

2. Plot 기초

MATLAB의 큰 장점은 그래픽 툴이 내장되어 있어 데이터를 시각적으로 표현하기 편하다는 것이다. 그래픽 툴은 GUI에 필요한 다양한 작업이 몇 개의 함수로 해결할 수 있도록 구조적으로 만들어져 있다. 그러나 MATLAB이나 그래픽에 익숙한 사람이 아니면 쉽게 배우기 힘든 면도 있다. 여러 가지 handle을 이용한 작업을 해야 하는데, 이는 MATLAB의 구조를 어느 정도 알고 있어야 가능하기 때문이다. 따라서, 본 장에서는 간단한 2D 그래프를 그리는 수준에서 마무리하며, 더 공부하고 싶을 경우, <임종수의 MATLAB7>이나 <MATLAB 고급 GUI 개발>을 참조하기 바란다.

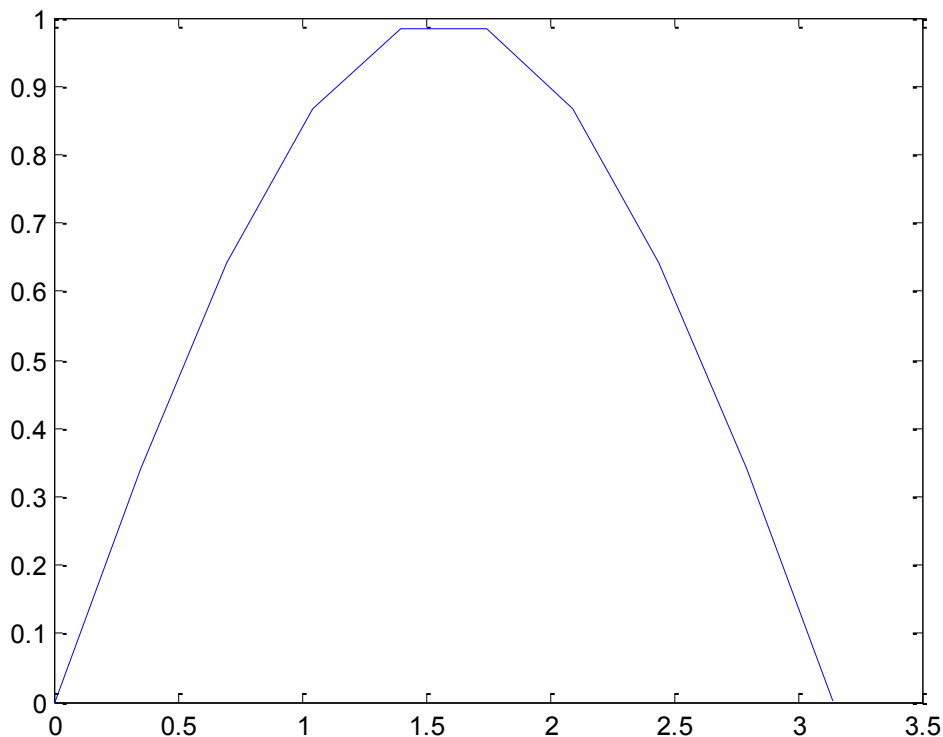
2.1. Plot 시작

함수 plot은 MATLAB에서 가장 기본적인 그래프 함수이다. 2차원 평면 상의 좌표를 입력 받아 차례대로 이어 그려준다. 첫 번째 인자와 두 번째 인자에는 주로 벡터를 입력하며, 크기(length)가 같아야 한다. 첫 번째 인자가 x축, 두 번째 인자가 y축에 해당한다. 다음은 사용 예이다.

예제2.1. plot 그리기

```
>> x = linspace(0, pi, 10);  
>> y = sin(x);  
>> plot(x, y)
```

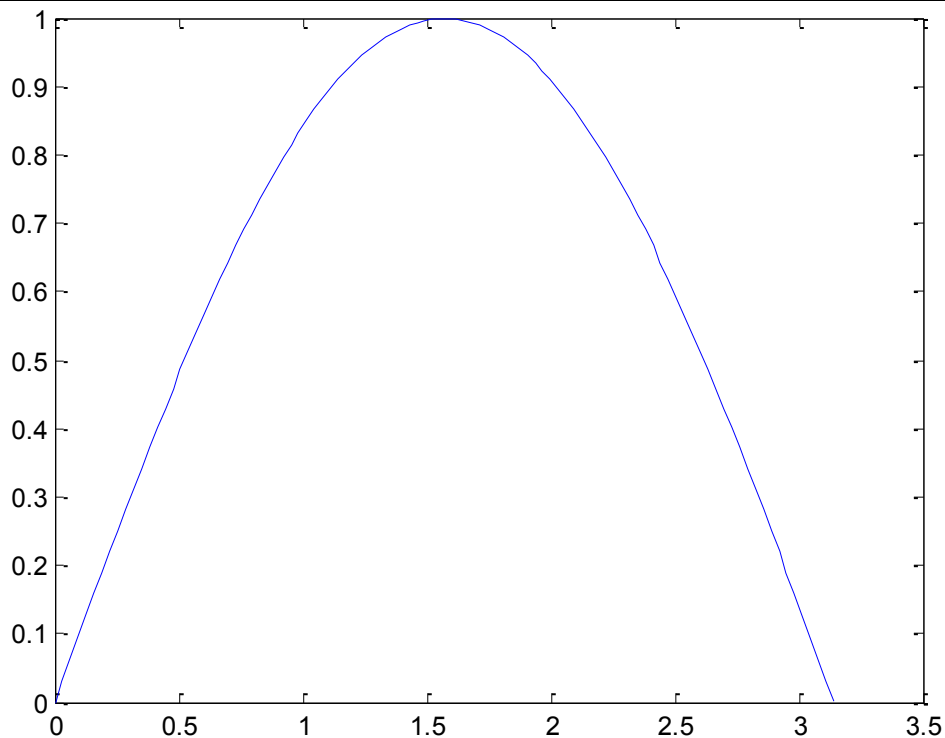
0에서 π 까지 10개의 점으로 나눈 후, 각 값에 sin을 구한다. 이를 plot으로 나타내었다.



Sin 그래프의 형태가 부드럽지 않고 꺾인 선이 보인다. 특히, $\pi/2$ 부근에서 값이 1에 도달하지 못하는 것을 확인할 수 있다. 이는 실제 sin 그래프를 그린 것이 아니라 각 x 값에 맞는 y 값을 계산한 후, 좌표를 차례대로 직선으로 이어 그렸기 때문이다.

```
>> x = linspace(0, pi, 100);
>> y = sin(x);
>> plot(x, y)
```

촘촘히 나타내기 위해 0과 π 사이를 100개의 점으로 나누어 그래프를 다시 그렸다.



10개의 점을 이었던 앞의 그래프와 달리, 곡선이 부드럽게 나타난 것을 확인할 수 있다. 점의 개수를 늘리면 더욱 실제에 가까운 그래프를 그릴 수 있을 것이다.

