Annales UMCS Informatica AI 6 (2007) 103-115

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Wavelet analysis of speech signal

Ireneusz Codello*, Wiesława Kuniszyk-Jóźkowiak

Institute of Computer Science, Maria Curie-Skłodowska University,
pl. M.Curie-Skłodowskiej 1, 20-031 Lublin, Poland

Abstract

This paper concerns the issue of wavelet analysis of signals by continuous and discrete wavelet transforms (CWT – Continous Wavelet Transform, DWT – Discrete Wavelet Transform). The main goal of our work was to develop a program which, through the CWT and the DWT analyses, would obtain graph of time-scale changes and would transform it into the spectrum, that is a graph of frequency changes. In this program we also obtain spectra of Fourier Transform and Linear Prediction. Owing to this, we can compare the Wavelet Transform results to those from the Fourier Transform and Linear Prediction.

1. Introduction

A signal can be analyzed applying many methods. One of the most popular and basic method is Fourier Transform (FT). Based on it information on frequencies which make up the examined signal is obtained. Unfortunately FT expects a stationary signal as a input data. If we want to process a speech signal (which is very unstationary) we have to use the STFT algorithm (Short-Time Fourier Transform). In this method we split a signal into small pieces (windows), assuming that in each window signal is stationary (because human speech, in small time intervals, is almost invariable – this assumption is near the truth) and we compute FT for each window separately. Unfortunately FT, because it assumes a stationarity of a signal, does not show us when a given frequency appears in the window. Additionally exactness (resolution) of frequencies in spectrum increases with window length. So exactness of frequency and exactness of occurrence of frequencies in time (in window) are inversely proportional. Therefore deciding which window length to choose, we have to make a decision what is more important to us – good time resolution (narrow window) or good frequency resolution (wide window).

*Corresponding author: *e-mail address*: irek.codello@gmail.com

The Wavelet Transform completes this inconvenience in some way, but to show this, we have to get familiar with few basic notions.

2. Wavelets

Wavelet $\psi(t)$ is a function, which meets conditions:

– its mean value equals 0, i.e.

$$\sum_t \psi(t) = 0\,, \tag{1}$$

– has non-zero values only in finite interval

$$\langle m,n \rangle\,. \tag{2}$$

Each wavelet has specified such an interval <*m,n*>. Futhermore wavelet can begin and end with 0-values, so F(m) does not have to be the first non-zero value and F(n) does not have to be the last one.

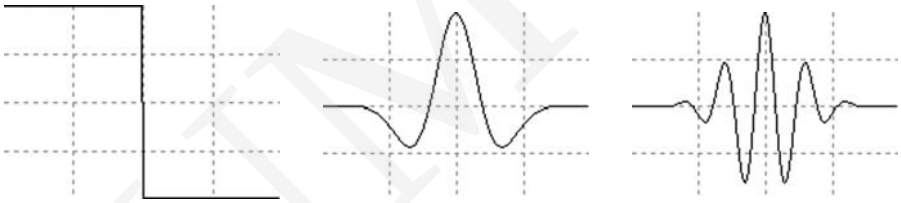


Fig. 1. Exemplary wavelets. From the left: Haar, Mexican Hat, Molet

Additionally from every basic wavelet $\psi(t)$ we create whole family of wavelets, by changing *a* and *b* parameters:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), \qquad \begin{array}{l} a-\text{scale} \\ b-\text{ offset} \end{array} \tag{3}$$

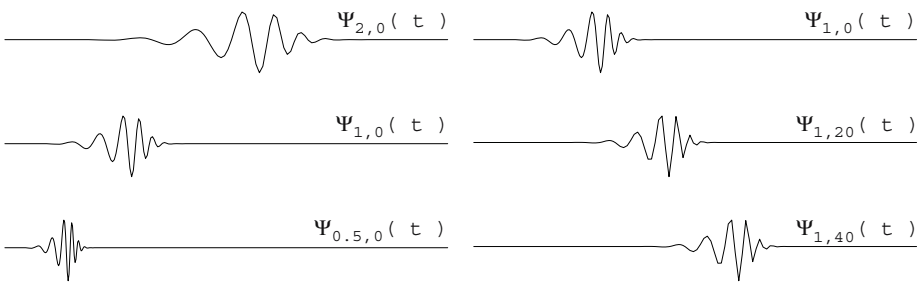


Fig. 2. Exemplary functions from the wavelet family $\psi_{a,b}(t)$

The coefficient *a* corresponds to horizontal scaling of the wavelet (stretching and expanding), and the coefficient *b* corresponds to shifting the wavelet.

