Data Analysis

Experimental Methods in Marine Hydrodynamics
Lecture in week 36

By Valentin Chabaud, post-doc in experimental methods, teaching assistant for this course in previous years. On behalf of Pr. Sverre Steen.

Objectives of this lecture:

- Give you an overview of the most important methods of data analysis in use in experimental marine hydrodynamics
- Give some examples of how to do data analysis using Matlab

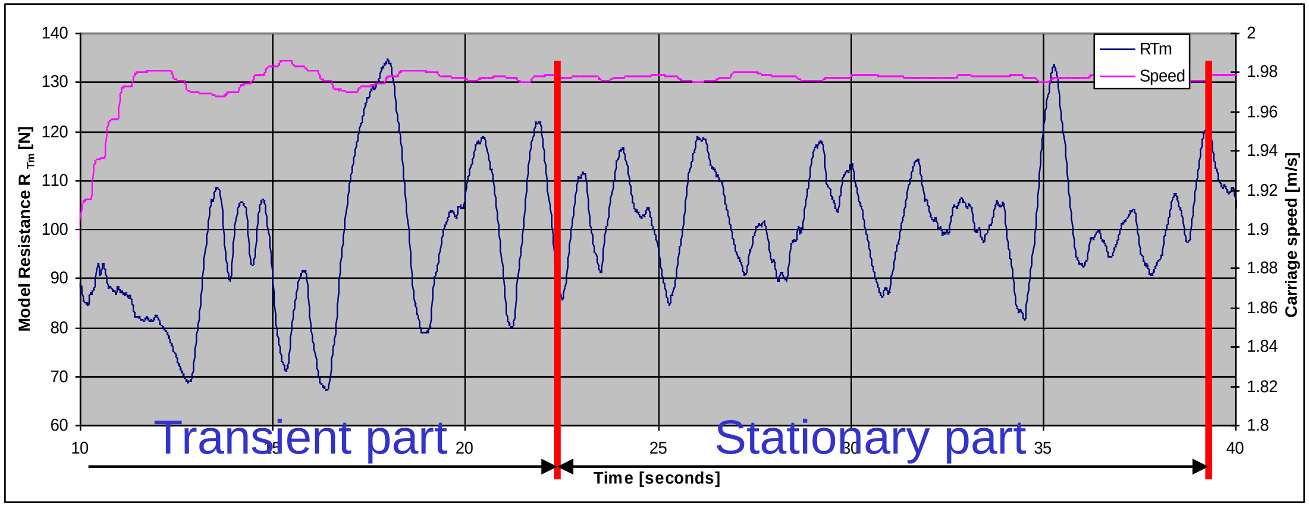
Covers Chapter 10 in the Lecture Notes

Contents

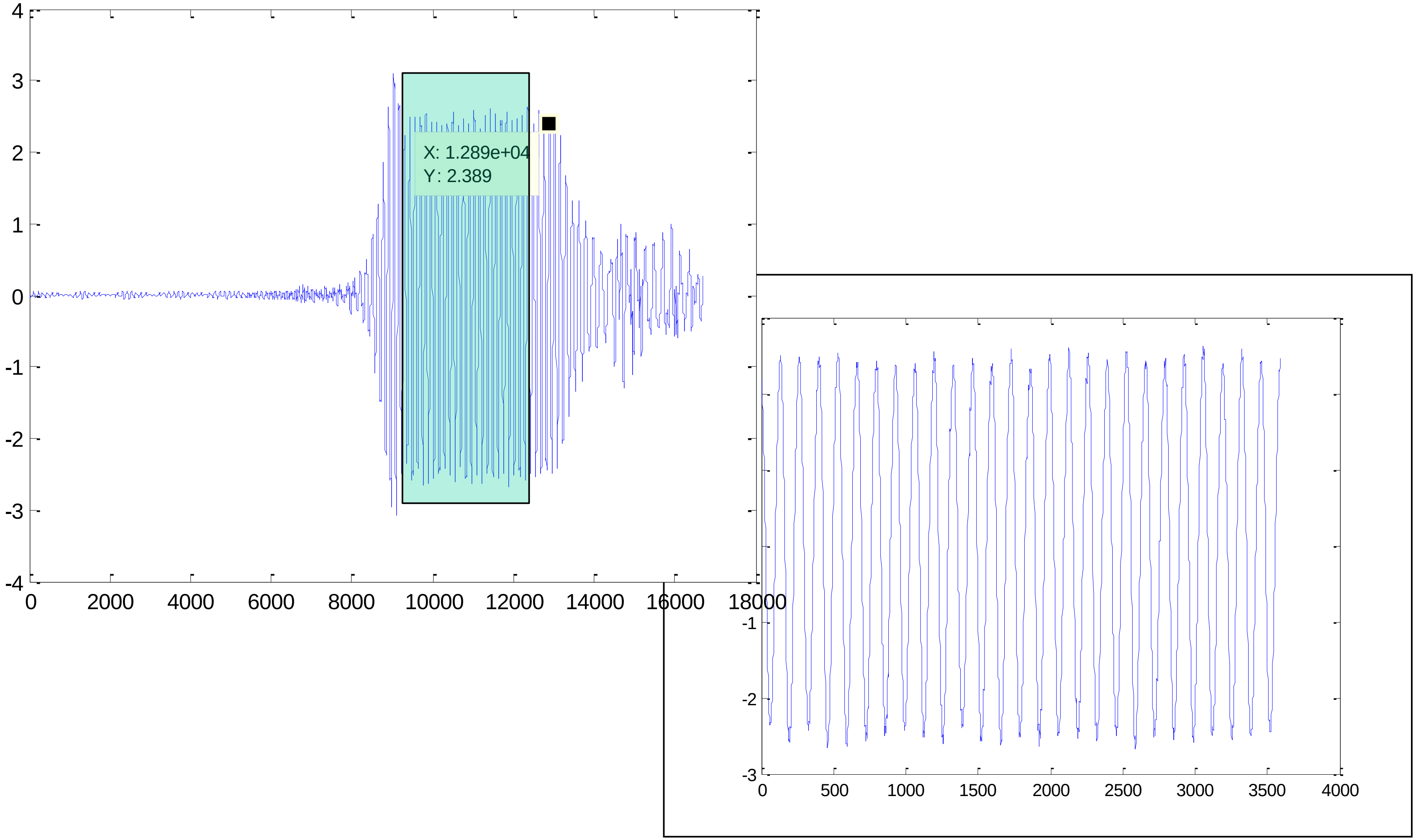
- Typical types of tests:
  - 1. Static tests
  - 2. Decay tests
  - 3. Regular wave tests
  - 4. Irregular wave tests
- Pre-processing data
- Filtering
- Spectral Analysis
  - Fourier transform
  - Power Spectral Density (PSD)
- Example

Static tests

- Tests expected to give a constant measured value
  - Example: Ship resistance, propulsion and open water tests
- Only the mean value is used in further analysis
- Take care to avoid transient effects at start-up
- Notice that even for tests of stationary phenomena like ship resistance in calm water, there will be oscillations in the signal
  - To create a reliable average at least ten oscillations should be included in the time window
  - If the signal is polluted by oscillations at a single low frequency, an entire number of oscillations should be included in the time window

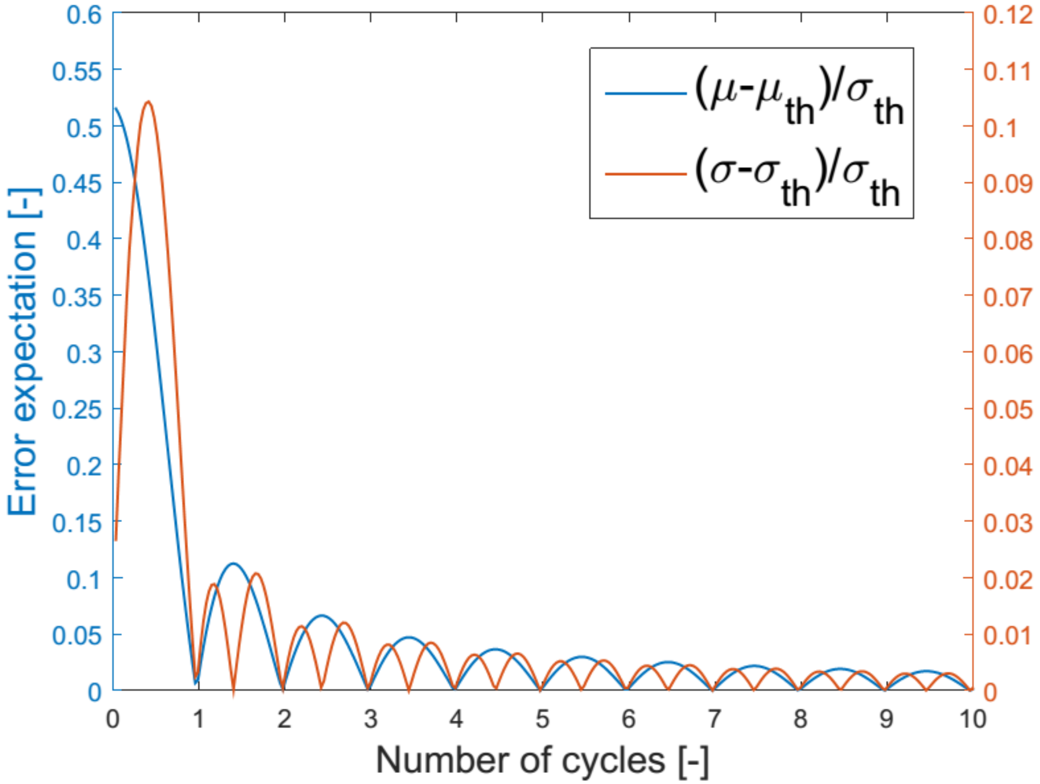


Pre-processing data in Matlab (for all tests)



Effect of the record length

- Sinusoidal wave $x = A\,sin(\omega t) + \mathrm{B}$
- Theoretical mean value $\mu_{th} = B$
- Theoretical standard deviation $\sigma_{th} = \frac{A}{\sqrt{2}}$
- The error expectation (the actual error depends on the initial phase) is 0 for entire numbers of cycles, else decreasing with number of cycles



Pre-processing data in Matlab (for all tests)

- Select the start and end times `tstart` and `tend`
- Interpolate to make data uniformly sampled

Selected time array → `t=tstart:dt:tend`

Uniformly sampled selected data → `x=interp1(t0,x0,t)`

Raw data and time arrays

- Clean data. Equipment limitations (especially in MC lab) lead to:
  - Erroneous data: Infinite (very large) or NaN (not a number).
  - Missing data: 0. Can occur for a somewhat long period of time and thus affects the results even if the mean value is small, even 0.

Pre-processing data in Matlab (for all tests)

- The data can be cleaned by the function:

Original data (uniformly sampled), row vector.

```
xclean=clean_data(x’,CrtSTD,CrtCONV)
```

Cleaned data, row vector

Iterative outlier criterion

Convergence criterion

Play around with these criteria to get the desired result

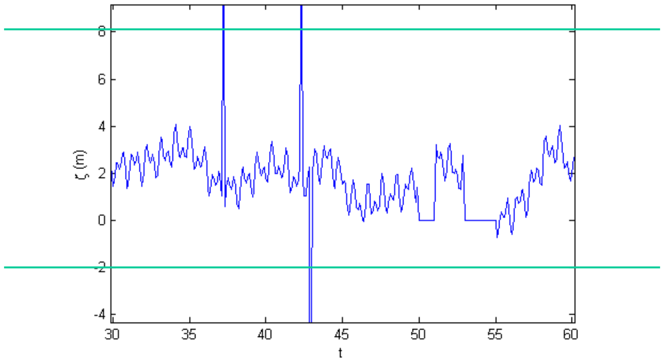
- Home made function. Tested on a limited number of time series only. Yet, always check the results! Modifications and suggestions are welcome.
- Smoothen x using `smooth(x,round(fs/fx)+1)` if sampled at fx<fs (stair-like signal)
- `clean_data` function is found in the Resource-section of the TMR7 webpage and at the end of this presentation

Pre-processing data in Matlab (for all tests)

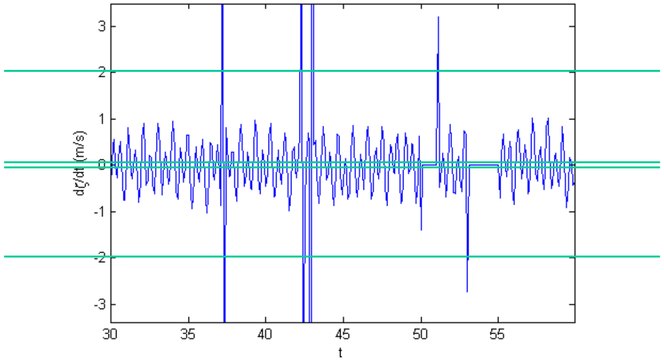
How `clean_data` works

μ : Mean value

σ : Standard deviation



Signal



Derivative

1) If

$$|x_i - \mu_x| \geq CrtSTD * \sigma_x$$

Tuning parameter

- CrtSTD > 1
- CrtSTD should be large when signal has uneven amplitudes (if too small, cleaning can affect valid parts of the signal)

