

로봇학실험4

[OT & MATLAB 기초]

2023.09 1주차

I OT

1. 강의 일정 및 강의 소개

II MATLAB 기초

1. MATLAB 소개
2. 기초연산·함수 · 문법
3. MATLAB Graph

❖ 소개

- ◆ 이름 : 고은영
- ◆ Office : 누리관 318호
- ◆ E-mail : rhdmstdud282@gmail.com

오픈채팅방

- ◆ 학번+이름 으로 들어와 주세요!

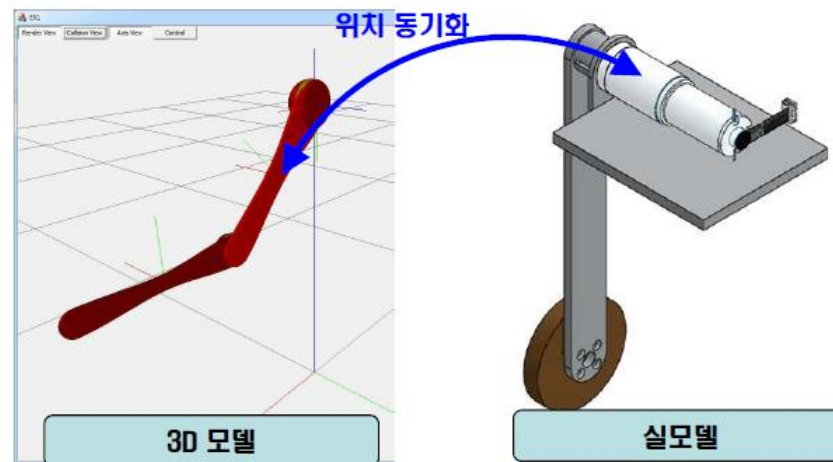


❖ 강의 일정

일정	수업내용
1주차	OT & MATLAB 기초
2주차	Motor Modeling
3주차	Geared Motor Modeling & Inertia
4주차	Motor 제어(PID, Cascade)
5주차	Motor 제어(PID, Cascade)
6주차	Robot Modeling
7주차	Kinematics (ODE)
8주차	중간고사
9주차	Motor Control (AVR)
10주차	Motor Control (AVR)
11주차	Window Programming (MFC)
12주차	MFC & ODE 연동
13주차	Communication (MFC <-> AVR)
14주차	시리얼 통신, 통신 패킷
15주차	Term Project 진행
16주차	Term Project 발표

❖ 강의 소개

◆ Term Project



◆ 성적 평가 기준

항목	비중
출석	10%
중간고사	30%
과제	20%
텀프	40% (보고서10%)

■ 출석 (10%)

출석	점수
출석	-0
지각	-1
결석	-2

■ 과제보고서 (60%)

Term Project (40%)

- 2인 1조
- 채점 기준은 추후 공개

■ 중간고사 (30%)

중간고사 시험 (30%)

Homework (20%) - 조별 과제X, 1인 1과제

Quality	정시	지각
상	10	6
중	6	3
하	3	0

I OT

1. 강의일정 및 강의 소개

II MATLAB 기초

1. MATLAB 소개
2. 기본연산·함수·문법
3. MATLAB Graph

□ MATLAB 기초

❖ MATLAB 소개

◆ 실험 목적

- MATLAB의 사용법을 익힌다.
- MATLAB에서 자주 사용되는 연산과 함수, 문법에 대해 살펴본다.
- MATLAB에서 그래프 그리는 방법을 익힌다.

◆ 실험 의미

- Simulation을 할 때 많이 사용하는 Tool에 익숙해진다.

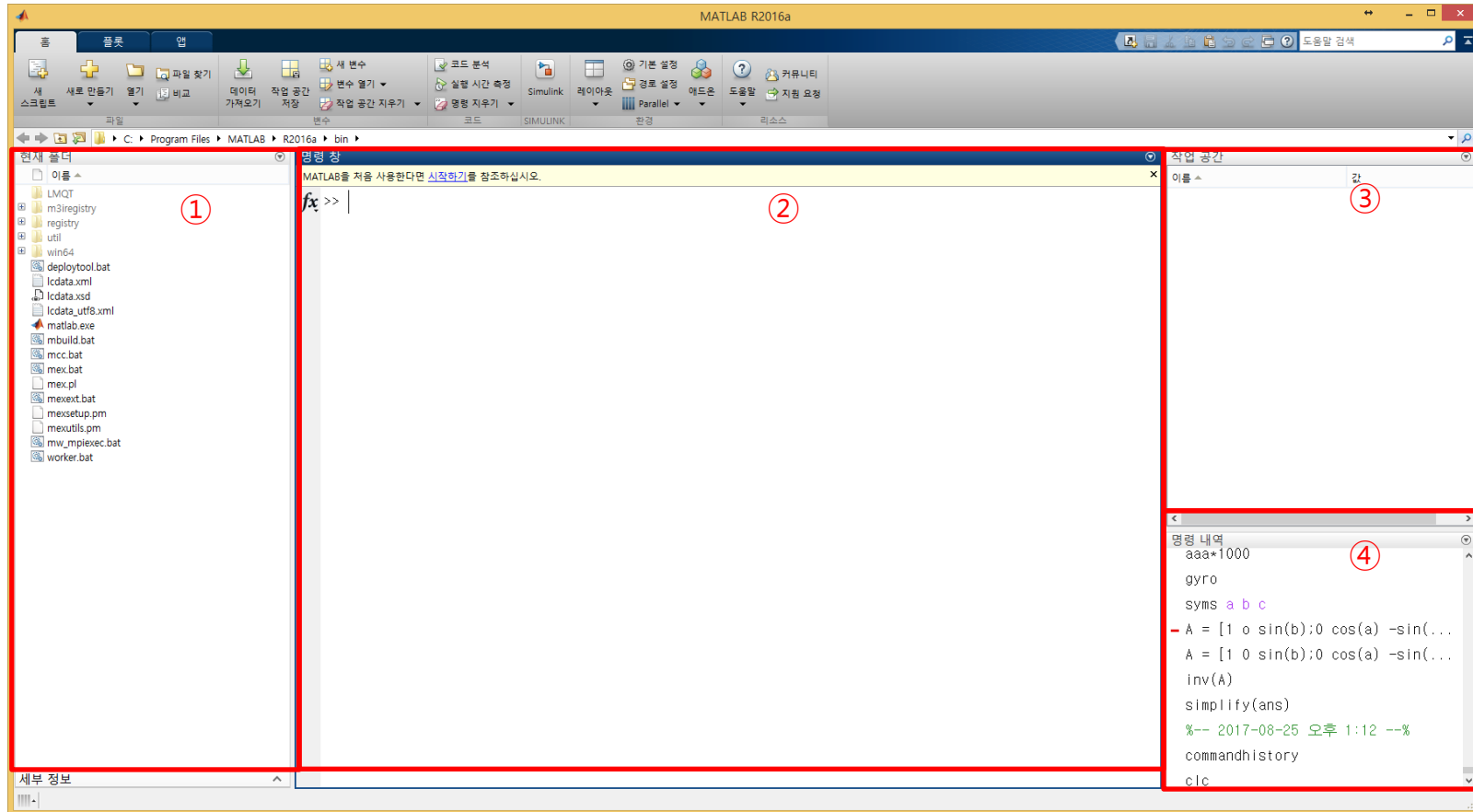
□ MATLAB 기초

◆ MATLAB

- 1977년 Cleve Moler는 선형 방정식 시스템을 풀고 행렬 계산을 수행할 수 있는 MATLAB(Matrix Laboratory) 소프트웨어를 처음 개발(80개 정도의 함수)
- 1980년대에서 1990년대에 걸쳐 FORTRAN 기반 행렬계산 응용환경으로부터 수천 개의 내장된 계산 및 그래프 함수들을 지닌 범용 계산 도구로 발전
- 1990년대 중반 The MathWorks社は GUI를 추가
- MATLAB의 장점
 - 고급언어를 이용한 신속하고 용이한 프로그램 작성
 - 행렬 연산
 - 그래프 및 애니메이션 등을 쉽게 구현
 - 함수를 사용하여 복잡한 신호처리를 기본 함수를 이용해 쉽게 구현

□ MATLAB 기초

◆ MATLAB 실행(2016a)



1. 현재 폴더(Current Directory Window)- 현재 작업 중인 파일 위치
2. 명령 창(Command Window)- MATLAB을 실행시키면 나타나는 메인 창, 기본연산•함수실행•변수확인 등 가능
3. 작업공간(Workspace Window)- 사용된 변수들에 대한 정보 제공
4. 명령내역(Command History Window)- 명령어 창에서 입력된 명령어들이 기록

□ MATLAB 기초

❖ 기본 연산·함수·문법

◆ 정수의 사칙연산

```
>> a = 1
a =
    1
>> b = 2
b =
    2
>> a + b
ans =
    3
>> a - b
ans =
   -1
>> a * b
ans =
    2
>> a/b
ans =
0.500000000000000000
```

◆ 행렬의 표현

→ 열은 여백과 쉼표로 구분

```
>> A = [1 2 3]
A =
    1    2    3
>> A = [1,2,3]
A =
    1    2    3
```

→ 행은 세미콜론으로 구분

```
>> B = [4; 5; 6]
B =
    4
    5
    6
```

□ MATLAB 기초

◆ 행렬의 사칙연산

→ 행렬의 덧셈

```
>> A + A
ans =
     2     4     6
```

→ 행렬의 뺄셈

```
>> B - B
ans =
     0
     0
     0
```

→ 행렬의 곱셈

```
>> A*B
ans =
    32
```

→ 행렬의 나눗셈(역행렬 연산)

```
>> C = [1; 2]
C =
     1
     2
```

```
>> D = [1 2; 2 1]
D =
     1     2
     2     1
```

```
>> inv(D)*C
```

```
ans =
```

$$C = DX$$

$$\begin{bmatrix} c_{11} \\ c_{21} \end{bmatrix} = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix}$$

$$\begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix}^{-1} \begin{bmatrix} c_{11} \\ c_{21} \end{bmatrix} = \begin{bmatrix} x_{11} \\ x_{21} \end{bmatrix}$$

