중간 점검 보고서

2차원 횡단면의 3차원 복원

3조	
학번	이름
2014010786	김하은
2014010788	김홍관
2017010694	박나영
2018010702	박세영
2018010705	신영민

목차

Ⅰ. 수행개요

1. 목적과 필요성

현대에는 2D에서 3D로의 복원기술이 많이 사용되고 있다. 3D프린터, 의료수학뿐만 아니라 건축설계에서도 이 개념이 많이 쓰이고 최근에는 서울시가 자율주행 시스템을 위해 서울시 전체를 3D로 구현하고 있다. 이를 위해 수학은 없어서는 안 될 중요한 학문이다.

위의 복원기술을 위해서는 수학 개념 중 수치해석 분야가 많이 응용된다. 우리는 수치해석 분야 중 보간법과 최소자승법을 사용해 횡단면을 이용한 3차원 복원이라는 프로젝트를 진행하기로 하였다.

이 프로젝트에서는 3D 구현을 위한 수학적 개념을 학습하는 것이 목적이다. 최종적으로는 컴퓨터 언어를 이용해 구현하는 것이 목표이다. 이 과제를 통해 우리는 다양한 수치 해법들을 배우고 지금까지 배운 다양한 수학 전공을 실생활에 응용할 수 있을 것이다.

2. 목표

1) 2차원 횡단면을 이용해 3차원 물체를 재건하기 위한 수학적 내용과 컴퓨터 언어를 학습한다.

2) 보간법, 최소자승법을 이해하고 이를 학습한다.

3) 위 내용을 사용하여 MATLAB에서 모델을 구현한다.

4) 최종 모델을 구현하고 다양한 도형의 단면을 사용하여 테스트한다.

Ⅱ. 수행내용

1. 선행 학습

 - 2차원 횡단면을 이용해 3차원 물체를 재건하기 위해서는 수학적 내용과 컴퓨터 언어에 관한 선행 학습이 필요하다.

 - 학습 방식: 각자 맡은 내용을 공부한 후 한주에 한 명씩 돌아가면서 발표형식으로 설명해주며 피드백을 하며 학습하고, 발표에 사용했던 피피티를 사용하여 보고서를 작성한다.

1) 수학적 내용

 - 보간법: 수치해석학의 수학 분야에서 보간법은 알려진 데이터 지점의 고립점 내에서 새로운 데이터 지점을 구성하는 방식이다. 보간법에는 라그랑지 보간법, 뉴턴 보간법, 스플라인 보간법이 있다. 우리는 이 중에서 스플라인 보간법에 집중한다. 스플라인 보간법은 전체구간을 소구간별로 나누어 저차수의 다항식으로 매끄러운 함수를 구하는 방법이다.

 - 최소자승법: 해 방정식을 근사적으로 구하는 방법으로, 근사적으로 구하려는 해와 실제 해의 오차의 제곱의 합이 최소가 되는 해를 구하는 방법이다. 이 방법은 값을 정확하게 측정할 수 없는 경우에 유용하게 사용될 수 있으며, 특히 계의 방정식이 어떤 형태인지를 알고 있을 때 방정식의 상수값들을 추청하는 데에 사용된다.

2) 컴퓨터 언어

 - MATLAB : MATLAB은 수식계산에 탁월한 기능과 간편성을 제공하는 컴퓨터 언어이다. 우리가 사용할 선형대수, 수치계산법 등 수학에서 풀기 어려운 복잡한 문제 풀이에 도움이 된다.

2. 재건 과정

- 여러 횡단면의 경계를 점으로 나눈다. 이때 각 면에 대한 점의 개수는 동일하게 한다. 각 면에 대한 점들에 대해 점들끼리 매치를 시킨다. 매치 된 점들을 사용해 보간법과 최소자승법을 사용하여 횡단면 사이 값을 예측한다. 횡단면 사이 값을 예측 후 이를 매트랩을 이용해 그래픽으로 나타낸다.

3. 오차 확인

-오차 확인: 다양한 도형의 단면을 재건하기 위해 보간법과 최소자승법을 이용해 보았다. 이 둘을 사용한 결과에 대해 원래의 도형과 비교해 오차를 구하고 어떤 방법이 더 오차가 작게 나오는지 효율이 좋은가에 대해 이야기 해본다.

Ⅲ. 수행일정

일시	내용
1주차	팀 구성 및 주제 설정
2주차	계획안 작성 및 발표
3주차 (세영)	매트랩 기초 학습
4주차 (나영)	수치 해석학 학습 - 라그랑지 보간법 - 매트랩 실습
5주차 (하은)	수치 해석학 학습(2) - 스플라인 보간법 - 매트랩 실습
6주차 (영민)	수치 해석학 학습(3) - 스플라인 보간법(2) - 매트랩 실습
7주차 (홍관)	수치 해석학 학습(4) - 최소자승법 - 매트랩 실습
8주차	중간 보고서 작성 및 발표
9주차 (세영)	수치해석학 학습(5) - 최소자승법(2) - 매트랩 실습
10주차 (나영)	보간법을 이용한 3차원 복원
11주차 (하은)	최소자승법을 이용한 3차원 복원
12주차 (영민)	예제에 적용하기
13주차 (홍관)	다양한 도형에 적용 및 오차 분석
14주차	기말 보고서 작성 및 검토
15주차	기말 보고서 발표

Ⅳ. 활동내용

2주차 - 계획서 작성

1주차에서 정한 주제 “2차원 횡단면을 이용한 3차원 복원”에 대한 계획서를 작성하였고 교수님의 피드백을 받아 수정하고 작성을 끝냈다.

전체적인 이미지 및 내용 구상은 신영민 학생이, 작성 및 맞춤법 검사는 박세영, 박나영 학생이, 추가내용 삽입 및 내용 검토는 김홍관, 김하은 학생이 맡았다.

<그림 2-1 계획서>

종합설계 계획서

2차원 횡단면의 3차원 복원

3조	
학번	이름
2014010786	김하은
2014010788	김홍관
2017010694	박나영
2018010702	박세영
2018010705	신영민

목차

Ⅰ. 수행개요 ……………… 3p

Ⅱ. 수행내용 ……………… 4p

Ⅲ. 수행일정 ……………… 6p

Ⅳ. 참고자료 ……………… 7p

Ⅰ. 수행개요

1. 목적과 필요성

2. 목표

Ⅱ. 수행내용

1. 선행 학습

1) 수학적 내용

2) 컴퓨터 언어

2. 재건 과정

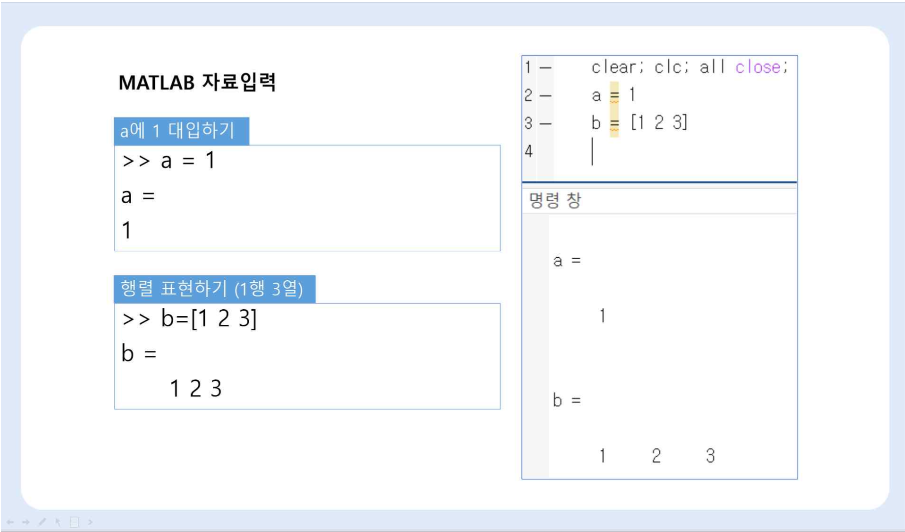
3. 오차 확인

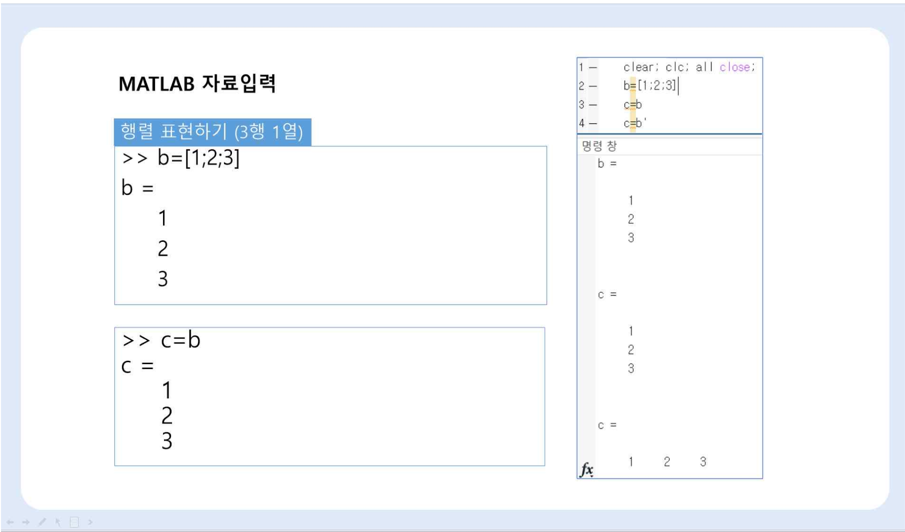
Ⅲ. 수행일정

일시	내용
1주차	팀 구성 및 주제 설정
2주차	계획안 작성 및 발표
3주차 (세영)	매트랩 기초 학습
4주차 (나영)	수치 해석학 학습
5주차 (하은)	수치 해석학 학습(2)
6주차 (영민)	수치 해석학 학습(3)
7주차 (홍관)	수치 해석학 학습(4)
8주차	중간 보고서 작성 및 발표
9주차 (세영)	수치해석학 학습(5)
10주차 (나영)	보간법을 이용한 3차원 복원
11주차 (하은)	최소자승법을 이용한 3차원 복원
12주차 (영민)	예제에 적용하기
13주차 (홍관)	다양한 도형에 적용 및 오차 분석
14주차	기말 보고서 작성 및 검토
15주차	기말 보고서 발표

3주차 - 매트랩 기초 학습

라그랑지 보간법, 스플라인 보간법, 최소자승법 학습과 3차원 복원을 매트랩으로 하기 위해 매트랩을 설치하고 간단한 자료입력 방법을 학습하였다.
Industrial Mathematics책에서 제 1장 MATLAB 기초를 참고하여 박세영 학생이 ppt 형식으로 정리해 발표하였고 매트랩으로 for ~ end 문, plot 문을 학생들과 실행해 보았다.





MATLAB 자료입력

행렬의 각 원소에 대해 각각 곱하기

```
>> a = [7 8 9];
>> a.*b
ans =
7 16 27
```

영행렬

```
>> zeros(1,3)
ans =
      0 0 0
```

```
clear; clc; all close;
a = [7 8 9];
b = [1 2 3];
a.*b

zeros(1,3)
```

명령 창

```
ans =

     7    16    27


ans =

     0     0     0
```

MATLAB 자료입력

min

```
>> min(1, 2)
ans =
       1
```

max

```
>> max(1, 2)
ans =
       2
```

```
clear; clc; all close;
min(1, 2)
max(1, 2)
```

명령 창

```
ans =

     1


ans =

     2
```

MATLAB 자료입력

sum

```
>> A=[1 2 3; 4 5 6; 7 8 9];
>> sum(A)
ans =
      12 15 18
```

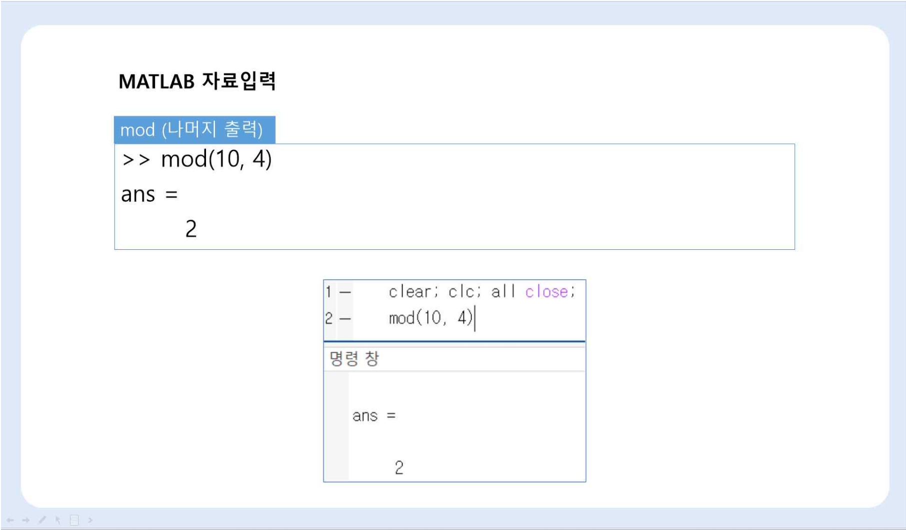
모든 원소의 합

```
>> sum(sum(A))
ans =
      45
```

```
clear; clc; all close;
A=[1 2 3; 4 5 6; 7 8 9];
sum(A)
 sum(sum(A))
```

명령 창

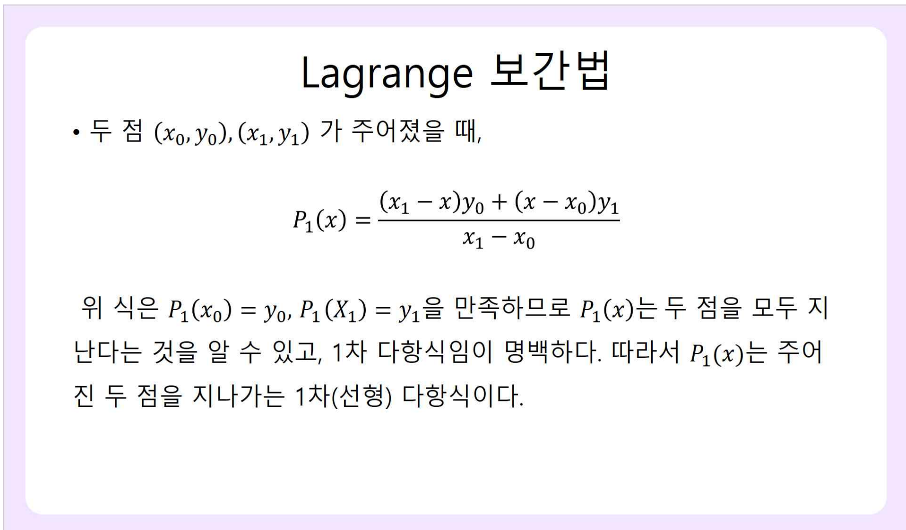
```
ans =

    12    15    18


ans =

    45
```

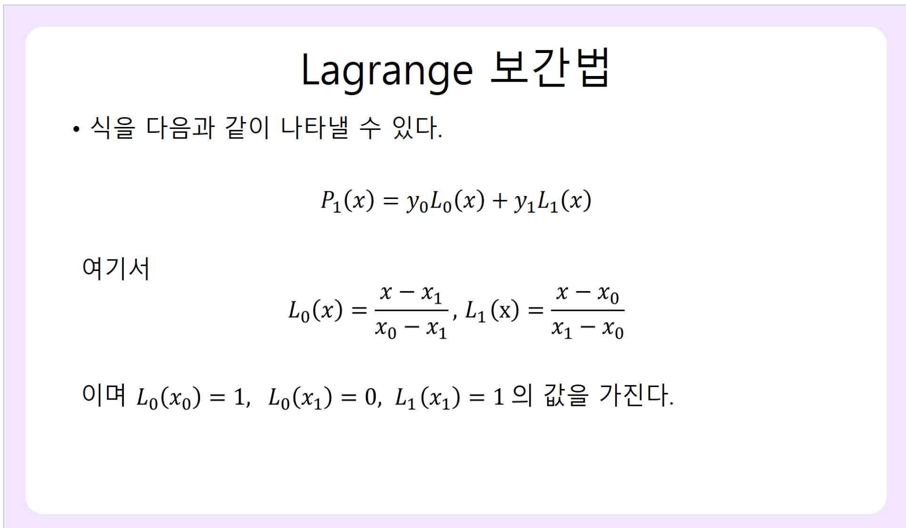


<그림 3-1, 박세영 학생의 MATLAB 기초학습에 대한 PPT설명 일부>

4주차 – Lagrange 보간법

여러 점 사이의 값을 다항식을 사용하여 추정하는 라그랑주 보간법에 대한 개념을 학습하고 1차 보간 다항식을 사용하여 두 점 사이의 값을 추정해 보고, ‘이공학을 위한 MATLAB 활용 수치해석’ 서적에 나와 있는 1차 보간 다항식에 대한 예제를 풀어보았다.





