Comp. Laboratory #4: Fourier Analysis using Python

Dominik Kuczynski (student id: 21367544)

November 30, 2022

1 Introduction

I use python to do Fourier analysis of different signals. First, the coefficients of a Fourier Series for various periodic signals are determined, with the help of a numerical integration method - 'Simpson's Rule'. The analysed functions include both sinusoidal signals, as well as non-differentiable functions such as the square function. These are then reconstructed from the coefficients and compared to the original.

In the next part, I use an analogous method to perform Discrete Fourier Transforms, intended for non-periodic functions sampled at a finite number of points. I investigate the relation of sampling rate to calculated Fourier components and calculate the Nyquist frequency for different examples. The methods are then used to calculate the DFT of several functions.

2 Methodology

2.1 Simpson's Rule

The numerical integration method that was used is Simpson's Rule. It works by approximating a given function with multiple parabolic elements, whose areas are computed and added up. this gives the approximate result:

$$\int_a^b f(x)dx \approx \frac{h}{3}\left[f(x_0) + 2\sum_{j=1}^{n/2-1} f(x_{2j}) + 4\sum_{j=1}^{n/2} f(x_{2j-1}) + f(x_n)\right]$$

with x_j being the chosen points of approximation, namely for some (small) h, $x_j = a + j \cdot h$.

2.2 Exercise 1,2: Fourier Series

Thanks to the Simpson's rule, the Fourier coefficients of different functions could now be computed according to the formulae:

$$f(t) \approx a_0 + \sum_{k=1}^{\infty} a_k cos(\frac{2\pi k}{T}t) + \sum_{k=1}^{\infty} b_k sin(\frac{2\pi k}{T}t) \tag{1}$$

$$\begin{aligned} a_0 &= \frac{1}{T}\int_0^T dt f(t) \\ a_k &= \frac{2}{T}\int_0^T dt f(t) cos(k\omega t) \\ b_k &= \frac{2}{T}\int_0^T dt f(t) sin(k\omega t) \end{aligned} \tag{2}$$

With T being the period of the original function, and $\omega = 2\pi/T$. To generate these coefficients numerically, a finite value of k_{max} that is large enough is used instead of ∞ in the sum limits. After obtaining a_k and b_k, (1) can be used to reconstruct the function.

2.3 Exercise 3: Discrete Fourier Transform

For non-periodic functions, the Fourier coefficients turn into a continuous Fourier Transform function. However, here the function is sampled only at some discrete points f_m ($m \in [0, N]$), and for numerical calculation purposes, the transform is also divided into N discrete steps (called F_n for $n \in [0, N]$), each of which is evaluated by:

$$F_n = \sum_{m=0}^{N-1} f_m e^{\frac{-2\pi imn}{N}} \tag{3}$$

To reconstruct the function, one uses:

$$f_m = \frac{1}{N} \sum_{n=0}^{N-1} F_n e^{\frac{2\pi imn}{N}} \tag{4}$$

Technical note: to speed up computation time (and out of curiosity), I looked up and used the Fast Fourier Transform method instead of the regular DFT. This, however does not significantly change the described algorithm.

Note that the values of F_n are complex.

The variables N and h determine how accurate the DFT will be, with the optimal values being such that $N \cdot h$ is a multiple of the period of the original function: $h = T/N$.

Another interesting thing to note about the computed Fourier Transform is that there is a symmetry around the middle point, $N/2$. It can be easily shown, as $F_n = F_{N-n}^*$:

$$\begin{aligned} F_{N-n} &= \sum_{m=0}^{N-1} f_m e^{\frac{-2\pi im(N-n)}{N}} = \sum_{m=0}^{N-1} f_m e^{-2\pi im(1-n/N)} = \sum_{m=0}^{N-1} f_m e^{-2\pi im}\, e^{2\pi imn/N} \\ &= \sum_{m=0}^{N-1} f_m e^{\frac{2\pi imn}{N}} = F_n^* \qquad \blacksquare \end{aligned} \tag{5}$$

The above derivation uses the facts, that $m \in \mathbb{Z} \implies e^{2\pi im} = 0$ and $e^{-x} = (e^x)^*$.

