โปรแกรมประยุกต์ Matlab

น.ต.สิทธิรักษ์ พรมณีย์
กองวิชาวิศวกรรมศาสตร์

โปรแกรม Matlab เป็นโปรแกรมประยุกต์โปรแกรมหนึ่งที่นิยมนำมาใช้ช่วยในการประกอบการเรียนที่เกี่ยวข้องกับการคำนวณต่าง ๆ ตั้งแต่ขั้นพื้นฐานจนกระทั่งขั้นประยุกต์ใช้งานเฉพาะด้านในการเรียนสาขาทางวิศวกรรมไฟฟ้าต่าง ๆ ที่สำคัญ เช่น ระบบควบคุม ระบบสื่อสาร เป็นต้น หรืออาจจะนำไปใช้ในการออกแบบและทดลองระบบทางไฟฟ้าต่าง ๆ ขณะนี้มีการพัฒนาโปรแกรมให้สามารถใช้งานได้สะดวกมากขึ้น โดยเฉพาะอย่างยิ่งการทำงานของโปรแกรมสามารถใช้กับระบบปฏิบัติการของวินโดว์ได้เป็นอย่างดี (Window 95 / 98 / 2000 /Me / Xp) และปัจจุบันพัฒนาถึง version 6.0 (Releases 12) ผู้เขียนเคยมีประสบการณ์การใช้โปรแกรมนี้ในช่วงที่ศึกษาอยู่ต่างประเทศจึงเห็นประโยชน์ที่จะแนะนำนักเรียนนายเรือและผู้ที่สนใจเพื่อนำไปใช้กับคำสั่งและการเขียนโปรแกรมการทำงาน เนื่องจากวิชาที่เปิดสอนสาขาวิศวกรรมไฟฟ้าสามารถแสดงผลการทำงานของสมการต่าง ๆ ด้วยวิธีสร้างแบบจำลองจากโปรแกรมได้จึงเป็นการช่วยให้ผู้เรียนเข้าใจง่ายขึ้น เนื้อหาจะเป็นการแนะนำการใช้งานของคำสั่งที่จำเป็นในการคำนวณขั้นพื้นฐานที่สำคัญ จนกระทั่งถึงขั้นการเขียนโปรแกรมสำหรับการประมวลผล โดยจะเน้นวิชาระบบควบคุมเสียเป็นส่วนใหญ่

การเข้า และ การออกจากโปรแกรม

เมื่อติดตั้งโปรแกรมเรียบร้อยแล้ว จะเรียกโปรแกรมมาใช้งานสามารถกระทำได้โดยการคลิกเมาส์ที่ไอคอนของ Matlab หรือที่ Matlab bar ในโปรแกรมไฟล์ก็จะเป็นการเข้าสู่โปรแกรม เมื่อเข้าไปในโปรแกรมแล้วสักครู่จะมีสัญลักษณ์ที่เรียกว่า **prompt (>> หรือ EDU >>)** ปรากฏขึ้น แสดงว่าโปรแกรมพร้อมจะรับคำสั่งเพื่อทำงานทันทีที่มีการป้อนคำสั่ง สำหรับการออกจากโปรแกรมกระทำได้โดยการพิมพ์คำสั่ง **quit** หรือ **exit** หรืออาจจะใช้วิธีการปิดโดยตรงที่เครื่องหมายกากบาทที่ขอบด้านบนขวามือ

การใช้ help และ demo

การใช้คำสั่ง **demo** โปรแกรมจะแสดงตัวอย่างการใช้ Matlab tools เพื่อเป็นการแนะนำการใช้เครื่องมือรวมทั้งแบบจำลองต่าง ๆ ที่กำหนดให้ไว้ในโปรแกรมพร้อมการแสดงกราฟของวงจรนั้น ๆ ด้วยการเปิดหน้าจอใหม่

>> demo

การใช้คำสั่ง **help** เพื่อช่วยในการค้นหาความหมายของคำสั่งบางคำสั่งที่ไม่เข้าใจวิธีการใช้ หรือใช้คำสั่ง **helpdesk** จะเป็นการเข้าสู่หน้าต่างของวินโดว์เพื่อค้นหาคำที่ต้องการต่อไป

>> help (จะปรากฏ tools box ของแต่ละระบบ)

>> help คำสั่งที่ต้องการค้นหา (จะปรากฏคำอธิบายคำสั่งนั้น)