□ MATLAB 기초

◆ MATLAB에서 이용되는 특수 변수 및 정수

특수변수 및 정수	기 능
ans	최신 출력
pi	3.1415926535897
i, j	허수 단위
inf	무한대
NaN	부정치

◆ 기본 명령어의 옵션

명령어	동작
clear all	저장된 변수 또는 함수 제거
clc	Clear Command Window
format long	15자리로 조정된 고정소수점 표현
format compact	여분의 라인을 생략

□ MATLAB 기초

◆ 자주 사용되는 MATLAB 연산자 및 기호

연산자 및 기호	의미
+	더하기(스칼라, 벡터, 행렬)
-	빼기(스칼라, 벡터, 행렬)
*	곱하기(스칼라, 벡터, 행렬)
/	나누기(스칼라)
^	지수(스칼라, 정방행렬)
:	사이 값들을 지정
'	전치(실수벡터, 행렬)
...	줄이음
%	코멘트(주석)
==	논리적 equals
	논리적 or
&	논리적 and
~=	논리적 not

□ MATLAB 기초

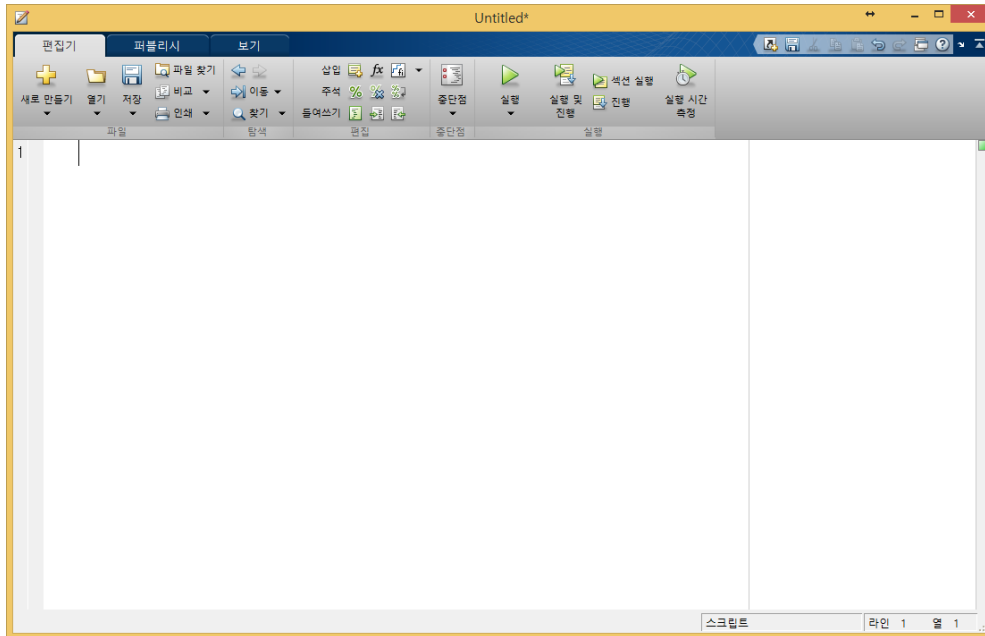
◆ 기타 기본 함수

```
>> exp(1)
ans =
    2.718281828459046
>> sqrt(2)
ans =
    1.414213562373095
>> eig(D)
ans =
    -1
     3
>> 1:0.1:3
ans =
Columns 1 through 3
    1.000000000000000    1.100000000000000    1.200000000000000
Columns 4 through 6
    1.300000000000000    1.400000000000000    1.500000000000000
Columns 7 through 9
    1.600000000000000    1.700000000000000    1.800000000000000
Columns 10 through 12
    1.900000000000000    2.000000000000000    2.100000000000000
Columns 13 through 15
    2.200000000000000    2.300000000000000    2.400000000000000
Columns 16 through 18
    2.500000000000000    2.600000000000000    2.700000000000000
Columns 19 through 21
    2.800000000000000    2.900000000000000    3.000000000000000
```

```
>> eye(3)
ans =
     1     0     0
     0     1     0
     0     0     1
>> eye(3,3)
ans =
     1     0     0
     0     1     0
     0     0     1
>> zeros(2,3)
ans =
     0     0     0
     0     0     0
>> ones(3,2)
ans =
     1     1
     1     1
     1     1
```

□ MATLAB 기초

◆ Editor(MATLAB 코드 작성)



MATLAB 화면 왼쪽 상단 file → new → script

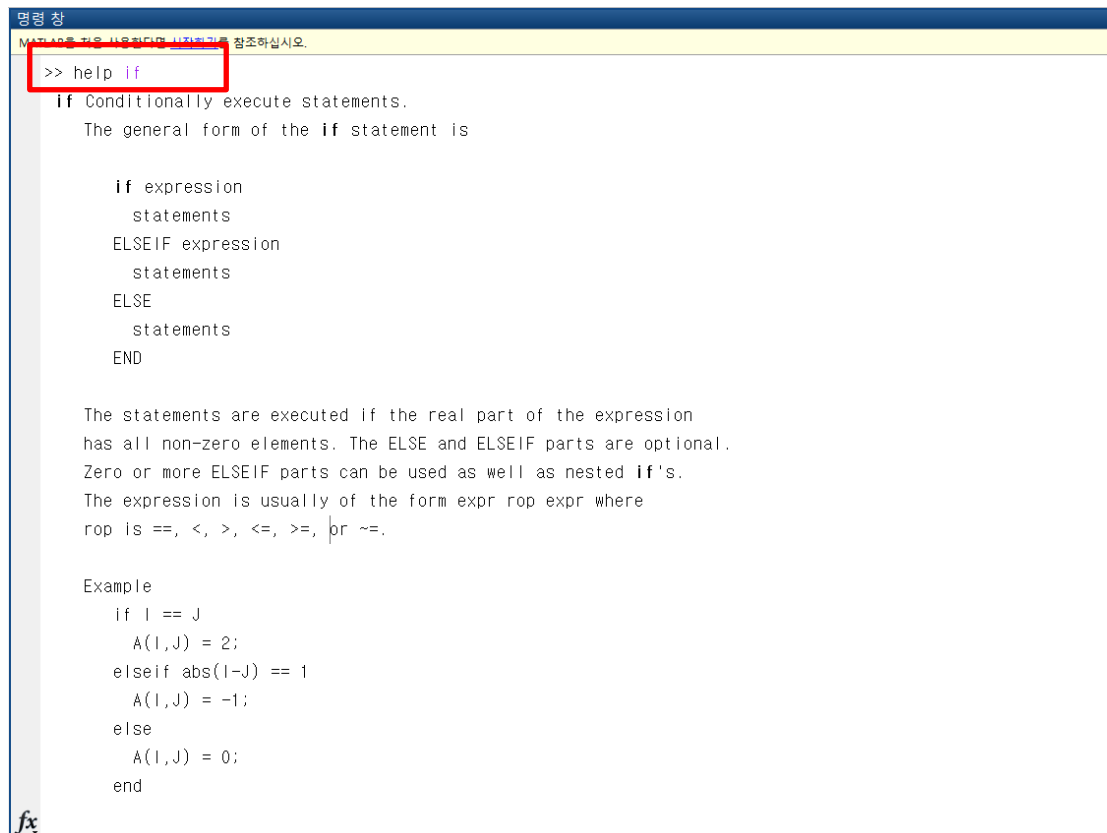
일반 저장 시 m-file(Matlab 코드 파일)이 현재폴더(Current Directory)에 저장

□ MATLAB 기초

◆ 함수

함수이름을 알면 help 명령어를 통해 함수 사용법 검색 가능

Ex) if문



```

명령 창
MATLAB을 처음 사용한다면 시작 가이드를 참조하십시오.
>> help if
if Conditionally execute statements.
The general form of the if statement is

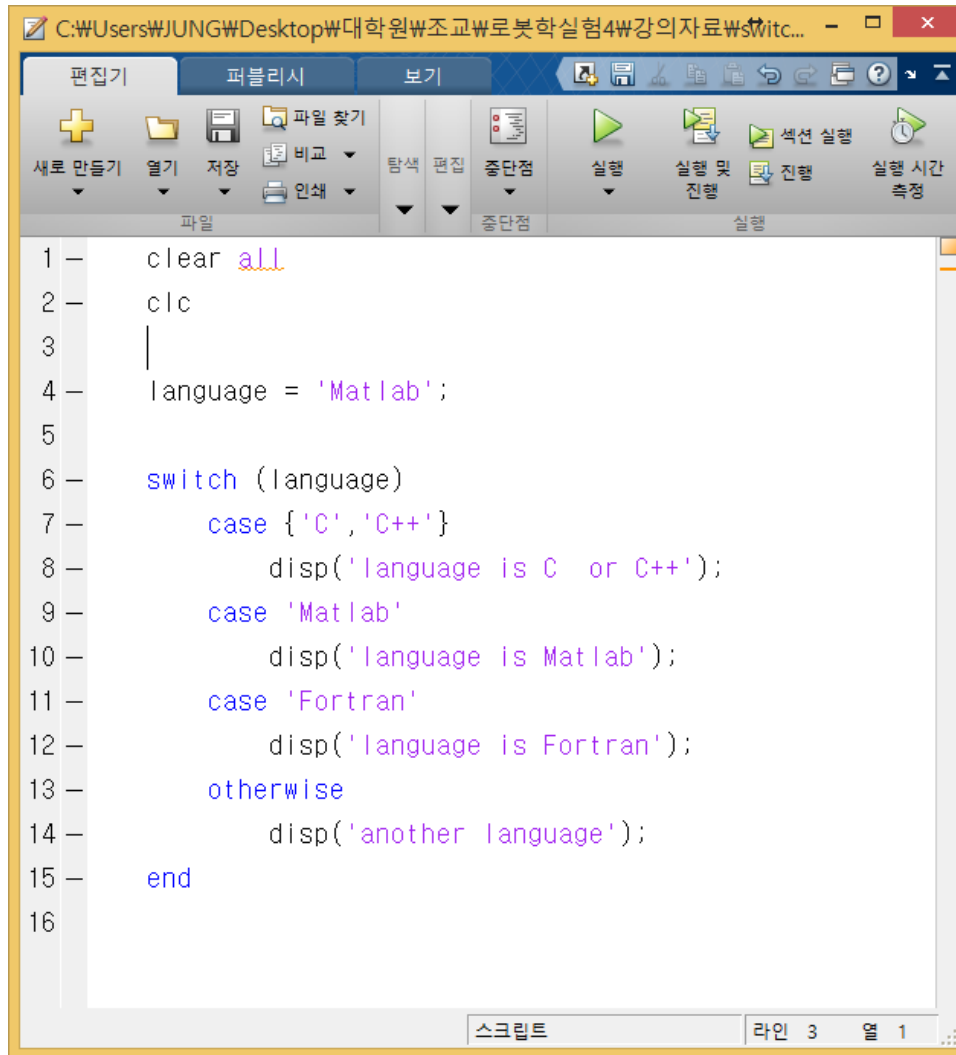
    if expression
        statements
    elseif expression
        statements
    ELSE
        statements
    END

The statements are executed if the real part of the expression
has all non-zero elements. The ELSE and ELSEIF parts are optional.
Zero or more ELSEIF parts can be used as well as nested if's.
The expression is usually of the form expr rop expr where
rop is ==, <, >, <=, >=, or ~=.

Example
    if I == J
        A(I,J) = 2;
    elseif abs(I-J) == 1
        A(I,J) = -1;
    else
        A(I,J) = 0;
    end
  