3 Results

3.1 Exercise 1

First, the accuracy of the Simpson's Rule method was investigated by calculating the integral $\int_0^1 e^x dx = e - 1$ for n=8 steps. The value obtained was 1.718284154699897, and it differs from the analytical result only at the 6th decimal place, showing impressive accuracy.

The Fourier Series was calculated for some test functions:

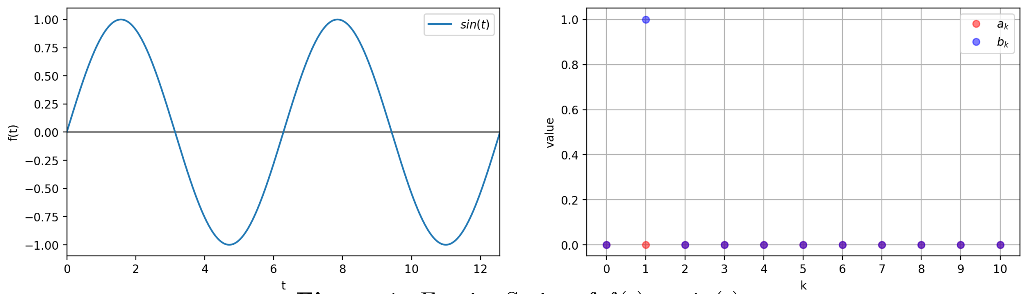


Figure 1: Fourier Series of $f(t) = sin(t)$

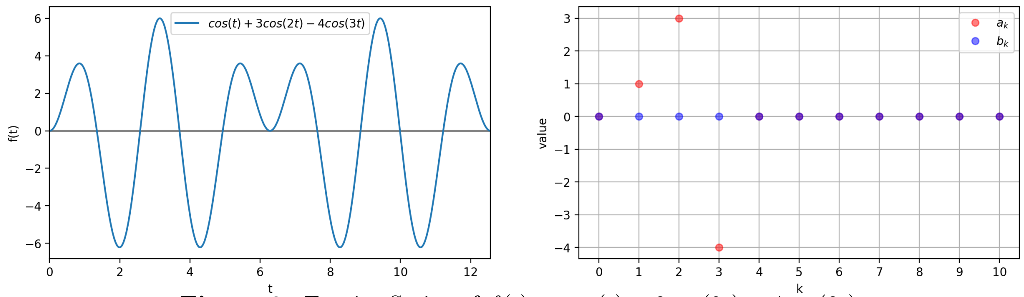


Figure 2: Fourier Series of $f(t) = cos(t) + 3cos(2t) - 4cos(3t)$

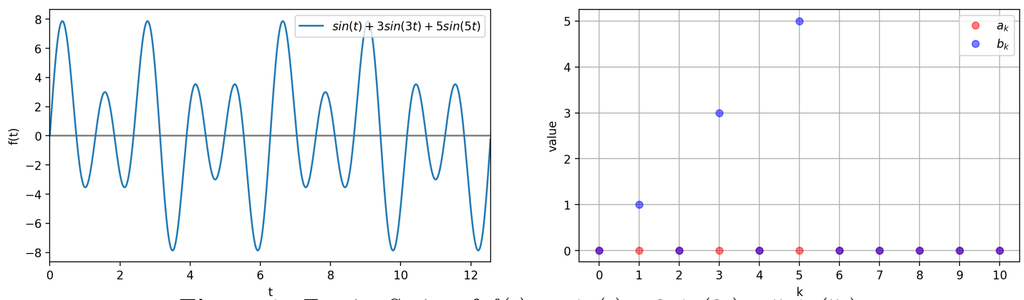


Figure 3: Fourier Series of $f(t) = sin(t) + 3sin(3t) + 5sin(5t)$

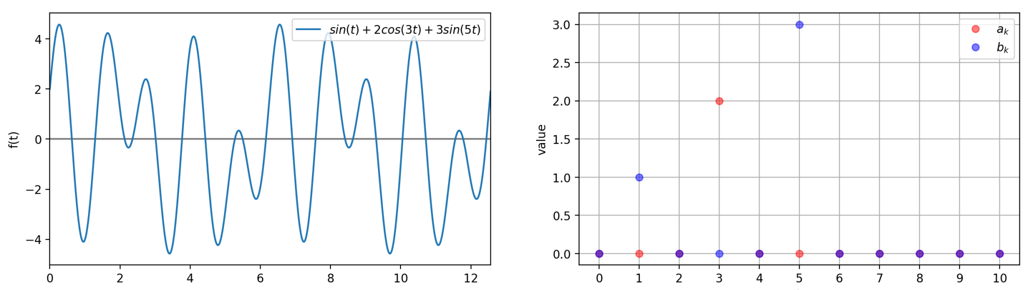


Figure 4: Fourier Series of $f(t) = sin(t) + 2cos(3t) + 3sin(5t)$