3. CWT - Continuous Wavelet Transform

Continuous Wavelet Transform can be presented by the formula:

$$CWT_{a,b} = \sum_{t} f(t) \cdot \psi_{a,b}(t), \quad f(t) - input\ signal. \tag{4}$$

The procedure rule of CWT exemplifies Figure 3. Consecutive wavelets $\psi_{a,b}(t)$, hailed from basic wavelet $\psi(t)$ (they belong to the same family), are multiplied by input signal (sample by sample). For every pair *a,b* we obtain a $CWT_{a,b}$ value. In Figure 3, horizontal scale step (parameter *a*) decreases exponentially (2^x), however offset (parameter *b*) increases linearly (step = 20). Of course *a* and *b* values can be arbitrary and therefore this transform is called continuous.

Additionally we assume in this paper, that $CWT_{a,b}$ value which is equal to 3 and -3 represents the same level of similarity, hence all figures represent the $|CWT_{a,b}|$ values.

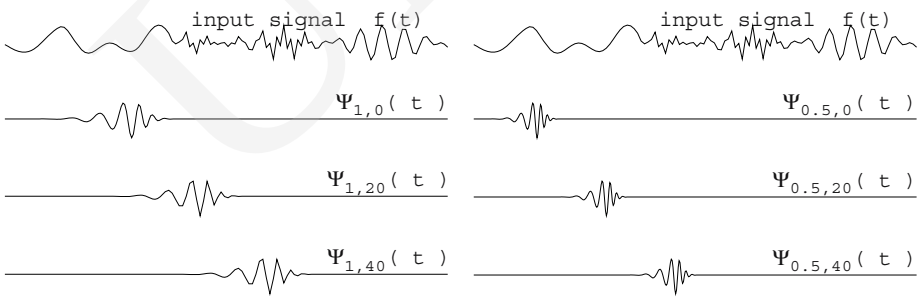


Fig. 3. Computational scheme of CWT

It is important to take into account the boundaries situations. Because input signal is discrete – a wavelet has to consist of at least two samples. If it would contain from one sample, it would have to be equal to 0 – according to (1) condition, and then it would not meet condition number (2). Wavelet can not be infinitely stretched as well – its length should not be larger then input signal length. Secondarily, input signal is finite, so after shifting a wavelet to right, wavelet non-zero values will eventually stick out of the end of the input signal. We can handle this situation in several ways:

– we can complete the signal witch zeros – …,s_{n-3}, s_{n-2}, s_{n-1},**0,0,0**,…

– we can complete the signal witch mirror reflections of its end – …,s_{n-3}, s_{n-2}, s_{n-1}, $\mathbf{s_{n-1}}$, $\mathbf{s_{n-2}}$, $\mathbf{s_{n-3}}$,…
– we can complete the signal periodically, that is with the samples from the beginning of the signal – …,s_{n-3}, s_{n-2}, s_{n-1}, $\mathbf{s_0}$, $\mathbf{s_1}$, $\mathbf{s_2}$,…

Of course the parameter *b* can be at best equal to signal length minus one.

The representation of $CWT_{a,b}$ values in a figure can be, for instance, shown in the following form:

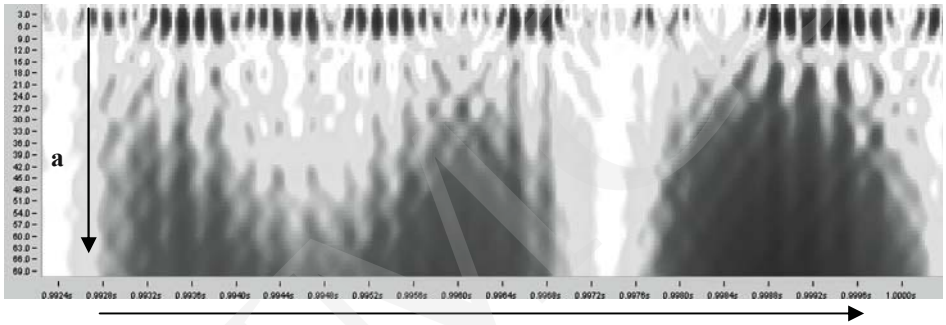


Fig. 4. Fragment of an exemplary CWT. The parameter *b*, which is proportional to the time, was transformed into seconds owing to a sampling rate of an output signal

