



# **Introduction to MATLAB**

***Tae-Seong Kim, Ph.D.***

***Department of Biomedical Engineering  
College of Electronics & Information  
Kyung Hee University***



## Contents

---

- MATLAB 소개 및 기본 구성
- MATLAB의 유용한 명령어
- MATLAB 변수 선언 방법
- MATLAB에서의 각종 연산 방법
- Graph 그리는 법
  - plot
  - stem



---

# Basics of MATLAB



# Introduction

---

- 개요
  - Mathworks ([www.mathworks.com](http://www.mathworks.com))
  - MATLAB은 Matrix Laboratory의 약어
  - 여러 분야의 Toolbox(일종의 Library)를 제공
  - 현재 버전 R2011b
  - 강의자료: MATLAB R13인 6.5 Version 기준
- 이용범위
  - 수학과 관련된 계산
  - 알고리즘 개발
  - 상황 모델링과 데이터 분석
  - 여러 가지 학문적 그래픽 표현
  - GUI를 통한 응용 프로그램 개발



# MathWorks

- [www.mathworks.co.kr](http://www.mathworks.co.kr)

The screenshot shows the MathWorks Korea homepage. At the top is the MathWorks logo with the tagline "Accelerating the pace of engineering and science". Navigation links include "한국" (Korea), "연락처" (Contact), "store", "create account", and "로그인" (Login). A main banner features the text "MATLAB을 사용한 GPU 컴퓨팅" (GPU Computing using MATLAB) and "하위 단계의 프로그래밍 없는 NVIDIA CUDA GPU 연산 속도" (NVIDIA CUDA GPU operation speed without lower-level programming), with a "비디오 보기" (Watch Video) button. Below the banner are tabs for "제품" (Products), "이벤트 안내" (Event Guide), and "교육 과정" (Courses). The "제품" tab is active, showing a list of products: Parallel Computing Toolbox, Data Acquisition Toolbox, Instrument Control Toolbox, Bioinformatics Toolbox, Simscape, Simulink HDL Coder, xPC Target, and Polyspace code verifiers. To the right, there are two featured sections: "최신 릴리즈" (Latest Release) for R2011b, highlighting updates to 1 new product and 85 other products, and "제품 스포트라이트" (Product Spotlight) for MATLAB Mobile on iPad, with a "다운로드" (Download) link.

MathWorks® Accelerating the pace of engineering and science

한국 | 연락처 | store

create account | 로그인

제품 및 서비스 | 솔루션 | 대학 커리큘럼 | Support | User Community | 이벤트 | 회사소개

## MATLAB을 사용한 GPU 컴퓨팅

하위 단계의 프로그래밍 없는 NVIDIA CUDA GPU 연산 속도

Run existing CUDA kernels directly from MATLAB

비디오 보기

최근 소식 | 200개 이상의 채용중인 직무 - 지금 지원하십시오! | 2012년1월25일

### 제품

테크니컬 컴퓨팅 언어인 **MATLAB** 과 모델 기반 설계를 위한 **Simulink** 관련 제품 자세히 알아보기

#### 주요 제품

- Parallel Computing Toolbox
- Data Acquisition Toolbox
- Instrument Control Toolbox
- Bioinformatics Toolbox
- Simscape
- Simulink HDL Coder
- xPC Target
- Polyspace code verifiers

#### 최신 릴리즈

**R2011b**

1개의 신제품 및 기타 85개 제품에 대한 업데이트

» 릴리즈 하이라이트

#### 제품 스포트라이트

iPad에서 MATLAB Mobile를 실행해 보십시오

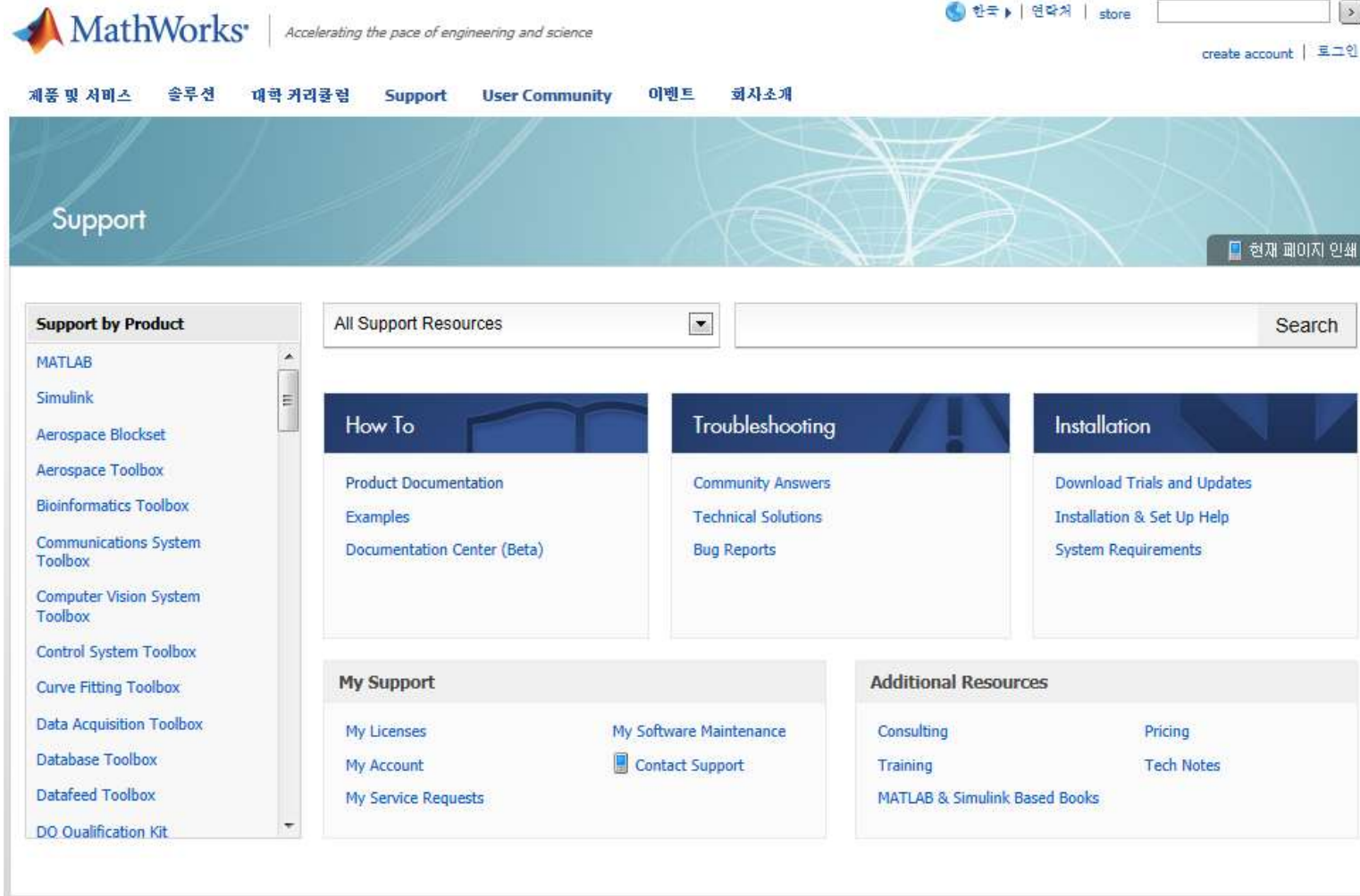
» 다운로드

전체 제품 보기 및 소프트웨어 평가판 신청



# Documentation

- <http://www.mathworks.co.kr/support/>



The screenshot shows the MathWorks Support page. At the top, the MathWorks logo is followed by the tagline "Accelerating the pace of engineering and science". To the right, there are links for "한국" (Korea), "연락처" (Contact), "store", a search bar, and "create account" / "로그인" (Login). Below this is a navigation bar with links for "제품 및 서비스" (Products and Services), "솔루션" (Solutions), "대학 커리큘럼" (University Curriculum), "Support", "User Community", "이벤트" (Events), and "회사소개" (About Us). The main header area has a "Support" title and a "현재 페이지 인쇄" (Print this page) button. The content area is divided into several sections: "Support by Product" on the left with a list of toolboxes; "All Support Resources" at the top center with a dropdown menu and a search bar; three main resource columns: "How To" (Product Documentation, Examples, Documentation Center (Beta)), "Troubleshooting" (Community Answers, Technical Solutions, Bug Reports), and "Installation" (Download Trials and Updates, Installation & Set Up Help, System Requirements); "My Support" at the bottom left with links for My Licenses, My Account, My Service Requests, My Software Maintenance, and Contact Support; and "Additional Resources" at the bottom right with links for Consulting, Training, Pricing, Tech Notes, and MATLAB & Simulink Based Books.

