Feature Extraction Using Multisignal Wavelet Packet (WP) Decomposition

This is a very simple and naive introduction to feature extraction using wavelet packet transform, which by no means is meant to be a solid technical note but rather an easy description of wavelet packet based signal feature extraction process.

The Wavelet Packet transform may be thought of as a tree of subspaces, with $\Omega_{0,0}$ representing the original signal space, i.e., the root node of the tree. The main idea here is similar to that of the Fourier transform, but, instead of using sines and cosines we are using a family of wavelet functions. The idea is to decompose the signal into a number of subspaces using the wavelet functions chose (Symmlet, daubechies, etc…). Such decomposition is usually achieved by translating (pushing to the right or to the left) and scaling (amplifying or attenuating) the selected wavelet function and projecting the signal on the subspace represented by the specific scaling and translation. If you can control the scaling and the translation and know about the frequency division then you have all what you need to prepare a time-frequency plot.

Let's talk in a simpler way, consider the wavelet packet tree in Fig.1. Let us assume that the signal of interest comes from an electromyogram (EMG) sensor that measures your muscle activity as shown in Fig.2 below. The figure has two sensors, we will consider one of them for now, but the same apply on both sensors. The EMG is a nonstationary signal that is stochastic (stochastic means random) in its nature, with the most important energy distributed in the frequency range of 15Hz–to-500Hz (different books report slightly different range, don't worry about that).

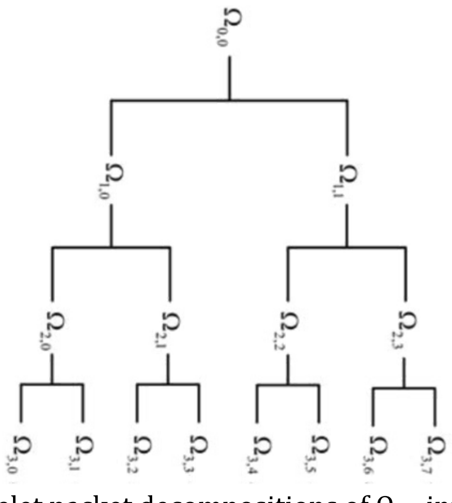


Fig.1. An example of the wavelet packet decompositions of $\Omega_{0,0}$ into tree-structured subspaces

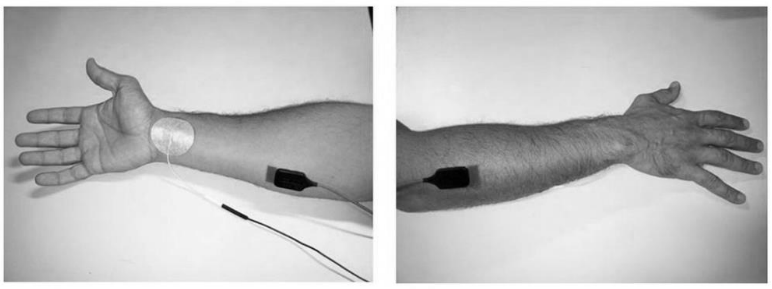


Fig.2 Two EMG sensors on a subject forearm (interior and exterior views).

Let us assume that using these sensors and their corresponding hardware we sample the EMG signal from each sensor with a sampling frequency of 4096Hz, i.e., we get 4096 samples of data in each second. As this is the sampling frequency then our Nyquist frequency is 2048Hz (The Nyquist frequency, named after electronic engineer Harry Nyquist, is 1/2 of the sampling rate). Now look carefully at Fig.3, and note that the signal at the original signal subspace ($\Omega_{0,0}$) has a frequency range from 0Hz to 2048 Hz. Further divisions of the this subspace into subspace ($\Omega_{1,0}$) and subspace ($\Omega_{1,1}$) will divide the frequency range into 0Hz-to-1024Hz for subspace $\Omega_{1,0}$ and 1024Hz-to-2048Hz for subspace $\Omega_{1,1}$ and so on for the rest of the subspaces.

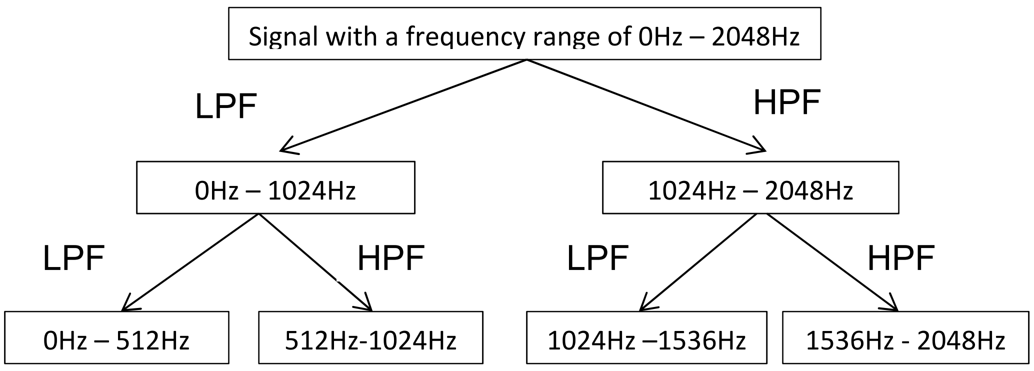


Fig.3 Example decomposition of an EMG signal that has not been prefiltered.