<그림 4-1. 1차 보간다항식에 대한 설명>

Lagrange 보간법

예제2.1

다음 자료를 사용 하여 두 점 (0.6, 1.822119), (0.7,2.013753)을 지나가는 1차 보간다항식을 구하고, $x = 0.63$에서 함수의 근사값을 계산해 보자.

점 (0.6, 1.822119)를 첫 점 (x_0, y_0)로,
점 (0.7, 2.013753)를 두 번째 점 (x_1, y_1)로 간주하고
$P_1(x)$에 대입하면

x	$y = f(x)$
0	1.0
0.4	1.491825
0.5	1.638721
0.6	1.822119
0.7	2.013753
0.8	2.225541
1	2.718282

Lagrange 보간법

$$P_1(x) = \frac{(x_1 - x)y_0 + (x - x_0)y_1}{x_1 - x_0}$$
$$= \frac{(0.7 - x)1.822119 + (x - 0.6)2.013753}{0.7 - 0.6}$$
$$= 1.916340x + 0.672315$$

이다. 그러면 $P_1(0.63) = 1.879609$
가 된다.

x	$y = f(x)$
0	1.0
0.4	1.491825
0.5	1.638721
0.6	1.822119
0.7	2.013753
0.8	2.225541
1	2.718282

오른쪽 표는 정확하게 $f(x) = e^x$ 이고 점 $x = 0.63$
의 실제 값은 $f(0.63) = e^{0.63} = 1.877611$ 이고 이 값에 근사한 값이 나온다.

<그림 4-2, 1차 보간다항식의 예제>

example_2_1.m

```
% 예제 2.1 간단한 실습

clear; clc; close all;

x=0.63;              % 근사할 값

x0=0.6; x1=0.7; y0=1.822119; y1=2.013753;        % 초기 값 설정

p=(x-x1)/(x0-x1)*y0+(x-x0)/(x1-x0)*y1;          % 보간 다항식

ap=p;
real=exp(0.63);      % 실제 데이터
error=abs(ap-real)  %에러
```

명령 창

MATLAB을 처음 사용한다면 시작하기를 참조하십시오.

```
error =

   0.001998620735657
```

Lagrange 보간법

- 2차 보간다항식

세 점 $(x_0, y_0), (x_1, y_1), (x_2, y_2)$ 를 지나는 2차 보간다항식의 형태는

$$P_2(x) = y_0 L_0(x) + y_1 L_1(x) + y_2 L_2(x)$$

이다. 여기서

$$L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}$$

$$L_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}$$

$$L_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}$$

으로 정의된다.

Lagrange 보간법

- 1차 식에서의 예시를 세 점 $(0.5, 1.648721), (0.6, 1.822119), (0.7, 2.013753)$ 세 점과 2차 보간다항식을 사용하여 구해보자.

$P_2(x)$를 정리하면 $P_2(x) = ax^2 + bx + c$의 형태로 나타낼 수 있다.

여기에 0.63을 대입하면 $P_2(0.63) = 1.877694$가 된다.

이것은 두 점을 이용한 1차 보간다항식에 의한 근사 값보다 실제 값에 더 가깝다는 것을 알 수 있다.

x	$y = f(x)$
0	1.0
0.4	1.491825
0.5	1.638721
0.6	1.822119
0.7	2.013753
0.8	2.225541
1	2.718282

Lagrange 보간법

<u>예제 2.2</u>

1차 식에서의 예시를 세 점 $(0.5, 1.648721), (0.6, 1.822119), (0.7, 2.013753)$ 세 점과 2차 보간다항식을 사용하여 구해보자.

x	$y = f(x)$
0	1.0
0.4	1.491825
0.5	1.638721
0.6	1.822119
0.7	2.013753
0.8	2.225541
1	2.718282

$$L_0(x) = \frac{(x-0.6)(x-0.7)}{(0.5-0.6)(0.5-0.7)} = 50(\mathrm{x}^2 - 1.3\mathrm{x} + 0.42)$$

$$L_1(x) = \frac{(x-0.5)(x-0.7)}{(0.6-0.5)(0.6-0.7)} = \text{-}100(x^2 - 1.2x + 0.35)$$

$$L_2(x) = \frac{(x-0.5)(x-0.6)}{(0.7-0.5)(0.7-0.6)} = 50(x^2 - 1.1x + 0.3)$$

$$P_2(x) = y_0L_0(x) + y_1L_1(x) + y_2L_2(x)$$

$$= 1.648721L_0(x) + 1.822119L_1(x) + 2.013753L_2(x)$$

가 된다.

example_2_1.m | example_2_3.m | prac_2_1.m | prac_2_2.m | example_2_2.m

```
% 예제 2.1 간단한 실습

clear; clc; close all;

x=0.63;              % 근사할 값

x0=0.5; x1=0.6; x2=0.7; y0=1.648721; y1=1.822119; y2=2.013753;       % 초기 값 설정


p1=(x-x1)/(x0-x1)*y0+(x-x0)/(x1-x0)*y1;            % 보간 다항식
p2=(x-x1)*(x-x2)/((x0-x1)*(x0-x2))*y0+(x-x0)*(x-x2)/((x1-x0)*(x1-x2))*y1+...
    (x-x0)*(x-x1)/((x2-x0)*(x2-x1))*y2;
error_1=abs(p1-exp(0.63))
error_2=abs(p2-exp(0.63))
```

명령 창

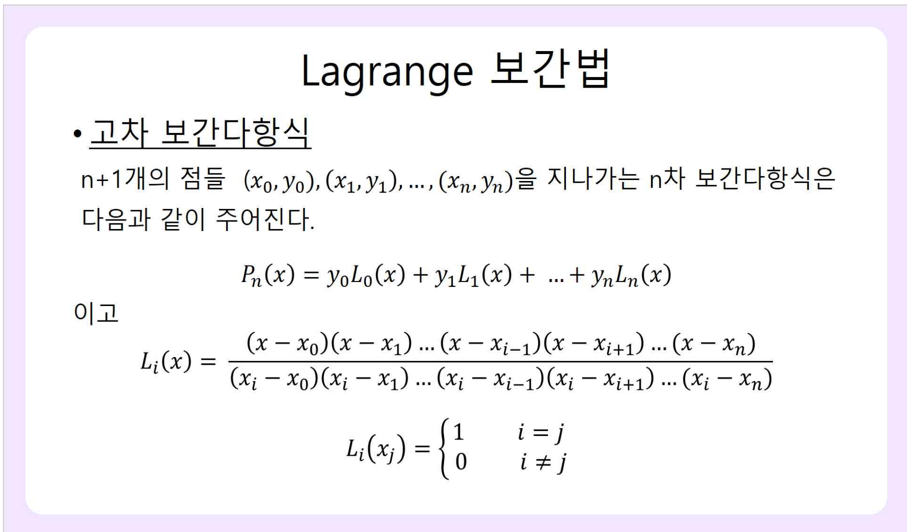
MATLAB을 처음 사용한다면 시작하기를 참조하십시오.

```
error_1 =

    0.0035


error_2 =

   8.3841e-05

fx >>
```

<그림 4-3. 2차 보간다항식 계산과 매트랩 실습>

세 점을 사용한 2차 보간 다항식을 사용하여 그 사이값을 추정하였고, 1차 보간 다항식 보다 오차가 더 적다는 사실을 확인하였다.

위 내용을 바탕으로 고차 보간 다항식을 세울 수 있으며, 이를 사용하여 오차를 점차 줄여나가며 추정 값을 더 정확하게 찾을 수 있다.



Lagrange 보간법

• <u>고차 보간다항식</u>

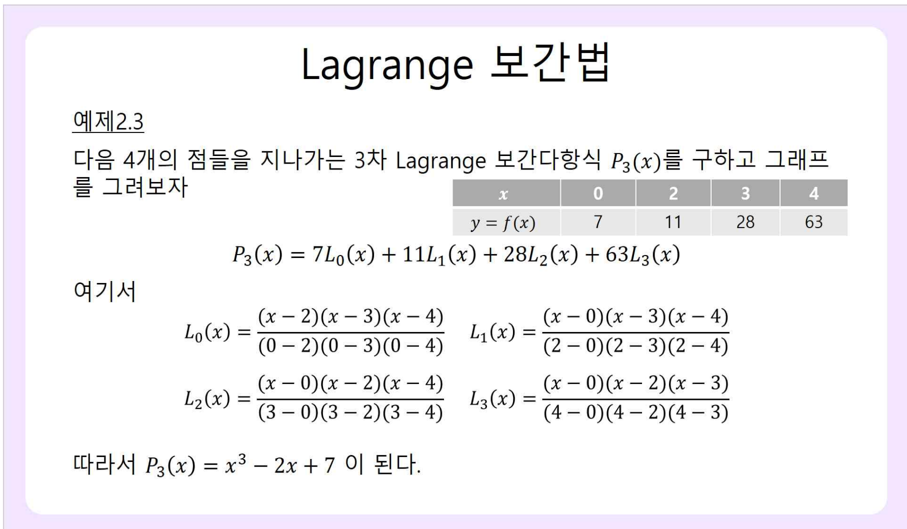
n+1개의 점들 $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$을 지나가는 n차 보간다항식은 다음과 같이 주어진다.

$$P_n(x) = y_0 L_0(x) + y_1 L_1(x) + \dots + y_n L_n(x)$$

이고

$$L_i(x) = \frac{(x - x_0)(x - x_1)\dots(x - x_{i-1})(x - x_{i+1})\dots(x - x_n)}{(x_i - x_0)(x_i - x_1)\dots(x_i - x_{i-1})(x_i - x_{i+1})\dots(x_i - x_n)}$$

$$L_i(x_j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$



Lagrange 보간법

<u>예제2.3</u>

다음 4개의 점들을 지나가는 3차 Lagrange 보간다항식 $P_3(x)$를 구하고 그래프를 그려보자

x	0	2	3	4
$y = f(x)$	7	11	28	63

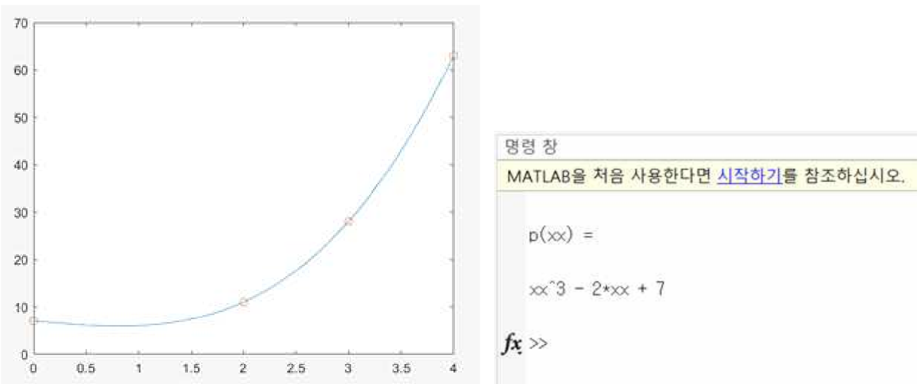
$$P_3(x) = 7L_0(x) + 11L_1(x) + 28L_2(x) + 63L_3(x)$$

여기서

$$L_0(x) = \frac{(x-2)(x-3)(x-4)}{(0-2)(0-3)(0-4)} \qquad L_1(x) = \frac{(x-0)(x-3)(x-4)}{(2-0)(2-3)(2-4)}$$

$$L_2(x) = \frac{(x-0)(x-2)(x-4)}{(3-0)(3-2)(3-4)} \qquad L_3(x) = \frac{(x-0)(x-2)(x-3)}{(4-0)(4-2)(4-3)}$$

따라서 $P_3(x) = x^3 - 2x + 7$ 이 된다.



<그림 4-4, 고차 다항식을 이용하여 매트랩으로 함수그래프 그리기>

연습문제와 같이 이산적인 자료들은 라그랑주 보간법을 사용하여 사이 값을 추정할 수 있다.

Lagrange 보간법

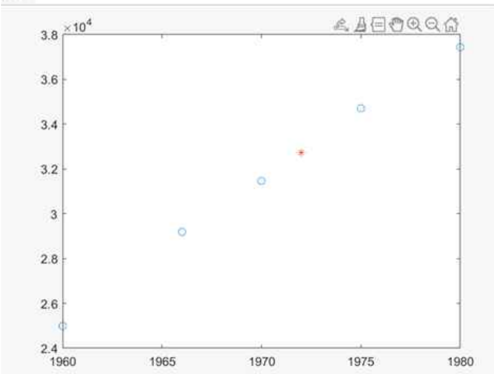
연습문제 2.1

대한민국 정부 수립 후 연도별로 조사한 총 인구 수는 아래와 같다.
4차 보간 다항식을 구하여 1972년도의 우리나라 인구를 추정해 보아라.

조사 년도	인구(단위:천명)
1944	16,565
1949	20,189
1955	21,526
1960	24,989
1966	29,193
1970	31,466
1975	34,707
1980	37,436
1985	40,448
1990	43,411
1995	44,609
2000	46,136
2005	47,279

example_2_1.m | example_2_3.m | prac_2_1.m | prac_2_2.m

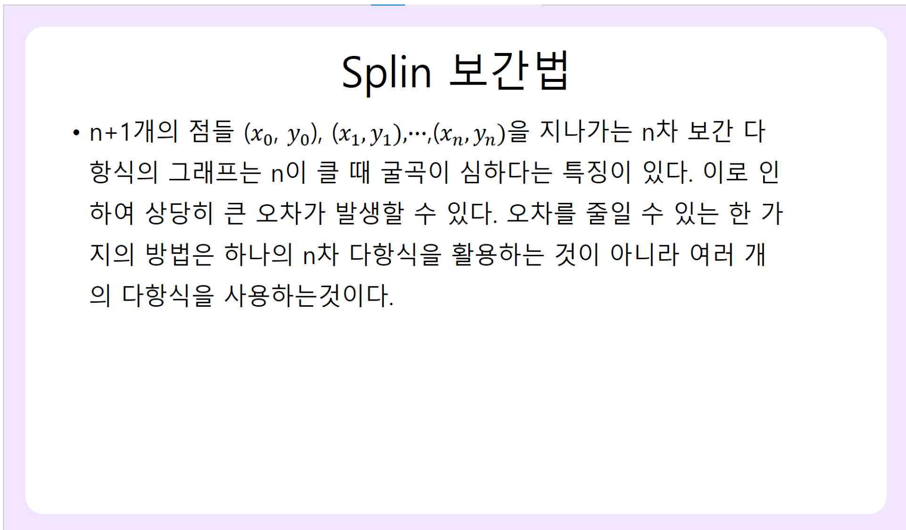
```
clear; clc; close all;

year=[1960 1966 1970 1975 1980];            % 년도
num=[24989 29193 31466 34707 37436];        % 인구 수

x=1972;     % 구하고 싶은 년도

l0=(x-year(2))*(x-year(3))*(x-year(4))*(x-year(5))...
/((year(1)-year(2))*(year(1)-year(3))*(year(1)-year(4))*(year(1)-year(5)));     % l0 구하기

l1=(x-year(1))*(x-year(3))*(x-year(4))*(x-year(5))...
    /((year(2)-year(1))*(year(2)-year(3))*(year(2)-year(4))*(year(2)-year(5)));     % l1 구하기

l2=(x-year(1))*(x-year(2))*(x-year(4))*(x-year(5))...
    /((year(3)-year(1))*(year(3)-year(2))*(year(3)-year(4))*(year(3)-year(5)));     % l2 구하기

l3=(x-year(1))*(x-year(2))*(x-year(3))*(x-year(5))...
    /((year(4)-year(1))*(year(4)-year(2))*(year(4)-year(3))*(year(4)-year(5)));     % l3 구하기

l4=(x-year(1))*(x-year(2))*(x-year(3))*(x-year(4))...
    /((year(5)-year(1))*(year(5)-year(2))*(year(5)-year(3))*(year(5)-year(4)));     % l4 구하기

p=l0*num(1)+l1*num(2)+l2*num(3)+l3*num(4)+l4*num(5);      % 1972년도 인구 수
```

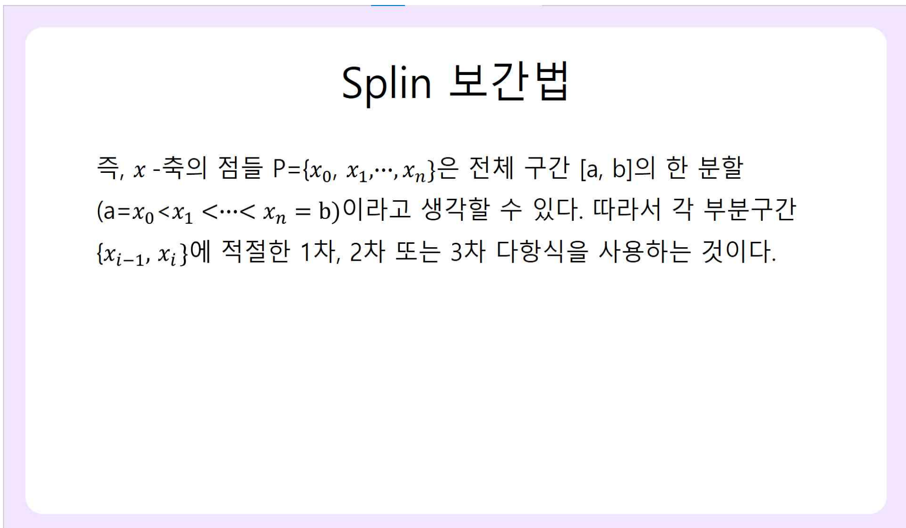


<그림 4-5, 매트랩을 이용하여 연습문제 풀기>

5주차 - 스플라인 보간법(1)

2차원 횡단면의 3차원 복원을 위해서 ‘이공학을 위한 MATLAP 활용 수치 해석’이라는 책을 참고하여 스플라인 보간법의 개념을 알아보고 식으로 나타내보았다. 그리고 S'(x)와 S”(x)가 연속인 3차 스플라인 보간함수인 S(x)를 자연경계조건을 사용하여 유도해보는 시간을 가졌다. 김하은 학생이 ppt로 정리하여 설명하는 방식으로 진행했다.





