1) Calculate CCT

|  |  |  |  |
| --- | --- | --- | --- |
| C |  |  |  |
| 0.5 | 0.5 | 0.5 | 0.5 |
| 0.5 | 0.5 | -0.5 | -0.5 |
| 0.5 | -0.5 | -0.5 | 0.5 |
| 0.5 | -0.5 | 0.5 | -0.5 |
|  |  |  |  |
| C(T) |  |  |  |
| 0.5 | 0.5 | 0.5 | 0.5 |
| 0.5 | 0.5 | -0.5 | -0.5 |
| 0.5 | -0.5 | -0.5 | 0.5 |
| 0.5 | -0.5 | 0.5 | -0.5 |
|  |  |  |  |
| C\*C(T) |  |  |  |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |

행렬 C와 를 곱한 결과, 단위행렬이 나오는 것을 알 수 있었다.

2) Change the rows of C, and calculate CCT

|  |  |  |  |
| --- | --- | --- | --- |
| C |  |  |  |
| 0.5 | 0.5 | -0.5 | -0.5 |
| 0.5 | -0.5 | 0.5 | -0.5 |
| 0.5 | 0.5 | 0.5 | 0.5 |
| 0.5 | -0.5 | -0.5 | 0.5 |
|  |  |  |  |
| C(T) |  |  |  |
| 0.5 | 0.5 | 0.5 | 0.5 |
| 0.5 | 0.5 | -0.5 | -0.5 |
| 0.5 | -0.5 | -0.5 | 0.5 |
| 0.5 | -0.5 | 0.5 | -0.5 |
|  |  |  |  |
| C\*C(T) |  |  |  |
| 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |

먼저 C행렬 을 으로 바꾸었다.

이때 함수는 C행렬의 각 열들을 이루고 있는 행렬 4개를 입력으로 받고, 이들을 입력하는

순서에 따라 열의 순서가 바뀌는 형식으로 정의하였다.

그리고 기존 1번의 행렬과 곱한 결과를 보았을 때, 각 행에서 1의 위치가 우리가 바꾸어

준 행렬번호와 일치하는 것을 알 수 있었다.

