Numerical Algorithm 11

Lorenzo Ferri, Shalini Pandurangan

December 2018

Exercise 1

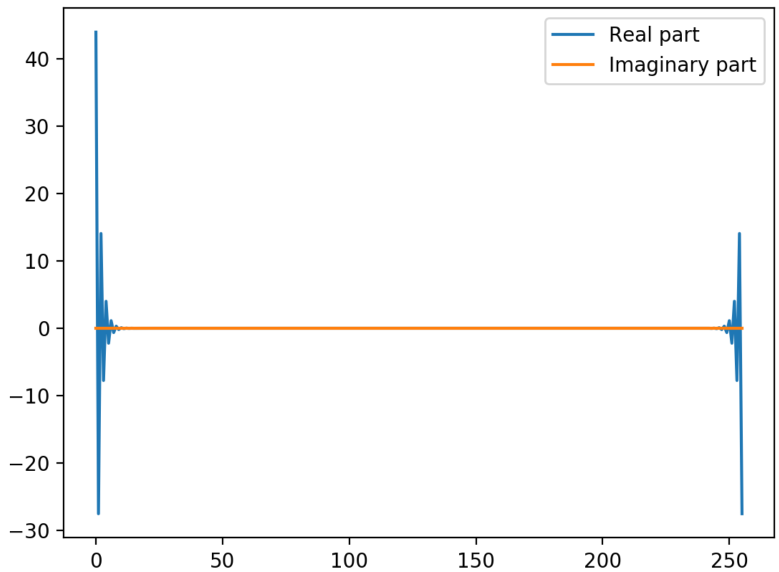
Submitted on paper.

Exercise 2

In order to compute the Fast Fourier Transform we need first to compute all the j_i. j contains all the number from 0 to n but ordered in a particular way. To achieve that we create a list simply from 0 to n and we transform each number like this:
We first convert each j into binary, also specifying how many digits in total we need, in our case 8. So for example 4 becomes 0000100.
Now we flip this binary number and convert it into integer again, in the end the process is like this:
$4 \to 0000100 \to 0010000 \to 16$
We do this in the function `reverse_bit`. Now that we have the j we can compute the f_j as specified with $f(j/n)$
Then we use those function value to compute the Fast Fourier Transform using the recursive formula:

$$\mathcal{F}[f_0, f_n, f_1, f_{n+1}, \ldots, f_{n-1}, f_{2n-1}]_k = \mathcal{F}[f_0, \ldots, f_{n-1}]_k + e^{-ik2\pi/n}\mathcal{F}[f_n, \ldots, f_{2n-1}]_k$$
$$\mathcal{F}[f_0, f_n, f_1, f_{n+1}, \ldots, f_{n-1}, f_{2n-1}]_{n+k} = \mathcal{F}[f_0, \ldots, f_{n-1}]_k - e^{-ik2\pi/n}\mathcal{F}[f_n, \ldots, f_{2n-1}]_k$$

Then we simply scale the result by multiplying with $\frac{1}{\sqrt{n}}$.
You can check our results in the file `output.txt`, if we plot them we get:


Real part
Imaginary part
40
30
20
10
0
−10
−20
−30
0
50
100
150
200
250