Splin 보간법

- 식을 다음과 같이 나타낼 수 있다.

$$S(x)=\begin{cases} S_0(x), & x_0 \le x \le x_1 \\ S_1(x), & x_1 \le x \le x_2 \\ \vdots & \vdots \\ S_{n-1}(x), & x_{n-1} \le x \le x_n \end{cases}$$

Splin 보간법

-만약 각 소구간에 1차 다항식을 사용하면 S(x)를 1차 스플라인 보간 함수라고 부른다. (S′(x)는 연속이 아니기 때문에 각 마디점이 꺾이는 상황이 일어난다.)

-각 소구간에 2차 다항식을 사용하면 S(x)를 2차 스플라인 보간 함수라고 한다. (1차보다는 매끄러운 함수가 생성 되지만 S″(x)는 연속이 되지 않는다.)

-위와 같은 상황 때문에 3차 스플라인 보간 함수를 사용한다.

-3차 스플라인 보간 함수 : $a\_i+bi(x-xi)+ci(x-xi)\text{^}2+di(x-xi)\text{^}3, (i=0,2,\dots,n-1)$

Splin 보간법

- 3차 스플라인 보간 함수 조건

Si(x)=ai+bi(x-xi)+ci(x-xi)^2+di(x-xi)^3, (i=0,2,...,n-1)

1. Si(xi)=yi, Si(xi+1)=yi+1
2. S(i-1)′(xi)=Si′(xi)
3. S(i-1)″(xi)=Si″(xi)
4. 자연경계 조건 : S0″(x0)=0, Sn-1″(xn)=0 (지금은 이 조건 사용)

+고정경계 조건 : S0′(x0)=y0′, Sn-1′(xn)=yn′

Splin 보간법

- 조건 1. Si(xi)=yi, $Si(x_{i+1})=y_{i+1}$

Si(x)=ai+bi(x-xi)+ci(x-xi)^2+di(x-xi)^3
Si(xi)=ai=yi ①
Si(xi+1)=ai+bi(xi+1-xi)+ci(xi+1-xi)^2+di(xi+1-xi)^3
=ai+bi*hi+ci*hi^2+di*hi^3=yi+1
yi+1-yi= bi*hi+ci*hi^2+di*hi^3 ②

Splin 보간법

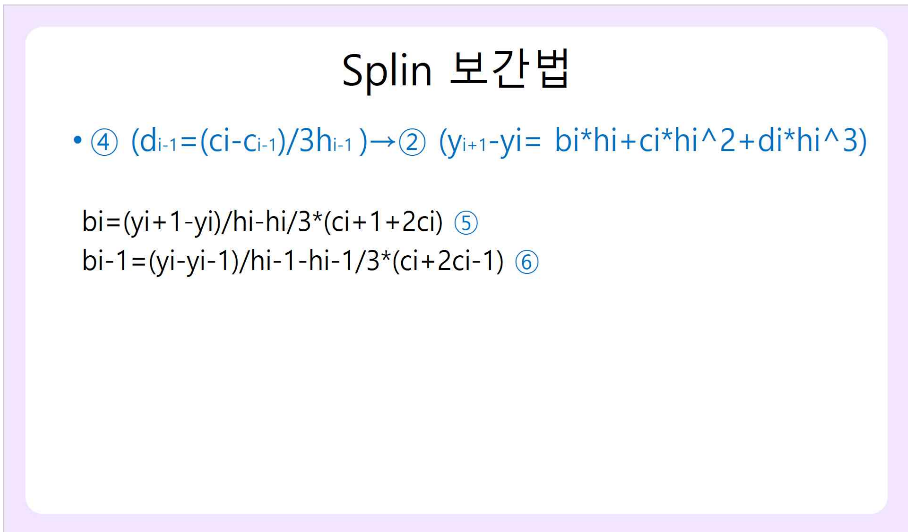
- 조건 2. $S_{(i-1)}$'(xi)=Si'(xi)

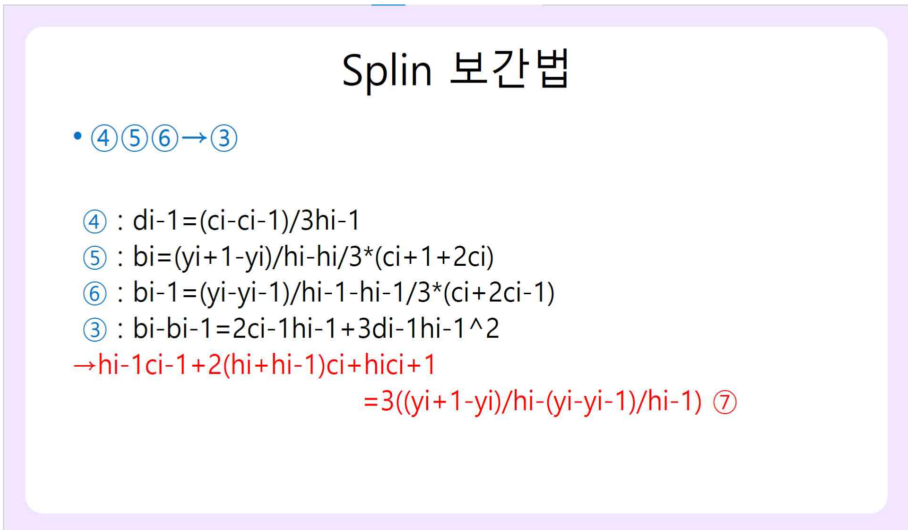
Si'(x)=bi+2ci(x-xi)+3di(x-xi)^2
Si'(xi)=bi
Si-1'(xi)=bi-1+2ci-1hi-1+3di-1hi-1^2
bi-bi-1=2ci-1hi-1+3di-1hi-1^2 ③

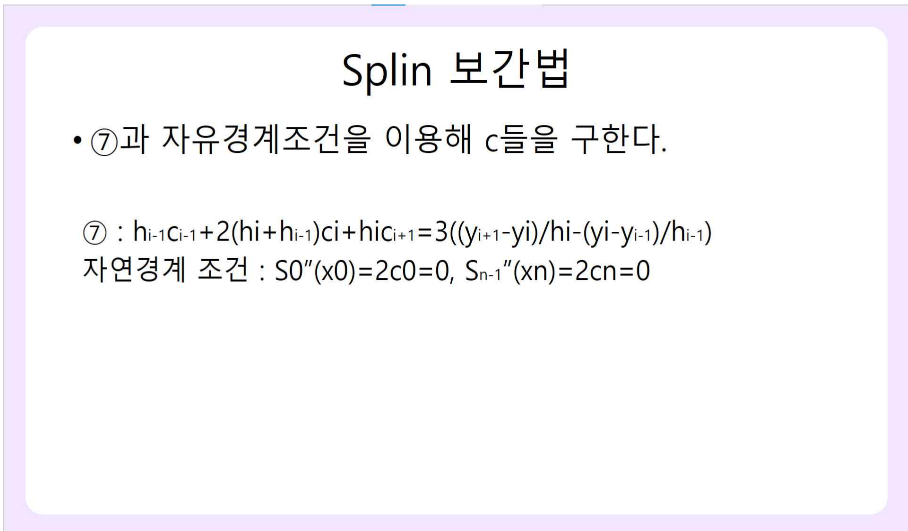
Splin 보간법

- 조건 3. S_{i-1}"(xi)=Si"(xi)

Si"(x)=2ci+6di(x-xi)
Si"(xi)=2ci
Si-1"(xi)=2ci-1+6di-1hi-1
di-1=(ci-ci-1)/3hi-1 ④



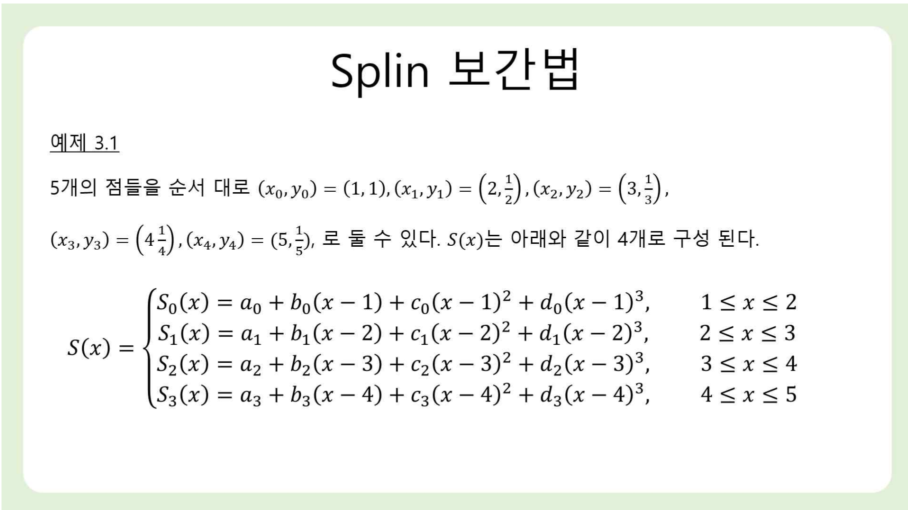




<그림 5, 김하은 학생의 Splin 보간법 실습에 대한 PPT설명 일부>

6주차 - 스플라인 보간법(2)

4월 5일과 4월 7일에 걸쳐서 2차원 횡단면의 3차원 복원을 위해서 3월 29일, 3월 31일 학습했던 스플라인 보간법의 이론을 바탕으로 '이공학을 위한 MATLAB 활용 수치해석'에 나와 있는 예제와 연습문제를 풀어보았다. 또한 복원에도 사용할 MATLAB이라는 컴퓨팅 언어를 사용해 예제 및 연습문제 풀이를 진행했다. 신영민 학생이 발표하고 질의 응답을 통해 진행했다.



Splin 보간법

예제 3.1

spline_3.m

```
clear; clc; close all;

data=[1 2 3 4 5
    1 1/2 1/3 1/4 1/5];     % data 값 넣기

x=data(1,:); y=data(2,:);   % data를 x값과 y값의 넣기
n=size(x',1)-1;     % data의 개수 보다 하나 작게 보간함수 생성

a=y; c=zeros(1,n+1); b=zeros(1,n); d=zeros(1,n);        % 초기값 설정해주기
h=zeros(1,n);       % h 초기값 설정해주기
```

매트랩으로 풀기 위하여 data를 입력하고 a, b, c, d, h의 초기값을 설정해 주었다.

Splin 보간법

예제 3.1

```
for ik=1:n
    h(ik)=x(ik+1)-x(ik);        % h값 설정
end
```

$h_i = x_{i+1} - x_i$ 이므로 for 반복문을 사용해 설정해 준다.

```
%%% c 값 구하기 %%%

A=zeros(n-1);               % 선형 연립 방정식을 풀기 위한 A 행렬
u=zeros(1,n-1);             % 선형 연립 방정식을 풀기 위한 u 행렬
```

c값들을 구하기 위한 선형 연립 방정식을 풀기 위하여 그에 필요한 A, u 행렬을 초기화 해준다.

Splin 보간법

예제 3.1

```
%%% A 행렬 설정하기 %%%
A(1,1)=2*(h(1)+h(2)); A(1,2)=h(2);
A(n-1,n-2)=h(n-2); A(n-1,n-1)=2*(h(n-2)+h(n-1));

for ik=2:n-2
    A(ik,ik-1)=h(ik);
    A(ik,ik)=2*(h(ik)+h(ik+1));
    A(ik,ik+1)=h(ik);
end
```

명령 창

MATLAB을 처음 사용한다면 시작하기를 참조하십시

```
A =

     4     1     0
     1     4     1
     0     1     4

fx >>
```

A행렬을 설정하기 위해 반복 되지 않는 첫번째 줄과 마지막 줄은 따로 설정하고 그 외 나머지 반복 구간은 for 반복문을 사용해 설정한다.

Splin 보간법

예제 3.1

```
%%% U 행렬 설정하기 %%%
for ik=1:n-1
    u(ik)=3*((y(ik+2)-y(ik+1))/h(ik+1)-(y(ik+1)-y(ik))/h(ik));
end

c(2:n)=A₩u';        % c 값 구하기
```

명령 창

MATLAB을 처음 사용한다면 시작하기를 참조하십시

```
>> u

u =

    1.0000    0.2500    0.1000

fx >>
```

```
>> c

c =

         0    0.2518   -0.0071    0.0268         0
```

u행렬을 for 반복문을 사용해 설정한다. A, u를 사용해 c를 구한다.

Splin 보간법

예제 3.1

구한 값들을 적용해 $S(x)$를 다시 써보면,

$$S(x) = \begin{cases} S_0(x) = 1 - 0.5839(x-1) + 0.0839(x-1)^3, & 1 \le x \le 2 \\ S_1(x) = \dfrac{1}{2} - 0.3321(x-2) + 0.2518(x-2)^2 - 0.0863(x-2)^3, & 2 \le x \le 3 \\ S_2(x) = \dfrac{1}{3} - 0.0875(x-3) - 0.0071(x-3)^2 + 0.0113(x-3)^3, & 3 \le x \le 4 \\ S_3(x) = \dfrac{1}{4} - 0.0679(x-4) + 0.0268(x-4)^2 - 0.0089(x-4)^3, & 4 \le x \le 5 \end{cases}$$

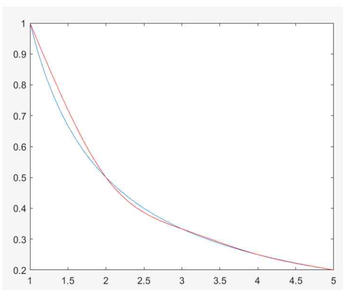
이다.