MathWorks Accelerating the pace of engineering and science

한국 | 연락처 | store  >

create account | 로그인

제품 및 서비스 솔루션 대학 커리큘럼 Support User Community 이벤트 회사소개

Support

현재 페이지 인쇄

Support by Product

- MATLAB
- Simulink
- Aerospace Blockset
- Aerospace Toolbox
- Bioinformatics Toolbox
- Communications System Toolbox
- Computer Vision System Toolbox
- Control System Toolbox
- Curve Fitting Toolbox
- Data Acquisition Toolbox
- Database Toolbox
- Datafeed Toolbox
- DO Qualification Kit

All Support Resources  Search

How To

- Product Documentation
- Examples
- Documentation Center (Beta)

Troubleshooting

- Community Answers
- Technical Solutions
- Bug Reports

Installation

- Download Trials and Updates
- Installation & Set Up Help
- System Requirements

My Support

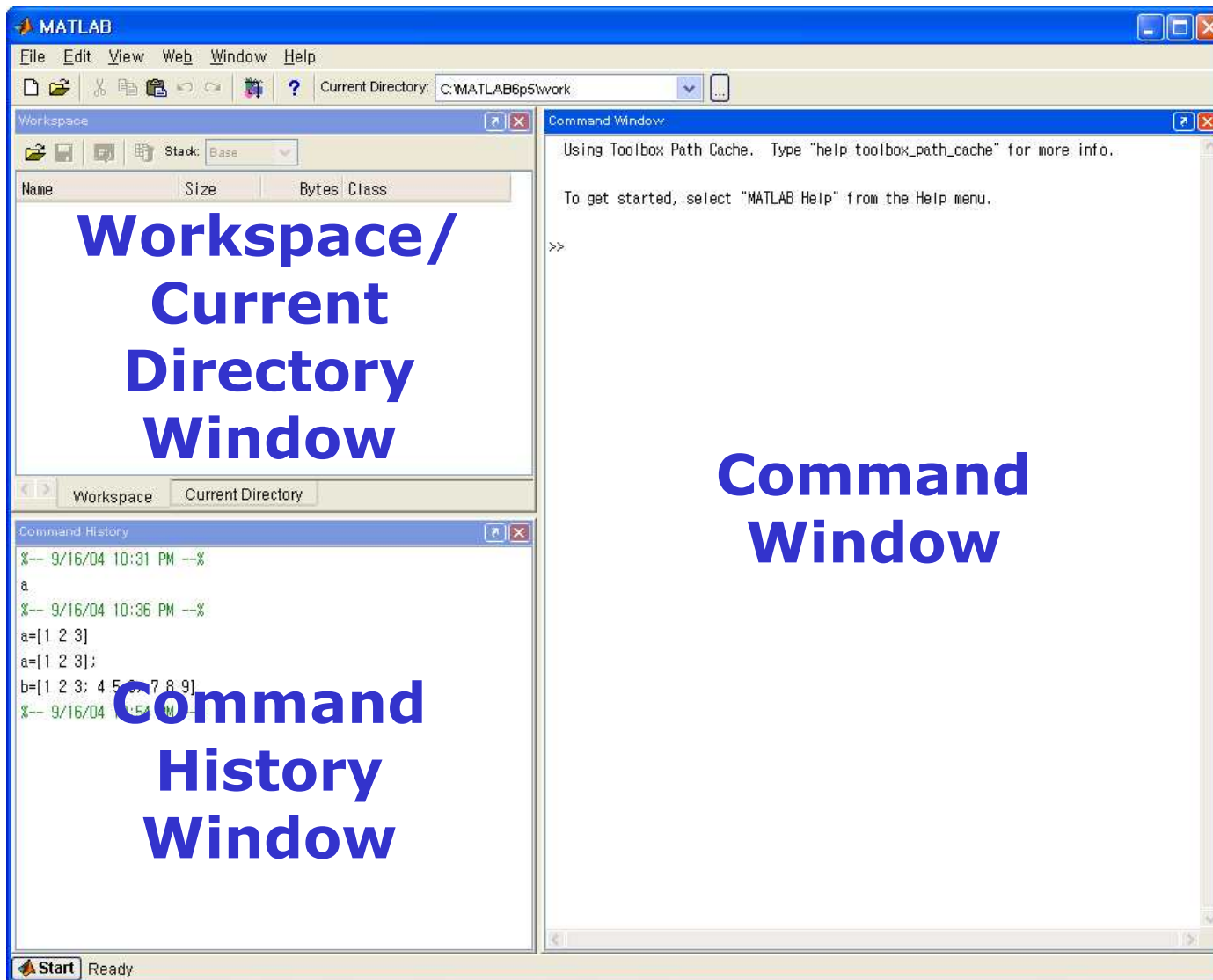
- My Licenses
- My Account
- My Service Requests
- My Software Maintenance
- Contact Support

Additional Resources

- Consulting
- Training
- MATLAB & Simulink Based Books
- Pricing
- Tech Notes

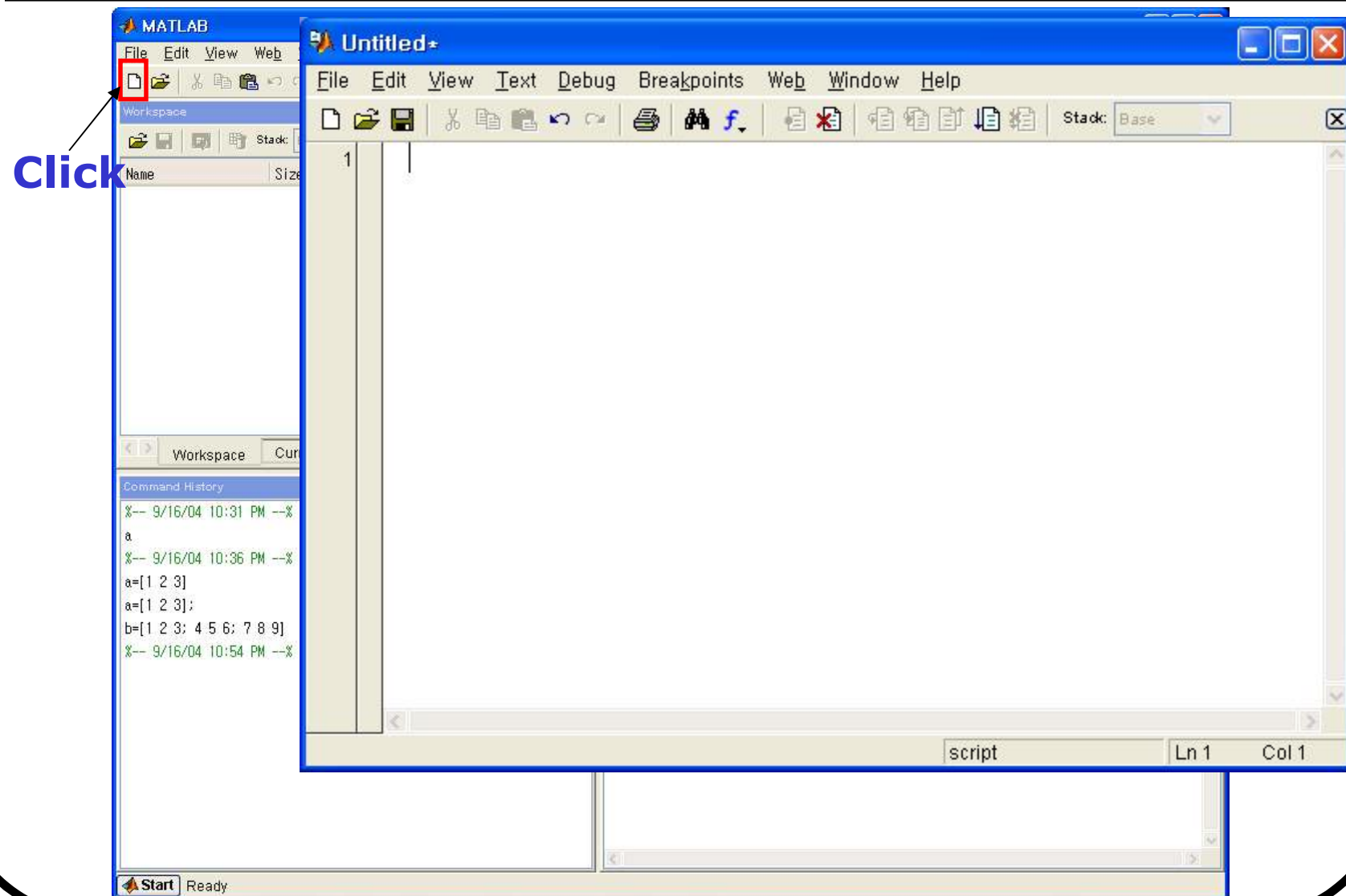


## Window 구성





# M-File Editor/Debugger







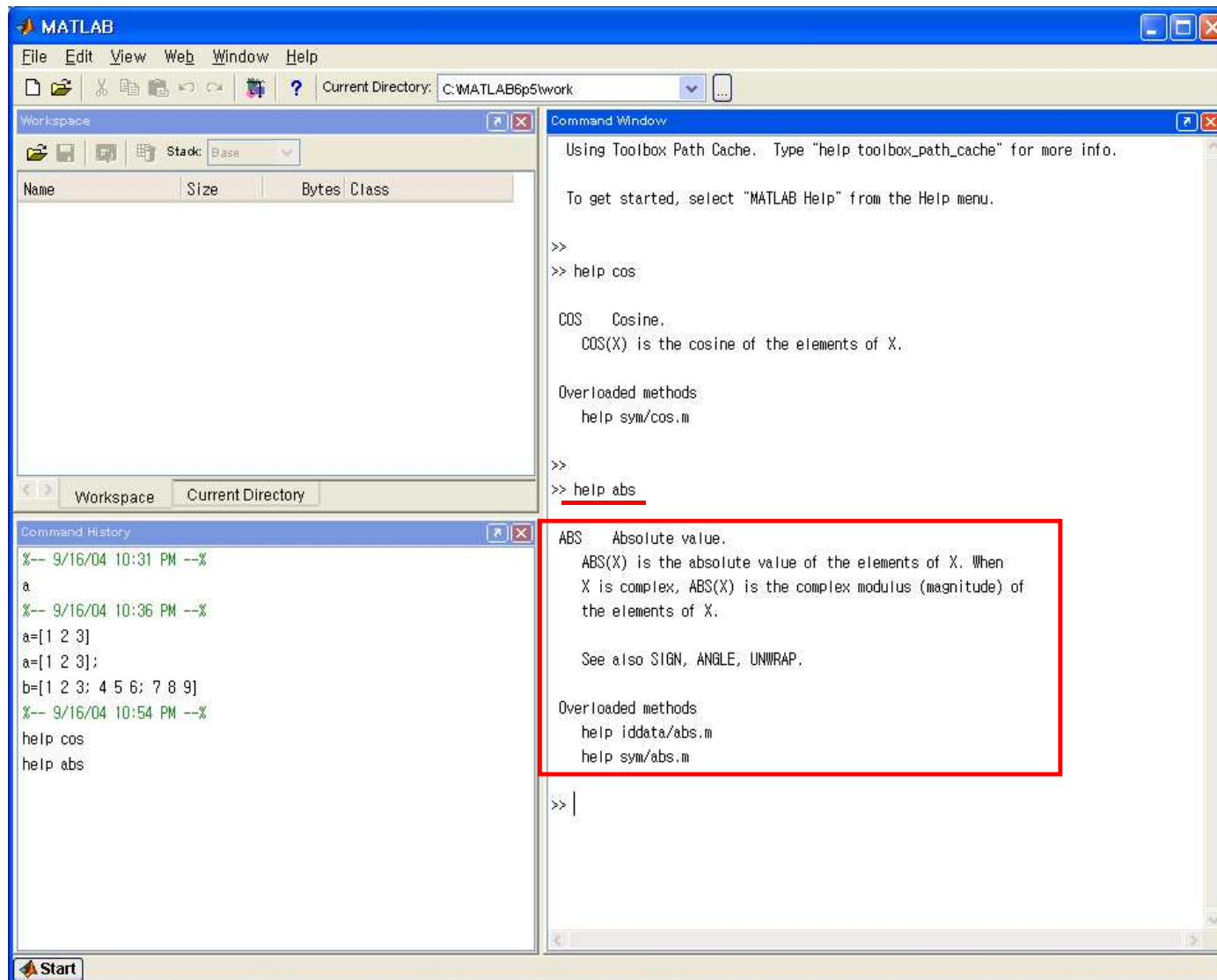
## 유용한 명령 (commands)

