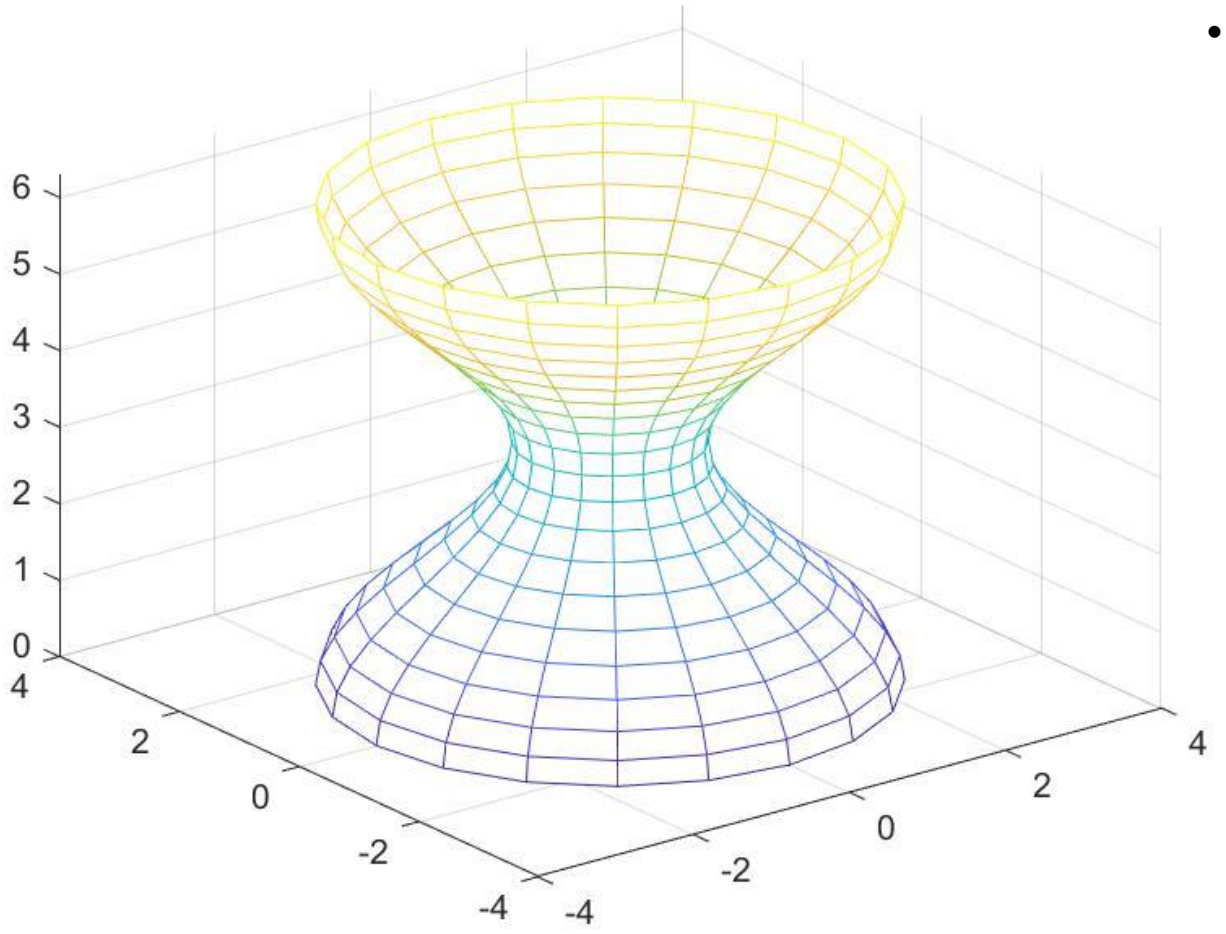
12주차

2차원 횡단면의 3차원 복원

12. 예제에 적용하기

Department of Mathematics
Gyeongsang National University
Group 3

적용한 예제



- 가운데가 오목한 원통과 10주차와 11주차에서 토의한 내용을 바탕으로 진행했습니다.

```
% 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
```