Or

$$|\dot{x}_i| \geq CrtSTD * \sigma_{\dot{x}}$$

Or

$$|\dot{x}_i| \leq \frac{\sigma_{\dot{x}}}{10 * CrtSTD}$$

Then

2) **Replace** x_i by a linear interpolation of the nearest valid points

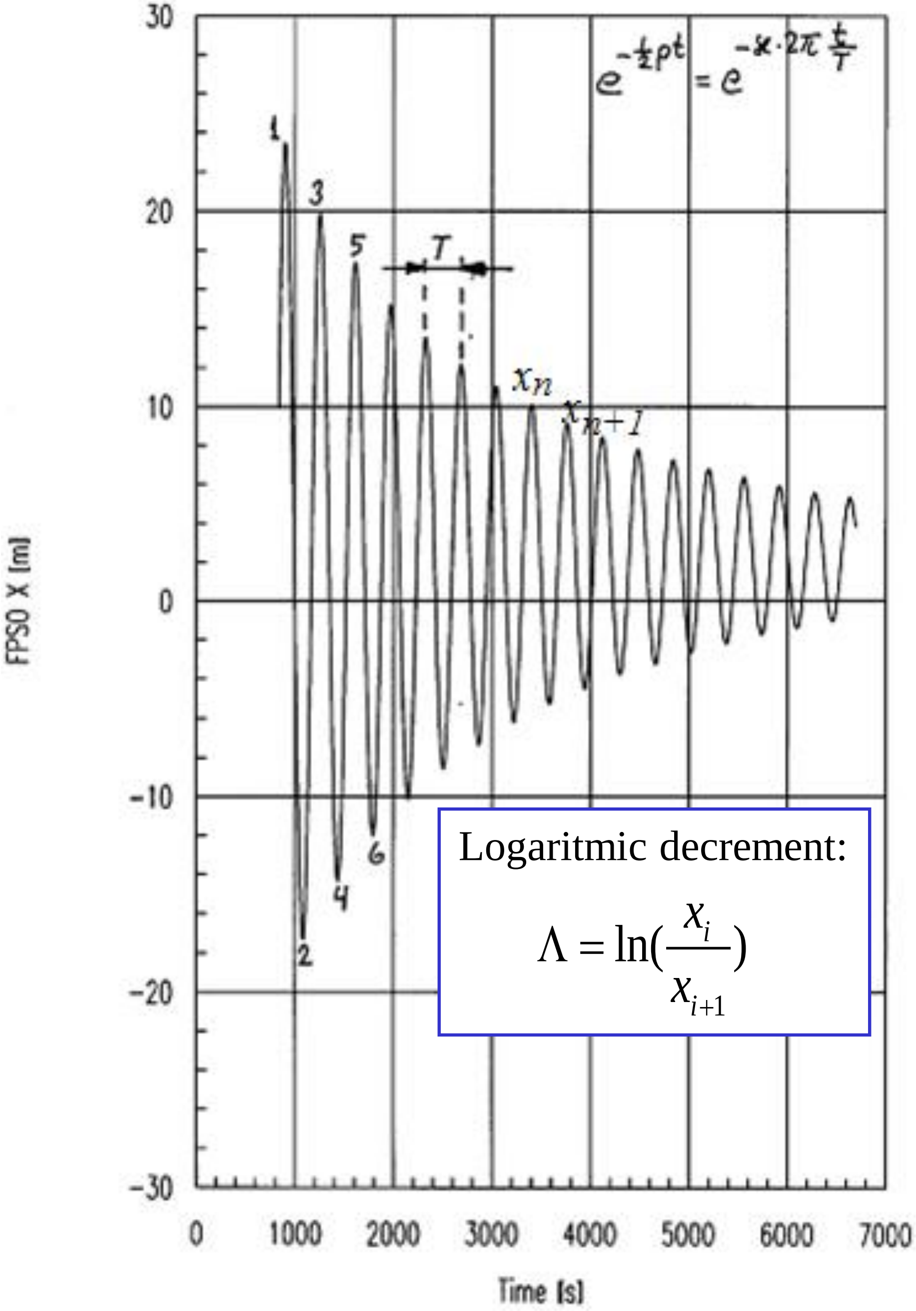
Less error is induced by **keeping** corrupt points than simply **removing** them!

3) Recompute μ_x and σ_x and iterate until it has converged:

$$\frac{\left|\sigma_{x_n} - \sigma_{x_{n-1}}\right|}{\sigma_{x_{n-1}}} \leq 0.1$$

Decay Tests

- Model is oscillated and then released, and response is measured
- Provides information about natural period and linear and quadratic damping terms
- Very useful for lightly damped degrees of freedom (system dependent). For ships:
  - Well suited: Natural period and damping in roll, horizontal motions
  - Difficult, but possible: pitch
  - Close to impossible: heave



Logaritmic decrement:

$$\Lambda = \ln(\frac{x_i}{x_{i+1}})$$

Analysis of decay tests

The damping ratio:

$$\xi = \frac{p}{p_{cr}} = \frac{p}{2M\omega_0}$$

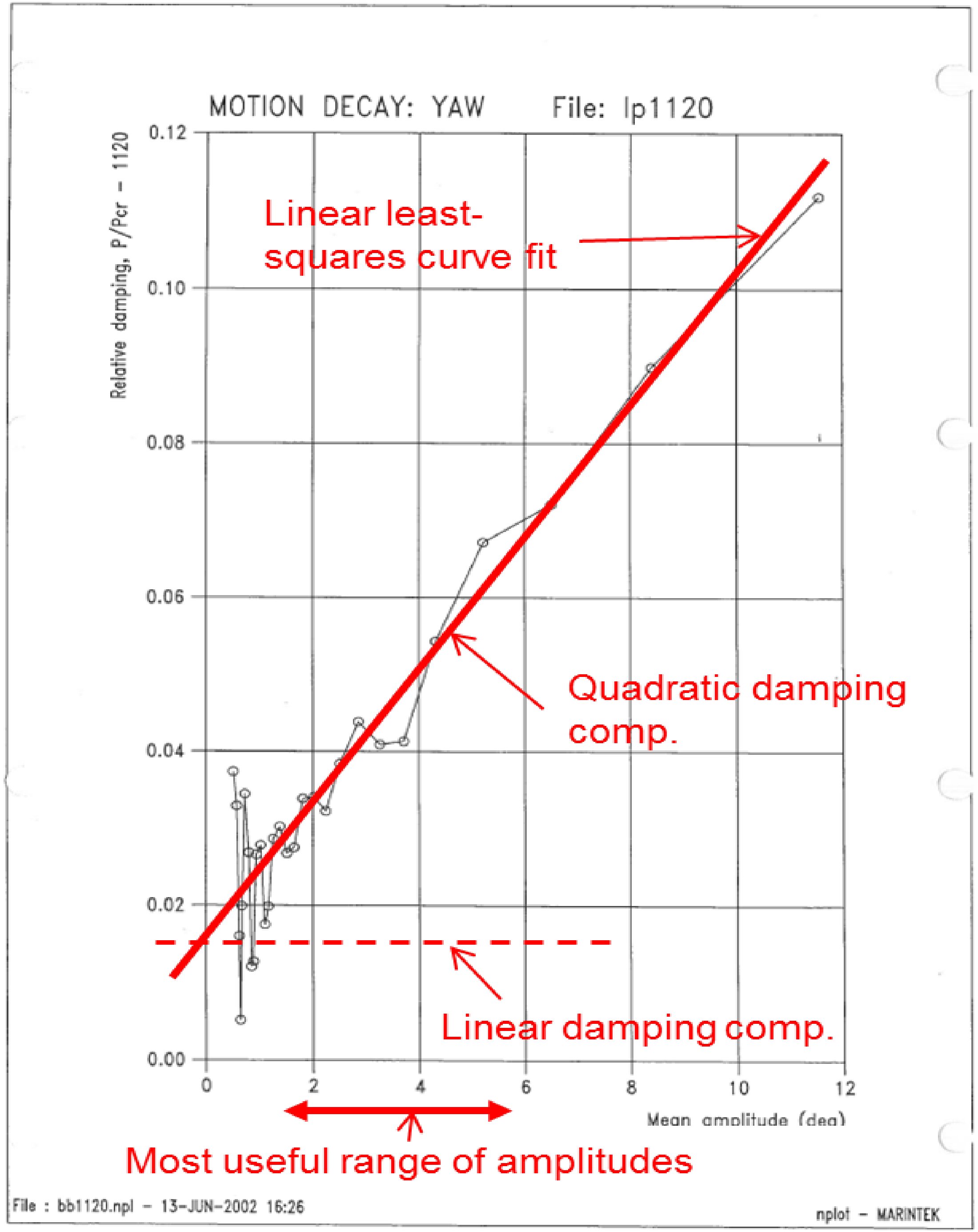
For low damping ratios (ξ<0.2): $\Lambda = 2\pi\xi$

The logarithmic decrement:

$$\Lambda = \ln(\frac{x_i}{x_{i+1}})$$

Equivalent damping:

$$p_{EQ} = 2M\omega_0\xi = \frac{2C\xi}{\omega_0}$$



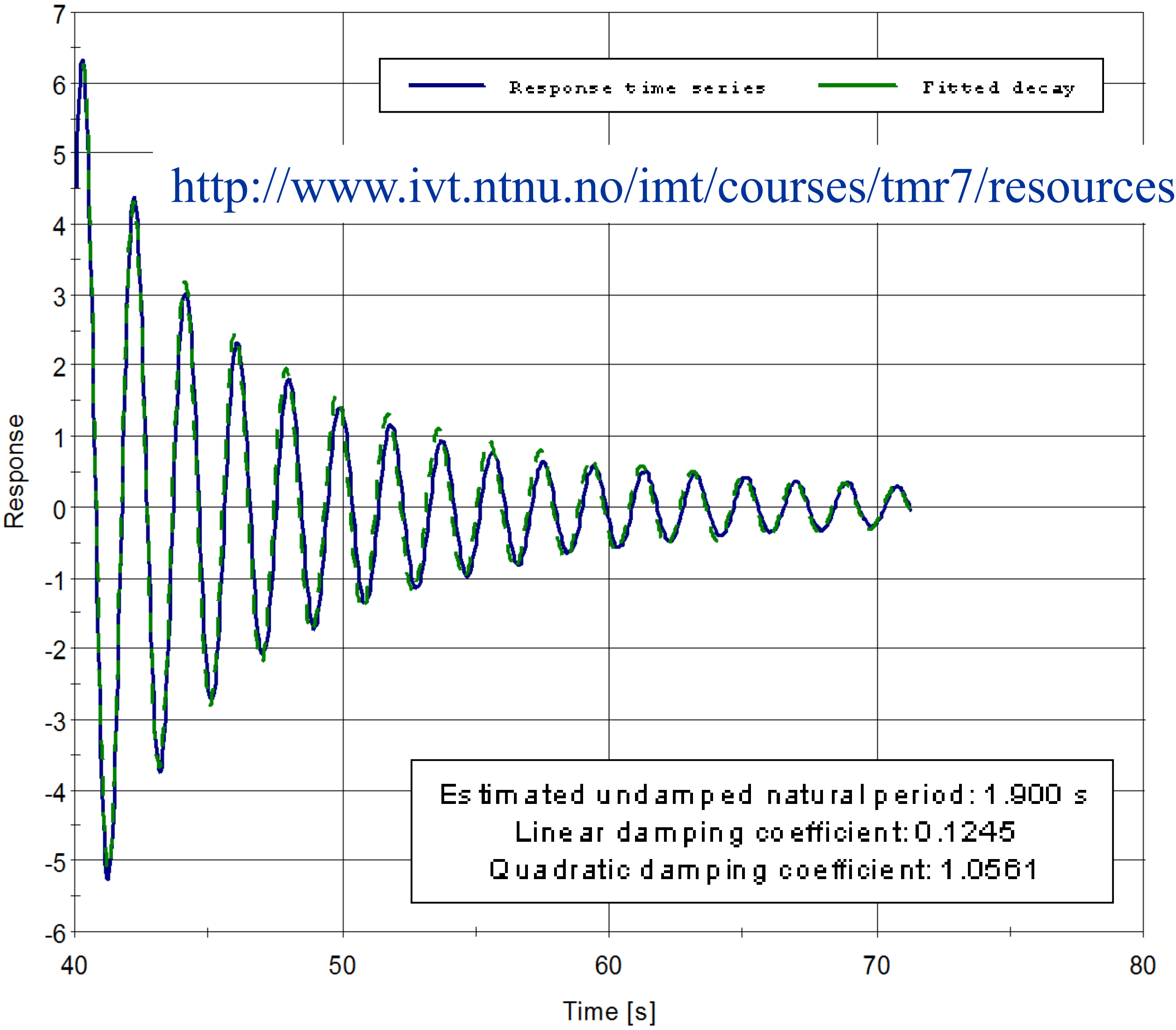
Linear least-squares curve fit

Quadratic damping comp.

Linear damping comp.

