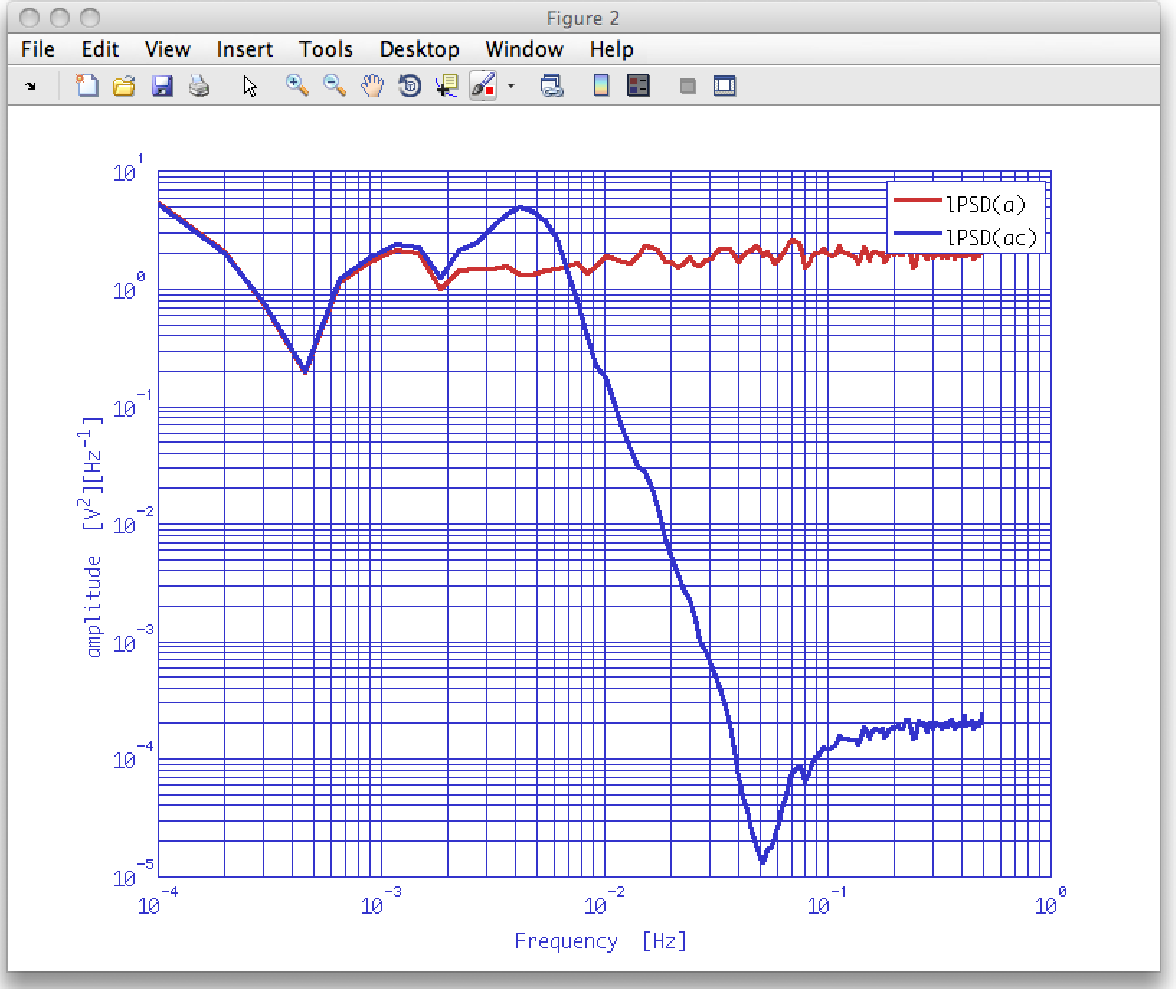
- Build a pzmodel
- Convert to a miir
- Color white noise

```
pzm = pzmodel(1, [0.005 2], [0.05 4]);

filt   = miir(pzm, plist('fs', 1));
filt.setIunits('m');
filt.setOunits('V');

% Filter the data
ac = filter(a,filt);
ac.simplifyYunits;
```

Make PSD and inspect results

```
axx  = lpsd(a);
acxx = lpsd(ac);
iplot(axx,acxx)
```

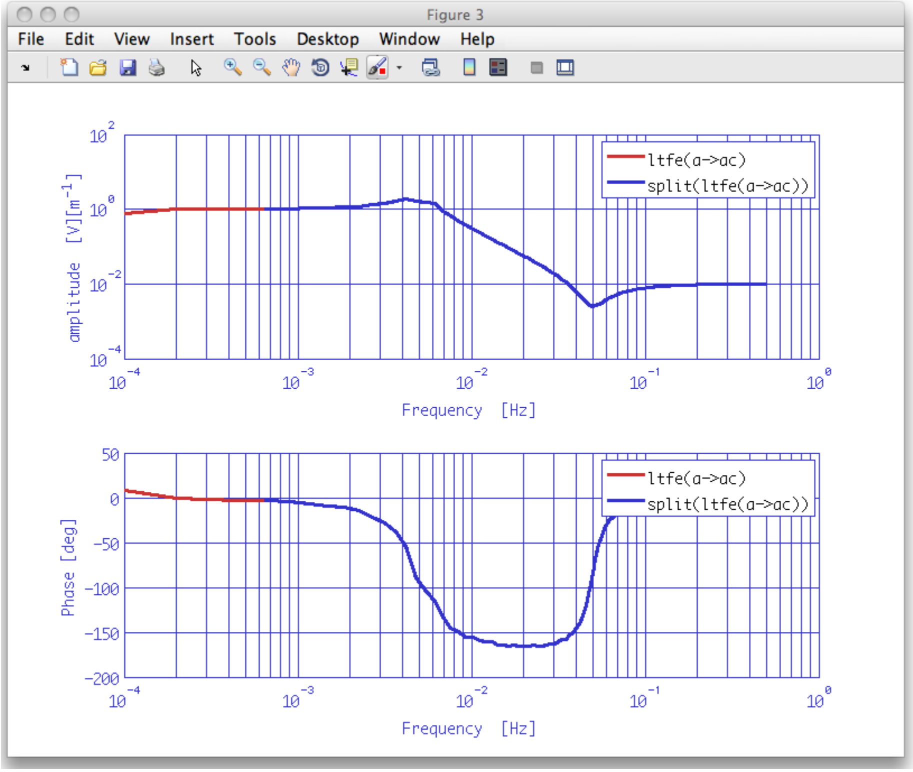
Topic 5 – Exercise 1



Topic 5 – Exercise 1

Calculate transfer function and cut away first bins

```
tf = ltfe(a,ac);
tf = tf.index(1,2);
tfsp = split(tf,plist('frequencies', [5e-4 5e-1]));

iplot(tf,tfsp)
```



Topic 5 – Exercise 1

Fit Transfer Function

```
% Set up the parameters
plfit = plist('FS',1,...    % Sampling frequency for the model filters
  'AutoSearch','on',...     % Automatically search for a good model
  'StartPolesOpt','c1',...  % Define the properties of the starting poles - complex
  'maxiter',50,...          % maximum number of iteration per model order
  'minorder',2,...          % minimum model order
  'maxorder',9,...          % maximum model order
  'weightparam','abs',...   % assign weights as 1./abs(data)
  'ResLogDiff',0.5,...      % Residuals log difference
  'ResFlat',[],...          % Residuals spectral flatness
  'RMSE',5,...              % Root Mean Squared Error Variation
  'Plot','on',...           % set the plot on or off
  'ForceStability','on',... % force to output a stable poles model
  'CheckProgress','off');   % display fitting progress on the command window

% Do the fit
fobj = zDomainFit(tfsp,plfit);

% Set the input and output units for fitted model
fobj.setIunits('m');
fobj.setOunits('V');
```

Topic 5 – Exercise 1

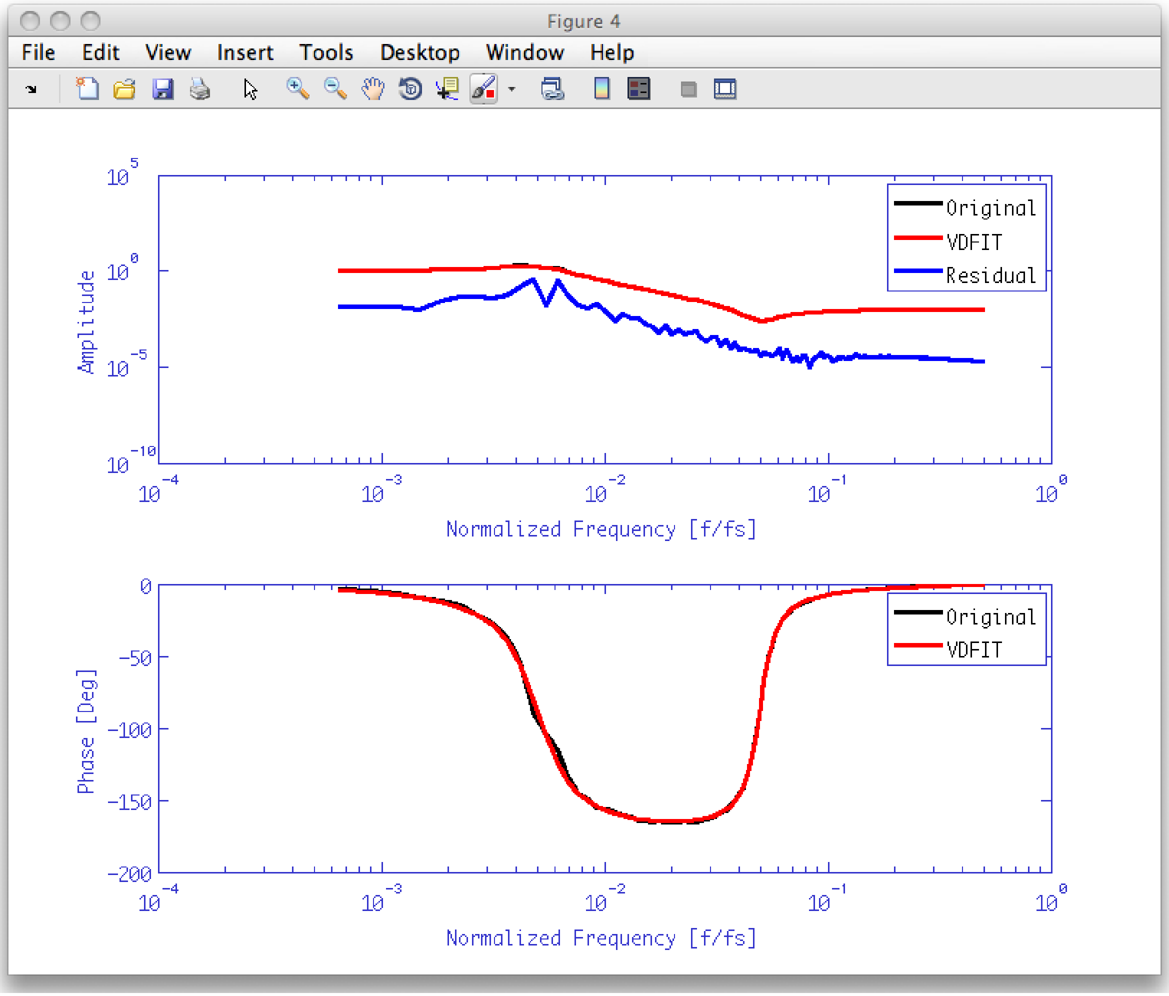
Fit Transfer Function

Check if the log-scale difference between data and fit residuals is larger than the assigned value