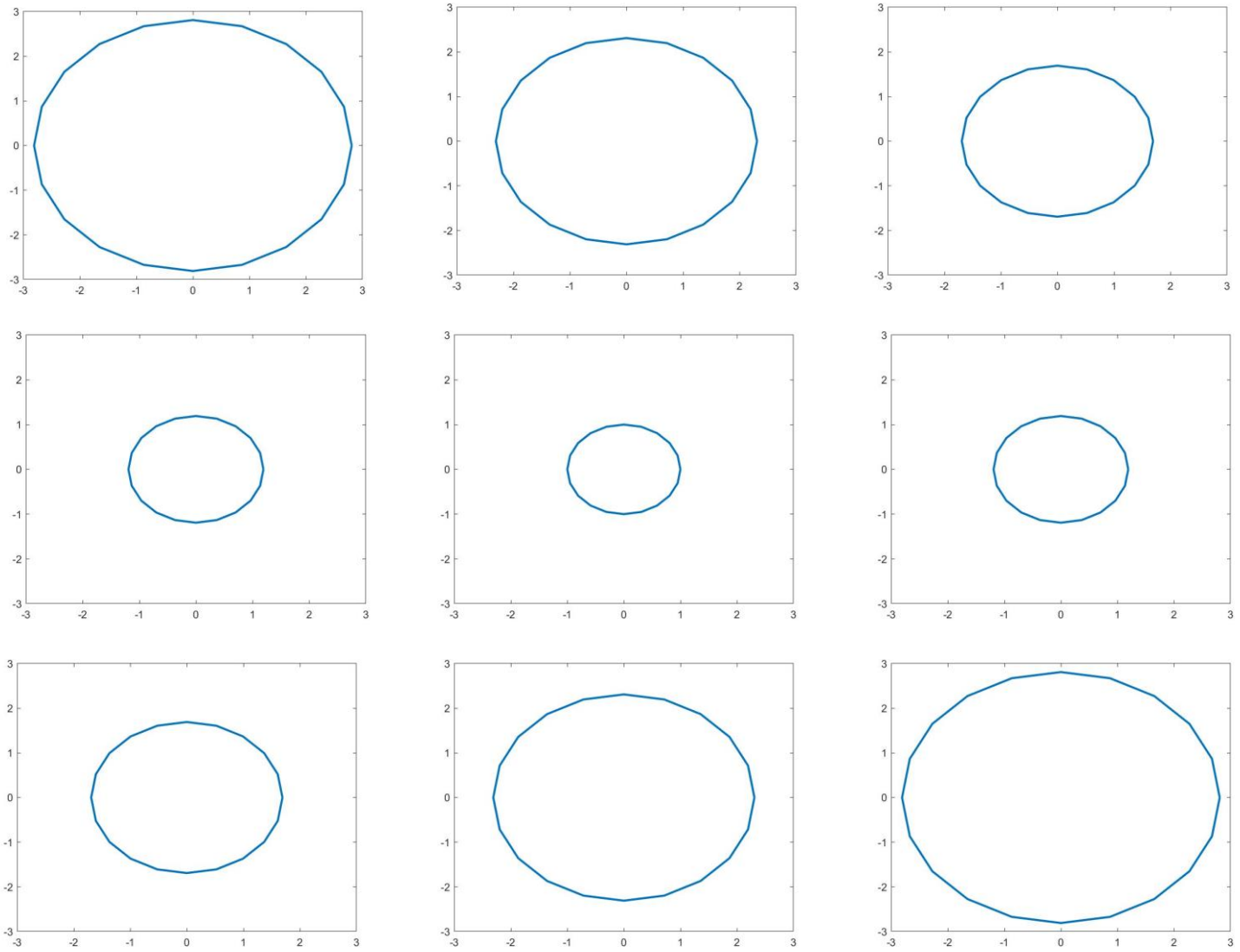
- 초기 값을 위와 같이 설정합니다.

```
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.

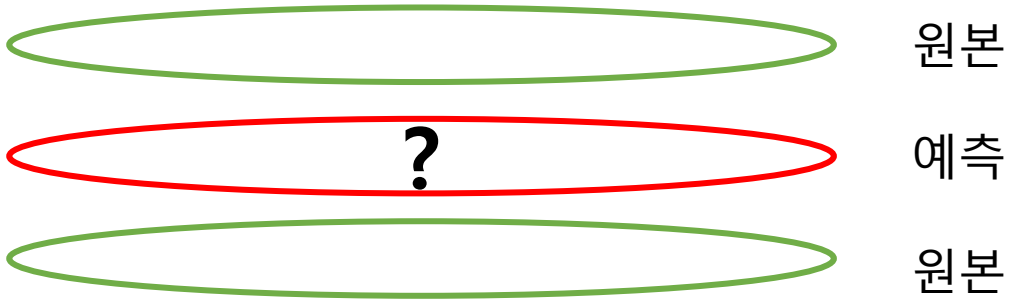


- 단면을 그린 결과 이다.