```


□ MATLAB 기초

◆ switch



```

1 - clear all
2 - clc
3 -
4 - language = 'Matlab';
5 -
6 - switch (language)
7 -     case {'C', 'C++'}
8 -         disp('language is C or C++');
9 -     case 'Matlab'
10 -         disp('language is Matlab');
11 -     case 'Fortran'
12 -         disp('language is Fortran');
13 -     otherwise
14 -         disp('another language');
15 - end
16

```

switch 변수 또는 식

case 값1

명령어 문장

case 값2

명령어 문장

otherwise

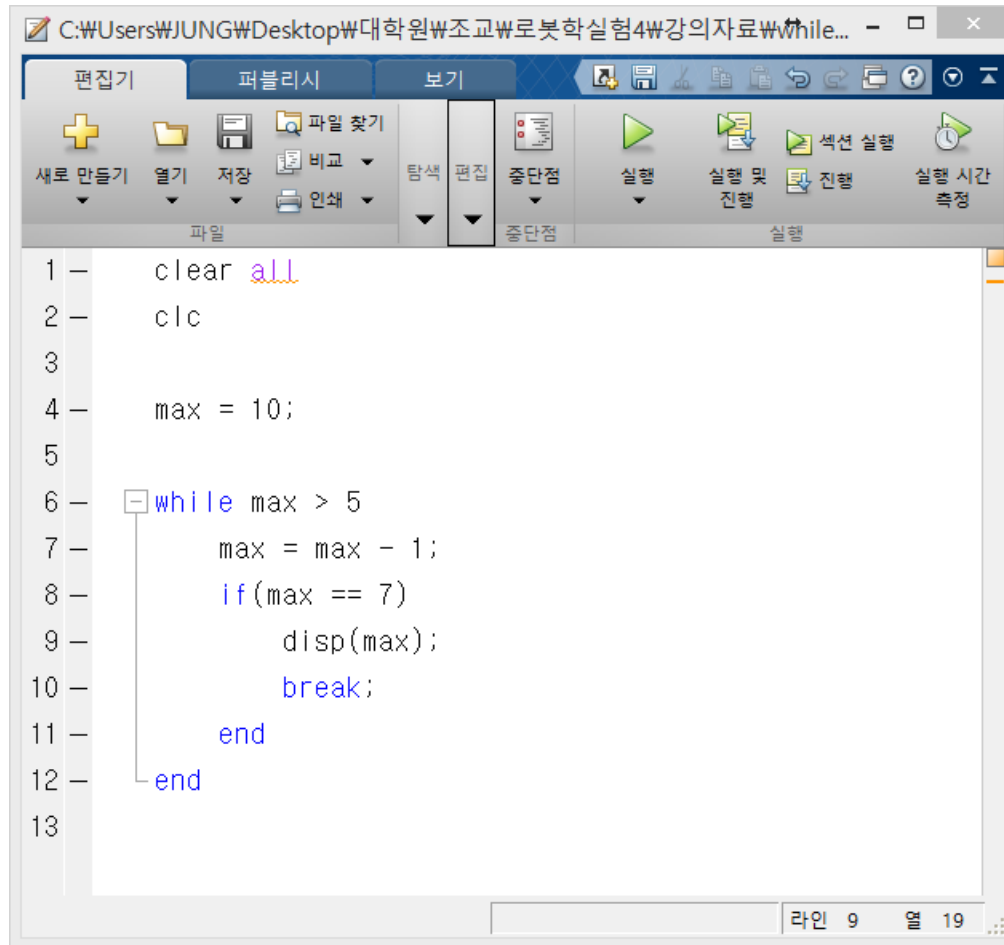
명령어 문장

end

disp(): 문자열을 출력하는 함수

□ MATLAB 기초

◆ while



```

1 - clear all
2 - clc
3
4 - max = 10;
5
6 - while max > 5
7 -     max = max - 1;
8 -     if(max == 7)
9 -         disp(max);
10 -        break;
11 -     end
12 - end
13

```

라인 9 열 19

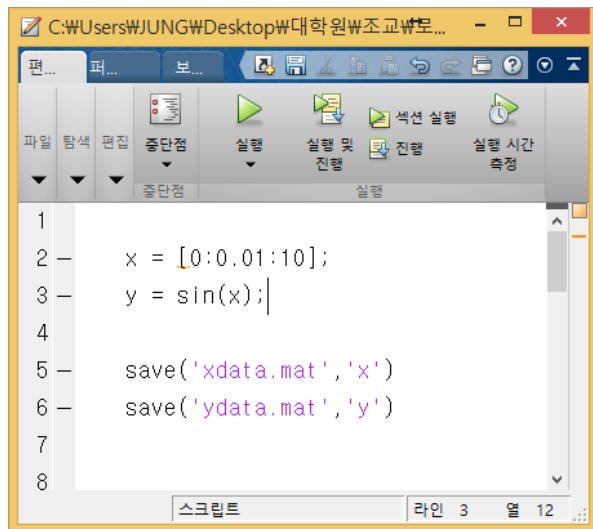
while 논리적인 조건
명령어 문장
end

□ MATLAB 기초

◆ 변수 저장 및 불러오기(save,load 명령어)

• Save 명령어

작업공간(Workspace)의 변수를 mat 파일로 현재폴더(Current Directory)에 저장



x는 0~10까지 0.01 간격으로 1000개의 값을 갖는 변수
y는 x에 대응되는 1000개 sin 값을 갖는 변수

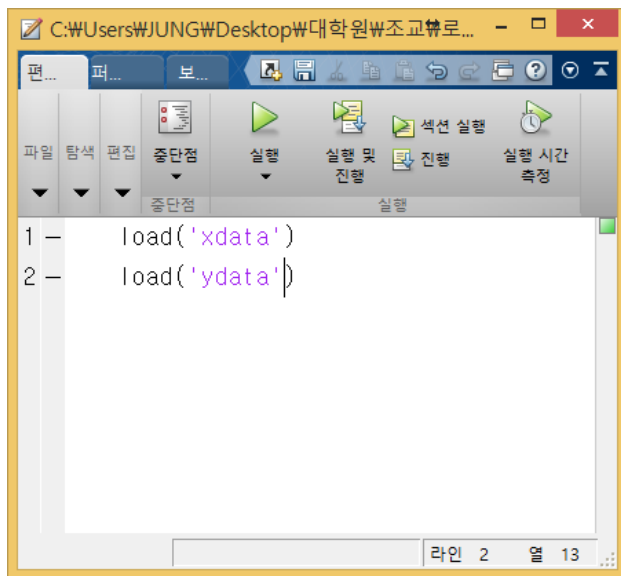
save('mat 파일 명','변수 명')

□ MATLAB 기초

◆ 변수 저장 및 불러오기(save,load 명령어)

- load 명령어

현재폴더(Current Directory)에 있는 mat 파일에서 변수를 불러와 작업공간(Workspace)에 추가



xdata.mat , ydata.mat 에 저장된 변수 x, y 를
작업공간(Workspace)에 추가

load('mat 파일 명')

-주의-

현재폴더(Current Directory)에 있는 파일을 불러오기 때문에 다른 폴더에 있는
mat 파일을 불러오려면 현재 폴더의 경로를 변경해야 한다.

또는 load함수에서 직접 경로를 입력해 불러온다.