Check if the step-by-step root mean square error variation is lower than 10^{-d} (d is the assigned value)

```
% Set up the parameters
plfit = plist('FS',1,...        % Sampling frequency for the model filters
  'AutoSearch','on',...         % Automatically search for a good model
  'StartPolesOpt','c1',...      % Define the properties of the starting poles - complex
  'maxiter',50,...              % maximum number of iteration per model order
  'minorder',2,...              % minimum model order
  'maxorder',9,...              % maximum model order
  'weightparam','abs',...       % assign weights as 1./abs(data)
  'ResLogDiff',0.5,...          % Residuals log difference
  'ResFlat',[],...              % Residuals spectral flatness
  'RMSE',5,...                  % Root Mean Squared Error Variation
  'Plot','on',...               % set the plot on or off
  'ForceStability','on',...     % force to output a stable poles model
  'CheckProgress','off');       % display fitting progress on the command window

% Do the fit
fobj = zDomainFit(tfsp,plfit);

% Set the input and output units for fitted model
fobj.setIunits('m');
fobj.setOunits('V');
```

Topic 5 – Exercise 1

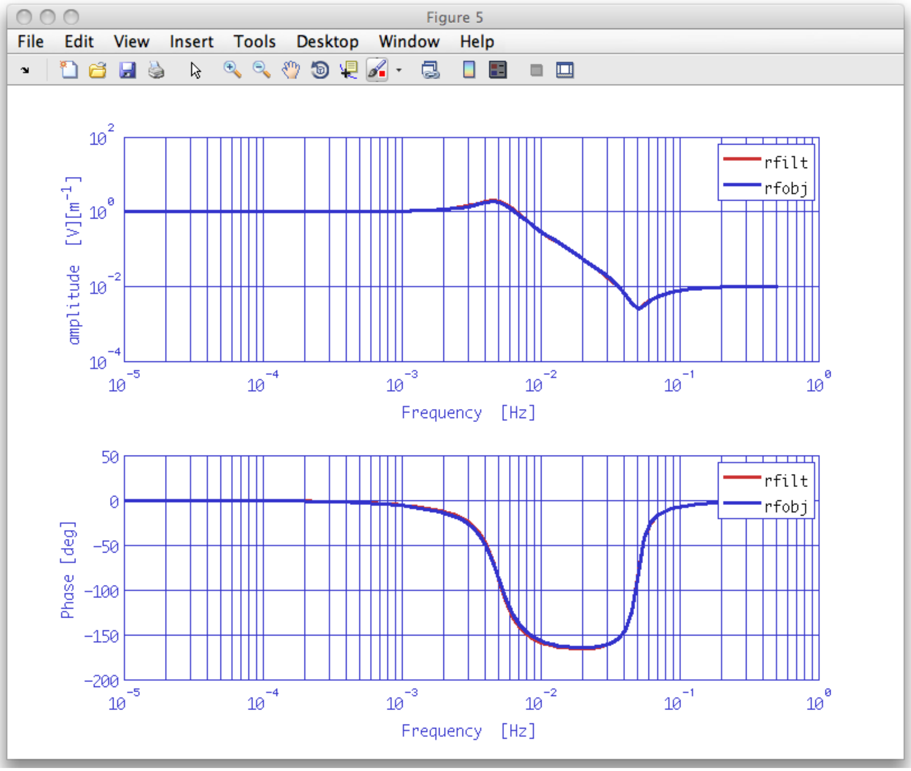


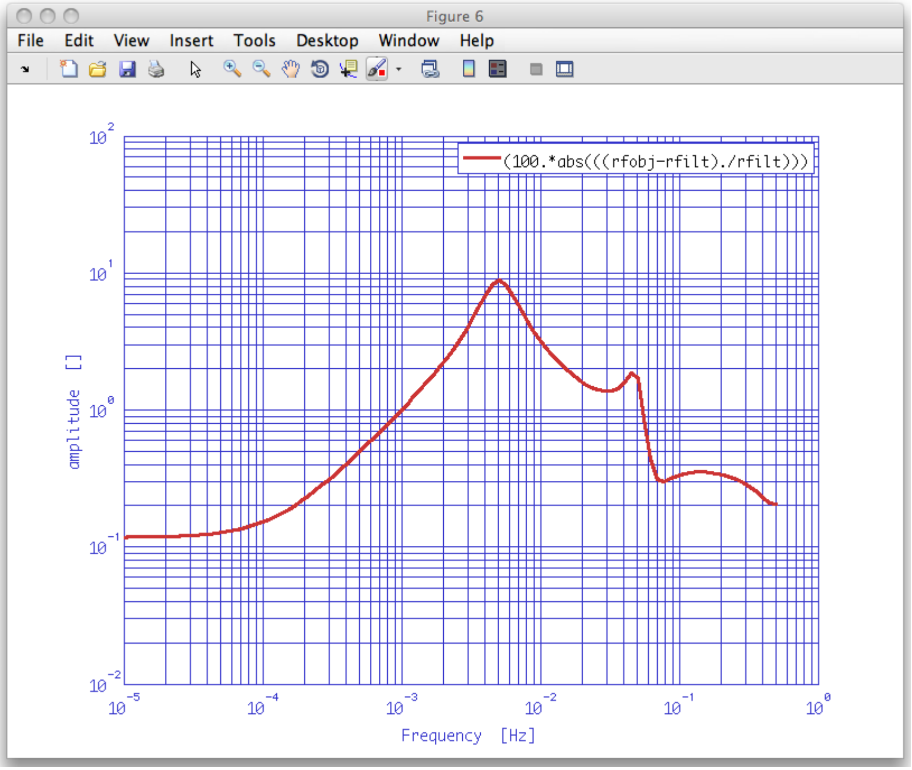
Topic 5 – Exercise 1

Calculate filters response and check fit results

```
% set plist for filter response
plrsp = plist('bank','parallel','f1',1e-5,'f2',0.5,'nf',100,'scale','log');

% compute the response of the original noise-shape filter
rfilt = resp(filt,plrsp);
rfilt.setName;

% compute the response of our fitted filter bank
rfobj = resp(fobj,plrsp);
rfobj.setName;

% compare the responses
iplot(rfilt,rfobj)

% and the percentage error on the magnitude
pdiff = 100.*abs((rfobj-rfilt)./rfilt);
pdiff.simplifyYunits;
iplot(pdiff,plist('YRanges',[1e-2 100]))
```

Topic 5 – Exercise 1



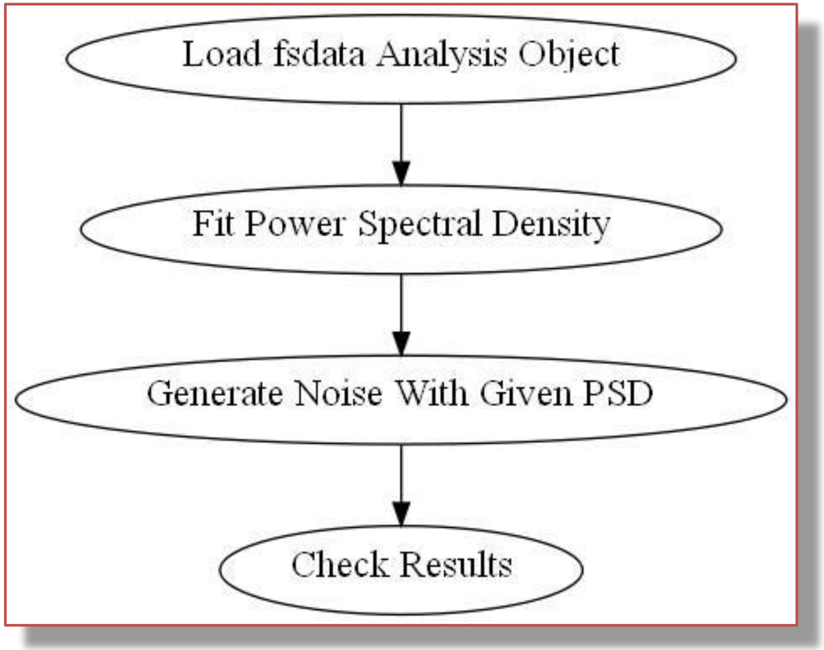


Topic 5 – Exercise 2

- Go to help section
  - LTPDA Toolbox
    - LTPDA Training Session 1
      - Topic 5 - Model fitting
      - Open the page of the second exercise
        - » Generation of noise with given psd
      - Open a new editor window
        - » In Matlab command window type » edit

Topic 5 – Exercise 2
Generation of Noise with Given PSD

Relevant functions	
zDomainFit	It is used to get a smooth model for the calculated psd.
noisegen1D	Generate noise with the given power spectral density



Topic 5 – Exercise 2

Load test noise

```
tn = ao(plist('filename', 'topic5/T5_Ex03_TestNoise.xml'));
tn.setName;
```