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.

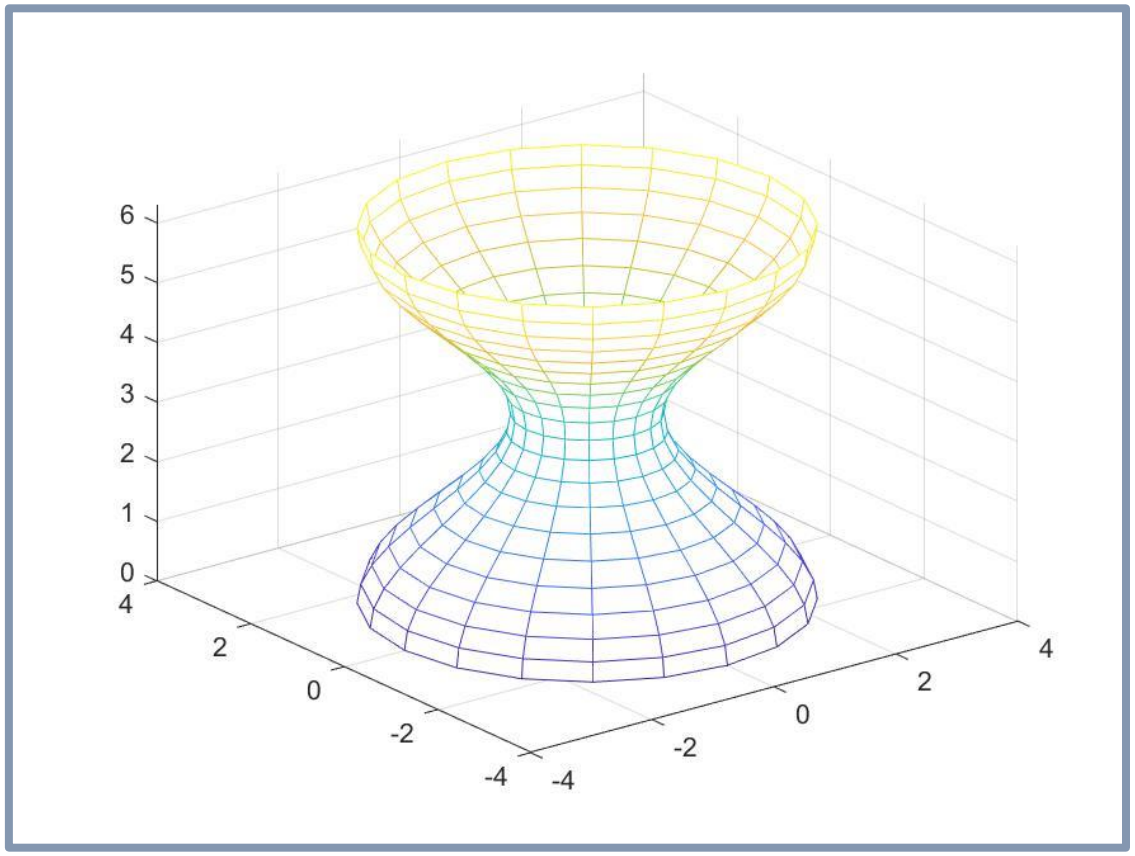
```
%%% A 행렬 설정하기 %%%
A(1,1)=2*(h(1)+h(2)); A(1,2)=h(2);
A(n-1,n-2)=h(n-2); A(n-1,n-1)=2*(h(n-2)+h(n-1));