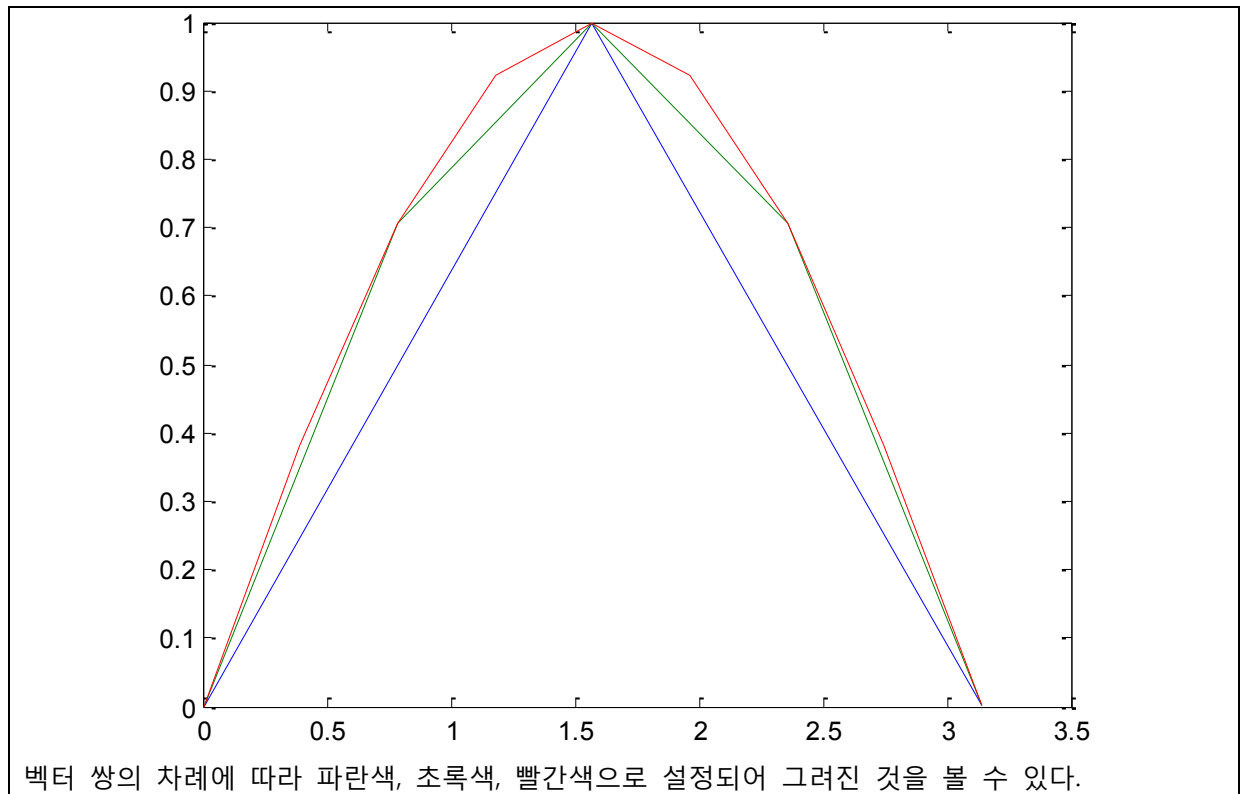
그래프를 그릴 때, 여러 데이터를 동시에 그리는 경우가 있다. 이 때, 좌표를 나타내는 벡터 쌍 x , y 를 `plot(x1, y1, x2, y2, x3, y3, ...)`와 같이 차례대로 입력하면 한 그래프에 동시에 그려준다.

예제2.2. plot 겹쳐 그리기1

```
>> x1 = linspace(0, pi, 3);
>> y1 = sin(x1);
>> x2 = linspace(0, pi, 5);
>> y2 = sin(x2);
>> x3 = linspace(0, pi, 9);
>> y3 = sin(x3);
>> plot(x1, y1, x2, y2, x3, y3)
```

0에서 π 사이를 점 3개, 5개, 9개로 나누어 계산한다.

벡터 쌍을 차례대로 인자로 넣으면 그래프를 겹쳐 그릴 수 있다.



경우에 따라 미리 그려진 그래프에 새로운 데이터를 겹쳐 보여야 할 때가 있다. 그러나 plot 함수는 실행할 때마다 이전 창을 지운 후 새로 그리도록 설정되어 있다. 이럴 때, hold on이라는 명령을 사용한다. 명령어 hold on을 사용하면 hold off를 입력하기 전까지 새로 그리는 모든 plot이 기존의 plot 위에 겹쳐 나타난다.

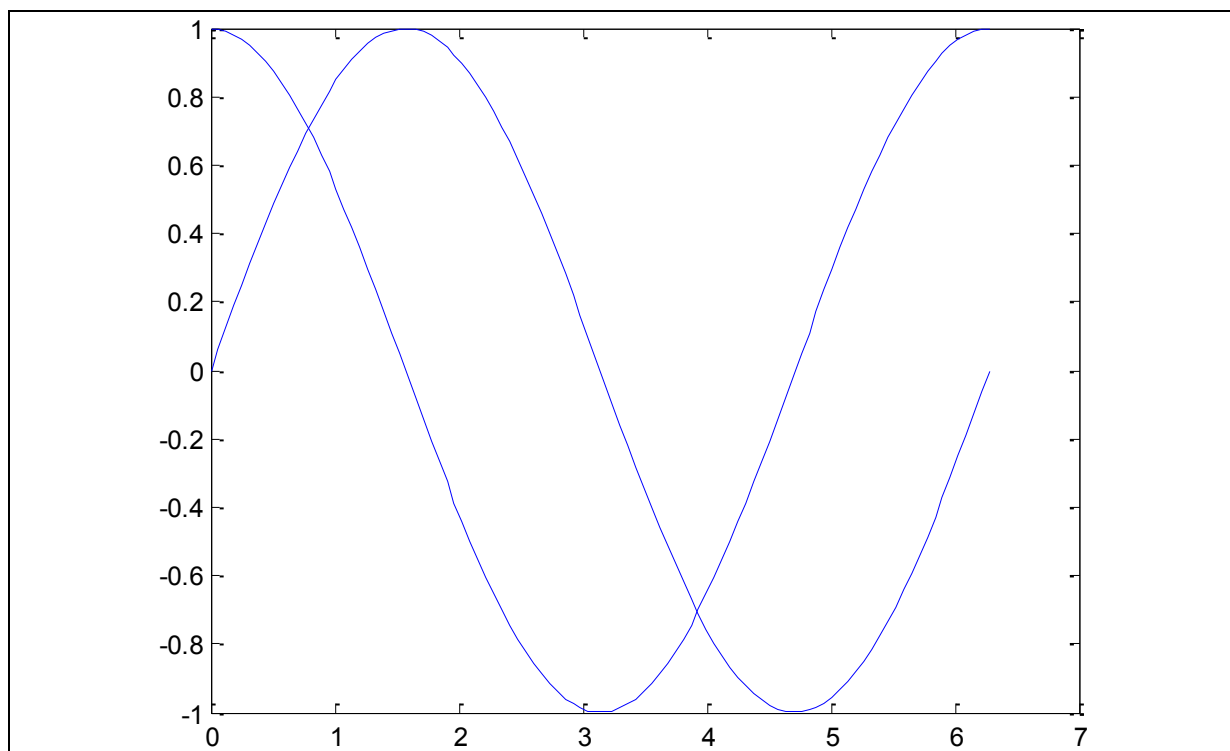
예제2.3. plot 겹쳐 그리기2

```
>> x = linspace(0, 2*pi, 100);
>> y1 = sin(x);
>> y2 = cos(x);
>> plot(x, y1)
>> hold on
>> plot(x, y2)
>> hold off
```

0에서 2π 까지의 sin과 cos을 계산한 후, 두 그래프를 겹쳐 그린다.

Hold on을 입력한 후의 plot은 원래 그래프에 겹쳐 그려진다.

겹침 모드를 해제한다.



