เรียนรู้พื้นฐานโปรแกรม

MATLAB

ภายใน*สาม*ชั่วโมง

คำนำ

บทความนี้แต่งขึ้นเพื่อให้ผู้อ่านสามารเริ่มต้นใช้โปรแกรม MATLAB ได้อย่างรวดเร็ว บทความนี้แต่งขึ้นจากประสบการณ์ของผู้เขียน บทความนี้เขียนขึ้นจากการที่มีผู้มาถามผู้เขียนเป็นจำนวนมากว่า MATLAB ใช้งานอย่างไร เพื่อความสะดวกของตัวผู้เขียนเองจึงตัดสินใจที่จะเขียนบทความนี้ขึ้น บทความนี้ประกอบด้วยสามบท โดยผู้อ่านจะใช้เวลาประมาณบทละหนึ่งชั่วโมงในการอ่านและทำความเข้าใจ หลังจากจบ สามบทแล้วผู้เขียนคาดว่าผู้อ่านจะสามารถใช้โปรแกรมนี้ ได้อย่างมีประสิทธิภาพ ที่เหลือคือการนำไปประยุกต์ใช้

กิตติพันธุ์ เตชะกิตติโรจน์

ชั่วโมงแรก (คำสั่งเบื้องต้น)

อะไรคือ MATLAB

MATLAB คือ “เครื่องคิดเลข” เป็นเครื่องคิดเลขที่มีความสามารถ คิดเลข ใช้ตัวแปร (Memory ในเครื่องคิดเลขทั่วไป) ใช้โปรแกรม เขียนโปรแกรม และสามารถเขียนกราฟได้หลายประเภท สาเหตุที่ MATLAB เป็นที่นิยมใช้ในปัจจุบัน เนื่องจากความสามารถในการคิดเลขที่ซับซ้อน และการที่มีโปรแกรมมากมายให้ใช้

MATLAB เป็นโปรแกรมคอมพิวเตอร์ ที่สามารถใช้ได้ในเครื่องคอมพิวเตอร์หลายประเภท (Cross Platform) อาทิเช่น เครื่องคอมพิวเตอร์ที่ใช้ Windows 95/98/NT เครื่องคอมพิวเตอร์ Macintosh และ เครื่องคอมพิวเตอร์ที่ใช้ UNIX

ข้อตกลงของการใช้หนังสือเล่มนี้

หนังสือเล่มนี้จะมีตัวอย่างการใช้ MATLAB ทั้งข้อมูลที่ต้องพิมพ์เข้าไปในโปรแกรม และข้อมูลที่เป็นผลลัพธ์จากโปรแกรม MATLAB เพื่อความสะดวกในการอ่าน สิ่งที่ต้องพิมพ์เข้าในโปรแกรมจะเป็นตัวอักษรประเภท `fixed font` และขึ้นต้นด้วย “»” อาทิเช่น

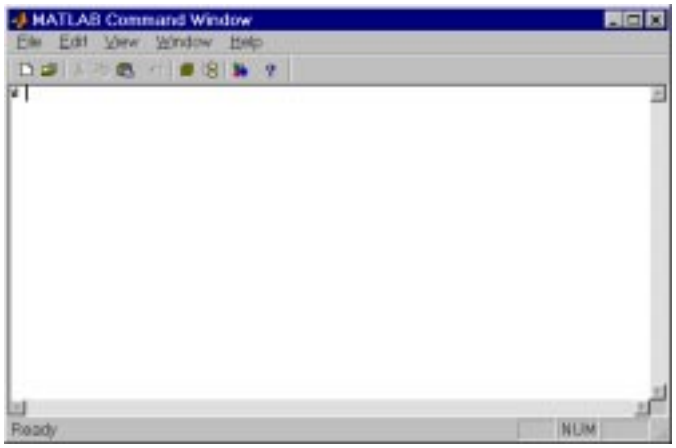

```
» 2+3
```

หลังจากท่านกดปุ่ม “Enter” จะได้ผลลัพธ์ที่หน้าจอ ข้อมูลที่เป็นผลลัพธ์จาก MATLAB จะเป็นตัวอักษรประเภท ***bold italics fixed font*** และขึ้นต้นด้วย “ans =” อาทิเช่น

```
ans =     5
```

เริ่มใช้ MATLAB

ก่อนอื่นเราต้องมีโปรแกรม MATLAB ติดตั้งอยู่ภายในเครื่อง (แนะนำให้ใช้ Version 5.0, 5.1, 5.2 หรือ 5.3 แต่ว่า Version 4.2 ก็ใช้ได้) และเรียกใช้โปรแกรม MATLAB หน้าจอที่ได้ควรจะมีลักษณะดังรูปข้างล่าง



MATLAB เป็นเครื่องคิดเลข

เราจะเริ่มใช้ MATLAB เป็นเครื่องคิดเลข เราสามารถใช้ MATLAB ได้เหมือนเครื่องคิดเลขสมัยใหม่ทั่วไป อาทิเช่น ถ้าเราต้องการบวก 12 กับ 15 ได้ 27 เราใช้

```
» 12+15
ans =    27
```

ภายใน MATLAB มีตัวเลขพิเศษที่เราควรรู้อยู่ 3 ตัว คือ “pi”, “i” และ “j” ค่าจินตภาพ คือ ค่า “i” และ “j” ซึ่งเราสามารถใช้ในการสร้างจำนวนเชิงซ้อน

```
» pi
ans =    3.1416
» 2+3*i
ans =    2.0000 + 3.0000i
» 2+3*j
ans =    2.0000 + 3.0000i
```