The expected values are the red points (A, B) for each component $Acos(Bt)$ of f, and blue points (C, D) for each $Csin(Dt)$ component of f - so the vertical axis corresponds to the 'strength' of a (co)sinusoidal component with a frequency given by k.

3.2 Exercise 2

Next, the Fourier coefficients were calculated for the square function:

$$f(\theta) = \begin{cases} 1 & 0 \leq \theta \leq \pi \\ -1 & \pi \leq \theta \leq 2\pi \end{cases}$$

The analytic result is given by:

$$a_k = 0 \qquad k = 1, 2, 3, \ldots$$
$$b_k = \begin{cases} 4/(\pi k) & k = 1, 3, 5, \ldots \\ 0 & k = 2, 4, 6, \ldots \end{cases}$$

The obtained numerical result:

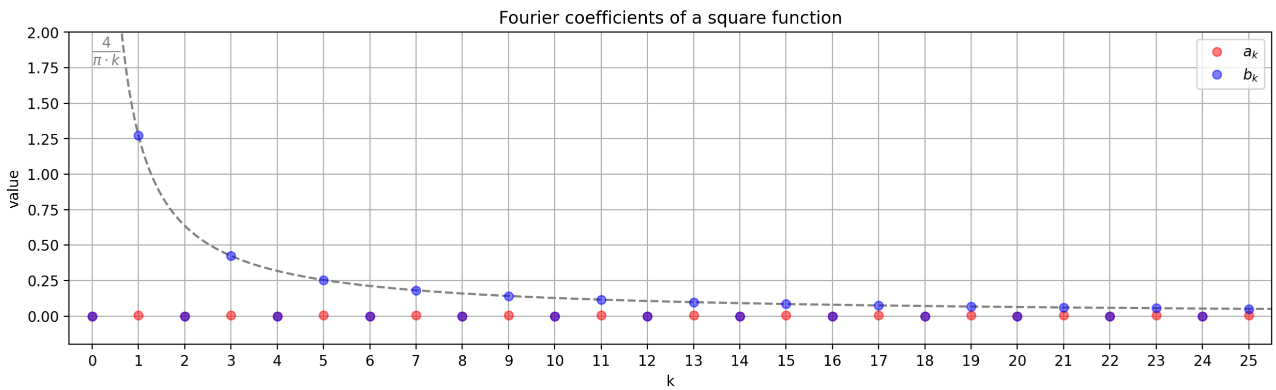


Figure 5: Fourier coefficients of a square function

This Fourier Series was then used to reconstruct the original function, with different values of k_{max} - the number of Fourier coefficients:

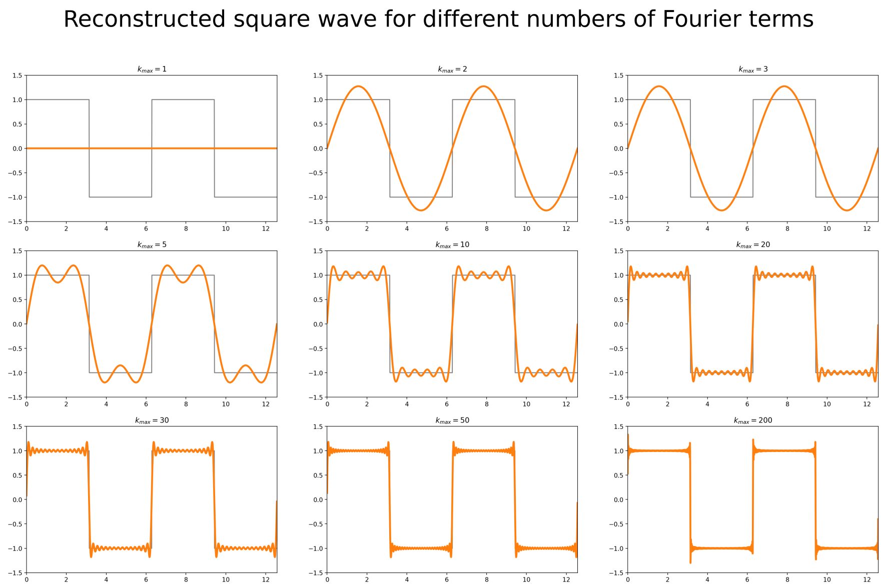


Figure 6: Reconstructing the square function

A slightly modified, rectangular wave was then analysed (with $\omega\tau = 2\pi/\alpha$):

$$f(\theta) = \begin{cases} 1 & 0 \leq \omega t \leq \omega\tau \\ -1 & \omega\tau \leq \omega t \leq 2\pi \end{cases}$$

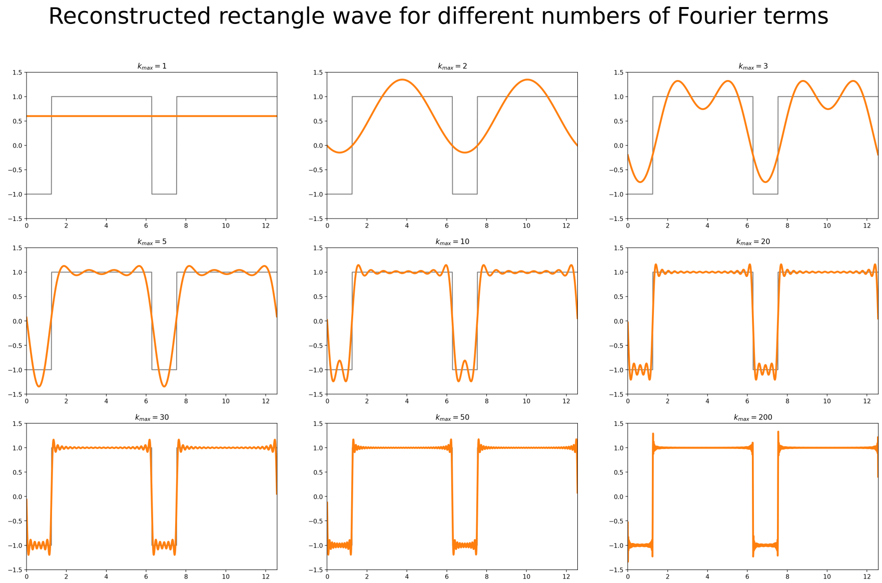


Figure 7: Reconstructing the rectangle function

3.3 Exercise 3

Here, the Discrete Fourier Transform was calculated for $f(t) = sin(0.45\pi t)$, sampled at $N = 128$ points every $h = 0.1$. The sampled points were:

