PROJECT

Media Authentication with Electric Network Frequency Analysis

Completion Date: November 30, 2011

1. Objective

The objective of this project is to investigate the use of various signal processing techniques in extracting *Electric Network Frequency* (ENF) information from audio signals as a means of media authentication.

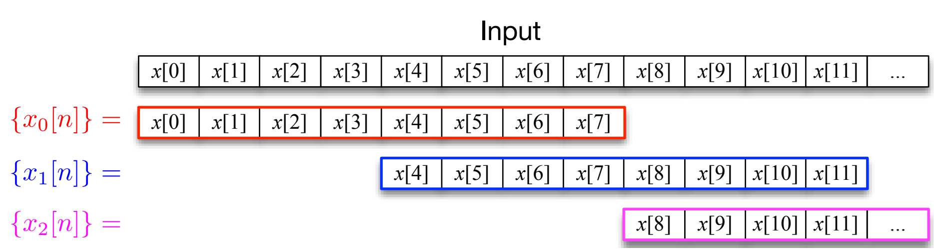
2. The Algorithm

In this project you will use *Short-Time Fourier Transform* (STFT) to extract time varying ENF information at the line frequency and its harmonics. The ENF information extracted from the audio signal will be compared to a reference ENF signal. Comparison of the two ENF signals can provide valuable clues about the similarity between the two ENF signals and consequently about the authenticity of the audio signal being investigated.

Note that the purpose of this document is not provide a detailed description of the signal processing algorithms that you will use to extract ENF information or to compare two ENF signals. Full description of the algorithms are provided the references listed at the end of this document (in particular see the first reference, Garg (2011). Please also refer to the slides used in the presentation that introduced the project.

2.1 Notation and Formulation

STFT: Let $\{x[n]\}$ be the sampled data sequence to be analyzed. The input sequence is segmented into blocks of N-samples which may overlap. Let $\{x_m[n]\}$ be the mth block extracted from $\{x[n]\}$. For example, if $\{x_0[n]\}$ and $\{x_1[n]\}$ are respectively the zero*th* and first subblocks extracted from $\{x[n]\}$ with $N = 8$ and 50% overlap, then:



The STFT of the the segmented input sequence is defined as:

$$X[k, m] = \sum_{n=0}^{N-1} x_m[n]\, W_N^{nk} \tag{1}$$

where $W_N = e^{-2\pi/N}$, k is the frequency index and m is the block index. Observe that the frequency index k can be easily transformed into frequency values using the sampling frequency F_s and the block size parameter N used to generate the input sequence $\{x[n]\}$. Similarly, the block index m can be transformed into time index using the sampling frequency F_s, the block size parameter N and the overlap between successive input blocks. The *spectogram* of the signal being analyzed is the time-varying energy spectrum extracted from the STFT of the signal:

$$\mathcal{E}[k,m] = \big|X[k,m]\big|^2 \tag{2}$$

Maximum energy: In this method, the frequency corresponding to the maximum energy in each signal block within the energy spectrum is identified, Garg (2011). Let

$$\mathcal{E}_{\max}[k_{\max}[m], m] = \max_k \mathcal{E}[k,m]. \tag{3}$$

Thus, $k_{\max}[m]$ represents the frequency bin index for the mth block where the signal energy is maximized. The corresponding maximum energy frequency for the mth block $F_{\max}[m]$ can be determined as before using the sampling frequency F_s and the block size parameter N.

Weighted energy: The weighted energy method records the weighted average frequency around the ENF value in each block, i.e., time bin, of the spectogram, Garg (2011). Let $\{F[k]\}$ be the frequency array measured in Hz at kth frequency-bin. The *weighted frequency* array $\{F_{wf}[m]\}$ is defined as:

$$F_{wf}[m] = \frac{\displaystyle\sum_{k=K_1}^{K_2} F[k]\mathcal{E}[k,m]}{\displaystyle\sum_{k=K_1}^{K_2} \mathcal{E}[k,m]} \tag{4}$$

where $K_1 = k_{ENF} - 1$, $K_2 = k_{ENF} + 1$ and k_{ENF} is the frequency bin index corresponding to the ENF value being analyzed.

Normalized cross correlation: The normalized cross-correlation between the arrays $\{a[m]\}$ and $\{b[m]\}$ at lag l is defined as:

$$\rho[l] = \frac{\displaystyle\sum_{m=0}^{M-1} \big(a[m] - \mu_a\big)\big(b[m-l] - \mu_b\big)}{\sqrt{\displaystyle\sum_{m=0}^{M-1} \big(a[m] - \mu_a\big)^2 \sum_{m=0}^{M-1} \big(b[m-l] - \mu_b\big)^2}} \tag{5}$$

where μ_a and μ_b are the mean values of the $\{a[m]\}$ and $\{b[m]\}$ arrays, respectively.