>> helpdesk (เปิดหน้าวินโดว์ใหม่)

การป้อนคำสั่ง

การป้อนคำสั่งต่าง ๆ ของโปรแกรม ด้วยวิธีการพิมพ์คำสั่งหลังเครื่องหมาย Prompt ; **>>** เมื่อจบคำสั่งกด Enter จะเป็นการขึ้นบรรทัดใหม่ เพื่อแสดงผลหรือรอรับคำสั่งใหม่ต่อไป

การป้อนคำสั่งด้วยวิธีการเขียนโปรแกรมจะใช้คำสั่ง **input** (ด้วยตัวพิมพ์เล็ก) รูปแบบของคำสั่ง **input (' ใส่ความหมายของคำที่ต้องการให้แสดงผล : (เว้นช่องว่าง) ')** ความหมาย คือ ค่าที่ป้อนหรือค่าที่กำหนด จะถูกเก็บไว้ที่พารามิเตอร์ที่กำหนด ดังตัวอย่างต่อไปนี้

```
>> M = input ('Enter the values for M in bracket:        ')
Enter the values for M in bracket:  2  (พิมพ์หมายเลข 2)
M    =     2
```

การแสดงผล

การแสดงผลของโปรแกรม ปกติเมื่อกด Enter จะมีการประมวลผลการทำงานทันที แต่ถ้าต้องการให้แสดงผลของการเขียนโปรแกรมหรือกราฟจะต้องมีการใช้คำสั่ง ดังนี้

disp เป็นคำสั่งที่ใช้ในการแสดงเกี่ยวกับกลุ่มคำหรืออักษร รวมทั้งผลของค่าต่าง ๆ ที่เก็บไว้ในค่าของตัวแปร รูปแบบของคำสั่ง **disp (ตัวแปรหรือพารามิเตอร์ที่กำหนดค่าแล้ว) หรือ disp ('อักษรหรือ ข้อความที่ต้องการให้แสดง')** ดังตัวอย่างต่อไปนี้

```
>> temp = 0:2:10;
>> disp( ' The number of temp is : ' ); disp( temp )
The number of temp is :
   0    2    4    6    8    10
```

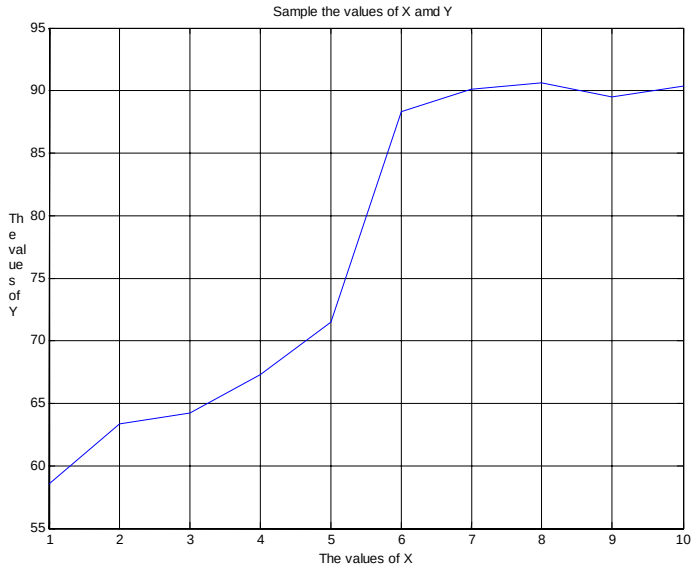
fprintf เป็นคำสั่งที่ใช้ในการแสดงผลอีกแบบหนึ่งที่คล้ายกับคำสั่ง disp แต่สามารถควบคุมการแสดงผลได้ รูปแบบของคำสั่ง **fprintf (' ตัวอักษรหรือข้อความที่ต้องการให้แสดง % รูปแบบการ**