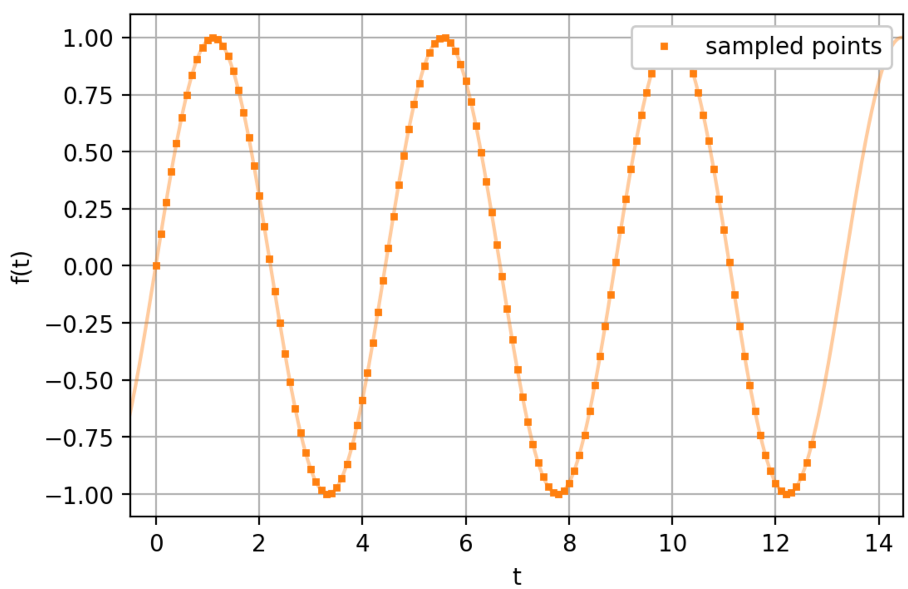


Figure 8: Plot of the function $f(t) = sin(0.45\pi t)$ and the sampled points

The calculated DFT values were plotted:

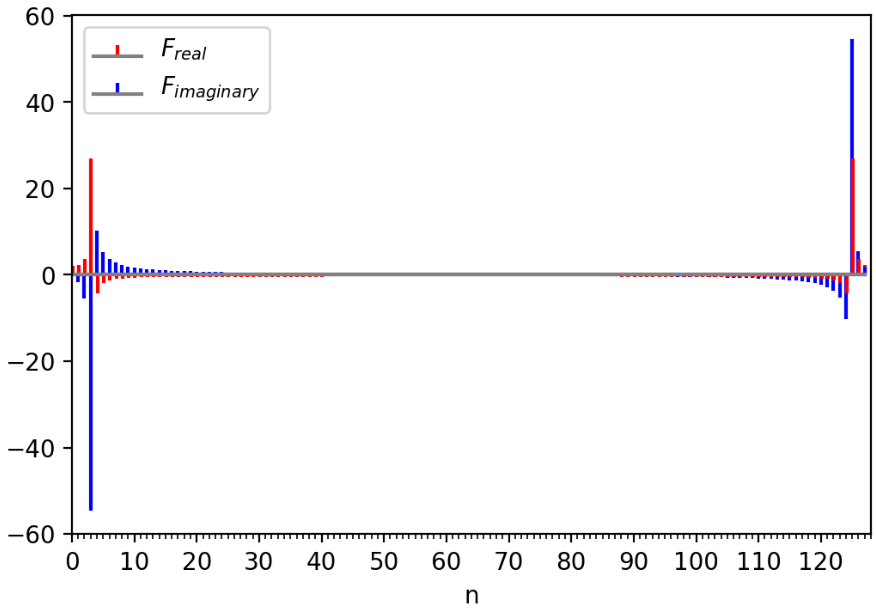
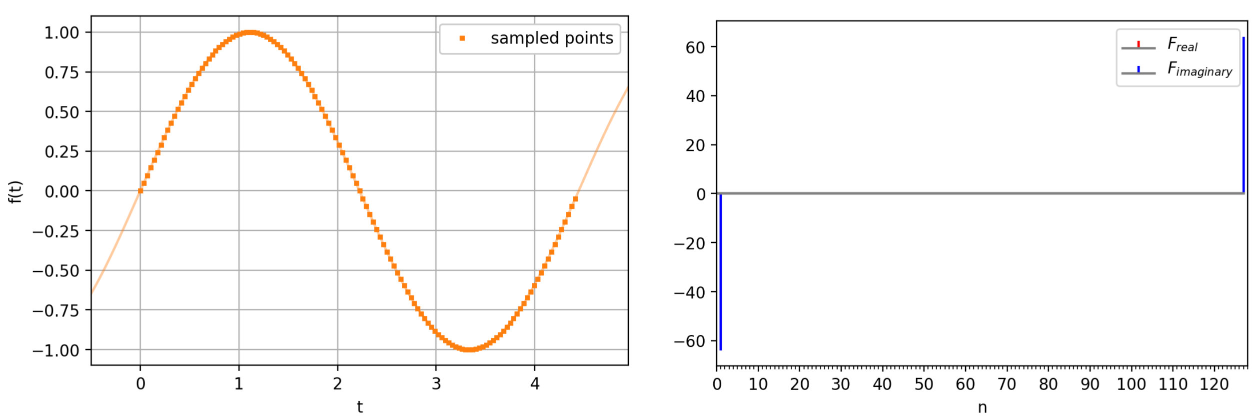


