**Applied Geoscience Data Analysis using Matlab**

24 Oct 2013  
(Lecture 16)

Pwelch, Spectrogram  
Filtering & Convolution (part 1)

Trauth Ch 6.1-6.7

---

**Welch’s Average Method**

**Approach:**
- Divide a long time series up into segment (which might overlap).
- Calculate FFT of each
- Average the spectra

Segment 1: \( t = 1:100 \)
Segment 2: \( t = 51:150 \)
Segment 3: \( t = 101:200 \)

Window length = 100  
Overlap = 50% or 50 pts

---

**Time series plot of ADCP data**

![Time series plot of ADCP data](image)

**PSD plot of ADCP data**

```matlab
[Pxx,Freq]=periodogram(c,hanning(N),NFFT,48);  
figure; plot(Freq,Pxx);  
```

![PSD plot of ADCP data](image)
%[Pxx,F] = PWELCH(X,WINDOW,NOVERLAP,NFFT,Fs)
[Pxx,F] = pwelch(c,hanning(1024),512,[],48);
Fs=48; N = 7448; in this example we use 1024 point long windows that overlap by 50%
The empty bracket for NFFT, tells Matlab to use the nextpow2(length(WINDOW))
Note that window is 21.33 days (=1024/48) long

Fs=1000

%[S,F,T,P] = SPECTROGRAM(X,WINDOW,NOVERLAP,NFFT,Fs)
%[S,F,T,PSD]=spectrogram(s,250,200,256,Fs);

In this example Fs = 1000, so the windows are 1/4th of a second, with 80% overlaps
imagesc(T,F,10*log10(abs(PSD))); axis xy

Spectrogram
Time – Frequency – Power Representation of a signal

Let’s consider a chirp....
Visualizing Acoustic Signals

Spectrogram:
Red Colors Indicate Greater Power at a Given Frequency & Time

Frequency Filtering – Time Domain

High-Pass Filter = pass the high frequencies
Low-Pass Filter = pass the low frequencies

Filtering Data... Why?

A graphical description of filter (Frequency Domain)

a. Low-pass
b. High-pass
c. Band-pass
d. Band-reject
Filters are systems with inputs and outputs!

This is a cartoon in the time domain!

Important characteristics of a system:

- **Linearity** – Scaling and Additive Conditions

Scaling - Can multiple by a constant before or after passing thru the system

• **Linearity** – Scaling and Additive Conditions

Addition – Filtering the sum of the signals is the same as adding the two filtered outputs
**Invertible** - For an invertible system the original input can be reproduced from the system's output.

**LINEAR SYSTEM**
- x=1:100
- y=0.5*x

**NON-LINEAR SYSTEM**
- x=1:100
- y=x²

We can recover x: x=2*y

Since the sqrt(4) could indicate x=-2 or x=2.

**Causality** – the system response depends on present and past inputs, where as future inputs have no effect on the output.

**Stability** – both the input and output are finite.

**CONVOLUTION**

The convolution of a signal x with a filter b is written as [x ⊗ b]. It is defined as the summation of the product of the two time series after one is reversed and shifted.

\[ y(n) = \sum_{k=1}^{K} x(n - k + 1)b[k] \]

For example, if K = 3:

\[ y(n) = x(n - 2)b(3) + x(n - 1)b(2) + x(n)b(1) \]

\[ b = [b_1, b_2, b_3] \]
\[ x = [x_1, x_2, x_3, x_4...] \]

- \[ y(1) = 0b_3 + 0b_2 + x_1b_1 \]
- \[ y(2) = 0b_3 + x_1b_2 + x_2b_1 \]
- \[ y(3) = x_1b_3 + x_2b_2 + x_3b_1 \]
- \[ y(4) = x_1b_3 + x_2b_2 + x_3b_1 \]

etc.
$y(n) = x(n - 2)b(3) + x(n - 1)b(2) + x(n)b(1)$

$b = [6, 7, 8]; x = [1, 2, 3, 4, 5]$

- $y(1) = 0b_3 + 0b_2 + x_1b_1 = 1 \times 6 = 6$
- $y(2) = 0b_3 + x_1b_2 + x_2b_1 = 1 \times 7 + 2 \times 6 = 19$
- $y(3) = x_1b_3 + x_2b_2 + x_3b_1 = 1 \times 8 + 2 \times 7 + 3 \times 6 = 40$
- $y(4) = x_2b_3 + x_3b_2 + x_4b_1 = 2 \times 8 + 3 \times 7 + 4 \times 6 = 61$
- $y(5) = x_1b_3 + x_3b_2 + x_4b_1 = 3 \times 8 + 4 \times 7 + 5 \times 6 = 82$
- $y(6) = x_3b_3 + x_5b_2 + 0b_1 = 4 \times 8 + 5 \times 7 = 67$
- $y(7) = x_2b_3 + 0b_2 + 0b_1 = 5 \times 8 = 40$