เลขในรูปแบบวิทยาศาสตร์ เช่น $2*10^2 = 200$ สามารถใช้ “e” แทนค่าฐาน 10 เช่น

```
» 2e2
ans =    200
```


การคำนวณใน MATLAB สามารถใช้เครื่องหมาย ที่พบเห็นในเครื่องคิดเลขสมัยใหม่ได้ ตารางที่ 1 แสดงถึงเครื่องหมาย หน้าที่ ตัวอย่าง และผลลัพธ์ คำสั่งเหล่านี้ได้จากการพิมพ์

```
» help ops
```

ตารางที่ 1 เครื่องหมายสำหรับการคำนวณอย่างง่าย

เครื่องหมาย	หน้าที่	ตัวอย่าง	ผลลัพธ์
+	แสดงถึงจำนวนบวก	» +13	ans = ***13***
+	บวกเลขสองจำนวน	» 12+13	ans = ***25***
-	แสดงถึงจำนวนลบ	» -13	ans = ***-13***
-	ลบเลขสองจำนวน	» 12-13	ans = ***-1***
*	คูณเลขสองจำนวน	» 12*13	ans = ***156***
/	หารเลขสองจำนวน	» 12/20	ans = ***0.6***
^	ยกกำลัง	» 12^2	ans = ***144***

ในการคำนวณที่ซับซ้อน เราสามารถใช้การเปิดวงเล็บและปิดวงเล็บ ได้เช่นเดียวกับการเขียนด้วยมือ

```
» 3+12*(15+4)/(12+3)
ans =    18.2
```

MATLAB คำนวณเมตริกซ์ และเวกเตอร์

เมตริกซ์ป้อนไว้ใน MATLAB ด้วยการพิมพ์ เปิดวงเล็บก้ามปู "[" พิมพ์ตัวเลขภายในเมตริกซ์ และ ปิดวงเล็บก้ามปู "]" ภายในวงเล็บ เราจะใช้ช่องว่างในการแบ่งสดมภ์ (column) และใช้ semicolumn ";" ในการแบ่งแถว (row) ตัวอย่างเช่น ถ้าเราต้องการป้อน เมตริกซ์ $\begin{bmatrix} 1 & 3 & 5 \\ 9 & 13 & 7 \\ 2 & 4 & 6 \end{bmatrix}$

```
» [1 3 5 ; 9 13 7 ; 2 4 6]
ans =    1    3    5
         9   13    7
         2    4    6
```

เวกเตอร์คือเมตริกซ์ ที่มีเพียงหนึ่งแถว (column vector) หรือหนึ่งสดมภ์ (row vector)

```
» [3 ; 2]
ans =    3
         2
» [3 2]
ans =    3    2
```

ในความเป็นจริงแล้ว MATLAB คิดตัวเลขทุกตัวเป็นเมตริกซ์ ถ้าเป็นเพียงตัวเลขธรรมดา MATLAB จะถือว่าเป็นเมตริกซ์ขนาด 1 แถว (row) 1 สดมภ์ (column)

```
» [3]
```



```
ans =    3
```

เครื่องหมายทุกอย่างที่ใช้กับตัวเลขธรรมดาในตารางที่ 1 สามารถใช้กับเมตริกซ์ได้

เช่น ถ้าต้องการ $\begin{bmatrix} 3 \\ 5 \end{bmatrix} + \begin{bmatrix} 3 & 5 \\ 7 & 2 \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ เราจะป้อนใน MATLAB ได้ดังนี้

```
» [3 ; 5] + [3 5 ; 7 2] * [1 ; 2]
ans =   16
        16
```

ในกรณีที่เราต้องการการคูณหรือหาร ของค่า**ภายในเมตริกซ์ตำแหน่งเดียวกัน** อาทิเช่น คูณ $\begin{bmatrix} 3 & 5 \\ 7 & 2 \end{bmatrix}$ ด้วย $\begin{bmatrix} 2 & 4 \\ 3 & 1 \end{bmatrix}$ ตำแหน่งต่อกัน และได้ผลลัพธ์เป็น $\begin{bmatrix} 3*2 & 5*4 \\ 7*3 & 2*1 \end{bmatrix}$ สามารถทำได้ด้วยใส่จุดข้างหน้าเครื่องหมายคูณหรือ หาร ดังนี้

```
» [3 5 ; 7 2] .* [2 4 ; 3 1]
ans =    6   20
        21    2
```

เมตริกซ์ทรานโพส สามารถหาได้โดยใช้เครื่องหมายหยาดน้ำค้าง (single quote)

```
» [3 5 ; 7 2]'
ans =    3    7
         5    2
```

การคำนวณเมตริกซ์ที่ซับซ้อนเราจะกลับมากล่าวถึงเมื่อ เราศึกษาเรื่องการใช้ฟังก์ชัน

การใช้ฟังก์ชันภายใน MATLAB

สิ่งที่ทำให้ MATLAB เป็นที่นิยมใช้กันทั่วไป คือ การที่มีฟังก์ชันให้ใช้เป็นจำนวนมาก ฟังก์ชันโดยทั่วไปคือการรับค่าเข้าไปคำนวณ และให้ผลลัพธ์ออกมา เราสามารถได้ผลลัพธ์ของฟังก์ชันด้วยการเรียกใช้ชื่อฟังก์ชัน ตามด้วยค่าที่ต้องการใช้ในการคำนวณอยู่ภายในวงเล็บ เช่น ถ้าเราต้องการค่า sin(20) เราจะใช้

```
» sin(20)
ans =    0.9129
```