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
```

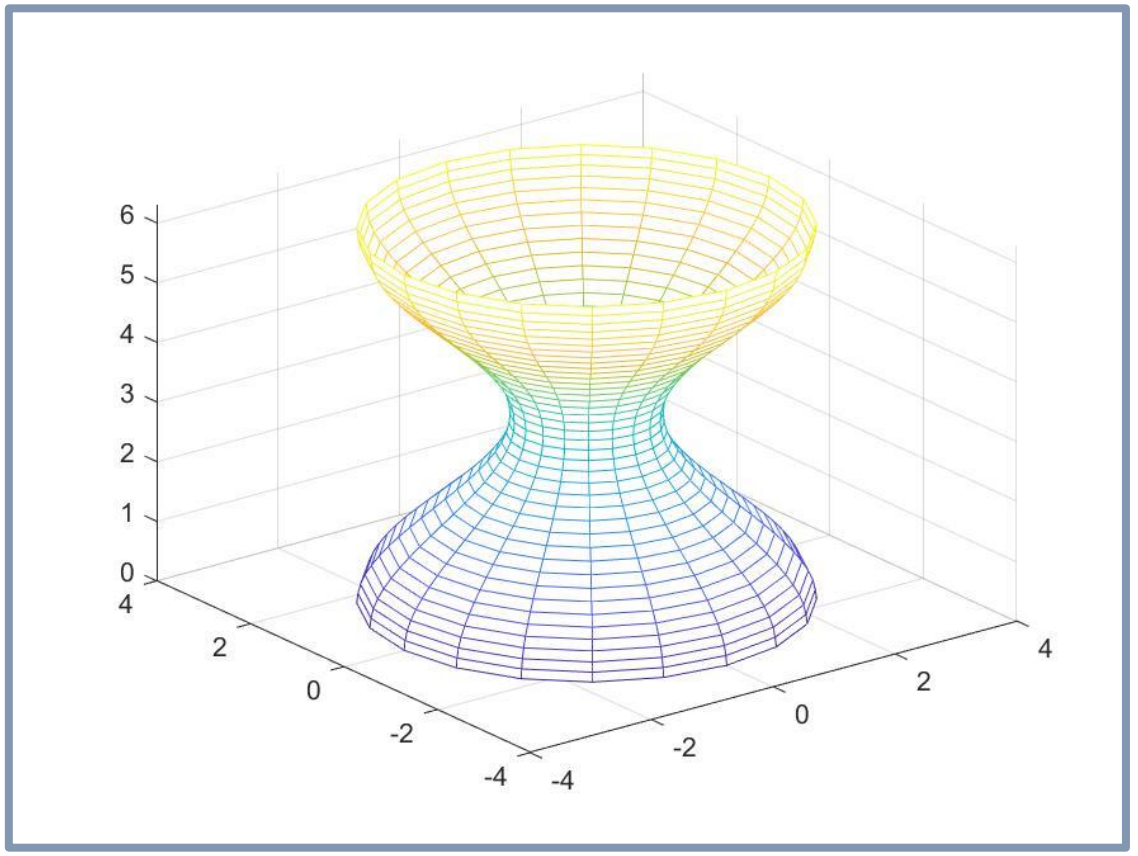
```
    % 보간한 좌표를 평면에 맞춰서 넣기
    for it=1:n-1
        x=t(it):h/2:t(it+1);
        y=a(it)+b(it).*(x-t(it))+c(it).*(x-t(it)).^2+d(it).*((x)-t(it)).^3;
        XX(2*it-1,ik)=y(1)*cos(the(ik)); YY(2*it-1,ik)=y(1)*sin(the(ik));
        XX(2*it,ik)=y(2)*cos(the(ik)); YY(2*it,ik)=y(2)*sin(the(ik));
     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