Most useful range of amplitudes

Alternative analysis of decay tests
- fitting of equivalent theoretical system



Estimated undamped natural period: 1.900 s
Linear damping coefficient: 0.1245
Quadratic damping coefficient: 1.0561

http://www.ivt.ntnu.no/imt/courses/tmr7/resources/decay.pdf

Decay analysis in Matlab

- Use the function `findpeaks` to measure amplitudes and periods

Amplitudes

Times

```
[PKS,LOCS] = findpeaks(x,t)
T=diff(LOCS)
Gamma=log(PKS(1:end-1)/PKS(2:end))
```

Periods

Decrement

Filtering

- Noise is undesirable in measurements
  - Impairs accuracy at frequencies of interest by folding (aliasing) if

$$Noise\ frequency > Nyquist\ frequency = \frac{Sampling\ frequency}{2}$$

  - Impairs readability of time series
  - Impairs accuracy of statistical properties of the signal (increases standard deviation, modifies mean value for low frequency noise)
- Two types of noise
  - Measurement noise: Unphysical noise specific to sensor (e.g. grid frequency). Typically removed by low-pass filtering in the hardware, i.e. prior to data acquisition.
  - Process noise: Undesired dynamics of the system
    - Transients: Decay of motion of undesired degrees of freedom. Typically low frequency: High-pass filter (also removes mean value)
    - Structural vibrations: excitation of off-interest eigenfrequencies of the system. Typically high frequency: Low-pass filter
    - Applied in the post-processing phase (by you!)

Filtering, cont.

How does it work?

The gain of the filter’s transfer function attenuates some parts of the frequency content of the signal.

$$G(\omega) = |H(j\omega)|$$

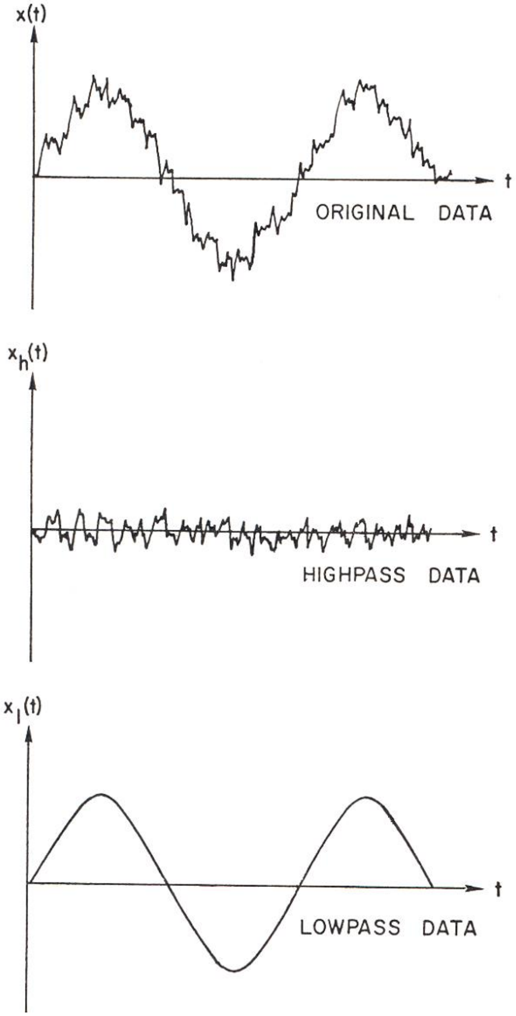
$$\underset{\underset{\text{Filtered signal}}{\uparrow}}{x_{filit}(t)} = \underset{\underset{\text{IFFT}}{\uparrow}}{\mathfrak{F}^{-1}}\{\underset{\underset{\text{Gain}}{\uparrow}}{G(\omega)} * \underset{\underset{\text{FFT of the signal}}{\uparrow}}{\mathfrak{F}}\{x(t)\}(\omega)\}(t)$$

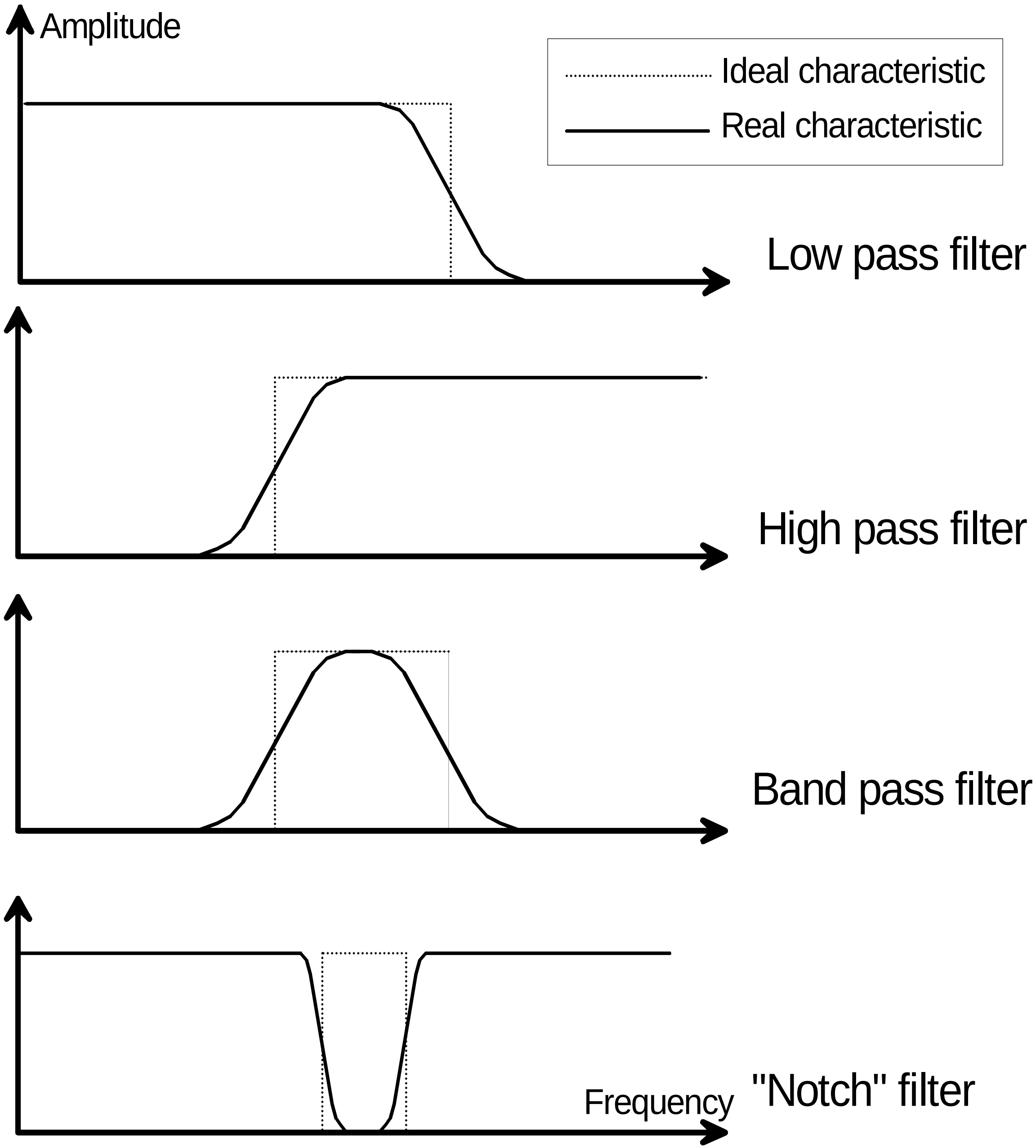
- In the frequency domain, no difference is made from 2 different processes having the same frequency

  ➔ In order for filtering to be successful, undesired processes should have a distinct frequency content from that of the studied process.
- $G(\omega)$ must be continuous for the IFFT to exist.

  ➔ The attenuation evolves gradually with the frequency. A sharp cut in the frequency content of a signal is not possible with low-order filters.

Various Filters





Low pass filter

High pass filter

Band pass filter

"Notch" filter

Filtering, cont.

Digital Butterworth filters:

- Most commonly used filters for this kind of application. Hardware filter=Butterworth filter order 4
- Described by its transfer function (in practice in the discrete domain)

$$H(z) = \frac{b(1)+b(2)z^{-1}+\cdots+b(n+1)z^{-n}}{1+a(2)z^{-1}+\cdots+a(n+1)z^{-n}}$$

- Designed in Matlab by

`'low'`	low-pass filter	filters frequencies > cutoff freq.
`'high'`	high-pass filter	filters frequencies < cutoff freq.
`'bandpass'`	band-pass filter	filters frequencies outside the cutoff freq interval.

Order of the filter

```
[b,a]=butter(order,wstar,'ftype')
```

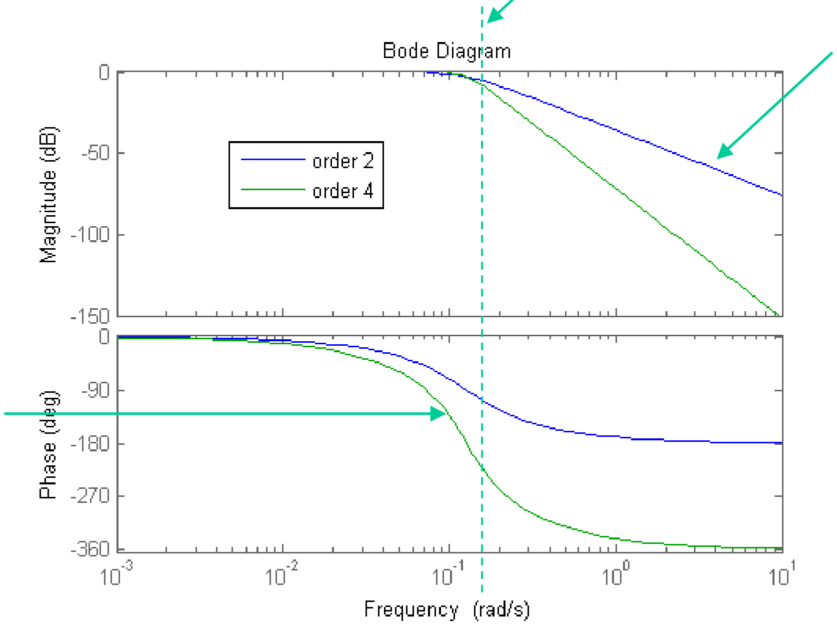
$$w^* = \frac{Cutoff\ frequency\ (Hz)}{Nyquist\ frequency} \qquad \frac{1}{2 * time\ step}$$

Normalized cutoff frequency
(Or interval of frequencies for bandpass filter)

Filtering, cont.

The filtering effect is best described by Bode diagrams of the filter’s continuous transfer function

```
[b,a]=butter(order,wstar,’low’)
Figure()
Bode(d2c(tf(b,a,dt)))
```



Cut-off frequency

Slope in gain reduction:
- = «filtering strength»
- Increasing with the order
- Increasing with frequency (for a low-pass filter) from cut-off frequency

Filtering induces a phase shift in the signal, increasing with order and frequency

➔ The cut-off frequency should be higher than the undesired frequencies, but lower than the frequencies of interest.
Else the signal will be badly filtered or **the amplitude attenuated**!

Filtering, cont.

- A so-called “spectral gap” is needed for efficient filtering
  = No energy in the spectrum around the cut-off frequency

If this is not the case, uncertainties will be introduced, take note of them!

- To avoid phase shift (improves readability in time domain plots), the time series are filtered first forward, then backward (symmetric or “zero-phase” filtering). This is not possible in real-time.

In Matlab:

Original data (uniformly sampled)

```
xfilt=filtfilt(b,a,x)
```

Filtered data

Digital filter coefficients

Aims of analysis of regular wave tests

- Response amplitude
- Response amplitude operator (transfer function in frequency domain)
  - Gain = Response amplitude/wave amplitude
  - Phase angle (between wave at reference location and response)
- Response frequencies
  - In addition to wave frequency, nonlinear excitation of the natural frequencies of the system

Reminder: Take care to leave out transient response at the start of the time series!