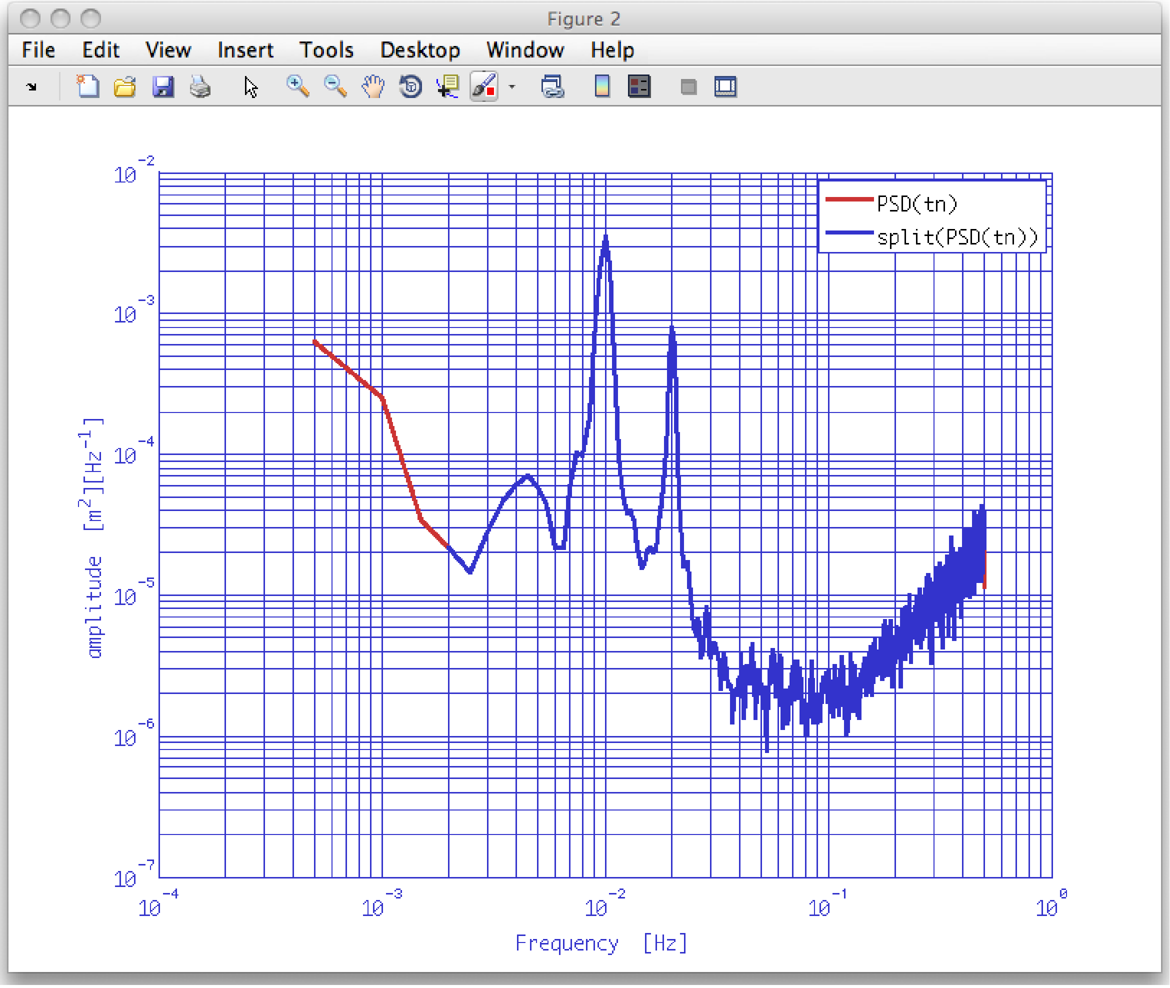
Calculate power spectral density

```
tnxx = tn.psd(plist('Nfft',2000));
```

Cut away first bins and plot

```
tnxxr = split(tnxx,plist('frequencies', [2e-3 5e-1]));
iplot(tnxx,tnxxr)
```

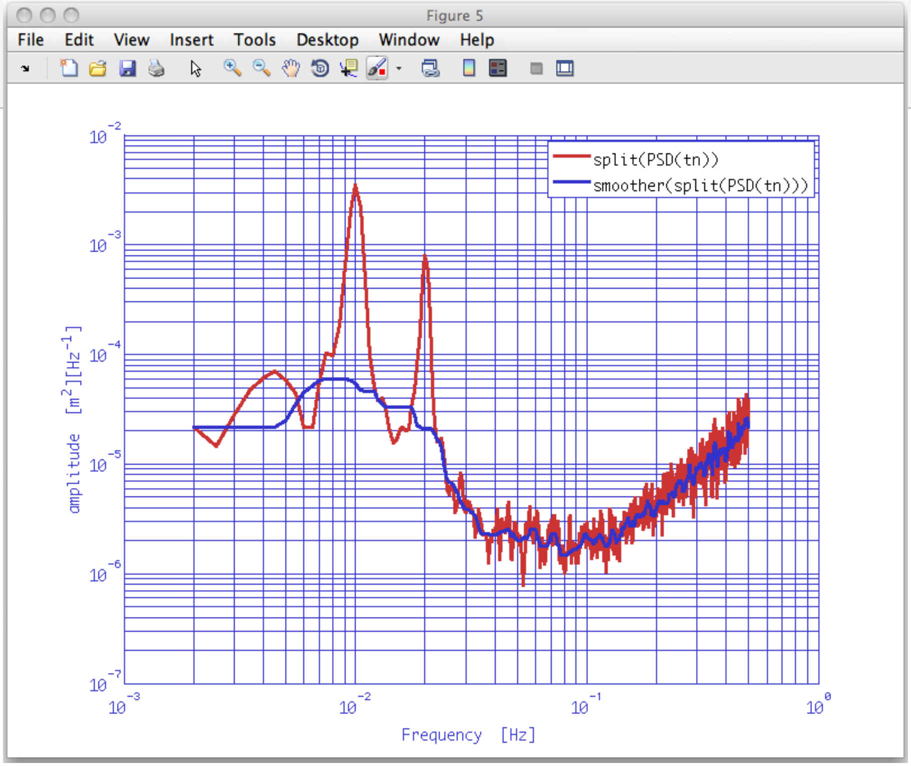
Topic 5 – Exercise 2



Topic 5 – Exercise 2

We smooth PSD data and then define the weights as the inverse of the absolute value of smoothed PSD. This should help the fit function to do a good job with noisy data

```
stnxx = smoother(tnxxr);
iplot(tnxxr, stnxx)
wgh = 1./abs(stnxx);
```



Topic 5 – Exercise 2

Fit PSD

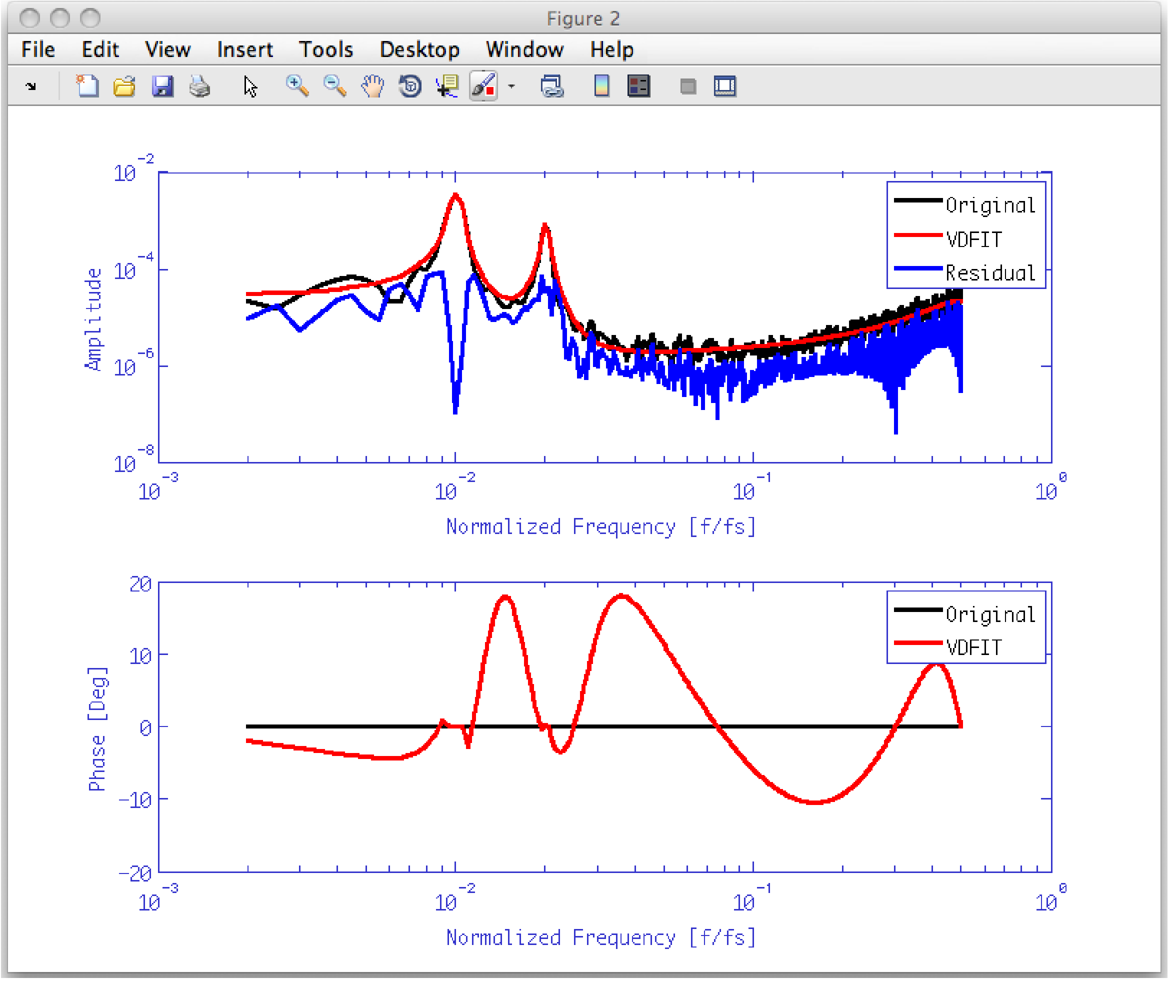
```
plfit = plist('FS',1,...
              'AutoSearch','on',...
              'StartPolesOpt','c1',...
              'maxiter',50,...
              'minorder',10,...
              'maxorder',45,...
              'weights',wgh,... % assign externally calculated weights
              'ResLogDiff',[],...
              'ResFlat',0.77,...
              'RMSE',5,...
              'Plot','on',...
              'ForceStability','off',...
              'CheckProgress','off');

% Do the fit
[param,fmod] = zDomainFit(tnxxr,plfit);
```

