MCE Coding Camp

MATLAB Basic

Handong Global University
School of Mechanical and Control Engineering

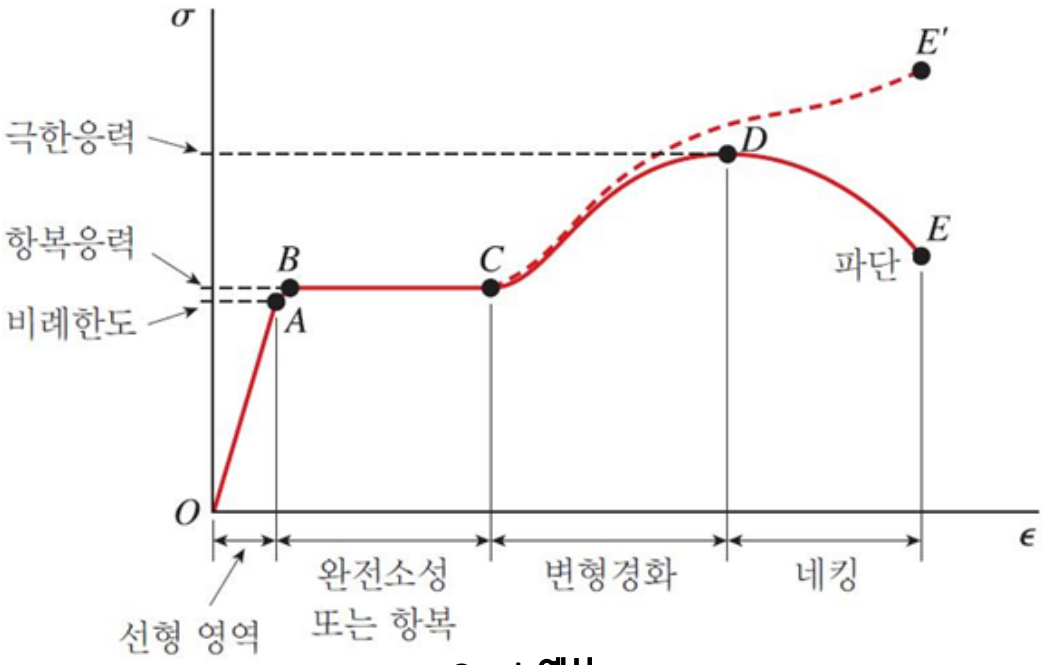
21900031 Jin Kwak

2024. 07. 23

Content

I. Introduction (Why do we use MATLAB?)

1. 공학 계산 용이하다.
2. 데이터 분석 및 처리가 빠르다.
3. Plot 등을 통하여 시각화를 쉽게 할 수 있다.
4. Simulink 등을 통해 시뮬레이션을 직관적으로 할 수 있고, 3번을 통해 즉각적으로 시각화할 수 있다.
5. (C와 같은 프로그래밍 언어에 비해) 간단하다.