Ex) load('C:\WMATLAB\xdata.mat')

□ MATLAB 기초

❖ MATLAB Graph

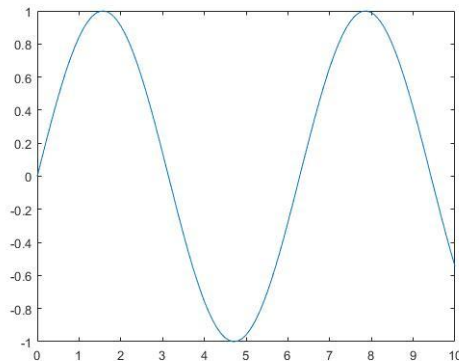
◆ plot 명령어

- plot (X1, Y1, 'opt1', X2, Y2, 'opt2',...)
 - 그래프를 그리는 명령어
 - Xn, Yn : x, y축 데이터
 - opt : 그래프 모양 관련 옵션
- axis ([X0 Xf Y0 Yf])
 - 그래프의 범위를 설정하는 명령어
 - X0, Y0 : 그래프에서의 x,y축 최소값
 - Xf, Yf : 그래프에서의 x,y축 최대값
- xlabel('Xd'); ylabel('Yd')
 - X, Y축에 설명을 추가하는 명령어
 - Xd, Yd : 그래프에서 x, y축에 대한 설명
- title('그래프 이름')
 - 그래프의 이름을 정해주는 명령어
- legend('F1', 'F2',...)
 - 각 함수에 대한 설명을 덧붙이는 명령어
 - Fn : 각 함수에 대한 설명
- hold
 - hold on : 현재 그래프 유지
 - hold off : 현재 그래프 해제
- grid
 - grid on : 점선 그리기
 - Tick : 점선 간격
- figure
 - figure(숫자): 'figure(숫자)'의 이름을 가진 새로운 창 생성

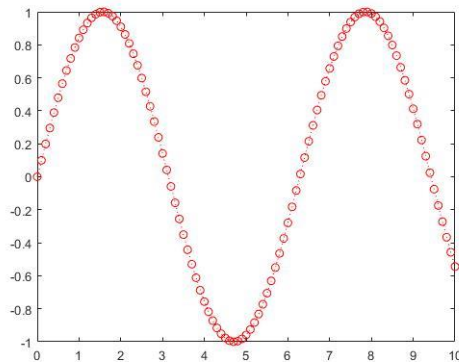
□ MATLAB 기초

◆ OPT: 그래프 모양 옵션

plot(x,y)



plot(x,y,'or')



Point Type	Indicator
point	.
circle	o
x-mark	x
plus	+
star	*
square	s
diamond	d
triangle down	v
triangle up	^
triangle left	<
triangle right	>
pentagram	p
hexagram	h

Line Type	Indicator
solid	-
dotted	:
dash-dot	-.
dashed	--

Color	Indicator
blue	b
green	g
red	r
cyan	c
magenta	m
yellow	y
black	k

□ MATLAB 기초

- figure 이용 여러 개의 그래프 창 만들기

```
x = [0:0.01:100];
```

```
y1 = sin(x);
```

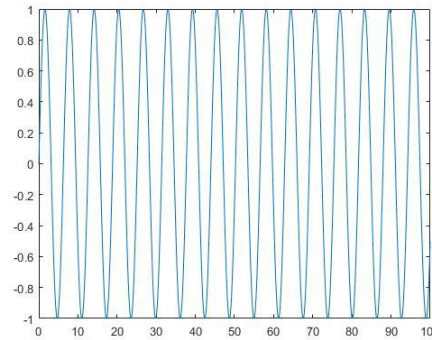
$$y_2 = \sin(2x);$$

```
figure(1)
```

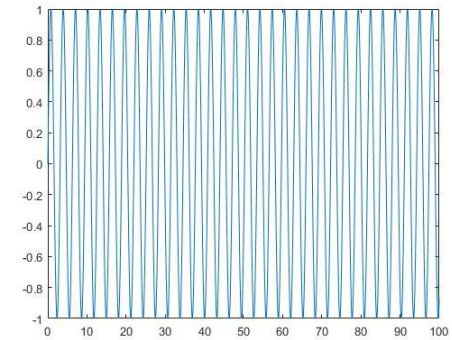
```
plot(x,y1);
```

figure(2)

```
plot(x,y2);
```



figure(1)



figure(2)

- figure 위치, 크기 옵션

```
x = [0:0.01:100];
```

```
y1 = sin(x);
```

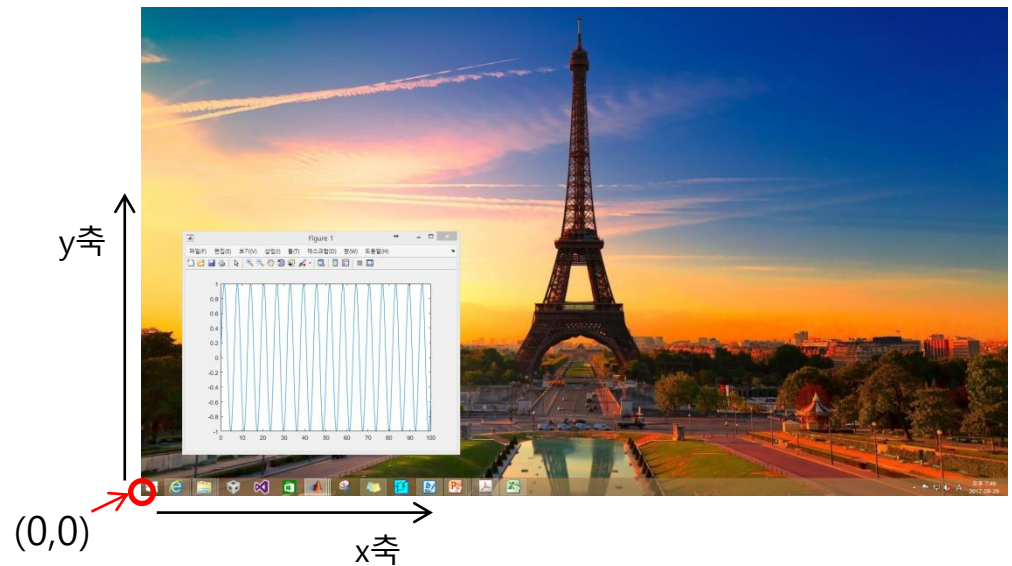
```
figure('units','pixels','pos',[100 100 600 400])
```

```
plot(x,y1);
```

구성 단위 결정

위치 및 크기

[x축 위치 y축 위치 가로 크기 세로 크기]



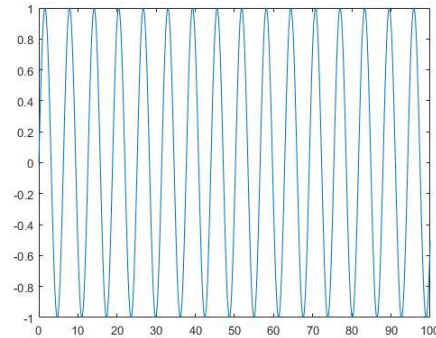
□ MATLAB 기초

• LineWidth : 그래프 두께 옵션

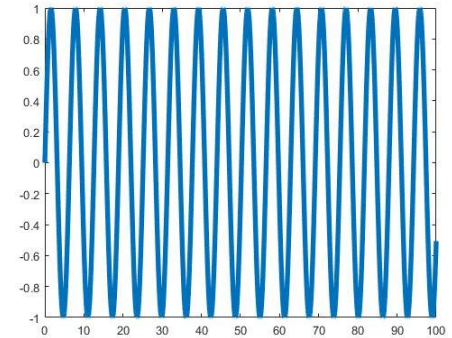
```
x = [0:0.01:100];
y = sin(x);

figure(1)
plot(x,y)

figure(2)
plot(x,y, 'LineWidth', 4)
```



figure(1)



figure(2)

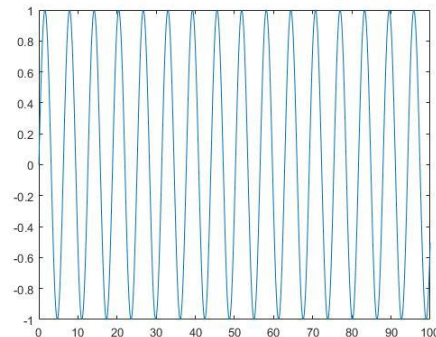
• axis : 그래프 범위 옵션

```
x = [0:0.01:100];
y = sin(x);

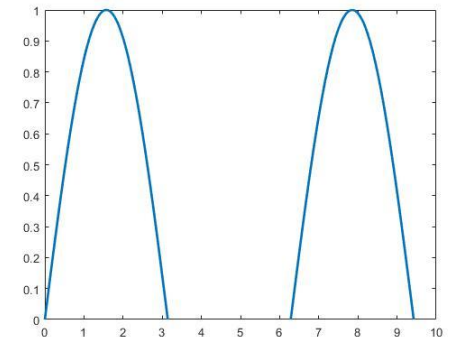
Xmin = 0;
Ymin = 0;
Xmax = 10;
Ymax = 1;

figure(1)
plot(x,y)

figure(2)
plot(x,y, 'LineWidth', 2)
axis([Xmin Xmax Ymin Ymax])
```



figure(1)



figure(2)

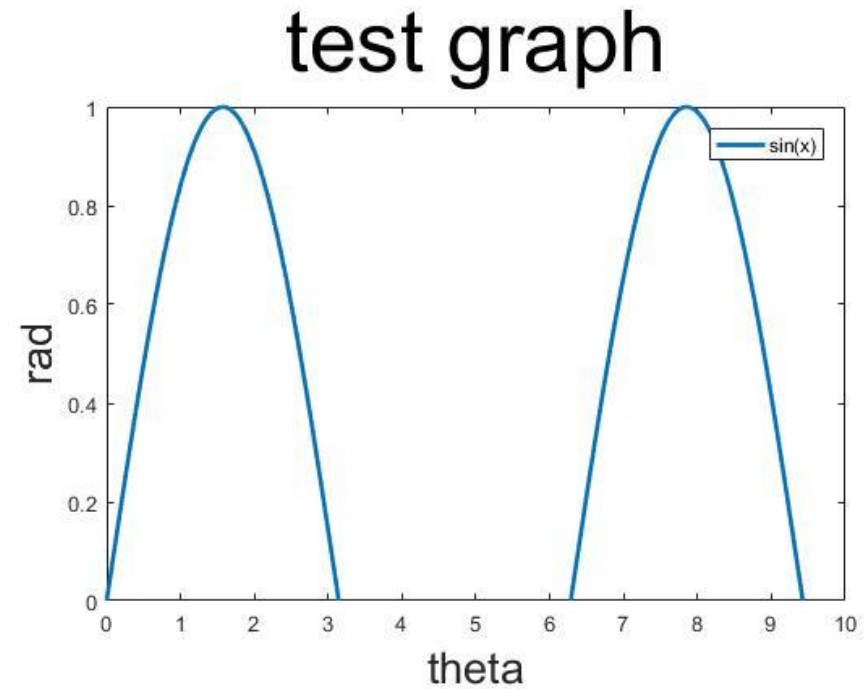
□ MATLAB 기초

- `xlabel('Xd');` `ylabel('Yd')` : 축 이름
- `legend` : 함수 설명
- `title` : 그래프의 이름

```
x = [0:0.01:100];
y = sin(x);

Xmin = 0;
Ymin = 0;
Xmax = 10;
Ymax = 1;

plot(x,y, 'LineWidth',2)
axis([Xmin Xmax Ymin Ymax])
xlabel('theta','fontsize',20)    % x 축 명
ylabel('rad','fontsize',20)    % y 축 명
legend('sin(x)');              % 함수 설명
title('test graph','fontsize',40) % 그래프 명
```