Figure 9: DFT of $f(t)$

There is a clear dominant component - the imaginary part of F_3. This corresponds to a frequency of $3 \cdot 2\pi/(Nh) \approx 1.47$. The actual frequency is $0.45\pi \approx 1.41$ so the two more or less correspond. However, the values of Fourier coefficients appear to be spread around the correct value, indicating that the values of N and h could be chosen to give better results.

Keeping $N = 128$ fixed, the optimal h is such, that $N \cdot h$ is a multiple of the period of the original function, here $N \cdot h = k \cdot \frac{2\pi}{0.45\pi} = k \cdot 2/0.45, \quad k \in \mathbb{Z}$. Choosing $k = 1$, the lowest optimal h comes out to be about 0.035, and gives the following:



(a) Plot of $f(t)$ and sampled points

(b) Fourier Transform components of $f(t)$

Figure 10: DFT of $f(t) = sin(0.45\pi t)$ with optimal parameters

For this optimal case, the back Fourier Transform was calculated, generating a series of points ("reconstructed points") f_m, according to (4):

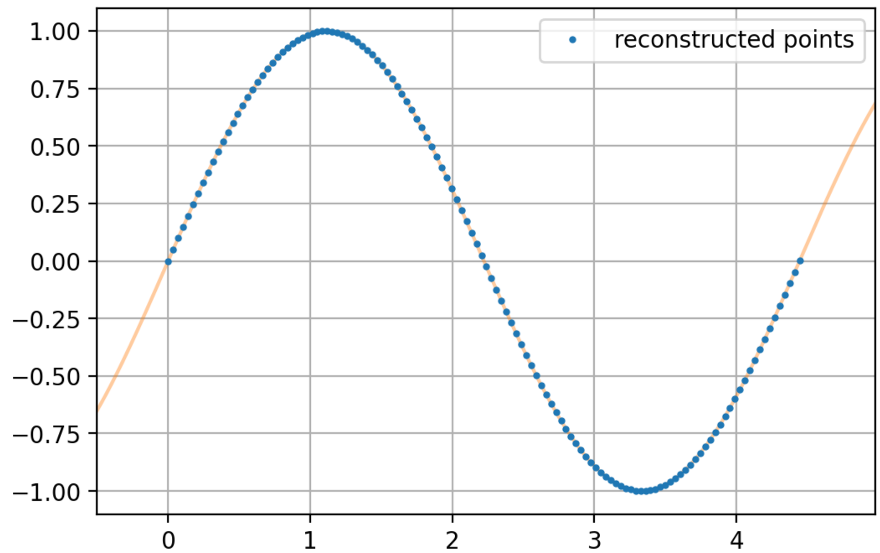


Figure 11: Plot of $f(t)$ and the reconstructed points

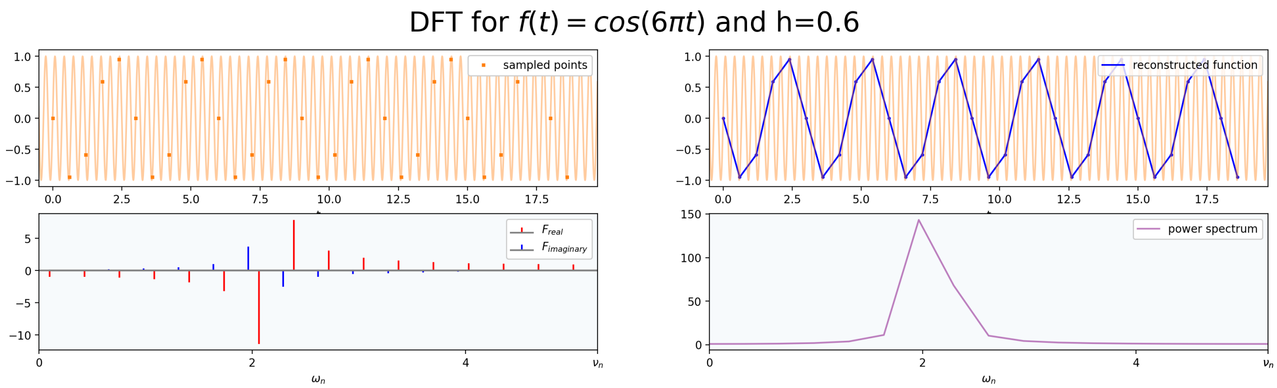