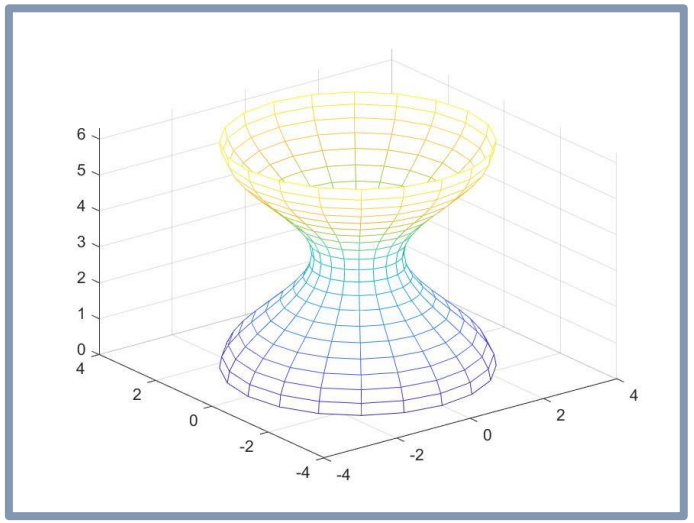
```
function C=least_polynomial(data,m)

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.

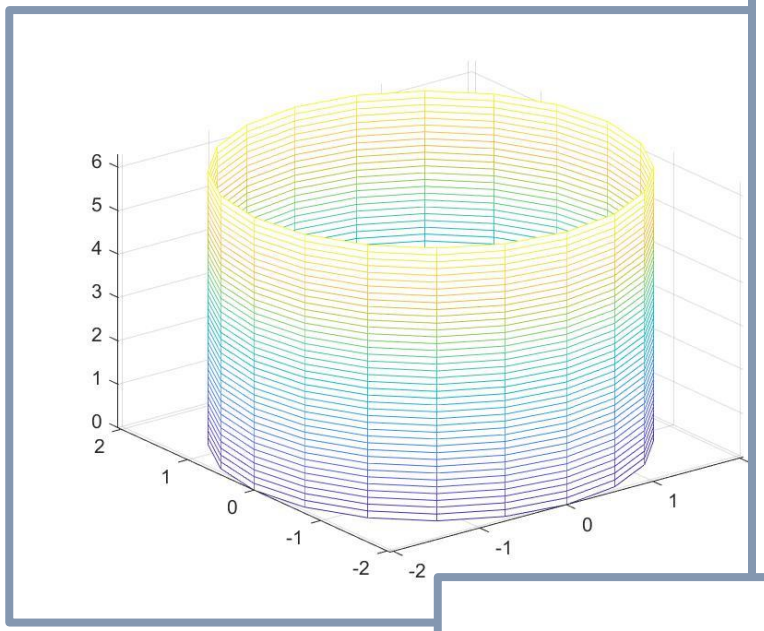
```
    for iz=1:m+1
        y=y+x.^(m+1-iz)*C(iz);          % 각 열에 최소자승법을 사용해 보간한 Z축과의 거리
    end
%     return
    XX(:,ik)=y*cos(the(ik));            % x 좌표 값 넣기
    YY(:,ik)=y*sin(the(ik));            % y 좌표 값 넣기
end