함수 plot을 이용한 그래프는 실선이 기본이며, plot에 입력한 차례대로 파란색, 초록색, 빨간색 등의 색으로 그려진다. 그래서 앞에서 hold on 명령을 이용하여 두 plot을 겹칠 경우, 둘 다 함수의 첫 번째 인자로 사용되었기에 예제2.3의 그래프와 같이 파란색으로 그려진다. 이는 혼란을 가져올 소지가 많으므로, 각 그래프의 색상을 따로 지정해줄 필요가 있다. 또한, 흑백 인쇄물일 경우, 색상 대신 점선이나 파선 등으로 구별해줘야 할 때가 있으며, 각 데이터를 명확히 하기 위해 좌표에 마커를 추가할 수도 있다.

```
>> plot( x1, y1, '...', x2, y2, '...' )
```

위와 같이 각 벡터 쌍 다음 인자에 문자열 '...'에 적절한 문자를 삽입함으로써 원하는 선 스타일로 그래프를 그릴 수 있다. 다음은 선 스타일을 위한 문자 표이다. 차례에 관계없이 원하는 속성을 문자열 내에 입력하면 된다. 선 형태의 경우, 관련 문자를 쓰지 않으면 선이 사라진다.

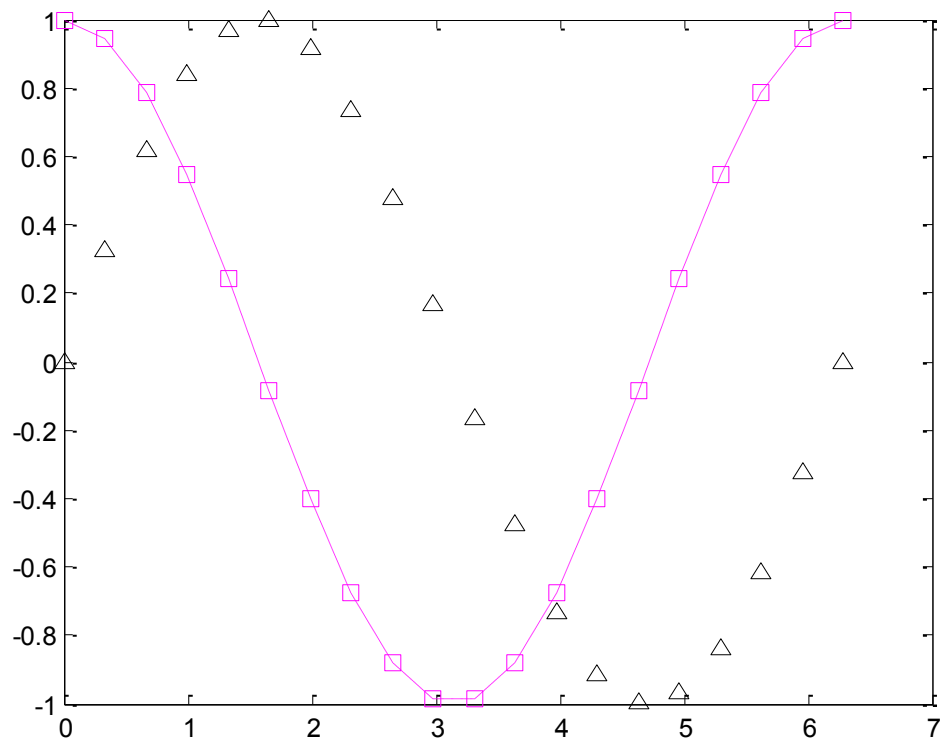
색상		마커		형태	
B	파랑	.	점	-	실선
G	초록	O	원	:	점선
R	빨강	X	X자	-.	쇄선
C	시안	+	십자	--	파선
M	마젠타	*	별	(없음)	선 지움
Y	노랑	S	사각형		
K	검정	D	마름모		
w	하양	V	삼각형(아래)		
		^	삼각형(위)		
		<	삼각형(좌)		

		>	삼각형(우)		
		P	오각형		
		H	육각형		

예제2.4. 선 스타일

```
>> x = linspace(0, 2*pi, 20);
>> y1 = sin(x);
>> y2 = cos(x);
>> plot(x, y1, '^k', x, y2, '--sm')
```

'^k'는 검정색으로 위를 보는 삼각형 마커를 그리라는 의미이며, 선을 지운다.
'--sm'은 마젠타 색으로 파선을 그리고, 사각형 마커를 추가하라는 의미이다.

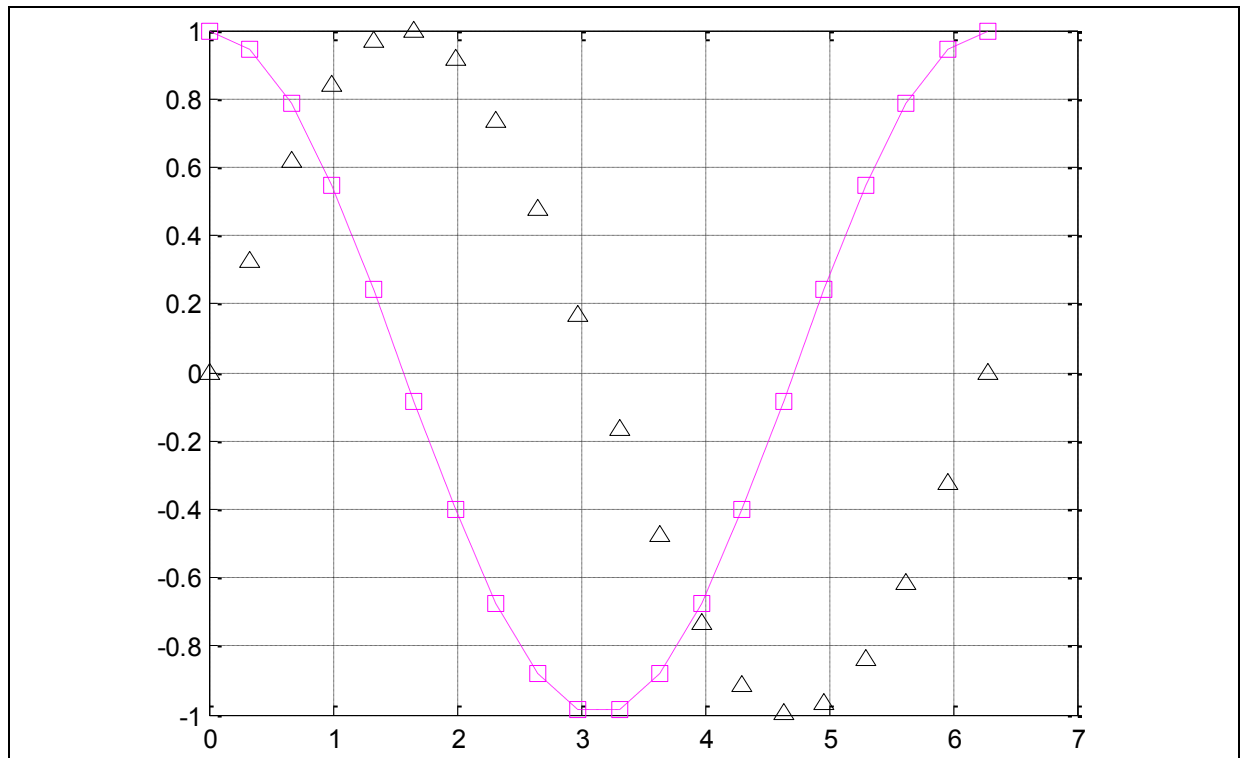


그래프의 스케일을 쉽게 인지하기 위해 격자가 필요할 수 있다. 명령어 `grid on`을 사용하면 그래프 상에 격자를 형성할 수 있다. 명령어 `hold on`과 마찬가지로, 격자를 해제하려면 `grid off`를 쓰면 된다.