แสดงจำนวน (format) ' , ตัวแปรหรือพารามิเตอร์ที่กำหนดค่าแล้ว) ดังตัวอย่างต่อไปนี้

```
>> temp = 5
>> fprint (' The number of temp is  %s   \n' , temp)
The number of temp is 5
```

plot เป็นคำสั่งที่ใช้แสดงผลในรูปแบบของกราฟที่สัมพันธ์กับแกน (X, Y) คำสั่งนี้จะมีคำสั่งที่ทำงานร่วมกันคือ **title** เป็นคำสั่งที่ใช้กำหนดความหมายของกราฟ และ **xlabel** กับ **ylabel** ใช้ในการกำหนดความ หมายของแกน **x** และ **y** รูปแบบของคำสั่ง **plot (พารามิเตอร์หรือ ตัวแปรของแกน x, พารามิเตอร์หรือตัวแปรของแกน y)**, **title (' ตัวอักษรหรือข้อความที่ต้องการให้แสดง ')**, **xlabel** กับ **ylabel (' ตัวอักษรหรือข้อความที่ต้องการให้แสดง '),** **grid** (เป็นคำสั่งการแสดงตารางของกราฟ) ดังตัวอย่างต่อไปนี้

```
>> x = 1:1:10 ;
>> y = [ 58.5  63.4  64.2  67.3  71.5  88.3  90.1  90.6  89.5  90.4 ] ;
>> plot(x,y), title('Sample the values of X and Y')
>> xlabel('The values of X'),ylabel('The values of Y'),grid
```



คำสั่งเกี่ยวกับจำนวน สกาลา เวกเตอร์ และ เมตริกส์ (Scalars, Vectors and Matrices)

Matlab จะแสดงจำนวนแบ่งออกเป็น ๓ แบบ คือ จำนวนสเกลลา (Scalar) จำนวนเวกเตอร์ (Vector) และ เมตริกส์ (Matrices) รวมทั้งตัวอักษรที่ใช้กำหนดความหมายหรือขยายความ การป้อนคำสั่งก็พิมพ์หลังเครื่องหมาย prompt การเขียนคำสั่งเช่นเดียวกับการเขียนสมการทางคณิตศาสตร์ทั่วไป ดังตัวอย่างต่อไปนี้

```
>> A   =    1.2          (สำหรับจำนวน สเกลลา)
A      =    1.200
>> B   =    [1.5  3.1]   (สำหรับจำนวนเวกเตอร์ระหว่างจำนวนต้องเว้นวรรค)
B      =    1.5   3.1
>> C   =    [ 1,2,3 ; 4,5,6 ; 7,8,9]
C      =    1    2    3
            4    5    6
            7    8    9
```

การใช้ Semicolon (;) และ Colon (:)

Semicolon (;) ใช้ในการขึ้นบรรทัดใหม่เมื่อวางไว้ที่ท้ายบรรทัดนั้น ๆ แสดงว่าการทำงานของคำสั่งยังต่อเนื่องกับบรรทัดต่อไป โดยที่โปรแกรมยังไม่แสดงผล ดังตัวอย่างต่อไปนี้

```
>> D   =    [ 1.5  3.1 ] ;
>> E   =    [ 3.0  4.5; D ]
E      =    3.0000    4.5000
            1.5000    3.1000
```

เมื่อนำไปใช้กับ เมตริกส์ จะหมายถึงการเริ่มบรรทัดใหม่ ดังตัวอย่างต่อไปนี้

```
>> D   =    [ 3.0  4.5; 1.5  3.1]
D      =    3.0000    4.5000
            1.5000    3.1000
```

colon (:) ใช้ระหว่างจำนวนตัวเลขสองจำนวนจะแสดงค่าการเพิ่มของตัวเลขระหว่างสองจำนวนนั้นทีละหนึ่ง แต่ถ้าแบ่งระหว่างสามจำนวนจะเป็นการเพิ่มค่าระหว่าง จำนวนแรกกับจำนวนสุดท้าย โดยมีอัตราการเพิ่มเท่ากับจำนวนที่อยู่ระหว่างสองจำนวน ดังตัวอย่างต่อไปนี้

```
>> E   =    1 : 3          (เพิ่มทีละหนึ่ง)
E      =    1    2    3
>> F   =    1 : 2 : 9      (เพิ่มทีละสอง)
F      =    1    3    5    7    9
```

การบวก ลบ คูณ หาร จำนวนสเกลลา เวกเตอร์ และ เมตริกส์

เงื่อนไขในการบวก ลบ คูณ หาร จำนวนสเกลลา เวกเตอร์ และ เมตริกส์ มีรายละเอียดดังนี้

ชนิด	รูปแบบ	Matlab
บวก	a + b	**a + b**
ลบ	a – b	**a – b**
คูณ	a × b	**a * b**
หาร	a / b	**a / b**
เอ็กซ์โพนเนลเชียล	a^b	**a ^ b**