Topic 5 – Exercise 2

Check if residuals spectral flatness is larger than the assigned value

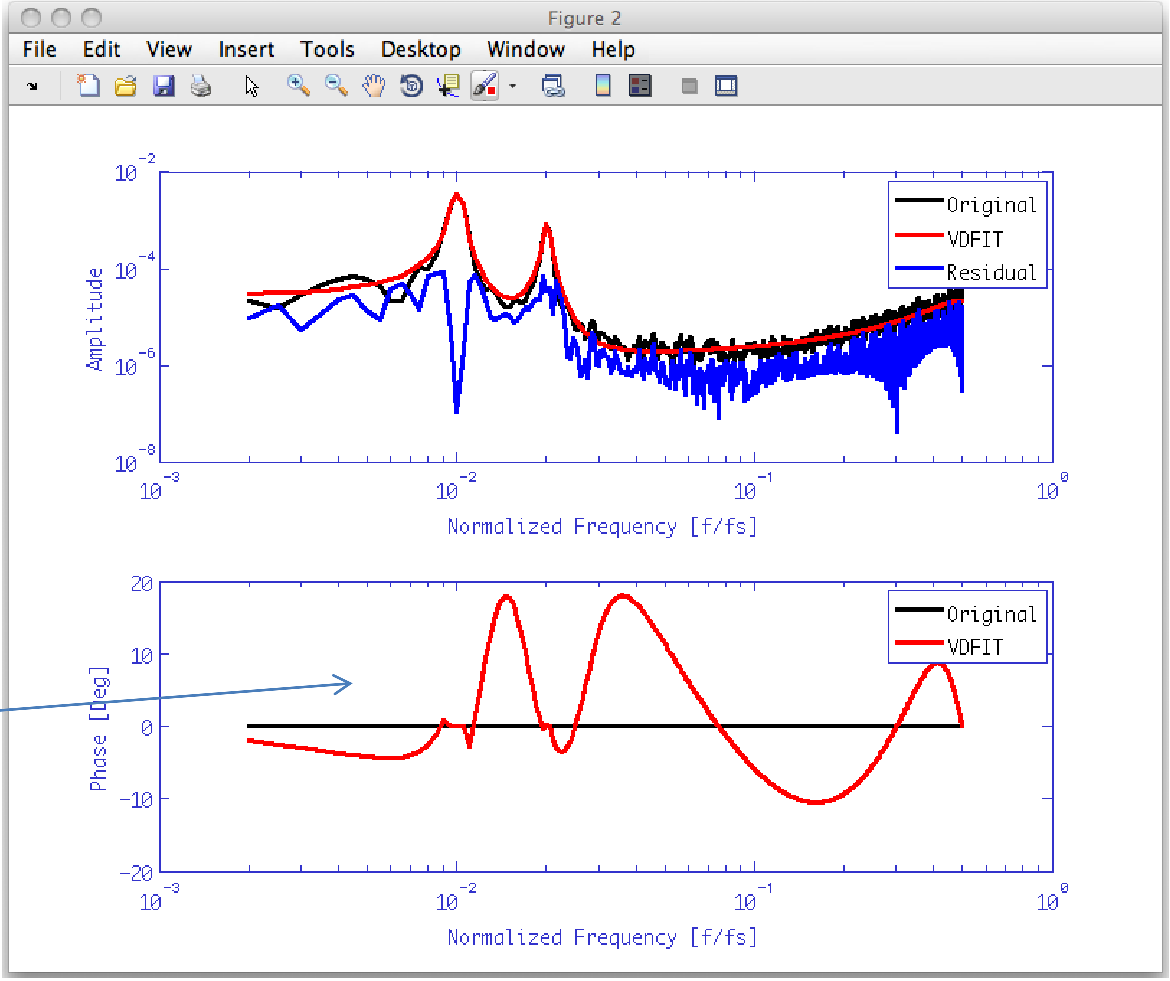
```
plfit = plist('FS',1,...
              'AutoSearch','on',...
              'StartPolesOpt','c1',...
              'maxiter',50,...
              'minorder',10,...
              'maxorder',45,...
              'weights',wgh,... % assign externally calculated weights
              'ResLogDiff',[],...
              'ResFlat',0.77,...
              'RMSE',5,...
              'Plot','on',...
              'ForceStability','off',...
              'CheckProgress','off');

% Do the fit
[param,fmod] = zDomainFit(tnxxr,plfit);
```

Topic 5 – Exercise 2



Topic 5 – Exercise 2

Phase is not perfectly fitted – this will be solved with the next release



Topic 5 – Exercise 2

Generate white noise

```
a = ao(plist('tsfcn', 'randn(size(t))', 'fs', 1, 'nsecs', 10000,'yunits','m'));
```

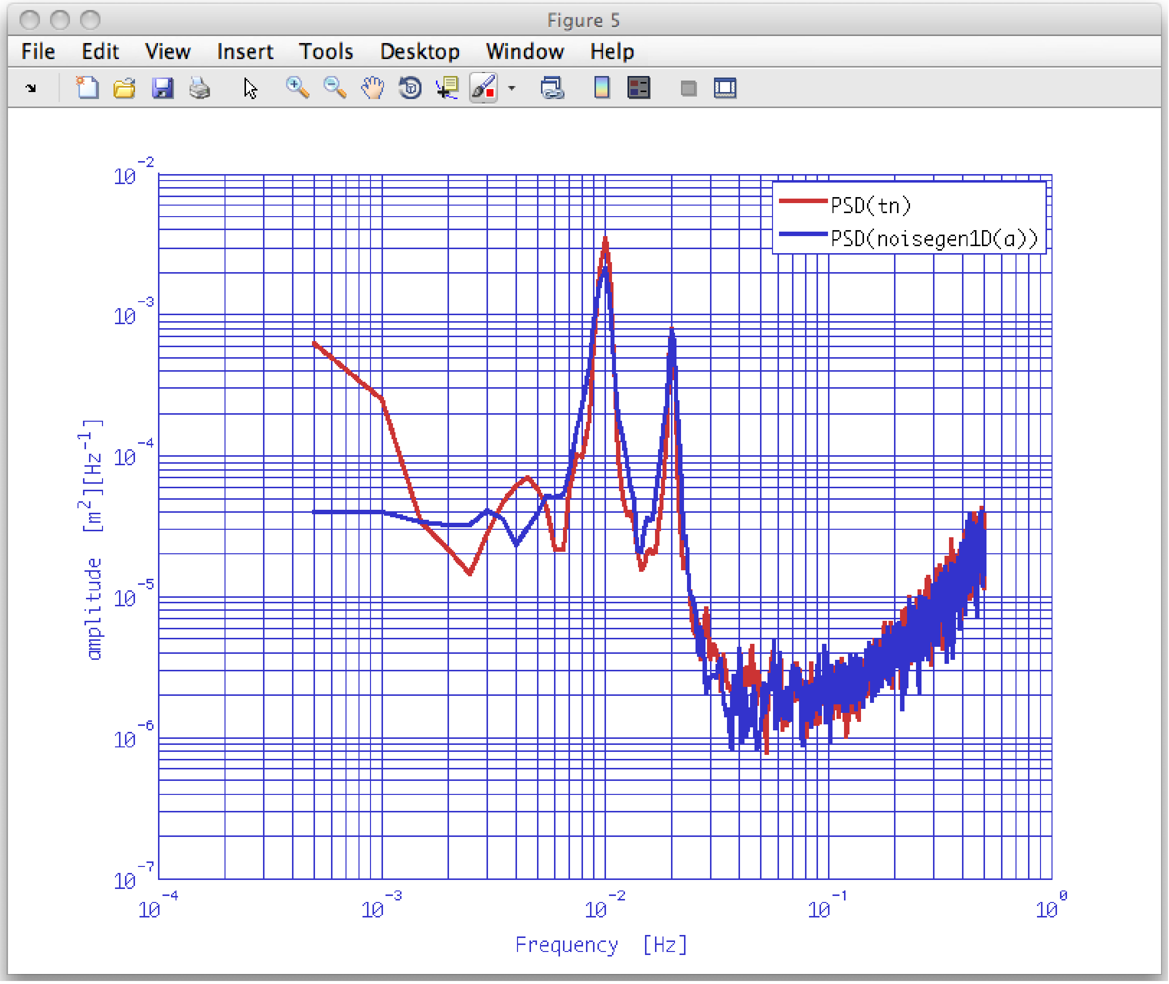
Color noise with given psd

```
plng = plist(...
      'model', abs(fmod), ... % model for colored noise psd
      'MaxIter', 50, ...      % maximum number of fit iteration per model order
      'PoleType', 2, ...      % generates complex poles distributed in the unitary circle
      'MinOrder', 20, ...     % minimum model order
      'MaxOrder', 50, ...     % maximum model order
      'Weights', 2, ...       % weight with 1/abs(model)
      'Plot', false,...       % on to show the plot
      'Disp', false,...       % on to display fit progress on the command window
      'RMSEVar', 7,...        % Root Mean Squared Error Variation
      'FitTolerance', 2);     % Residuals log difference

ac = noisegen1D(a, plng);
```

Calculate psd and plot

```
acxx = ac.psd(plist('Nfft',2000));
iplot(tnxx,acxx)
```

Topic 5 – Exercise 2

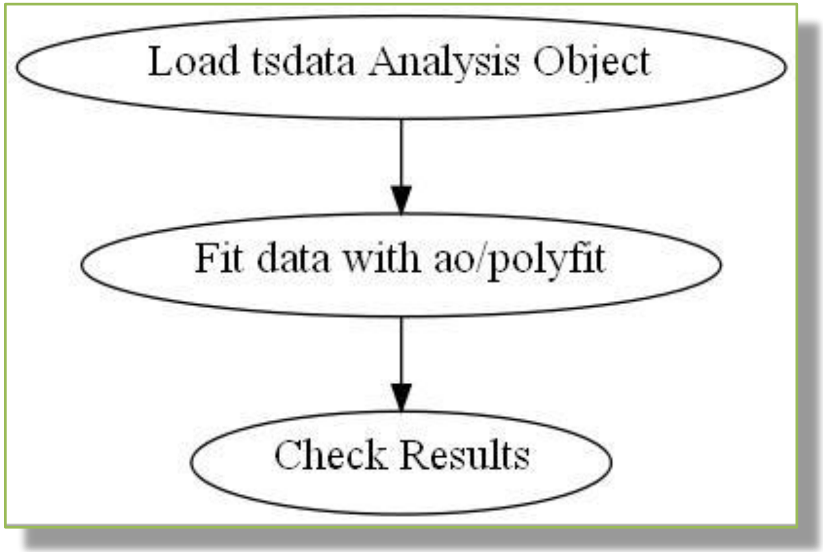


Topic 5 – Exercise 3

- Go to help section
  - LTPDA Toolbox
    - LTPDA Training Session 1
      - Topic 5 - Model fitting
      - Open the page of the first exercise
        - » Fitting time series with polynomials
      - Open a new editor window
        - » In Matlab command window type » edit

Topic 5 – Exercise 3
Fit Time Series with Polynomials

Relevant functions	
polyfit	Fit a time series with a 6 order polynomial



Topic 5 – Exercise 3

Load test data

```
a = ao(plist('filename', 'topic5/T5_Ex04_TestNoise.xml'));
a.setName;
```

Fit data with a polynomial

```
plfit = plist('N', 6);
p     = polyfit(a, plfit);
```