ถ้าเป็นฟังก์ชั่นที่ต้องการค่า 2 จำนวนเราจะใช้ลูกน้ำคั่นระหว่างกลาง เช่น

```
» ones(1 , 2)
ans =    1    1
```

ใน MATLAB มีฟังก์ชันเป็นจำนวนมาก ฟังก์ชันที่ควรรู้บรรจุในตารางต่อไปนี้

ตารางที่ 2 ฟังก์ชันตรีโกณมิติ (พิมพ์ "» help elfun")

หน่วยที่ใช้สำหรับฟังก์ชันเหล่านี้คือ radian ถ้าต้องการใช้ degree ต้อง คูณด้วย $\frac{\pi}{180}$ อาทิเช่น ถ้าต้องการ Sine ของ 20° เราใช้ "» help sin(20*pi/180)"			
ฟังก์ชั่น	หน้าที่	ตัวอย่าง	ผลลัพธ์
sin	Sine	» sin(1)	ans = ***0.8415***
cos	Cosine	» cos(1)	ans = ***0.5403***
tan	Tangent	» tan(1)	ans = ***1.5574***
asin	Inverse sine	» asin(0.5)	ans = ***0.5236***
acos	Inverse cosine	» acos(0.5)	ans = ***1.0472***
atan	Inverse tangent	» atan(2)	ans = ***1.1071***

ตารางที่ 3 ฟังก์ชันเอ็กโปเนนเชียล (พิมพ์ "» help elfun")

ฟังก์ชั่น	หน้าที่	ตัวอย่าง	ผลลัพธ์
exp	เอ็กโปเนนเชียล	» exp(0.5)	ans = ***1.6487***
log	ล็อกการิทึม ฐานธรรมชาติ	» log(5)	ans = ***1.6094***
log10	ล็อกการิทึม ฐาน 10	» log10(5)	ans = ***0.6990***
log2	ล็อกการิทึม ฐาน 2	» log2(5)	ans = ***2.3219***
sqrt	รากที่ 2	» sqrt(2)	ans = ***1.4142***

ตารางที่ 4 ฟังก์ชันสำหรับเลขเชิงซ้อน (พิมพ์ "» help elfun")

ฟังก์ชั่น	หน้าที่	ตัวอย่าง	ผลลัพธ์
abs	ขนาดของเลขเชิงซ้อน (magnitude)	`» abs(2+i)`	`ans = 2.2361`
angle	ค่ามุมของเลขเชิงซ้อน (phase)	`» angle(2+i)`	`ans = 0.4636`
conj	ค่าคอนจูเกต (complex conjugate)	`» conj(2+i)`	`ans = 2 - 1i`
imag	ค่าจินตภาพ (imaginary part)	`» imag(2+i)`	`ans = 1.0000`
real	ค่าจริง (real part)	`» real(2+i)`	`ans = 2.0000`

ฟังก์ชันต่างๆ ข้างต้นสามารถใช้ได้กับเมตริกซ์ และเวกเตอร์ โดยจะเทียบเท่ากับการใช้ฟังก์ชันกับทุกๆ ตำแหน่งของค่าภายในเมตริกซ์ เช่น

```
» sin([3 5 ; 7 2])
ans =    0.1411   -0.9589
         0.6570    0.9093
```

ผลลัพธ์ข้างต้นคือ $\begin{bmatrix} \sin(3) & \sin(5) \\ \sin(7) & \sin(2) \end{bmatrix}$

```
» log([3 5 ; 7 2])
ans =    1.0986   1.6094
         1.9459   0.6931
```


ผลลัพธ์ข้างต้นคือ $\begin{bmatrix} \log(3) & \log(5) \\ \log(7) & \log(2) \end{bmatrix}$ ไม่ใช่ค่าของ $\log\left(\begin{bmatrix} 3 & 5 \\ 7 & 2 \end{bmatrix}\right)$ ค่าของ $\log\left(\begin{bmatrix} 3 & 5 \\ 7 & 2 \end{bmatrix}\right)$ สามารถหาได้โดยใช้ฟังก์ชันสำหรับคำนวณเมตริกซ์ ซึ่งสร้างขึ้นเพื่อเมตริกซ์โดยเฉพาะ ฟังก์ชันสำหรับคำนวณเมตริกซ์ที่ใช้ทั่วไปได้แก่

ตารางที่ 5 ฟังก์ชันสำหรับคำนวณเมตริกซ์ (พิมพ์ "» help matfun")

ฟังก์ชั่น	หน้าที่
expm	เมตริกซ์เอกโปเนนเชียล
logm	เมตริกซ์ลอกการิทึม
sqrtm	รากที่สองของเมตริกซ์
inv	อินเวอสเมตริกซ์
eig	Eigenvalue
det	ดิเทอร์มิเนนต์
rank	แรงก์ของเมตริกซ์
norm	นอร์มของเมตริกซ์

สำหรับฟังก์ชั่นอื่นอาทิเช่น ค่า Sine ของเมตริกซ์ สามารถหาได้จากคำสั่ง "funm" เช่น

```
» funm([3 5 ; 7 2], 'sin')
```

```
ans =    0.5859   0.2288
         0.3203   0.5401
```