ดังตัวอย่างต่อไปนี้

```
>> a = 5 ;  b = 5 ;
>> c = a + b            (การ บวก ลบ คูณ หาร เปลี่ยนเครื่องหมาย)
c       =       10
>> A    =       [ 2  5  6 ] ; B = [ 2 ; 3 ; 5 ] ;
>> C = A * B
C       =       49
```

การใช้ฟังก์ชันทางคณิตศาสตร์

โปรแกรม Matlab สามารถที่ใช้ฟังก์ชันทางคณิตศาสตร์ทุกอย่าง สำหรับฟังก์ชันที่สำคัญมี ดังนี้

Mathematical Function

คำสั่งที่เกี่ยวกับฟังก์ชันคณิตศาสตร์ทั่วไปมีดังนี้

abs(x) การหาค่า absolute ของค่า x

sqrt(x) การหาค่า square root ของค่า x

rem(x,y) การแสดงเศษของผลหารของ x กับ y

exp(x) การหาค่า exponential ของ x

log(x) การหาค่า natural logarithms ของ x ฐาน e

log10(x) การหาค่าของ $\log_{10} x$

Trigonometric และ Hyperbolic Functions

การใช้คำสั่งนี้มีสิ่งที่ต้องคำนึงถึง คือ ค่าที่คำนวณได้จะมีค่าเป็นเชิงมุม (Radian angle) ดังนั้นการที่จะต้องการให้แสดงผลออกมาเป็น มุมองศา (Degree angle) ทำได้โดยการแทนค่าดังนี้

มุมองศา = มุมเรเดียน * (180 / pi)

มุมเรเดียน = มุมองศา * (pi / 180)

คำสั่งที่เกี่ยวกับฟังก์ชัน มีดังนี้

sin(x) การหาค่าของ sine ของ x

cos(x) การหาค่าของ cosine ของ x

tan(x) การหาค่าของ tangent ของ x

asin(x) การหาค่าของ arcsine ของ x

acos(x) การหาค่าของ arccosine ของ x

atan(x) การหาค่าของ arctangent ของ x

atan 2(y, x) การหาค่าของส่วนกลับของ y / x **sinh(x)** การหาค่าของ hyperbolic sine ของ x

cosh(x) การหาค่าของ hyperbolic cosine ของ x **tanh(x)** การหาค่าของ hyperbolic tangent ของ x

asinh(x) การหาค่าของ inverse hyperbolic sine ของ x

acosh(x) การหาค่าของ inverse hyperbolic cosine ของ x

atanh(x) การหาค่าของ inverse hyperbolic tangent ของ x

ส่วนฟังก์ชันอื่นที่นอกเหนือจากคำสั่งนี้สามารถหาได้จากความสัมพันธ์ ดังนี้

$$\sec(x)=\frac{1}{\cos(x)} \quad ; \quad \csc(x)=\frac{1}{\sin(x)} \quad ; \quad \cot(x)=\frac{1}{\tan(x)}$$

$$\operatorname{arcsec}(x)=\arccos\left(\frac{1}{x}\right) \; ; \; \operatorname{arccsc}(x)=\arcsin\left(\frac{1}{x}\right) \; ; \; \operatorname{arccot}(x)=\arccos\left(\frac{x}{\sqrt{1+x^2}}\right)$$

$$\coth(x)=\frac{\cosh(x)}{\sinh(x)} \quad ; \quad \operatorname{sech}(x)=\frac{1}{\cosh(x)} \quad ; \quad \operatorname{csch}(x)=\frac{1}{\sinh(x)}$$

$$\operatorname{acoth}(x)=\ln\sqrt{\frac{x+1}{x-1}} \; ; \; \operatorname{asech}(x)=\ln\left(\frac{1+\sqrt{1-x^2}}{x}\right) \; ; \; \operatorname{acsch}(x)=\ln\left(\frac{1}{x}+\frac{\sqrt{1+x^2}}{|x|}\right)$$

ฟังก์ชันของจำนวนเชิงซ้อน (Complex Number Functions)

สำหรับฟังก์ชันจำนวนเชิงซ้อน จะเขียนอยู่ในรูปแบบ คือ $a \pm b*j$ หรือ $a \pm b*i$ ดังตัวอย่างต่อไปนี้

```
>> X    =    1 + 5 * i
X       =    1.0000 + 5.000i
```