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

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

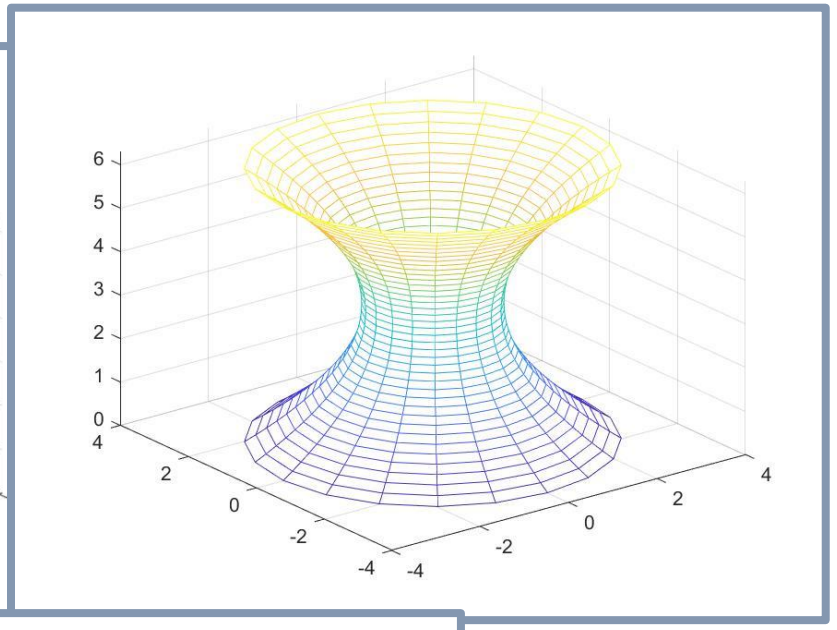
결과



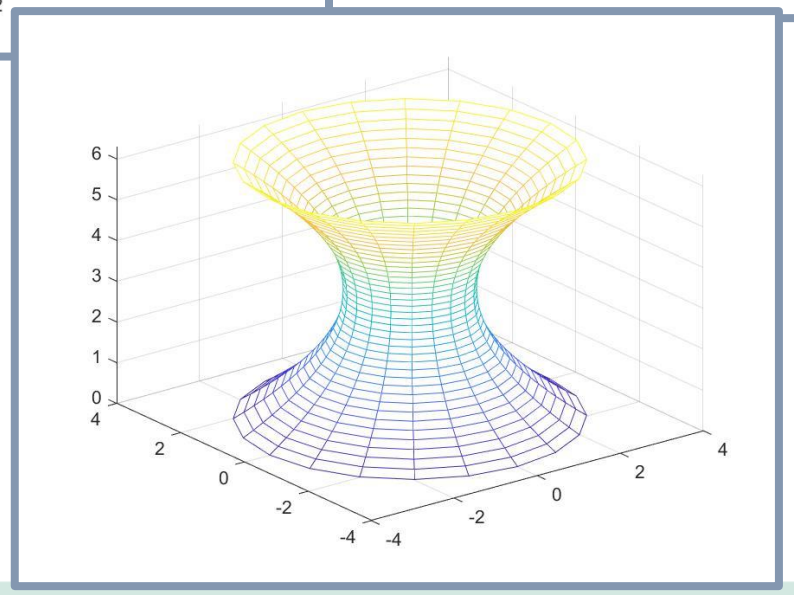
원본



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.