4. Spectrum

The result of the CWT analysis is a time-scale characteristics, i.e. it represents scale fluctuations in time. Each horizontal line (corresponding to some scale *a*) in Figure 4, represents similarity changes between the input signal and $\psi_{a,b}(t)$ wavelet. The darker a fragment is, the similarity is greater, i.e. absolute value of formula (4) is greater. To obtain a spectrogram we have to transform scale values into frequency values. Unfortunately it is not a precise transformation, because a wavelet instead of one frequency, represents a group of frequencies. Therefore, this group approximated to only one, central frequency $F_{C,}$ with is a maximum in the FFT spectrum of the wavelet.

We can change each scale *a* into corresponding pseudo-frequency:

$$F = \frac{F_C \cdot F_S}{d}, \tag{5}$$

where
F_C – central frequency,
F_S – input signal sampling rate,
d – wavelet width of scale *a,* that is an amount of samples in the given scale *a* that belongs to the <*m,n*> bracket from condition (2).

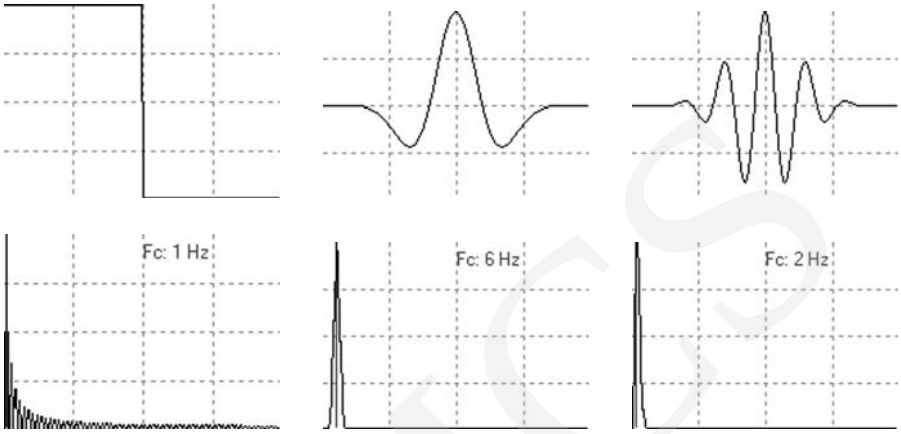


Fig. 5. Wavelets: Haar, Mexican Hat, Morlet and the corresponding FFT spectra

After such a change we get the same figure with changed vertical axis:

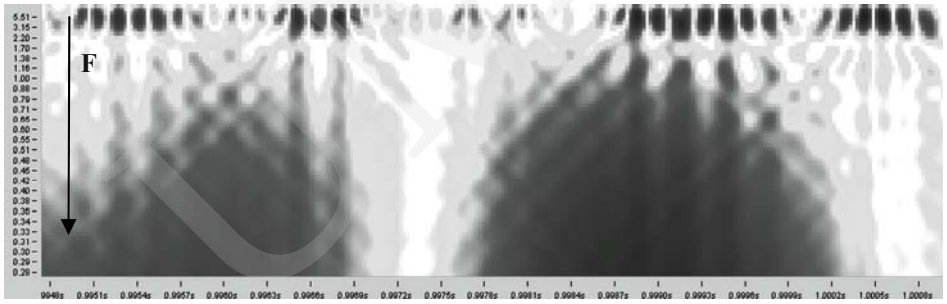


Fig. 6. Another piece of CWT. Consecutive scale numbers was changed into pseudo-frequencies

On vertical axis the scale *a* was linear, however after change, the pseudo-frequencies is not. We need to transform the figure to get linear pseudo-frequency scale.