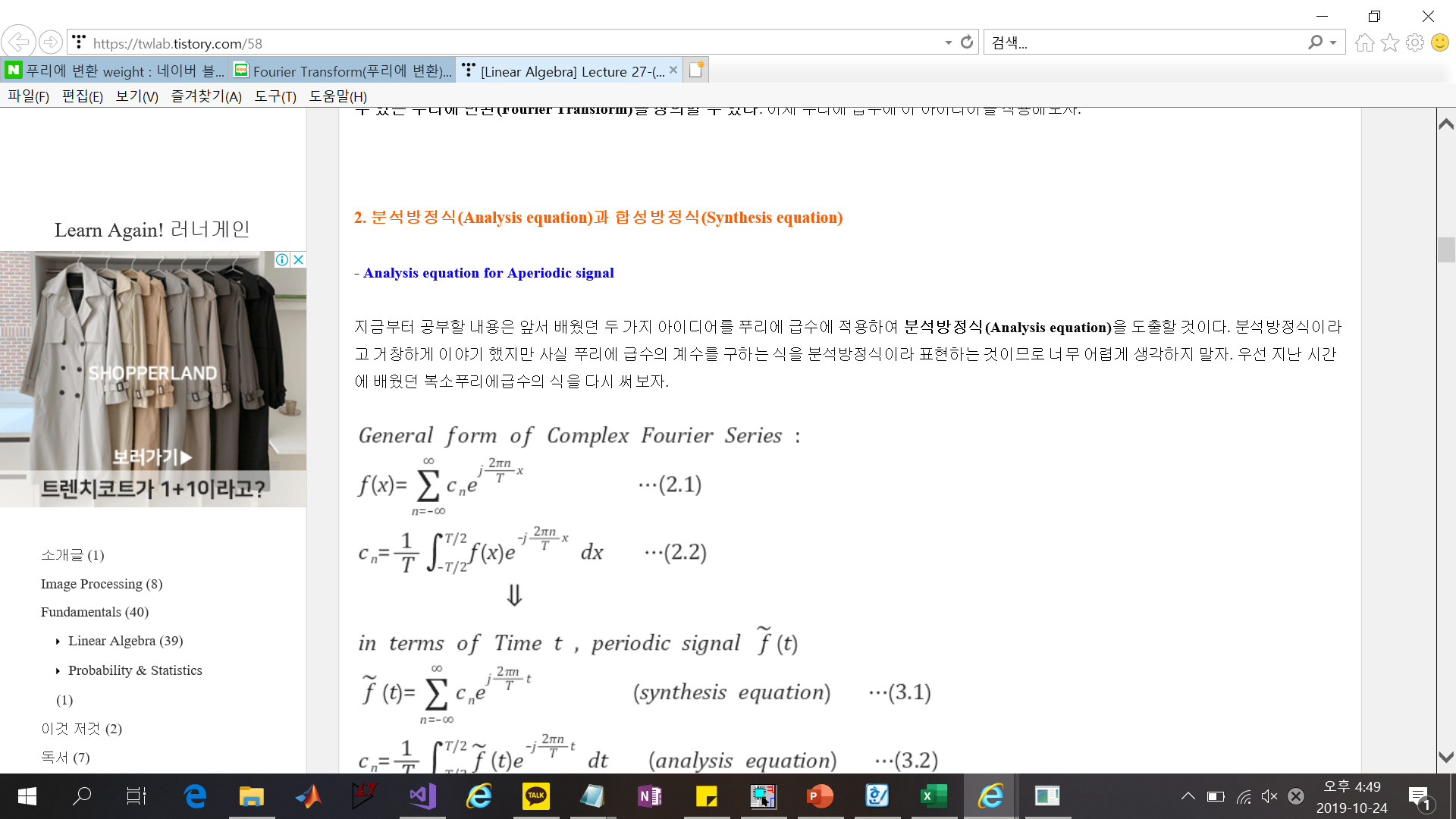
Assignment 2

1) Calculate for a1, a2, a3

|  |  |  |  |
| --- | --- | --- | --- |
| Ca = A 구하기 | a = {100,120,-4,2} | a = {100,20,120,32} | a = {10,120,108,8} |
| A[0] | 109 | 136 | 123 |
| A[1] | 111 | -16 | 7 |
| A[2] | -7 | -4 | -105 |
| A[3] | -13 | 84 | -5 |

a 행렬을 원신호, C를 곱하는 것을 Fourier Transform과 같은 변환이라고 생각하여, 변환을 적용한 A 행렬을 구해보았다.

Fourier Transform 은 다음과 같은 식으로 나타나게 된다.



이때 우리는 weight라고 부르는 을 통해 원하는 특정 주파수의 신호를 증폭시키거나 감쇠시킬 수 있는데, a행렬을 원신호라고 했을 때 C라는 특정 계수를 곱하는 것이 Fourier Transform의 과정을 나타낸다고 생각하였다.

2) Calculate to get back a1, a2, a3

|  |  |  |  |
| --- | --- | --- | --- |
| A = a 구하기 | a = {100,120,-4,2} | a = {100,20,120,32} | a = {10,120,108,8} |
| a[0] | 100 | 100 | 10 |
| a[1] | 120 | 20 | 120 |
| a[2] | -4 | 120 | 108 |
| a[3] | 2 | 32 | 8 |

를 계산 했을 때, 원래의 a 행렬과 같은 행렬이 결과로 도출되는 것을 확인하였다.

C와 C의 전치행렬 를 곱했을 때 I 행렬이 나오므로 orthogonal하고 이에 따라,

= 이 되는 것을 확인할 수 있다.

이 때 이러한 과정은 위 1번에서 했던 과정의 역연산으로, 말하자면 Fourier Transform을 한 신호에서 다시 원신호를 복구해내는 Inverse Fourier Transform 과정이라고 볼 수 있다.

3) Set two smallest out of to be zeros and do 2 again.

|  |  |  |  |
| --- | --- | --- | --- |
| A’ = a’구하기 | A’ = {109,111,0,0} | A’ = {136,0,0,84} | A’ = {123,0,-105,0} |
| a’[0] | 110 | 110 | 9 |
| a[‘1] | 110 | 26 | 114 |
| a[2] | -1 | 110 | 114 |
| a[3] | -1 | 26 | 9 |

에서 가장 작은 두 성분을 0으로 만들고 를 곱해본 결과 위와 같은 값을 얻을 수 있었다.

신호의 성분을 이용하여 의 값을 구해보면

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | a = {100,120,  -4,2} | a = {100,20,120,32} | a = {10,120,108,8} | a’ = {110,110,  -1,-1} | a’= {110,26,110,26} | a’ = {65,65,58,58} |
|  | 14404 | 1424 | 14464 | 12101 | 1352 | 13077 |

각각 같은 색으로 매칭되는 행렬끼리의 오차율을 계산해본 결과

빨간색 첫 번째의 경우 16%

파란색 두 번째의 경우 5.1%

녹색 세 번째의 경우 9.6% 오차를 보였다.

음악을 일정 용량에 압축하여 저장한 MP3 파일의 예시처럼 신호에서 매우 작은 성분을 생략하여도 원 신호와 유사하게 복원시킬 수 있다는 것을 알 수 있다.