ผลลัพธ์ข้างต้นคือ $\sin\left(\begin{bmatrix}3 & 5\\ 7 & 2\end{bmatrix}\right)$ มิใช่ $\begin{bmatrix}\sin(3) & \sin(5)\\ \sin(7) & \sin(2)\end{bmatrix}$

ฟังก์ชั่นที่ใช้สำหรับงานทั่วไปของเมตริกซ์ และเวกเตอร์ที่ควรรู้ 7 ฟังก์ชั่นสุดท้ายคือ

1) `eye` ใช้ในการหาเมตริกซ์เอกลักษณ์ เราต้องให้ขนาดของเมตริกซ์กับฟังก์ชัน อาทิเช่น เมตริกซ์เอกลักษณ์ขนาด 2x2 ใช้

```
» eye(2)
ans = 1   0
      0   1
```

2) `length` ใช้ในการหาขนาดของเวกเตอร์ ซึ่งเป็นค่าของจำนวนแถว (สำหรับ column vector) หรือจำนวนสดมภ์ (สำหรับ row vector) เช่น

```
» length([1 2 3 4 5])
ans = 5
```

3) `size` ใช้ในการหาขนาดของเมตริกซ์ ฟังก์ชั่นนี้จะให้ 2 ค่า คือค่าของจำนวนแถว และจำนวนสดมภ์ เช่น

```
» size([1 2 3 4 ; 5 6 7 8])
ans = 2   4
```

4) `ones` ใช้ในการสร้างเมตริกซ์ ที่มีแต่เลข 1 ฟังก์ชั่นนี้จะรับ 2 ค่า คือค่าของจำนวนแถว และจำนวนสดมภ์ เช่น

```
» ones(2 , 3)
```



```
ans = 1   1   1
      1   1   1
```

5) `zeros` ใช้ในการสร้างเมตริกซ์ ที่มีแต่เลข 0 ฟังก์ชั่นนี้จะรับ 2 ค่า คือค่าของจำนวนแถว และจำนวนสดมภ์ เช่น

```
» zeros(3 , 2)
ans = 0   0
      0   0
      0   0
```

6) `max` หรือ `min` ใช้ในการหาค่ามากที่สุด และน้อยที่สุดของแต่ละสดมภ์ภายในเมตริกซ์ เช่น

```
» max([1 2 ; 3 4]
ans = 3   4
```

ถ้าเราต้องการค่ามากที่สุดหรือน้อยที่สุดของทั้งเมตริกซ์ เราต้องใช้ฟังก์ชัน `max` หรือ `min` สองครั้ง เช่น

```
» max(max([1 2 ; 3 4])
ans = 4
```

ต้องการหาฟังก์ชันใน MATLAB

หลายๆ ครั้งเราต้องการที่จะหาฟังก์ชั่นภายใน MATLAB อื่นๆ เราสามารถหาได้สองวิธี

1) help ใช้ในการหารายละเอียดของฟังก์ชัน โดยทั่วไปวิธีนี้เราใช้ในกรณีที่เรารู้ชื่อฟังก์ชัน หรือสามารถเดาชื่อของฟังก์ชัน คำสั่ง “help” จะตามด้วยชื่อของฟังก์ชันที่ต้องการรายละเอียด เช่น ถ้าเราต้องการจะรู้รายละเอียดของฟังก์ชั่น Sine เราใช้

```
» ones(2 , 3)
ans = SIN    Sine.
          SIN(X) is the sine of the elements of X.
      Overloaded methods
          help sym/sin.m
```

2) lookfor ใช้ในการหารายชื่อของฟังก์ชัน โดยทั่วไปวิธีนี้เราใช้ในกรณีที่เราไม่รู้ชื่อฟังก์ชัน คำสั่ง “lookfor” จะตามด้วยคำที่สามารถอธิบายหน้าที่การทำงานของฟังก์ชัน เช่น ถ้าเราต้องการจะหาฟังก์ชันที่ทำหน้าที่เกี่ยวกับฟูเรียร์ เราควรพิมพ์

```
» lookfor ‘fourier‘
ans = FFT Discrete Fourier transform.
      FFT2 Two-dimensional discrete Fourier Transfor
      FFTN N-dimensional discrete Fourier Transform.
      IFFT Inverse discrete Fourier transform.
      IFFT2 Two-dimensional inverse discrete Fourier
      IFFTN N-dimensional inverse discrete Fourier
      XFOURIER Graphics demo of Fourier series expan
      INSTDFFT Inverse non-standard 1-D fast Fourier
      NSTDFFT Non-standard 1-D fast Fourier transfor
      EXPFOU Write data to a Fourier vector or a (ma
```



```
IMPFOU Read complex amplitudes from a Fourier
MODIFYFV Modify Fourier (maybe also variance)
SIMFOU Generate simulated Fourier amplitudes.
PLOTFOU Plot contents of Fourier files
DFTMTX Discrete Fourier transform matrix.
FOURIER Fourier integral transform.
IFOURIER Inverse Fourier integral transform.
```

ชั่วโมงที่สอง (ตัวแปร และ กราฟ)