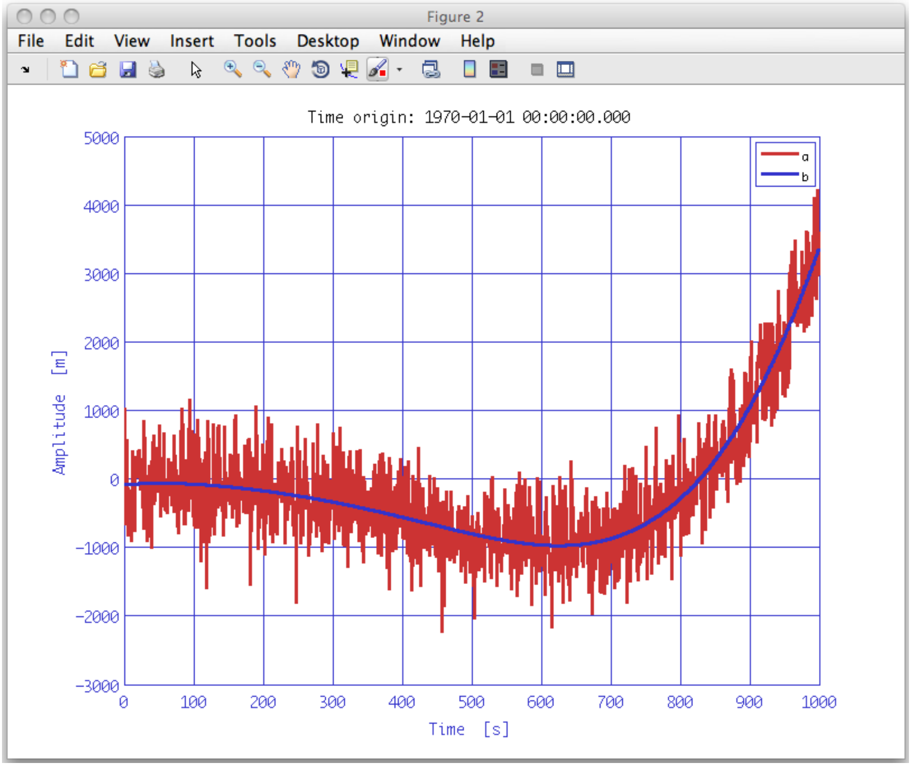
Evaluate fitted model

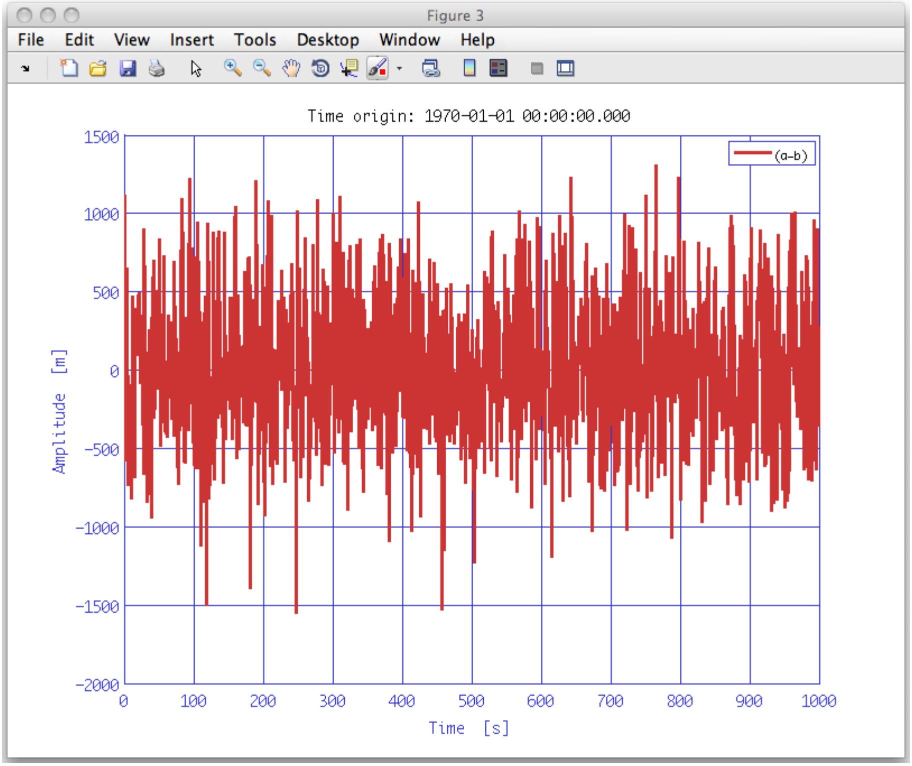
```
b = ao(plist('polyval', p, 't', a));
b.setYunits(a.yunits);
b.setName;
```

Check results - plot data, model and fit residuals

```
iplot(a,b)
iplot(a-b)
```

Topic 5 – Exercise 3





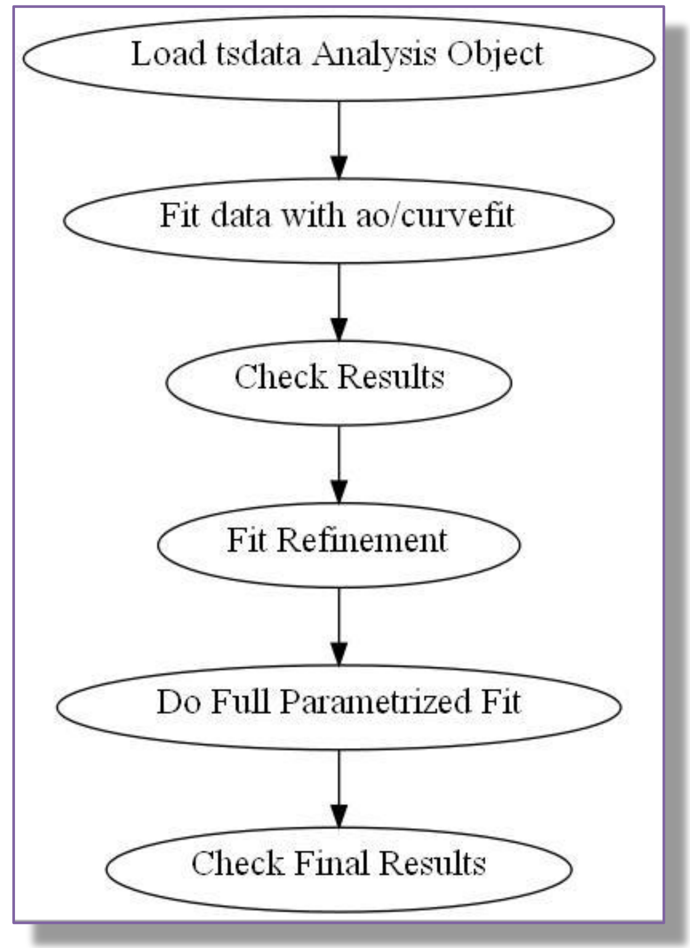
Topic 5 – Exercise 4

- Go to help section
  - LTPDA Toolbox
    - LTPDA Training Session 1
      - Topic 5 - Model fitting
      - Open the page of the first exercise
        - » Non-linear least square fitting of time series
      - Open a new editor window
        - » In Matlab command window type » edit

Topic 5 – Exercise 4
Non-linear least square fit of time series

Relevant functions	
curvefit	Fit a time series with a non-linear model (linearly chirped sine wave)

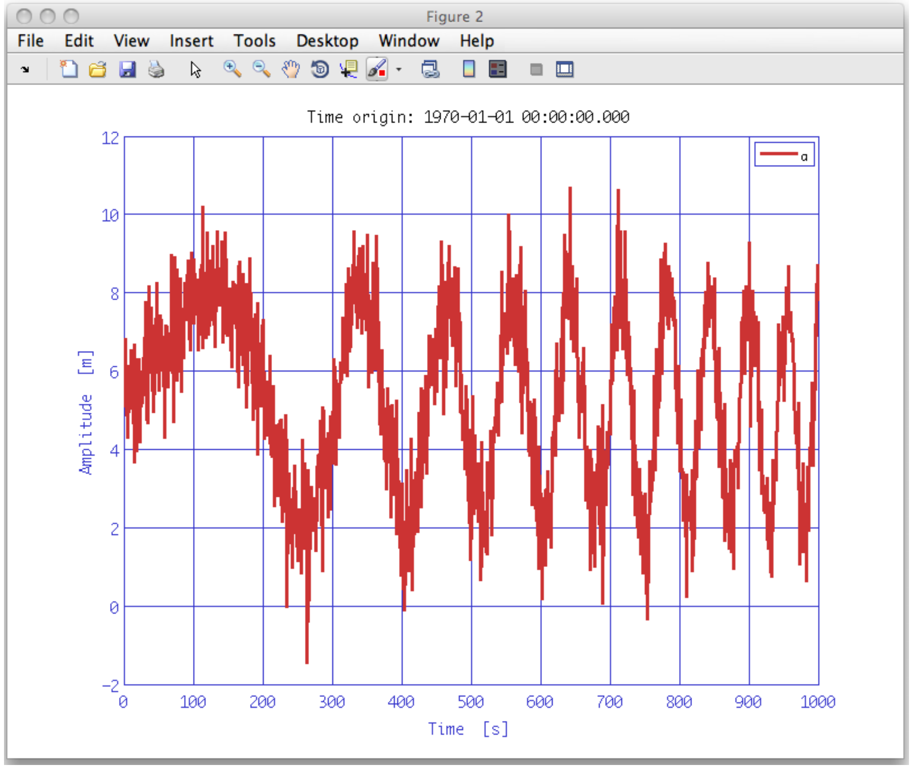
$$A\sin\left[2\pi \;\; f_0 + kt \;\; t + \varphi\right]$$



Topic 5 – Exercise 4

Load data and plot

```
a = ao(plist('filename', 'topic5/T5_Ex05_TestNoise.xml'));
a.setName;
iplot(a)
```



Topic 5 – Exercise 4

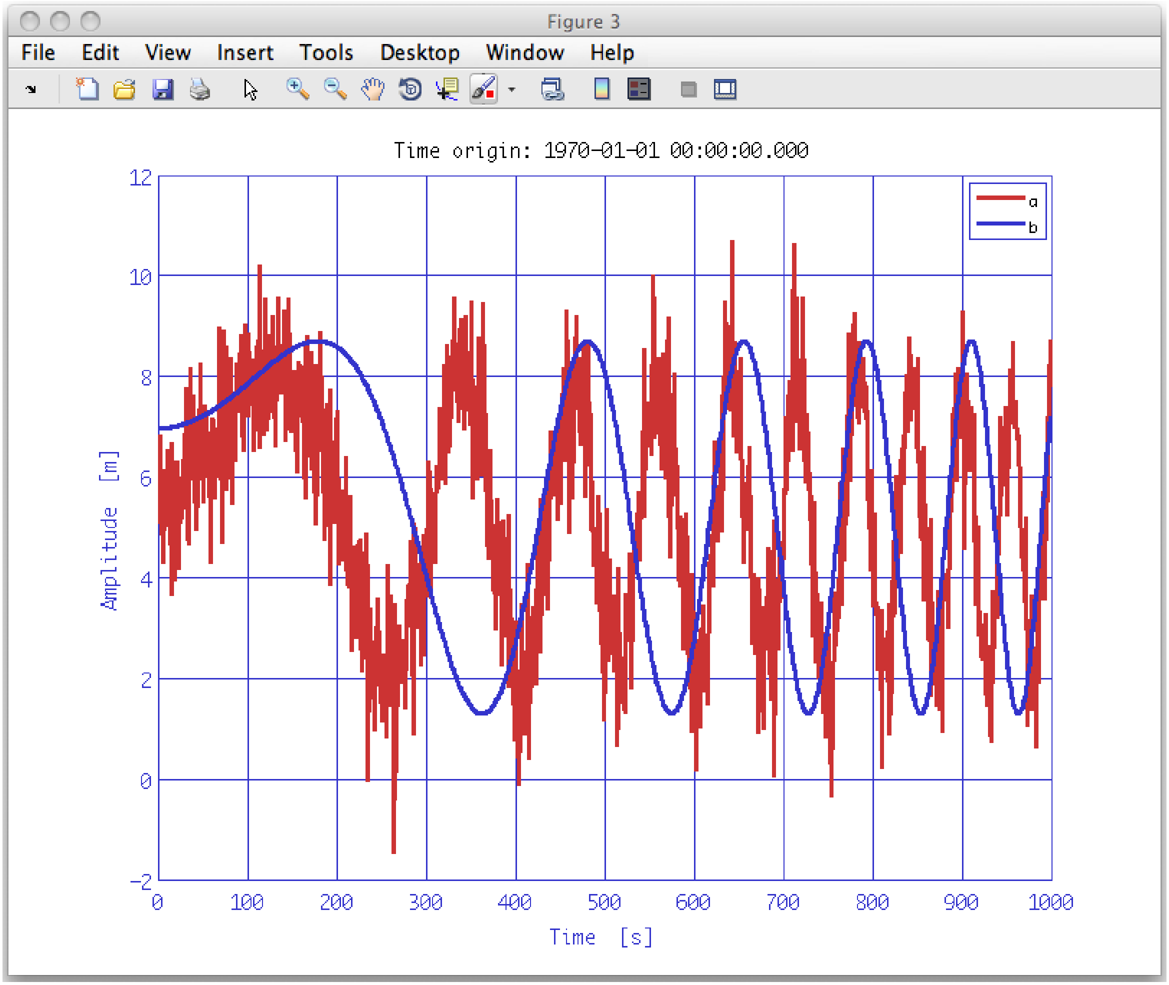
Fit data – Try fitting Amplitude, Frequency, Chirp Parameter and Phase

```
plfit = plist('Function', 'ADDP(1) + P(1).*sin(2.*pi.*(P(2) + P(3).*Xdata).*Xdata + P(4))', ...
          'P0', [4 3e-5 5e-6 0.5], ...
          'LB', [1 0 0 -pi], ...
          'UB', [5 1 1 pi],...
          'ADDP', {5});
params = curvefit(a, plfit);
```

