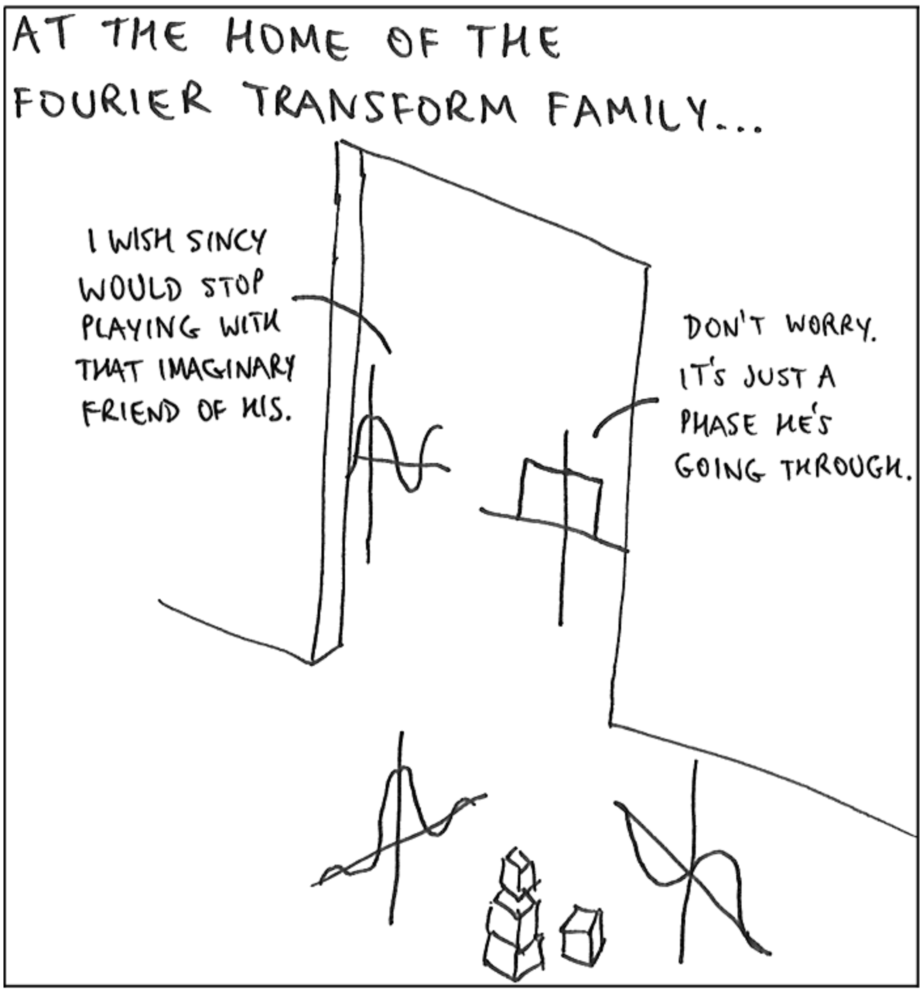
DTU Health Technology, Hearing Systems section
Building 352, DTU, 2800 Lyngby
Phone: 45 25 39 30, Fax: 45 88 05 77 www.hea.healthtech.dtu.dk/
Course 22051: Signals and Linear Systems in Discrete Time

Nerdy...


AT THE HOME OF THE FOURIER TRANSFORM FAMILY...
I WISH SINCY WOULD STOP PLAYING WITH THAT IMAGINARY FRIEND OF HIS.
DON'T WORRY. IT'S JUST A PHASE HE'S GOING THROUGH.

Bastian Epp: 45 25 39 53 (office), e-mail: bepp@dtu.dk

Time, frequency and butterflies

1 Convolution revisited

We discovered the duality between time and frequency representation of signals. In this part you will use the frequency representation of signals to perform an operation in the frequency domain that corresponds to a convolution in the time domain. Concepts like this are generally very helpful, since mathematical operations might be much easier in the frequency domain compared to the time domain (or vice versa). This becomes very relevant once you need to care about resources (e.g., embedded systems).

1.1 Equivalence of linear and circular convolution

The convolution we met can be divided into "circular convolution" and "linear convolution". Circular convolution somewhat assumes that your signal is periodic, i.e., once you shift the signal, everything that is pushed out "at the end of the vector" is added to the beginning (as if turning it on a circle). Linear convolution on the other hand can be interpreted as if the signal is zeros all the way back in time and all the way to infinity after the offset of the signal. Both of these are equivalent under certain conditions - and this is what we are going to explore now.

Please implement a function that processes your input signals `x` and `h`, such that the equivalent of a linear convolution is performed in the frequency domain to obtain the output `y`. The function should take the input signals as input arguments and return the convolved time domain signal.

- Consider the the following algorithm (don't forget the "good practice of Matlab programming"!):

  1. Compute (and store) the length of the convolution product using `length()` (of course without actually convolving the signals)

     $$y(n) = x(n) * h[n]$$

  2. Zero-pad the input signals to the length of the convolution product by using the `zero_pad()` function you wrote in the assignment
  3. Perform the convolution in the frequency domain to obtain $H(\omega)$ by pointwise multiplication using the operator `.*`
  4. Use the function `ifft` to obtain the time domain signal $h(n)$

- Compare your algorithm with the output of the `conv` function provided by Matlab. Either use some defined signals of your choice (exponentials, triangles, rects) or just generate a random sequence using `randn(N,M)` and look at the summed absolute difference of the two results

  `helpful commands:  rand, randn, norm`

- Compare your implemented algorithm in its performance with the `conv` function provided by Matlab. Use different combinations of input vectors (e.g., a short click together with a speech signal, or a speech signal together with the impulse response of a church - NOTE: remember the sampling frequencies might differ! In this case, use `resample()` to match the sampling frequencies).

  `useful commands:  tic, toc`

2 Convolution theorem applied to CORONA-numbers

Better than just looking at what the newspapers show in terms of CORONA numbers, we can as well just get the raw data and process them ourselves. Let's check out if the convolution theorem also can be applied to the count of positive detections of a lipid based nano particle.

2.1 Smoothening

Generate a moving average filter of order N and convolve its impulse response with the latest numbers of the pandemic in all Denmark. Please implement the solution in a way so you can easily change the order of your filter:

- Define the order of the filter and based on that generate the impulse response (IR)
- Convolve the IR with the raw data
- Plot the raw data and the data filtered with the moving average filter on top of each other
- Use averages over three days, one week and four weeks

It might look plausible, but the picture probably varies from what you have seen in the newspapers. That's odd.

- Estimate the shift you observe in the filtered signal compared to the raw data as a function of the order of the filter
- How could you compensate for the delay you observe?'
- How can the resulting delay be predicted?

2.2 Filtering by multiplication

Knowing that we also can do the processing in the frequency domain, this is probably what we want to do. So let's try to replicate the results of the previous session - just by operating in the frequency domain rather than the time domain:

- Define the order of the filter and based on that generate the impulse response (IR)
- Transform the raw data and the IR into the frequency domain
- Multiply the time series and inverse Fourier transform to get a time series
- Plot the raw data and the data filtered with the moving average filter on top of each other
- Use averages over three days, one week and four weeks

Hopefully you get the same picture as when operating in the time domain.

- Compare the spectra of the time series before and after processing
- How can this result be explained?
- How is the frequency-specific time delay caused by filtering encoded?