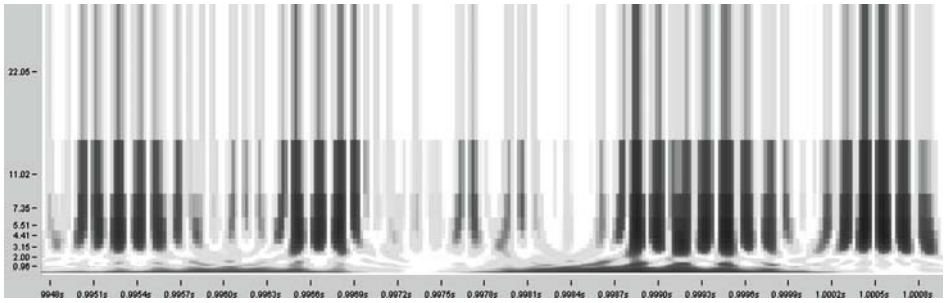


Fig. 7. Spectrogram created from transforming Figure 6 to get linear pseudo-frequency scale

5. Discrete wavelets

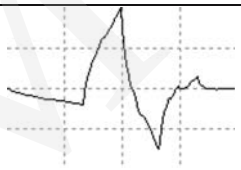
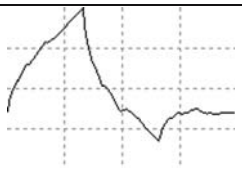
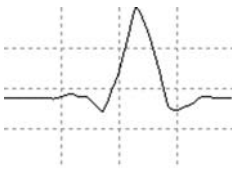
It exists a group of wavelets, whose figures are created from finite set of values, which are called wavelets coefficients *h(n)*. From those values we obtain scaling coefficients *g(n)*.

$$g(n) = (-1)^n \cdot h(N-1-n), \quad n = 0 \ldots N-1 \text{ , N – coefficients count}$$

Thereafter using recurrence formulae:

$$\psi(t) = \sqrt{2}\sum_{n=0}^{N-1} h(n)\varphi(2t-n), \qquad \varphi(t) = \sqrt{2}\sum_{n=0}^{N-1} g(n)\varphi(2t-n) \tag{7}$$

we can obtain wavelet function $\psi(t)$ and a scaling function $\varphi(t)$ with arbitrary resolution.

Wavelet name	*h(n)*	$\psi(t)$	$\varphi(t)$
Daubechies 2	0,48296291314453, 0,83651630373781, 0,22414386804201, -0,12940952255126		
Symmlet 4	0.03222310060408, -0.01260396726205, -0.09921954357696, 0.29785779560561, 0.80373875180680, 0.49761866763256, -0.02963552764603, -0.07576571478936		

Algorithm of obtaining those figures is simple:

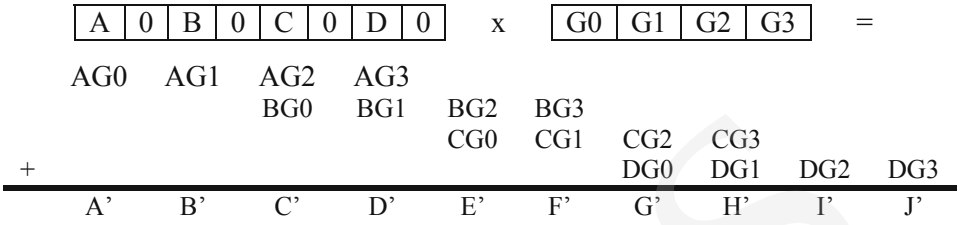
– we put *h(n)* coefficients into an array if we want to get $\psi(t)$ function, or *g(n)* coefficients if we want to obtain $\varphi(t)$ function

A	B	C	D

– thereafter, until the array will have a suitable number of elements, we repeat the operations:
  – we do the „upsample” operation, i.e. after each value we insert 0

A	0	B	0	C	0	D	0

  – we multiply all elements of the array thought scaling coefficients *g(n)* according to the scheme:



By this algorithm, we obtain more and more precise approximation of a function:

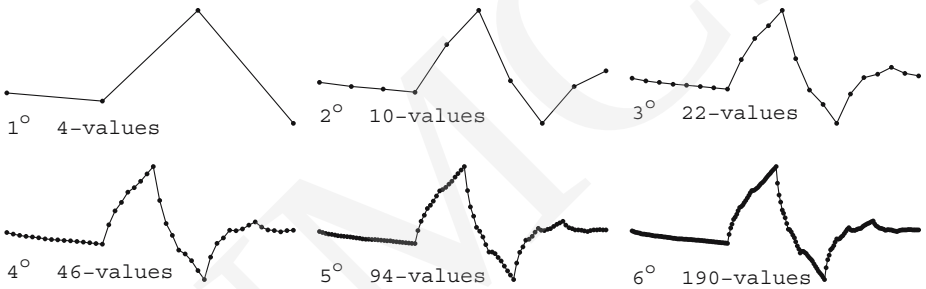


Fig. 8. Consecutive approximations of the Daubechies 2 wavelet obtained by the recursive algorithm