Evaluate fit results

```
pleval = plist('Function', 'ADDP(1) + P(1).*sin(2.*pi.*(P(2) + P(3).*Xdata).*Xdata + P(4))', ...
          'Xdata', a, ...
          'dtype', 'tsdata', ...
          'ADDP', {5});
b = evaluateModel(params, pleval);
b.setYunits(a.yunits);
b.setXunits(a.xunits);
b.setName;
iplot(a,b)
```

Topic 5 – Exercise 4

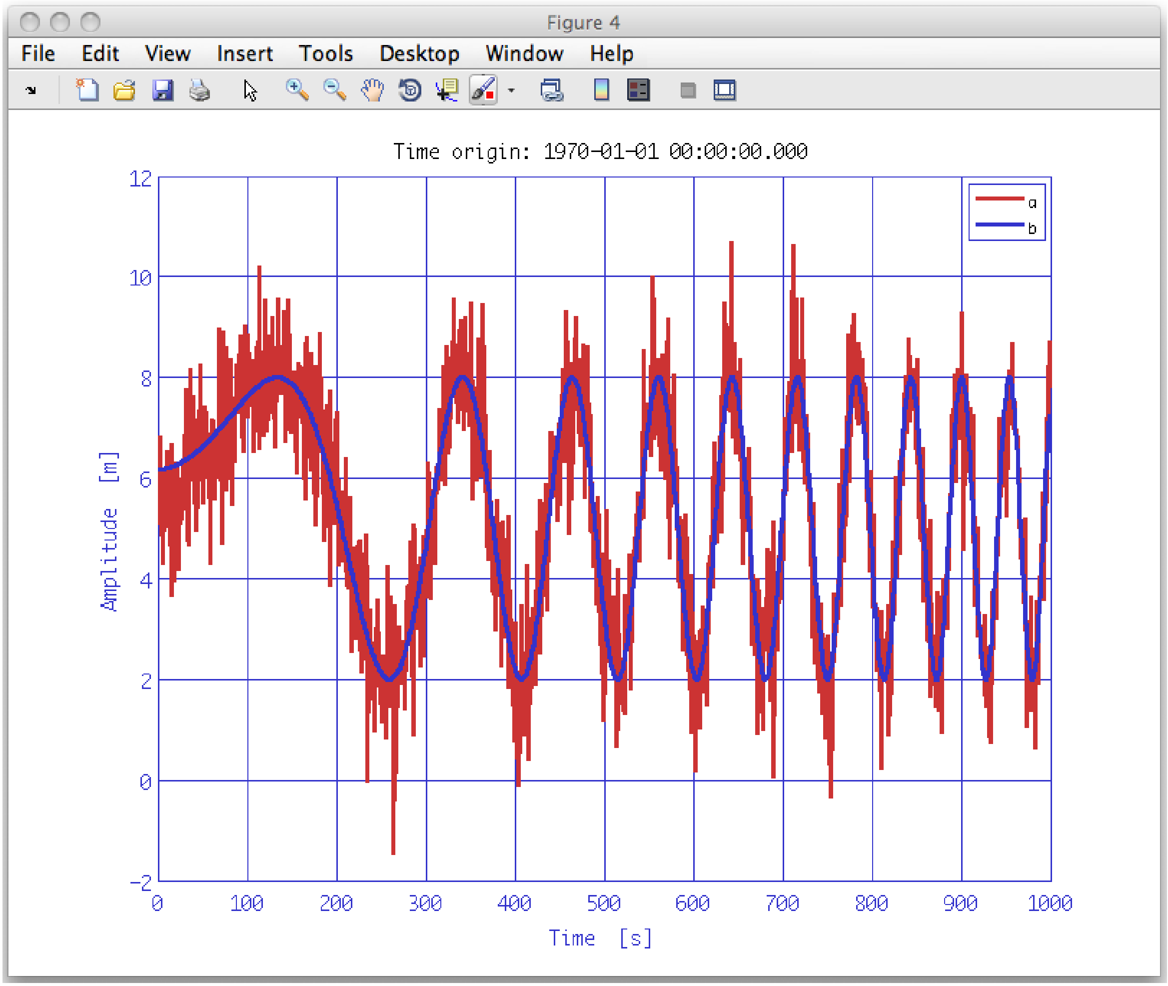


Topic 5 – Exercise 4

Run a fit with a reduced set of parameters – fix amplitude and phase

```
% Do the fit again
plfit = plist('Function', 'ADDP(1) + 3.*sin(2.*pi.*(P(1) + P(2).*Xdata).*Xdata + 0.4)', ...
            'P0', [7e-4 9e-6], ...
            'LB', [1e-7 1e-7], ...
            'UB', [1 1e-4],...
            'ADDP', {5});

params = curvefit(a, plfit);

% Evaluate the model
pleval = plist('Function', 'ADDP(1) + 3.*sin(2.*pi.*(P(1) + P(2).*Xdata).*Xdata + 0.4)', ...
              'Xdata', a, ...
              'dtype', 'tsdata', ...
              'ADDP', {5});
b = evaluateModel(params, pleval);
b.setYunits(a.yunits);
b.setXunits(a.xunits);
b.setName;

iplot(a,b)
```

Topic 5 – Exercise 4

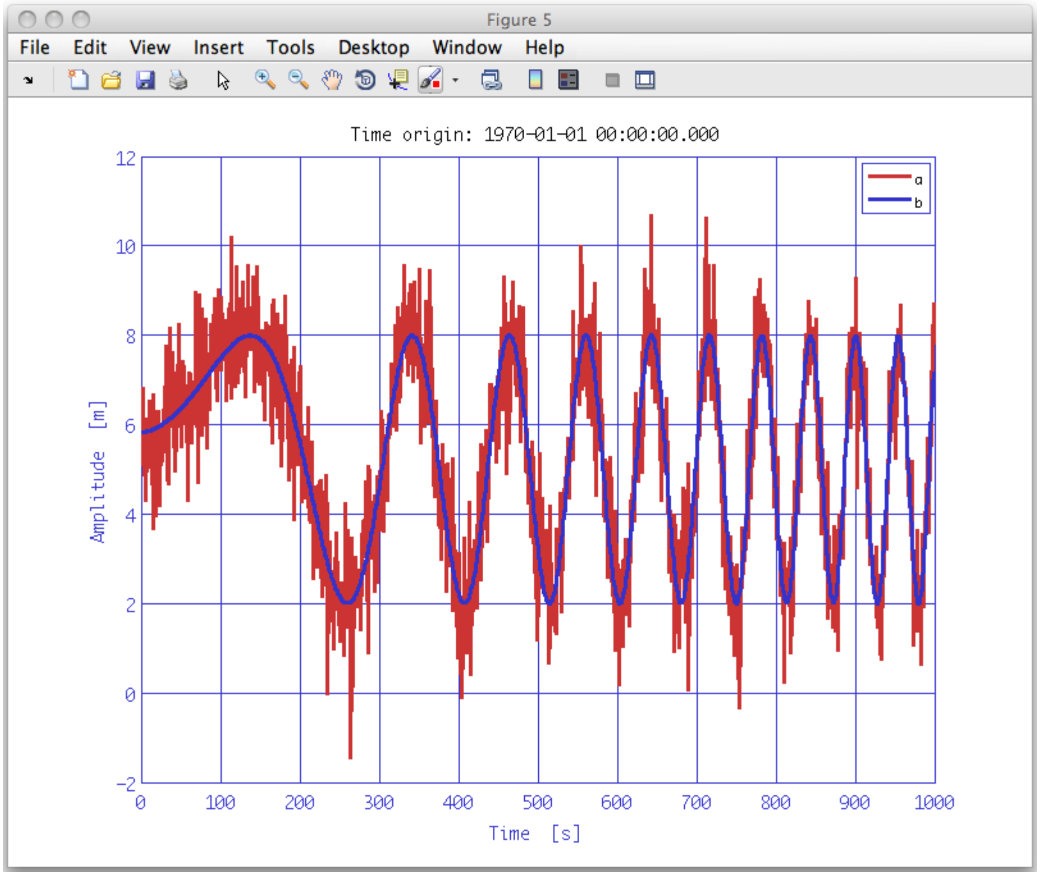


Topic 5 – Exercise 4

Run a fit with the full set of parameters – use former results as starting guess for frequency and chirp parameter

```
% Do the fit again
plfit = plist('Function', 'ADDP(1) + P(1).*sin(2.*pi.*(P(2) + P(3).*Xdata).*Xdata + P(4))', ...
              'P0', [3 5e-5 1e-5 0.4], ...
              'LB', [2.8 1e-5 1e-6 0.2], ...
              'UB', [3.2 5e-4 5e-4 0.5],...
              'ADDP', {5});

params = curvefit(a, plfit);

% Evaluate the model
pleval = plist('Function', 'ADDP(1) + P(1).*sin(2.*pi.*(P(2) + P(3).*Xdata).*Xdata + P(4))', ...
              'Xdata', a, ...
              'dtype', 'tsdata', ...
              'ADDP', {5});
b = evaluateModel(params, pleval);
b.setYunits(a.yunits);
b.setXunits(a.xunits);
b.setName;

iplot(a,b)
```

Topic 5 – Exercise 4



True parameters

```
ADDP = 5
P(1) = 3
P(2) = 1e-4
P(3) = 1e-5
P(4) = 0.3
```

Fitted parameters

```
ADDP = 5
P(1) = 2.993
P(2) = 0.000121
P(3) = 9.983e-006
P(4) = 0.278
```

