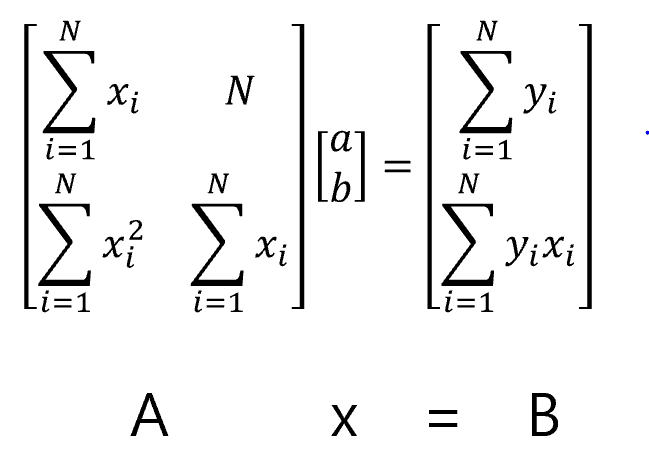
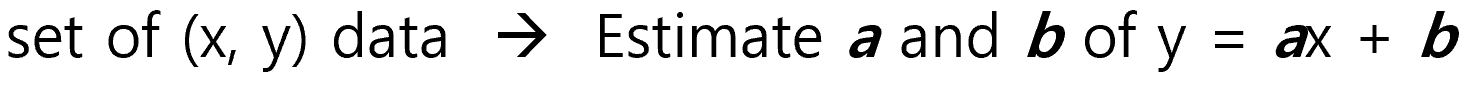
**Least Square Method**

1. **data2.txt를 2차식으로**

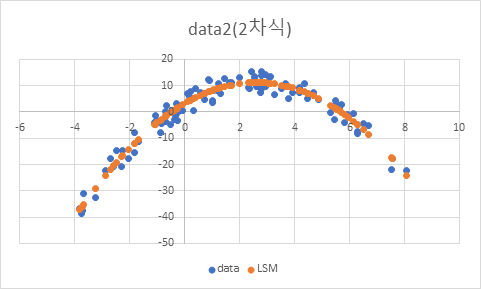
Klas에 제공되어 있는 matrixLSM2.cpp 파일을 참조하여, eigen 라이브러리를 이용하여 구현하도록 코드를 변경하였다.



1차 방정식의 경우 위와 같은 방식으로 LSM을 구현하는데, A행렬의 역행렬을 양변에 곱하면 a와 b값을 구할 수 있다. 이를 이용하여 eigen 라이브러리의 inverse() 함수를 사용해 역행렬을 양변에 곱해주면, 고차식의 경우도 문제없이 방정식의 계수들을 구할 수 있을 것이라고 생각 했다.

코드를 작성한 후, 실행시켜본 결과 아래와 같은 식을 얻을 수 있었다.



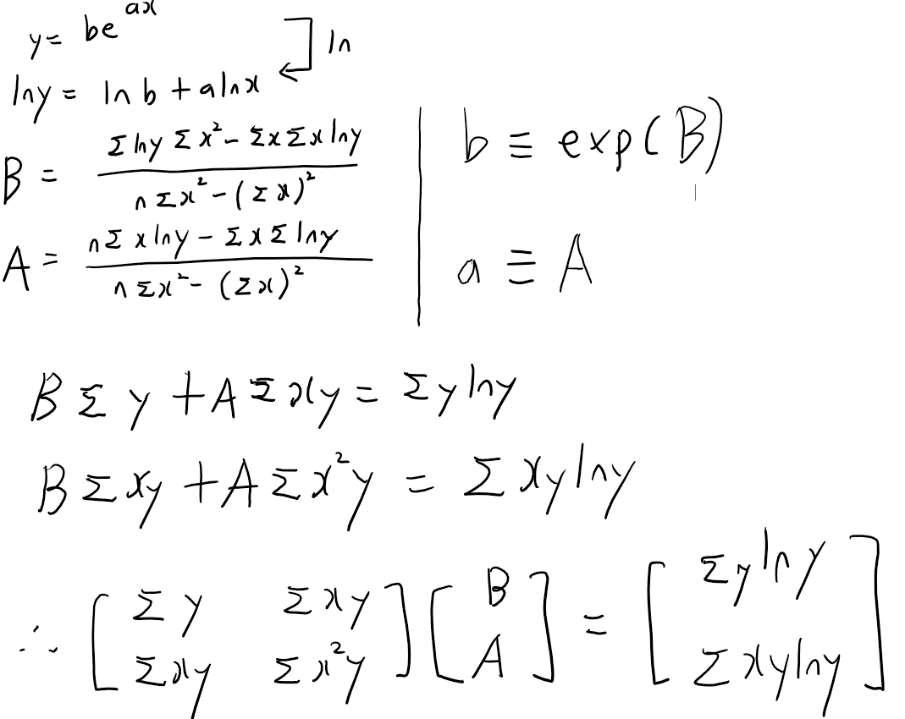
excel에 주어진 data값을 입력하고 각 x값을 위의 식에 대입시켰을 때의 y값을 입력한 후 그래프를 그려보니

위와 같은 그래프를 그릴 수 있었고, data값들과 LSM을 통해 얻은 값들이 비슷한 2차 함수곡선을 그리는 것으로 보아 eigen 라이브러리를 이용하여 LSM을 제대로 구현했음을 확인 할 수 있다. 기존의 matrixLSM2.cpp 파일과도 같은 결과임을 확인하였다.