Now wait a second, if the most imprtant range for the EMG signal is within 0Hz to 500 Hz why did we not filter the signal before hand? Well do it and write down the new tree partitioning of frequencies as an excersis. Right now you may have many questions, like how is the frequency partitioning achieved? Easy, use low pass filters (LPF) and high pass filters (HPF). Adjust the scale of the wavelet functions and their translation and do that at different scales and translations and you can get the above tree. In comparison to FFT, we usually project our signals into subspaces as well, but, these with sines and cosines that have different frequencies, scaling and phase. The theory is beyond this tutorial am giving here, just refer to any wavelet book.

Now, how do we extract features? Up to what you should understand from above is that so far we have been only dividing our signal into new subspaces that have different frequency coverage. **WAIT**, we should say and time domain coverage as well. The wavelet transform in general is a time-frequency analysis tool, but where is the time aspect here? Did you know that in Fig.3, the first node could have for example only 512 samples at the first node and much less than that at the rest of the nodes (its up to you to choose the window size of data). For example in EMG signal classification we must make sure that everything is done in less than 300 msec, i..e, feature extraction and decision making for each window should be done quickly. Assume the amount of data we look at is 512 samples (512 (samples) /4096 sample/sec = 125 msec). This will give us around 175 msec of time to do classsification and any other tasks. Okay, so in Fig,3 if we start at node $\Omega_{0,0}$ with 512 samples, then we get around 256 samples at node $\Omega_{1,0}$ and around another 256 samples at node $\Omega_{1,1}$. Thus we are having a time domain partitioning as well. Remember it's a time-frequency tool.

Now, lets answer some questions that may rise here.

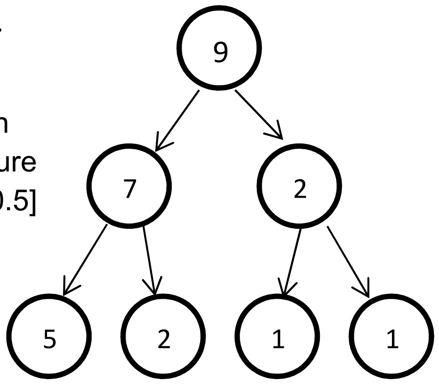
- **How do you extract features using this tool?** Well, it is all up to you.

In the problem of EMG classification we know that the energy of the EMG signal is one of the most important features. One simple solution to the aforementioned question is use the enery of the signal that you have at each subspace or simply node. You can also look at the root mean square, the variance of the signal at each node, or entropy, or any other statistical descriptor. Once you have the feature vale for each node then just put them all beside each other and that is your feature vector as shown in the example below (just an example, it does not have any special meaning).

In this example the feature vector is [9,7,2,5,2,1,1].

But, what if I have multiple sensors or channels? Well do the same for the next channle and put both feature vectors together. As an exmaple if the feature vector from the second channel is [10,9,1,6,3,0.5,0.5] then the resultant feature vector would be the concatenation of both vectors

Features =[9,7,2,5,2,1,1,10,9,1,6,3,0.5,0.5]



Note that the first level in the above example from which we extracted the feature value of 9 is located at level 1 in matlab, while in the books this is usually denoted as level 0.

Another example could be just use the resultant signals at each node and put all of these beside each other and then you have a huge feature vector. By the way, each of the resultant signals at each node is made up of a set of samples called the wavelet coefficients at that specific subspace or node.

- **How many levels of decomposition do I chose?** Well, it all depends on your application and how many features you want to extract.

  In the code, I ask the user to input his preselected number of decomposition levels. I also give an option at line 66 to do that for him/her if they don't know what to do using `J = log(size(x,1))/log(2);`
  For example, I told you previously that the EMG signal we care about is in the range of 15 Hz to 512 Hz, so for me to extract at least 8 features from that range then how many decompsiition levels do I need to use?. Note that in Fig.3 I had to deompose the signal twice and I only got the range of interest 0Hz to 512 Hz at the first nose of level 3 (how to get 8 more features from this range?). A smart solution would be just pre-filter the signal to get more control into the number of decomposition levels and frequency ranges of interest. Or keep deomposing further down till you reach your goal.

- **Level 1 already covers the frequency range of level 2, and level 2 already covers the frequency range of levels 3 and so on. Why do I need to decompose into further levels?** Well, it all depends again on your own application and your signal properties.

  Your signal could show its best features at different scales so decompse and seach for these. Also, remember the concept of the scale, may be for example your signal shows a powerful feature the range of 0 -512Hz that could highly interact with the feature at 8-32Hz!!!

Now lets talk matlab,

x	is you input signal (one column or multiple columns to represent multiple signals),
winsize	Window size (512 samples in my example above),
wininc	Windows iuncrement depends on your problem,
J	number of decomposition levels to go down to,
toolbox	('matlab' if you have the matlab wavelet toolbox or 'wmtsa' if you plan to use the wmtsa toolbox from the intrenet).

- **Can I extract the features only from the last level?** Yes, just go to line 132 and change `for i=1:nL` into `for i=nL:nL` with nL = number of levels.

- **What do you really mean by multisignal**? Well the smart guys at mathworks created a function that allowed us to deompse multiplie signals at the same time using the wavelet transform to save the time. What I did was to use this function to write down the multisignal wavelet packet decomposition and extract features in a computationally efficient manner. Just try to do the same task using the normal wpdec command and observe the difference in time.

- **Finally, do I need feature selection?**
  Obviously, if you decompse so many times then you end up with so many features and feature selection will help you reduce the computational cost by focusing on the most important features only to aovid redundancy and focus of the relevant features. Don't forget, the features extracted from each level do already cover the frequency range of the features extracted from the lower level, i.e., you do have information redundancy here that you need to carefully consider.