□ MATLAB 기초

• 하나의 figure에 다수의 그래프 그리기

▪ Case1: plot함수 1개 사용

```
x = [0:0.05:10];
y1 = sin(x);
y2 = sin(2*x);

Xmin = 0;
Ymin = -1;
Xmax = 10;
Ymax = 1;

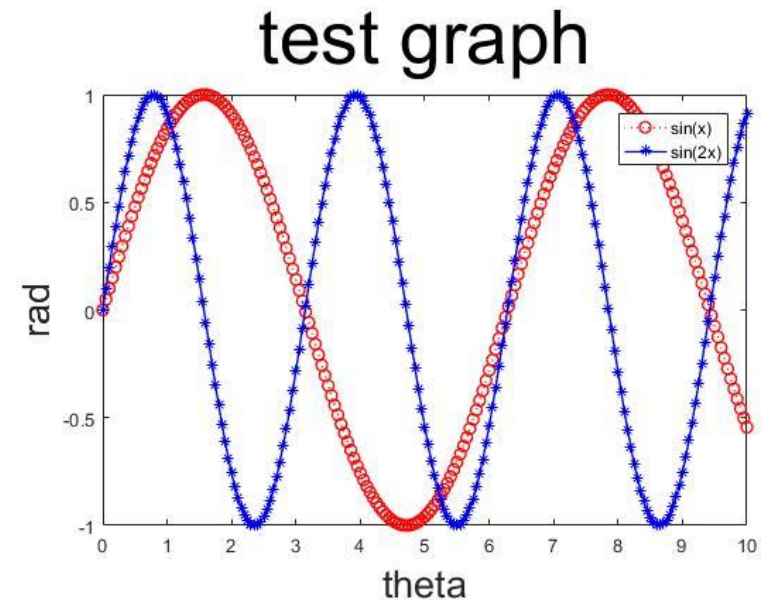
plot(x,y1,'or:',x,y2,'*b-','LineWidth',1)
axis([Xmin Xmax Ymin Ymax])
xlabel('theta','fontSize',20) % x 축 명
ylabel('rad','fontSize',20) % y 축 명
legend('sin(x)','sin(2x)'); % 함수 설명
title('test graph','fontSize',40) % 그래프 명
```

▪ Case2: plot함수 2개, hold on 사용

```
x = [0:0.05:10];
y1 = sin(x);
y2 = sin(2*x);

Xmin = 0;
Ymin = -1;
Xmax = 10;
Ymax = 1;

plot(x,y1,'or:', 'LineWidth',1)
hold on
plot(x,y2,'*b-','LineWidth',1)
axis([Xmin Xmax Ymin Ymax])
xlabel('theta','fontSize',20) % x 축 명
ylabel('rad','fontSize',20) % y 축 명
legend('sin(x)','sin(2x)'); % 함수 설명
title('test graph','fontSize',40) % 그래프 명
```



□ MATLAB 기초

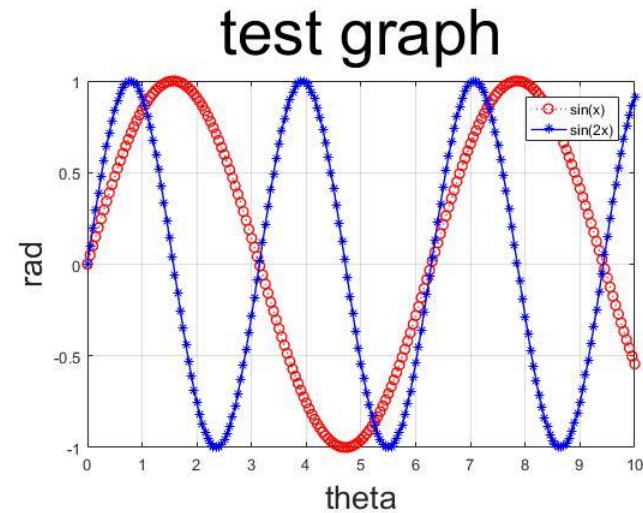
• grid : 점선 그리기

▪ Case1

```
x = [0:0.05:10];
y1 = sin(x);
y2 = sin(2*x);

Xmin = 0; Ymin = -1;
Xmax = 10; Ymax = 1;

plot(x,y1,'or:', 'LineWidth',1); hold on
plot(x,y2,'*b-', 'LineWidth',1)
axis([Xmin Xmax Ymin Ymax])
xlabel('theta','fontsize',20)      % x 축 명
ylabel('rad','fontsize',20)      % y 축 명
legend('sin(x)', 'sin(2x)');      % 함수 설명
title('test graph','fontsize',40) % 그래프 명
grid on
```

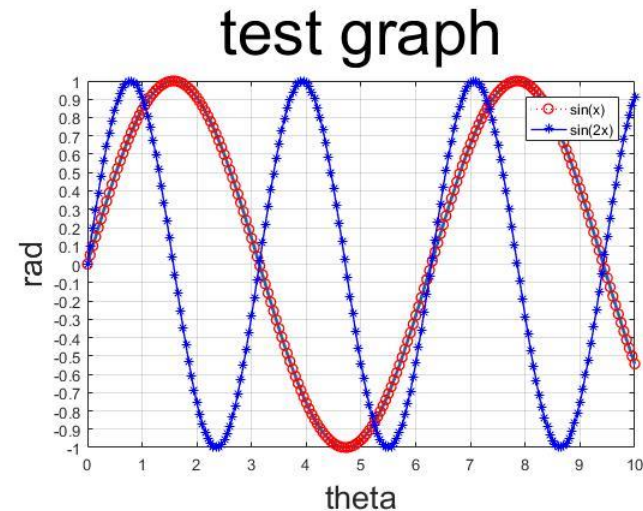


▪ Case2: 점선 간격 설정

```
x = [0:0.05:10];
y1 = sin(x);
y2 = sin(2*x);

Xmin = 0; Ymin = -1;
Xmax = 10; Ymax = 1;
XTick = 1; YTick = 0.1;

plot(x,y1,'or:', 'LineWidth',1); hold on
plot(x,y2,'*b-', 'LineWidth',1)
axis([Xmin Xmax Ymin Ymax])
xlabel('theta','fontsize',20)      % x 축 명
ylabel('rad','fontsize',20)      % y 축 명
legend('sin(x)', 'sin(2x)');      % 함수 설명
title('test graph','fontsize',40) % 그래프 명
grid on;
set(gca, 'XTick', [Xmin:XTick:Xmax]);
set(gca, 'YTick', [Ymin:YTick:Ymax]);
```



□ MATLAB 기초

• subplot : 다수의 그래프 그리기

-subplot(rows,columns,location)

```
x = [0:0.05:10];
```

```
y1 = sin(x);
```

```
y2 = sin(2*x);
```

```
Xmin = 0; Ymin = -1;
```

```
Xmax = 10; Ymax = 1;
```

```
subplot(2,1,1)
```

```
plot(x,y1,'or','LineWidth',1)
```

```
axis([Xmin Xmax Ymin Ymax])
```

```
xlabel('theta','fontsize',20) % x 축 명
```

```
ylabel('rad','fontsize',20) % y 축 명
```

```
legend('sin(x)'); % 함수 설명
```

```
title('test graph','fontsize',40) % 그래프 명
```

```
subplot(2,1,2)
```

```
plot(x,y2,'*b-','LineWidth',1)
```

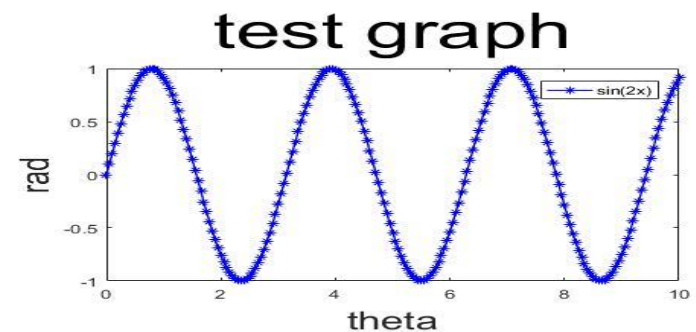
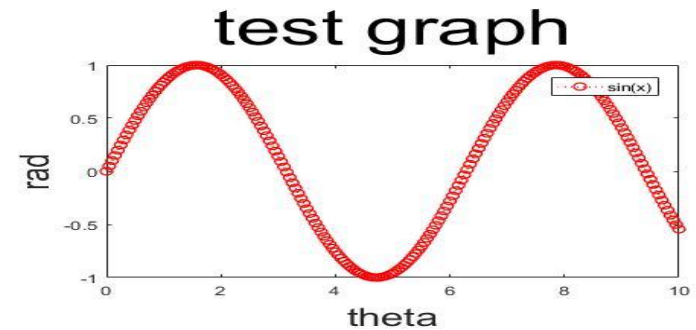
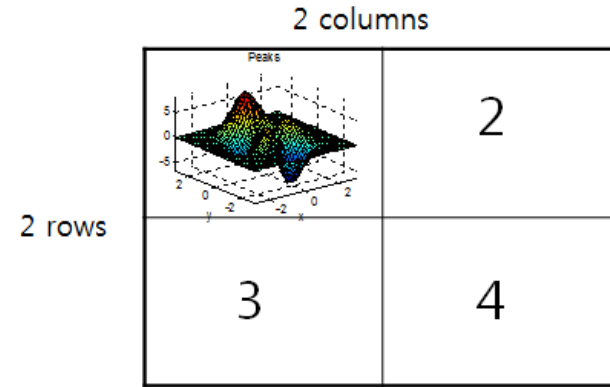
```
axis([Xmin Xmax Ymin Ymax])
```

```
xlabel('theta','fontsize',20) % x 축 명
```

```
ylabel('rad','fontsize',20) % y 축 명
```

```
legend('sin(2x)'); % 함수 설명
```