Splin 보간법

예제 3.1

그래프를 그려보면,(파란색이 실제 값이고, 빨간색이 splin을 이용한 보간 값들이다.)

```
%%% 그래프 그리기 %%%

plot(1:0.1:5,1./(1:0.1:5));     % 실제 값 plot하기

hold on
xx=1:0.1:2;
plot(xx,a(1)+b(1).*(xx-1)+c(1).*(xx-1).^2+d(1).*(xx-1).^3,'r')

hold on
xx=2:0.1:3;
plot(xx,a(2)+b(2).*(xx-2)+c(2).*(xx-2).^2+d(2).*(xx-2).^3,'r')

hold on
xx=3:0.1:4;
plot(xx,a(3)+b(3).*(xx-3)+c(3).*(xx-3).^2+d(3).*(xx-3).^3,'r')

hold on
xx=4:0.1:5;
plot(xx,a(4)+b(4).*(xx-4)+c(4).*(xx-4).^2+d(4).*(xx-4).^3,'r')
```



Splin 보간법

<u>연습문제 3.1</u> 다음 4개의 점들을 지나가는 3차 스플라인 보간 함수 $S(x)$를 구하고 $x = 1.5$에서 f의 값을 추정하여라

x	-1	0	1	2
$y = f(x)$	1	2	4	6

Splin 보간법

<u>연습문제 3.1</u>

```
clear; clc; close all;

data=[-1 0 1 2
    1 2 4 6];      % data 값 넣기

x=data(1,:); y=data(2,:);    % data를 x값과 y값의 넣기
n=size(x',1)-1;      % data의 개수 보다 하나 작게 보간함수 생성

a=y; c=zeros(1,n+1); b=zeros(1,n); d=zeros(1,n);        % 초기값 설정해주기
h=zeros(1,n);        % h 초기값 설정해주기

for ik=1:n
    h(ik)=x(ik+1)-x(ik);        % h값 설정
end
```

매트랩을 활용해 a, b, c, d를 구해 봤다.

Splin 보간법

<u>연습문제 3.1</u>

```
명령 창
MATLAB을 처음 사용한다면 시작하기를 참조하십시오.
>> syms xx
>> S(xx)=a(3)+b(3)*(xx-1)+c(3)*(xx-1)^2+d(3)*(xx-1)^3

S(xx) =

(32*xx)/15 - (xx - 1)^2/5 + (xx - 1)^3/15 + 28/15

>> S(1.5)

ans =

201/40

>> 201/40

ans =

    5.0250

fx >>
```

답 : 5.0250

Splin 보간법

<u>연습문제 3.2</u> 다음 7개의 점들을 지나가는 3차 스플라인 보간 함수 $S(x)$를 구하고 $S(x)$의 그래프를 그려보아라

x	0	0.2	0.3	0.5	0.6	0.9	1.0
$y=f(x)$	10	15	16	16	14	12	11

Splin 보간법

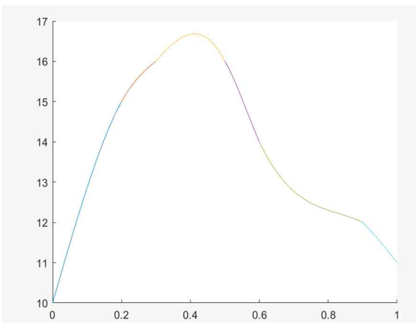
<u>연습문제 3.2</u>

```
clear; clc; close all;

data=[0 0.2 0.3 0.5 0.6 0.9 1.0
    10 15 16 16 14 12 11];      % data 값 넣기

x=data(1,:); y=data(2,:);   % data를 x값과 y값의 넣기
n=size(x',1)-1;     % data의 개수 보다 하나 작게 보간함수 생성

a=y; c=zeros(1,n+1); b=zeros(1,n); d=zeros(1,n);      % 초기값 설정해주기
h=zeros(1,n);      % h 초기값 설정해주기

for ik=1:n
    h(ik)=x(ik+1)-x(ik);       % h값 설정
end
```

매트랩을 활용해 a, b, c, d를 구해 봤다.

Splin 보간법

<u>연습문제 3.2</u>

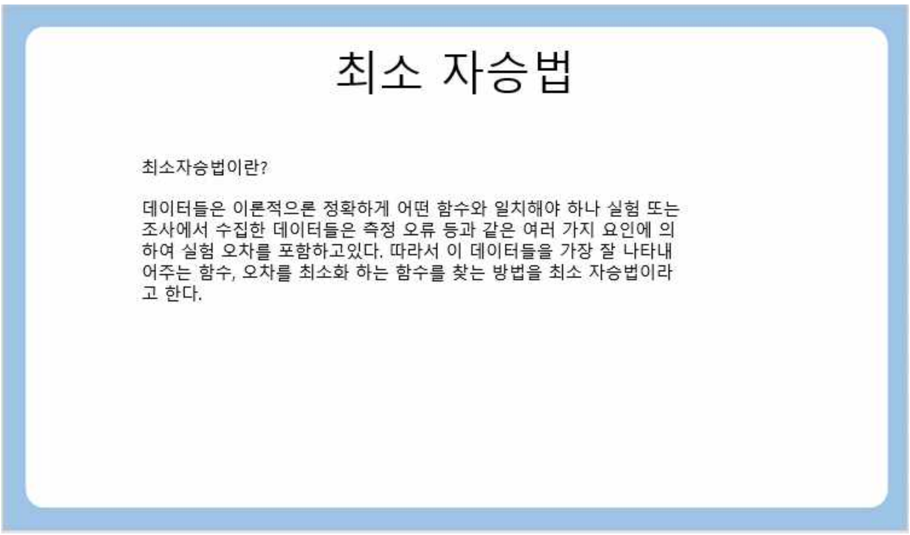


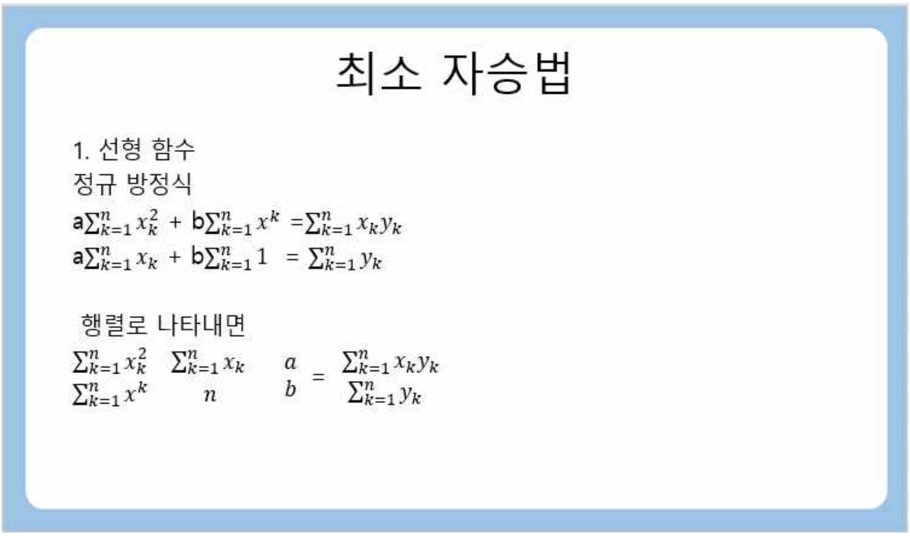
매트랩을 활용해 그래프를 그렸다.

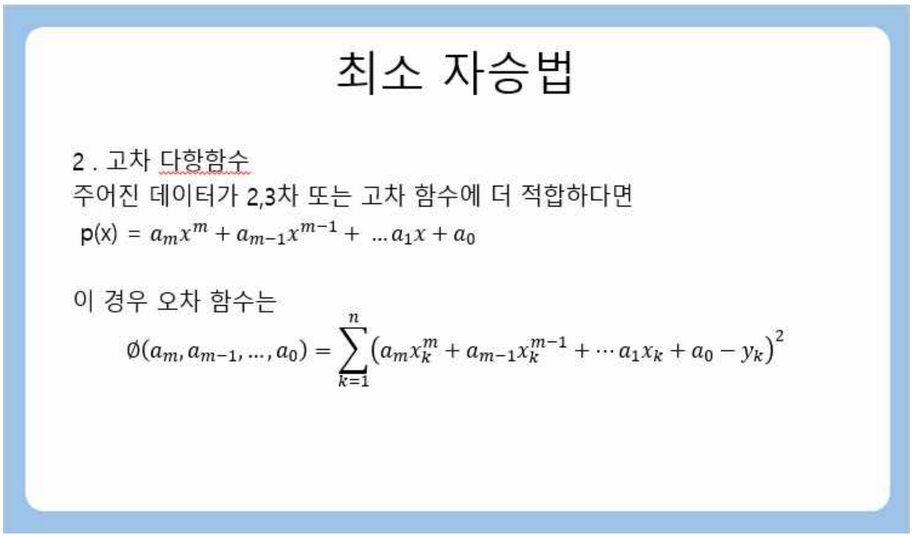
<그림 6, 신영민 학생의 Splin 보간법 실습에 대한 PPT설명 일부>

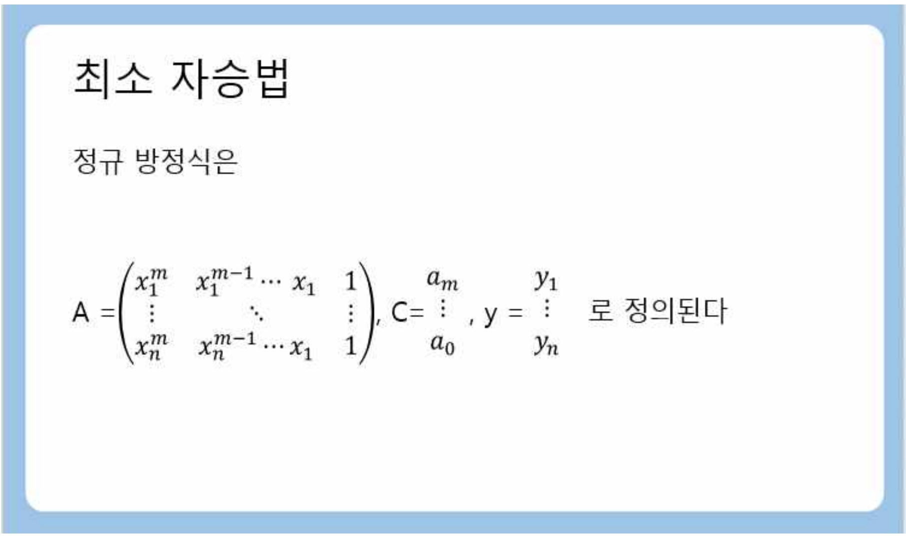
7주차 - 최소자승법(1)

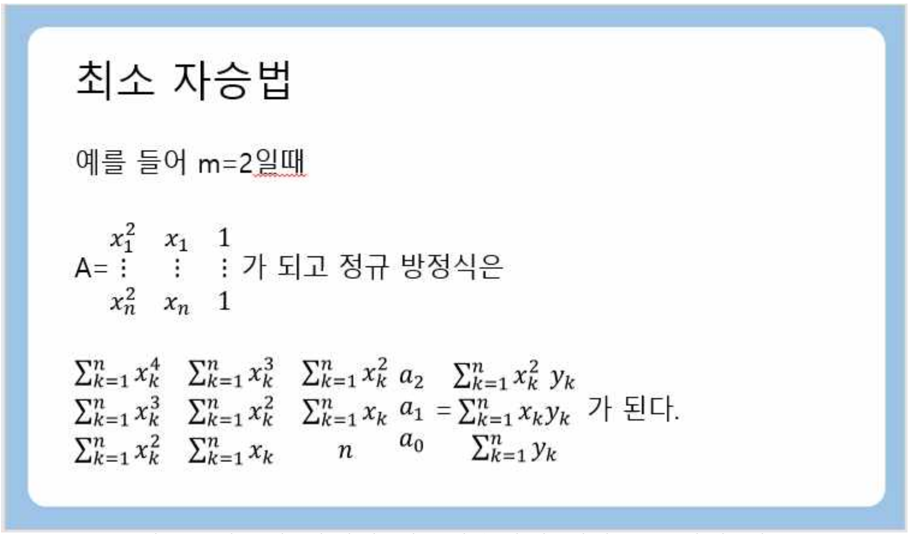
김홍관 학생 주도하에 2차원 평면을 이용한 3차원 복원을 위한 최소자승법 대한 개념을 학습하고 ‘이공학을 위한 MATLAB 활용 수치해석’ 서적에 1차, 2차, 고차 다항식 함수의 오차 함수와 정규방정식을 찾는 방법을 공부했다. 최소자승법은 오차를 최소화해야 하는 상황에서 사용하면 좋은 방법이다.







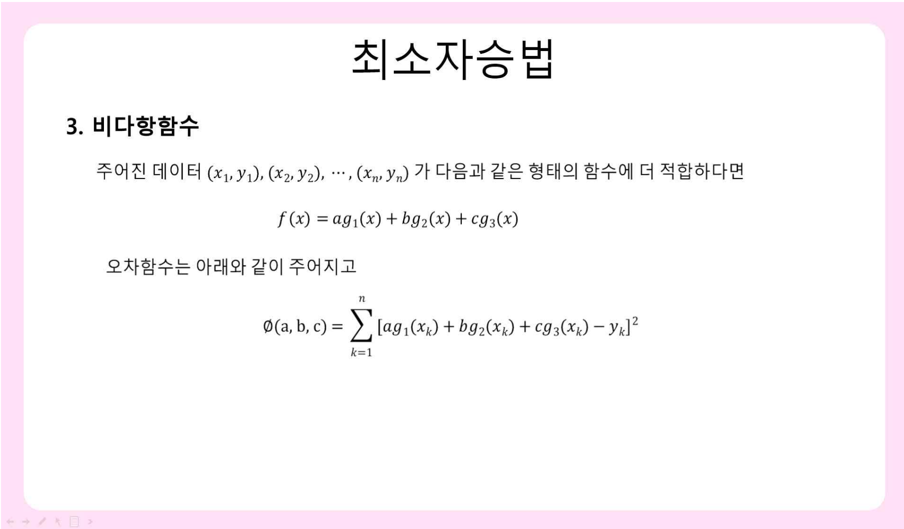


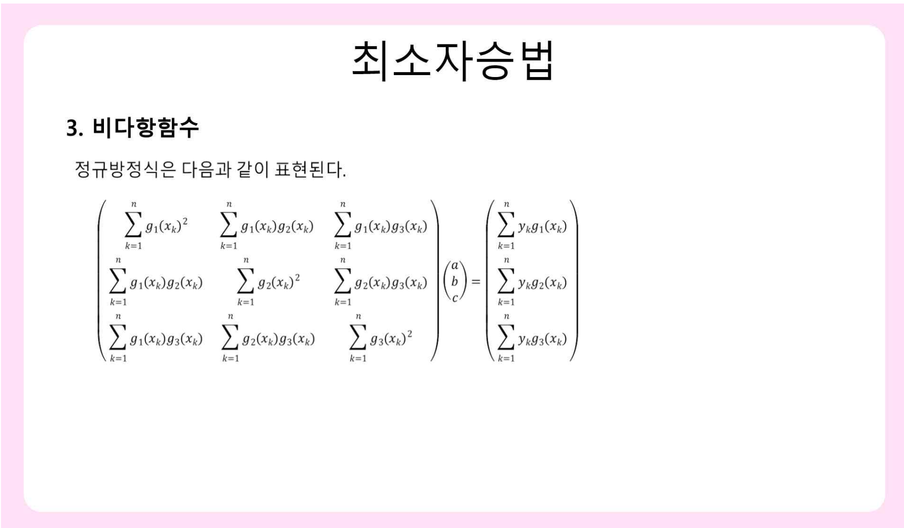


<그림 7, 김홍관 학생의 최소자승법에 대한 PPT설명 일부>

9주차 - 최소자승법(2)

7주차에 이어서 1차, 2차, 고차 다항식 함수의 오차 함수와 정규방정식을 찾는 방법을 바탕으로 박세영 학생이 '이공학을 위한 MATLAB 활용 수치해석' 서적에 비다항함수, 비선형 문제를 ppt형식으로 정리하여 설명하고 MATLAB을 이용하여 연습문제를 풀어 보았다.





최소자승법

4. 비선형 문제

비선형 방정식 또는 비선형 연립방정식 형태로 나타낼 수 있다.
주어진 데이터 $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$가 함수 $y = e^{ax}$에 적합하다고 예상하면 가장 적절한 a의 값을 구해주면 된다. 이 경우 오차함수는

$$\emptyset(a) = \sum_{k=1}^{n} [e^{ax_k} - y_k]^2$$

가 되고 다루어야 할 방정식은

$$2\sum_{k=1}^{n} (e^{ax_k} - y_k) e^{ax_k} x_k = 0$$

가 된다. 위의 식은 a에 대하여 비선형이다.