Analysis procedure for regular wave tests

Time-domain visual analysis

- Global assesment of the validity of the test
- First manual estimation of main period and amplitude

Frequency-domain visual analysis

- Global assesment of the validity of the test
- Overview of response frequencies, estimation of the noise level
- Design of possible filtering and checking filtering results

Computing Response Amplitude

- Divided by wave amplitude = Gain of RAO (Response amplitude operator)
- From standard deviation of noise-free signal

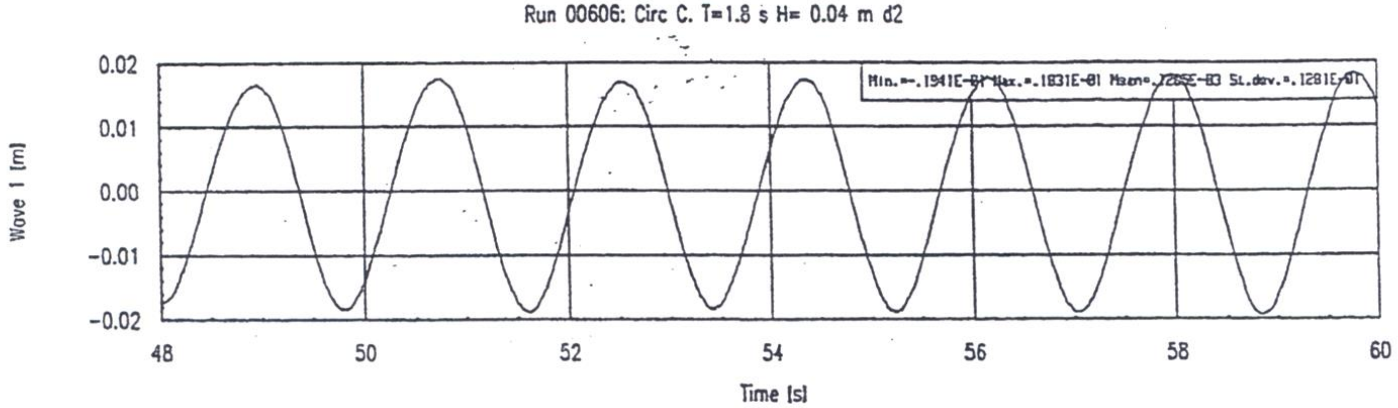
```
Amp=sqrt(2)*std(x)
```

- Average of peaks or $\frac{min+max}{2}$ are less accurate methods

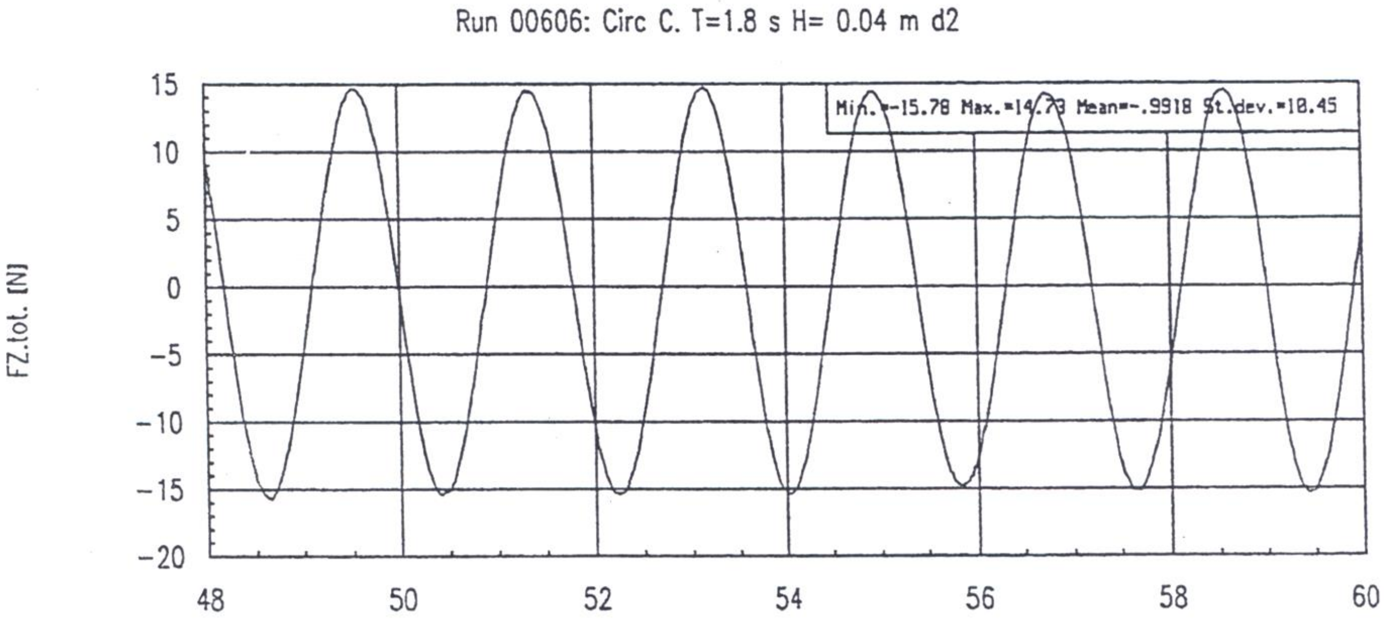
Computing Response Phase

- Fourier series analysis. Gives both phase and amplitude at a specified frequency (wave frequency) and its harmonics.
- Compared with incident wave: gives the total RAO (frequency-domain transfer function)
- Possible to extend to multiple frequencies (sum of sines)

a:

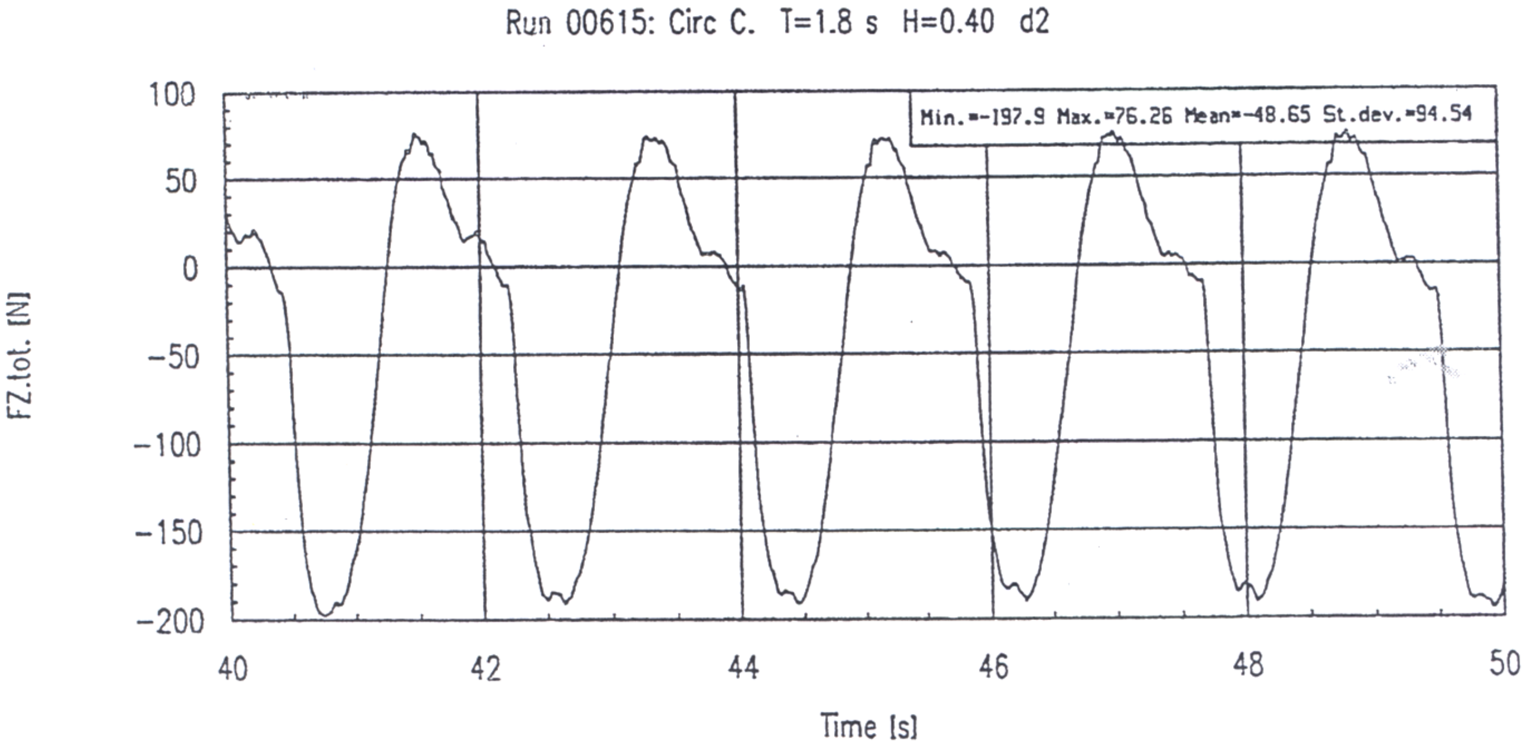


b:



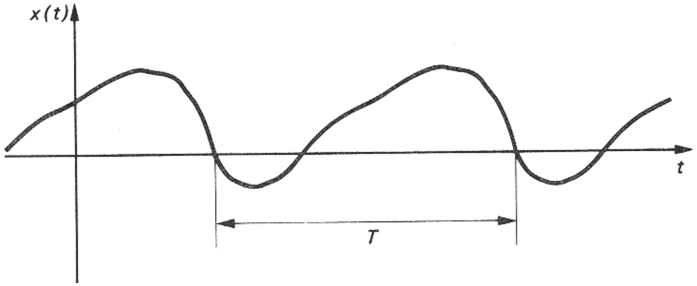
Chn no	Chn Name	Chn Unit	Ampl. 0. harm.	Ampl. 1. harm.	Ampl. 2. harm.	Period 1. harm. sec.	Period 2. harm. sec.	Freq. 1.harm Hz	Freq. 2.harm Hz	RAO 1. harm.	Rel.Phase 1. harm. degr.
1	Wave 1	m	-.7834E-03	.2140E-01	.8533E-03	1.807	.9037	.5533	1.107		
18	FZ.tot.	N	-1.017	14.96	.7834	1.807	.9037	.5533	1.107	699.1	33.05

C:



Chn no	Chn Name	Chn Unit	Ampl. 0. harm.	Ampl. 1. harm.	Ampl. 2. harm.	Period 1. harm. sec.	Period 2. harm. sec.	Freq. 1.harm Hz	Freq. 2.harm Hz	RAO 1. harm.	Rel.Phase 1. harm. degr.
1	Wave 1	m	-.7834E-03	.2140E-01	.8533E-03	1.807	.9037	.5533	1.107		
18	FZ.tot.	N	-1.017	14.96	.7834	1.807	.9037	.5533	1.107	699.1	33.05
18	FZ.tot.	N	-49.87	129.9	36.74.	1.787	.8935	.5596	1.119	661.8	18.57

Fourier series Analysis



- Goal: Extracting the linear component of the response to regular waves and deriving the gain and phase of the RAO at a given frequency (or period).
- *A <u>periodic</u> signal with period T can be fully described by an infinite sum of harmonic components, called Fourier series*:

$$f(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi kt}{T}\right) + b_k \sin\left(\frac{2\pi kt}{T}\right)$$

With coefficients defined as:

$$a_0 = \frac{1}{T}\int_0^T f(t)dt$$

$$a_k = \frac{2}{T}\int_0^T f(t)\cos\left(\frac{2\pi kt}{T}\right)dt$$

$$b_k = \frac{2}{T}\int_0^T f(t)\sin\left(\frac{2\pi kt}{T}\right)dt$$

Fourier series Analysis (cont.)

- Fit a Fourier Series to a time series in Matlab:
  1. Open the Curve Fitting Toolbox (`>> cftool`)
  2. Choose Fourier from the model type list.
  3. Use Fit Options and number of terms to control the fit.
  4. Check if frequency (given by w) is correct
  5. Retrieve the a_1 and b_1 coefficients (linear terms).
  6. Calculate gain and phase:

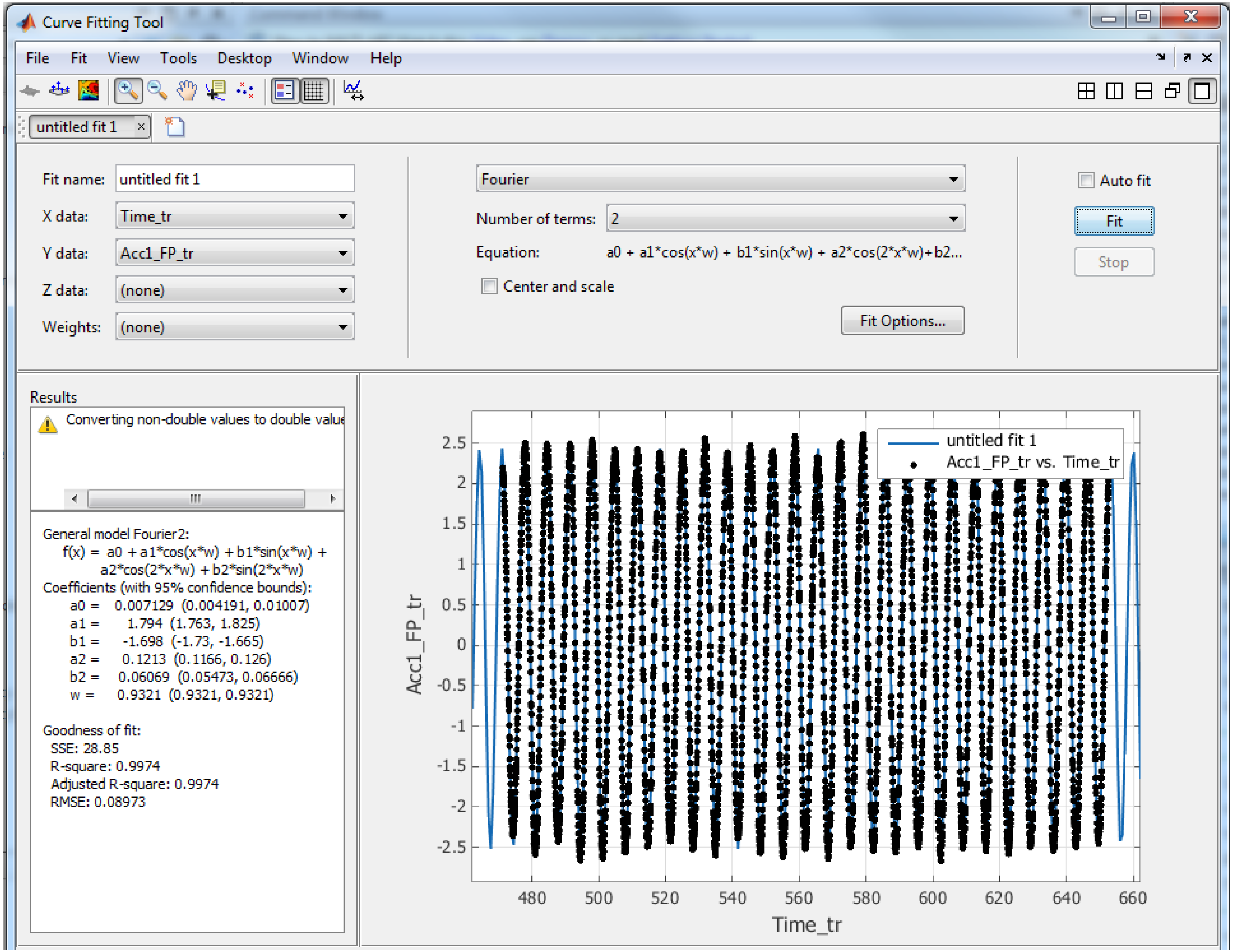
$$Gain = \sqrt{a_1^2 + b_1^2}$$

$$Phase = atan\left(\frac{b_1}{a_1}\right)$$

- Do it for both incident wave and response. Compute Gain and Phase of RAO by

$$Gain_{RAO} = \frac{Gain_{Response}}{Gain_{Wave}}$$

$$Phase_{RAO} = Phase_{Response} - Phase_{Wave}$$



Fit name: untitled fit 1
X data: Time_tr
Y data: Acc1_FP_tr
Z data: (none)
Weights: (none)

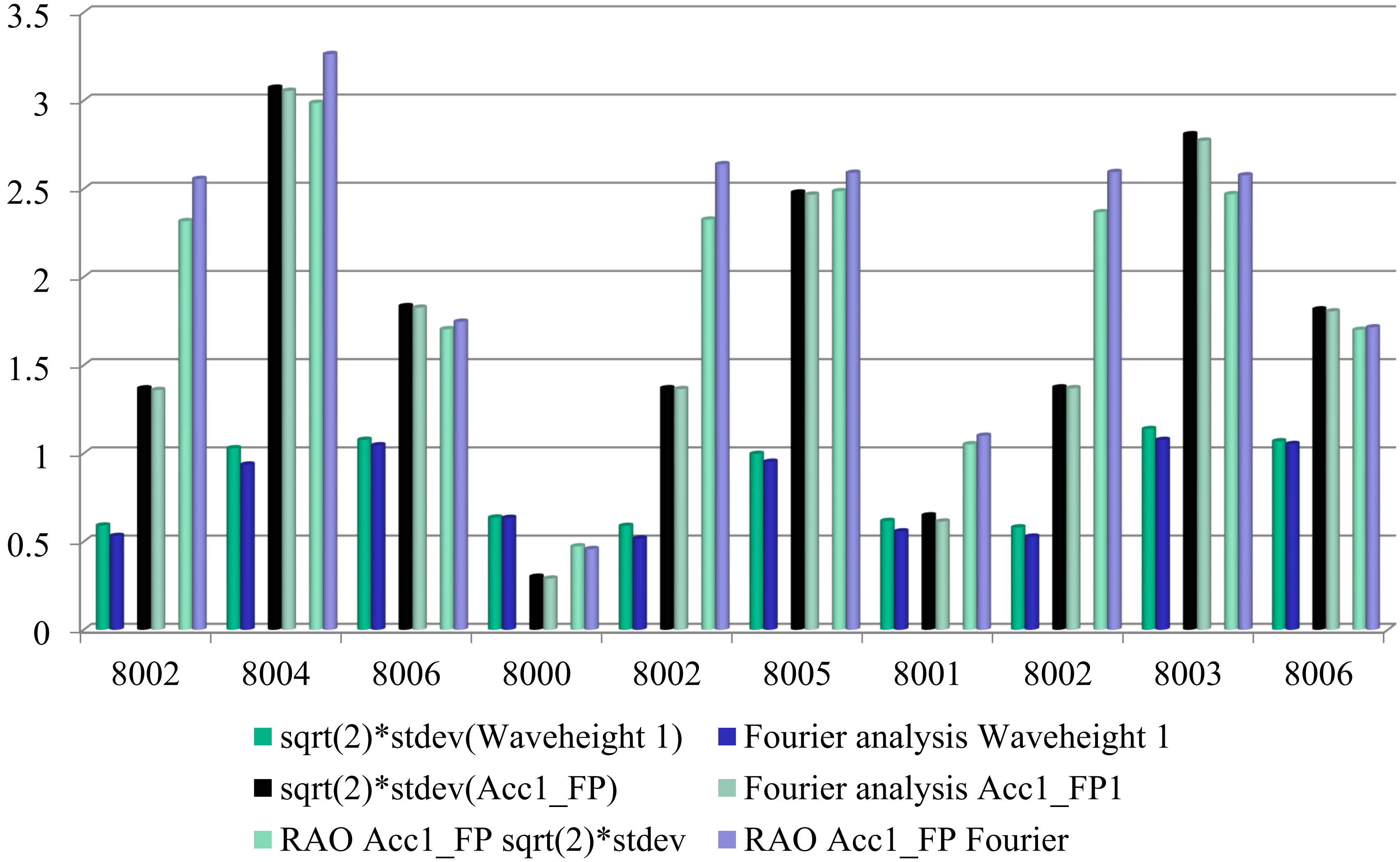
Fourier
Number of terms: 2
Equation: a0 + a1*cos(x*w) + b1*sin(x*w) + a2*cos(2*x*w)+b2...
☐ Center and scale
☐ Auto fit

```
Converting non-double values to double value

General model Fourier2:
    f(x) =  a0 + a1*cos(x*w) + b1*sin(x*w) +
            a2*cos(2*x*w) + b2*sin(2*x*w)
Coefficients (with 95% confidence bounds):
       a0 =    0.007129  (0.004191, 0.01007)
       a1 =       1.794  (1.763, 1.825)
       b1 =      -1.698  (-1.73, -1.665)
       a2 =      0.1213  (0.1166, 0.126)
       b2 =     0.06069  (0.05473, 0.06666)
       w =       0.9321  (0.9321, 0.9321)

Goodness of fit:
  SSE: 28.85
  R-square: 0.9974
  Adjusted R-square: 0.9974
  RMSE: 0.08973
```

Example of accuracy of estimating amplitude from st.dev. in regular waves



Irregular wave tests

- *Direct representation of the full scale sea condition*
- Typically wanted results:
  - Response spectra
  - Response spectrum parameters:
    - Spectral moments
    - Standard deviation
    - Peak period
  - Response amplitude operator (RAO)
  - Statistical results:
    - Max and min values,
    - Information about statistical distribution
    - Extreme value statistics (extrapolation using the statistical distribution)
    - Weibull plots etc.

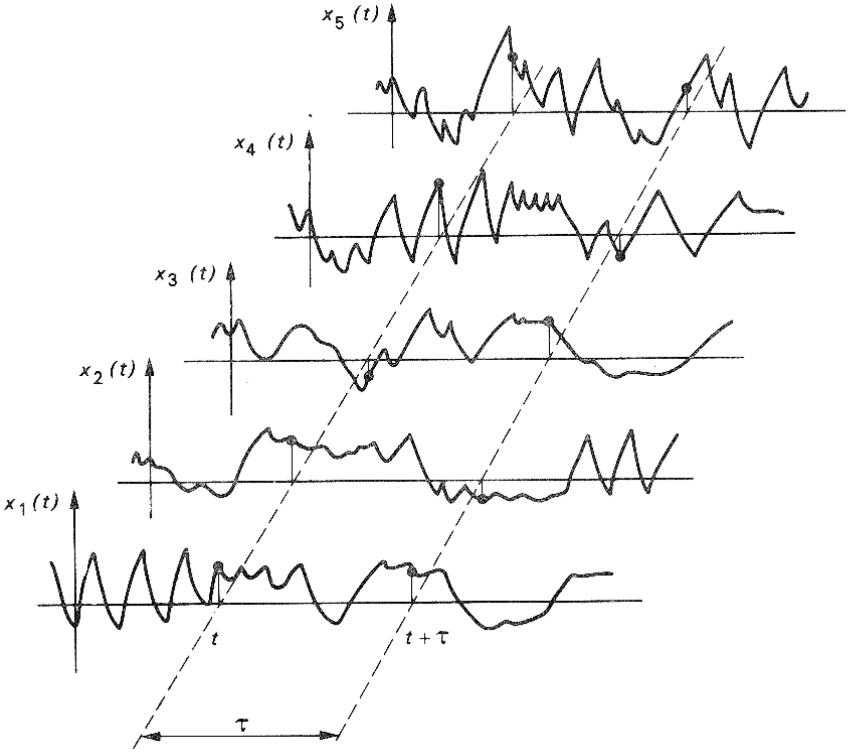
Properties of stochastic processes

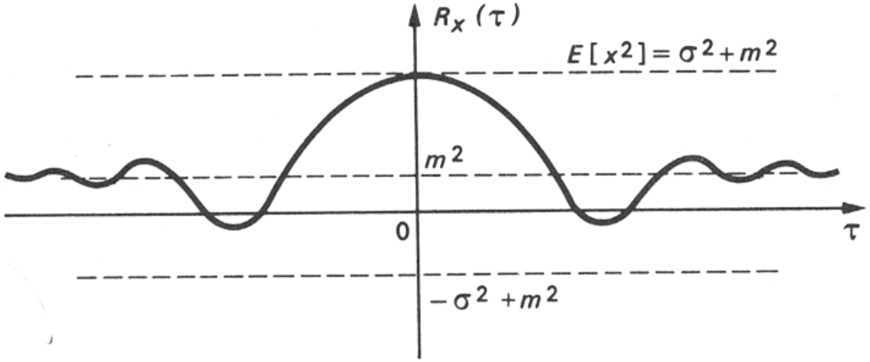
- Stationary: - Statistical properties constant with time
- Homogeneous: - Statistical properties constant in space
- Ergodic: - Time can be replaced by space as primary variable without changing the statistical properties

- The wave environment is commonly assumed to be a stationary, ergodic process
- This assumption greatly simplifies the analysis, and is a necessity for all established analysis methods
- It is not exactly true in a towing tank
  - Viscous damping: wave amplitude decreasing with distance to the wave maker, transmission of energy between frequencies. Significant for long towing tanks.
  - Wave reflection: Non-homogeneous and non-stationary effects.

Autocorrelation function

$$R_{xx}(\tau) = \underset{T \to \infty}{Limit} \left\{ \frac{1}{T} \int_0^T x(t)x(t+\tau)dt \right\}$$



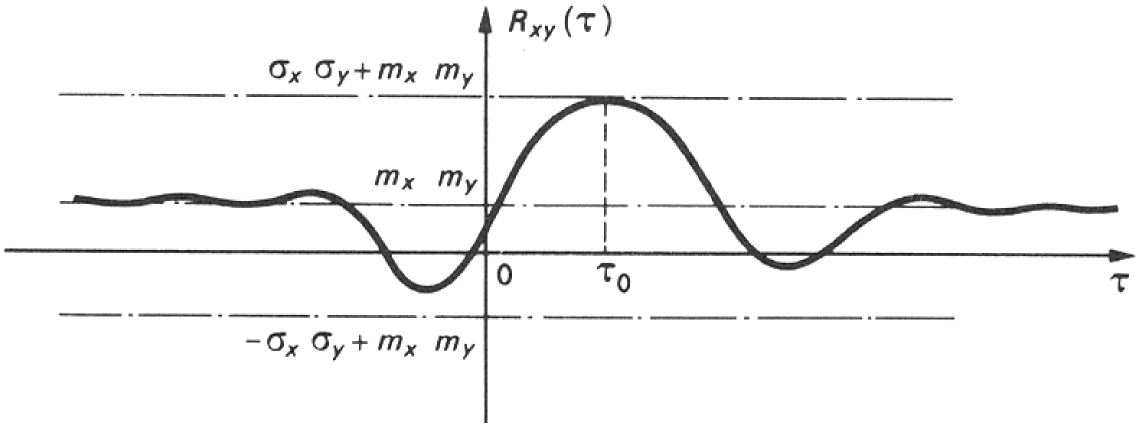


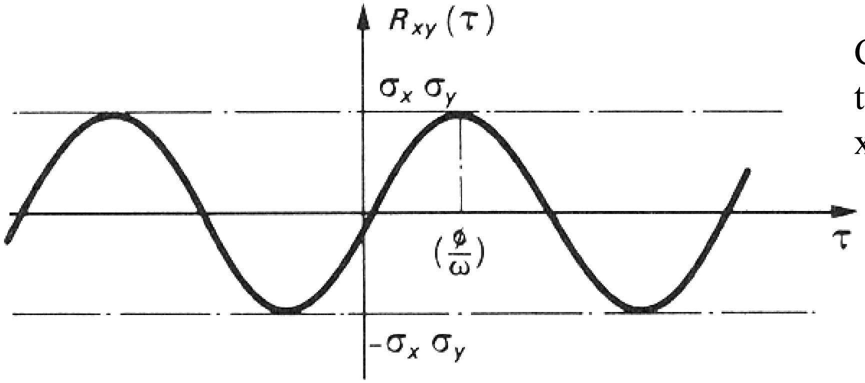
σ is the standard deviation and m is the average value