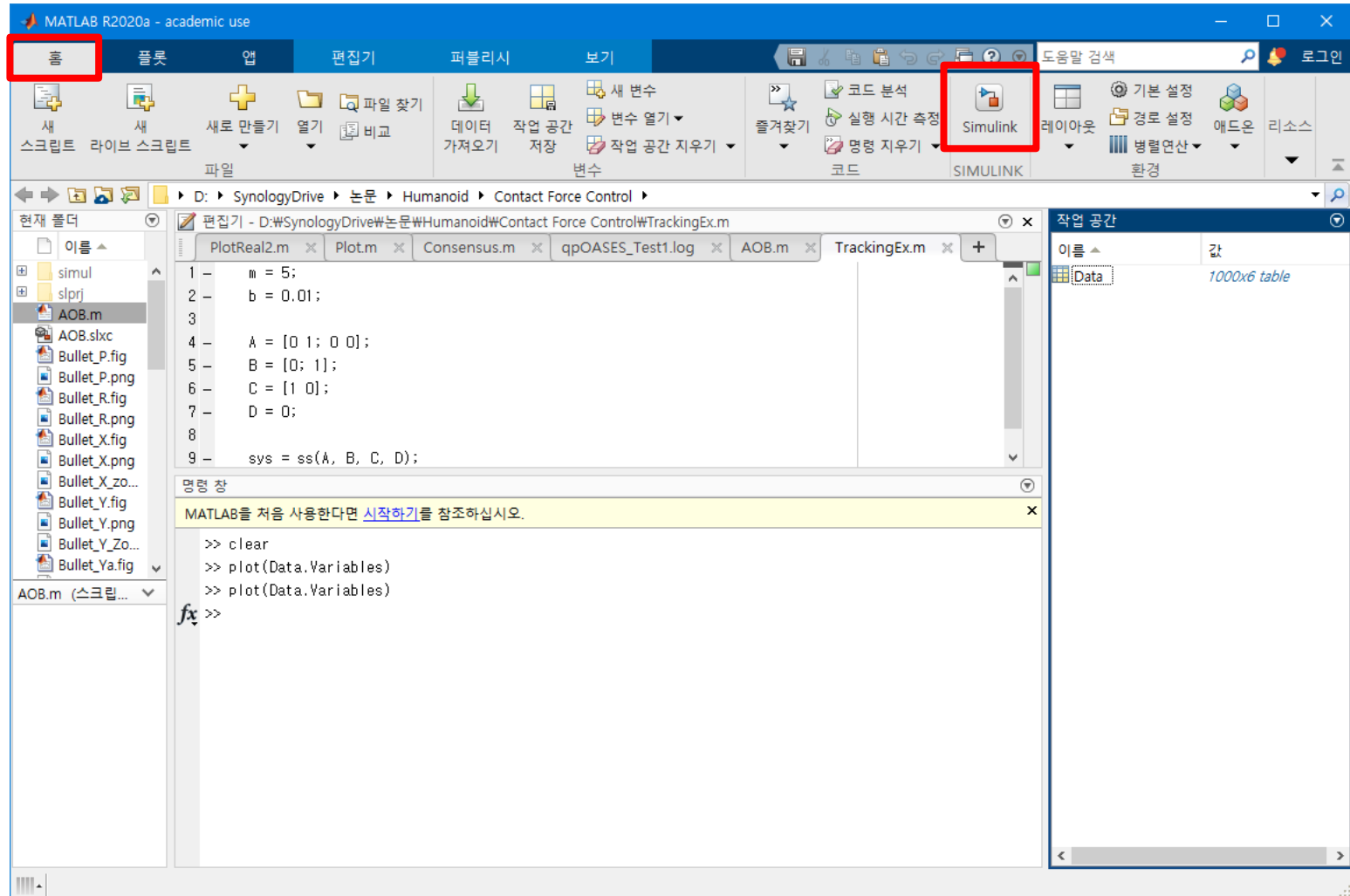
```
title('test graph','fontsize',40) % 그래프 명
```



MATLAB 기초

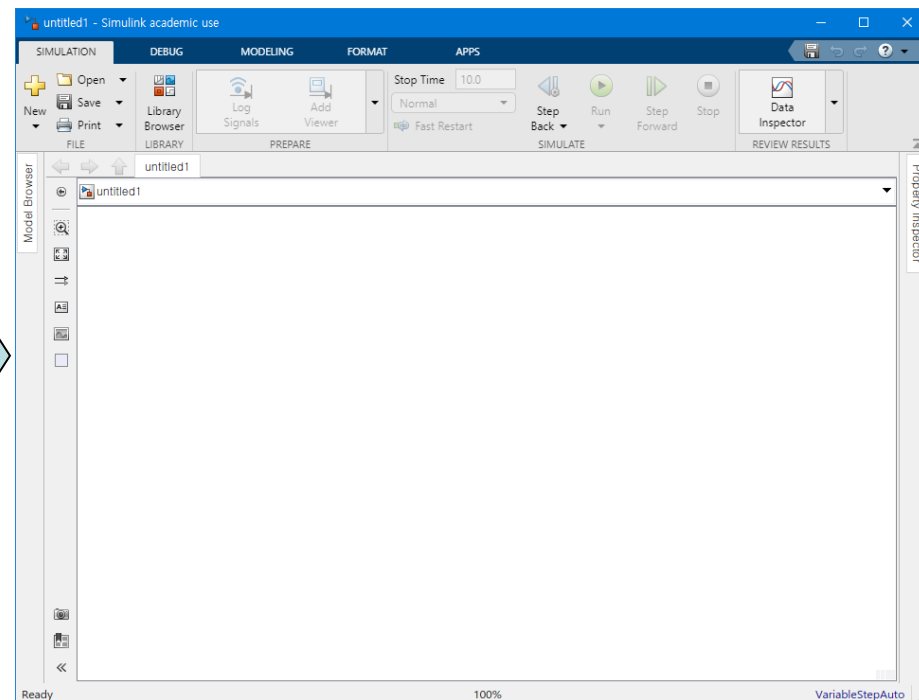
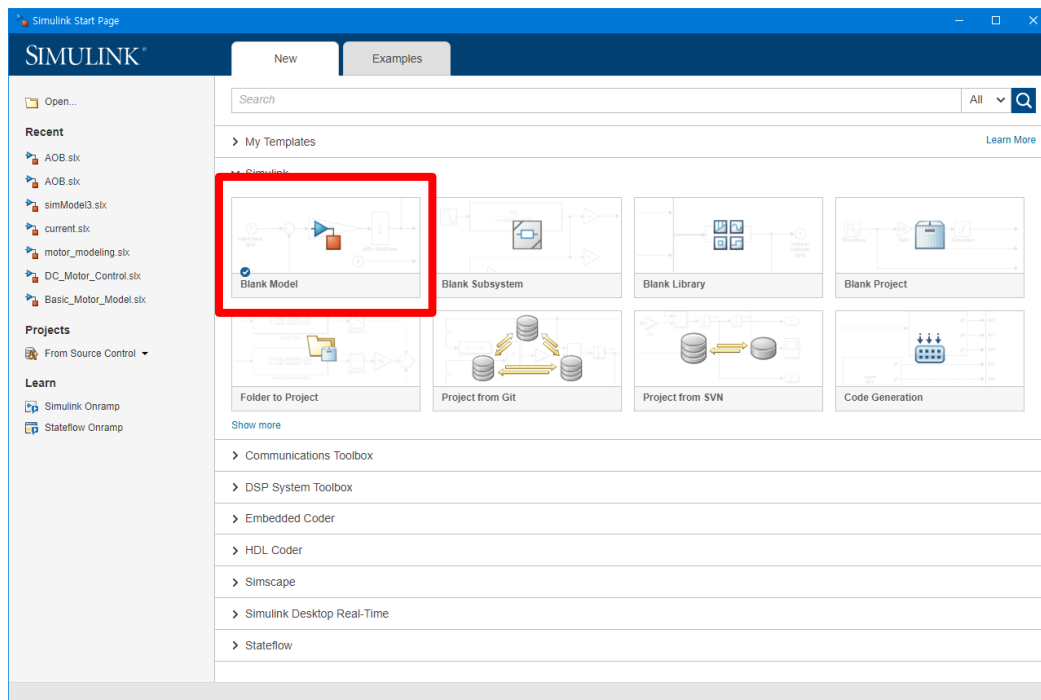
• Simulink 실행