We need one more step, to use a such created wavelet in the CWT. Because CWT consists if many scales *a,* therefore we need $\psi(t)$ wavelet with various width *d*. In continuous wavelets it is not a problem, because having a formula, we can generate a wavelet with arbitrary width *d.* However this recursive algorithm gives only discrete widths (e.g. 4, 10, 22, 46, 94…), other widths we have to interpolate. To create such a interpolated wavelet $\bar{\psi}(t)$ (for instance – width $d = 18$) we have to remember about several things:

– $\left(\sum_t \psi(t) = 0\right) \Rightarrow \left(\sum_t \bar{\psi}(t) = 0\right)$. If we want a $\bar{\psi}(t)$ to meet the condition (1), we can lower or higher the function so that a sum was equal zero. Therefore we have to compute its sum $\Delta = \sum_{t=m}^{n} \psi(t)$, compute an arithmetic mean $\delta = \Delta / (m - n + 1)$, and subtract this value from every sample $\bar{\psi}(t) = \bar{\psi}(t) - \delta$, where $t = m,\ldots,n$. It is a good approximation, because $\bar{\psi}(t)$ does not differ much from $\psi(t)$, so Δ is small, so δ is even smaller. Instead of shifting samples $\bar{\psi}(t)$ horizontally to meet the condition (1), we move them vertically.

– All wavelets $\bar{\psi}(t)$ have to be interpolated from one wavelet. We can not $\bar{\psi}_{d=18}(t)$ wavelet interpolate from $\bar{\psi}_{d=22}(t)$ wavelet, and $\bar{\psi}_{d=24}(t)$ wavelet from $\bar{\psi}_{d=46}(t)$ wavelet. $\bar{\psi}_{d=22}(t)$ and $\bar{\psi}_{d=46}(t)$ wavelets are consecutive approximation of recursive algorithm, but they can be for instance slightly shifted against each other. Then, on a spectrum, we will see horizontal stripes, that are shifted against each other. We should generate the largest needed wavelet, e.g. $\bar{\psi}_{d=3070}(t)$ and from it we should generate all $\bar{\psi}(t)$ wavelets.

Fig. 9. Another piece of CWT. The spectrum, on which wavelets $\bar{\psi}(t)$ was interpolated from various wavelets $\psi(t)$. We can see clearly the shifts between scales, with different interpolation source wavelet

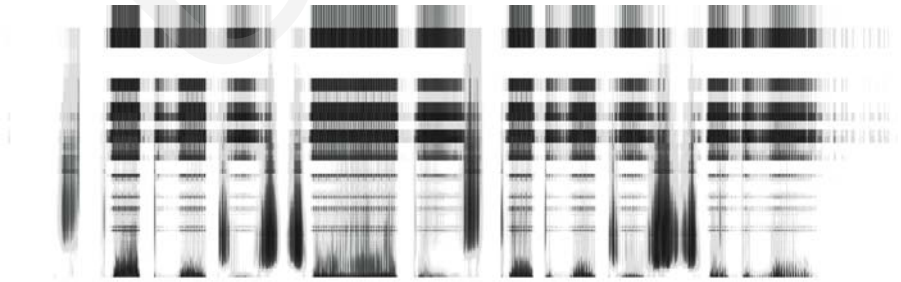


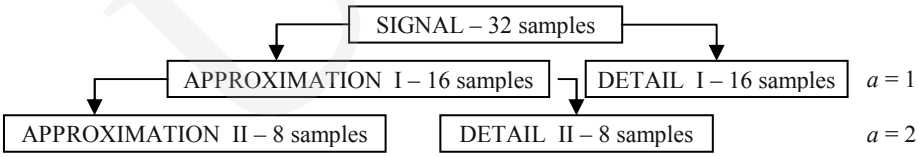
Fig. 10. Spectrum of 3-secunds speech signal with badly normalized scales. We can see stripes of overstated values (darker stripes) for scales with to large energy