<u>ข้อควรระวัง</u> คือ ขณะที่ใช้ฟังก์ชันจำนวนเชิงซ้อนอยู่ จะต้องไม่มีการกำหนดตัวแปรอื่นเป็น i หรือ j

คำสั่งที่เกี่ยวกับฟังก์ชันมีดังนี้

conj(x) การหาค่า Conjugate ของ **X** เช่น $X=1+5j$ ค่า Conjugate ของ **X** คือ $X=1-5j$

real(x) การแสดงค่าจำนวนจริงของจำนวนเชิงซ้อน เช่น $X=1+5j$ ค่า **real(X)** คือ 1

imag(x) การแสดงค่าจำนวนจินตภาพของจำนวนเชิงซ้อน เช่น $X=1+5j$ ค่า **imag (X)** คือ 5

abs(x) การแสดงขนาด (Magnitude) ของจำนวนเชิงซ้อน เช่น $X=1+5j$ ค่า **abs (X)** คือ 5.0990

angle(x) การแสดงค่ามุมของจำนวนเชิงซ้อน หรือ ใช้คำสั่ง $a\tan 2(imag(x), real(x))$ เป็นการแสดงค่าของ

มุมที่อยู่ระหว่าง π และ $-\pi$

Polynomial Function

ฟังก์ชันของตัวแปรเดียวที่มีรูปแบบดังนี้

$f(x) = a_0 x^N + a_1 x^{N-1} + a_2 x^{N-2} + \cdots + a_{N-2} x^2 + a_{N-1} x^1 + a_N$ การใช้คำสั่งเกี่ยวกับโพลีโนเมียล จะนำเฉพาะสัมประสิทธิ์ (Coefficient) ของสมการมาเขียนในรูปแบบของเวกเตอร์ที่เรียงลำดับจากอันดับสูงสุดไปถึงน้อยสุด ในการป้อนคำสั่งจะต้องมีการกำหนดค่าของ **X** ทุกครั้ง
คำสั่งที่เกี่ยวกับฟังก์ชันมีดังนี้

polyval(a,x) การหาค่าโพลีโนเมียลของ **X** ที่กำหนด โดยการกำหนดสัมประสิทธิ์มาเขียนเวกเตอร์เท่ากับ **a** ดังตัวอย่างต่อไปนี้

```
>> x   =   5 ;
>> a   =   [ 1 , 2 , 3 , 4] ;
>> f   =   poly( a , x )
f      =   194
```

การบวกและลบ Polynomial Function

การบวกและลบ สมการที่เป็นโพลีโนเมียลสามรถกระทำได้ด้วยการนำเวกเตอร์สัมประสิทธิ์ของทั้งสองสมการที่มีขนาดเท่ากันมาทำการ บวกและลบ (ตัวที่ไม่มีค่าให้แทนด้วยศูนย์) ดังตัวอย่างต่อไปนี้

```
>> a   =   [ 2 , 4 , 6 , 0 , 2 ] ;
>> b   =   [ 1 , 3 , -2 , 4 , 9 ] ;
>> c   =   a + b
c      =   3   7   4   4   11
```

conv(a,b) การหาค่าผลคูณของโพลีโนเมียลสองจำนวนแสดงอยู่ในรูปของเวกเตอร์สัมประสิทธิ์ อาจจะมีขนาดไม่เท่ากันก็ได้ ดังตัวอย่างต่อไปนี้

```
>> a   =   [ 3, -5, 6, -2 ] ;
>> b   =   [ 1, 3, 0, -1, 0, 2.5 ] ;
>> g   =   conv( a,b )
g      =   [3, 4, -9, 13, -1, 1.5, -10.5, 15, -5]
```

($g(x) = 3x^8 + 4x^7 - 9x^6 + 13x^5 - x^4 + 1.5x^3 - 10.5x^2 + 15x - 5$**))**

[q,r] = deconv(n,d) การหาค่าผลหารของโพลีโนเมียลสองจำนวนแสดงอยู่ในรูปของเวกเตอร์สัมประสิทธิ์ อาจจะมีขนาดไม่เท่ากันก็ได้ โดยที่ค่า q จะเป็นจำนวนจริง และ r เป็นเศษที่เหลือ ดังตัวอย่างต่อไปนี้

```
>> g   =   [3, 4, -9, 13, -1, 1.5, -10.5, 15, -5] ;
>>  b   =   [ 1, 3, 0, -1, 0, 2.5 ] ;
```

```
>> [ q,r ]      =      deconv( g,b )
q               =      3  -5   6   -2
r               =      0   0   0   0
```

