Accelerated SWT based de-noising technique for EEG to correct the Ocular Artifact

Mahesh Khadtare, Pragati Dharmale
Plan of Session

I. Introduction
II. Overview
III. Stationary Wavelet Transform (SWT)
IV. Algorithmic & implementation of EEG denoise
V. Results
VI. Discussions
Introduction

• Every time we think, move, feel or remember something, our neurons are at work. That work is carried out by small electric signals that zip from neuron to neuron as fast as 250 mph [source: Walker]

• Using data recorded from the brain, the BCI processes it, interprets the intention of the user, and acts on it, Figure shows Awake state and Asleep state [source: Wikipedia].

• Eye movements and blinks cause a severe problem for EEG measurements [Source: EEG Lead placements-EEGLab]
Introduction

Neurologists

• Brain Computer

Neurologist
Introduction

The references used for research are Evenly spread.

Most are recent years.

3/19/2015
Overview

• Almost all existing approaches to ocular artifact (OA) detection and removal use one or more electro oculogram (EOG) signals either directly or indirectly as a reference.
• Some feature of the output of the algorithm (e.g., signal-to-noise ratio, or SNR) is compared to the original artifact-free EEG.
• For real EEG, the artifact-free (“true”) EEG is not known, so the performance of the algorithm on real data is usually reported, often based on visual inspection of the resulting waveforms.
• So the challenge is removal of EOG artifact without or less EEG data distortion
Overview

- **What is Wavelet**
  - A small wave
- **Wavelet Transforms**
  - Convert a signal into a series of wavelets
  - Provide a way for analyzing waveforms, bounded in both frequency and duration
  - Allow signals to be stored more efficiently than by Fourier transform
  - Be able to better approximate real-world signals
  - Well-suited for approximating data with sharp discontinuities
### Overview

#### Fourier Vs Wavelet Transform

<table>
<thead>
<tr>
<th>Fourier Transform</th>
<th>Wavelet Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Information only, Time/space information is lost</td>
<td>Joint Time and Frequency Information</td>
</tr>
<tr>
<td>Single Basis Function</td>
<td>Many Basis Functions</td>
</tr>
<tr>
<td>Computational Cost High</td>
<td>Low computational costs</td>
</tr>
</tbody>
</table>
| Analysis Structures:  
  - FS( periodic functions only)  
  - FT and DFT | Numerous Analysis structures:  
  - CWT,  
  - DWT(2 Band and M Band),  
  - WP  
  - SWT,  
  - Frame structure |
Standard DWT

- Classical DWT is not shift invariant: This means that DWT of a translated version of a signal \( x \) is not the same as the DWT of the original signal.
- Shift-invariance is important in many applications such as:
  - Change Detection
  - Denoising
  - Pattern Recognition
- In DWT, the signal is convolved and decimated (the even indices are kept.)
- The decimation can be carried out by choosing the odd indices.
Standard DWT (2 Stage)

Decimated Wavelet Transform-Analysis Stage
SWT

- Apply high and low pass filters to the data at each level
- Do not decimate
- Modify the filters at each level, by padding them with zeroes
- Computationally more complex
SWT (2 Stage)

Signal -> High Pass -> High Freq Details

Signal -> Low Pass -> Low Freq Approx

High Pass -> High pass

Low Pass -> Low pass
Different Implementations

• A Trous Algorithm: Upsample the filter coefficients by inserting zeros

• Beylkin’s algorithm: Shift invariance, shifts by one will yield the same result by any odd shift. Similarly, shift by zero → All even shifts.
  – Shift by 1 and 0 and compute the DWT, repeat the same procedure at each stage
  – Not a unique inverse: Invert each transform and average the results

• Undecimated Algorithm: Apply the lowpass and highpass filters without any decimation.
Traditional parallel algorithm

Dependencies
• Scheme depends upon the PE’s Configuration
• PE’s Interconnectedness
• Message passing Protocols
• Data transmission bandwidth

Traditional distributed processing algorithm does not include any parallel processing algorithm for Signal processing
Proposed parallel algorithm
Proposed parallel algorithm

![Diagram of parallel algorithm involving EEG and G-PE nodes]
Mathematical Model
Wavelet Coefficients

• Detail Coefficient:

\[ y_{high}[k] = \sum_{n} x[n] \cdot g[2k - n] \]

• Approximate Coefficient:

\[ y_{low}[k] = \sum_{n} x[n] \cdot h[2k - n] \]
Algorithm Flow

Signal domain \rightarrow basis functions \rightarrow Transformed domain, Coeffs

Transformation \rightarrow Analysis

Information extraction

Recon Signal \rightarrow Reconstruction

Signals
Wavelet Decomposition

- The maximum frequency of EEG data sample is 125 Hz.
- The original data can be decomposed in up to 10 detail levels (D₁ – D₁₀) and a last approximation (Apr).
- The frequency limits of each scale are approx. calculated dividing by 2 the sampling rate. In the case of 250 Hz, these are (with a 5 scales decomposition):
  - D₁: 64 – 128 Hz;
  - D₂: 32 – 64 Hz
  - D₃: 16 – 32 Hz;
  - D₄: 8 – 16 Hz;
  - D₅: 4 – 8 Hz;
  - A₅: 0 – 4 Hz.