– Wavelet, on consecutive *a* scales, have to have normalized energy, because with increase of a wavelet width, increases samples count representing the wavelet. So, to keep energy of all $\bar{\psi}(t)$ wavelets at the same level, the amplitude of the wavelets should be inversely proportional to its samples count. Otherwise some scale will be "stronger" or "weaker", which will cause the corresponding scales to have overstated or underrated values. In the case of continuous wavelets, we have $1/\sqrt{a}$ factor in formula (3),

which is liable for normalization. Discrete wavelets are generated in different way. If we want the wavelet to have energy equal C, we only have to compute its energy $E = \sum_{t=n}^{m} \bar{\psi}(t)^2$ and scale vertically every sample $\bar{\psi}(t) = \bar{\psi}(t) \cdot \sqrt{C/E}$ where $t = m,\ldots,n$.

6. DWT – Discrete Wavelet Transform

The DWT algorithm can only be applied to discrete wavelets. It is much faster than CWT, but much less precise. Its procedure rule is to divide signal on a group of values that describes general values and a group describing detail. Number of samples on each group is by half smaller then samples count of input signal. Values that belong to those groups represents one scale *a* of DWT spectrum. We repeat this procedure, taking approximation as an input data. Thereby we get another scale a+1, which will be by half smaller (in number of samples). We can compute consecutive scales while samples count is greater then 1. Because in each scale we have to times fewer samples, increasing scale by 1 in DWT is equivalent to doubling scale in CWT. Therefore in those transforms, scale *a* has different meaning, but they related with each other with simple correlation.



Procedure scheme is simple. Let us assume, that input signal consists of 8 samples *I0,...,I7*. Having wavelet coefficients *H0,H1,H2,H3* and scaling coefficients *G0,G1,G2,G3,* we have to multiply the signal through appropriate matrix:

$$\begin{bmatrix} L0 \\ H0 \\ L1 \\ H1 \\ L2 \\ H2 \\ L3 \\ H3 \end{bmatrix} = \begin{bmatrix} G0 & G1 & G2 & G3 & & & & \\ H0 & H1 & H2 & H3 & & & & \\ & & G0 & G1 & G2 & G3 & & \\ & & H0 & H1 & H2 & H3 & & \\ & & & & G0 & G1 & G2 & G3 \\ & & & & H0 & H1 & H2 & H3 \\ G2 & G3 & & & & & G0 & G1 \\ H2 & H3 & & & & & H0 & H1 \end{bmatrix} \cdot \begin{bmatrix} I0 \\ I1 \\ I2 \\ I3 \\ I4 \\ I5 \\ I6 \\ I7 \end{bmatrix},$$

Lx – down-pass values, i.e. approximation
Hx – high-pass values, i.e. detail.

Then we have to sort output array and repeat the procedure for *Lx* values, until there is more than one of them. The final result consists of all detail values and last approximation value.

I0	***I1***	***I2***	***I3***	***I4***	***I5***	***I6***	***I7***
1L0	*1L1*	*1L2*	*1L3*	*1H0*	*1H1*	*1H2*	*1H3*
2L0	*2L1*	*2H0*	*2H1*				
3L0	*3H0*						
3L0	***3H0***	***2H0***	***2H1***	***1H0***	***1H1***	***1H2***	***1H3***

Now we only have to properly illustrate our result. Location of approximation and detail values is depicted below:

I0	I1	I2	I3	I4	I5	I6	I7

1H0	1H1	1H2	1H2
2H0		2H1	
3H0			
3L0			

Fig. 11. Location scheme of DWT values

I0	I1	I2	I3	I4	I5	I6	I7

CWT 1,0	CWT 1,1	CWT 1,2	CWT 1,3	CWT 1,4	CWT 1,5	CWT 1,6	CWT 1,7
CWT 2,0	CWT 2,1	CWT 2,2	CWT 2,3	CWT 2,4	CWT 2,5	CWT 2,6	CWT 2,7
CWT 3,0	CWT 3,1	CWT 3,2	CWT 3,3	CWT 3,4	CWT 3,5	CWT 3,6	CWT 3,7
CWT 4.0	CWT 4.1	CWT 4.2	CWT 4.3	CWT 4.4	CWT 4.5	CWT 4.6	CWT 4.7