The reconstructed points coincide indistinguishably with the sample points, showing the effectiveness of the DFT method with optimal parameters.

Next, a different function was investigated: $f(t) = cos(6\pi t)$. It was treated like a function with an unknown period, and the DFT was performed for $N = 32$ and different values of h. To better illustrate the results, the Fourier coefficients and the power spectrum were plotted against corresponding frequencies, and up to the Nyquist frequency ν_n.

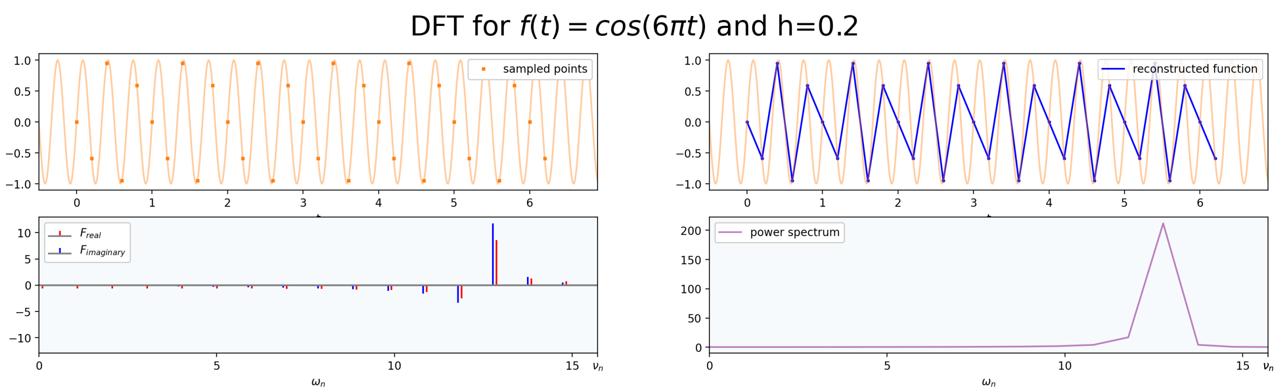
It is clear that for values of h such as 0.6, or 0.2, the resulting reconstructed signal does not represent the original well. This is particularly visible in 12a, where the reconstructed points seem to be assuming a different, much lower frequency. One of the reasons is because the actual frequency of the signal ($6\pi \approx 18.85$) is higher than the Nyquist frequency - the highest possible to resolve. As the sampling rate increases, we see a corresponding increase in ν_n, meaning that for low enough h, the original signal can be reconstructed accurately, as is in the case of 12d.