Figures from Newland (1984)

Cross-correlation function

$$R_{xy}(\tau) = \underset{T \to \infty}{Limit} \left\{ \frac{1}{T} \int_0^T x(t) y(t+\tau) dt \right\}$$





Cross-correlation function for two sine waves with y(t) lagging x(t) by an angle ϕ

Figures from Newland (1984)

Fourier transform

- In practice in the discrete (digital) domain: The continuous and ergodic signal $f(t)$ is sampled (assumed uniformly) over the record duration T at a rate f_s (in Hz), giving the time series $\{f_k\}$ with

$$k = 0,1,\dots,N-1; \;\; N = Tf_s$$

- The nth component of the Discrete Fourier Transform (DFT) of $\{f_k\}$ reads

$$\hat{f}_n = \frac{1}{N}\sum_{k=0}^{N-1} f_k e^{-\mathrm{i}(2\pi \mathrm{n}k/N)}$$

- $\{f_k\}$ can then be exactly retrieved by the Inverse DFT:

$$f_k = \sum_{n=0}^{N-1} \hat{f}_n e^{i\left(\frac{2\pi nk}{N}\right)}$$

Fast Fourier Transform (FFT)

- FFT is a computer algorithm for calculation of DFT. It is a core function of all digital data analysis.
- Conventional Discrete Fourier Transform (DFT) can also be implemented in a computer
  - DFT requires n^2 multiplications
  - FFT requires $n \cdot \log(n)$ multiplications
  - FFT is more accurate (due to fewer multiplications)
- FFT requires that N is a power of 2. Two solutions:
  - Truncate the signal to the nearest, lower power of 2
  - Augment the signal to the nearest, higher power of 2 by adding L zeros (or samples equal to mean value if non zero-mean signal).
    - Recommended for more accuracy.
    - Corrections must be applied to the output
    - Automatically done in the function `fft` in Matlab

Spectral Density

- Frequency-domain representation of the correlation.
    - Fourier Transform of Autocorrelation = Power Spectral Density (PSD), key tool to represent the frequency content (in terms of energy) of a signal
    - Fourier Transform of Cross-correlation = Cross Spectral Density (CSD)
- Time series are in theory of infinite length and aperiodic. A direct FT is ill-defined. The frequency content can only be obtained through FTs of correlation functions instead, leading to PSDs and CSDs.
- In the discrete domain and over a finite record time T, the periodicity requirement can be lifted. SDs are then in practice not computed using their original definition (i.e. from autocorrelation), but from the product of FFTs

Complex conjugate

$$S_{fg_n}(\omega_n) = \hat{f}_n^{\,*}\,\hat{g}_n$$

for each frequency $\omega_n = \frac{2\pi n}{T}$, $n = 0,1,\dots,N-1$. $f = g$ gives PSD

- From circular to linear frequency: $S(f) = 2\pi S(\omega)$

Meaning of spectral moments

- The n’th moments of the spectrum is defined as:

$$m_n = \int_0^\infty \omega^n S(\omega) d\omega$$

- Standard deviation of response: $\sigma = \sqrt{m_0}$
- Significant value of response: $x_{1/3} = 4\sqrt{m_0}$
- Average period of response: $T_1 = \frac{m_0}{m_1}$
- Average zero crossing period : $T_2 = \sqrt{\frac{m_0}{m_2}}$

Accuracy and resolution of PSDs

- The accuracy of the SD computation can be written as (Newland, 1984):

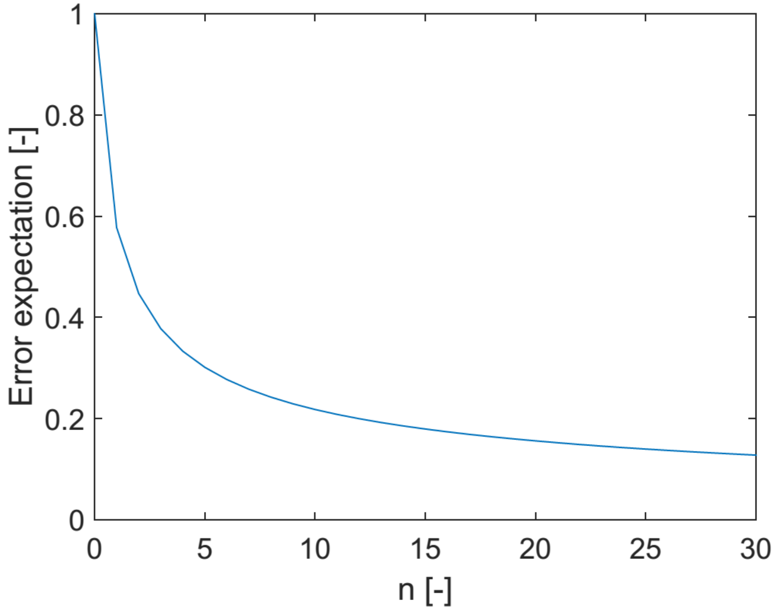
Standard deviation and mean value of PSD at frequency *f* in a set of records of length *T*

$$\frac{\sigma}{\mu} \cong \frac{1}{\sqrt{2n+1}}$$

Moving average smoothing with the *n* previous and *n* following frequencies

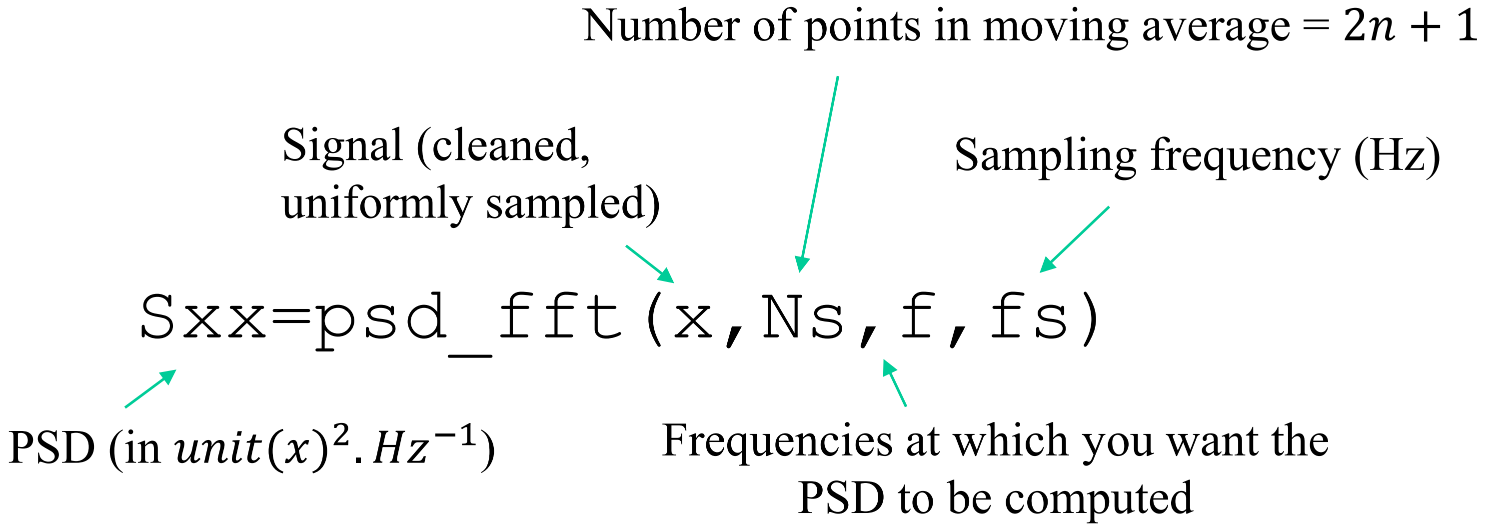
Assuming *1/T* is much lower (>5 -10 times) than *f*

- *n* determines the smoothness of the spectrum. Increasing smoothness increases accuracy, but decreases resolution.
- Assuming $\frac{1}{T}$ sufficiently small wrt *f,* the accuracy is not dependent on *T,* nor on *f* !
- Increasing T increases resolution and enables the study of lower frequencies.
- The effect of the record length on the standard deviation of sine waves can be applied to frequency components of PSDs



PSD in Matlab

- Many functions, hard to tune because complex underlying mathematics
- `psd_fft` is a home made function computing the PSD directly from the Fourier transform.



Number of points in moving average = $2n + 1$

Signal (cleaned, uniformly sampled)

Sampling frequency (Hz)

```
Sxx=psd_fft(x,Ns,f,fs)
```

PSD (in $unit(x)^2.Hz^{-1}$)

Frequencies at which you want the PSD to be computed

- Designed from: *An introduction to Random Vibrations and Spectral Analysis*, by *D.E. Newland* and The Mathworks website

  http://se.mathworks.com/help/signal/ug/psd-estimate-using-fft.html
- `psd_fft.m` is found in the Resource-section of the TMR7 webpage and in this presentation (next slide)

psd_fft.m

```
function [S,Sraw]=psd_fft(x,Ns,f,fs)
    %Calculate PSD from raw fft and smoothing. From Newland: “An introduction to
    % random vibrations and spectral analysis” and The Nathworks website
    % http://se.mathworks.com/help/signal/ug/psd-estimate-using-fft.html
    %x: signal
    %Ns: Number of points in moving average (=2*n+1, odd number)
    %f: desired output frequencies
    %fs: sampling frequency
    %S: PSD @ frequencies f
    %Sraw: Structure with field S=PSD and field f=frequencies as defined by fft
    Nt=floor(size(x,1)/2)*2;
    x=x(1:Nt,:);
    dt=1/fs; %Step size
    T=Nt*dt; %Record length
    S=fft(x); %Compute dft by fft. fft specificities (added zeroes) are handled
    % internally.
    S=2*dt/Nt*abs(S(1:Nt/2+1,:)).^2; %Compute PSD: one-sided (multiply by 2),
    % distributed (divide by sampling freq) average of fft (divide by number of
    % points) squared
    fS = 1/dt*(0:(Nt/2))/Nt; %Output frequencies, up to number of points/2 (higher
    % frequencies only show a folded version of low frequencies)
    for i=1:size(x,2)
        S(:,i)=[S(1,i)/2;smooth(S(2:end,i),Ns)]; %Smoothing by moving average
    end
    Sraw.S=S;
    Sraw.f=fS;
    S=interp1(fS,S,f); %Interpolation to desired output frequencies
```

PSD in Matlab, cont.

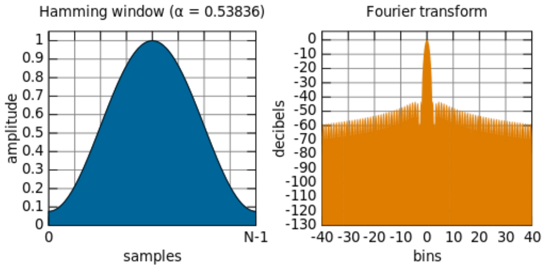
`pwelch` is the standard built-in Matlab function for PSD calculation.

- Default values often lead to inaccurate results.
- Excessively computationally demanding for long time series
- More accurate than `psd_fft` for short time series, because moving average smoothing diffuses uncertainties of low frequencies onto frequencies of interest.

```
Sxx=2*pwelch(x,Window,Noverlap,f,fs)
```

Change from two-sided to one-sided PSD

Number of overlapping samples between windows. Does not have a big influence. Window/10 is a good start.



The signal is segmented into «windows». The FFT is computed segment by segment which are then assembled to give the PSD.
The broader the window, the finer the spectrum. The narrower, the smoother. Adjust it to get a readable yet accurate spectrum (Use values from NFFT/2 to NFFT/10).