roots(a) การหารากของสมการโพลีโนเมียลในรูปแบบของเวกเตอร์สัมประสิทธิ์ ดังตัวอย่างต่อไปนี้

```
>> p    =      [1, -2, -3, 10] ;
>> r    =      roots( p )
r       =      -2.0000        2.0000 + 1.0000i        2.0000 - 1.0000i
```

poly(r) การแปลงค่ารากของสมการให้อยู่ในรูปแบบของโพลีโนเมียล

```
>> a    =      poly( [-1, 1, 3] )
a       =      1    -3    -1     3
```

การสร้างโปรแกรมใช้งานในโปรแกรม Matlab

การเขียนโปรแกรมที่ต้องให้มีการทำงานแบบกำหนดเงื่อนไขและแสดงความเป็นเหตุ – ผล คำสั่งที่เกี่ยวข้อง มีดังนี้

๑. คำสั่ง **if** เป็นคำสั่งที่แสดงเงื่อนไขในการทำงานเป็นเหตุ – ผล มีรูปแบบดังนี้

```
if การแสดงเงื่อนไขทางตรรกวิทยา (logical expression)
สถานภาพที่ต้องการให้กระทำ (Statements)
end
>> if   g < 50
count = count + 1 ;
disp( g );
end
```

การทำงานของโปรแกรมนี้หมายความว่า ถ้า g มีค่าน้อยกว่า 50 ก็ให้ทำการบวกเพิ่ม 1 จนกว่าค่า g = 50 โปรแกรมจะสิ้นสุดการทำงานทันที

๒. คำสั่ง **else** และ **elseif** คำสั่งนี้จะใช้ร่วมกับคำสั่ง if เป็นการแสดงเงื่อนไขที่นอกเหนือจากที่กำหนดด้วย if โดยที่ elseif จะใช้กำหนดเงื่อนไขที่นอกเหนือจาก if ส่วน else จะแสดงเงื่อนไขสุดท้ายที่เหลือ ดังตัวอย่างต่อไปนี้

```
>> if  temperature  >  100
disp('Too hot – equipment malfunctioning.')
elseif  temperature  >  90
disp('Normal operating range')
```

```
elseif  temperature  >  50
disp('Teperature below desired operating range.')
else
disp('Too cold – turn off equipment')
end
```

๓. คำสั่ง **for** ใช้ในการกำหนดการทำงานในลักษณะที่เป็นวงรอบจะทำซ้ำจนกระทั่งได้ผลลัพธ์ตามที่กำหนด ส่วนใหญ่จะใช้ร่วมกับคำสั่ง if มีรูปแบบดังนี้ คือ

```
for ค่าเริ่มต้น : จุดประสงค์หรือเงื่อนไข
สถานภาพที่กำหนด
end
```

ดังตัวอย่างต่อไปนี้

```
>> for  k = 1 : 10
if k < 10
 k = k+1 ; disp(k) ;
else
disp ('ok')
end        (สำหรับ if statement)
end        (สำหรับ for statement)
```

๔. คำสั่ง **while** ใช้ในการกำหนดการทำงานในลักษณะที่เป็นวงรอบจะทำซ้ำจนกระทั่งเงื่อนไขที่กำหนดจะเป็นจริง มีรูปแบบดังนี้คือ

```
while จุดประสงค์หรือเงื่อนไข
สถานภาพที่กำหนด
end
```

การใช้กับสมการเชิงเส้นตรง เวกเตอร์ และ เมตริกส์

๑. การ transpose เมตริกส์ เป็นการกลับ ตำแหน่งจากแถว (row) ให้เป็นสดมภ์ (column) หรือจาก สดมภ์ให้เป็นแถว เช่น กำหนดให้ $A=(i,j)$ เมื่อ transpose A' จะได้ $A=(j,i)$ ดังตัวอย่างต่อไปนี้

```
 A      =       [1  2  3] ;
>> A'
ans   1
        2
        3
```