Graph 예시

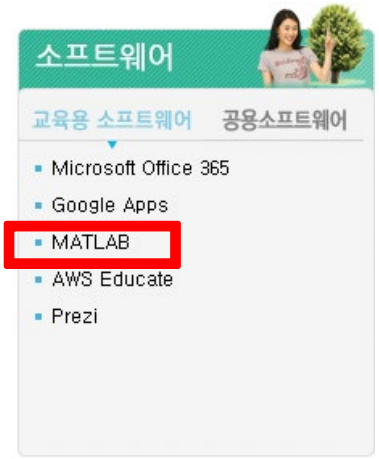
I. Introduction

1) Objectives

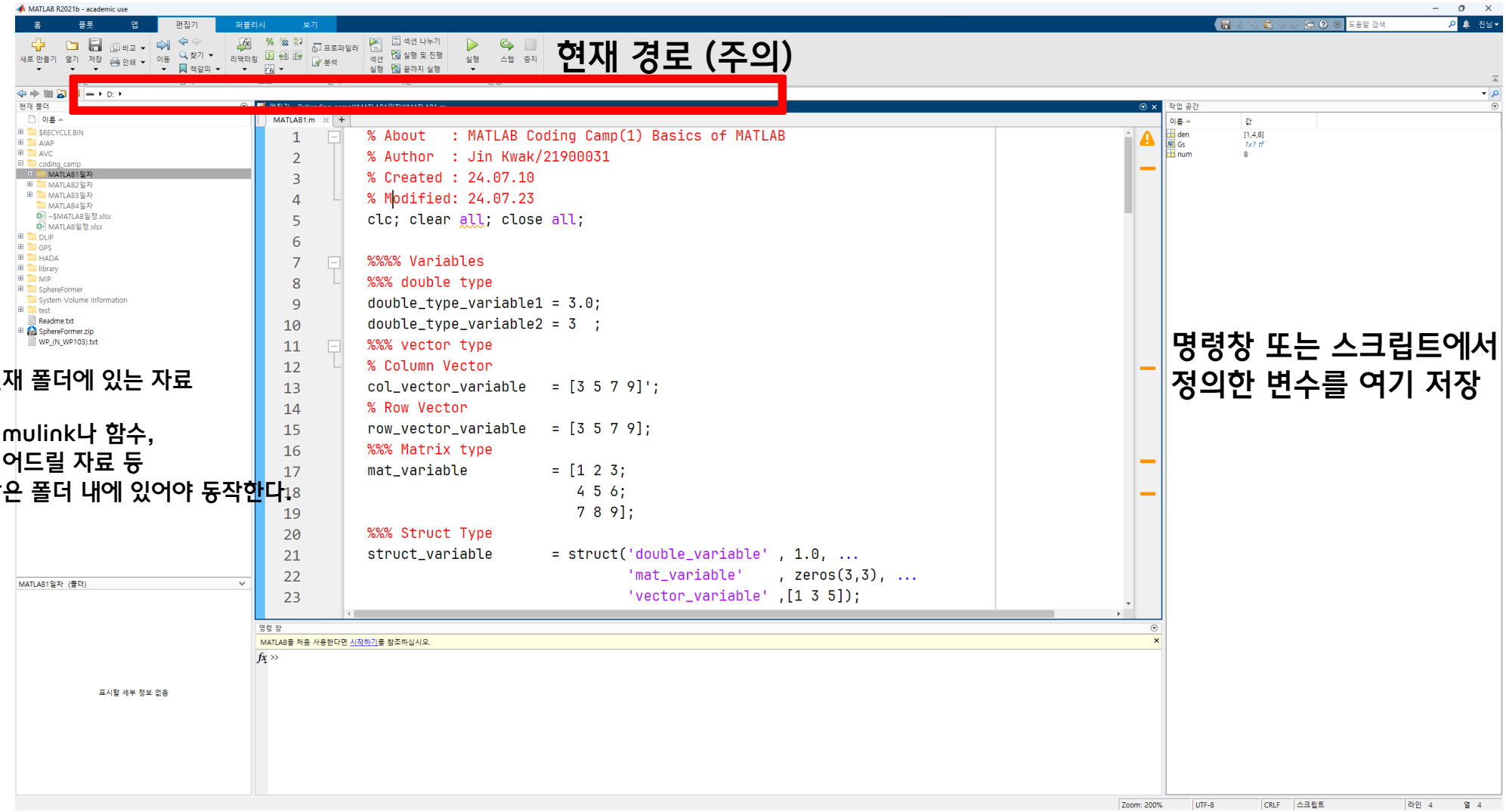
- 필요에 따라 MATLAB 및 알맞은 MATLAB Toolbox를 설치할 수 있다.
- Double, 벡터, 행렬, 구조체, 셀형 등 다양한 변수를 사용할 수 있다.
- 조건문, 반복문을 통해 변수 값/행렬 요소 등 접근할 수 있다.
- 조건문, 반복문을 통해 변수 및 행렬 요소 값을 업데이트할 수 있다.
- 수식을 MATLAB에 적용하여, 수치적 결과를 도출할 수 있다.
- 함수를 작성/사용할 수 있다.

II. How to Install MATLAB

1. Hisnet 로그인 → Quick Link(IT Support) → 교육용 소프트웨어(MATLAB)
2. MathWorks 회원가입/로그인 → MATLAB 버전 선택 → 설치



III. Basics of MATLAB (MATLAB 레이아웃)



III. Basics of MATLAB (단축키)

1. Ctrl + R: 주석 처리
2. Ctrl + T: 주석 처리 해제
3. Ctrl + Enter: 현재 섹션 실행
4. %%: 편집 섹션 나누기
5. Ctrl + D: 함수 정의 부분로 이동

III. Basics of MATLAB (스크립트 작성 기초)

1. 문서화 주석

1. 누가 작성했는지
2. 무엇을 작성했는지
3. 언제 작성했는지
4. (e.g. V.0.0.2)
5. 스크립트 상단에 작성

MATLAB1.m

```
% About    : MATLAB Coding Camp(1) Basics of MATLAB
% Author   : Jin Kwak/21900031
% Created  : 24.07.10
% Modified: 24.07.23
```