최소자승법

4. 비선형 문제

만약 주어진 데이터가 2개의 매개변수를 갖는 함수 $y = be^{ax}$에 더욱 적합하다면 오차함수는

$$\emptyset(a, b) = \sum_{k=1}^{n} [be^{ax_k} - y_k]^2$$

가 되며 도출되는 방정식은 다음과 같이 비선형 연립방정식이 된다.

$$2\sum_{k=1}^{n} (be^{ax_k} - y_k) be^{ax_k} x_k = 0$$

$$2\sum_{k=1}^{n} (be^{ax_k} - y_k) e^{ax_k} = 0$$

위의 연립방정식을 Newton 방법으로 다루면 가장 적절한 a, b의 값을 구할 수 있다.

최소자승법

연습문제 7.

아래에 주어진 자료를 가장 잘 나타내는 방정식 $y = c\log x$를 구하여라.

x	-1	0	1
y	0.5	1	3

```
clear; clc; close all;

data=[-1 0 1
    0.5 1 3];        % data

x=data(1,:);         % split data
y=data(2,:);         % split data

a=0; error=0.0001;       % initial value & error
```

최소자승법

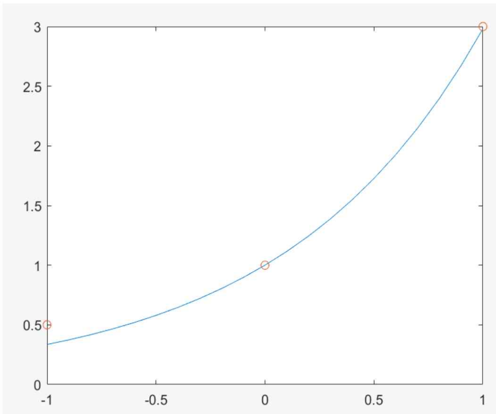
연습문제 7.

```
while 1
    sumf=0; sumfp=0;        % Initialization
    for ik=1:3
        sumf=sumf+exp(2*a*x(ik))*x(ik)-exp(a*x(ik))*x(ik)*y(ik);        % fx value
        sumfp=sumfp+2*exp(2*a*x(ik))*x(ik)^2-exp(a*x(ik))*x(ik)^2*y(ik);    % fx prime value
    end

    an=a-sumf/sumfp;        % newton method

    if abs(an-a)<error
        break        % check error
    end

    a=an;

end
```

최소자승법

연습문제 7.

```
t=-1:0.1:1;      % graph
plot(t,exp(a*t))
hold on
plot(data(1,:),data(2,:),'o')
```



최소자승법

연습문제 8.

문제 7에 주어진 자료가 방정식 $y = be^{ax}$에 더 적합하다면 계수 a, b를 구할 수 있도록 식을 세워 보아라.

```
clear; clc; close all;

data=[-1 0 1
    0.5 1 3];        % data

x=data(1,:);         % split data
y=data(2,:);         % split data

a=1; b=1; error=0.0001;      % initial value & error
```

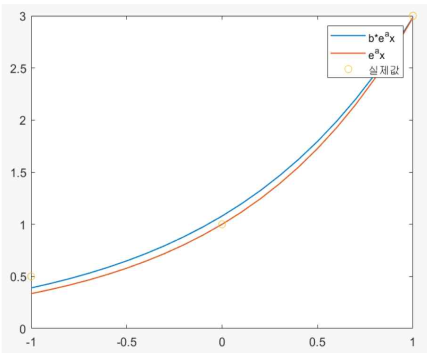
연습문제 8.

```
while 1
    fa=0; fb=0; ga=0; gb=0; f=0; g=0;        % Initialization

    for ik=1:3
        fa=fa+2*b^2*exp(2*a*x(ik))*x(ik)^2-b*exp(a*x(ik))*x(ik)^2*y(ik);
        fb=fb+2*b*exp(2*a*x(ik))*x(ik)-exp(a*x(ik))*x(ik)*y(ik);
        ga=ga+2*b*exp(2*a*x(ik))*x(ik)-exp(a*x(ik))*x(ik)*y(ik);
        gb=gb+exp(2*a*x(ik));
        f=f+b^2*exp(2*a*x(ik))*x(ik)-b*exp(a*x(ik))*x(ik)*y(ik);
        g=g+b*exp(2*a*x(ik))-exp(a*x(ik))*y(ik);
    end

    jac=[fa fb;ga gb];       % jacobian matrix

    abn=[a;b]-jac₩[f;g];        % newton method

    if abs(abn(1)-a)<error && abs(abn(2)-b)<error        % check error
        break
    end

    a=abn(1);
    b=abn(2);

end
```

최소자승법

연습문제 8.

```
t=-1:0.1:1;      % graph
plot(t,b*exp(a*t),'LineWidth',1)
hold on
w=1.0924;
plot(t,exp(w*t),'LineWidth',1)
hold on
plot(data(1,:),data(2,:),'o')

legend('b*e^ax','e^ax','실제값')
```



최소자승법

연습문제 9.

다음 데이터는 세 함수의 일차 결합인 $y = ax^2 + b\sin x + ce^x$을 따른다고 한다.

x	-2	-1	0	1	2
y	5	-8	-1	2	3

a) 계수 a, b, c를 구할 수 있도록 최소자승법에 의한 식을 세워 보아라.
b) 계수 a, b, c의 값들을 구하여라.

```
clear; clc; close all;

data=[-2 -1 0 1 2
      5 -8 -1 2 3];        % data

x=data(1,:);        % split data
y=data(2,:);        % split data

g1_2=0; g2_2=0; g3_2=0; g1g2=0; g1g3=0; g2g3=0;
yg1=0; yg2=0; yg3=0;        % initialization
```

최소자승법

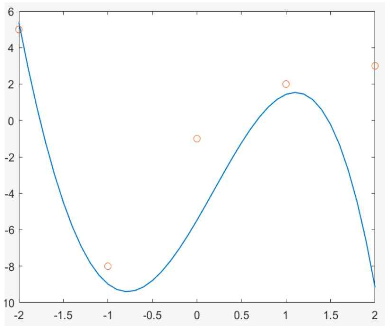
연습문제 9.

```
for ik=1:4
    g1_2=g1_2+x(ik)^4;
    g1g2=g1g2+x(ik)^2*sin(x(ik));
    g1g3=g1g3+x(ik)^2*exp(x(ik));
    g2_2=g2_2+(sin(x(ik)))^2;
    g2g3=g2g3+sin(x(ik))*exp(x(ik));
    g3_2=exp(x(ik)*2);

    yg1=yg1+y(ik)*x(ik)^2;
    yg2=yg2+y(ik)*sin(x(ik));
    yg3=yg3+y(ik)*exp(x(ik));

end
```

최소자승법

연습문제 9.

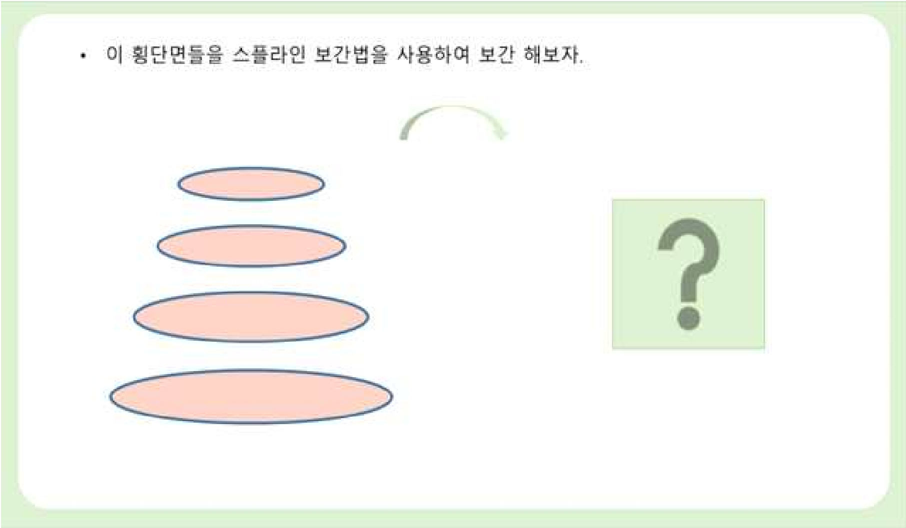
```
A=[g1_2 g1g2 g1g3
    g1g2 g2_2 g2g3
    g1g3 g2g3 g3_2];

B=[yg1;yg2;yg3];

v=A₩B;

a=v(1); b=v(2); c=v(3);

t=-2:0.1:2;      % graph
plot(t,a*t.^2+b*sin(t)+c*exp(t),'LineWidth',1)
hold on
plot(data(1,:),data(2,:),'o')
```

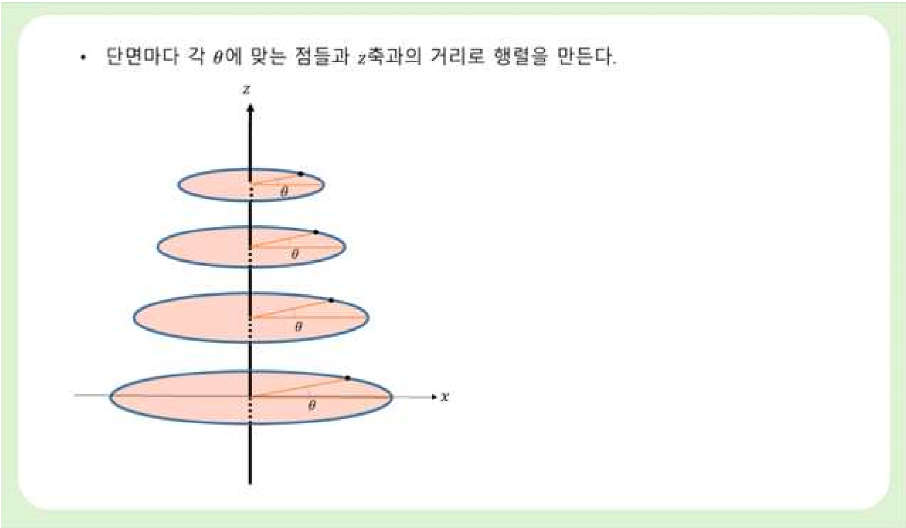


<그림 9. 박세영 학생의 최소자승법에 대한 PPT설명 일부>

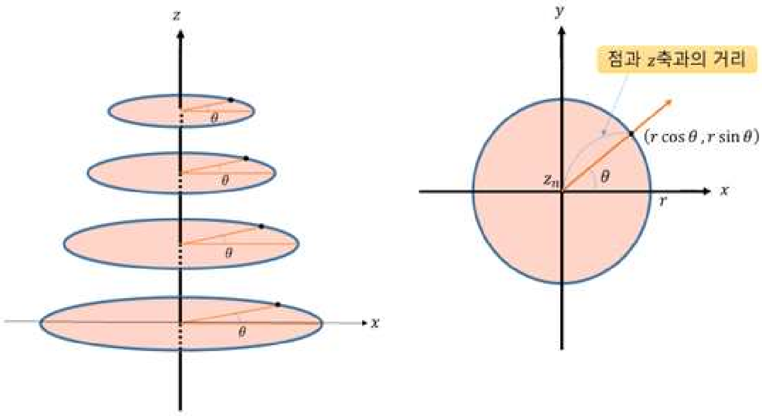
10주차 - 보간법을 이용한 3차원 복원

스플라인 보간법을 사용하여 그래프 사이의 값을 추정하여 복원한다. 횡단면의 x축과의 각이 θ만큼인 모든 점들의 z축과의 거리를 행렬로 만들고, 그 값을 스플라인 보간법을 사용하여 보간한 다음 그 점으로 다시 단면을 그려서 그래프를 완성한다. 이 내용을 함께 공부한 후 박나영 학생이 정리하였다.

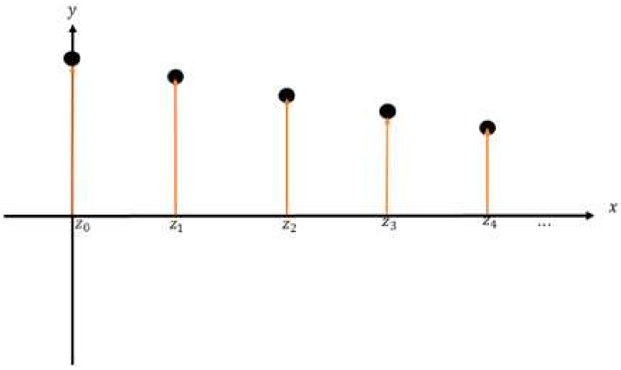




- 단면마다 각 θ에 맞는 점들과 z축과의 거리로 행렬을 만든다.



- z축과의 거리를 y축으로 하여 점을 찍는다.

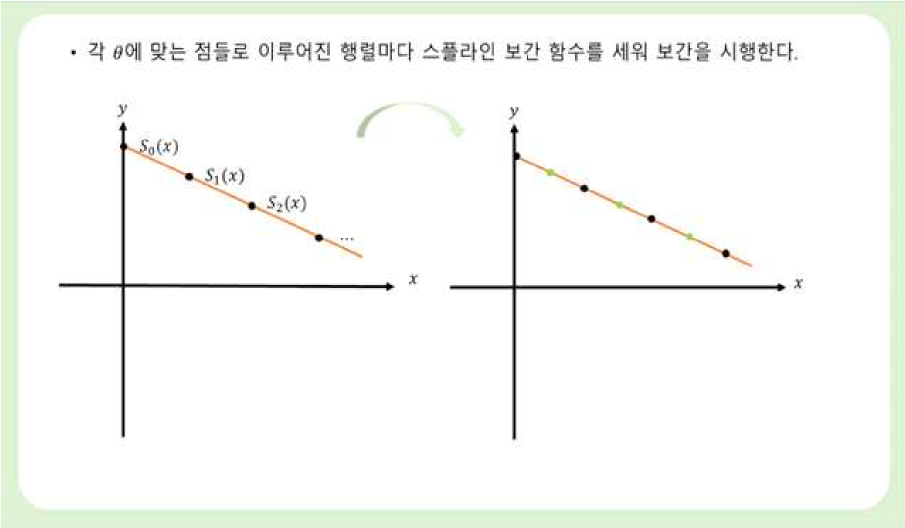


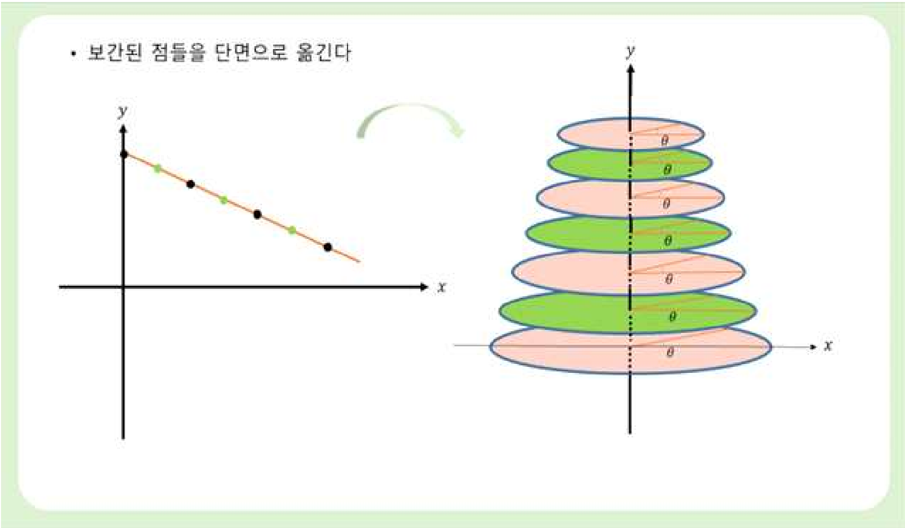
- # 스플라인 보간법

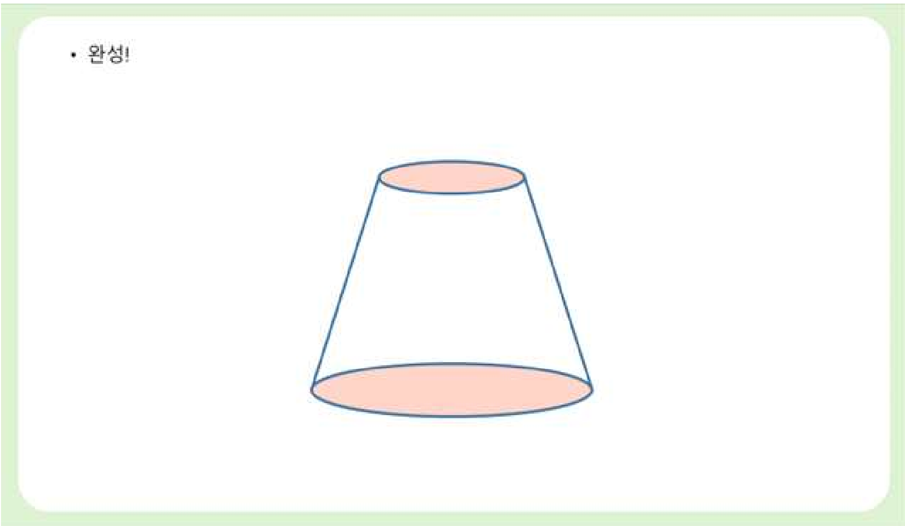
-n+1개의 점들을 지나는 n차 보간다항식의 그래프는 n이 클 때 굴곡이 심하다는 특징이 있다. 이로 인하여 상당히 큰 오차가 발생할 수 있다.