Fig. 12. Location scheme of CWT values

Our DWT spectrum will look like this:

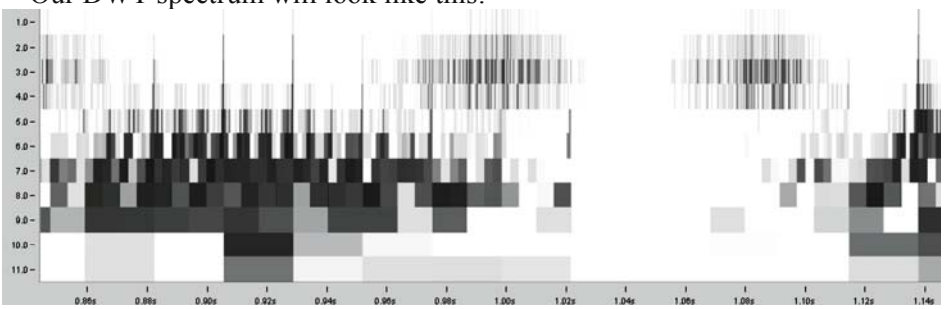


Fig. 13. A piece of DWT spectrum with scale *a* on vertical axis

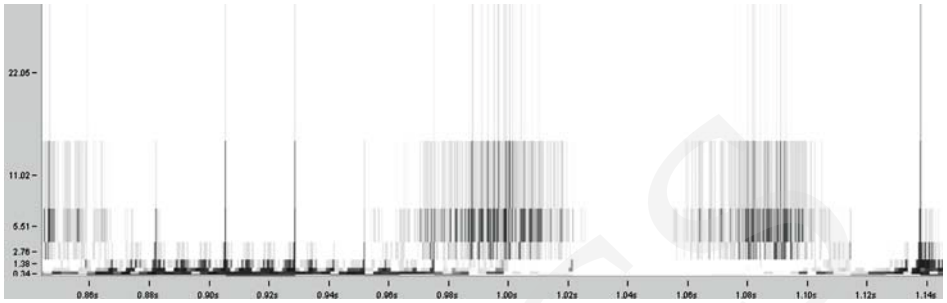
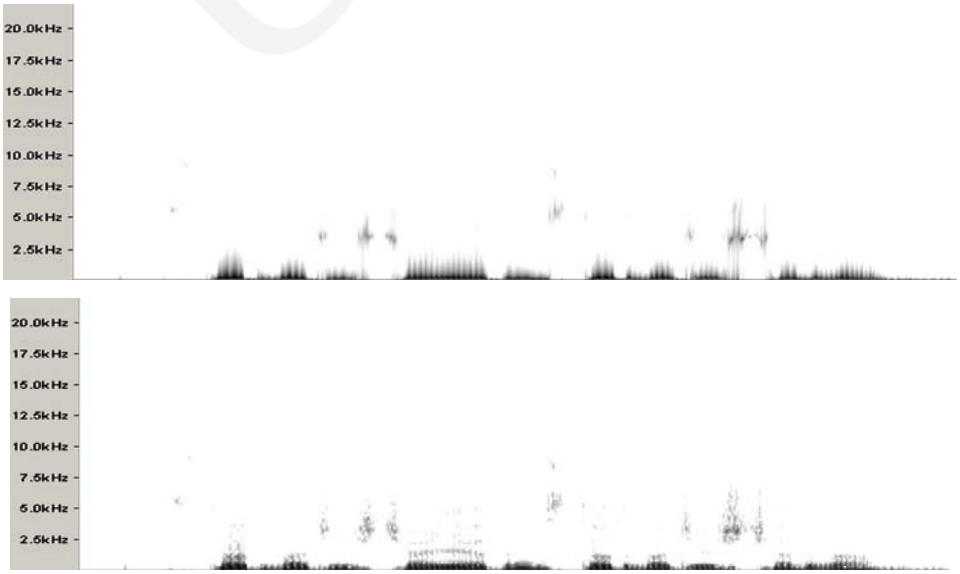


Fig. 14. The same piece of DWT with frequency on vertical axis

7. Summary

As we can see in Figures 11 and 12, wavelet transform does not determine a fixed time-frequency resolution, as it is in the Fourier Transform. For high frequencies, because a signal changes very fast, a pressure is put on time localization (horizontal axis is dense) instead of frequency localization (vertical axis is rare). For lower frequencies the time is less important, so this proportion is inverse. The CWT algorithm goes even one step further, because its time resolution is everywhere the same – regrettably we achieve this through increasing the computation time.