3. The Assignment: What is to be done?

For this project you need to write a Matlab function and a Matlab script.

Matlab function `enf.m`

The Matlab function **`enf.m`** will extract the ENF signal from a sound file. The call to **`enf.m`** should be in the form:

```
[y1, y2] = enf(x, Fs, BlockSize, ZeroPad, Overlap, Window, Frequency);
```

Input Variables and Parameters:

`x`:
: L_x-sample long one-dimensional input sequence $x[n]$ containing samples of the sound file to be analyzed. In particular, `x` is a real-valued column vector of dimension $L_x \times 1$.

`Fs`:
: Sampling rate of the input sequence `x`.

`BlockSize`:
: Parameter that determines the length of each subblock extracted from the input sequence `x`. For example, if $\{x_1[n]\}$ is the first subblock extracted from `x`, then $\{x_1[n]\} = \{\, \texttt{x}[1], \ \ldots\ , \texttt{x}[\texttt{BlockSize}]\,\}$—note that index starts at 1 to conform with Matlab programming standards. `BlockSize` also determines the size of the FFT algorithms to be used to transform blocks of data from time-domain to frequency-domain.

`ZeroPad`:
: Amount of zero padding to be used for Fourier analysis. For `ZeroPad` = N_z each windowed data segment extracted from the input sequence will be appended with N_z zero-valued sampled before transforming into the frequency domain.

`Overlap`:
: Input parameter that specifies overlap between successive subblocks extracted from the input sequence `x`. Overlap is specified as fraction of `BlockSize` with $0 \leq \texttt{Overlap} < 1$ such that $\texttt{Overlap} = 0$ corresponds to *no overlap* and $\texttt{Overlap} = 1$ corresponds to *full overlap* conditions.

`Window`:
: Window function $\{w[n]\}$ to be applied to each input block $\{x_m[n]\}$ before transformed to frequency domain by DFT. The data array $\{w[n]\}$ is a column vector of dimension `BlockSize` $\times 1$. For example, `Window` = **`hanning`**`(BlockSize)`.

`Frequency`:
: The ENF value to be analyzed. For example if you want to analyze ENF at line frequency then `Frequency` = 60, or if you want to analyze the first harmonic of the line frequency then `Frequency` = 120.

Output Variables:

`y1`:
: $M \times 1$ dimensional **maximum energy** array as described in Equation (2).

`y2`:
: $M \times 1$ dimensional **weighted energy** array as described in Equation (4).

Number of blocks M and therefore lengths of the `y1` and `y2` output data arrays is a function of the length of the input $x[n]$, and the `BlockSize`, `ZeroPad` and `BlockSize` parameters.

Matlab script **`ENFproject.m`**

The Matlab script **`ENFproject.m`** is to demonstrate how the **`enf.m`** function developed earlier is to be used to operate on the maximum energy and the weighted energy arrays extracted from the reference sound file and the test sound file using the normalized cross-correlation analysis given in Equation (5).

The **`ENFproject.m`** scripts should compare the the maximum energy and the weighted energy arrays extracted from the reference sound file and the test sound file using with and without pre-processing the sound files.

Comparison of reference and test sound files **without pre-processing**

Use the parameters:

$\gg$ `Fs` $= 44100;$
$\gg$ `BlockSize` $= Fs * 16;$
$\gg$ `ZeroPad` $= 0;$
$\gg$ `Overlap` $= 0.5;$
$\gg$ `Window` = **`hanning`**`( BlockSize, 'periodic' );`
$\gg$ `Frequency` $= 120;$

with sound files:

- Reference file: `ground truth.wav`;
- Test file: `recording.wav`.

Alternatively, you can use `ground truth 2.wav` and `recording 2.wav`.

The Matlab script file should use the above listed parameters to extract the maximum energy and weighted energy arrays from the specified reference and test sound files using the Matlab function **`enf.m`**. Perform normalized cross-correlation analysis first on the maximum energy arrays extracted from the reference and the test sound files to determine (if possible) the time-delay between the reference and test signals. Repeat the same analysis using the weighted energy arrays to estimate the time-delay between the reference and test signals.

Comparison of reference and test sound files **with pre-processing**

In this scenario you should first pre-process signals by decimating the sound files by a factor of 100. To achieve this objective design an appropriate lowpass filter, filter the audio signal and then downsample by 100 before processing it further as before. In particular, use the following analysis parameters on the decimated audio signals:

```
>> Fs = 441;          % sampling frequency after decimation
>> BlockSize = Fs * 16;
>> ZeroPad = 2^14 - Fs * 16;
>> Overlap = 0.5;
>> Window = hanning(???,'periodic'); ...
                    % calculate the new block size after zero padding
>> Frequency = 120;
```

Repeat the same analysis as in the case of "without pre-processing" using the maximum energy and the weighted energy arrays to estimate the time-delay between the reference and test signals.