-스플라인은 n차 다항식을 만들지 않고 n+1개의 점들을 이용해서 n개의 다항식을 만든다.

$$S(x) = \begin{cases} S0(x), & (x0 \le x \le x1) \\ S1(x), & (x1 \le x \le x2) \\ \vdots & \vdots \\ Sn(x) & (xn-1 \le x \le xn) \end{cases}$$



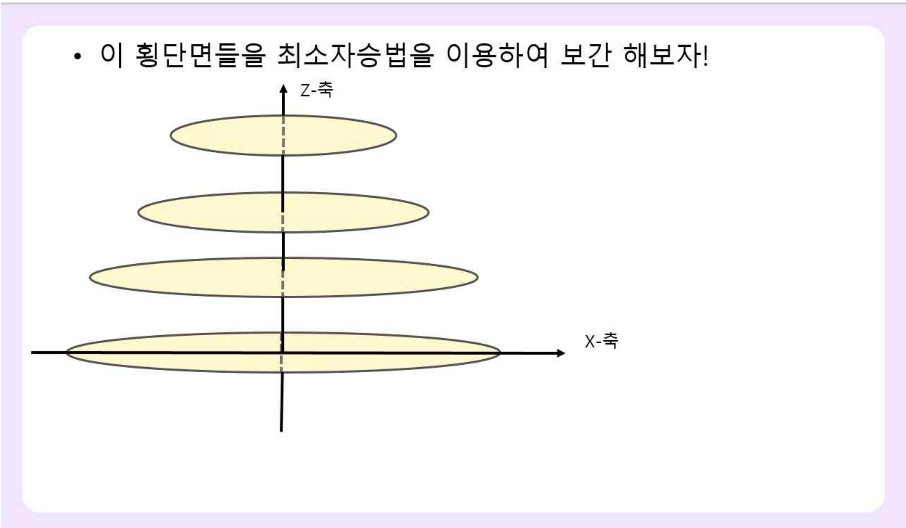


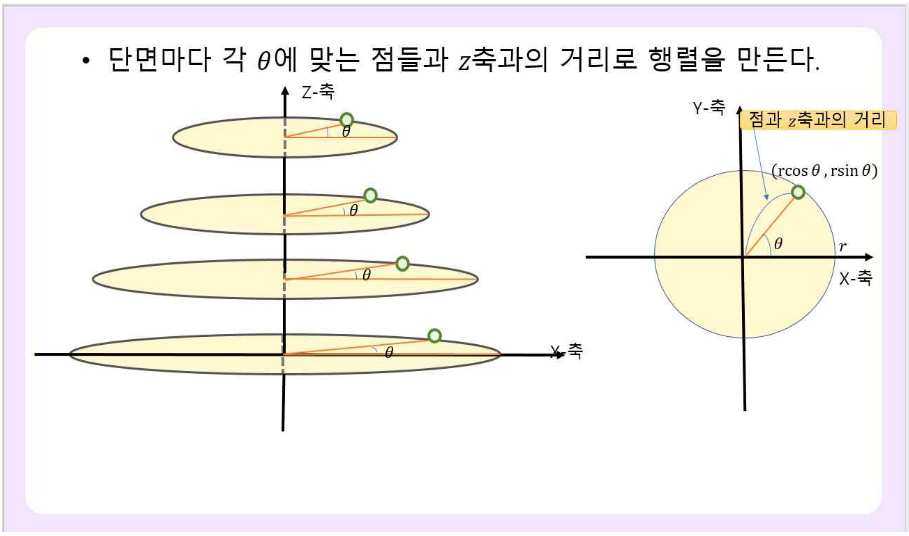


<그림 10, 박나영 학생의 보간법을 이용한 3차원 복원 PPT설명 일부>

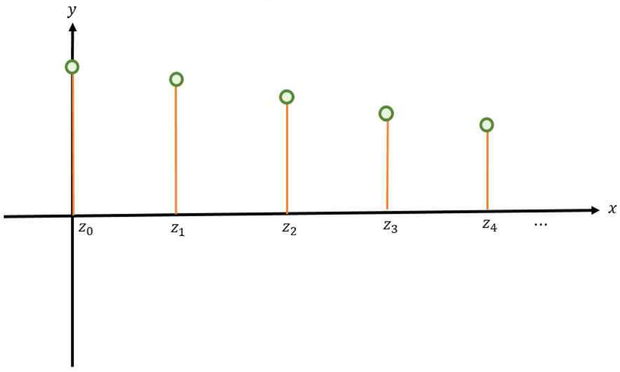
11주차 - 최소자승법을 이용한 3차원 복원

최소자승법을 이용하여 그래프 사이의 값을 추정하여 복원한다. 횡단면의 x축과의 각이 θ만큼인 모든 점들의 **z**축과의 거리를 행렬로 만들고, 그 값을 최소자승법을 이용해 오차가 가장 작은 함수를 생성하고 이에 대해 사이점들을 예측하여 다시 단면을 그려서 그래프를 완성한다. 이 내용을 함께 공부한 후 김하은 학생이 정리하였다.





- z축과의 거리를 y축으로 하여 점을 찍는다.



최소 자승법

2 . 다항함수
주어진 데이터가 다항 함수에 적합하다면
p(x) = $a_m x^m + a_{m-1}x^{m-1} + \ldots a_1 x + a_0$

이 경우 오차 함수는

$$\emptyset(a_m, a_{m-1}, \ldots, a_0) = \sum_{k=1}^{n}\left(a_m x_k^m + a_{m-1}x_k^{m-1} + \cdots a_1 x_k + a_0 - y_k\right)^2$$

라고 나타낼 수 있다.

최소 자승법

정규 방정식은

A = $\begin{pmatrix} x_1^m & x_1^{m-1} \cdots x_1 & 1 \\ \vdots & \ddots & \vdots \\ x_n^m & x_n^{m-1} \cdots x_1 & 1 \end{pmatrix}$, C= $\begin{matrix} a_m \\ \vdots \\ a_0 \end{matrix}$, y = $\begin{matrix} y_1 \\ \vdots \\ y_n \end{matrix}$ 로 정의된다

최소자승법

4. 비선형 문제

비선형 방정식 또는 비선형 연립방정식 형태로 나타낼 수 있다.
주어진 데이터 $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$가 함수 $y = e^{ax}$에 적합하다고 예상하면 가장 적절한 a의 값을 구해주면 된다. 이 경우 오차함수는

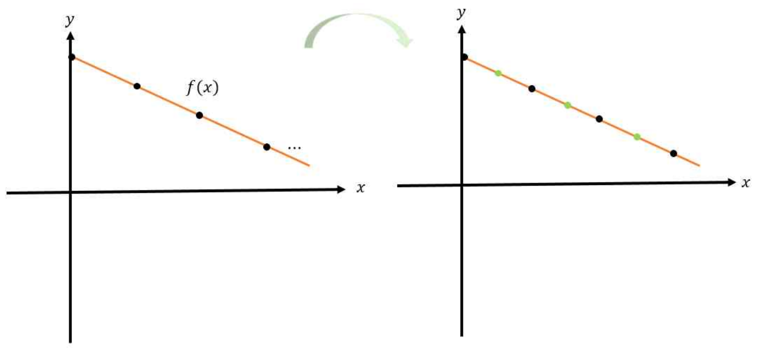
$$\emptyset(a) = \sum_{k=1}^{n} [e^{ax_k} - y_k]^2$$

가 되고 다루어야 할 방정식은

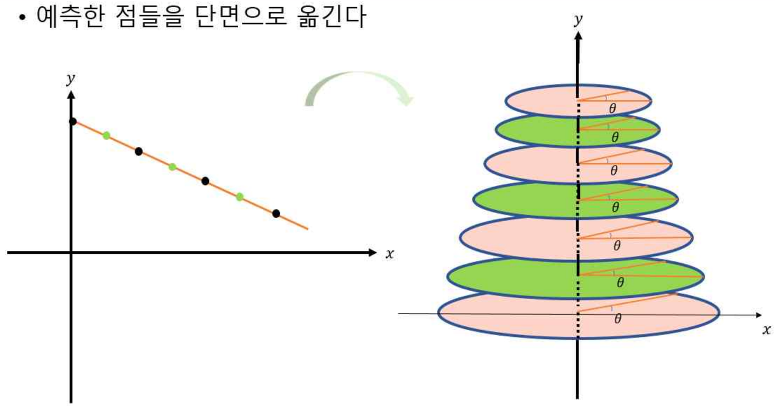
$$2\sum_{k=1}^{n} (e^{ax_k} - y_k)e^{ax_k}x_k = 0$$

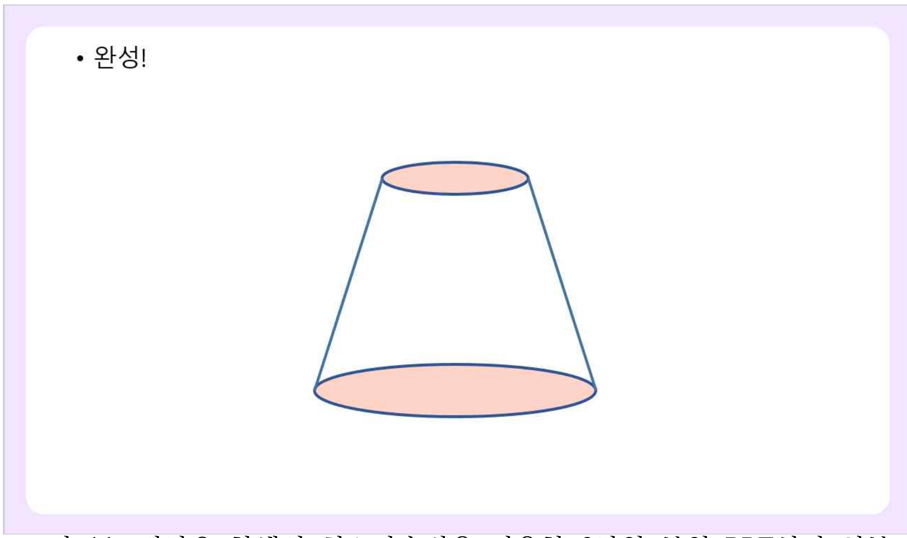
가 된다. 위의 식은 a에 대하여 비선형이다.

- 각 θ에 맞는 점들로 이루어진 행렬마다 최소자승법을 이용해 오차가 가장 작은 함수를 생성하고 이에 대해 사이 점들을 예측합니다.



- 예측한 점들을 단면으로 옮긴다

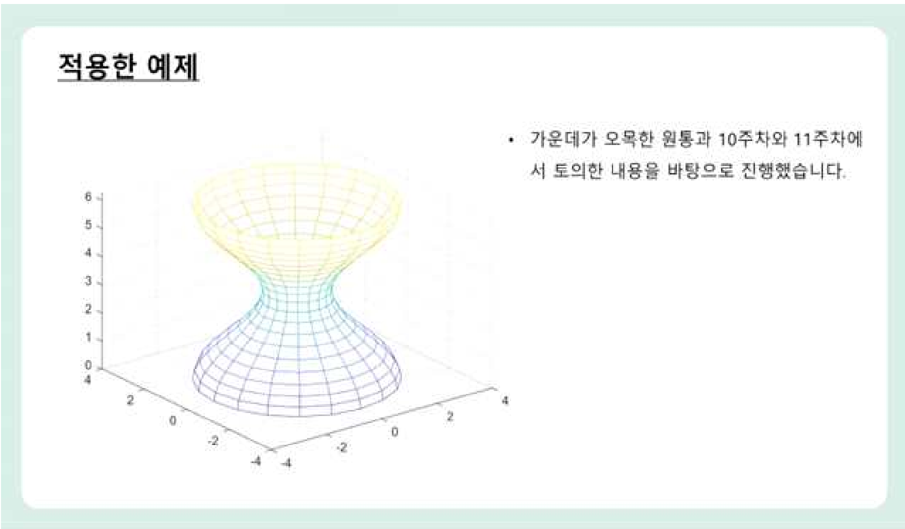




<그림 11, 김하은 학생의 최소자승법을 이용한 3차원 복원 PPT설명 일부>

12주차 – 예제에 적용하기

박나영 학생과 김하은 학생이 10주차와 11주차의 내용을 정리한 것을 바탕으로 신영민 학생이 스플라인 보간법과 최소자승법으로 횡단면의 3차원 복원을 코드로 짜고 예제를 적용하고 이를 ppt형식으로 정리하여 설명했다. 피피티와 코드를 바탕으로 다 같이 실습해 보고 질의 응답시간을 가졌다.



```
% 초기값 설정 %

h=pi/10;
t = 0:h:2*pi; n=size(t',1);     % t는 z의 범위, n은 단면의 개수
the=t;                          % 나눌 평면의 개수
X=zeros(n); Y=X;  Z=X;          % 단면을 나타낼 X행렬과 Y행렬을 초기화
```

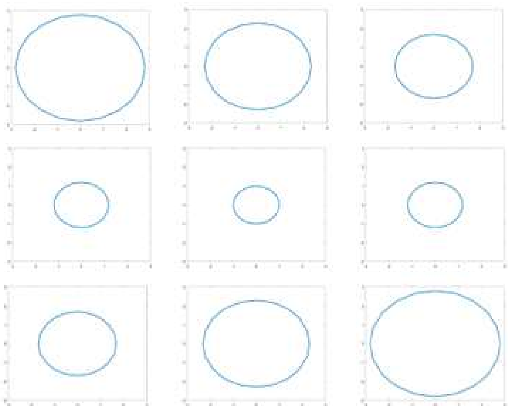
- 초기 값을 위와 같이 설정합니다.

```
for ik=1:n
    r=2*cos(t(ik));             % 반지름

    X(ik,:)=r*cos(the);
    Y(ik,:)=r*sin(the);
    Z(ik,:)=t(ik);
end
```

- 각 단면의 반지름을 r이라고 설정하고, 그에 맞춰 단명을 생성하고 저장한다.

```
% 단면 그리기 %
STR1='one';
STR2 = '.jpg';

for ik=1:2:n
    figure
    plot(X(ik,:),Y(ik,:),'LineWidth',2);
    axis([-3 3 -3 3]);
    filename = strcat(STR1,int2str(ik), STR2); %input 파일명 조합
    saveas(gcf, filename); %저장한다. 인자는 이미지명, 출력이름, 확장자순이다.
end
```

- 단면 이미지 확인을 위해 단면을 그린다.

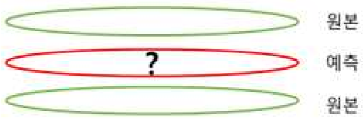


- 단면을 그린 결과 이다.

3차 스플라인 보간

```
tt = 0:h/2:2*pi; nn=size(tt',1);        % 가운데 값 하나 더 넣기
XX=zeros(nn,n); YY=XX; ZZ=XX;           % 근사한 값을 넣기 위한 XX와 YY
```

- 단면 사이의 값을 하나 씩 더 넣었다.




```
for ik=1:n

    coor=[X(:,ik)';Y(:,ik)'];           % X, Y 좌표값 넣기
    data=zeros(2,n);                    % 스플라인에 넣을 데이터 초기화

    for it=1:n
        data(1,it)=t(it);               % x를 Z축으로 잡기
        data(2,it)=sqrt(coor(1,it)^2+coor(2,it)^2);     % z축과의 거리가 y가 된다
    end
```

- 같은 θ 값을 갖는 x, y 값을 찾아 넣는다.