-상단 메뉴에서 홈-Simulink 클릭



MATLAB 기초

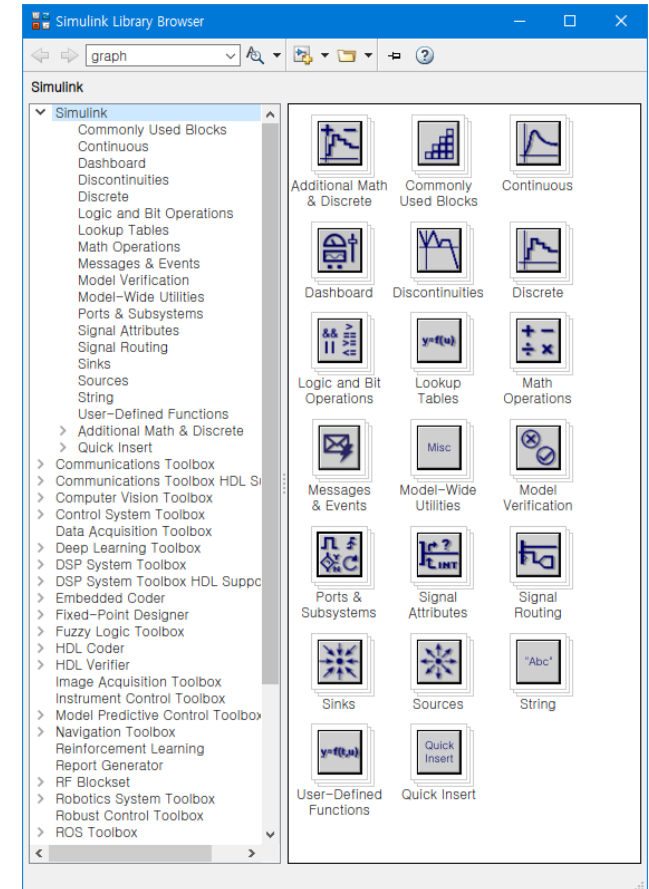
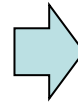
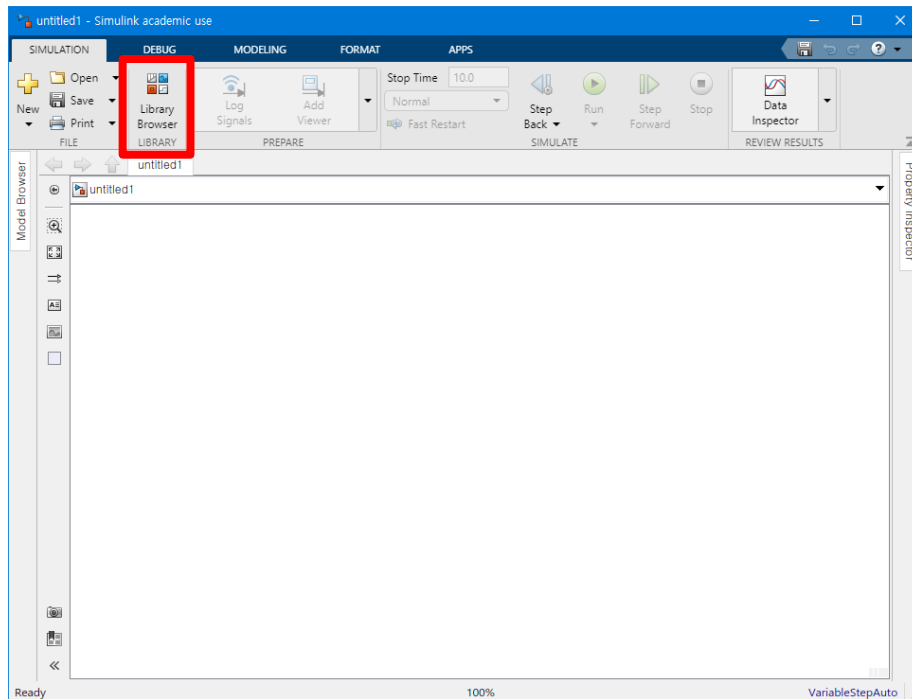
- Simulink 새 모델 만들기
-Blank Model 버튼 클릭 → 새 창이 실행됨



MATLAB 기초

• Simulink 실행

-Simulink 상단 메뉴에서 Library Browser를 클릭하면 Simulink 블록 브라우저 창이 나타남



□ MATLAB 기초

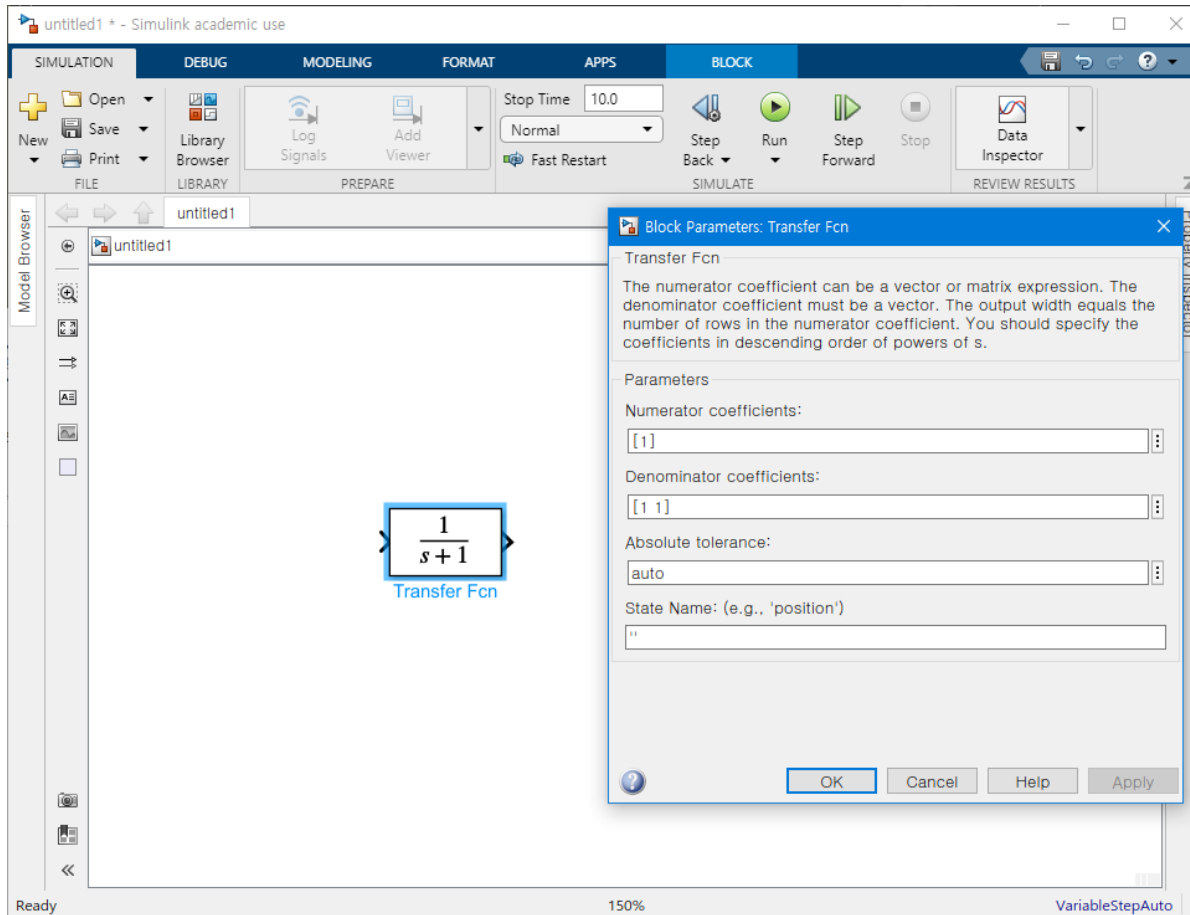
• Simulink 기초적인 시스템 구성 (Step 입력, 1차 전달함수)

- Library Browser에서 Transfer Fcn을 찾아 Simulink로 드래그
- Library Browser 상단에서 검색으로 찾을 수 있음

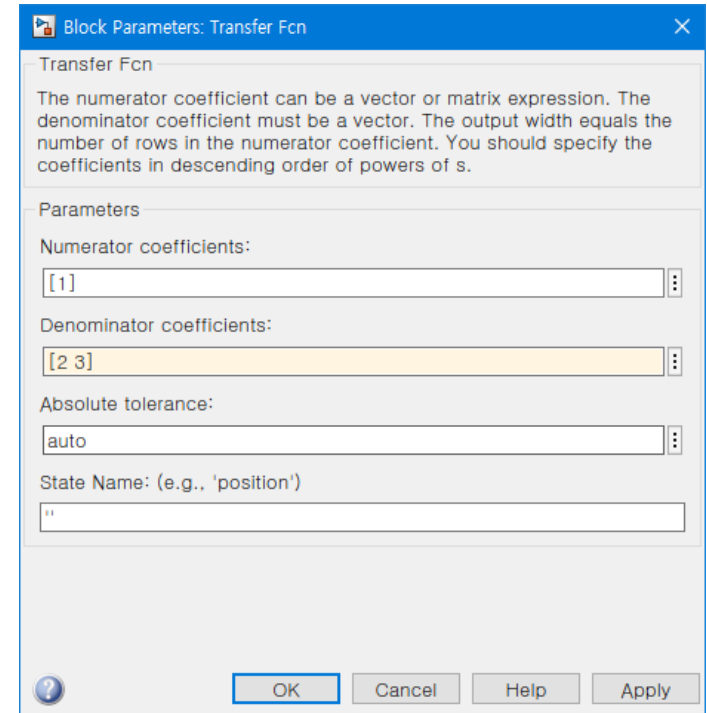
The screenshot displays the MATLAB Simulink environment. On the left, the 'Simulink Library Browser' window is open, showing a search for 'transfer'. The search results list various blocks, with the 'Transfer Fcn' block highlighted. A red box is drawn around the 'Transfer Fcn' block in the library. A red arrow points from this box to the 'untitled1' model window on the right. In the model window, the 'Transfer Fcn' block is shown in the workspace, with the transfer function $\frac{1}{s+1}$ displayed inside it. The top toolbar shows the 'Run' button and other simulation controls. The bottom status bar indicates 'Ready' and 'VariableStepAuto'.