2. 초기화

```
clc; clear all; close all;
```

1. 스크립트 상단에 초기화함으로 여러 스크립트 실행해서 발생하는 오류 방지
2. 명령창, 작업 공간, Figure, Window 초기화

III. Basics of MATLAB (변수 정의)

1. double 형
2. 벡터 형
    1. 열 벡터
    2. 행 벡터
3. 행렬
4. 구조체 형
5. 셀 형
6. Symbolic 형
7. Table형
8. 더 많지만 다 Cell형과 Struct형처럼 접근 가능.

III. Basics of MATLAB (변수 요소 접근)

1. double 형
2. 벡터 형 요소 읽기/ 쓰기 (e.g. 세번째 요소)
    1. 열 벡터

```
col_vector_variable1 = [3 5 7 9]';
col_vector_element = col_vector_variable1(3); % Read
col_vector_variable1(3) = 4; % Write
```

    2. 행 벡터

```
row_vector_variable = [3 5 7 9];
row_vector_element = row_vector_variable(2);
row_vector_variable(2)= 9;
```

3. 행렬

```
mat_variable = [1 2 3;
                4 5 6;
                7 8 9];
mat_element = mat_variable(3,2); % 3행 2열에 있는 요소 가져오기
mat_variable(3,2) = 10;          % 3행 2열에 있는 요소 바꾸기
mat_row = mat_variable(3,:);     % 3행 요소들 가져오기 (행 벡터)
mat_col = mat_variable(:,2);     % 2열 요소들 가져오기 (열 벡터)
```

III. Basics of MATLAB (변수 정의)

1. 구조체 형

```
struct_variable = struct('double_member' , 1.0       , ...
                         'mat_member'    , ones(3,3), ...
                         'vector_member' ,[1 3 5]);
struct_member  = struct_variable.double_member;
struct_member2 = struct_variable.mat_member(2,1);
```

2. 셀 형

```
cell_variable = {3,3};
cell_variable{1,1} = [(0:0.01:30)', (sin(0:0.01:30))'];
```

III. Basics of MATLAB (변수 정의)

셀 형을 왜 쓰는가?

1. 보통 데이터를 여러 번 받을 때 같은 셀 형 안에 저장할 수 있다.
2. Struct에 배열 넣을 수 있다하지 않았냐?
    1. 숫자 Indexing 별로 사용할 수 있어서 편하다
    2. 어떻게 썼는지 한번 확인해보자
3. 그럼 Struct는 뭐할 때 쓰냐?
    1. 가까운 연관이 있는 변수끼리 모은다.
    2. 예시

```
TIME = struct('Start', 0.0  ,...
              'Final',30.0  ,...
              'Ts'   , 0.001,...
              'time' , 0    ,...
              'Ntime', 0    ,...
              'idx'  , 1);

ENUM = struct('AccX', 1, 'AccY', 2, 'AccZ',3, 'N_Acc', 3, ...
              'VelX', 1, 'VelY', 2, 'VelZ',3, 'N_Vel', 3, ...
              'PosX', 1, 'PosY', 2, 'PosZ',3, 'N_Pos', 3);
```

III. Basics of MATLAB (자주 쓰이는 함수)

엄청 많은 함수를 모두 나열할 수는 없다. 하지만 이정도는 바로 쓸 줄 알자!

주의할 점: 덧셈, 곱셈 등 수학 연산을 할 때, 차원을 맞추자.

연산	기호	예	연산	예	연산	예
덧셈	+	5+3	제곱근	sqrt(64)	사인	sin(2*pi)
뺄셈	-	5-3	지수함수	exp(3)	사인	sind(90)
곱셈	*	5*3	절대값	abs(-15)	코사인	cos(0.5*pi)
오른쪽 나눗셈	/	5/3	자연 로그	log(30)	코사인	cosd(90)
왼쪽 나눗셈	\	5\3 = 3/5	로그10	log(10)	역탄젠트	atan2(30,40)
지수 연산	^	5^3 = 5^3	팩토리얼	factorial(5)	반올림	round(3.4)