Topic 5 – Exercise 4

We could look at the covariance matrix in the process info
» In the command window of Matlab »

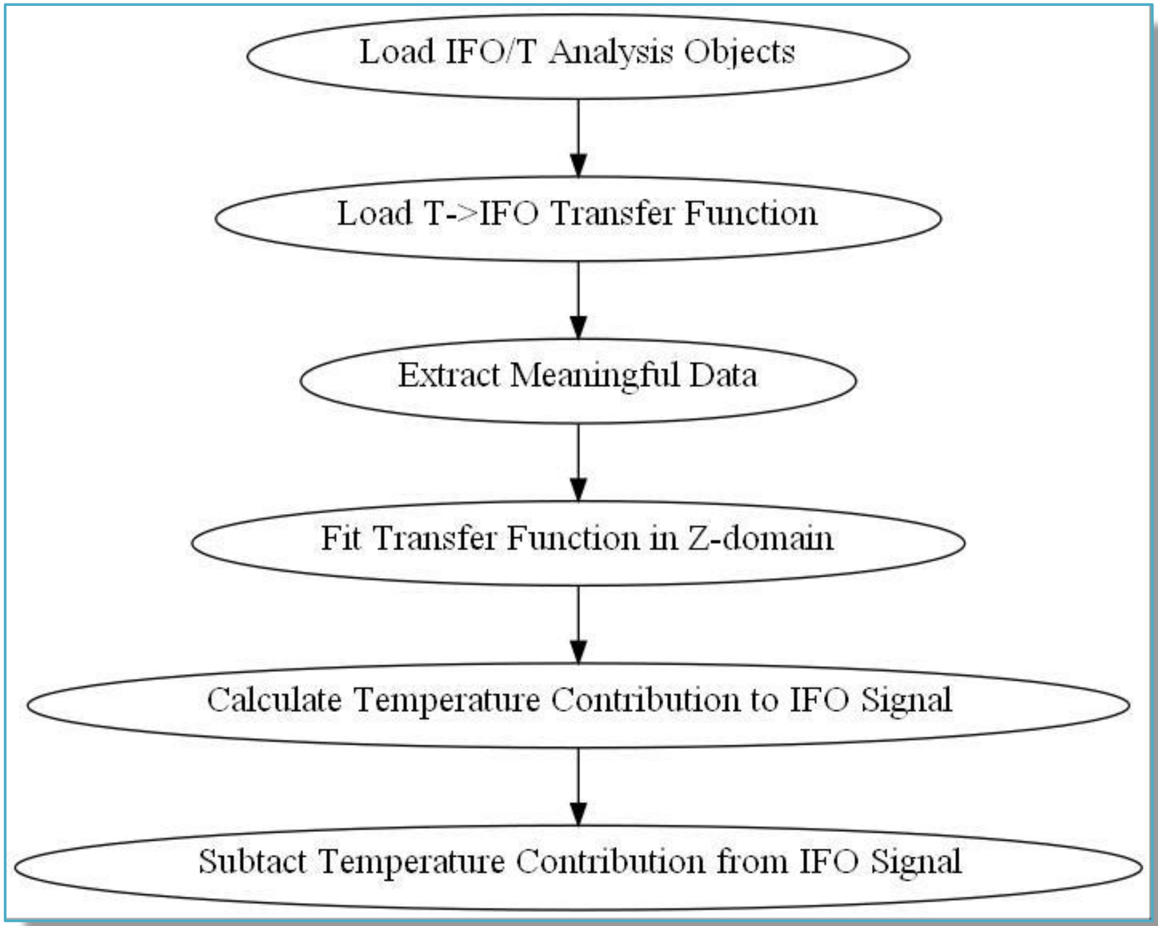
```
    params.procinfo.find('cor')
Columns 1 through 3

                         1        0.0220543553134523       0.00840698749447142
        0.0220543553134523                         1       -0.963274881180157
       0.00840698749447142        -0.963274881180157                        1
       -0.0911417933676055        -0.833580057704702        0.692767145487321

  Column 4

       -0.0911417933676055
        -0.833580057704702
         0.692767145487321
                         1
```

Topic 5 – Exercise 5

- Go to help section
  - LTPDA Toolbox
    - LTPDA Training Session 1
      - Topic 5 - Model fitting
      - Open the page of the first exercise
        - » Time-domain subtraction of temperature contribution to interferometer signal
      - Open a new editor window
        - » In Matlab command window type » edit

Topic 5 – Exercise 5
IFO/Temperature Example

Relevant functions
zDomainFit
Fit a z domain model to measured transfer function. The model is used to filter data so as isolating temperature contribution to IFO signal



Topic 5 – Exercise 5

Load data from exercise 2 and split to extract the good part

```
ifo = ao(plist('filename', 'ifo_temp_example/ifo_fixed.xml'));
ifo.setName;
T = ao(plist('filename', 'ifo_temp_example/temp_fixed.xml'));
T.setName;

% Split out the good part of the data
pl_split = plist('split_type', 'interval', ...
            'start_time', ifo.t0 + 40800, ...
            'end_time', ifo.t0 + 193500);

ifo_red = split(ifo, pl_split);
T_red = split(T, pl_split);
```

Plot to inspect data

```
iplot(ifo_red,T_red,plist('arrangement', 'subplots'))
```

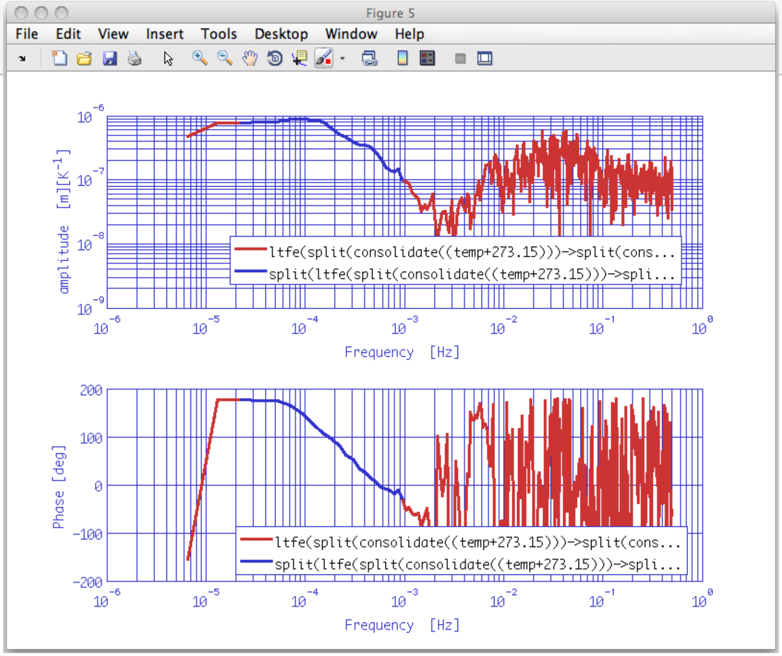
Topic 5 – Exercise 5

Load transfer function data from exercise 4

```
tf = ao('ifo_temp_example/T_ifo_tf.xml');
```

Split to extract meaningful data

```
tfsp = split(tf,plist('frequencies', [2e-5 1e-3]));
iplot(tf,tfsp)
```

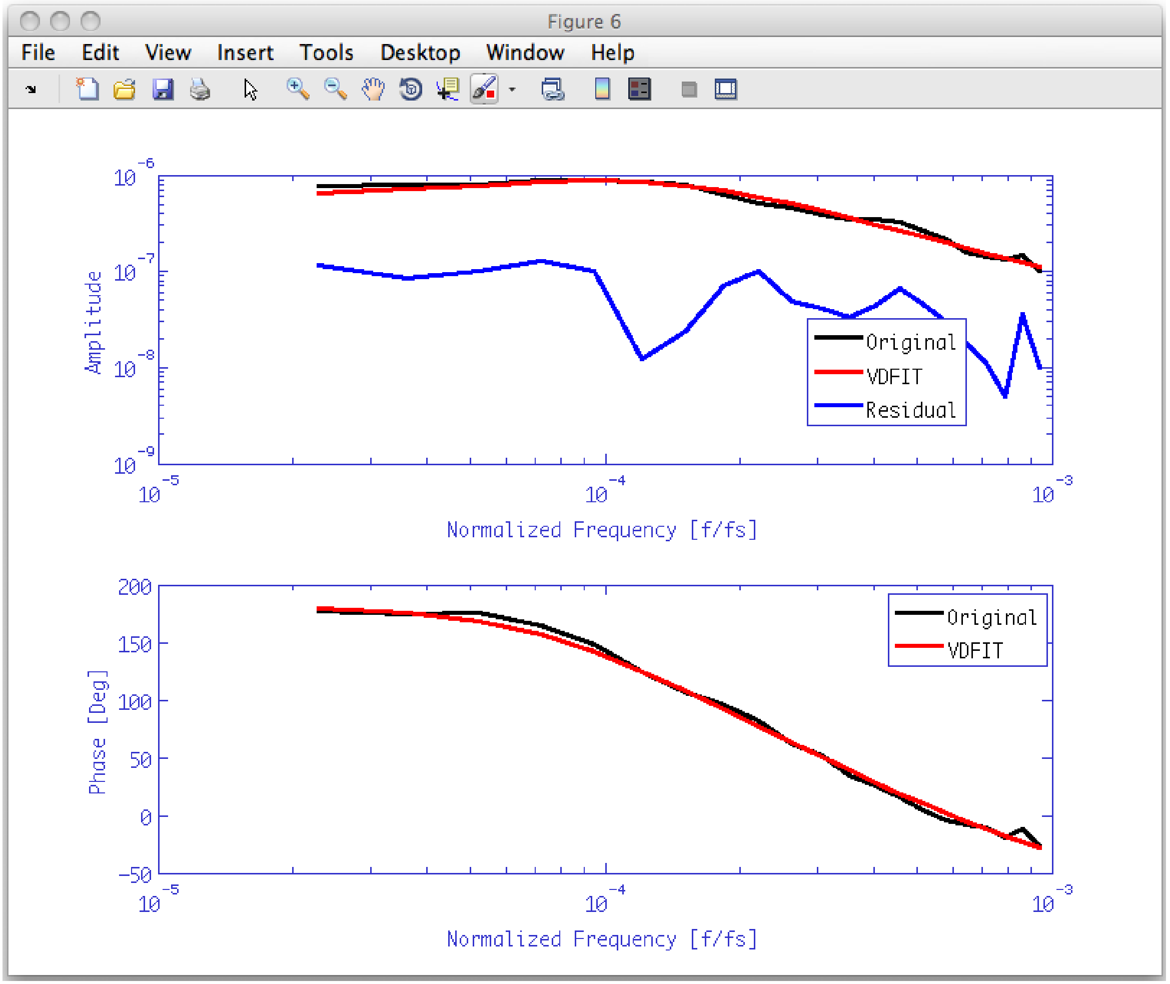


Topic 5 – Exercise 5

Fit transfer function – fit with a fixed order

```
plfit = plist('FS',1,...
    'AutoSearch','off',...
    'StartPolesOpt','c1',...
    'maxiter',20,...
    'minorder',3,...
    'maxorder',3,...
    'weightparam','abs',...
    'Plot','on',...
    'ForceStability','on',...
    'CheckProgress','off');

fobj = zDomainFit(tfsp,plfit);
fobj.setIunits('K');
fobj.setOunits('m');
```