예제2.5. 격자

```
>> x = linspace(0, 2*pi, 20);
>> y1 = sin(x);
>> y2 = cos(x);
>> plot(x, y1, '^k', x, y2, '--sm')
>> grid on
```

명령어 `grid on`을 사용하면 그래프 상에 격자를 그린다.



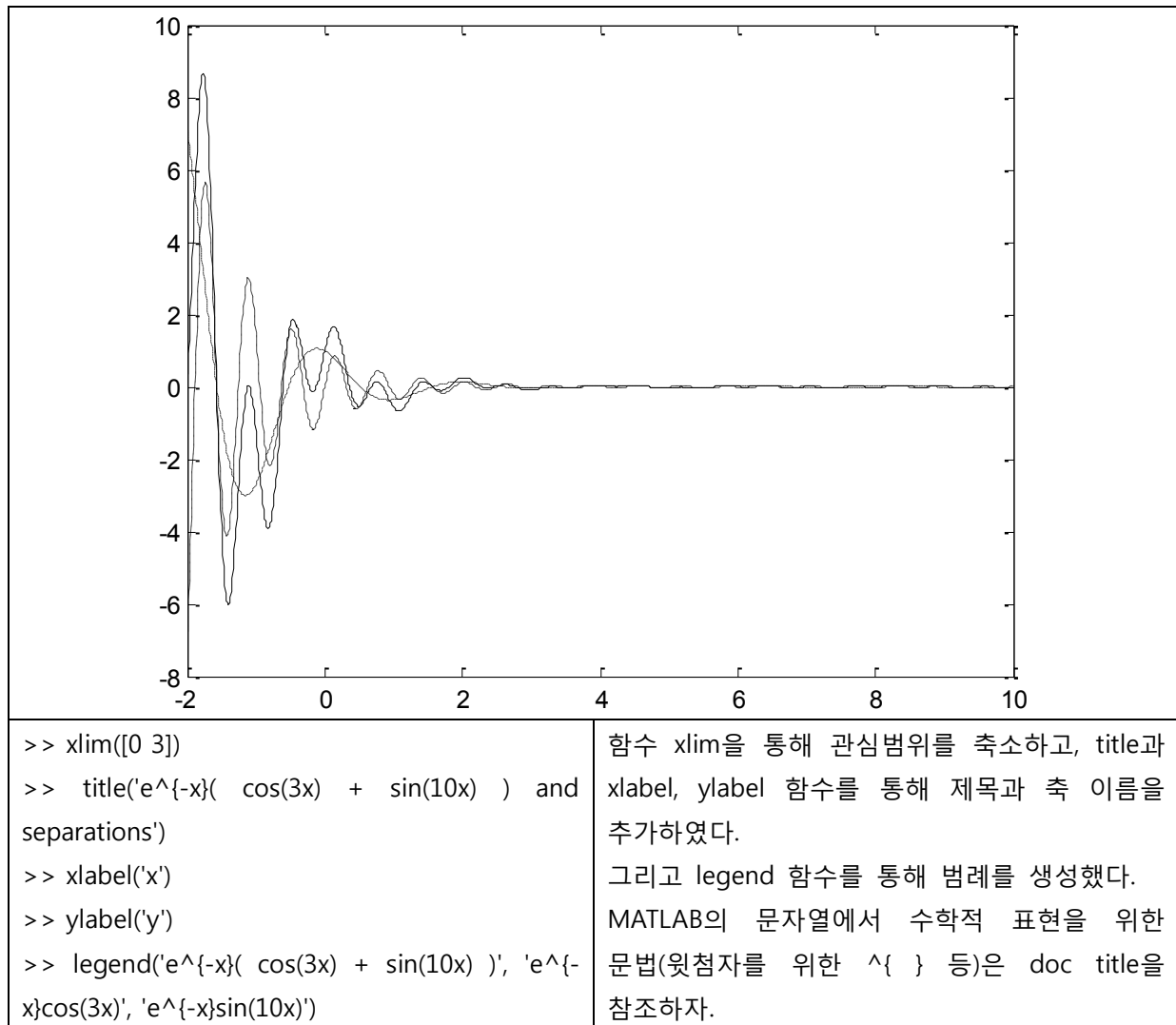
함수 plot의 자세한 사용법은 help나 doc을 참조하자. 자세한 설명이 제공된다.

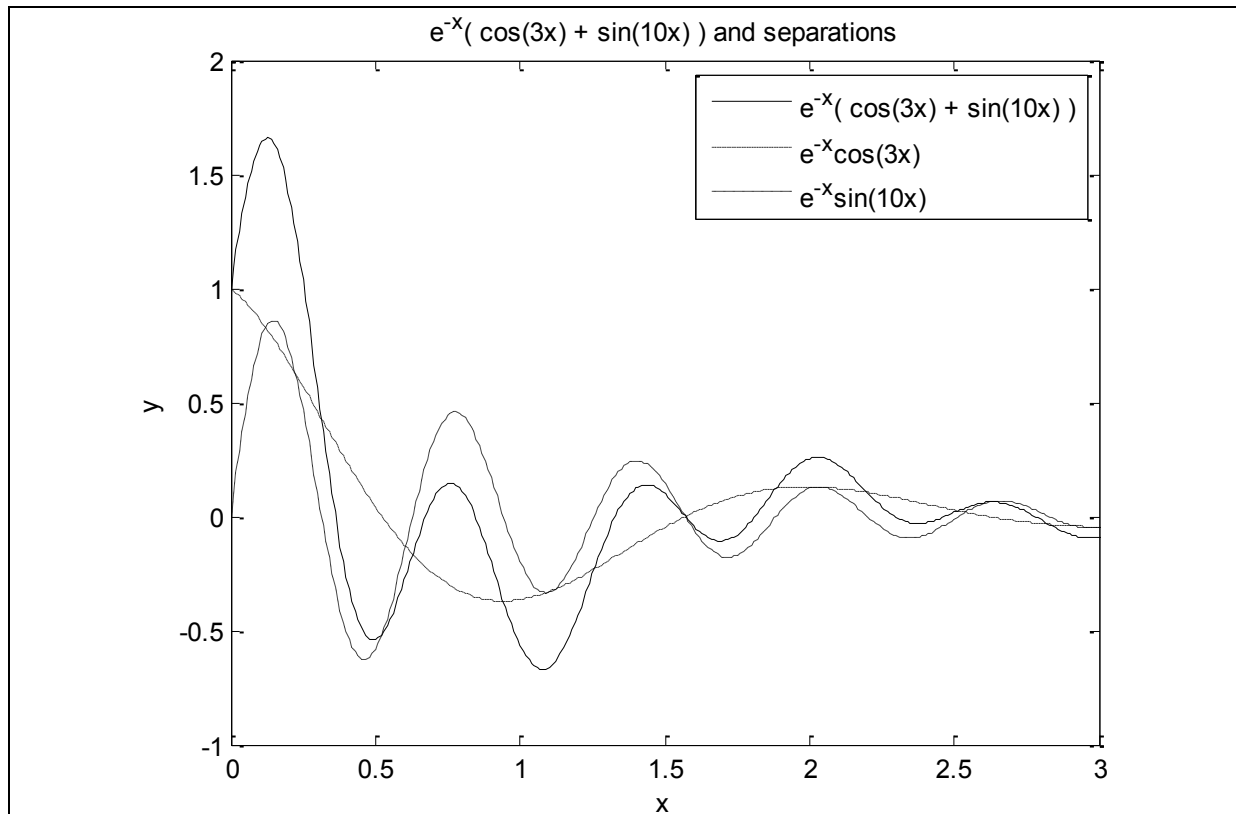
2.2. Plot 꾸미기

그래프에는 데이터가 무엇인지 나타내기 위한 제목, 축 라벨, 범례 등이 필요하며, 다음과 같은 함수를 통해 그래프에 추가할 수 있다.