**2. datae를 지수함수로**



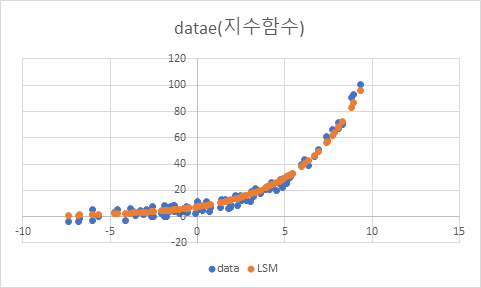
이 식의 a와 b를 구하기 위해,

위와 같은 과정을 통해 a,b 값을 도출하고자 하였다.

위의 과정에서 얻은 행렬들을 코드화 시켜 실행시켜본 결과,

의 식을 얻을 수 있었고,

1번과 마찬가지로 excel에 주어진 data값을 입력하고 각 x값을 위의 식에 대입시켰을 때의 y값을 입력한 후 그래프를 그려보니

위와 같은 그래프를 그릴 수 있었다.

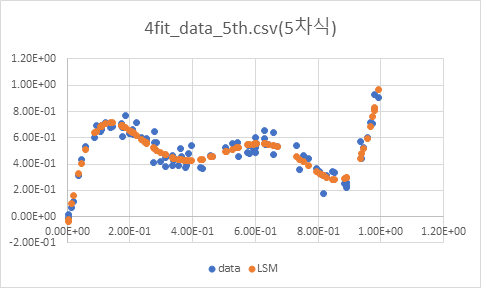
data값들과 LSM을 통해 얻은 값들이 비슷한 지수 함수 곡선을 그리는 것으로 보아 eigen 라이브러리를 이용하여 LSM을 제대로 구현했음을 확인 할 수 있다.

**3. 4fit\_data\_5th.csv를 5차식으로**

csv파일의 값들을 txt파일로 변환한 후, 1번의 2차식에 대한 코드를 5차식 코드로 확장시켜서 실행시켜본 결과 다음과 같은 식을 얻을 수 있었다.



마찬가지로 excel에 주어진 data값을 입력하고 각 x값을 위의 식에 대입시켰을 때의 y값을 입력한 후 그래프를 그려보니



위와 같은 그래프를 그릴 수 있었고 data값들과 LSM을 통해 얻은 값들이 비슷한 5차 함수 곡선을 그리는 것으로 보아 eigen 라이브러리를 이용하여 LSM을 제대로 구현했음을 확인 할 수 있다.

**4. 삼각함수(sin)로 시도**

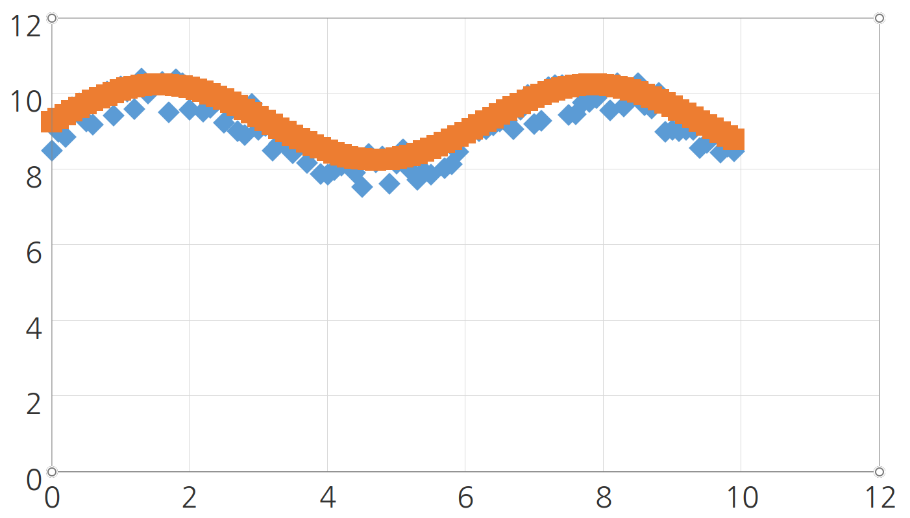
기존 과제 이외에 다른 종류의 함수로 LSM을 시도해보기 위해 삼각함수를 선택하였다.

sin함수의 데이터를 생산하기 위해서 a\*sin(x)+b = y에서 a와 b를 0~10사이의 random한 float값을 집어넣고 0~10 범위에서 0.1 interval로 각각의 y값에 -0.5~0.5사이의 random한 노이즈를 주어 100개의 data를 생산했다.