Topic 5 – Exercise 5



Topic 5 – Exercise 5

Filter Temperature data with the fitted model in order to extract temperature contribution to interferometer signal

```
ifoT = filter(T_red,fobj,plist('bank','parallel'));
ifoT.detrend(plist('order',0));
ifoT.simplifyYunits;
ifoT.setName;
```

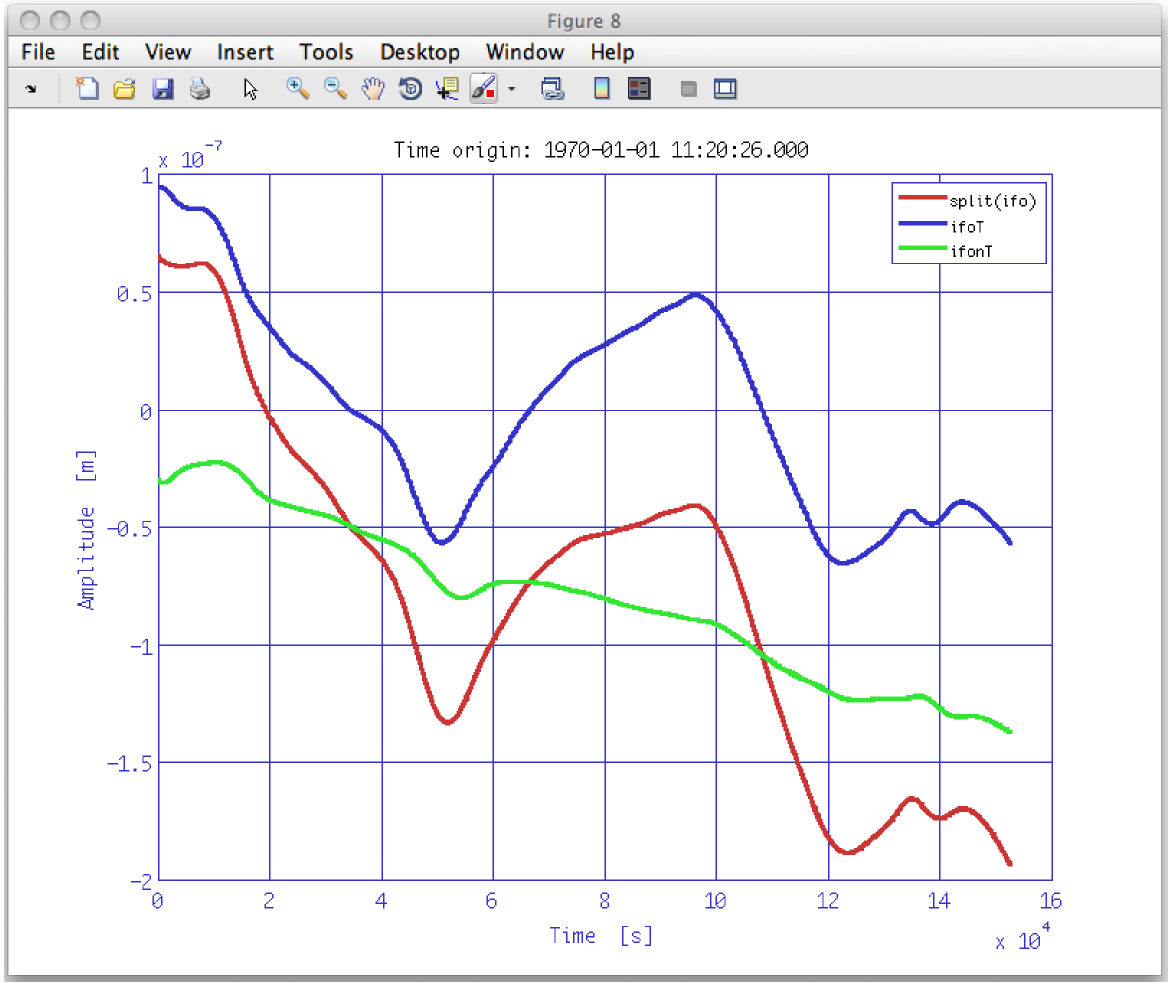
Subtract temperature contribution

```
ifonT = ifo_red - ifoT;
ifonT.setName;
```

Plot to check results

```
iplot(ifo_red,ifoT,ifonT)
```

Topic 5 – Exercise 5

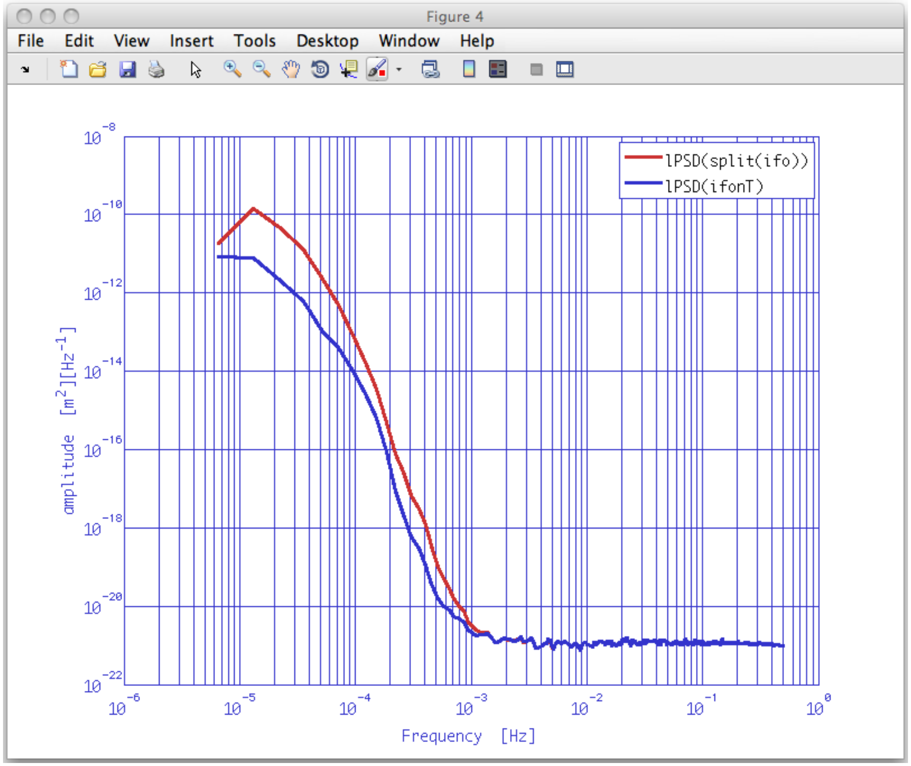


Topic 5 – Exercise 5

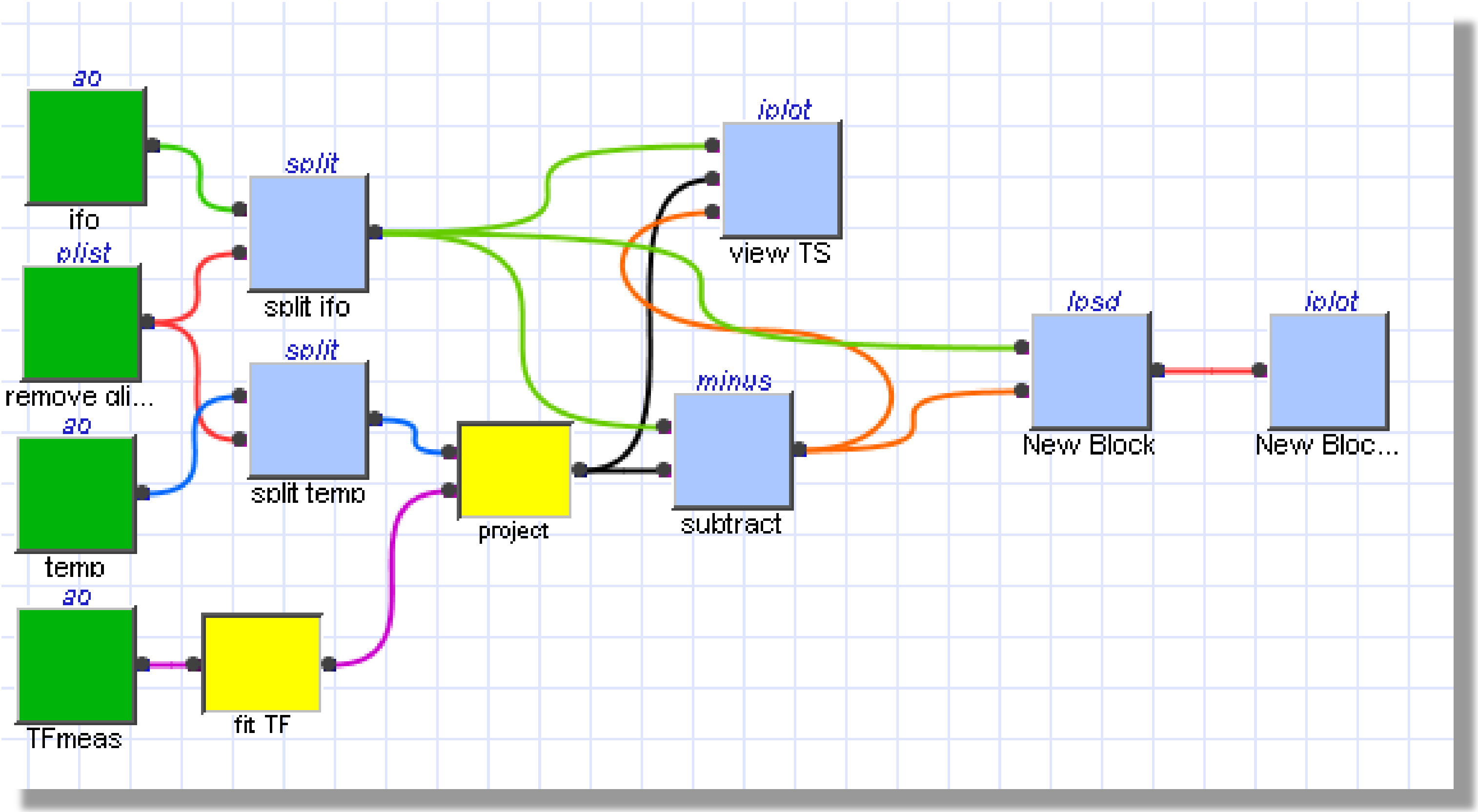
Compare power spectral density of IFO and IFO-Temp signals

```
ifoxx = ifo_red.lpsd;
ifonTxx = ifonT.lpsd;
iplot(ifoxx,ifonTxx)
```

Temperature subtraction changes low frequencies power content



Topic 5 – Exercise 5



Topic 5 – Exercise 5