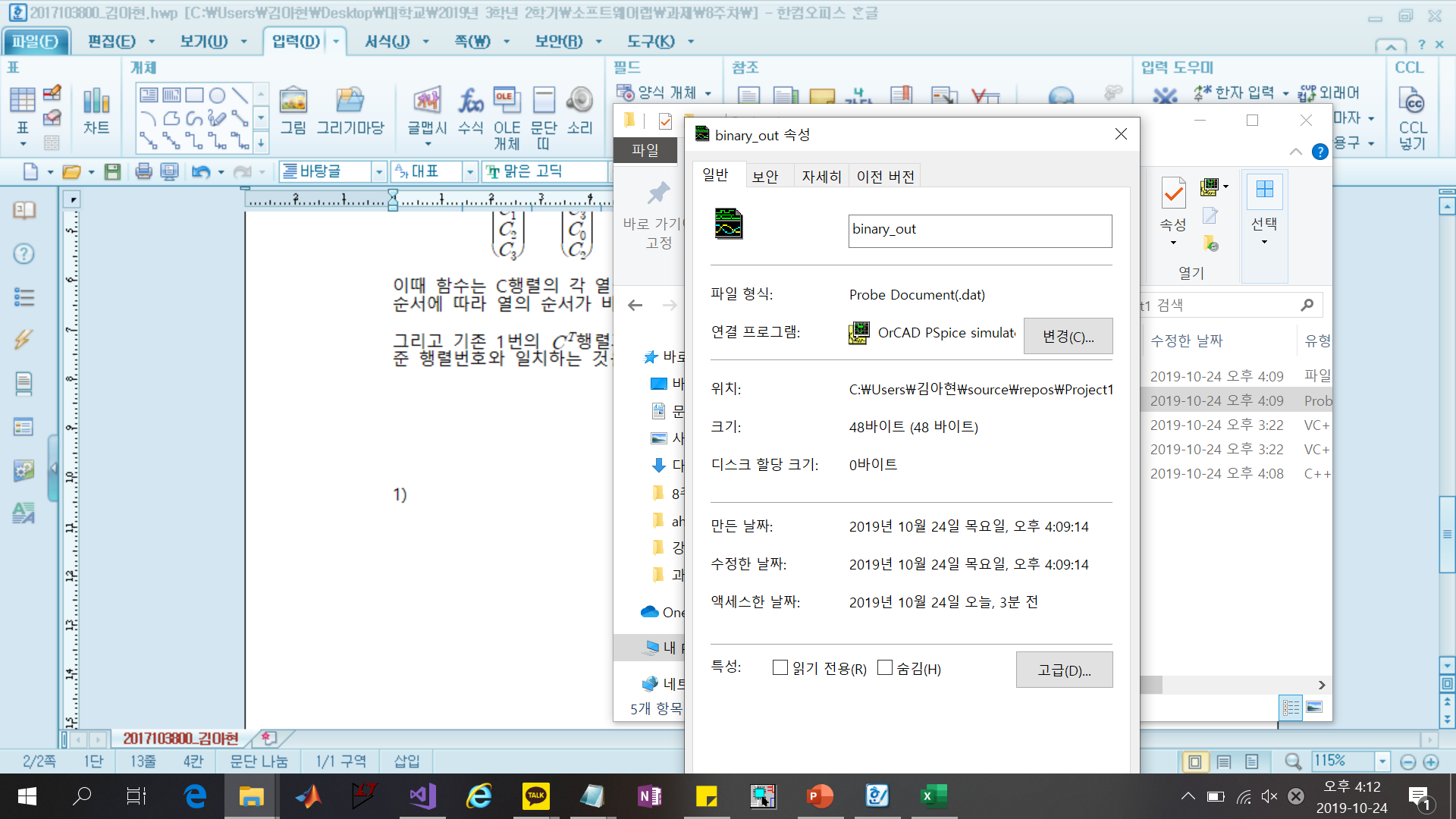
Assignment 3

1) Write 12 values of 3 sets of on a binary file, and check the file size.

Binary 파일로 A 행렬값을 출력한 후 다시 읽어와 표로 만든 결과, A1,A2,A3은

|  |  |  |  |
| --- | --- | --- | --- |
| A1 |  |  |  |
| 109 | 111 | -7 | -13 |
| A2 |  |  |  |
| 136 | -16 | -4 | 84 |
| A3 |  |  |  |
| 123 | 7 | -105 | -5 |

다음과 같이 나타났다. 또 총 12개의 float인자를 write했으므로, 파일 용량은 12\*4=48byte여야 하고, 확인해본 결과 이와 일치하였다.



2) Read the binary file and get back by using .

|  |  |  |  |
| --- | --- | --- | --- |
| a1 |  |  |  |
| 100 | 120 | -4 | 2 |
| a2 |  |  |  |
| 100 | 20 | 120 | 32 |
| a3 |  |  |  |
| 10 | 120 | 108 | 8 |

1)에서 프로그래밍을 하여 저장했던 Binary file을 다시 읽어온 후 , 이 A행렬을 와 다시 곱하여 a행렬을 구한 뒤 출력하였다. 그리고 기존의 a행렬 값과 비교해본 결과 일치하는 것을 확인할 수 있었다.