4. Discussion

You may find the following discussion and information helpful in developing your code.

- **Reading audio files:** Use the Matlab function **`audioread`** to read sound files into your code or workspace.

- **Window function (`Window`):** Two windows with perfect reconstruction properties are the *Bartlett* and the *Hanning* windows defined respectively as:

$$w_{Bart}[n] = \begin{cases} 2n/N, & 0 \le n \le N/2; \\ 2 - 2n/N, & N/2 < n \le N-1; \\ 0, & \text{otherwise}. \end{cases} \tag{6}$$

$$w_{Hann}[n] = \begin{cases} 0.5 - 0.5\cos(2\pi n/N), & 0 \le n \le N-1; \\ 0, & \text{otherwise}. \end{cases} \tag{7}$$

  where N is the length of the window. Matlab functions **`bartlett`** and **`hanning`** allow you to generate data arrays representing these two window function with ease.

  It is recommended that you use the *Hanning* window. In particular, generate the window array using the call:

  ```
  hanning( BlockSize, 'periodic' )
  ```

  The above call generates samples of the *Hanning* window that conforms with Equation (7) and that is suitable for periodic application of the window function as in data segmentation (generate the window data array without the specifying the `periodic` parameter and compare the resulting window data arrays).

- **Reshaping an array for processing:** Your code will convert each segmented and windowed subblock of length `BlockSize` into corresponding DFT coefficient arrays using `BlockSize`-element FFT. You can do this in a loop. But for computational efficiency, try to utilize the array processing capabilities of Matlab. For example, if the signal has $N_{nb}K$ elements, consider creating a $N_{nb} \times K$ matrix by reshaping the Matlab array that contains the signal that will be converted into the frequency domain. The built-in Matlab function **`reshape`** can easily accomplish this task. If `x_array` is the $N_{nb} \times K$ array generated by reshaping, then the single command **`fft`**(`x_array`, N_{nb}) will compute the N_{nb}-point FFT of each column, i.e., of each block.

References

1. R. Garg, A.L. Varna, and M. Wu, "Seeing" ENF: Natural Time Stamp for Digital Video via Optical Sensing and Signal Processing, *Proceedings of the 19th ACM International Conference on Multimedia*, New York, NY, USA, pp. 23–32, 2011. http://doi.org/10.1145/2072298.2072303.

2. R. Garg, A.L. Varna, A. Hajj-Ahmad and M. Wu, "Seeing" ENF: Power-Signature-Based Timestamp for Digital Multimedia via Optical Sensing and Signal Processing, *IEEE Transactions on Information Forensics and Security,* Vol. 8(9), pp. 1417–1432, 2013. http://doi.org/10.1109/TIFS.2013.2272217

3. A. Hajj-Ahmad, R. Garg and M. Wu, ENF based location classification of sensor recordings, *IEEE International Workshop on Information Forensics and Security (WIFS),* pp. 138–143, 2013. http://doi.org/10.1109/WIFS.2013.6707808.

4. A.V. Oppenheim and R.W. Schafer, *Discrete-Time Signal Processing*, 3rd Edition, Pearson Higher Education Inc., 2010.

LAB PROJECT: Submission Information and Instructions

Deadlines

Code: November 30, 2017

Completion of the Matlab function **enf.m** and Matlab script **ENFproject.m** that implements the *ENF Analysis* algorithm as described in this document.

Report: November 30, 2017

Completion and submission of a report no more than 10 pages long. This report should address the following issues:

- a brief description of the problem;
- explanation of how you perform the segmentation of the input array into subblocks as a function of `BlockSize` and `Overlap` parameters;
- explanation of how you convert the value of the `Frequency` variable into the start and stop frequency values which are in turn mapped onto corresponding DFT indices;
- comparison and analysis of time-delay estimation performance using *maximum energy* and *weighted frequency* arrays;
- comparison and analysis of processing audio signals with and without pre-processing; discussion of how pre-processing affects the outcome of ENF analysis;
- explanation of how the lag value(s) estimated using the normalized cross-correlation analysis are converted into time-delay values.

Format your report as a PDF file (PDF format only please!).

Grading/Evaluation

- Lab Project is worth 60% of your laboratory grade (and therefore 24% of your course grade). In particular, the individual parts of Lab Project will be graded as follows:

Project-Code	. . . :	20 points
Project-Report	. . . :	10 points

- Efficiency of your Matlab code will be evaluated.
- All your Matlab code should be well commented.
- By the assignment completion deadlines, submit your files and report using the command:

```
submit ele792 project myfile
```

where **`myfile`** is name of the file (or archive) that you want to submit.