이 데이터를 가지고 LSM방식으로 코드를 실행시켜본 결과 다음과 같은 식을 얻을 수 있었다.



excel에 주어진 data값을 입력하고 각 x값을 위의 식에 대입시켰을 때의 y값을 입력한 후 그래프를 그려보니



위와 같은 그래프를 그릴 수 있었고, LSM을 통해 삼각함수의 계수 또한 도출할 수 있다는 것을 확인하였다.

**5. y=ax+b의 표준오차 구해보기**

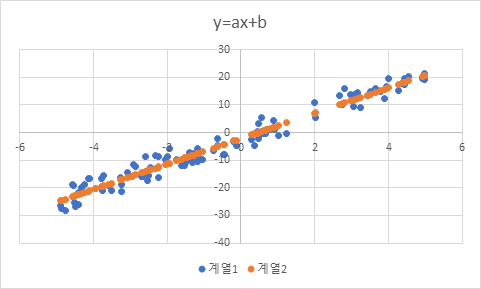
LSM에 관한 정보를 찾던 중, y=ax+b에서 a와 b에 대한 표준 오차를 구하는 방법을 찾게 되었다.

표준 오차란 모집단에 있는 대상들을 일부 추출한 표본집단에서 표본평균들을 구할 때 평균 값이 바뀌는 정도를 의미한다.

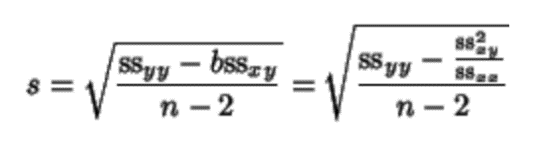
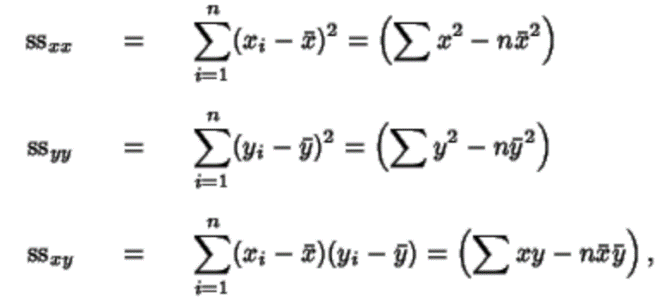
이를 시도해보기 위해, y= ax+b에서 a와 b를 –5~5사이의 랜덤한 float 값으로 설정하였고, x의 값을 –5~5사이의 랜덤한 float 값 100가지로 설정하고 이것을 y=ax+b에 대입하여 얻은 값에 noise를 더하여 y값을 100가지 설정하였다.

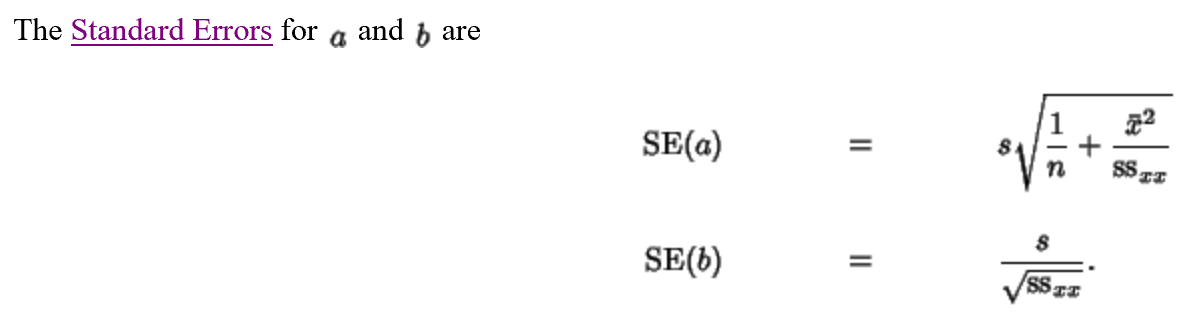
이후에 앞서 시도했던 LSM 방식을 이용하여 똑같이 실행시켜보니

**라는 식을 얻을 수 있었고 이를 그래프를 그려본 결과,**



위와 같이 성공적으로 1차 함수 곡선을 도출할 수 있었다.

이제 직접 생성한 100개의 x,y데이터를 통해 a와 b에 대한 표준 오차를 구한다.



x의 표준편차, y의 표준편차, x와 y의 covariance를 구하고, 이것들을 공식에 대입하는 코드를 작성하여, SE(a), SE(b)를 구해본 결과

**SE(a) = 0.052104**

**SE(b) = 0.0907625**

를 얻을 수 있었다.

이 표준오차를 줄이기 위해서는 표준편차를 크게 하거나, 표본의 크기를 키워야 한다.

두 가지 방법 중에서 표본의 크기를 키워보고자 N = 100에서 1000으로 10배 증가시키고 표준 오차를 구했더니

**SE(a) = 0.0105878**

**SE(b) = 0.0315716**

으로 감소된 것을 확인할 수 있었다.

N = 10000일 경우,

**SE(a) = 0.00283551**

**SE(b) = 0.0100057**

으로 더욱 감소한 것을 확인하였다.