- z축이 x값, z축과 점 사이의 거리가 y값이 된다.

```
[a,b,c,d]=spline_3(data);               % 스플라인으로 보간함수 만들기
```

- 코드가 너무 길어 지기 때문에 스플라인 보간은 함수를 만들어 사용했다.

```
function [a,b,c,d]=spline_3(data)

x=data(1,:); y=data(2,:);   % data를 x값과 y값의 넣기
n=size(x',1)-1;     % data의 개수 보다 하나 작게 보간함수 생성

a=y; c=zeros(1,n+1); b=zeros(1,n); d=zeros(1,n);        % 초기값 설정해주기
h=zeros(1,n);       % h 초기값 설정해주기

for ik=1:n
    h(ik)=x(ik+1)-x(ik);        % h값 설정
end

%%% c 값 구하기 %%%

A=zeros(n-1);               % 선형 연립 방정식을 풀기 위한 A 행렬
u=zeros(1,n-1);             % 선형 연립 방정식을 풀기 위한 u 행렬
```

- Data를 입력하면 3차 스플라인 보간법을 실행해 주는 함수를 만들었다.

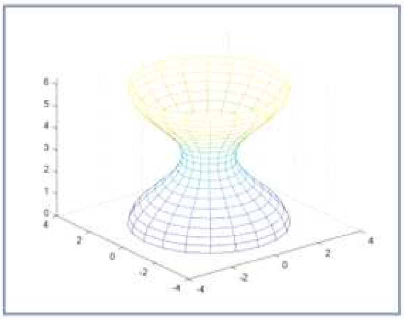
```
%%% A 행렬 설정하기 %%%
A(1,1)=2*(h(1)+h(2)); A(1,2)=h(2);
A(n-1,n-2)=h(n-2); A(n-1,n-1)=2*(h(n-2)+h(n-1));

for ik=2:n-2
    A(ik,ik-1)=h(ik);
    A(ik,ik)=2*(h(ik)+h(ik+1));
    A(ik,ik+1)=h(ik);
end

%%% U 행렬 설정하기 %%%
for ik=1:n-1
    u(ik)=3*((y(ik+2)-y(ik+1))/h(ik+1)-(y(ik+1)-y(ik))/h(ik));
end

c(2:n)=A\u';        % c 값 구하기
```

```
%%% d 값 구하기 %%%
for ik=1:n
    d(ik)=(c(ik+1)-c(ik))/(3*h(ik));
end

%%% b 값 구하기 %%%
for ik=1:n
    b(ik)=(y(ik+1)-y(ik))/h(ik)-h(ik)*(c(ik+1)+2*c(ik))/3;
end
```

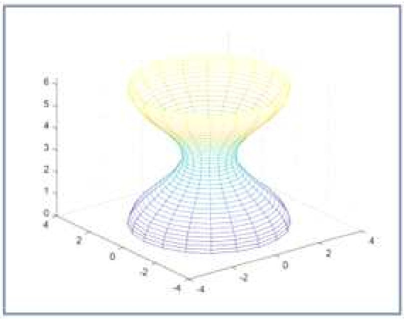
```
    % 보간한 좌표를 평면에 맞춰서 넣기
    for it=1:n-1
        x=t(it):h/2:t(it+1);
        y=a(it)+b(it).*(x-t(it))+c(it).*(x-t(it)).^2+d(it).*((x)-t(it)).^3;
        XX(2*it-1,ik)=y(1)*cos(the(ik)); YY(2*it-1,ik)=y(1)*sin(the(ik));
        XX(2*it,ik)=y(2)*cos(the(ik)); YY(2*it,ik)=y(2)*sin(the(ik));
        XX(2*it+1,ik)=y(3)*cos(the(ik)); YY(2*it+1,ik)=y(3)*sin(the(ik));
    end

end

for ik=1:nn
    ZZ(ik,:)=tt(ik);
end
```

- 스플라인 보간 후 받은 계수들을 사용해 중간에 값을 하나씩 더 넣어 보간 했다.

결과



원본



결과

최소자승법: 다항식

```
x = 0:h/2:2*pi; nn=size(x',1);          % 가운데 값 하나 더 넣기
XX=zeros(nn,n); YY=XX; ZZ=XX;            % 근사한 값을 넣기 위한 XX와 YY

for ik=1:n
    y=zeros(1,nn);
    coor=[X(:,ik)';Y(:,ik)'];            % X, Y 좌표값 넣기
    data=zeros(2,n);                     % 최소자승법에 넣을 데이터 확인
    for it=1:n
        data(1,it)=t(it);                % Z를 x로 잡기
        data(2,it)=sqrt(coor(1,it)^2+coor(2,it)^2);     % z축과의 거리가 y가 된다
    end
```

- 값을 넣는 방법과 데이터를 생성하는 방법은 스플라인 보간법과 동일하게 진행했다.

```
C=least_polynomial(data,m);          % 최소자승법으로 보간함수 만들기. 이 때 m이 우리 가원하는 고차다항식의 차수
```

- 코드가 너무 길어 지기 때문에 최소자승법은 함수를 만들어 사용했다.

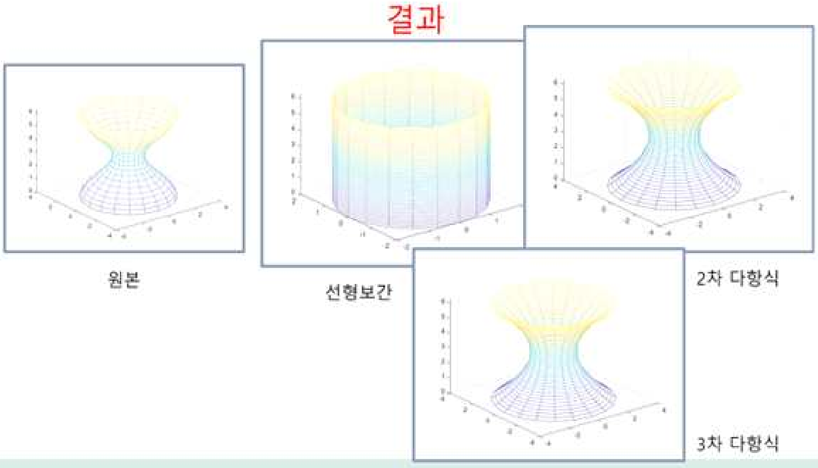
```
function C=least_polynomial(data,m)

X=data(1,:); Y=data(2,:);

n=size(X',1);

A=ones(n,m+1);

for ik=1:m
    for it=1:n
        A(it,ik)=X(it)^(m+1-ik);
    end
end

C=(A'*A)₩A'*Y';

end
```

- Data와 차수를 입력하면 계수들을 반환해주는 함수를 만들어 사용했다.

```
    for iz=1:m+1
        y=y+x.^(m+1-iz)*C(iz);           % 각 열에 최소자승법을 사용해 보간한 Z축과의 거리
    end
%     return
    XX(:,ik)=y*cos(the(ik));             % x 좌표 값 넣기
    YY(:,ik)=y*sin(the(ik));             % y 좌표 값 넣기
end

for ik=1:nn
    ZZ(ik,:)=x(ik);
end
```

- 최소자승법 사용 후 받은 계수들을 사용해 중간에 값을 하나씩 더 넣어 보간 했다.

결과



원본

선형보간

2차 다항식

3차 다항식

최소자승법: $y = e^{ax}$

```
x = 0:h/2:2*pi; nn=size(x',1);          % 가운데 값 하나 더 넣기
XX=zeros(nn,n); YY=XX; ZZ=XX;            % 근사한 값을 넣기 위한 XX와 YY

for ik=1:n
    y=zeros(1,nn);
    coor=[X(:,ik)';Y(:,ik)'];            % X, Y 좌표값 넣기
    data=zeros(2,n);                     % 최소자승법에 넣을 데이터 확인
    for it=1:n
        data(1,it)=t(it);                % Z를 x로 잡기
        data(2,it)=sqrt(coor(1,it)^2+coor(2,it)^2);     % z축과의 거리가 y가 된다
    end
```

- 값을 넣는 방법과 데이터를 생성하는 방법은 위 방법들과 동일하게 진행했다.

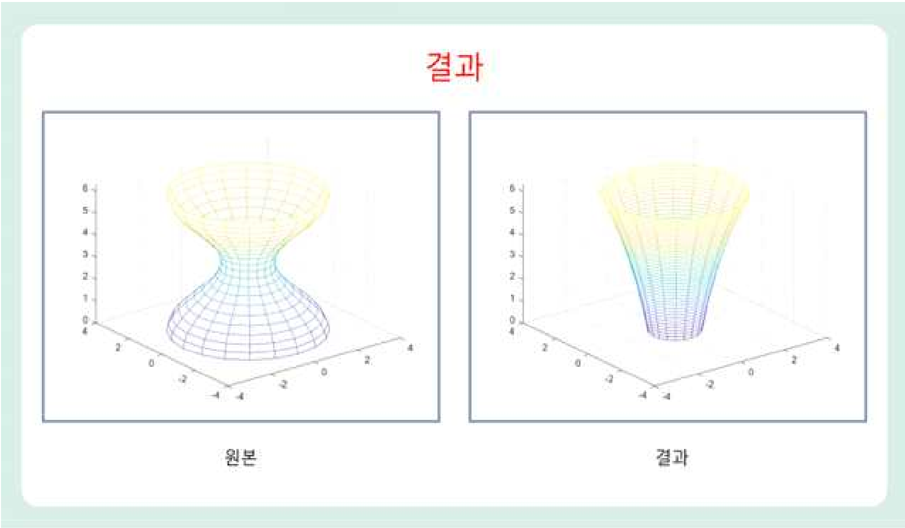
```
C=newton_least1(data);              % y=b*exp(a*x)로 근사해 보기
```

- 코드가 너무 길어 지기 때문에 최소자승법은 함수를 만들어 사용했다.

```
function C=newton_least1(data)

x=data(1,:);         % split data
y=data(2,:);         % split data

n=size(x',1);

a=1; error=0.0001;      % initial value & error

while 1
    sumf=0; sumfp=0;        % initialization
    for ik=1:n
        sumf=sumf+exp(2*a*x(ik))*x(ik)-exp(a*x(ik))*x(ik)*y(ik);        % fx value
        sumfp=sumfp+2*exp(2*a*x(ik))*x(ik)^2-exp(a*x(ik))*x(ik)^2*y(ik);    % fx prime value
    end

    an=a-sumf/sumfp;        % newton method
```

```
    if abs(an-a)<error
        break       % check error
    end

    a=an;

end
C=a;

end
```

- 비선형 연립 방정식의 수치해법인 뉴턴법을 사용해 진행했다.

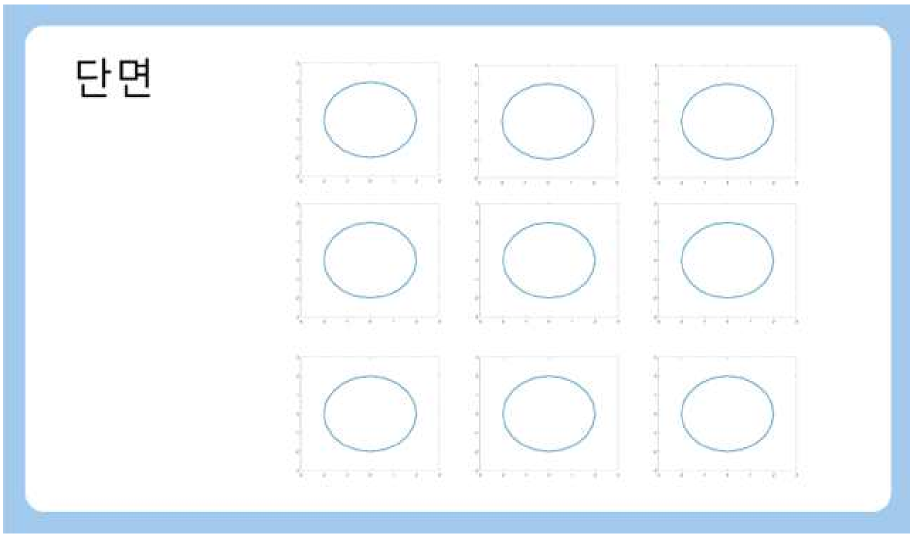


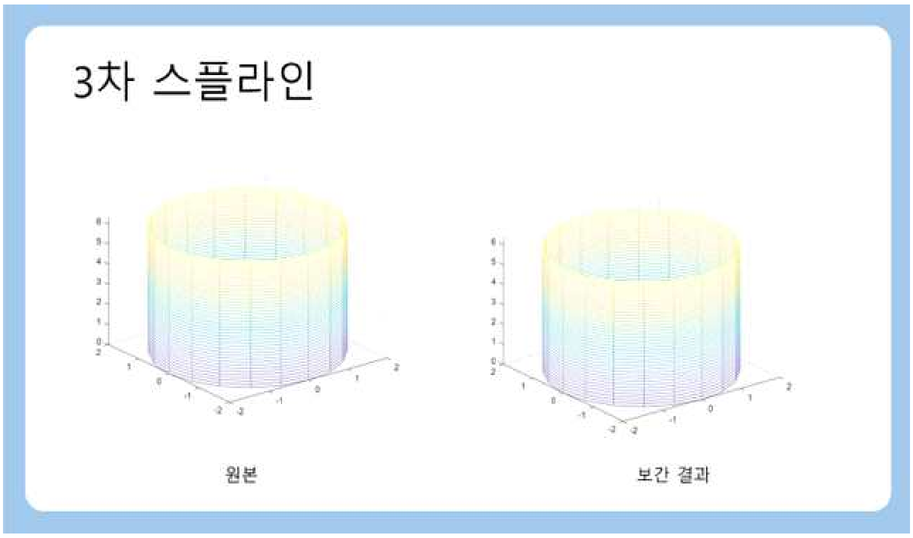
<그림 12. 신영민 학생의 예제에 적용하기 PPT설명 일부>

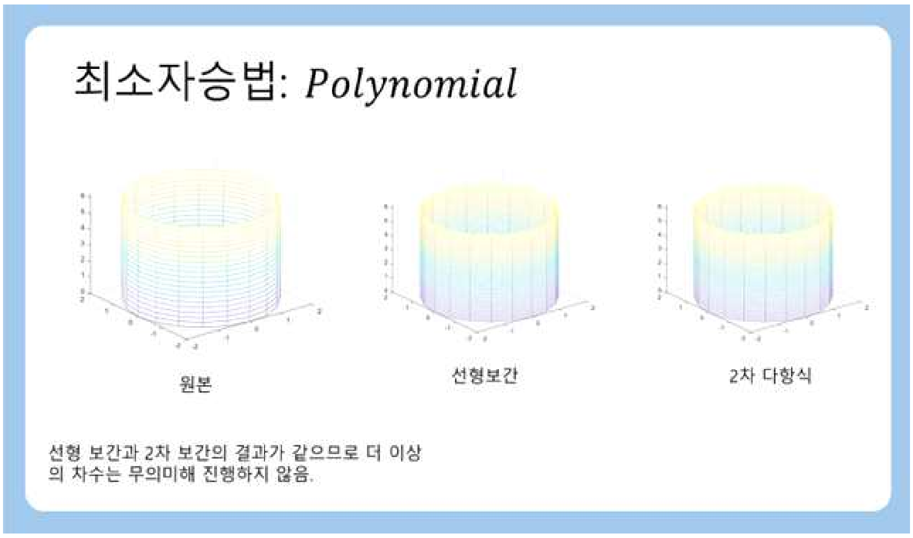
13주차-다양한 도형에 적용 및 오차분석

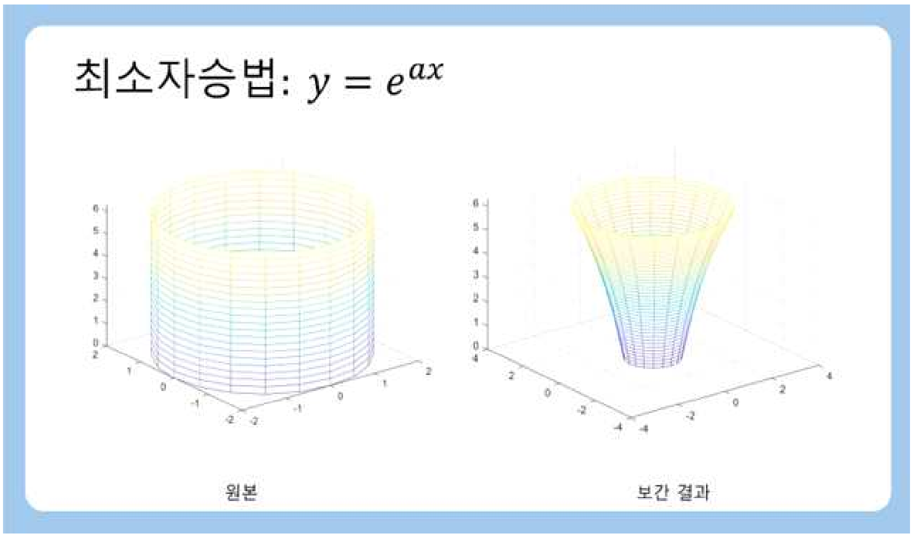
신영민 학생이 적용한 도형을 바탕으로 스플라인 보간법, 다항식을 이용한 최소자승법, 지수함수를 이용한 최소자승법을 사용하여 다양한 도형을 복원해 보았다. 원기둥, 변형된 원기둥, 원뿔, 구 이 4가지 도형에 적용해보았다.

