Time-Frequency Analysis of Internet traffic measurements

Mentor: Amos Ron

Level I At the IDR FrameNet portal, you will find internet traffic datasets in the collection called *framenet::internet-1*. All the data in the collection was collected at one location on the campus network at a sample rate of 1 point for every 5 minutes. Each signal is length 2048, which means that the signals span just over 7 days.

The measurements are of different types of internet traffic moving either into or out from one part of campus to another. The types of traffic are: bytes, packets, flows and tx/rx (which is a count of active computers (t)ransferring or (r)eceiving data.)

1. Explore the time localization properties of framelets. Find the data *internet/Internet I/10.07/out/flows*. Using the systems RS4, and PS4-1typeII, generate two figures, each with three graphs. For each figure, fix a system. The first graph is to capture only events of duration $\geq$ 5 minutes and $<$ 3 hours. In the second graph, capture events lasting $\geq$ 3 hours and $<$ 20 hours and in the third graph, capture events lasting $\geq$ 20 hours. (Perform the analysis in FrameNet. Download the results and build the figures using matlab.)

2. We momentarily leave wavelets to look at perfect frequency localization. Observe in the *flows* signal two clearly visible features: daily periodicity and a large spike towards the end of the signal. The daily periodicity is a feature that signal is well-localized in frequency. As a result, it should be possible to isolate this feature in the frequency domain. Using matlab, load *flows* into a variable called (say) **flows**. Using the matlab cammands `fft`, `ifft` and `find`, capture this daily periodicity by setting all values of `fft`(**flows**) below some threshold to 0. Let's call this new signal (in the time-domain) **th_flows**. Plot **th_flows** in the time domain. Compare **th_flows** and **flows** $-$ **th_flows** to the signals produced in part 1. (You may also find it interesting to compare **th_flows** to the function $a\mathbf{s} := a\sin(14\pi t/2048)$, where $t := [1 : 2048]$ and $a := \langle \mathbf{s}, \mathbf{th\_flows} \rangle / \|\mathbf{s}\|^2$.) What is the signal-to-noise (s/n) ratio of **th_flows** to **flows**?

3. Now capture the "spike" of **flows** using a framelet system of your choice. Demonstrate that you can make a "spikeless" signal called **nospike_flows** that matches **flows** everywhere except in a small neighborhood of the spike. Find the s/n ratio of **nospike_flows** to **flows**. Can you do as well using only thresholding and the fourier transform?

4. Create a corrupted signal called **crpt_flows** that is the same as **flows** except that all values between 1500 and 2000 are set to 0. Repeat part 2 (that is, use thresholding to capture the daily periodicity) with this corrupted signal and look at the result in the time domain. Capture, using the framelet system PS4-1typeII features of duration $\geq$ 3 hours. Which

reconstruction has a smaller s/n ratio? Visually, which reconstruction is "better"? Plot the graphs and highlight artifacts of the reconstructions.

5. In wavelet lingo, one refers to high freqnency events as those features which are captured in the first few frequency levels of a decomposition. Compare the "high-freqnency" graphs you made in part 1 to the signal **flows** − **th_flows** you built in part 2. How does the wavelet "high freqnency" signal compare to the "honest" high freqnencies of **flows** − **th_flows**? (You may wish to compare their fourier transforms.)

Level II Repeat **Level I** parts 1, 2 and 5 but this time, use the signal called *internet/Internet I/10.28/out/bytes*.

Level III This exercise explores the approximation orders of various systems.

1. In FrameNet, find the data collection called "*framenet::Tour*". For each system HAAR, RS2, and PS4-1typeII, partially reconstruct the function *splines/cubic* and *splines/random* by including only the frequency levels below -6.

2. Download the original signals and their partial reconstructions and compute (in matlab) the s/n ratios. Build a table that ranks the three systems according to their performance as measured by the s/n ratio. Include in the table the approximation orders of each system.

3. Make a second table, but this time analyze the two signals **flows** and **bytes**.

4. Does one system consistantly do a better job than the others? Explain what this has to do with approximation order.