ตัวแปรภายใน MATLAB

ตัวแปรภายใน MATLAB คือความสามารถในการเก็บข้อมูลตัวเลข ภายในหน่วยความจำ และสามารถอ้างถึงด้วยตัวหนังสือ (ชื่อของตัวแปร) ชื่อของตัวแปรสามารถเป็นตัวหนังสือภาษาอังกฤษใดๆ ก็ได้ การใส่ค่าตัวแปรจะใช้เครื่องหมายเท่ากับ "=" การใช้ตัวแปรจะใช้การเรียกชื่อ ตัวอย่างเช่น

```
» a=15
a =     15
» a*a
ans =   225
```

ฟังก์ชันต่างๆ สามารถใช้กับตัวแปรได้เหมือนตัวเลข เช่น

```
» abc=1.5
abc =   1.5
» sin(abc)
ans =   0.9975
```

ในบางครั้งเราไม่ต้องการคำตอบในขณะนั้น เช่นตอนที่เราตั้งค่าตัวแปร เราสามารถสั่งให้ MATLAB ไม่พิมพ์คำตอบด้วยการใส่เครื่องหมาย semicolon ";" ที่สุดบรรทัด เช่น


```
» abc=1.5;
» abcd=1.2;
» abcd+abc
ans =   2.7
```

ด้วยวิธีเดียวกับการตั้งค่าตัวแปร เราสามารถที่จะใส่ค่าของตัวแปร โดยใช้ค่าของการคำนวณที่เสร็จแล้ว เช่น

```
» a=1.1;
» b=1.2;
» c=a*b
c =     1.32
» a+c
c =     2.42
```

สิ่งสำคัญที่เราควรรู้คือ ตัวแปรทุกตัวใน MATLAB คือเมตริกซ์ เราสามารถที่จะใส่ค่าเมตริกซ์ลงในตัวแปร ได้เช่นเดียวกับข้างต้น เช่น

```
» a = [1 2 ; 3 4]
a =     1   2
        3   4
```

การคำนวณ และฟังก์ชั่นสามารถใช้ได้เช่นเดียวกับชั่วโมงก่อนหน้านี้ เช่น

```
» a = [1 2 ; 3 4];
» b = [3 5 ; 7 9];
» a*b
ans =   17  23
        37  51
```

การมีตัวแปรที่ซับซ้อนทำให้เราสามารถทำการคำนวณที่ซับซ้อน ได้ง่าย เช่น

```
» a = [1 2 3 4];
» b = [3 4 5 6]';
» c = [1 3 ; 5 7];
» d = (a*b)*c
d =      50 150
        250 350
```

หรือ

```
» s = -2.3;
» (s^3+5*s^2+11*s+1)/s/(s^2+0.7*s+0.49)
ans =    1.0444
```

ในบางครั้งเราต้องการกำหนด เวกเตอร์ที่มีความยาวมากๆ เช่น เวกเตอร์ที่มี ค่าจาก 0 0.1 0.2 ไปจนถึง 1 เราสามารถทำได้ด้วยการใช้เครื่องหมาย colon ":" เช่น

```
» a = [0:0.1:1]
a =      0   0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
```

สังเกตได้ว่ามีตัวเลขอยู่สามตัวคั้นด้วย colon ที่เราใช้ ตัวแรก (0) เป็นการบอกค่าเริ่มต้น ตัวที่สอง (0.1) เป็นการบอกว่าตัวเลขภายในเวกเตอร์ ห่างกันเท่าใด และตัวสุดท้าย (1) เป็นการบอกว่าตัวเลขในเวกเตอร์สิ้นสุดที่ใด ตัวเลขตัวที่สองสามารถละไว้ได้ ถ้าไม่มีตัวเลขตัวที่สอง โปรแกรมจะให้ค่าตัวเลขห่างกัน 1 เช่น

```
» a = [0:5]
```



```
a =      0   1   2   3   4   5
```

นอกจากนี้ในกรณีที่เราต้องการนำค่า แถว หรือ สดมภ์ของเมตริกซ์มาเป็นค่าเวกเตอร์ อาทิเช่นถ้าเราต้องการค่าในแถวที่ 2 ของ $\begin{bmatrix} 3 & 5 \\ 7 & 2 \end{bmatrix}$ เราสามารถใช้เครื่องหมาย colon แทนทุกค่าที่อยู่ภายในแถวนั้น

```
» a = [3 5 ;7 2];
» a(2,:)
ans =    7   2
```

ถ้าเราต้องการค่าในสดมภ์ที่ 1 ของเมตริกซ์ "a" เราสามารถใช้เครื่องหมาย colon แทนค่าที่อยู่ในสดมภ์นั้นๆ

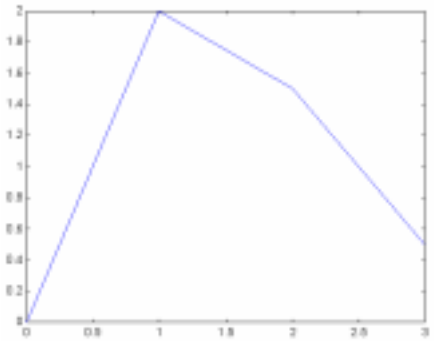
```
» a(:,1)
ans =    3
         7
```

การเขียนกราฟด้วย MATLAB

ความสามารถที่สำคัญอีกอย่างหนึ่งของ MATLAB คือความสามารถในการเขียนกราฟ MATLAB มีความสามารถในการเขียนกราฟได้หลายรูปแบบ กราฟที่เราใช้บ่อยมาก คือการเขียนกราฟ สองแกน (x-y plot) ในการเขียนกราฟประเภทนี้เราจำเป็นต้องมีสองเวกเตอร์ เวกเตอร์แรกเป็นค่าของแกน x และเวกเตอร์ที่สองเป็นค่าของแกน y เวกเตอร์ทั้งสองตัวนี้ต้องมีขนาดที่เท่ากัน อาทิเช่น ต้องการ เขียนกราฟ ระหว่างค่า (0,0) (1,2) (2,1.5) (3,0.5) เราสามารถเขียนกราฟด้วยคำสั่ง

```
» plot([0 1 2 3],[0 2 1.5 0.5]);
```