원기둥



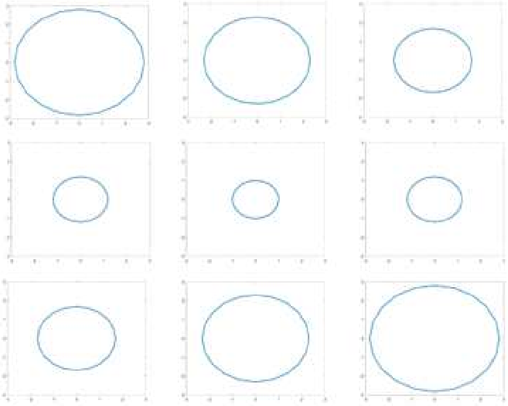



최소자승법: $Polynomial$
원본
선형보간
2차 다항식
선형 보간과 2차 보간의 결과가 같으므로 더 이상
의 차수는 무의미해 진행하지 않음.

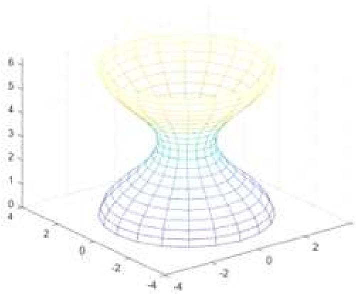

최소자승법: $y = e^{ax}$
원본
보간 결과

변형된 원기둥

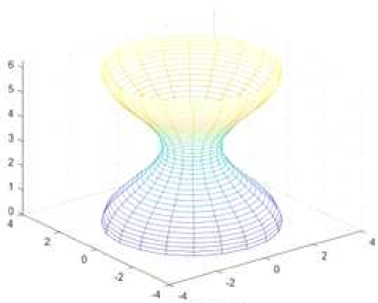
단면



3차 스플라인

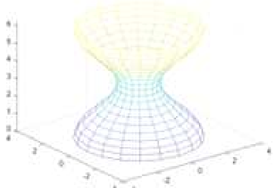


원본

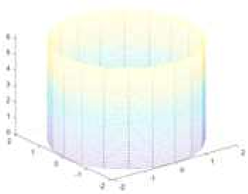


보간 결과

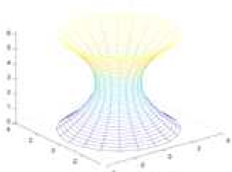
최소자승법: *Polynomial*



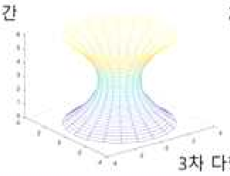
원본



선형보간

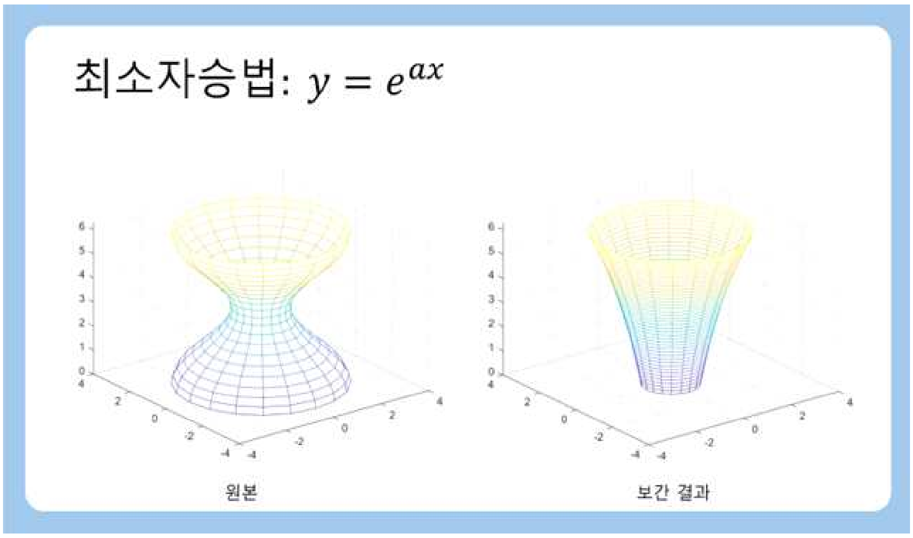


2차 다항식

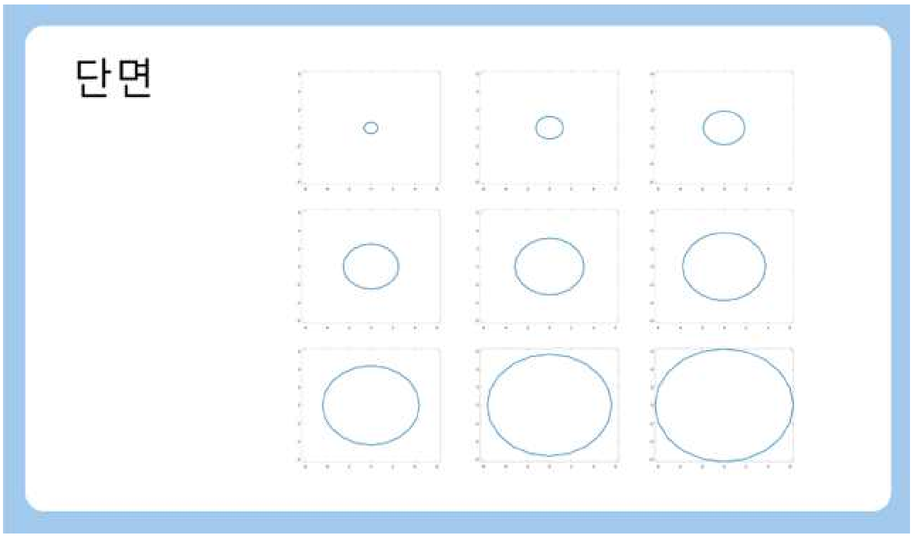


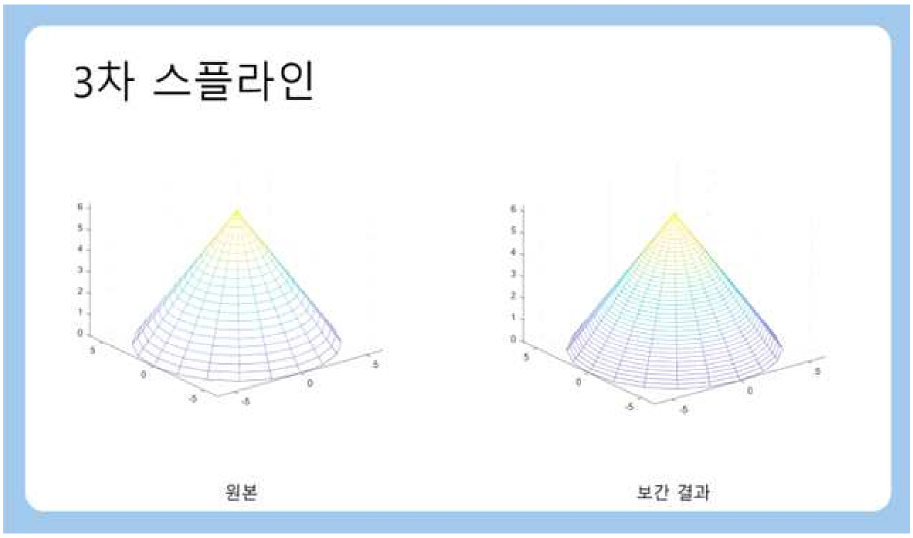
3차 다항식

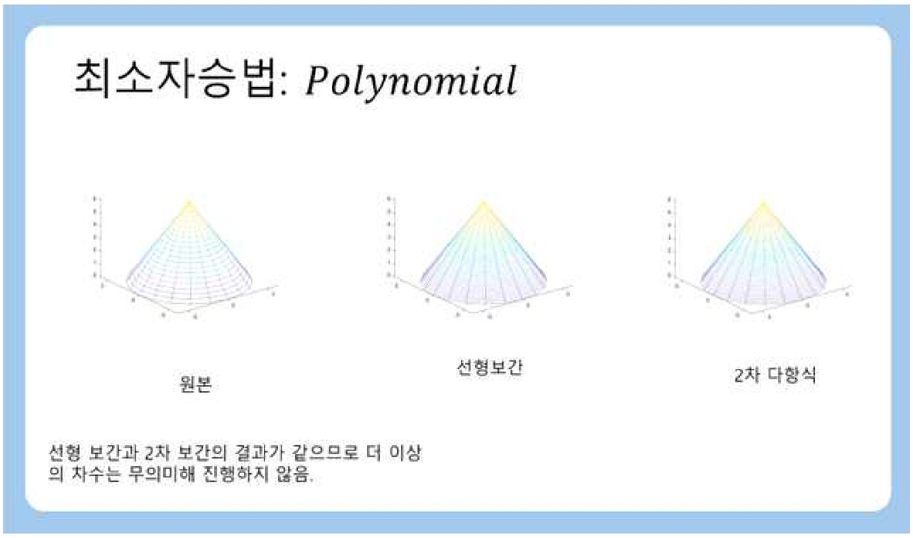
2차 보간과 3차 보간의 결과가 같으므로 더 이상의 차수는 무의미해 진행하지 않음.



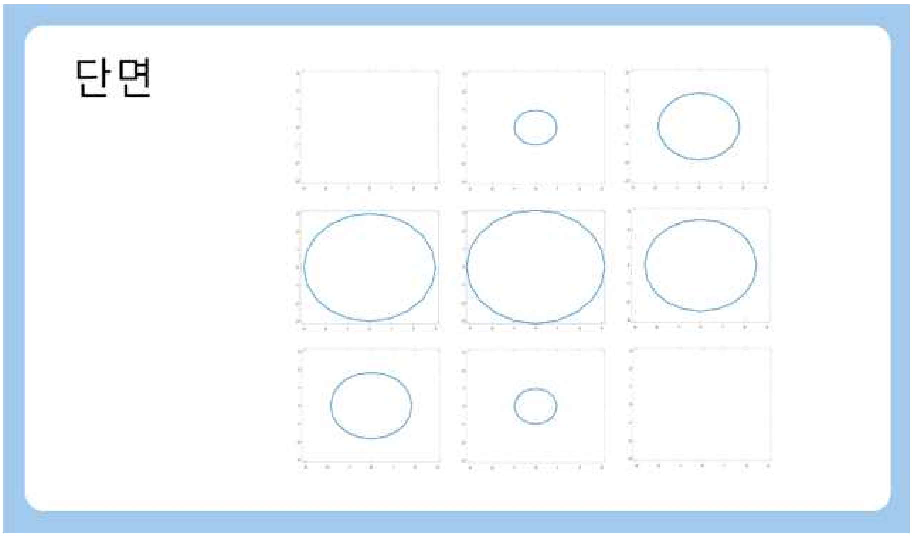
원뿔

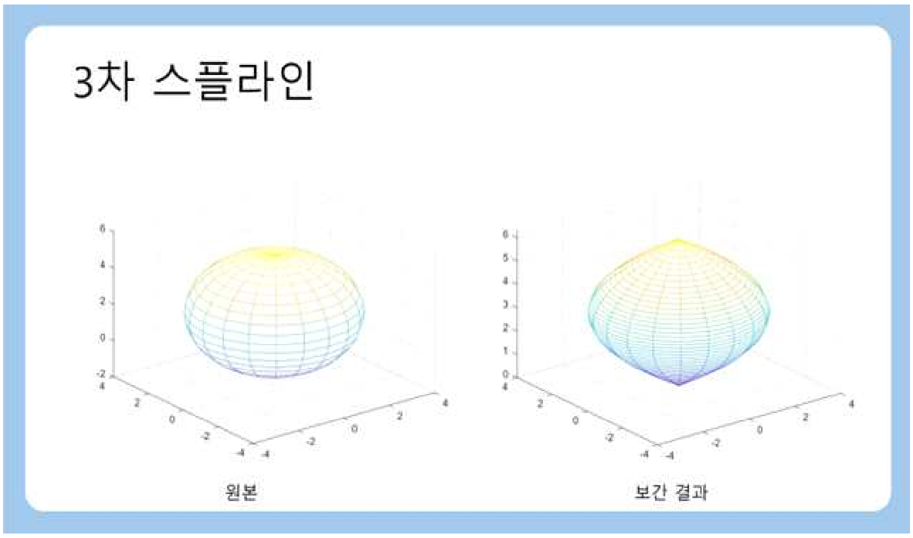


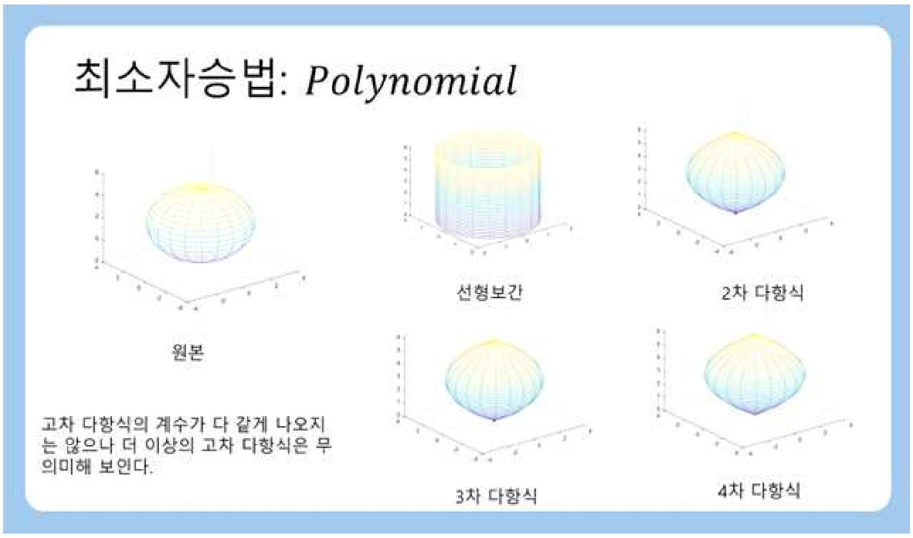



최소자승법: Polynomial
원본
선형보간
2차 다항식
선형 보간과 2차 보간의 결과가 같으므로 더 이상
의 차수는 무의미해 진행하지 않음.

구


단면


3차 스플라인
원본
보간 결과



<그림 13, 김홍관 학생의 다양한 도형에 적용 및 오차분석 PPT설명 일부>

Ⅴ. 참고자료

[1] https://zdnet.co.kr/view/?no=20200618092955

[2] https://ko.wikipedia.org/wiki/%EB%B3%B4%EA%B0%84%EB%B2%95

[3] https://ko.wikipedia.org/wiki/%EC%B5%9C%EC%86%8C%EC%A0%9C%EA%B3%B1%EB%B2%95

[3] 이공학을 위한 MATLAB 활용 수치해석(서적)

[4] https://kr.mathworks.com/products/matlab.html