L.EEC025 - FUNDAMENTALS OF SIGNAL PROCESSING

Academic year 2023-2024, week 13
P2P exercises

Topics: "Peer-to-peer learning/assessment" problems addressing the DFT of periodic signals, and a practical application of the auto-correlation using the definition and the DFT. Detailed information on the P2P procedure is available on Moodle ("Instructions regarding "Peer-to-peer learning/assessment" (P2P L/A)").

P2P Exercise 1

Consider the discrete-time signal $x[n] = 1 + \sin\left(\frac{2\pi}{N}n\right) \cdot \cos\left(\frac{4\pi}{N}n\right)$.

a) Find its period.

b) Obtain $X[k]$, the DFT of length N of $x[n]$.

Suggestion: expand $x[n]$ as a combination of complex exponentials and then establish a correspondence between that expansion and the terms of the definition of the inverse DFT transform (i.e. it is easier to find the result using the inverse DFT instead of the direct DFT)

Note: the solution should be: $X[k] = \begin{cases} N, & k = 0 \\ jN/4, & k = 1 \vee k = N-3 \\ -jN/4, & k = 3 \vee k = N-1 \\ 0, & other\ cases \end{cases}$

P2P assessment: **3pt /5** if demonstration is clear and complete, and results are correct

c) Verify numerically in Matlab the previous result using N=16.

P2P assessment: **2pt /5** if demonstration is clear and complete, and results are correct in demonstrating the consistency between the analytical solution (as per **b)**) and the numerical solution.

P2P Exercise 2

In this exercise, we use the auto-correlation in order the find the vibration frequency of the vocal folds (also commonly referred to as vocal cords) of a given speaker by analyzing a short segment of his/her voice sound corresponding to a sustained vowel, for example "aaa".

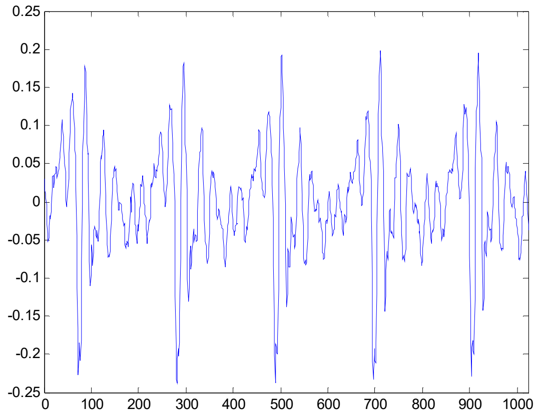
In Matlab read the `'vowel.wav'` WAV file that is available on Moodle and that includes a sustained vowel sound:

```
[s FS]=audioread('vowel.wav');
plot(s);
```

Using the graphical representation of the signal, select a 1024-samples segment exhibiting a clear periodicity. For example, after visualization, choose a `start` index (e.g. `start=40000`) to select that segment:

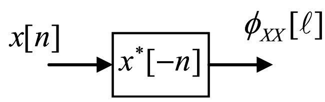
```
start=input('Please indicate start index: ');
L=1024;
x=[s(start:start+(L-1))];
plot([0:L-1],x)
pause
```

By representing graphically the segment `x`, a periodic pattern should be clear as illustrated in the following example.

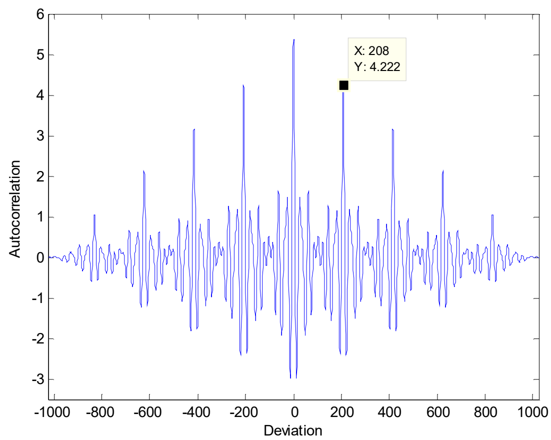


a) Consider the following Matlab code (some commands are deliberately incomplete or omitted) that implements the autocorrelation function illustrated on the right-hand side.

```
xr=... ;
ac=...(x, xr);
acx=[-1023:1023];
plot(acx,ac);
xlabel('Deviation'); % or 'Lag'
ylabel('Autocorrelation');
```



Complete the missing Matlab commands and obtain a figure similar to that illustrated next.