เราจะได้กราฟข้างล่างอยู่ในหน้าต่างใหม่ ถ้าเรามีการใช้คำสั่ง “plot” ก่อนหน้านี้ กราฟจะอยู่ในหน้าต่างที่เคยเปิดไว้ก่อน



ถ้าเราต้องการเขียนกราฟของฟังก์ชั่นเราสามารถทำได้ด้วยการสร้างเวกเตอร์ขึ้นมาสองเวกเตอร์ เวกเตอร์แรกจะเป็นค่าของแกน x และเวกเตอร์ที่สองเป็น

ค่าของฟังก์ชั่น ซึ่งเป็นค่าในแกน y โดยทั่วไปแล้วค่าในแกน x จะเป็นค่าตัวเลขที่เรียงต่อกันไปเช่นจาก 0 ถึง 10. เราสามารถใช้คำสั่งในการสร้างเวกเตอร์นี้โดยใช้เครื่องหมาย colon คือ

```
» t = [0:0.1:10];
```

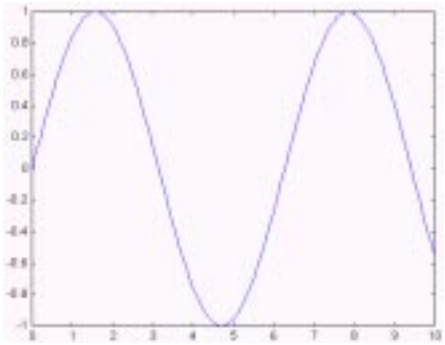
ตัวแปร “t” จะเป็นเวกเตอร์ ที่มีค่าตั้งแต่ 0 0.1 0.2 จนถึง 9.8 9.9 10 (ทบทวนในเรื่องของตัวแปร) ในส่วนของค่าฟังก์ชั่น สมมติว่าต้องการเขียนกราฟของฟังก์ชั่น sin(t) เราต้องสร้างเวกเตอร์สำหรับค่าในแกน y โดย

```
» ft = sin(t);
```

ตัวแปร “ft” จะเป็นเวกเตอร์ ที่เก็บค่า sin(0) sin(0.1) จนถึง sin(9.9) sin(10) ส่วนที่เหลือคือสั่งเขียนกราฟโดยใช้ค่าเวกเตอร์ทั้งสอง

```
» plot(t,ft);
```

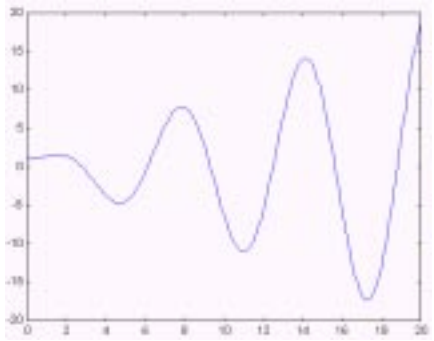
เราจะได้กราฟ



โดยสรุปแล้วการเขียนกราฟ เราใช้เวกเตอร์สองตัว ตัวอย่างถัดไป ถ้าเราต้องการเขียนกราฟของฟังก์ชั่น $t\sin(t)+\cos(t)$ โดยใช้เขียนกราฟจาก $t=0$ จนถึง $t=20$ เราจะสามารถเขียนโดย

```
» t = [0:0.05:20];
» x = t.*sin(t) + cos(t);
» plot(t,x);
```

สังเกตว่าเราใช้ ".*" (จุดก่อนเครื่องหมายคูณ) เพราะว่าเราต้องการที่จะคูณค่าในเวกเตอร์ค่าต่อค่า ถ้าเราไม่ใส่จุด จะมีความผิดพลาดที่ขนาดของเวกเตอร์ไม่เหมาะสมที่จะคูณกัน (Inner matrix dimensions must agree.)

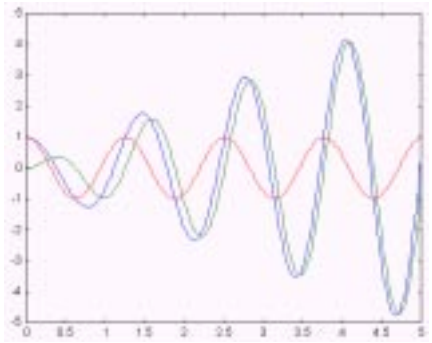


```
ans =    17   23
         37   51
```

ถ้าเราต้องการจะเขียนกราฟมากกว่าหนึ่งเส้นอยู่ด้วยกัน เราสามารถทำได้ด้วยการใส่เวกเตอร์ของแกน x และ แกน y ต่อไปเรื่อยๆภายในฟังก์ชั่น "plot" เช่น ถ้าเราต้องการเขียนกราฟ ของ $t\sin(5t)+\cos(5t)$ $t\sin(5t)$ และ $\cos(5t)$ เราสามารถทำได้โดย


```
» t = [0:0.01:5];
» x1 = t.*sin(5*t)+cos(5*t);
» x2 = t.*sin(5*t);
» x3 = cos(5*t);
» plot(t,x1,t,x2,t,x3);
```