---

- `help`
  - 특정 명령에 대한 설명
- `lookfor`
  - 검색어에 관련된 함수들과 간략한 설명
- `who/whos`
  - 변수에 대한 정보
- `ls`
  - List
- `cd`
  - Change Directory
- `clear/clear all`
  - Clear 변수

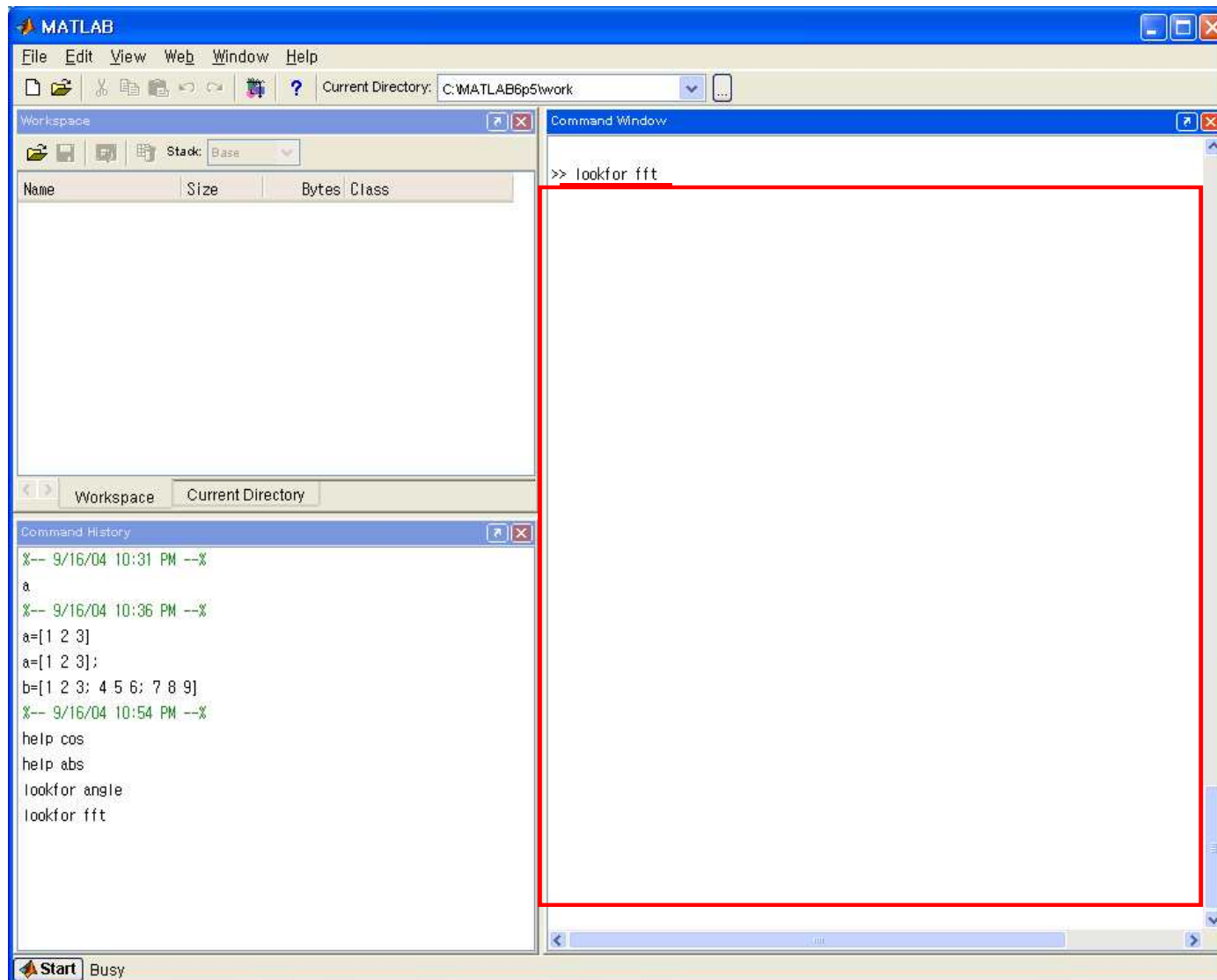


# help





# lookfor





## Expression

---

- Variable
  - 해당 이름을 가지는 1 by 1 matrix
  - 대소문자 구분
  - 수식의 결과는 자동적으로 ans라는 변수에 저장
  - Ex : num\_student=25, A = 1, a = 2
- Number
  - IEEE floating point standard (long)
  - 16개의 유효 숫자와 함께  $10^{-308}$ 에서  $10^{+308}$ 범위를 가짐
  - i 혹은 j를 통한 허수 표현
  - inf : 1/0과 같은 무한대
  - NaN : Not-a-Number, 무의미한 수 ex) 0/0, inf-inf



# Format

- Format

- 화면에 표시되는 자리 수, 형식 결정
- 화면상에 표시되는 수치 형식만을 결정!
- 가능한 format

- short
- short e
- short g
- long
- long e
- long g
- bank
- rat
- hex

The image shows a screenshot of the MATLAB Help window. The title bar says 'MATLAB'. The menu bar includes 'File', 'Edit', 'View', 'Web', 'Window', and 'Help'. The toolbar contains icons for file operations and a search icon. The 'Current Directory' is set to 'F:\Study\MATLAB\work'. The main text area displays the documentation for the 'FORMAT' command. It starts with 'FORMAT Set output format.' followed by a paragraph: 'All computations in MATLAB are done in double precision. FORMAT may be used to switch between different output display formats as follows:'. Below this is a list of format options: 'FORMAT' (Default. Same as SHORT.), 'FORMAT SHORT' (Scaled fixed point format with 5 digits.), 'FORMAT LONG' (Scaled fixed point format with 15 digits.), 'FORMAT SHORT E' (Floating point format with 5 digits.), 'FORMAT LONG E' (Floating point format with 15 digits.), 'FORMAT SHORT G' (Best of fixed or floating point format with 5 digits.), 'FORMAT LONG G' (Best of fixed or floating point format with 15 digits.), 'FORMAT HEX' (Hexadecimal format.), 'FORMAT +' (The symbols +, -, and blank are printed for positive, negative and zero elements. Imaginary parts are ignored.), 'FORMAT BANK' (Fixed format for dollars and cents.), and 'FORMAT RAT' (Approximation by ratio of small integers.). At the bottom, under the heading 'Spacing:', it lists 'FORMAT COMPACT' (Suppress extra line-feeds.) and 'FORMAT LOOSE' (Puts the extra line-feeds back in.). The Windows taskbar at the bottom shows the 'Start' button.

```
FORMAT Set output format.  
All computations in MATLAB are done in double precision.  
FORMAT may be used to switch between different output  
display formats as follows:  
FORMAT          Default. Same as SHORT.  
FORMAT SHORT    Scaled fixed point format with 5 digits.  
FORMAT LONG     Scaled fixed point format with 15 digits.  
FORMAT SHORT E  Floating point format with 5 digits.  
FORMAT LONG E   Floating point format with 15 digits.  
FORMAT SHORT G  Best of fixed or floating point format with 5 digits.  
FORMAT LONG G   Best of fixed or floating point format with 15 digits.  
FORMAT HEX      Hexadecimal format.  
FORMAT +        The symbols +, - and blank are printed  
                for positive, negative and zero elements.  
                Imaginary parts are ignored.  
FORMAT BANK     Fixed format for dollars and cents.  
FORMAT RAT      Approximation by ratio of small integers.  
  
Spacing:  
FORMAT COMPACT  Suppress extra line-feeds.  
FORMAT LOOSE    Puts the extra line-feeds back in.
```



---

# **Variables and Arrays**



## 변수 선언 – **Matrix** 형태로 선언

- 입력
  - 공백이나 콤마(,) 로 원소 구분
  - 세미콜론(;)으로 행 구분
  - 괄호를 이용하여 행렬 표시
  - [Ex] `a=[1 2 3];` → "1, 2, 3"이 저장된 1x3 행렬 **a**에 저장
  - [Ex] `b(1,1) = 1;` → "1"이 행렬 **b**의 (1,1)에 저장
- Transpose
  - (')붙임
  - [Ex] `A'`
- `diag()`
  - main diagonal을 얻음
  - [Ex] `diag(A)`



## 변수 선언 (Cont'd)

The screenshot displays the MATLAB environment with the following components:

- Workspace:** A table showing variables 'a' and 'b', both 1x1 double arrays. Variable 'b' is highlighted with a red box.
- Command Window:** Shows the execution of commands: `>>`, `>>`, `>> a=1`, and `>> b=1;`. The output for `a` is `1`. The command `>>` is highlighted with a red box.
- Command History:** Lists previous commands, including `a(1,1)=1`, `a=[1 2 3]`, `clear all`, and `b=[1 2 3; 4 5 6; 7 8 9]`.