□ MATLAB 기초

- Simulink 기초적인 시스템 구성 (Step 입력, 1차 전달함수)
 - Transfer Fcn를 더블클릭하여 파라미터 수정

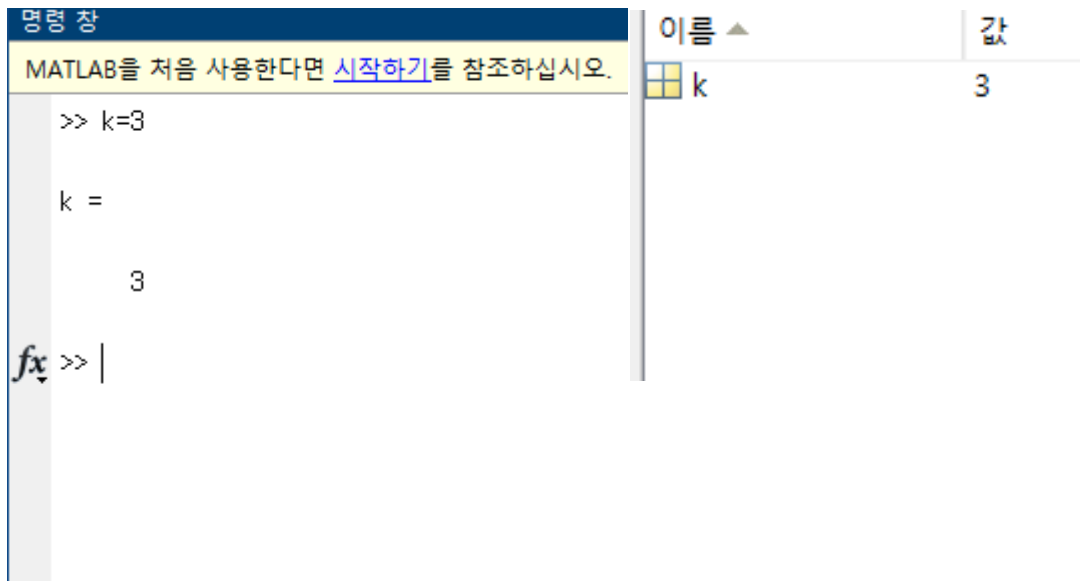


Ex) $\frac{1}{2s+3}$ 에 대한 파라미터

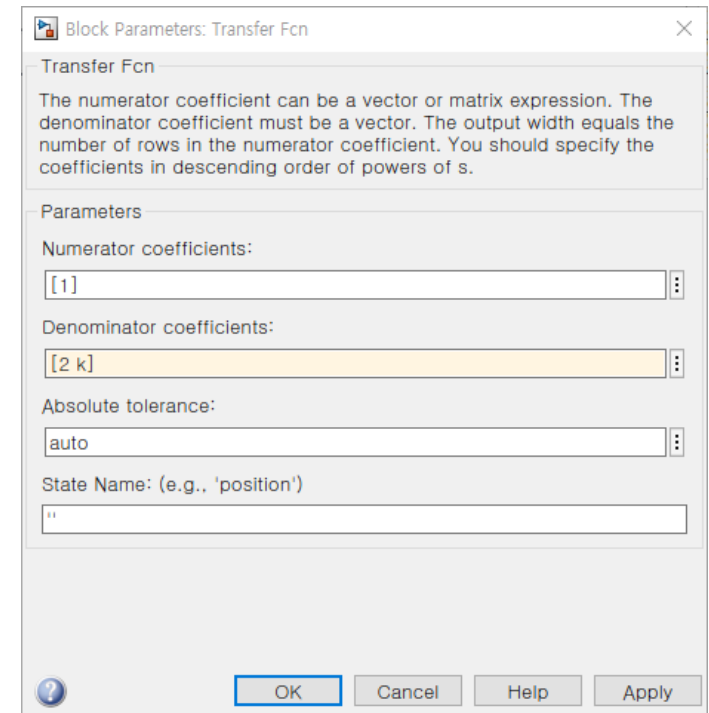


□ MATLAB 기초

- Simulink 기초적인 시스템 구성 (Step 입력, 1차 전달함수)
-다음과 같이 Matlab변수를 지정할 수 있음



Ex) $\frac{1}{2s+3}$ 에 대한 파라미터



□ MATLAB 기초

• Simulink 기초적인 시스템 구성 (Step 입력, 1차 전달함수)

- Library Browser에서 Step을 찾아 Simulink로 드래그 후 화살표 연결
- Step Time, Init Value, Final Value를 설정

The screenshot displays the MATLAB Simulink environment. On the left, the **Simulink Library Browser** is open, showing a search for 'step'. The search results list various blocks, with the **Step** block highlighted in the **Discrete** category. A red arrow points from the **Step** block in the library to the main workspace.

In the main workspace, a new model named **untitled1** is shown. It contains a **Step** block and a **Transfer Function** block with the transfer function $\frac{1}{2s + k}$. The **Step** block is connected to the input of the **Transfer Function** block.

The **Block Parameters: Step** dialog box is open, showing the configuration for the **Step** block. The parameters are set as follows:

- Step time:** 1
- Initial value:** 0
- Final value:** 1
- Sample time:** 0
- ☒ Interpret vector parameters as 1-D
- ☒ Enable zero-crossing detection

The **OK** button is highlighted, indicating that the parameters are being confirmed.

MATLAB 기초

• Simulink 기초적인 시스템 구성 (Step 입력, 1차 전달함수)

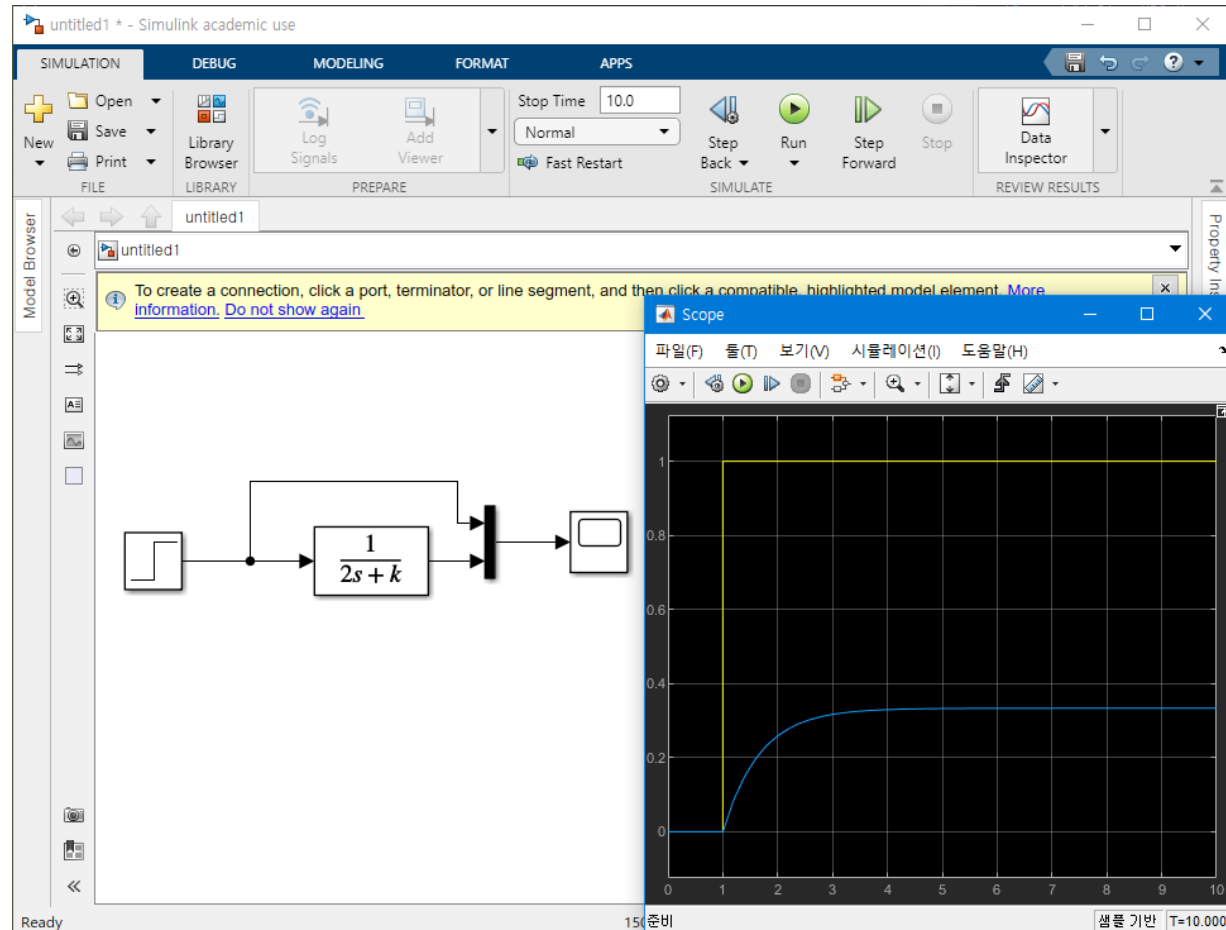
- Library Browser에서 Mux, Scope을 찾아 아래와 같이 구성
- 더블클릭하면 그래프 창이 나타남

The screenshot displays the MATLAB Simulink environment. On the left, the 'Simulink Library Browser' window is open, showing the 'Commonly Used Blocks' category. The 'Mux' block is highlighted with a red rectangle, and the 'Scope' block is also highlighted with a red rectangle. The main workspace shows a block diagram of a system. It starts with a 'Step' input block, followed by a 'Product' block (represented by a circle with an 'x'), then a 'Transfer Function' block with the equation $\frac{1}{2s + k}$. The output of the transfer function is connected to a 'Mux' block, which is then connected to a 'Scope' block. The 'Scope' block is highlighted with a red rectangle. The 'Scope' window is open on the right, showing a grid for plotting the signal. The status bar at the bottom indicates 'Ready' and '15(준비)'.

□ MATLAB 기초

• Simulink 기초적인 시스템 구성 (Step 입력, 1차 전달함수)

-상단 메뉴의 Run을 누르면 시뮬레이션이 진행되고 결과 그래프를 볼 수 있음



□ MATLAB 기초

◆ MATLAB 팁

- 대문자와 소문자는 대등하지 않다.
- 변수의 이름을 적으면 MATLAB은 그 변수의 현재 값을 스크린에 보여준다.
- 명령문의 마지막 부분에 세미콜론(;)을 두면 변수 값이 스크린에 나타나지 않는다.
- 위쪽 화살표와 아래쪽 화살표 키를 이용하여 이전에 입력한 명령문들을 순차 적으로 검색할 수 있다.
- 'help'를 입력하면 명령어, 함수에 대한 도움말에 접할 수 있다.
- 함수나 변수 이름을 일부분만 적고 탭 키를 누르면 MATLAB은 나머지 부분을 완성할 수 있는 경우들을 모두 제시하여 줌으로써 사용자가 한 가지를 선택하여 이름의 나머지 부분을 완성할 수 있게 하여 준다.
- MATLAB의 수행을 중간에 그만두려면 ctrl+c를 입력한다.