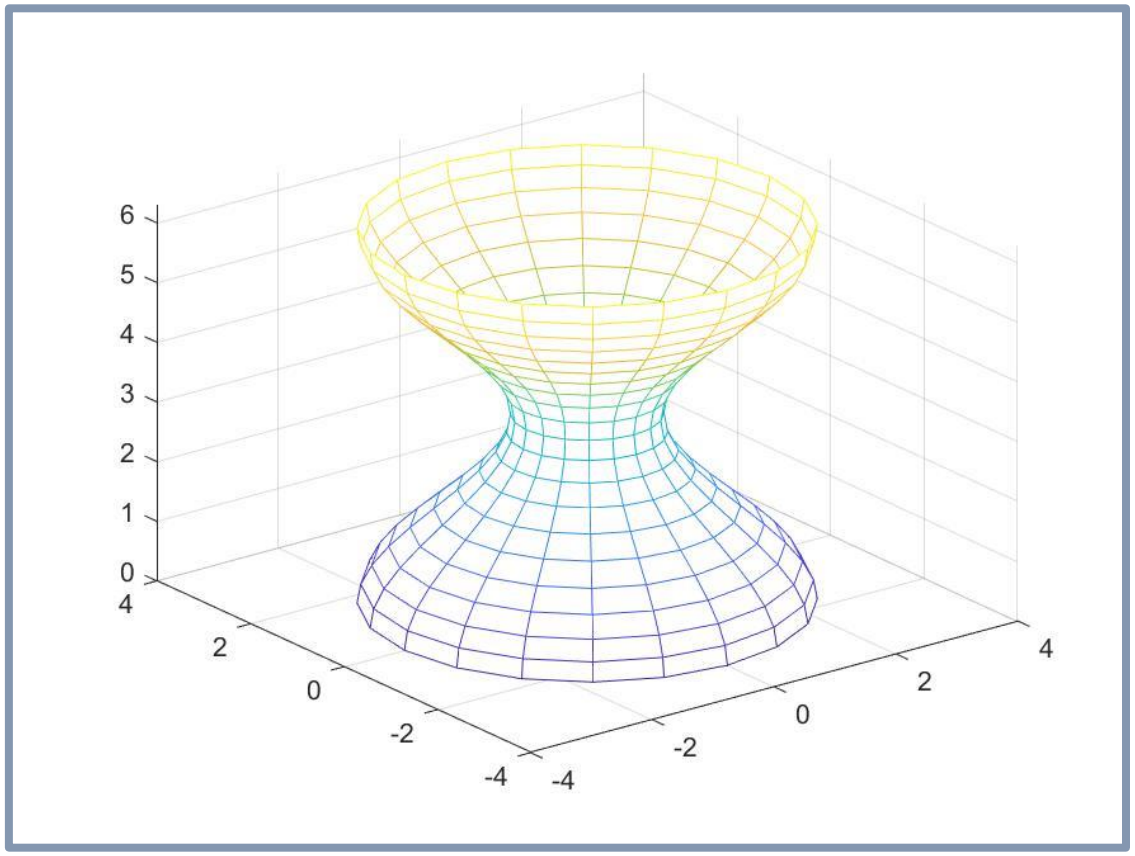
```
function C=newton_least1(data)

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.

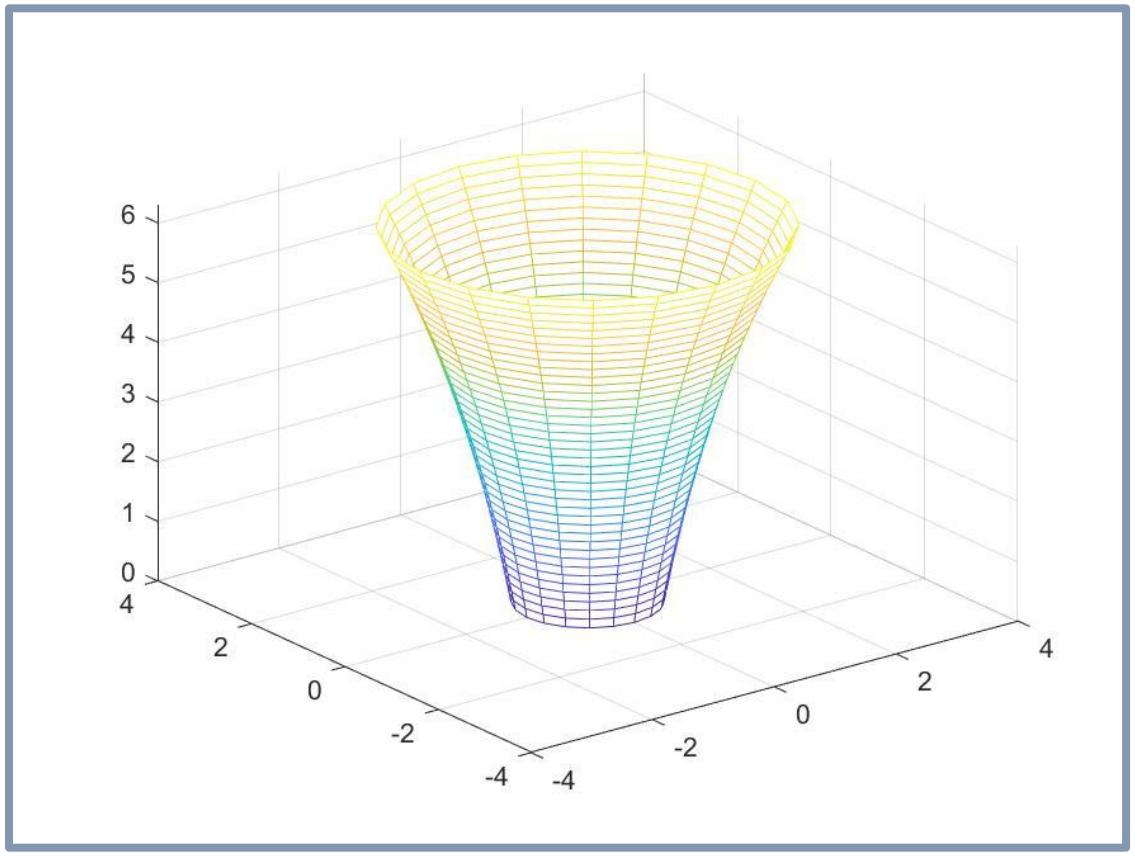
```
    y=exp(C(1)*x);          % 각 열에 최소자승법을 사용해 보간한 Z축과의 거리