Name	Size	Bytes	Class
a	1x1	8	double array
b	1x1	8	double array

```
>>  
>>  
>> a=1  
  
a =  
  
    1  
  
>>  
>>  
>> b=1;  
>>
```





## 변수 선언 (Cont'd)

The screenshot displays the MATLAB environment with the following components:

- Workspace:** A table showing variables 'a' and 'b'. Variable 'a' is a 1x3 double array (24 bytes), and variable 'b' is a 3x3 double array (72 bytes). Both are highlighted with a red box.
- Command Window:** Shows the execution of commands: `a=[1 2 3]` and `b=[1 2 3; 4 5 6; 7 8 9]`. The output for 'a' is `1 2 3` and for 'b' is `1 2 3; 4 5 6; 7 8 9`. The output for 'b' is highlighted with a red box.
- Command History:** Lists the commands entered, including `help cos`, `help abs`, `lookfor angle`, `lookfor fft`, `help format`, `a(1,1)=1`, `a=[1 2 3]`, `clear all`, and the final assignment `b=[1 2 3; 4 5 6; 7 8 9]`.

Name	Size	Bytes	Class
a	1x3	24	double array
b	3x3	72	double array

```
>>
>>
>> a=[1 2 3]

a =

     1     2     3

>>
>>
>> b=[1 2 3; 4 5 6; 7 8 9]

b =

     1     2     3
     4     5     6
     7     8     9

>>
```



# who/whos

The image shows a MATLAB interface with three main windows: Workspace, Command Window, and Command History.

**Workspace Window:** Displays a table of variables in the current workspace. A red box highlights the variables 'a' and 'b'.

Name	Size	Bytes	Class
a	1x3	24	double array
b	3x3	72	double array

**Command Window:** Shows the execution of MATLAB commands. A red box highlights the output of the 'whos' command.

```
>> a=[1 2 3]
a =
     1     2     3
>> b=[1 2 3; 4 5 6; 7 8 9]
b =
     1     2     3
     4     5     6
     7     8     9
>> who
Your variables are:
a b
>> whos
Name      Size      Bytes    Class
a         1x3         24    double array
b         3x3         72    double array
Grand total is 12 elements using 96 bytes
```

**Command History Window:** Shows a list of previously executed commands, including the 'whos' command.

```
%-- 9/17/04 2:43 PM --%
a=[1 2 3; 4 5 6]
k=length(a,1)
%-- 9/17/04 2:44 PM --%
a=[1 2 3; 4 5 6]
k=length(a)
whos(a)
who(a)
who
whos
%-- 9/17/04 2:52 PM --%
a=[1 2 3]
b=[1 2 3; 4 5 6; 7 8 9]
who
whos
```



## 변수에 저장된 값 확인

The screenshot displays the MATLAB environment. The **Workspace** window on the left lists variables **a** (1x3) and **b** (3x3). The **Array Editor: a** window is open, showing the values of the 1x3 array **a** in a grid. The **Size** field indicates 1 by 3. The **Command History** window shows the commands used to create and inspect the array.

**Array Editor: a**

	1	2	3
1	1	2	3

**Workspace**

Name	Size
a	1x3
b	3x3

**Command History**

```
%-- 9/17/04 2:43 PM --%  
a=[1 2 3; 4 5 6]  
k=length(a,1)  
%-- 9/17/04 2:44 PM --%  
a=[1 2 3; 4 5 6]  
k=length(a)  
whos(a)  
who(a)  
who  
whos  
%-- 9/17/04 2:52 PM --%  
a=[1 2 3]  
b=[1 2 3; 4 5 6; 7 8 9]  
who  
whos
```

**Variable Information**

Variable	Size	Bytes	Type
a	1x3	24	double array
b	3x3	72	double array

Grand total is 12 elements using 96 bytes

```
>> a
```



## Subscript

- ()와 ,로 표시, 첨자는 1부터 시작
  - Ex)  $A(2,1)$  : A의 2행 1열의 원소
- : operator
  - 시작:끝                      Ex)  $1:4 \rightarrow$                       1 2 3 4
  - 시작:증분:끝                Ex)  $5:-2:1 \rightarrow$                       5 3 1
  - 행 또는 열 전체            Ex)  $A(2,:) \rightarrow$                       A의 2행 전체
  - Ex)  $A(:,1) \rightarrow$                       A의 1열 전체
- size, length
  - 행렬의 크기를 보여줌



## 변수 선언 (Cont'd)

The screenshot displays the MATLAB environment with the following components:

- Workspace:** A table listing variables in the current workspace. Variable `b` is highlighted with a red box.
- Command Window:** Shows the execution of MATLAB commands. The output for `b` is highlighted with a red box.
- Command History:** Lists previously executed commands.

Name	Size	Bytes	Class
a	3x3	72	double array
b	1x1	8	double array

```
>>  
>>  
>> a=[1 2 3; 4 5 6; 7 8 9]  
  
a =  
  
     1     2     3  
     4     5     6  
     7     8     9  
  
>> b=a(1,2)  
  
b =  
  
     2  
  
>>
```

Command History:

```
a=[1 2 3]  
%-- 9/16/04 11:52 PM --%  
a=[1 2 3]  
b=[1 2 3; 4 5 6; 7 8 9]  
%-- 9/17/04 12:03 AM --%  
a=1  
%-- 9/17/04 12:03 AM --%  
a=1  
b=1;  
%-- 9/17/04 12:59 AM --%  
a=[1 2 3; 4 5 6; 7 8 9]  
b=a(1,2)  
%-- 9/17/04 1:00 AM --%  
a=[1 2 3; 4 5 6; 7 8 9]  
b=a(1,2)
```



## 변수 선언 (Cont'd)

The screenshot displays the MATLAB environment with three main panels:

- Workspace:** A table listing variables in the current workspace. Variable **f** is highlighted with a red box.
- Command Window:** Shows the execution of MATLAB commands and their outputs. The output for `f = a(2:3,:)` is highlighted with a red box.
- Command History:** A log of previously executed commands.

Name	Size	Bytes	Class
a	3x3	72	double array
c	1x3	24	double array
d	3x1	24	double array
e	2x2	32	double array
f	2x3	48	double array

```
>>
>> c=a(2,:)

c =

     4     5     6

>>
>> d=a(:,3)

d =

     3
     6
     9

>> e=a(2:3,1:2)

e =

     4     5
     7     8

>> f=a(2:3,:)

f =

     4     5     6
     7     8     9

>>
```

Command History:

```
%-- 9/17/04 12:03 AM --%
a=1
b=1;
%-- 9/17/04 12:59 AM --%
a=[1 2 3; 4 5 6; 7 8 9]
b=a(1,2)
%-- 9/17/04 1:00 AM --%
a=[1 2 3; 4 5 6; 7 8 9]
b=a(1,2)
%-- 9/17/04 1:09 AM --%
a=[1 2 3; 4 5 6; 7 8 9]
c=a(2,:)
d=a(:,3)
e=a(2:3,1:2)
f=a(2:3,:)
```



## 증감연산

The MATLAB interface displays the following workspace variables:

Name	Size	Bytes	Class
k	1x4	32	double array
x	1x10	80	double array

The Command Window shows the following commands and output:

```
>> Using Toolbox Path Cache. Type "help toolbox_path_cache" for more info.  
>> To get started, select "MATLAB Help" from the Help menu.  
>>  
>> clear all  
>> x=1:10  
  
x =  
  
     1     2     3     4     5     6     7     8     9    10  
  
>> k=-5:3:5  
  
k =  
  
    -5    -2     1     4  
  
>>
```

The Command History window shows the following commands:

```
a=[1 2 3; 4 5 6; 7 8 9]  
b=a(1,2)  
%-- 9/17/04 1:00 AM --%  
a=[1 2 3; 4 5 6; 7 8 9]  
b=a(1,2)  
%-- 9/17/04 1:09 AM --%  
a=[1 2 3; 4 5 6; 7 8 9]  
c=a(2,:)   
d=a(:,3)  
e=a(2:3,1:2)  
f=a(2:3,:)   
%-- 9/17/04 1:46 AM --%  
clear all  
x=1:10  
k=-5:3:5
```



# size

The MATLAB interface displays the following components:

- Workspace:** A table listing variables in the workspace.
- Command Window:** Shows the execution of the `size` function.
- Command History:** Shows the sequence of commands entered in the Command Window.