MATLAB 명령	설명
title('string')	그래프 상단에 제목 'string'을 추가한다.
xlabel('string'), ylabel('string')	각각 x축, y축의 이름 'string'을 추가한다.
xlim([xmin xmax]), ylim([ymin ymax])	그래프의 표시 범위를 지정한다.
legend('string1', 'string2', ...)	그래프의 범례를 나타내는 박스를 추가한다.
axis 명령어	그래프의 축 표현 방식을 변경한다.

예제2.6. 그래프 꾸미기	
<pre>>> x = -2:0.01:10; >> y = exp(-x).*(cos(3*x) + sin(10*x)); >> y1 = exp(-x).*cos(3*x); >> y2 = exp(-x).*sin(10*x); >> plot(x, y, 'k', x, y1, 'k-', x, y2, '--k')</pre>	<p>각 그래프는 검정으로 나타내었으며, 실선과 파선, 색선으로 구별할 수 있게 하였다.</p>



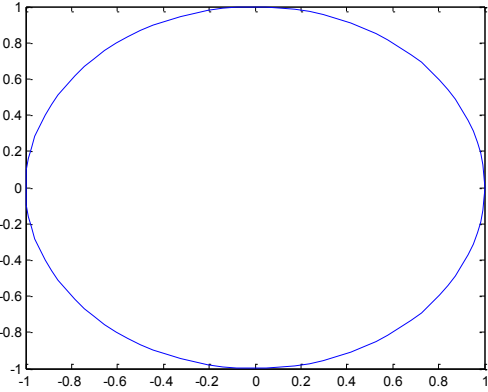
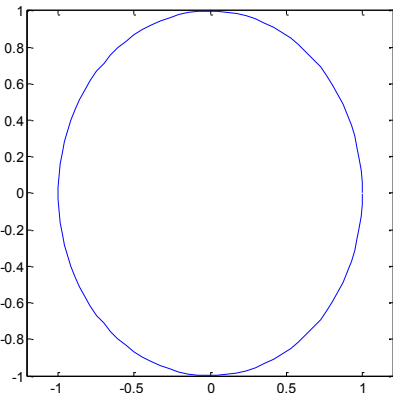
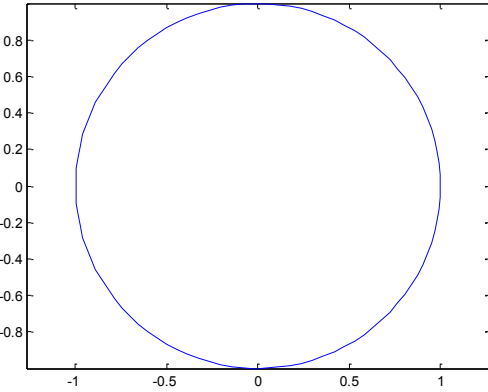
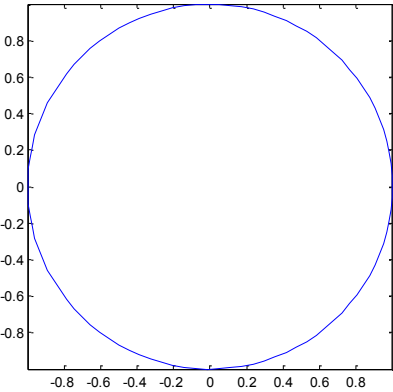


함수 plot을 이용하여 그래프를 그렸을 때, 각 축의 스케일은 자동으로 데이터의 범위와 그래프 창의 크기에 맞게 설정된다. 따라서, 그래프 상에 원을 표현해도 모니터 화면으로 보기에 타원처럼 일그러져 보일 수 있다. 또한, 그래프의 모양이 창 크기에 맞춘 직사각형 대신 정사각형으로 나타내어야 할 경우도 있다. 이와 같은 종류의 설정을 axis 명령으로 다룰 수 있는데, 자세한 내용은 doc axis에서 확인할 수 있으며 다음 예제2.7에서는 주로 쓰이는 square, equal, tight, image을 소개한다.

예제2.7. axis 명령

```
>> theta = 0:0.1:2*pi;
>> x = cos(theta);
>> y = sin(theta);
>> plot(x, y)
```

0에서 2π 까지의 각도를 나눈 후 cos를 x, sin을 y에 두면 원을 그릴 수 있다.

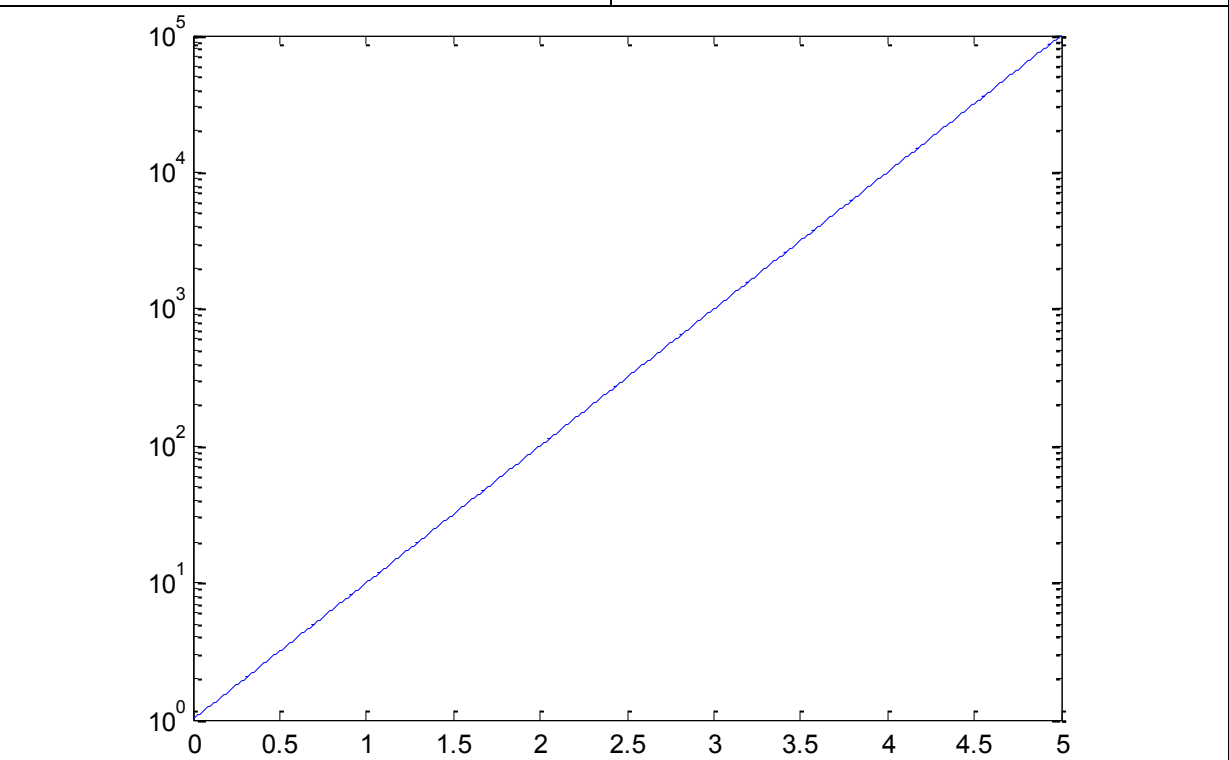
	<p>기본적인 상태로, 원이 약간 일그러져 보인다. 이 상태는 default 값인 axis auto로, MATLAB의 임의로 그래프의 범위와 축 스케일을 설정한 상태이다.</p>
	<pre>>> axis square >> xlim([-1.2 1.2])</pre> <p>위의 명령을 입력하면 왼쪽의 그래프와 같이 나타난다. 그래프의 모양은 정사각형이나, 축의 범위가 바뀌면 축 스케일이 바뀐다. Axis square를 입력한 직후에는 원처럼 보일 수 있으나, xlim을 통해 범위를 변경하면 다시 일그러져 보인다.</p>
	<pre>>> axis equal</pre> <p>각 축의 스케일이 동일하도록 만드는 명령이다. 왼쪽의 그래프를 보면 원의 모양이 그대로 나타난 것을 확인할 수 있으며, 축의 범위도 -1.2 ~ 1.2를 유지하는 것을 알 수 있다.</p>
	<pre>>> axis tight</pre> <p>각 축의 범위가 데이터의 범위와 일치하도록 설정하는 명령이다. 함수 xlim을 통해 설정했던 x축의 범위가 데이터에 맞게 재조정되어 그래프 상에 원만 나타나도록 바뀌었다. 앞에서 axis equal을 사용했기 때문에 축의 스케일은 동일한 상태이다. Axis tight와 equal을 동시에 적용하는 image라는 명령도 있다.</p>