III. Basics of MATLAB (자주 쓰이는 함수)

행렬, 배열 등을 편하게 정의할 수 있는 함수

함수	예
0으로 이뤄져 있는 행렬	zeros(행 개수,열 개수)
1로 이뤄져 있는 행렬	ones(행 개수,열 개수)
x부터 y까지 n개의 원소 벡터 생성	linspace(x,y,n)
x부터 y까지 e만큼 커지는 벡터 생성	x:e:y
단위 행렬 생성	eye(3,3)
행렬 전치	transpose(Mat) or Mat‘

III. Basics of MATLAB (자주 쓰이는 함수)

행렬, 배열 관련 함수, 산술

1. 저학년일 수록 많이 나는 에러 → 벡터 요소 별 곱셈

연산	예
행렬 덧셈	MatA+MatB
행렬 뺄셈	MatA-MatB
행렬 곱셈	MatA*MatB
스칼라 배	ScalarA*MatA
벡터 요소 별 곱셈	VecA.*VecB
역행렬	inv(MatA)
대각합	trace(MatA)
고유 값, 고유벡터	eig(MatA)

III. 반복문 (for)

반복문, 조건문은 프로그래밍 언어에 반드시 존재하는 문법이다.

```
for 루프 인덱스 변수= 첫번째 k값:k의 증분: k의 마지막 값
        MATLAB 명령어 그룹
end
```

MATLAB은 C와 다르게 {}가 아닌 end를 붙여 그 사이 명령어에 대한 반복문을 수행한다.

```
for idx = 1:300
        disp('Iteration = ‘);
        disp(num2str(idx) );
end
```

반복문에 다음과 같이 배열이 들어갈 수 있다.

```
iteration = 1:300;
for idx = iteration
        disp('For Loop Index = ‘);
        disp(num2str(idx) );
end
```

III. 반복문 (while)

보통 몇 번 루프 반복을 할지 결정되지 않은 상황에서 사용한다. (e.g. 시스템이 수렴할 때까지)

```
while 조건 수식 %(조건 수식이 참인 동안 반복)
       MATLAB 명령어 그룹
end
```

III. 반복문 예제

다음 테일러 급수를 매트랩으로 구현해보자!

종료 조건은 High Order Term의 절대값이 0.00001보다 작을 때 까지

이후에 매트랩 내장함수 sin()과 비교하여 수치 오차가 얼마나 있는지 확인하자

$$\sin(x) = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{(2k+1)!}$$

III. 조건문 (if)

조건문을 사용해보자. C와 논리구조는 동일하다. End는 센스 있게 붙여주자.

일주일동안 비슷하게 많이 했기 때문에 패스한다.

```
if 조건식 %(조건식이 참인 경우에)
        MATLAB 명령어 그룹
end
```

III. 함수 (Function)

함수를 만들고 사용해보자. 똑같이 end는 센스 있게 붙여주자.

- **C와 다르게 여러 출력인자를 반환할 수 있다.**
- **함수 파일은 일반 파일처럼 .m파일로 저장된다. → 함수명과 같은 파일 이름으로 저장한다.**
- **다른 폴더에 함수가 있을 때, 함수를 찾지 못하고 에러가 난다.**
  (틀리면서 적응하거나 탐색경로에 있어야한다.)

```
function  [출력인자]=함수이름(입력인자 목록)
        % help 함수 이름을 명령창에 실행할 때 나올 출력 문구
        함수 본체
        출력인자에 값을 할당
end
```

III. 사용자 정의 함수 (Function)

함수 출력 인자가 두개 이상일 경우, 대괄호를 꼭 사용해야한다.

```
function [cosA, sinA] = sinusoidal(angle)
% This function returns cosine, sine value of the angle
% Param : angle [rad]
% Returns: (2X1 double) [cos(angle) sin(angle)]
cosA = cos(angle);
sinA = sin(angle);
end
```

그렇다면 main문에서는 어떻게 사용될까? → 똑같이 대괄호로 값을 받는다.

```
[cosine, sine]=sinusoidal(0*pi/180);
```

III. 익명 함수 (Anonymous Function)

비교적 간단한 수식을 함수로 따로 만들어 줄 필요가 있을까? 없다

간단하게 만드는 한줄짜리 사용자 정의 함수다.

```
익명함수 이름= @(입력 인자 목록) 수식;
```

세제곱을 계산하는 익명 함수는 다음과 같다. (수식은 한 개의 유효한 수학식으로 구성된다.)

```
cube = @(x) x^3;
```

사용도 똑같이 편하게 할 수 있다.

```
three_cube= cube(3);
```