Name	Size	Bytes	Class
a	2x3	48	double array
b	1x2	16	double array
c	1x1	8	double array
d	1x1	8	double array
m	1x1	8	double array
n	1x1	8	double array

```
>> b=size(a)
b =
     2     3

>> [m n]=size(a)
m =
     2
n =
     3

>> c=size(a,1)
c =
     2

>> d=size(a,2)
d =
     3

>>
```

```
c=a(2,:)
d=a(:,3)
e=a(2:3,1:2)
f=a(2:3,:)
%-- 9/17/04 1:46 AM --%
clear all
x=1:10
k=-5:3:5
%-- 9/17/04 2:21 PM --%
clear all
a=[1 2 3; 4 5 6]
b=size(a)
[m n]=size(a)
c=size(a,1)
d=size(a,2)
```





# length

**Workspace**

Name	Size	Bytes	Class
a	2x3	48	double array
k	1x1	8	double array

**Command Window**

```
Using Toolbox Path Cache. Type "help toolbox_path_cache" for more info.  
To get started, select "MATLAB Help" from the Help menu.  
  
>> a=[1 2 3; 4 5 6]  
  
a =  
  
     1     2     3  
     4     5     6  
  
>> k=length(a)  
  
k =  
  
     3  
  
>>
```

**Command History**

```
x=1:10  
k=-5:3:5  
%-- 9/17/04 2:21 PM --%  
clear all  
a=[1 2 3; 4 5 6]  
b=size(a)  
[m n]=size(a)  
c=size(a,1)  
d=size(a,2)  
%-- 9/17/04 2:43 PM --%  
a=[1 2 3; 4 5 6]  
k=length(a,1)  
%-- 9/17/04 2:44 PM --%  
a=[1 2 3; 4 5 6]  
k=length(a)
```



---

## **MATLAB as a Calculator**



## 연산

---

- 산술연산자
  - **【주의】** 기본적으로 행렬 연산 수행
  - (.산술연산자) : element to element 연산
- 관계연산자
- 논리연산자



## 산술 연산자

Operator	기능
+	덧셈
-	뺄셈
*	곱셈
/	나눗셈 ( $A/B \rightarrow A \times \text{역행렬} B$ )
\	왼쪽나눗셈 ( $A \setminus B \rightarrow \text{역행렬} A \times B$ )
^	거듭제곱
,	<b>Complex conjugate transpose</b>
()	계산순서

### • 우선순위

우선 순위	연산	Matlab 수식
1	괄호	( )
2	지수	^
3	곱하기, 나누기	*, /
4	더하기, 빼기	+, -



## 산술 연산 – 덧셈, 뺄셈

The screenshot shows the MATLAB environment with three main panels:

- Workspace:** Displays variables `x`, `y`, and `z`, all of size 2x2 and class double array.
- Command Window:** Shows the execution of commands: `clear all`, `x=[1 2; 3 4]`, `y=[5 6; 7 8]`, `z=x+y`, and `z=x-y`. The result of `z=x-y` is highlighted with a red box.
- Command History:** Lists the commands entered, including `whos(a)`, `who(a)`, `who`, `whos`, `%-- 9/17/04 2:52 PM --%`, `a=[1 2 3]`, `b=[1 2 3; 4 5 6; 7 8 9]`, `clear all`, `x=[1 2; 3 4]`, `y=[5 6; 7 8]`, `z=x+y`, and `z=x-y`.

The Command Window output for `z=x-y` is:

```
>> z=x-y  
z =  
-4 -4  
-4 -4
```



## 산술 연산 - 곱셈

Workspace

Name	Size	Bytes	Class
k	2x2	32	double array
x	2x2	32	double array
y	2x2	32	double array
z	2x2	32	double array

Command Window

```
To get started, select "MATLAB Help" from the Help menu.  
>> x=[1 2; 3 4]  
x =  
    1    2  
    3    4  
>> y=[5 6; 7 8]  
y =  
    5    6  
    7    8  
>> z=x*y  
z =  
    19    22  
    43    50  
>> k=x.*y  
k =  
    5    12  
   21    32  
>>
```

Command History

```
b=[1 2 3; 4 5 6; 7 8 9]  
who  
whos  
%-- 9/17/04 3:36 PM --%  
clear all  
x=[1 2; 3 4]  
y=[5 6; 7 8]  
z=x+y  
z=x-y  
z=x*y  
%-- 9/17/04 3:49 PM --%  
x=[1 2; 3 4]  
y=[5 6; 7 8]  
z=x+y  
k=x.*y
```

\*앞에 .을 붙이면( .\* )  
element 곱셈



## 산술 연산 – 곱셈 (Cont'd)

The screenshot shows the MATLAB environment with the following components:

- Workspace:** Lists variables `a` (2x3), `b` (3x2), and `z` (2x2), all as double arrays.
- Command Window:** Shows the execution of `a = [1 2 3; 4 5 6]`, `b = [5 6 7; 8 9 10]`, and `z = a*b`. An error message is displayed: `??? Error using ==> ± Inner matrix dimensions must agree.` This occurs because the dimensions of `a` and `b` are incompatible for multiplication. The final output for `z` is shown in a red box: `z = [38 56; 92 137]`.
- Command History:** Lists the commands entered: `a=b`, `b=[4 5 6; 7 8 9]`, `z=a*b`, `z=a/b`, `b'`, `b=b'`, `z=a*b`, `z=a.*b`, `%-- 9/17/04 6:03 PM --%`, `clear all`, `a=[1 2 3; 4 5 6]`, `b=[5 6 7; 8 9 10]`, `z=a*b`, `b=b'`, and `z=a*b`.



## 산술 연산 - 나눗셈(Cont'd)

The MATLAB interface is shown with the following components:

- Workspace:** Displays three variables: `a`, `b`, and `c`, all of size `2x2` and class `double array`.
- Command Window:** Shows the execution of the following commands:

```
>> clear all
>>
>> a=[1 2; 3 4]

a =

     1     2
     3     4

>> b=[5 6; 7 8]

b =

     5     6
     7     8

>> c=a/b

c =

    3.0000   -2.0000
    2.0000   -1.0000

>>
```

The result for `c` is highlighted with a red box.
- Command History:** Shows the sequence of commands entered, including `clear all`, `a=[1 2 3; 4 5 6]`, `help fft`, and the current session's commands.





## Matrix 함수

- eye
  - 단위 행렬 ex) eye(3)
- ones
  - 1로 채워진 행렬 ex) ones(3), ones(1,2)
- zeros
  - 0으로 채워진 행렬 ex) zeros(3), zeros(1,2)
- rand / randn
  - Uniform / Normal random값 행렬 ex) rand(3), randn(2,4)
- det
  - determinant ex) det(A)
- inv
  - 역행렬 ex) inv(A)
- triu, tril
  - 상삼각, 하삼각행렬 ex) triu(A), tril(A)



## Matrix 함수 예

**Workspace**

Name	Size	Bytes	Class
a	2x2	32	double array
b	1x3	24	double array
c	2x3	48	double array

**Command Window**

```
>>  
>>  
>> clear all  
>> a=eye(2)  
a =  
    1    0  
    0    1  
>> b=ones(1,3)  
b =  
    1    1    1  
>> c=zeros(2,3)  
c =  
    0    0    0  
    0    0    0  
>>
```

**Command History**

```
a=2  
b=4  
c=a/b  
%-- 9/20/04 1:27 AM --%  
clear all  
a=[1 2; 3 4]  
b=[5 6; 7 8]  
c=a/b  
clear all  
a=eye(2)  
b=ones(1,3)  
c=zeros(2,3)
```

단위 행렬, ( ) 숫자 = 행렬 크기  
[ex] eye(2) -> 2 by 2 행렬

1로만 채워진 행렬 만들기  
[ex] ones(1,3) -> 1by3 행렬

0로만 채워진 행렬 만들기  
[ex] zeros(2,3) -> 2by3 행렬



# Save: Mat Files

---

## save

Save workspace variables to file

### Syntax

```
save(filename)
save(filename, variables)
save(filename, '-struct', structName, fieldNames)
save(filename, ..., '-append')
save(filename, ..., format)
save(filename, ..., version)
save filename ...
```

### Description

`save(filename)` stores all variables from the current workspace in a MATLAB formatted binary file (MAT-file) called `filename`.

`save(filename, variables)` stores only the specified variables.



# Load : Mat Files

---

## load

Load data from MAT-file into workspace

### Syntax

```
S = load(filename)
S = load(filename, variables)
S = load(filename, '-mat', variables)
S = load(filename, '-ascii')
load(filename, ...)
load filename ...
```

### Description

`S = load(filename)` loads the variables from a MAT-file into a structure array, or data from an ASCII file into a double-precision array.

`S = load(filename, variables)` loads the specified variables from a MAT-file.

`S = load(filename, '-mat', variables)` forces load to treat the file as a MAT-file, regardless of the extension. Specifying `variables` is optional.

`S = load(filename, '-ascii')` forces load to treat the file as an ASCII file, regardless of the extension.



---

## **Graphics & Visualization**



## Plot

- **plot (x values, y values, 'style-option')**

색	선 종류
<b>y (yellow)</b>	<b>- (solid)</b>
<b>m (magenta)</b>	<b>-- (dashed)</b>
<b>c (cyan)</b>	<b>: (dotted)</b>
<b>r (red)</b>	<b>-. (dash-dot)</b>
<b>g (green)</b>	<b>. (point)</b>
<b>b (blue)</b>	<b>e (circle)</b>
<b>w (white)</b>	<b>x (x-mark)</b>
<b>k (black)</b>	<b>+ (plus)</b>
	<b>* (star)</b>