보드 선도(bode plot)와 같이 x축과 y축이 로그 스케일로 표현해야 하는 그래프를 그리고자 할 때, semilogx와 semilogy, loglog 함수가 사용된다. 이 함수는 plot과 사용방식이 동일하며, semilogx는 x축이 로그 스케일, semilogy는 y축이 로그 스케일, loglog는 x축과 y축 모두 로그 스케일로 표현된다.

예제2.8. 로그 스케일 Plot

```
>> x = 0:0.01:5;
>> y = 10.^x;
>> semilogy(x, y)
```

10을 밑으로 하는 지수함수를 계산한 뒤, y축을 로그 스케일로 표현하였다. 다음과 같이, 지수함수가 일차함수처럼 보인다.



하나의 창에 여러 개의 그래프를 그려야 하는 경우, subplot이라는 함수를 통해 구역을 나누어 그릴 수 있다. Subplot(m, n, p)라고 쓸 경우, 하나의 창을 m개의 행과 n개의 열로 나눈 뒤, 그 중 p번째 위치에 그래프를 그리겠다는 뜻이 된다. 다음 예제를 살펴보자.

예제2.9. Subplot 1

```
H1 = [1.5 -0.5; -0.5 1.5];
H2 = [1.5 0.5; 0.5 1.5];

theta = linspace(0, 2*pi, 100);
X = [cos(theta); sin(theta)];
Y1 = H1*X;
```

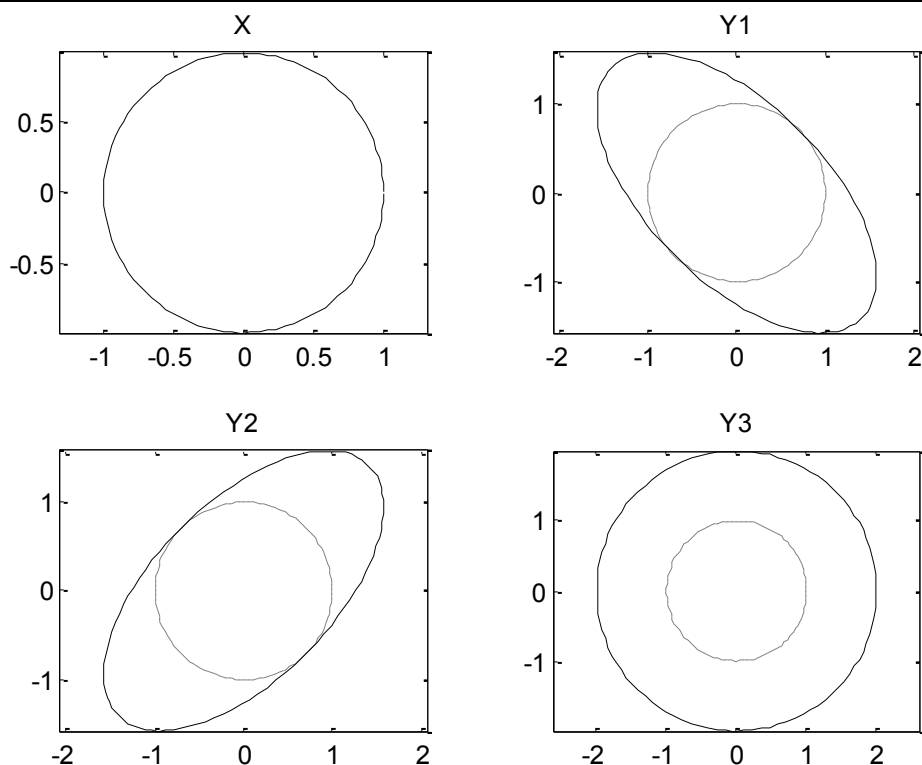
두 개의 선형변환 행렬 H1과 H2를 만들고, 원을 이루는 좌표인 X에 곱하여 선형변환을 한 후, 그래프로 그려 결과를 보고자 한다. Y1, Y2, Y3는 선형변환의 결과 좌표이다.

```
Y2 = H2*X;
Y3 = H1*H2*X;
```

```
subplot(2, 2, 1);
plot(X(1, :), X(2, :), 'k');
title('X'); axis equal
subplot(2, 2, 2);
plot(X(1, :), X(2, :), 'k', Y1(1, :), Y1(2, :), 'k');
title('Y1'); axis equal
subplot(2, 2, 3);
plot(X(1, :), X(2, :), 'k', Y2(1, :), Y2(2, :), 'k');
title('Y2'); axis equal
subplot(2, 2, 4);
plot(X(1, :), X(2, :), 'k', Y3(1, :), Y3(2, :), 'k');
title('Y3'); axis equal
```

한 창에 여러 개의 그래프를 그리기 위해 subplot을 사용하였다. 각 창마다 별도로 plot을 실행하고 title과 axis로 설정해주었으며, Y1, Y2, Y3에는 비교를 위해 X도 같이 나타내었다.

함수 subplot의 인자를 자세히 보면 세 번째 인자가 가리키는 그래프의 위치가 행을 우선한다는 것을 알 수 있다. 즉, 행렬을 벡터 인덱스로 접근할 때와 반대로, 1은 (1, 1), 2는 (1, 2), 3은 (2, 1), 4는 (2, 2)를 나타내는 것이다.



그래프의 위치를 좀 더 자유롭게 배치해야 할 경우도 있을 것이다. 이 때, 그래프의 위치를 나타내는 p 인자에 벡터를 넣어주면 해당하는 위치가 합쳐서 할당된다.