Figure 12: DFT of $f(t) = cos(6\pi t)$, and different parameters

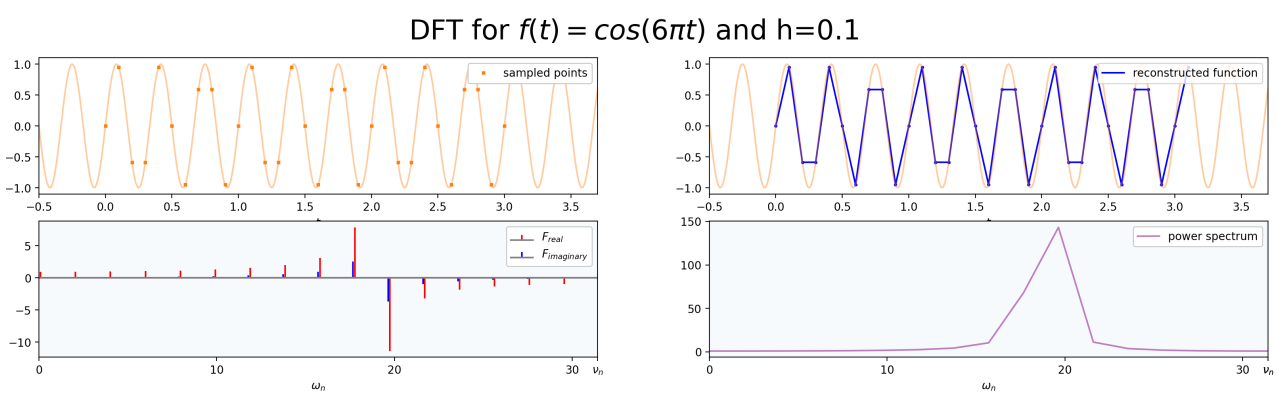
(a)



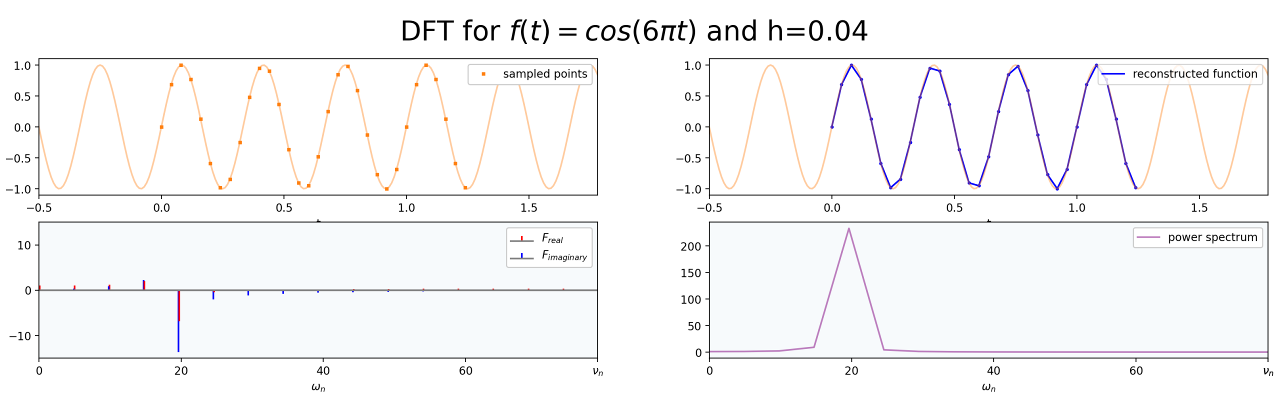
(b)



(c)



(d)



4 Conclusions

The Fourier Series Transform and the DFT are indispensable tools in signal processing, as they allow analysis in both the time- and frequency domains. Numerical versions of these two methods also perform very well when given the right parameters, however, especially with the DFT, one must make sure that the reconstructed signal actually corresponds to the given one. To ensure that, a high enough sampling rate must be used.