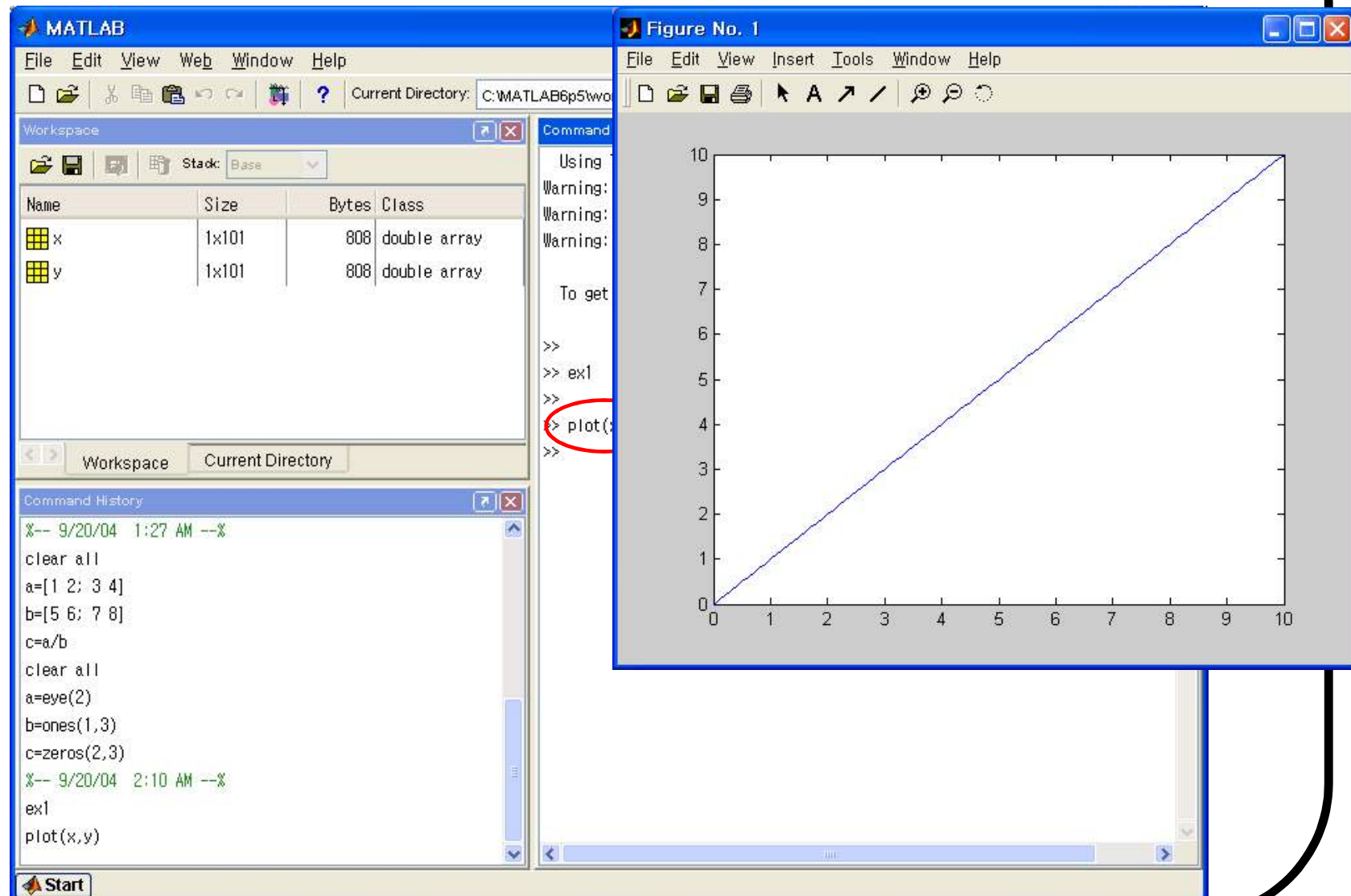
## Graphs Options

---

- `figure` : 새로운 그림창 생성
- `xlabel`, `ylabel` : x, y축에 이름 달기
- `title` : 표 제목 달기
- `legend` : 범례달기
- `subplot` : 그림창 분할
- `grid` : 격자표시
- `axis` : 표시영역 설정
- `clf` or `clear all` : Clear figure
- `hold on`, `hold off` : Hold current graph



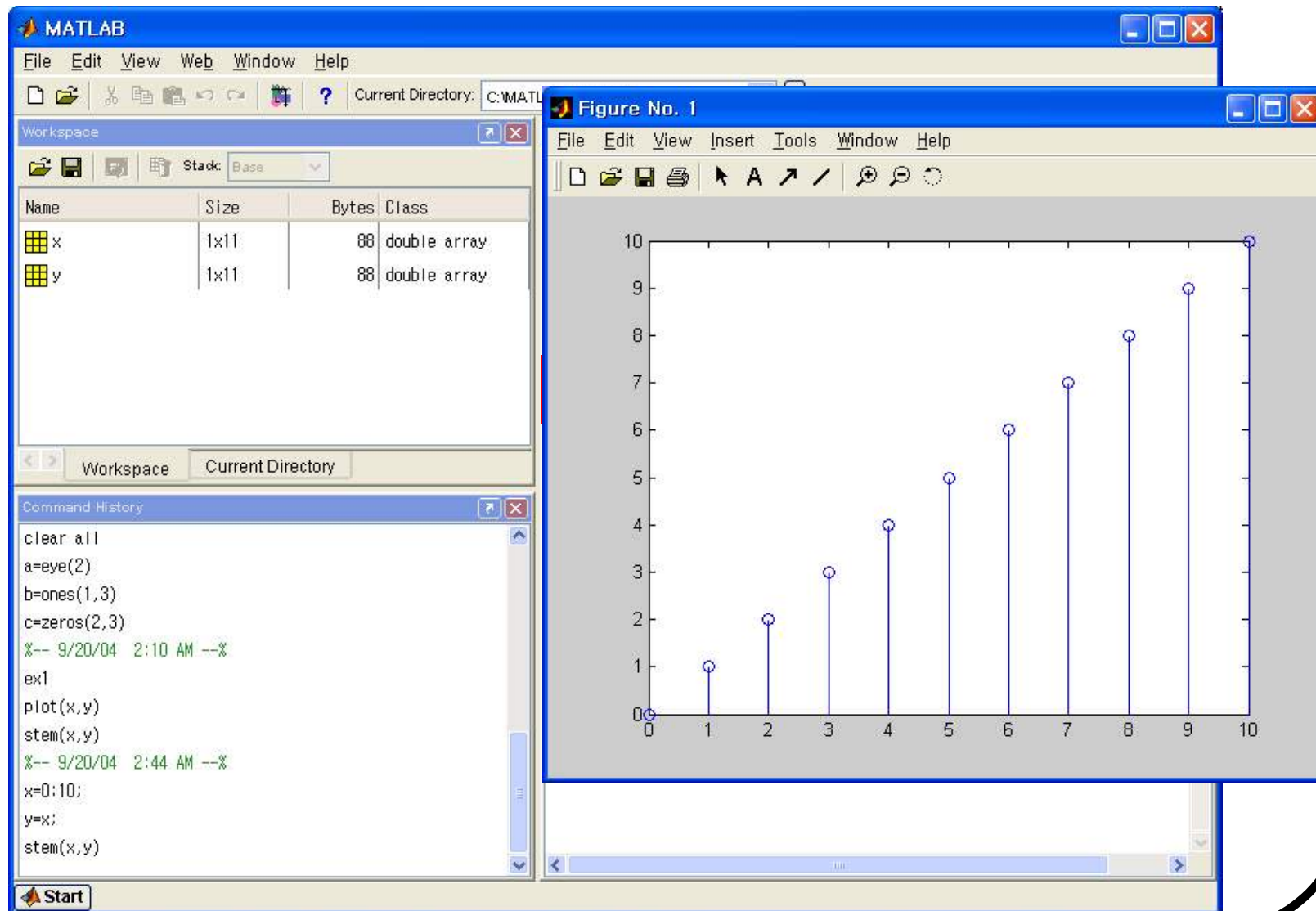
## Graph 출력 예 - plot







## Graph 출력 예 - stem





# 3D Graphics: Functions

- <http://www.mathworks.co.kr/help/techdoc/ref/f16-8867.html>

## R2011b Documentation → MATLAB

View documentation for other releases

Learn more about MATLAB

### 3-D Visualization

Surface and Mesh Plots	Plot matrices, visualize functions of two variables, specify colormap
View Control	Control the camera viewpoint, zooming, rotation, aspect ratio, set axis limits
Lighting	Add and control scene lighting
Transparency	Specify and control object transparency
Volume Visualization	Visualize gridded volume data

▲ Back to Top of Section

### Surface and Mesh Plots

Surface and Mesh Creation	Visualizing gridded and triangulated data as lines and surfaces
Domain Generation	Gridding data and creating arrays
Color Operations	Specifying, converting, and manipulating color spaces, colormaps, colorbars, and backgrounds

### Surface and Mesh Creation

<code>hidden</code>	Remove hidden lines from mesh plot
<code>meshc</code>	Plot a contour graph under mesh graph
<code>meshz</code>	Plot a curtain around mesh plot
<code>peaks</code>	Example function of two variables
<code>surface</code>	Create surface object
<code>surf</code>	Contour plot under a 3-D shaded surface plot
<code>surf</code>	Surface plot with colormap-based lighting
<code>tetramesh</code>	Tetrahedron mesh plot
<code>trimesh</code>	Triangular mesh plot
<code>triplot</code>	2-D triangular plot
<code>trisurf</code>	Triangular surface plot