예제2.10. Subplot 2

```
H1 = [1.5 -0.5; -0.5 1.5];
H2 = [1.5 0.5; 0.5 1.5];
```

앞의 예제2.9와 거의 동일하지만, subplot으로 각 그래프의 위치를 할당할 때 벡터를

```

theta = linspace(0, 2*pi, 100);
X = [cos(theta); sin(theta)];
Y1 = H1*X;
Y2 = H2*X;
Y3 = H1*H2*X;

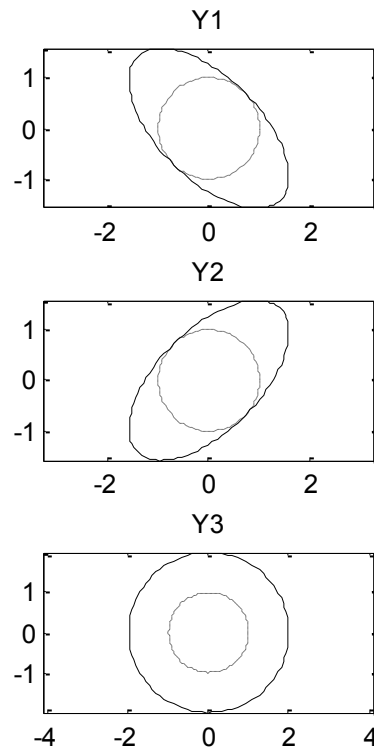
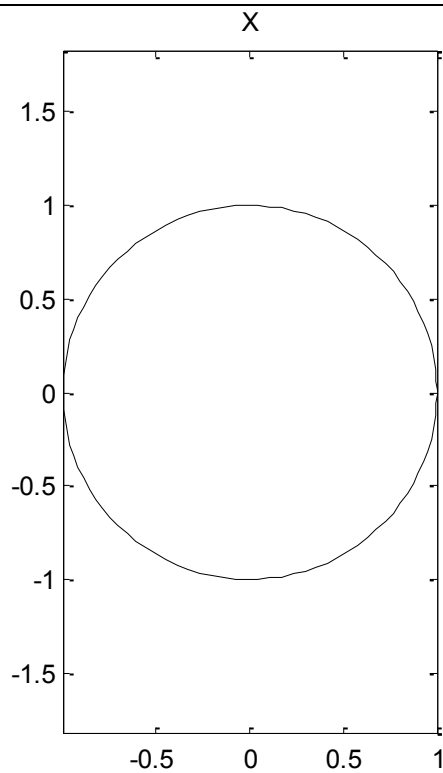
subplot(3, 2, [1 3 5]);
plot(X(1, :), X(2, :), 'k');
title('X'); axis equal
subplot(3, 2, 2);
plot(X(1, :), X(2, :), 'k', Y1(1, :), Y1(2, :), 'k');
title('Y1'); axis equal
subplot(3, 2, 4);
plot(X(1, :), X(2, :), 'k', Y2(1, :), Y2(2, :), 'k');
title('Y2'); axis equal
subplot(3, 2, 6);
plot(X(1, :), X(2, :), 'k', Y3(1, :), Y3(2, :), 'k');
title('Y3'); axis equal

```

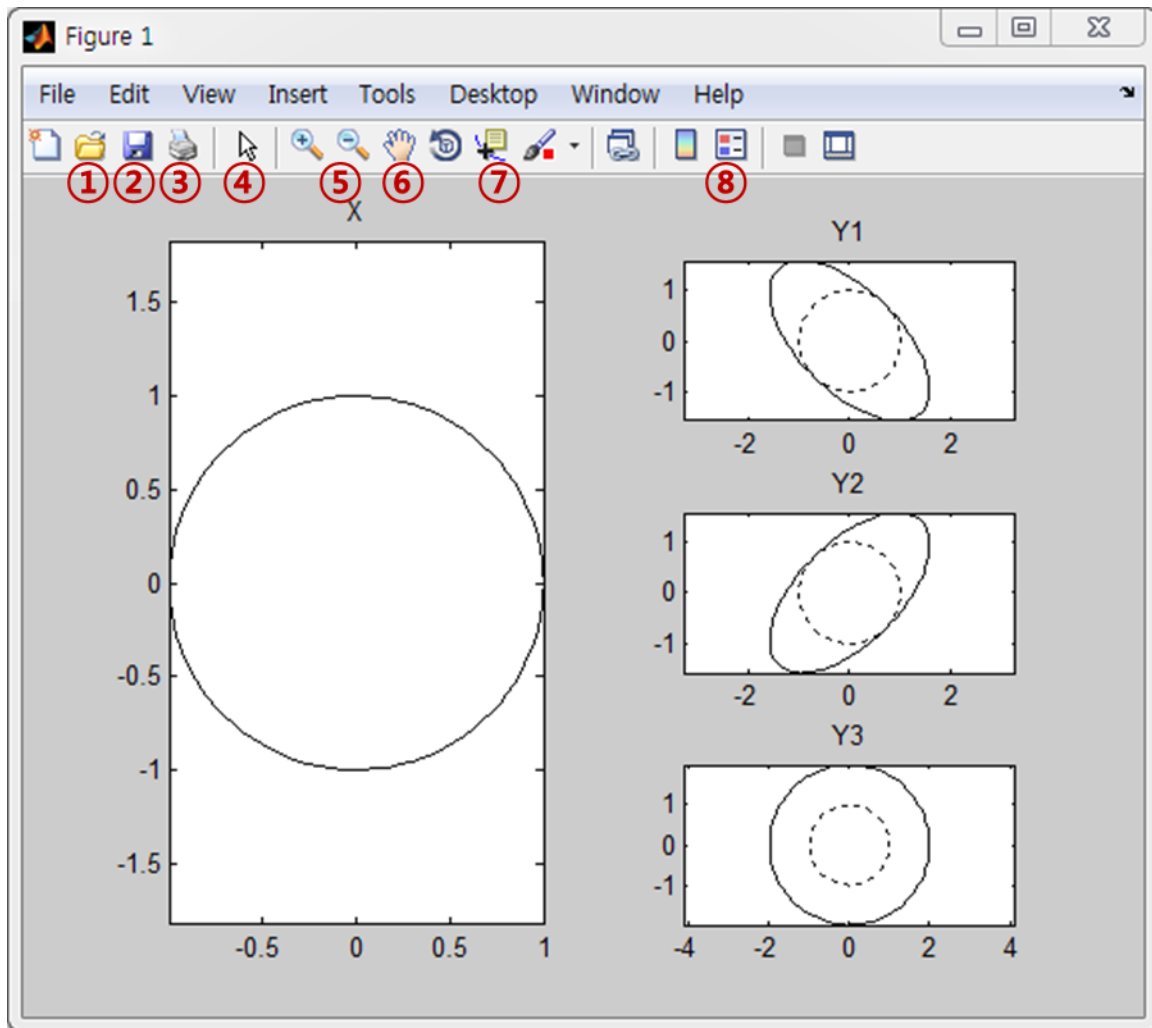
사용하였다.

기준이 되는 좌표 X가 창의 왼쪽 전체를 가지도록 하기 위해 3X2 모양으로 구역을 나눈 후, [1 3 5]에 해당하는 위치에 X의 그래프를 그렸다.

나머지 Y1, Y2, Y3는 2, 4, 6번 위치에 그렸다.