จะเห็นได้ว่าตัวแปร "x1" "x2" และ "x3" เป็นตัวแปรที่ใช้เก็บค่าของฟังก์ชั่นซึ่งเป็นค่าในแนวแกน y ตัวแปร "t" จะเก็บค่าในแนวแกน x กราฟที่ได้ควรจะเป็น

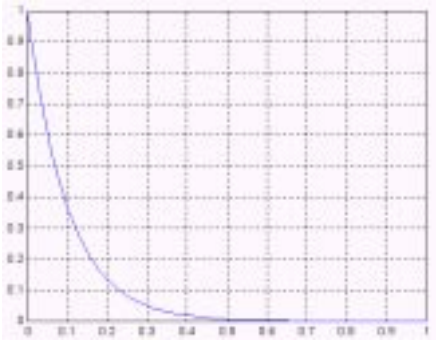


ในตอนนี้เราควรจะสามารถเขียนกราฟภายใน MATLAB ได้ ต่อจากนี้คือการทำให้กราฟง่ายต่อการอ่าน คำสั่งต่อไปนี้จะช่วยให้กราฟมีความง่ายต่อการใช้

1) `grid` ใช้ในการเขียนตารางเพื่อให้การอ่านค่าจากกราฟมีความง่ายขึ้น คำสั่ง "grid" เมื่อเรียกใช้จะเขียนตารางขึ้นภายในกราฟ ถ้ามีการเรียกใช้อีกครั้งจะลบตารางออก กลับไปกลับมา ถ้าเราต้องการมั่นใจว่าตารางถูกเขียนเราสามารถใช้ "grid on" เช่น

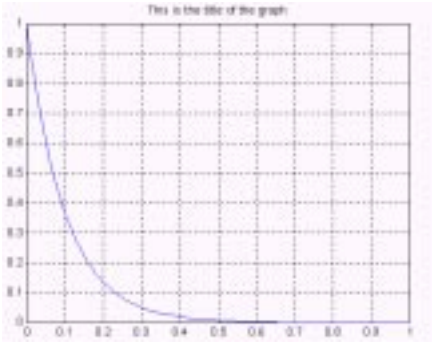
   ```
   » t = 0:0.01:1;
   ```

```
» x = exp(-10*t);
» plot(t,x);
» grid on
```



2) `title` ใช้ในการเขียนหัวข้อของกราฟ เช่น

```
» title('This is the title of the graph');
```

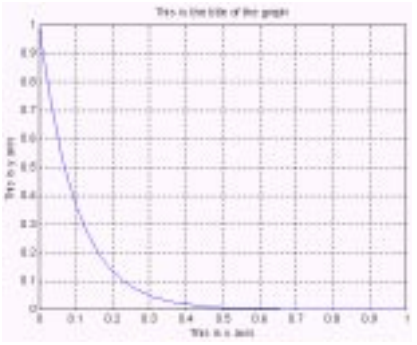


3) `xlabel` และ `ylabel` ใช้ในการเขียนชื่อของแกน x และ y ตามลำดับ เช่น

```
» xlabel('This is x axis');
```



```
» ylabel('This is y axis');
```



คำสั่งใน “grid” “title” “xlabel” และ “ylabel” นี้โดยทั่วไปจะใช้ภายหลังจากใช้คำสั่ง “plot”

นอกจากฟังก์ชั่น “plot” เรายังสามารถเขียนกราฟประเภทอื่นได้โดยใช้ฟังก์ชั่นภายในตารางข้างล่าง ฟังก์ชั่นเหล่านี้มีวิธีใช้ที่เหมือนฟังก์ชั่น “plot”

ตารางที่ 6 ฟังก์ชันสำหรับเขียนกราฟที่มีวิธีใช้เหมือน ฟังก์ชั่น “plot”

ฟังก์ชั่น	หน้าที่
semilogx	กราฟที่มีแกนนอนเป็น ลอกการิทึม
semilogy	กราฟที่มีแกนตั้งเป็น ลอกการิทึม
loglog	กราฟที่มีทั้งสองแกนเป็น ลอกการิทึม

ชั่วโมงที่สาม (script และ ฟังก์ชั่น)

การเขียน script ใน MATLAB

ความสะดวกสบายอย่างหนึ่งของ MATLAB คือความสามารถในการเก็บคำสั่งในการคำนวณไว้ในแฟ้มข้อมูล (file) และสามารถเรียกนำมาใช้ได้ใหม่ การเขียน script จะเป็นเหมือนการป้อนข้อมูลเข้าไปภายใน MATLAB เพียงแต่ว่าเราจะเขียนไว้ในแฟ้มข้อมูล เวลาจะนำการคำนวณมาใช้ ก็เพียงแต่เรียกชื่อแฟ้มข้อมูลนั้นๆ

ข้อที่ควรระวังคือ แฟ้มข้อมูลต้องมีนามสกุลเป็น ".m" และต้องอยู่ใน directory ที่ MATLAB กำลังทำงานอยู่ เราสามารถตรวจสอบ directory ที่ MATLAB กำลังทำงานอยู่โดยใช้คำสั่ง pwd เช่น

```
» pwd
ans =    D:\Program Files\MATLABR11\work
```