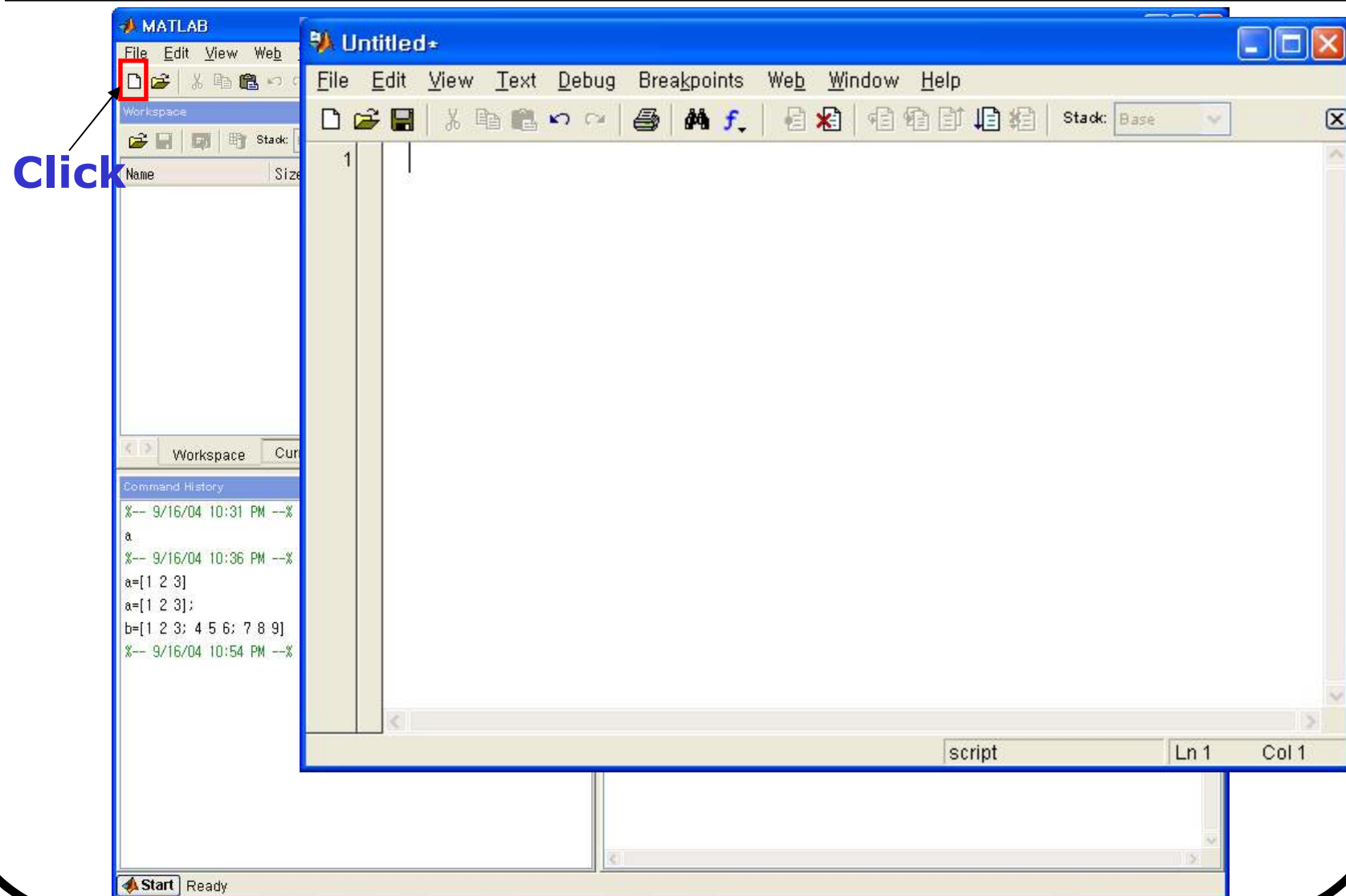
## .M File

---

- MATLAB 명령들의 집합 - 프로그램
- Script mode
  - 연속적인 MATLAB 명령들
- Function mode
  - 입출력 매개변수 이용, 파일명과 함수명 일치
  - Subfunction 사용 가능
- 주석은 %로 단다
  - 처음 열린 `lookfor` 검색 시 출력
  - `help` 검색 시 함수 정의 부와 MATLAB 명령어 사이의 주석문 출력

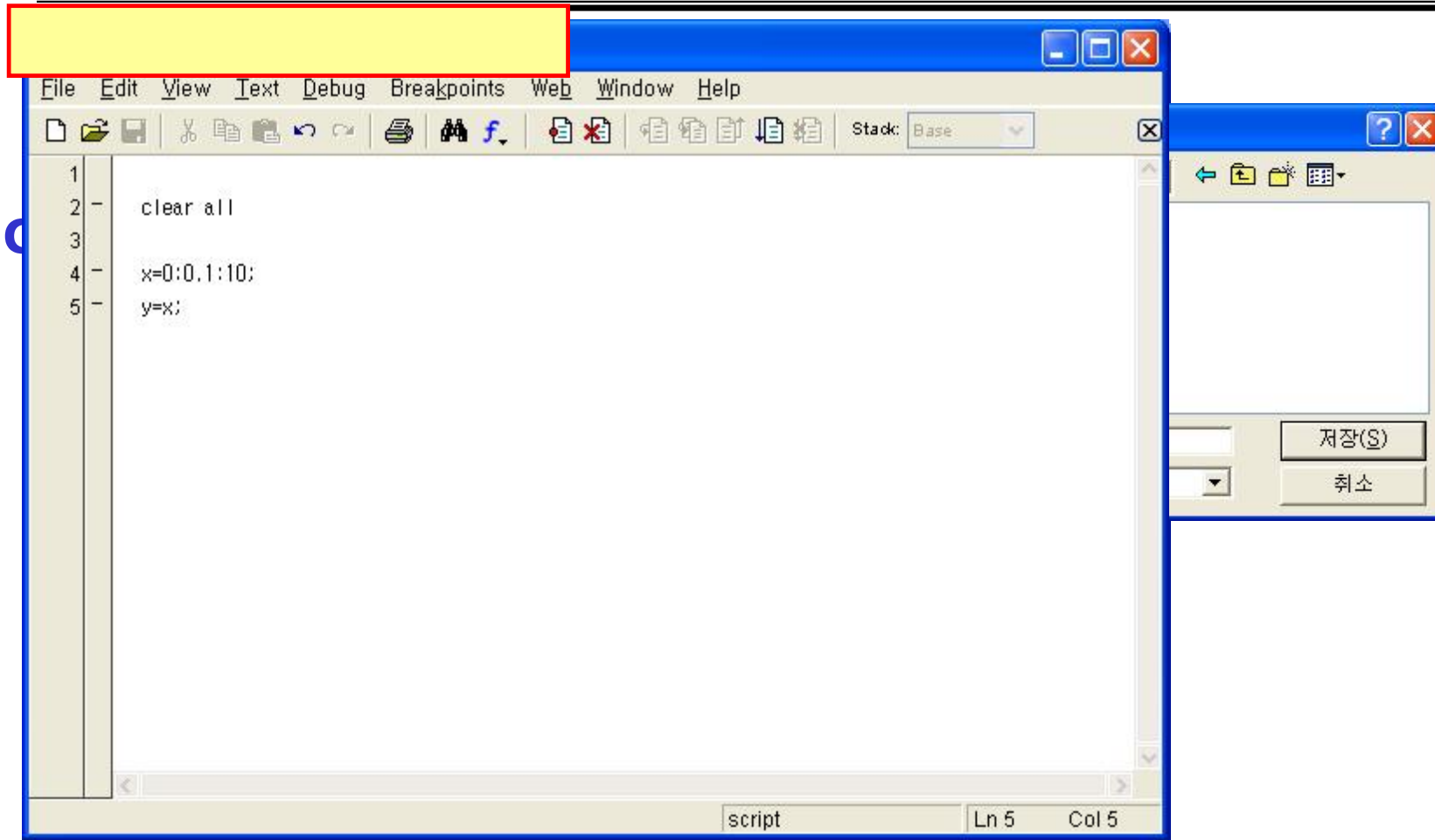


## M-File Editor/Debugger 열기





## M-File 작성 및 저장





## M-File 경로 설정 및 실행

The screenshot shows the MATLAB environment. The **Workspace** window displays two variables, `x` and `y`, both of size `1x101` and type `double array`. The **Command Window** shows the execution of `ex1`, which is circled in red. An arrow points from this command to the text "저장했던 M-File". The **Command History** window shows the sequence of commands executed, including `clear all`, `a=[1 2; 3 4]`, `b=[5 6; 7 8]`, `c=a/b`, `clear all`, `a=eye(2)`, `b=ones(1,3)`, `c=zeros(2,3)`, and `ex1`.

Name	Size	Bytes	Class
x	1x101	808	double array
y	1x101	808	double array

```
>>  
>> ex1  
>>
```

→ 저장했던 M-File





# Built-in MATLAB Functions (i.e., Libraries)

- MATLAB Built-in Functions = MATLAB Libraries = MATLAB Toolboxes

## MATLAB

MATLAB  
MATLAB Builder EX  
MATLAB Builder JA  
MATLAB Builder NE  
MATLAB Coder  
MATLAB Compiler  
MATLAB Distributed Computing Server  
MATLAB Report Generator  
Embedded Coder  
Parallel Computing Toolbox  
SimBiology  
SystemTest

## MATLAB Toolboxes

Aerospace Toolbox	Image Acquisition Toolbox
Bioinformatics Toolbox	Image Processing Toolbox
Communications System Toolbox	Instrument Control Toolbox
Computer Vision System Toolbox	Mapping Toolbox
Control System Toolbox	Model-Based Calibration Toolbox
Curve Fitting Toolbox	Model Predictive Control Toolbox
Data Acquisition Toolbox	Neural Network Toolbox
Database Toolbox	OPC Toolbox
Datafeed Toolbox	Optimization Toolbox
DSP System Toolbox	Partial Differential Equation Toolbox
Econometrics Toolbox	Phased Array System Toolbox
EDA Simulator Link	RF Toolbox
Filter Design HDL Coder	Robust Control Toolbox
Financial Toolbox	Signal Processing Toolbox
Financial Derivatives Toolbox	Spreadsheet Link EX
Fixed-Income Toolbox	Statistics Toolbox
Fixed-Point Toolbox	Symbolic Math Toolbox
Fuzzy Logic Toolbox	System Identification Toolbox
Global Optimization Toolbox	Vehicle Network Toolbox
	Wavelet Toolbox

## Simulink

Simulink  
DO Qualification Kit  
IEC Certification Kit  
Real-Time Windows Target  
SimDriveline  
SimElectronics  
SimEvents  
SimHydraulics  
SimMechanics  
SimPowerSystems  
SimRF  
Simscape  
Simulink 3D Animation  
Simulink Code Inspector  
Simulink Coder  
Simulink Control Design  
Simulink Design Optimization  
Simulink Design Verifier  
Simulink Fixed Point  
Simulink HDL Coder  
Simulink PLC Coder  
Simulink Report Generator  
Simulink Verification and Validation  
Stateflow  
xPC Target



# Built-in MATLAB Functions (i.e., Libraries)

## MATLAB

MATLAB  
MATLAB Builder EX  
MATLAB Builder JA  
MATLAB Builder NE  
MATLAB Coder  
MATLAB Compiler  
MATLAB Distributed Computing Server  
MATLAB Report Generator  
Embedded Coder  
Parallel Computing Toolbox  
SimBiology  
SystemTest

MATLAB®

### Functions:

- By Category
- Alphabetical List

### Handle Graphics:

- Object Properties

#### Desktop Tools and Development Environment

Startup, Command Window, help, editing and debugging, tuning, other general functions

#### Data Import and Export

General and low-level file I/O, plus specific file formats, like audio, spreadsheet, HDF, images