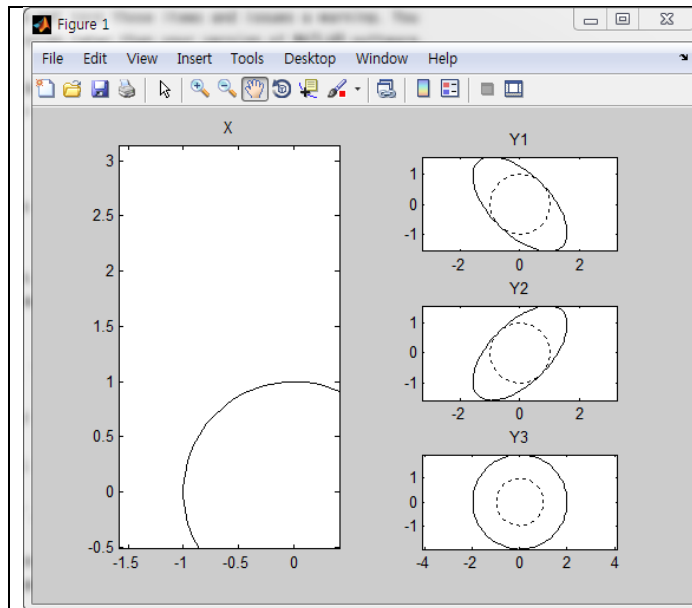


2.3 Figure GUI를 이용한 꾸미기

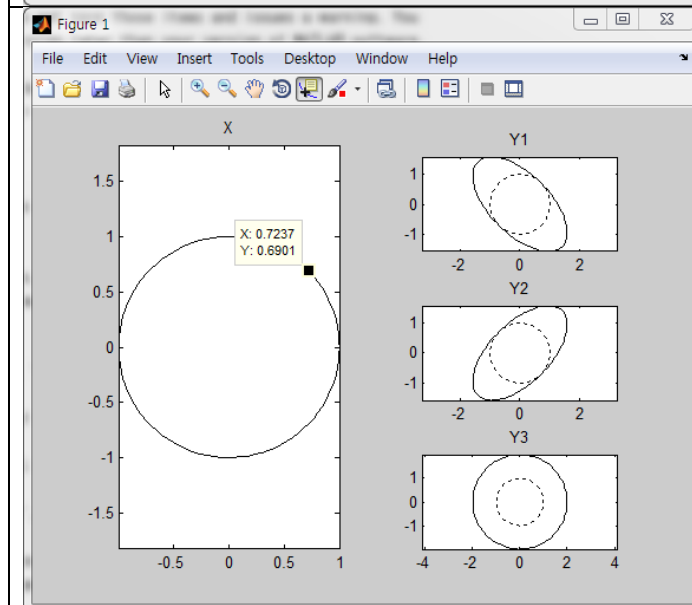


위 그림은 예제2.10에서 만든 창의 화면을 그대로 복사한 것이다. 메뉴 아래의 아이콘 툴바에는 마우스 클릭을 통해 사용할 수 있는 간단한 기능이 배치되어 있다. 다음은 자주 쓰이는 기능의 소개이다.

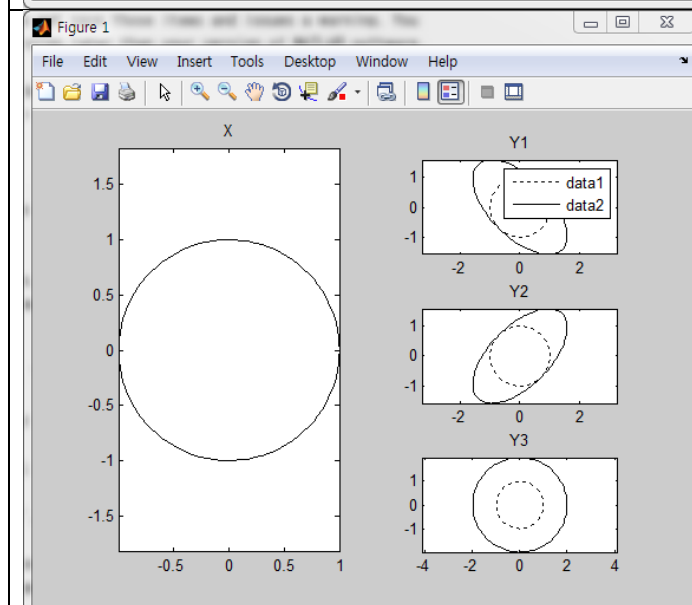
- ① Open File: fig 확장자의 MATLAB 그래프 파일을 불러온다.
- ② Save Figure: fig 확장자의 MATLAB 그래프 파일로 저장한다.
- ③ Print Figure: 그래프를 인쇄한다.
- ④ Edit Plot: 마우스 툴로써, 클릭하면 드래그를 통해 그래프나 텍스트상자의 위치를 옮길 수 있다.
- ⑤ Zoom In/Out: 확대/축소할 수 있다. 확대할 때는 드래그를 통해 영역을 지정할 수 있으며, 축소할 때는 클릭할 때마다 한 단계씩 축소된다.
- ⑥ Pan: 마우스 툴로, 그래프 안에서 스케일의 변경 없이 표시 영역을 움직일 수 있다. 즉, 드래그를 통해 그래프를 밀거나 당길 수 있다.
- ⑦ Data Cursor: 그래프 상의 좌표 값을 읽고 싶은 점을 클릭하면 표시해주는 마우스 툴이다.
- ⑧ Insert Legend: 범례를 추가한다. 범례 텍스트상자 안의 문자열을 더블클릭하면 내용도 수정할 수 있다.



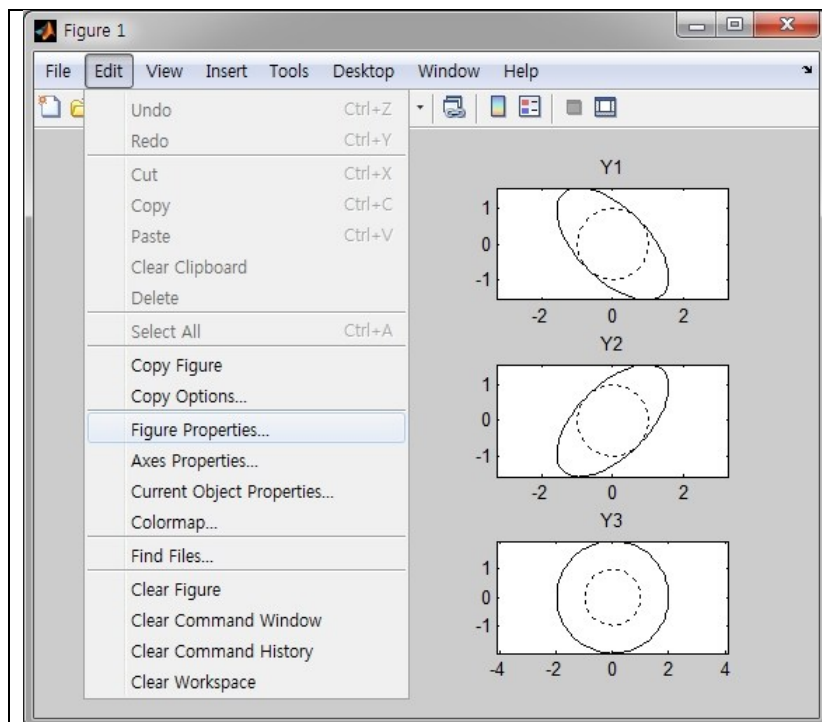
Pan 기능을 이용하여 그래프를 드래그로 당긴 결과. 한 가운데에 그려졌던 원이 오른쪽 아래로 내려갔다. 스케일이 바뀌지는 않는다.



Data Cursor로 점을 클릭하여 좌표를 읽는다. 이 상태로 이리저리 점을 드래그하여 다른 위치의 좌표를 읽을 수도 있다.



Insert Legend를 누르면 선택된 그래프의 범례가 그려진다. 왼쪽 그림의 경우, Y1 그래프를 클릭한 후 기능을 사용하여 Y1 위에 범례가 나타났다. 기본값으로 data1과 data2가 적혀있는데, 문자열을 더블클릭하면 수정할 수 있다.



메뉴의 Edit > Figure Properties...를 누르면 그래프 아래에 Property Editor가 생기며, 이 에디터로 그래프의 모든 속성을 변경할 수 있다. MATLAB command window의 명령어로 바꾸기 힘들었던 부분은 이 에디터로 해결하자.

Tip. 메뉴의 Edit > Copy Figure로 그래프를 복사하면 자동적으로 그래프 부분만 복사해주며, 확대해도 깨지지 않는다.