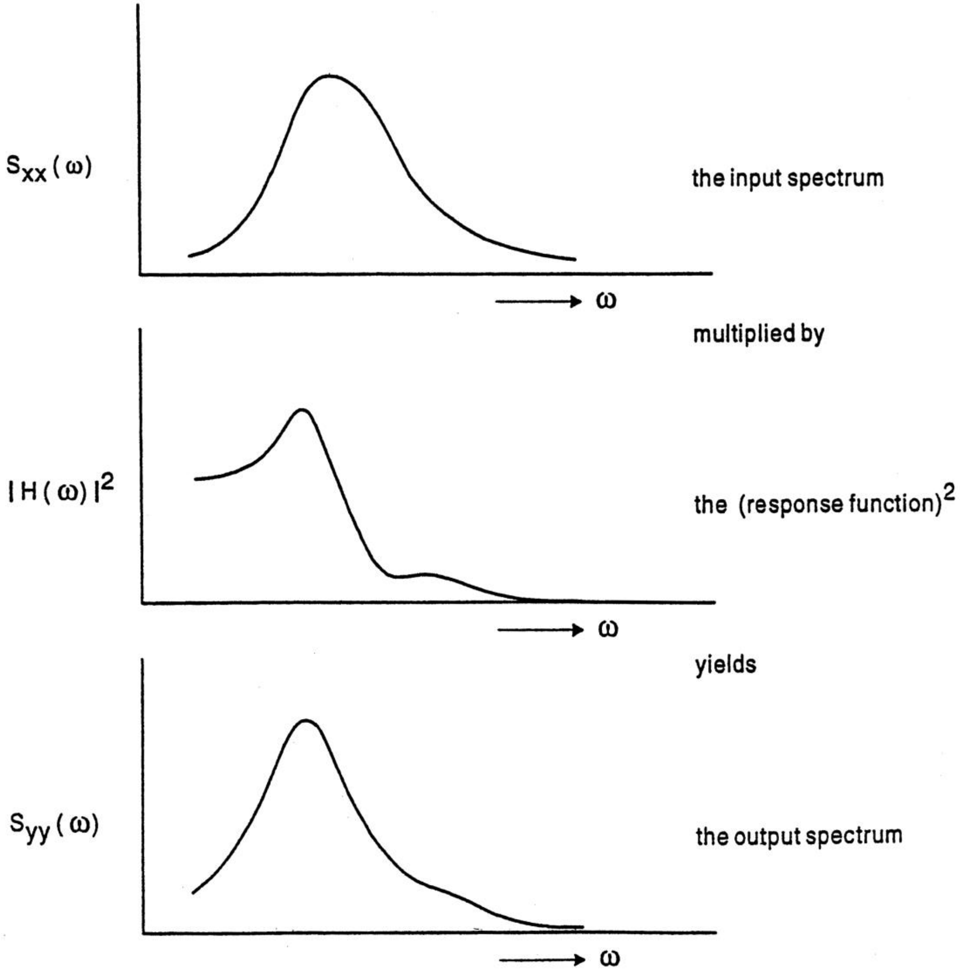
Transfer function in irregular waves (equivalent to RAO in regula waves)

Magnitude:

$$|H(\omega)|^2 = \frac{S_{yy}(\omega)}{S_{xx}(\omega)}$$

Phase can be obtained from CSD:

$$H(\omega) = \frac{S_{xy}(\omega)}{S_{xx}(\omega)}$$



Summary

- Static tests and pre-processing
  - The valid window of the time series to be analyzed must be sufficiently long
  - The data must be cleaned and uniformly sampled
- Filtering
  - Filter design in the frequency domain
  - Need for a spectral gap
  - Zero-phase filtering
- Regular wave tests
  - The amplitude should be calculated through the standard deviation of filtered signals
  - If the phase is desired, use Fourier Series
- Irregular wave tests
  - Spectral densities based on the Fourier transform are used for frequency domain analysis
  - Appropriate smoothing should be applied

Example of post-processing with Matlab: Irregular wave elevation

Generated from JONSWAP spectrum.

The following is artificially added:

- Erroneous and missing data
- Measurement noise
- Transients
- Mean offset

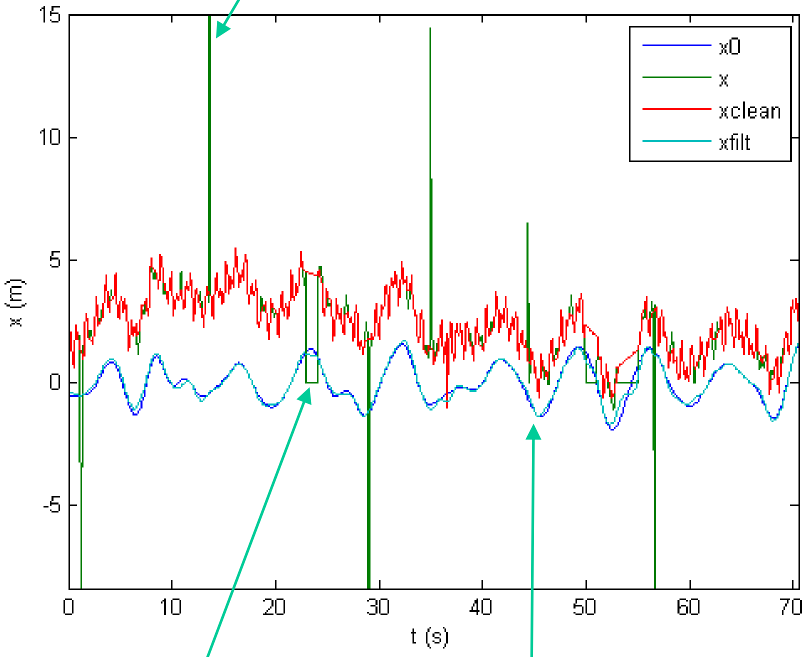
Example cont. : Matlab script

```
load(‘data.mat','x‘,’time’)             %Load wave elevation and time from file
duration=200;
dt=0.1;
t=0:dt:duration;
Nt=length(t);
xint=interp1(time,x,t);                 %Interpolate data
xclean=clean_data(xint,3,0.001);        %Clean data
cutoff=[0.3 4]/(2*pi);                  %Cut-off frequencies
fnyq=1/(2*dt);                          %Nyquist frequency
[b,a]=butter(4,cutoff/fnyq,'bandpass'); %Get filter coefficients
xfilt=filtfilt(b,a,xclean);             %Zero-phase filtering
df1=0.01;
df2=0.1;
f1=0.01:df1:0.99;                       %Small frequency step for low frequencies
f2=1:df2:10;                            %Large frequency step for high
frequencies
f=[f1 f2];
Sxx=psd_fft(xint-mean(xint),10,f,1/dt); %PSD of unfiltered data
Sxx_filt=psd_fft(xfilt-mean(xfilt),10,f,1/dt);     %PSD of filtered data

figure(1)
plot(t,[x0 xint xclean xfilt])                    %x0: original data generated
from JONSWAP

figure(2)
plot(w,jonswap,f*2*pi,Sxx/(2*pi),f*2*pi,Sxx_filt /(2*pi))
```

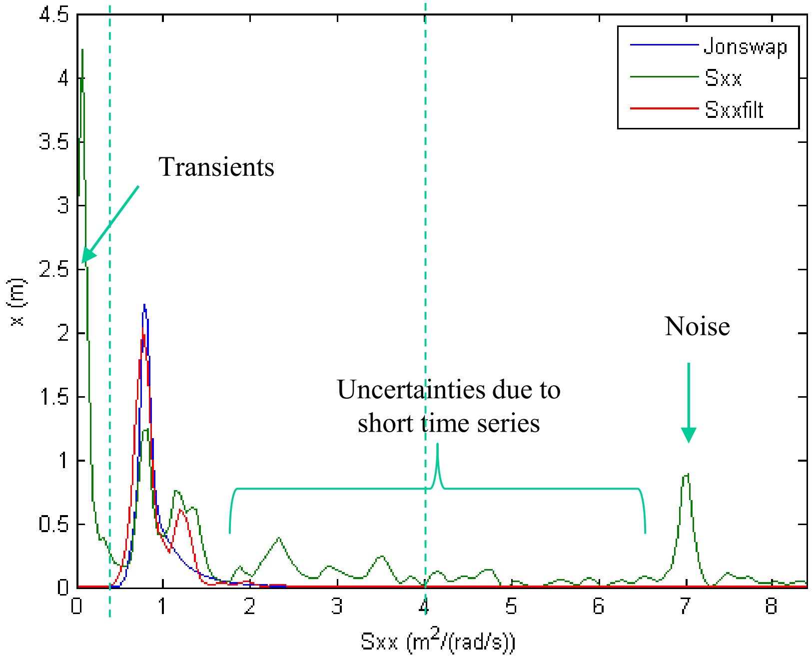
Example cont. : time and frequency domain plots



Outlier

Period of missing data

Band-pass filtering removes hig and low (including offset = 0 rad/s) frequencies



Cut-off frequencies

Transients

Uncertainties due to short time series

Noise

Uncomplete spectral gap: slightly uncertain filtering of the transients

Large spectral gaps allowing efficient filtering of the noise

Questions?

Teaching assistant:

bhushan.taskar@ntnu.no

Office D2.235

About this course:

valentin.chabaud@ntnu.no

Office G2.130

clean_data.m

```
function x=clean_data(data,CrtSTD,CrtCONV)

%Written by Valentin Chabaud. v3 - August
2015
%Removes erroneous values and outsiders
from time series

x=data';
sx=std(x);
mx=mean(x);
d=diff(x);


sd=std(d);
d=[d;d(end)];
% figure(3)
% plot([data';d])
std_prev=std(x)/CrtSTD;
N=10;
while abs((std(x)-std_prev)/std_prev)>CrtCONV
    flag=0;
    ind=[];
    for i=1:length(x)
        if abs(x(i)-mx)>sx*CrtSTD ||
abs(d(i))>sd*CrtSTD || abs(d(i))<sd/CrtSTD*0.1
            if flag==0
                flag=1;
                ind=[ind;[i 0]];
            end
        else
            if flag==1
                ind(end,2)=i;
                flag=0;
            end
        end
    end
    if(ind(end,end))==0
        ind(end,end)=length(x);
    end
    y=[ones(N,1)*x(1);x;ones(N,1)*x(end)];
    for i=1:size(ind,1)
        inttot=(1:length(y))';
        intrem=ind(i,1)+N:ind(i,2)+N;
        intfit=setdiff(inttot,intrem);
        z=y(intfit);
%         f = fit(intfit, z,
'smoothingspline','SmoothingParam', 0.1);
%         y(intrem)=feval(f,intrem);
        y(intrem)=interp1(intfit,y(intfit),intrem);
        x=y(N+(1:length(x)));
    end
    std_prev=std(x);
end

x=x';
```

Statistical distributions

- The probability distribution function, *P(x),* is the probability that a general value of the process *x(t)* is less than or equal to the value of x

$$P(x) = P(x(t) \leq x)$$

- The probability <u>density</u> function: $p(x) = \dfrac{dP(x)}{dx}$

- The probability that *a<x(t)<b* is given by the probability density function such that:

$$P(a \leq x(t) \leq b) = \int_a^b p(x)dx$$

Probability distributions used in the study of wave generated responses

- The distribution of the process itself, e.g. the distribution of the wave elevation x(t) and the measured response y(t)

⇒ Gaussian distribution

- The distribution of amplitudes; e.g. distribution of the wave amplitudes x_A and measured response amplitudes, y_A in the tests.

⇒ Rayleigh distribution

Rayleigh distribution of amplitudes

- Follows from the assumption that the elevation itself is Gaussian
- The cumulative distribution: $P(x) = 1 - \exp\left[-\frac{1}{2}\left(\frac{x-\mu_X}{\sigma_X}\right)^2\right]$
- Here is the mean or expected value of *x(t)* defined as:

$$\mu_X = E\left[x\right] = \int_{-\infty}^{\infty} xp(x)dx$$

- σ is the variance of *x(t)*, defined as:

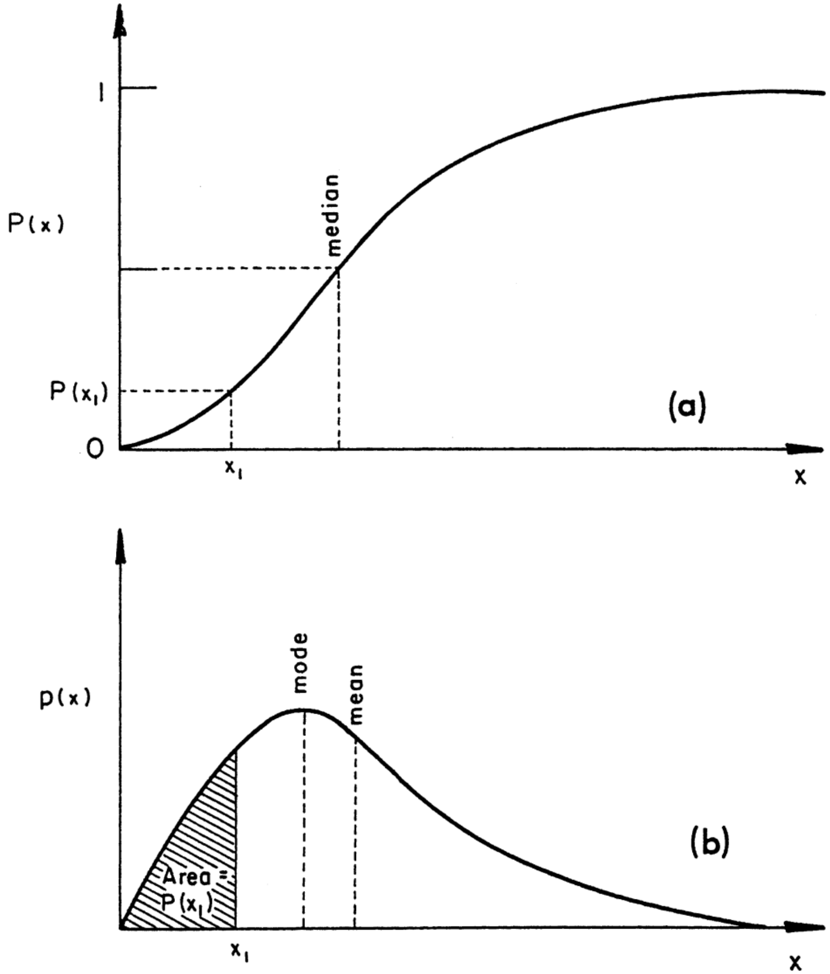
$$\sigma_X^2 = E\left[\left(x-\mu_X\right)^2\right] = E\left[x^2\right] - \mu_X{}^2$$