#### Mathematics

Arrays and matrices, linear algebra, other areas of mathematics

#### Data Analysis

Basic data operations, descriptive statistics, covariance and correlation, filtering and convolution, numerical derivatives and integrals, Fourier transforms, time series analysis

#### Programming and Data Types

Function/expression evaluation, program control, function handles, object oriented programming, error handling, operators, data types, dates and times, timers

#### Object-Oriented Programming

Functions for working with classes and objects

#### Graphics

Line plots, annotating graphs, specialized plots, images, printing, Handle Graphics

#### 3-D Visualization

Surface and mesh plots, view control, lighting and transparency, volume visualization

#### GUI Development

GUIDE, programming graphical user interfaces

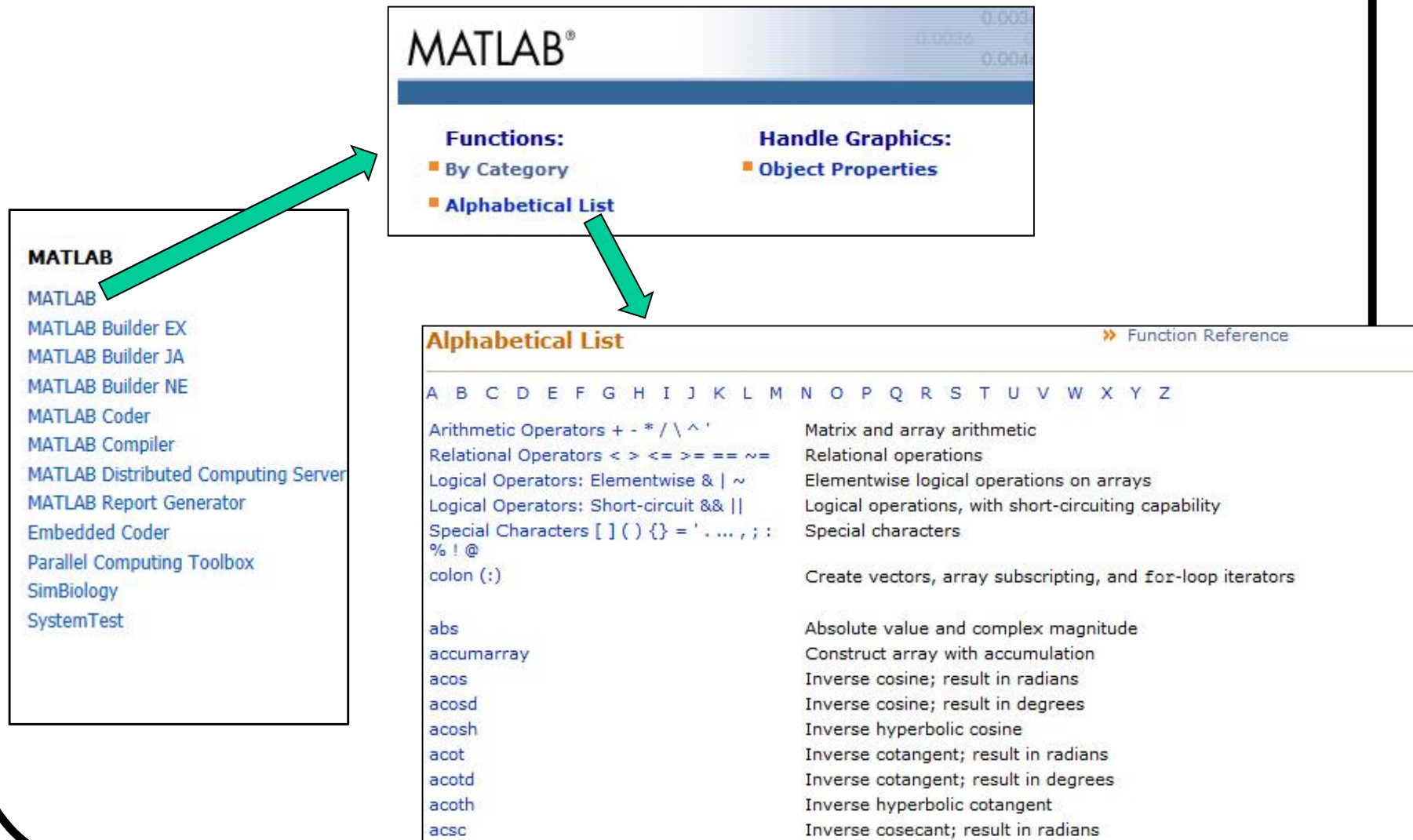
#### External Interfaces

Interfaces to shared libraries, Java, .NET, COM and ActiveX, Web services, and serial port devices, and C and Fortran routines

## Descriptive Statistics

<code>corrcoef</code>	Correlation coefficients
<code>cov</code>	Covariance matrix
<code>max</code>	Largest elements in array
<code>mean</code>	Average or mean value of array
<code>median</code>	Median value of array
<code>min</code>	Smallest elements in array
<code>mode</code>	Most frequent values in array
<code>std</code>	Standard deviation
<code>var</code>	Variance







## 기본 함수들

---

- 기본
  - 그래프: plot, axis, subplot, surface, figure, image,...
  - 명령어: help, who, save, load,...
  - 통계함수: sum, mean, sort, cov,...
  - 수학함수: abs, sin, cos, tan, sqrt, acos, angle, exp, gcd, lcm, real, imag, log, log10,...
- 기타 유용한 명령어
  - Diary file name(.out), diary on, diary off – log 파일 기록
  - linspace, logspace – linear, log 공간 생성
  - eval, feval – 문자열 실행, 함수 실행
  - echo, disp, input



# User-defined Functions

---

- How to Write My Own Functions

## function

Declare function

### Syntax

```
function [out1, out2, ...] = myfun(in1, in2, ...)
```

### Description

`function [out1, out2, ...] = myfun(in1, in2, ...)` declares the function `myfun`, and its inputs and outputs. The function declaration must be the first executable line of any MATLAB function.

### Example 1

The existence of a file on disk called `stat.m` containing this code defines a new function called `stat` that calculates the mean and standard deviation of a vector:

```
function [mean,stdev] = stat(x)
n = length(x);
mean = sum(x)/n;
stdev = sqrt(sum((x-mean).^2/n));
```

Call the function, supplying two output variables on the left side of the equation:

```
[mean stdev] = stat([12.7 45.4 98.9 26.6 53/1])
mean =
    47.3200
stdev =
    29.4085
```



## User-defined Functions (Cont.)

---

### Example 2

avg is a subfunction within the file stat.m:

```
function [mean,stdev] = stat2(x)
n = length(x);
mean = avg(x,n);
stdev = sqrt(sum((x-avg(x,n)).^2)/n);

function mean = avg(x,n)
mean = sum(x)/n;
```

Call the function and compare the answer with that of Example 1, above:

```
[mean stdev] = stat2([12.7 45.4 98.9 26.6 53/1])
mean =
    47.3200
stdev =
    29.4085
```



---

## **Logical & Branch Operations & Loops**



## 관계 / 논리 연산자

### • 관계 연산자

Operator	기능
<	작다
<=	작거나 같다
>	크다
>=	크거나 같다
==	같다
~=	같지 않다

### • 논리 연산자

Operator	기능
&	and
	or
~	not



## 제어 명령

- if
  - if, elseif, end

### if/elseif/else

Execute statements if condition is true

#### Syntax

```
if expression
    statements
elseif expression
    statements
else
    statements
end
```

- switch
  - switch, otherwise, end

### switch/case/otherwise

Switch among several cases based on expression

#### Syntax

```
switch switch_expression
    case case_expression
        statements
    case case_expression
        statements
    :
    otherwise
        statements
end
```

- For

### for

Execute statements specified number of times

#### Syntax

```
for index = values
    program statements
end
```

- while

### while

Repeatedly execute statements while condition is true

#### Syntax

```
while expression
    statements
end
```



---

**What else?**





## Advanced Topics

- Toolboxes
- Graphical User Interfaces (GUI): guide
  - <http://www.mathworks.co.kr/help/techdoc/ref/f16-40727.html>

### GUI Development

Predefined Dialog Boxes	Dialog boxes for error, user input, waiting, etc.
User Interface Deployment	Open GUIs, create the handles structure
User Interface Development	Start GUIDE, manage application data, get user input
User Interface Objects	Create GUI components
Objects from Callbacks	Find object handles from within callbacks functions
GUI Utilities	Move objects, wrap text
Program Execution	Wait and resume based on user input

- Simulink



## Third-Party Products & Services

### Image Systems Evaluation Toolbox (ISET): Digital Camera Simulator

Simulates image scene, capture, processing, and display

#### Highlights

- Database of calibrated multispectral image data
- Industry-standard test targets and image quality metrics
- Design and analysis of scene and optical parameters
- Design and analysis of pixel and sensor parameters
- Intuitive, point-and-click graphical user interface
- Open programming interface for proprietary algorithms

#### Description

The ISET Digital Camera Simulator simulates the complete digital camera reproduction pipeline. It combines optical modeling and sensor simulation with image processing algorithms. Beginning with calibrated multispectral scene data, the user interactively explores how changes in the scene's physical parameters, imaging optics, sensor electronics, and image processing influence image quality.

Engineers use ISET to select, evaluate, and optimize a complete digital imaging pipeline, including optics, sensor components, and image processing algorithms. Image quality tradeoffs are assessed with the aid of many built-in metrics, such as optical modulation transfer functions (MTFs), pixel and sensor signal-to-noise ratio (SNR), and color tools based on international standards (CIE chromaticity coordinates, color matching functions, CIELAB, SCIELAB, and others). ISET uses a MATLAB® graphical user interface to provide users with easy methods for changing and evaluating scene and system parameters. A set of MATLAB® interface routines makes it convenient to access and replace the data at any stage in the pipeline. The open architecture of MATLAB® enables students and engineers to access physically realistic data from the simulator and test novel hardware and algorithms.



## Model-based Design Using Matlab & Simulink

---

- <http://www.mathworks.co.kr/videos/model-based-design-with-matlab-and-simulink-69040.html>