APPENDIX

| # 문자열 | |
|---|---|
| >> s = 'Hello'<br><br>s =<br>Hello | MATLAB에서 문자열은 ' '로 둘러 나타낸다. 변수로 대입하는 것도 당연히 가능하다. |
| >> size(s)<br><br>ans =<br>1 5 | 문자열을 저장한 변수의 크기를 살펴보면 행벡터인 것을 알 수 있다. 'Hello'가 5개의 알파벳으로 이루어져 있어, 열의 크기가 5이다. |
| >> double(s)<br><br>ans =<br>72 101 108 108 111 | 함수 double을 사용하면 변수의 type을 실수로 바꿀 수 있으며, 문자열 변수를 입력하면 ASCII 코드로 바꾸어준다. |
| >> char(ans)<br><br>ans =<br>Hello | 함수 char는 double과 반대로 변수의 type을 문자열로 바꾸어주며, 자연수의 ASCII 코드에 해당하는 문자로 바꾸어준다. |
| >> w = 'World'<br><br>w =<br>World | |
| >> [s w]<br><br>ans =<br>HelloWorld | 둘 이상의 문자열을 합치고 싶을 경우, 벡터를 합치는 것과 동일하게 [ ]으로 감싸면 된다. |
| >> [s ' ' w]<br><br>ans =<br>Hello World | 문자열을 합칠 때 사이에 빈 공간이 자동으로 들어가지 않으므로, 띄어쓰기가 필요할 경우 ' '를 넣어주어야 한다. |
| >> size(ans) | 이 벡터의 길이를 확인해보면 11로, ' '도 |

| | |
|---|---|
| ans =<br>1 11 | 크기에 포함이 된다는 것을 알 수 있다. |
| >> n = 2;<br>>> [s n w]<br><br>ans =<br>HelloWorld | n=2는 문자열이 아닌 숫자이다. 이 변수를 그대로 문자열과 섞어서 합칠 경우, ASCII 코드로 변환이 된다. Hello와 World 사이의 ㄱ자 문자는 ASCII 코드의 2번 문자이다. |
| >> [s num2str(n) w]<br><br>ans =<br>Hello2World | 변수의 숫자를 문자열과 섞어서 나타내고 싶을 경우, 함수 num2str을 사용한다. 이 함수는 숫자를 문자열로 변환해준다. 반대의 경우인 str2num이라는 함수도 있다. |
| >> z = 0.5;<br>>> [s num2str(z) w]<br><br>ans =<br>Hello0.5World | 함수 num2str은 실수나 복소수도 문자열로 바꾸어준다. 상수 pi와 같은 무리수의 경우, 유효숫자 5개까지 나타낸다. |
| >> x = 3; y = 5;<br>>> eval('x+y')<br><br>ans =<br>8<br><br>>> eval('z = x+y')<br><br>z =<br>8 | 함수 eval은 입력받은 문자열을 command window에 입력한 것과 동일하게 작동시켜준다. 평상시에는 쓸 일이 별로 없으나, 후에 GUI와 같은 프로그램을 작성할 때 반드시 필요하며, 비슷한 모양의 코드를 반복해서 입력하는 경우에도 반복문 문법과 함께 사용하면 편리하다. |
| >> z = eval('x+y')<br><br>z =<br>8 | 출력 변수를 지정할 경우, 입력 받은 문자열의 결과를 출력 변수에 저장한다. |

| |
|---|
| # Symbolic Math Toolbox |

```
>> syms x real
>> y = x + x

y =
2*x

>> y = sin(x);
>> h = subs(y, x, pi/2)

h =
     1

>> syms a b t real
>> y = cos(a*t + b);
>> diff(y)

ans =
-a*sin(b + a*t)

>> diff(y, b)

ans =
-sin(b + a*t)

>> y = sin(2*x);
>> int(y)

ans =
sin(x)^2 - 1/2

>> int(y, 0, pi/2)

ans =
1

>> f = 'cos(2*x) + sin(x) = 1';
>> solve(f)

ans =
```

Symbolic math 기능은 컴퓨터로 사람과 같은 대수해석적 연산을 하기 위해 개발된 toolbox이다. 즉, x + x를 연산할 때 x의 값과 x의 값을 더하는 것이 아니라 2x라고 결과를 해석하는 방식이다. Symbolic math toolbox의 대부분의 함수는 변수가 symbolic math용으로 먼저 선언이 되어 있어야 한다.

함수 subs는 대입연산을 나타내는 것으로, 첫 번째 인자는 수식, 두 번째 인자는 원래 문자, 세 번째 인자는 대체할 식, 문자, 또는 숫자를 넣으면 두 번째 인자의 문자를 세 번째 인자의 입력으로 대체하여 연산한다.

Symbolic math의 장점은 여러 가지가 있지만, 대표적으로 해석적 연산을 수행할 수 있다는 점이 매우 중요하다. 즉, 미분을 할 때 수치적인 방식이 아니라 식 자체를 해석하여 도함수를 도출할 수 있는 것이다. 적분할 때도, 수치적 방식 대신 부정적분을 이용하여 정적분을 구한다.

미분의 경우, diff 함수를 사용하면 되는데, 그냥 식만 입력할 경우 자동으로 x에서 가장 가까운 알파벳을 선정하여 그 문자에 대하여 미분한다. 또는, 두 번째 인자에 미분할 문자를 정할 수 있다.

적분도 미분과 비슷하다. 식만 입력할 경우 부정적분을 수행하며, 두 번째 인자를 넣을 경우 그 문자에 대하여 부정적분을 행한다. 세 번째 인자까지 입력하면 정적분으로 인식하여 두 번째 인자부터 세 번째 인자까지의 범위를 적분하며, 네 번째 인자까지 넣으면 두 번째 인자에 대하여 세 번째와 네 번째 인자의 범위까지 정적분을 행한다.

Symbolic math의 유용한 함수로, solve와 dsolve가 있다. 함수 solve는 문자열로 입력

| | |
|---|---|
| 0<br>pi/6<br>(5*pi)/6<br><br>>> dsolve('D2y + 3*Dy + 2*y = 0', 'y(0)=1', 'Dy(0)=0')<br><br>ans =<br>2/exp(t) - 1/exp(2*t) | 받은 방정식의 해를 구해주며, dsolve는 미분방정식의 해를 구해준다. 함수 solve는 필요할 경우, 식 외에 해를 구할 문자를 입력해주어야 하며, dsolve는 초기 조건을 입력해주어야 한다. |
| >> pretty(ans)<br><br>2 1<br>------ - --------<br>exp(t) exp(2 t) | 문자로 출력된 식에 pretty 함수를 사용하면 일렬로 나열된 식을 2차원으로 표현하여 한결 이해하기 쉽게 도와준다. |
| >> simplify(ans)<br><br>ans =<br>2/exp(t) - 1/exp(2*t) | 매우 복잡한 식이 있을 경우, simplify 함수를 통해 최대한 간략하게 정리할 수 있다. 함수 simple은 다양한 방식으로 식을 정리해 보여줌으로써 정리할 방법을 선택할 수 있게 한다. |