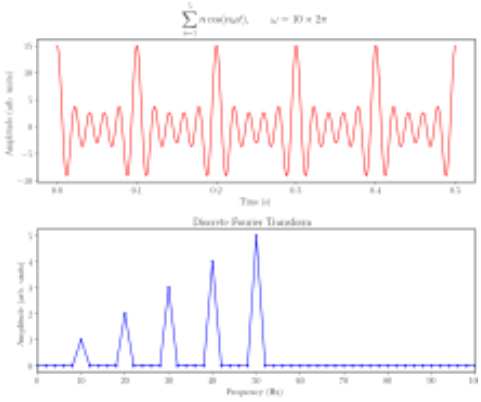
The fast fourier transform



FFT is one of the algorithms that have changed the world! By reducing the computational complexity of the Discrete Fourier Transform (DFT) from $O(n^2)$ to $O(n \log_2 n))$, it has made possible the analysis of signals that is now universally used in radio and radar communications, image processing, image compression…

Objectives:

-Write first a scalar code using Cooley-Tukey algorithm, following [1] and [2]. Try to avoid explicit recursion (recursion is nice but also inefficient). But, if you want to start with the recursive algorithm (simpler) do it.
- Write the parallel version [3]
-Write the code for inverse FFT

More complex stuff (maybe for a project)

-Extend it to the multidimensional case
-Apply it to image compression [4]
-Move to discrete wavelet transform (more interesting for images), see here for a general description

Note: for the direct FFT you can assume that the data is real, this may reduce computational complexity (at the price of a lower generality). However, the inverse FFT receives complex data (the coefficient of the Fourier transform are complex). Since inverse FFT follows the seme structure of direct FFT you can decide to do the complex version also for the direct case to reuse pieces of code. Or maybe go generic….

[1] https://en.wikipedia.org/wiki/Fast_Fourier_transform
[2] HandNotesOnFFT.pdf
[3] Quinn_Chap15.pdf
[4] summer_project_gillian_smith.pdf