๒. การ dot product ของเวกเตอร์จำนวนสองเวกเตอร์จะได้ลัพธ์ดังสมการ $\text{dot product} = A \cdot B = \sum_{i=1}^{N} a_i b_i$ สำหรับเมตริกส์จะเป็นการผลบวกของผลคูณในแต่ละสดมภ์ รูปแบบของคำสั่ง **dot (A , B)** ดังตัวอย่างต่อไปนี้

```
>> A   =    [1  2 ; 3  4 ; 5  6 ] ;
>> B   =    [2  8 ; 7  5 ; 1  3 ] ;
>> dot ( A , B )
ans    =    28   54
```

๓. การคูณเมตริกส์ ระหว่างสองเมตริกส์ (เมตริกส์ A กับ เมตริกส์ B) เป็นการทำ dot product ระหว่างแถวที่ i ของเมตริกส์ตัวแรก กับ สดมภ์ที่ j ของเมตริกส์ตัวสอง จะได้ผลดังสมการ $c_{i,j} = \sum_{k=1}^{N} a_{i,k} b_{k,j}$ ข้อควรระวังคือค่า k ของเมตริกส์ทั้งสองจะต้องเท่ากัน ดังตัวอย่างต่อไปนี้

```
>> A   =    [2  5  1 ; 0  3  -1 ] ;
>> B   =    [1  0  2 ; -1  4  -2 ; 5  2  1 ] ;
>> C   =    A * B
C      =    2   22   -5
           -8   10   -7
```

4. เมตริกส์ยกกำลัง (Matrix Powers) เป็นการนำเมตริกส์นั้นๆ มาคูณซ้ำกันเท่ากับจำนวนกำลังที่กำหนด เช่น เมตริกส์ $A^3 = A * A * A$ เมตริกส์ที่สามารถยกกำลังได้จะต้องเป็นเมตริกส์จัตุรัสเท่านั้น ดังตัวอย่างต่อไปนี้

```
>> A   =    [2  -5  1 ; 6  2  -1 ; 1  8  9 ] ;
>> A^3
ans    =    -106   229   131
           -163  -223    12
            690   463   642
```

๕. การ Inverse เมตริกส์ เป็นการหาส่วนกลับของเมตริกส์ เช่น กำหนดหา inverse ของเมตริกส์ $A = A^{-1}$ มีรูปแบบของคำสั่ง คือ $inv(A)$ เมตริกส์ที่สามารถหา inverse ได้จะต้องเป็นเมตริกส์จัตุรัสเท่านั้น ดังตัวอย่างต่อไปนี้

```
>> A   =    [1 -3 5 ; 4 -1 4 ; 2 3 7 ] ;
>> inv (A)
ans    =    -0.1712    0.3243   -0.0631
           -0.1802   -0.0270    0.1441
            0.1261   -0.0811    0.0991
```

๖. ค่า Determinants ของเมตริกส์ สามารถใช้คำสั่งที่มีรูปแบบของคำสั่ง คือ $\det(A)$ เมื่อกำหนดให้ A เป็นเมตริกส์ใด ๆ เมตริกส์ที่หาค่า Determinants ได้นั้นจะต้องเป็นเมตริกส์จัตุรัสเท่านั้น ดังตัวอย่างต่อไปนี้ Determinants ของเมตริกส์

```
>> A   =     [ 1 -3 5 ; 4 -1 4 ; 2 3 7 ] ;
>> det (A)
ans   =     111
```

๗. การใช้เมตริกส์แก้สมการทางคณิตศาสตร์ที่มีหลายตัวแปร สำหรับสมการที่มีหลายตัวแปรจะใช้วิธีการจัดรูปแบบสมการให้เป็นเมตริกส์ เช่น

$$3x + 2y - z = 10$$
$$-x + 3y + 2z = 5$$
$$x - y - z = -1$$

กำหนดให้อยู่ในรูปแบบของเมตริกส์ ดังนี้ คือ $AX = B$ เมื่อกำหนดให้

$$\mathbf{A}=\begin{bmatrix} 3 & 2 & -1 \\ -1 & 3 & 2 \\ 1 & -1 & -1 \end{bmatrix} \qquad \mathbf{X}=\begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \mathbf{B}=\begin{bmatrix} 10 \\ 5 \\ -1 \end{bmatrix}$$