Note that D₂, D₃, D₄, D₅, A₅ approximately correspond to the EEG frequency bands: Gamma, Beta, Alpha, Theta, Delta respectively.
General denoising procedure

- The involves three steps. The basic version of the procedure follows the steps described below:
  - **Decompose**: Select a wavelet, select a level N. Compute the wavelet decomposition of the signal at level N.
  - **Zeroing or Thresholding detail coefficients**: For each level from 1 to N, either make detail coefficient zero or select a threshold value and apply to the detail coefficients.
  - **Reconstruct**: Compute wavelet reconstruction using the original approximation coefficients of level N and the modified detail coefficients of levels from 1 to N.
Threshold method

• In wavelet denoising process, the threshold method is one of the main methods.
• Threshold function has important relationship with the continuity and accuracy of the reconstructed signals, and has a significant impact on wavelet denoising.
• At present, there are two choices of methods of which are hard threshold and soft threshold.
• But the hard threshold method makes the wavelet coefficients discontinuous in the threshold value position and leads to the oscillations of the reconstructed signals; while with the soft threshold method wavelet coefficient has improved continuity.
Threshold method

- The key point of wavelet threshold denoising is the selection of threshold and how to choose the threshold function when dealing with wavelet coefficient after decomposition.
- There are four commonly used methods to select threshold of which are Donoho-Johnstone methods:
  1. Fixed-form (default)
  2. Heursure
  3. Rigsure
  4. Minimax
EEG Data

• EEG Database –
• Physionet EEG database
  – 325 Recording with 7 Channel with different task
    • Baseline, Rotation, Letter composing, Counting, Multiplication
    • Duration: 8 min (Sampling Frequency (fs): 256Hz)
    • Typically: 840MB data size per recording

• Realtime EEG database
  – 60min Recording with 4 Channels
  – Eye blinking, all above activity

Source: http://sccn.ucsd.edu/~arno/fam2data/publicly_available_EEG_data.html
Distribution Of Execution Time

The rate of Execution Time* 60% 61% 30%

EEG Signal Processing

Sensor

Signal Data

SWT Forward Transform

Thresholding

SWT Inverse

Reconstructed Signal

*1 Thread, Conventional Corner Turn

Demo – Sample unit test
Algorithm Implementation

Read All Channel EEG Data for Signal Processing

Rearrange all channel Data (Block formation and groups)

Compute Thresholds

Rearrange Image Data (Block formation and Flipping)

DWT or SWT forward

DWT or SWT Inverse

Denoise Thresholds

Compute Thresholds
CPU code

• Lifting Step Wavelet

lifting_step_ti(double[,,] x,
    double[] h,
    int dir,
    int dist)

• Wavelet levels and direction

perform_wavelet_transf(double[,,] x,
    int Jmin,
    int dir)

CUDA Kernel

• Row Kernel –

__global__, void rowsKernel(
    double *d_Dst,
    double *d_Src,
    double *d_Kernel,
    int RowSize,
    int ColSize,
    int kernelRadius)

• Column Kernel

__global__ void columnsKernel(
    double *d_Dst,
    double *d_Src,
    double *d_Kernel,
    int RowSize,
    int ColSize,
    int kernelRadius)
Test Results

Input EEG

Denoise O/p MATLAB/C# Ch1/ CUDA

Amp(microV)

Time (s)

Amp(microV)

Time (s)
# Results – EEG Denoise

## EEG Denoising Timing

<table>
<thead>
<tr>
<th>CPU / GPU</th>
<th>Core Details</th>
<th>Timing (sec)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Core i3 – MATLAB</td>
<td>2</td>
<td>45.337</td>
<td>1x</td>
</tr>
<tr>
<td>Intel Core i3 – C#</td>
<td>2</td>
<td>12.121</td>
<td>3x</td>
</tr>
<tr>
<td>GeForce GT 525M</td>
<td>96</td>
<td>1.031</td>
<td>44x</td>
</tr>
</tbody>
</table>

Data size: 512 x 8

3/19/2015
Signal to Noise Ratio (SNR)

From SNR, we decided accuracy of the wavelet.

- It is defined as the ratio of signal power to the noise power corrupting the signal. A ratio higher than 1:1 indicates more signal than noise.

\[
SNR = \frac{P_{\text{signal}}}{P_{\text{noise}}}
\]

- SNR in dB is calculated as:

\[
SNR = 20 \log\left(\frac{A_{\text{signal}}}{A_{\text{noise}}}\right)
\]
Comparison Five Wavelet Types

<table>
<thead>
<tr>
<th>TASK</th>
<th>ALL COEFF. Max SNR</th>
<th>WAVELET TYPE</th>
<th>CD1,CD2 Max SNR</th>
<th>WAVELET TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BASELINE</td>
<td>119.6598</td>
<td>COIF1</td>
<td>13.2003</td>
<td>DMEY</td>
</tr>
<tr>
<td>COUNTING</td>
<td>9.4811</td>
<td>DMEY</td>
<td>18.35</td>
<td>SYM6</td>
</tr>
<tr>
<td>LETTER COMPOSING</td>
<td>7.17</td>
<td>SYM6</td>
<td>14.29</td>
<td>SYM6</td>
</tr>
<tr>
<td>MULTIPLICATION</td>
<td>9.37</td>
<td>DMEY</td>
<td>12.76</td>
<td>DMEY</td>
</tr>
<tr>
<td>ROTATION</td>
<td>7.8818</td>
<td>DB4</td>
<td>19.41</td>
<td>DMEY</td>
</tr>
</tbody>
</table>
Thank You
References