    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

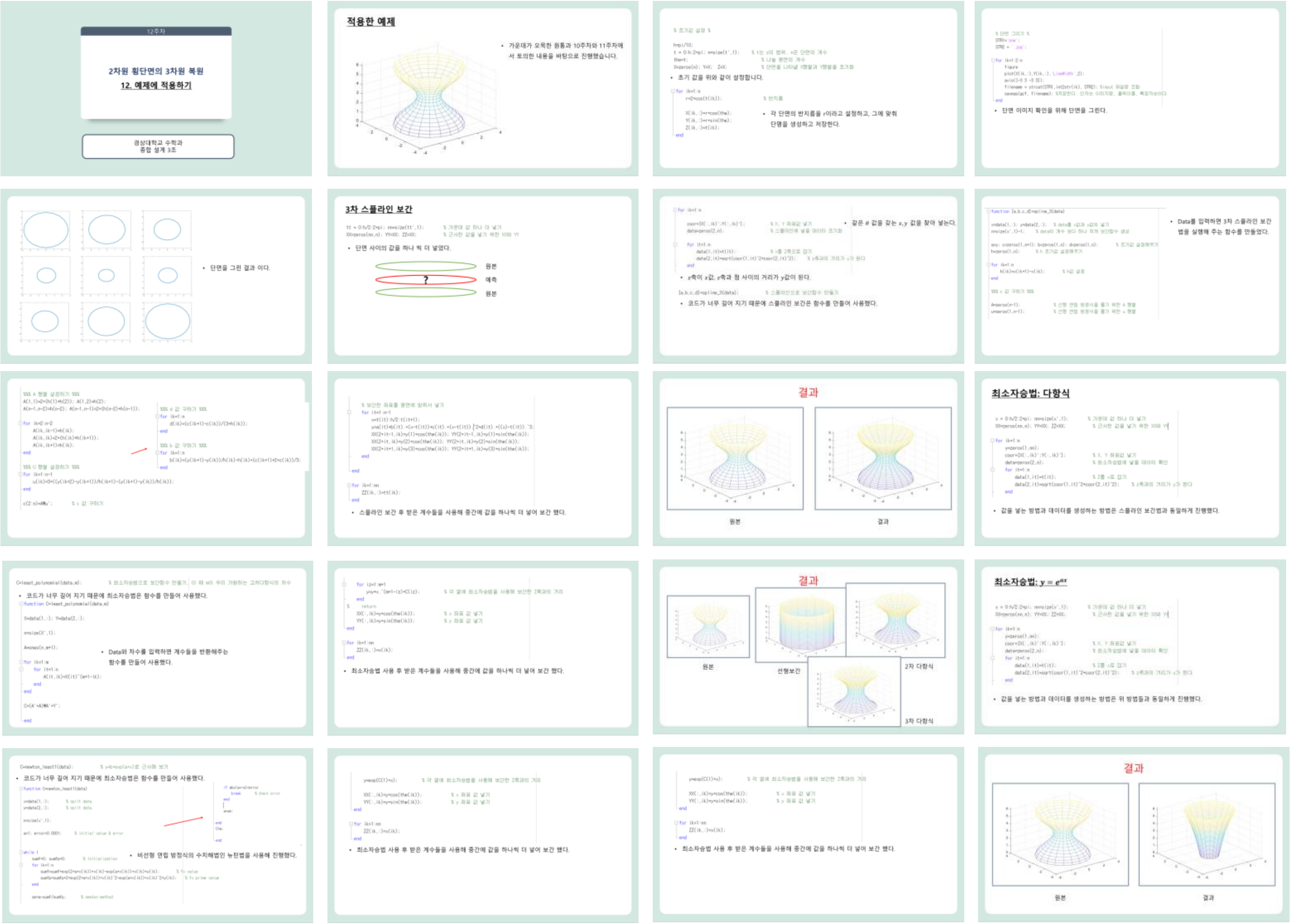


원본



결과

Thank you!


12주차
2차원 횡단면의 3차원 복원
12. 예제에 적용하기
경상대학교 수학과
종합 설계 3조
적용한 예제
가운데가 오목한 원통과 10주차와 11주차에서 토의한 내용을 바탕으로 진행했습니다.
초기 값을 위와 같이 설정합니다.
각 단면의 반지름을 r이라고 설정하고, 그에 맞춰 단면을 생성하고 저장한다.
단면 이미지 확인을 위해 단면을 그린다.
단면을 그린 결과 이다.
3차 스플라인 보간
단면 사이의 결과를 하나 씩 더 넣었다.
원본
예측
원본
같은 θ 값을 갖는 x,y 값을 찾아 넣는다.
코드가 너무 길어 지기 때문에 스플라인 보간은 함수를 만들어 사용했다.
Data를 입력하면 3차 스플라인 보간법을 실행해 주는 함수를 만들었다.
스플라인 보간 후 받은 계수들을 사용해 중간에 값을 하나씩 더 넣어 보간 했다.
결과
원본
결과
최소자승법: 다항식
값을 넣는 방법과 데이터를 생성하는 방법은 스플라인 보간법과 동일하게 진행했다.
코드가 너무 길어 지기 때문에 최소자승법은 함수를 만들어 사용했다.
Data와 차수를 입력하면 계수들을 반환해주는 함수를 만들어 사용했다.
최소자승법 사용 후 받은 계수들을 사용해 중간에 값을 하나씩 더 넣어 보간 했다.
결과
원본
선형보간
2차 다항식
3차 다항식
최소자승법: y = e^ax
값을 넣는 방법과 데이터를 생성하는 방법은 위 방법들과 동일하게 진행했다.
코드가 너무 길어 지기 때문에 최소자승법은 함수를 만들어 사용했다.
비선형 연립 방정식의 수치해법인 뉴턴법을 사용해 진행했다.
최소자승법 사용 후 받은 계수들을 사용해 중간에 값을 하나씩 더 넣어 보간 했다.
최소자승법 사용 후 받은 계수들을 사용해 중간에 값을 하나씩 더 넣어 보간 했다.
결과
원본
결과