Rayleigh distribution

- For a measured time series with N samples the mean value and the variance are calculated as:

$$\mu_X = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma_X^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_X)$$



Non-linear response

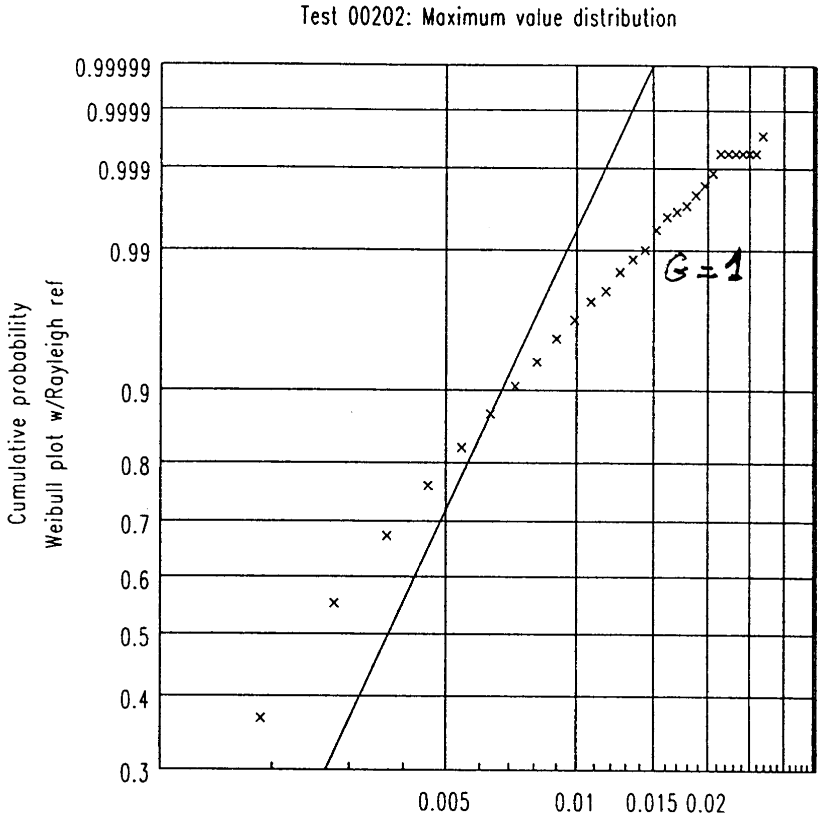
- The response *y(t)* follows a Rayleigh distribution only if it is a linear function of the wave elevation *x(t)*
- To describe non-linear response it is common to use the more general Weibull distribution:

$$P(x_A) = 1 - \exp\left[-\frac{1}{G}\left(\frac{x_A - \mu_X}{\sigma} \right)^G \right]$$

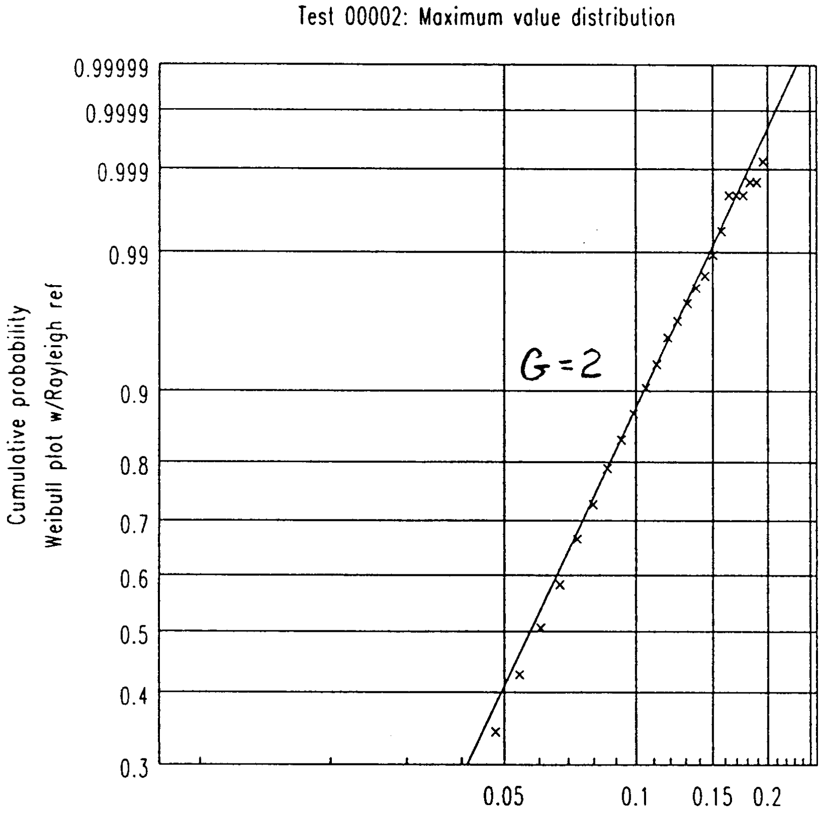
  - *G=2* gives the Rayleigh distribution
  - *G=1* gives the Exponential distribution

Weibull plots

P(x_A)-axis plotted as ln[−ln(1−P(x_A))]



Horisontal axis: response amplitude y_A



Horisontal axis: response amplitude y_A

Significant values

- Significant maxima:
  - the mean of the highest one-third of the crest-to-zero values of x_A,
- Significant minima:
  - the mean of the highest one-third of the trough-to-zero values of x_A,
- Significant double amplitude:
  - mean of the highest one-third of the maximum to minimum values of x_A

Maximum/Minimum Values

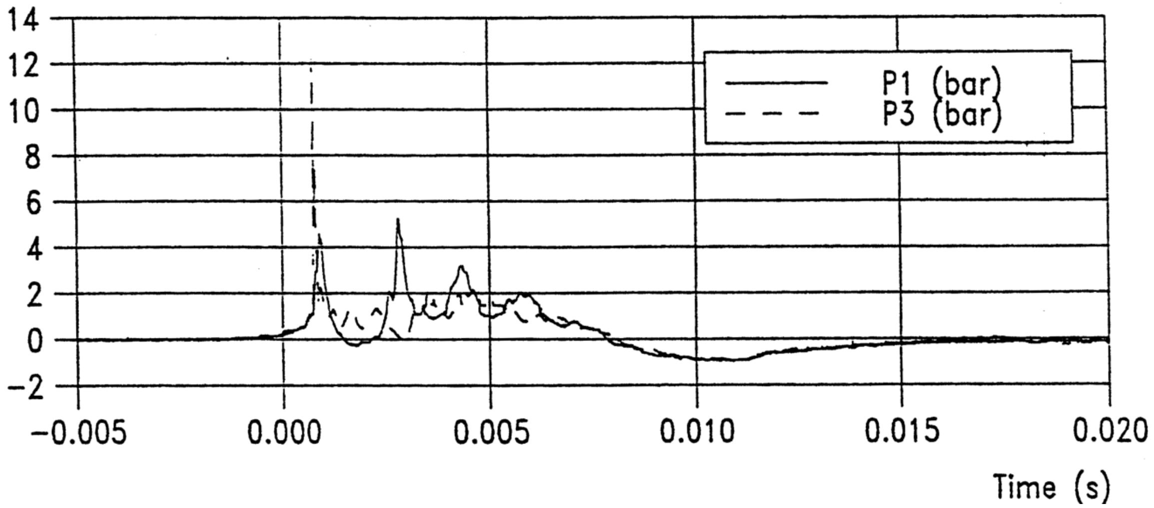
- Maximum Value:
  - Measured maximum value in the record
- Minimum Value:
  - Measured minimum value in the record
- Largest double amplitude:
  - Largest measured crest to trough value in the record

Examples of special analyses:

- Slamming
- Sea sickness incidence
- Ventilation and air injection to waterjets

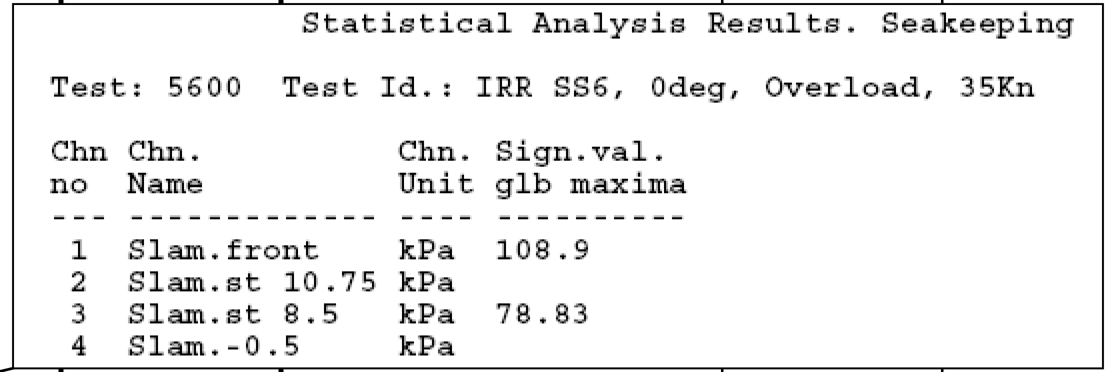
Slamming analysis

- Definition of slamming threshold value(s)
  - Typically 50 kPa, but depends heavily on context
- Counting (automatically) the number of slams above different threshold levels
- Detailed analysis of the time series of each slam reveals properties of the slam, the transducer and the model dynamic response



Summary of slamming tests at forward speed in towing tank (0°, 180°)

Test no	Slam.front (kPa)					Slam.st 10.75 (kPa)		Slam.st 8.5 (kPa)				Slam.-0.5 (kPa)	
	50 - 100	100 - 150	150 - 200	200 - 250	250 - 300	50 - 100	100 - 150	50 - 100	100 - 150	150 - 200	200 - 250	50 - 100	100 - 150
5000													
5020	1											2	
5040													
5060													
5080	4	1						3					
5120													
5140													
5160	1							1					
5180													
5200	1							2					
5220	10							2			1		
5300													
5320	1												
5340	13	2											
5360													
5380													
5400	1												
5420	1												
5440													
5460	4	2											
5500													
5520													
5540	1												
5560													
5580				1				7		1			
5600	2	1						4					
5680													
5700	2												
5720	7	1			1								
5740													

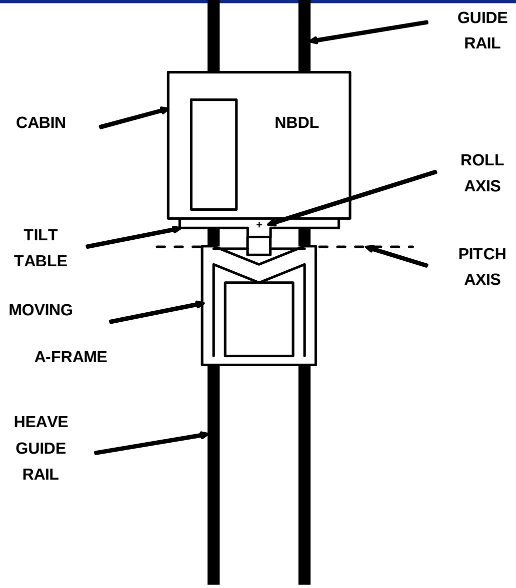


```
                    Statistical Analysis Results. Seakeeping

Test: 5600  Test Id.: IRR SS6, 0deg, Overload, 35Kn

Chn Chn.                 Chn. Sign.val.
no  Name                 Unit glb maxima
--- -------------        ---- ----------
 1  Slam.front           kPa  108.9
 2  Slam.st 10.75        kPa
 3  Slam.st 8.5          kPa  78.83
 4  Slam.-0.5            kPa
```

Sea sickness incidence



- Estimation of sea sickness incidence is based on:
    - Measurement of motions and accelerations of the model/ship
    - Measurement of motion sickness incidence MSI (percentage of people vomiting) to vertical accelerations of different frequency, amplitude and duration
- Empirical relations of motion sickness incidence (MSI) as function of frequency, RMS amplitude, and duration available in ISO standard ISO 2631 1-4

BIODYNAMIC EFFECTS: COMFORT AND FATIGUE