สำหรับคำตอบของตัวแปร คือ เมตริกส์ $\mathbf{X}$ จาก $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$ มีรูปแบบของคำสั่ง คือ $\mathbf{X} = \mathbf{inv(A)*B}$ ดังตัวอย่างต่อไปนี้

```
>> A   =     [ 3  2  -1 ; -1  3  2 ; 1  -1  -1 ] ;
>> B   =     [ 10 ; 5 ; -1 ] ;
>> X   =     inv(A)*B
X     =     -2.0000        (x = -2 , y = 5 , z = -6)
            5.0000
            -6.0000
```

การเขียนคำสั่งของสมการที่มีตัวแปรและการประมวลผลสมการ

การเขียนสมการที่มีตัวแปรและใช้คำสั่งทางคณิตศาสตร์ที่สำคัญเพื่อทำการแก้สมการเหล่านี้สามารถกระทำได้โดยจะกล่าวเป็นรายละเอียดดังนี้ คือ

๑. การแสดงสมการที่มีสัญลักษณ์ของตัวแปรอยู่ การเขียนสมการหรือฟังก์ชัน ลักษณะนี้จะต้องเขียนอยู่ในเครื่องหมายคำพูดที่มีรูปแบบ คือ ‘สมการที่มีตัวแปร’ ดังแสดงด้วยตัวอย่างต่อไปนี้ **' tan(y/x) ' ' x^3-2x^2+3 ' ' 3a*b-6 '** เป็นต้น

๒. คำสั่งที่เกี่ยวกับการแยกตัวประกอบ

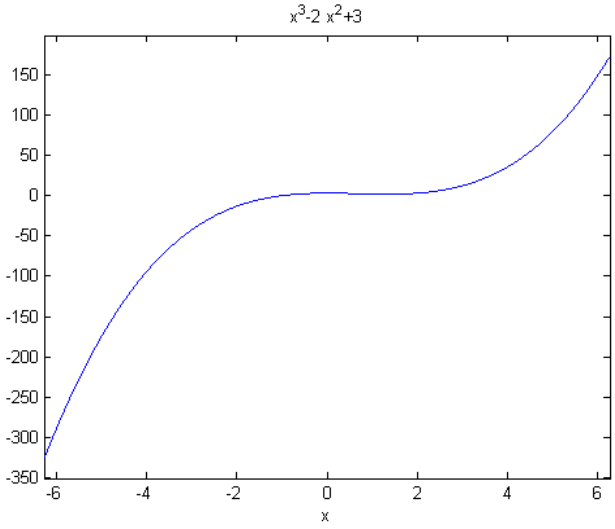
symvar(S) เป็นคำสั่งที่ใช้ในการตรวจสอบตัวแปรที่มีอยู่ในสมการของฟังก์ชัน **S** ดังแสดงด้วยตัวอย่างต่อไปนี้

```
>> S   =     ' 3 * a * b - 6 ' ;
```

```
>> symvar(S)
ans     =       'a'   'b'
```

ezplot(S) เป็นคำสั่งที่ใช้ในการพล็อตฟังก์ชั่นใด ๆ ที่สมมติมีตัวแปรเดียวเปรียบเทียบกับการแทนค่าตัวแปรในสมการแต่ถ้าสมการมีสองตัวแปรจะเป็นการเปรียบระหว่างค่าของทั้งสองตัวแปรนั้นดังแสดงด้วยตัวอย่างต่อไปนี้

```
>> S    =       'x^3-2*x^2+3';
>> ezplot(S)
```



collect(S) เป็นคำสั่งที่ใช้ในการรวบรวม สัมประสิทธิ์ของสมการ **S** ดังแสดงด้วยตัวอย่างต่อไปนี้

```
>> S    =       ' (x-3)^2+(y-4)^2 ' ;
>> collect(sym(S))
ans     =        x^2-6*x+9(y-4)^2
```

factor(S) เป็นคำสั่งที่ใช้ในการแยกแฟคเตอร์ของสมการ **S** ดังแสดงด้วยตัวอย่างต่อไปนี้

```
>> S    =       'x^3-1';
>> factor(sym(S))
ans     =       (x-1)*(x^2+x+1)
```

expand(S) เป็นคำสั่งที่ใช้ในการขยายฟังก์ชัน **S** ดังแสดงด้วยตัวอย่างต่อไปนี้

```
>> S    =       '(x-3)^2+(y-4)^2 ' ;
>> expand(sym(S))
ans     =       x^2-6*x+25+y^2-8*y
```

หนังสืออ้างอิง

Engineering Problem Solving with Matlab ของ Delores M. Etter