P2P assessment: **1pt /5** if demonstration is clear and complete, and results are correct

b) Check the numerical value of the maximum peak in the autocorrelation figure, i.e. `ac[0]`. What does this value correspond to ?

Note: that is the energy of the signal segment.

P2P assessment: **1pt /5** if explanation is clear and complete using the definition of the autocorrelation function.

© AJF

c) What is the meaning of the local peaks on each side of the maximum peak ?

d) By identifying the index of the local peak next to the right of the maximum peak, as illustrated above, find the vibration frequency of the vocal folds (i.e. the pitch of the recorded voice).

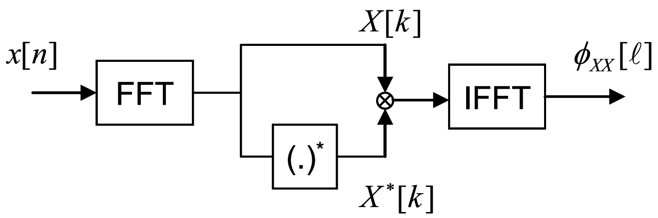
Note 1: the number that you read using the Matlab Figure Tool "Data Cursor" is a number of samples. You have to interpret this number and use it to find the vibration frequency.

Note 2: the vibration frequency should be a value between 50 and 500 Hz, *typically*, the pitch of an adult male speaker is around 110 Hz and the pitch of an adult female speaker is around 220 Hz, the frontier between male and female voices is around 160 Hz.

Note: the solution is: that number of samples reflects the signal periodicity and the resulting vibration frequency should be around 110 Hz

P2P assessment: **1 pt /5** if explanation is clear and complete, and result is correct.

e) The signal processing algorithm described next is an alternative way to compute the autocorrelation function. Complete the following Matlab code such that the resulting figure overlaps exactly with that obtained previously in **a).**




```
N=... ;
% next is zero-padding
z=[x.' zeros(1,N-length(x))];
Z=fft(z);
Y=... ;
y=ifft(Y);
y=real(fftshift(y));
hold on
plot([-L, acx], y, 'g');
hold off
```

f) Explain your choice of the value of N using the concepts or circular convolution and linear convolution.

Note: a possible solution is: N=2L

g) Explain why the computation of the autocorrelation function as in **e)** may be preferable to the computation as in **a)**.

Note: the solution involves using the following argument: for computational efficiency reasons due to computational gains permitted by the FFT. Considering only multiplication operations and admitting N=2L, the computational gain of the method in **e)** over the method in **a)** should be around 11 when real-valued multiplications are considered, or around 43 when complex-valued multiplications are considered.

P2P assessment: **2 pt /5** if explanation is clear and complete by expressing the computational gain of the solution in **e)** relative to the computation in **a)** when N=2L.

© AJF

OPTIONAL: if you want to use your own voice sound instead, use a microphone and the following Matlab code (or any other voice recorder) to record 3 seconds of your voice when you pronounce the sustained vowel “aaa”.

```
% this is for OLDER versions of MATLAB
FS=22050; NBITS=16; time=3;
s = wavrecord(time*FS,FS);
s = s / (1.2*max(abs(s))); % normalize
sound(s,FS); % or wavplay(s,FS)
wavwrite(s,FS,NBITS,'vowel.wav'); % or audiowrite()

% this is for NEWER versions of MATLAB
FS=22050; NBITS=16; time=3;
r = audiorecorder(FS, NBITS, 1);
fprintf('Start speaking...\n');
record(r);      % speak to the microphone...
pause(time);    % record during time seconds
stop(r);        % stop recording
s = getaudiodata(r, 'single'); % data is float array, range [-1, 1]
s = s / (1.2*max(abs(s))); % normalize
p = audioplayer(s, FS);
play(p);
audiowrite('myvowel.wav', s, FS, 'BitsPerSample', NBITS)
```

By representing graphically the signal, confirm first that it is not affected by ‘clipping’ (i.e., saturation of the signal due to its amplitude exceeding the maximum allowed dynamic range). If it does, repeat the recording such that it does not.

© AJF