**Graphic ways to do the convolution**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x</th>
<th>b)_1x</th>
<th>b)_2x</th>
<th>b)_3x</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>19</td>
<td>40</td>
<td>61</td>
</tr>
</tbody>
</table>

**Reversing the order has no effect**

<table>
<thead>
<tr>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td>18</td>
<td>21</td>
<td>24</td>
</tr>
<tr>
<td>24</td>
<td>28</td>
<td>32</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>40</td>
</tr>
</tbody>
</table>

| 6 | 19 | 40 | 61 | 82 | 67 | 40 |
CONVOLUTION -  
Alternative way

\[
\begin{align*}
\text{f} & = \{1 \ 1 \ 2 \ 2 \ 2\} \quad \text{(this is the time series)} \\
\text{g} & = \{4 \ 2 \ 1\} \quad \text{(this is the filter – the one to flip)}
\end{align*}
\]

\[
\begin{array}{c|c|c|c|c|c|c|c}
1 & 2 & 4 & 1 & 1 & 2 & 2 & 0 \\
\hline
1 & 2 & 4 & 1 & 1 & 2 & 2 & 1 \\
\hline
1 & 2 & 4 & 1 & 1 & 2 & 2 & 2 \\
\hline
1 & 2 & 4 & 1 & 1 & 2 & 2 & 3 \\
\hline
1 & 2 & 4 & 1 & 1 & 2 & 2 & 4 \\
\hline
1 & 2 & 4 & 1 & 1 & 2 & 2 & 5 \\
\hline
1 & 2 & 4 & 1 & 1 & 2 & 2 & 6 \\
\hline
1 & 2 & 4 & 1 & 1 & 2 & 2 & 7 \\
\end{array}
\]

\[
\begin{align*}
h & = f \ast g = \{4 \ 6 \ 7 \ 11 \ 13 \ 14 \ 6 \ 2\} \\
\text{Input signal 6 points long} \\
\text{Output Signal 8 points}
\end{align*}
\]

You try it... f convolved with g

\[
x = [3 \ 2 \ 1] \quad b = [5 \ 1]
\]

\[
y = \text{conv}(b, x) = ?
\]

You try it... x \ast b

\[
x = [3 \ 2 \ 1] \quad b = [5 \ 1]
\]

\[
\begin{align*}
3 & \ 2 \ 1 \ = \ 15 \\
1 & \ 5 \\
\end{align*}
\]

\[
\begin{align*}
3 & \ 2 \ 1 \ = \ 3 + 10 = 13 \\
1 & \ 5 \\
\end{align*}
\]

\[
\begin{align*}
3 & \ 2 \ 1 \ = \ 2 + 5 = 7 \\
1 & \ 5 \\
\end{align*}
\]

\[
\begin{align*}
3 & \ 2 \ 1 \ = \ 1 \\
1 & \ 5 \\
\end{align*}
\]

Moving Average Filter

\[
x = [1 \ 1 \ 5 \ 1 \ 5 \ 1 \ 5 \ 1 \ 5 \ 1 \ 5] \\
b = [1/3 \ 1/3 \ 1/3] \quad \text{“filter kernel”}
\]

\[
y = x \ast b
\]

x (input)

b (filter coefficients)

Y (output)
x=[3 2 1]
b=[5 1 4]

\[ y = \text{conv}(b,x) \]
\[ \begin{array}{cccc}
15 & 13 & 19 & 9 & 4 \\
\end{array} \]
\[ y = \text{conv}(x,b) \]
\[ \begin{array}{cccc}
15 & 13 & 19 & 9 & 4 \\
\end{array} \]

Need to adjust for the phase of the signal, since there are extra points on both ends.

For odd length filter \( b \)

\[ m = \text{length}(b) \]
\[ y = y(1+(m-1)/2:end-(m-1)/2) \]

**Options with matlab’s conv**

\[ x = [3 \ 2 \ 1] \]
\[ b = [5 \ 1 \ 4] \]
\[ \gg \text{conv}(x,b) \]
\[ \text{ans} = \begin{array}{cccc}
15 & 13 & 19 & 9 & 4 \\
\end{array} \]

\[ \gg \text{conv}(x,b,’valid’) \]
\[ \text{ans} = 19 \]

\[ \gg \text{conv}(x,b,’same’) \]
\[ \text{ans} = \begin{array}{cccc}
13 & 19 & 9 \\
\end{array} \]

\[ \text{C = CONV(A, B, SHAPE)} \text{ returns a subsection of the convolution with size specified by SHAPE:} \]

‘full’ - (default) returns the full convolution,

‘same’ - returns the central part of the convolution that is the same size as \( A \).

‘valid’ - returns only those parts of the convolution that are computed without the zero-padded edges.

\( \text{LENGTH(C)} \) is \( \text{MAX} (\text{LENGTH(A)} - \text{MAX}(0, \text{LENGTH(B)} - 1), 0) \).
High and Low Pass Filters via Convolution

a. Low-pass Filter

b. High-pass Filter

Invert the Signal via Convolution

This filter flips the signal and reduces the amplitude