As we can see in Figure 15, DWT, CWT, FFT and LPC spectrums are similar to each other, but non the less – they differ.



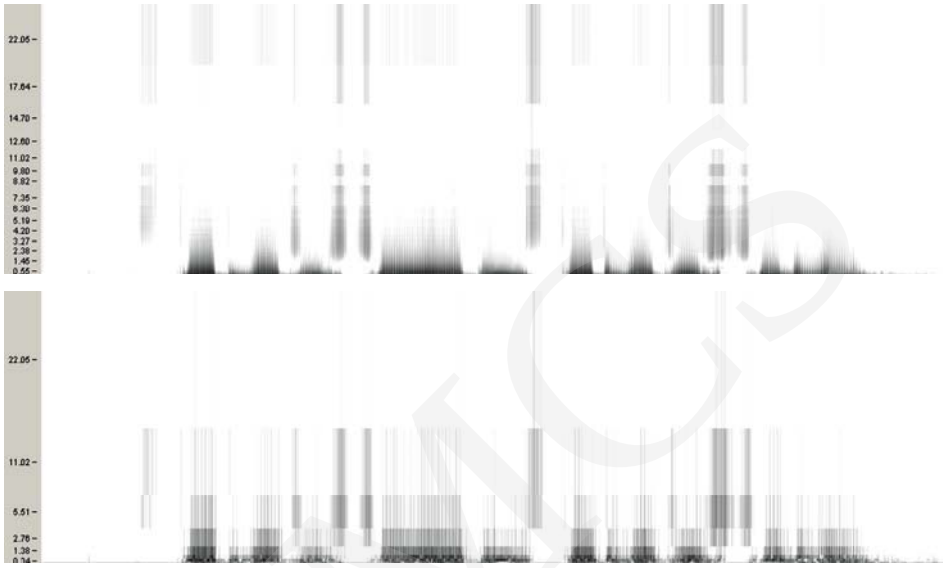


Fig. 15. Spectrum comparison of 3-second utterance "Sto dwadzieścia jeden, sto dwadzieścia dwa". From the top: LPC, FFT, CWT, DWT. In the case of DWT and CWT we used Daubechies 2 wavelet

As a conclusion, in our opinion, the wavelet analysis should be investigated through scales criterions rather than through frequency criterions. Frequency axis creation is loaded with probably to many roundings errors, to be the base to further computations. It is not a disadvantage – we should simply concentrate on modeling appropriate wavelets, so that finding a high correlation between such a wavelet and the input signal, was the information itself. In this case we would not have to analyze frequencies but scales.

Acknowledgements

Scientific work partially financed from the grant of Vice-Rector of Maria Curie-Skłodowska Univeristy.

The authors thanks Natalia Fedan for language corrections.

References

[1] Jansen A., la Cour-Harbo A., *Ripples in Mathematisc – The Discrete Wavelet Transform*. Springer-Verlag, Berlin Heidelberg, (2001).
[2] Białasiewicz J.T., *Falki i Aproksymacje*. Wydawnictwo Naukowo-Techniczne, Warszawa, (2000), in Polish.
[3] Polikar R., The wavelet tutorial. http://users.rowan.edu/~polikar/WAVELETS/WTpart1.html
[4] Barański R., Falki, falki, falki. http://kmiw.imir.agh.edu.pl/falki/
[5] Nayak J., Bhat P.S., Acharya R., Aithal U.V., *Classification and analysis of speech abnormalities*.

[6] The MathWorks, *Matlab 7 Help – Wavelets: A New Tool for Signal Analysis*
[7] Wołowik P., *Cyfrowa analiza sygnałów potencjałów wywołanych*. Politechnika Poznańska, (2002), in Polish.
[8] Getreuer P., *Filter Coefficients to popular wavelets*. May (2006).
[9] Philips W.J., *Wavelet and Filter Banks Course Notes*. January (2003), http://www.engmath.dal.ca/courses/engm6610/notes/notes.html