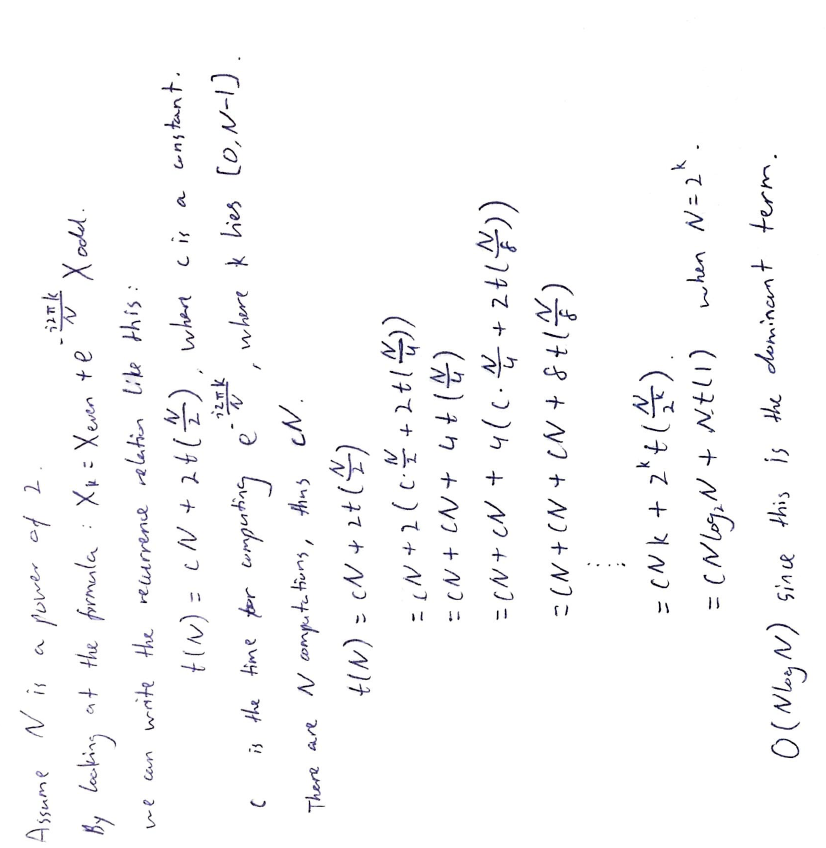
Analysis

The runtime of the naïve DFT for 1D is O[N2] because for every element in the 1D array we must do the summation. Each summation has N components, N being the length of the array. There are N elements in the array. In total, we have to do N2 computations. Thus the complexity is O[N2].

The derivation for the FFT is shown below.





The complexities for both 2D DFT and 2D FFT algorithms are also easy to compute.

For the 2D FFT algorithm, we can think of it as an 1D array containing N\*N elements. As mentioned earlier, the complexity of 1D FFT is O[N\*log N]. Therefore, all we need to do is to replace N, as the number of elements in 1D array, to N2, as the number of elements in 2D array. Thus, the complexity becomes

We simplify a little by moving the exponent inside the log function outside and it becomes

which shows the time complexity of the 2D FFT is O[N2log N].

For the 2DDFT algorithm, we apply the same technique as above. Recall the complexity of 1D DFT is O[N2]. We replace N to N2, as it is now an array with N2 elements. Therefore, the time complexity of the 2D DFT is O[N4].

The algorithm is a classic “divide and conquer” idea. Therefore, we write the algorithm by imitating “merge sort” algorithm. First we check if the size of the input array is a power of 2. If not, we return an error. Next we define our base case. By doing some experiments we set the base case to be 32. That is, if the array size is less or equal to 32, we use DFT to calculate directly. The recursive part is quite similar as the “merge sort”. We split the array into even and odd, which can be easily done using the slicing operation in python. The merging operation is done using an internal function called “concatenate” in numpy.

The Two-Dimensional Fourier Transform is a little tricky than One-Dimensional. We studied the formula for 2D FFT and realized that 2D FFT can just be done by using nested function. That is, functions like this form .



Figure 1

Figure 1 is the formula for 2D FFT. We can calculate the inner summation, which is 1D Fourier Transform with input being a 2D array. 2D array can be thought as an 1D array with N2 elements. Therefore, the inner summation can be easily achieved by using 1D FFT, with a little modification for the input from 1D array to 2D array. We then calculate the outer summation, which can be done using nested function. The FFT method outputs a 2D array, which becomes an input of another FFT method. This is how we achieve 2D Fourier Transform. The idea of inverse 2D Fourier Transform is the same except that we now need to divide N2 instead of N.

There are two inputs, both of which should be checked for validity. The program will print an appropriate exception and terminate when there is an error, such as wrong image location, invalid mode number, etc.