ถ้าเราต้องการเปลี่ยนไปอยู่ใน directory อื่น เราสามารถใช้คำสั่ง cd เช่น ถ้าเราต้องการเปลี่ยนไปใช้ข้อมูลใน drive a: เราจะพิมพ์

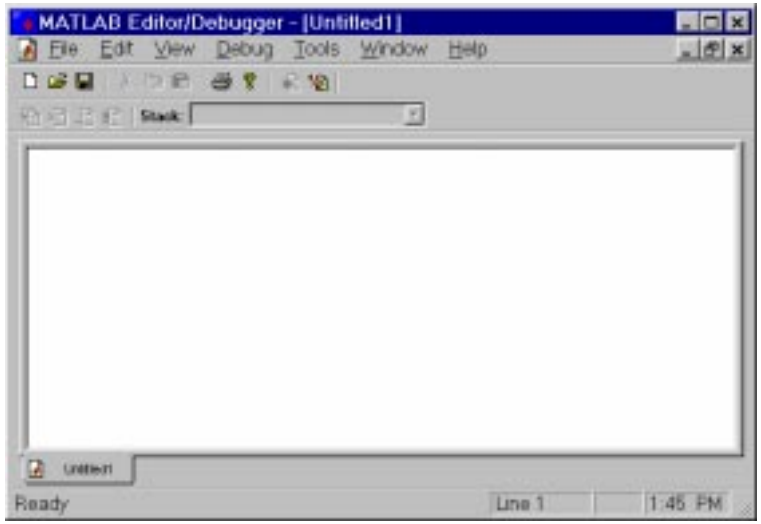
```
» cd a:
```

คำสั่ง dir จะใช้ดูชื่อของแฟ้มข้อมูลที่เรามีอยู่

เริ่มเขียน Script

เราสามารถที่จะเขียน script โดยใช้ text editor ได้ทุกประเภท ถ้าจะให้สะดวกขอแนะนำ medit ของ MATLAB เราสามารถเรียกใช้ medit โดยพิมพ์ edit

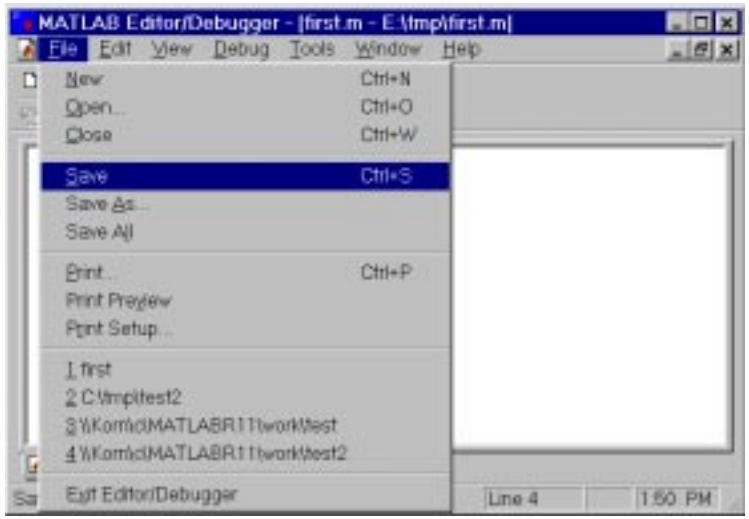
```
» edit
```



โปรแกรม medit จะถูกเรียกขึ้นมา ถ้าเราต้องการเขียน script เพื่อคำนวณบวก ลบ คูณ หาร เราจะพิมพ์ (ไม่ต้องพิมพ์ "»")

```
» a=2+3;
» b=2-3;
» a*b
» a/b
```

จากนั้น จัดเก็บแฟ้มข้อมูลพร้อมใส่ชื่อ ที่ต้องการเช่น “first” (ในกรณีแฟ้มข้อมูลชื่อ “first.m”)



เมื่อข้อมูลได้ถูกจัดเก็บแล้วเราสามารถเรียกใช้ script ที่เขียนขึ้นได้ภายใน MATLAB โดยใช้ชื่อของแฟ้มข้อมูล `first` ในกรณีนี้

```
» first
ans =    -5
ans =    -5
```

จะสังเกตได้ว่า สองบรรทัดแรกใน แฟ้มข้อมูล มีเครื่องหมาย semicolon ทำให้ไม่มีการแสดงผลลัพธ์ ของสองบรรทัดแรก ผลลัพธ์ที่ได้เป็นผลลัพธ์ของสองบรรทัดสุดท้าย

จะเห็นได้ว่าการเขียน script มีประโยชน์มากเมื่อมีการแก้ไข เช่นถ้าเราต้องการเปลี่ยนค่าของ ตัวแปร a ให้เป็น 20 เราสามารถเข้าไปแก้ภายใน script และเรียกให้คำนวณใหม่ได้ภายใน MATLAB

เขียนโปรแกรมใน MATLAB

ภายใน MATLAB อนุญาติให้เราสามารถใช้ความสามารถ ในลักษณะของโปรแกรมได้ นั่นคือ Condition และ Loop ความสามารถนี้ใช้ได้ทั้งใน MATLAB โดยตรง และ ใช้ใน script โดยทั่วไปนิยมใช้ใน script เนื่องจากมีความเข้าใจง่ายกว่า

เริ่มต้นด้วยการทำคำสั่งที่วนไปมา เช่นถ้าเราต้องการ หาค่าของ $\sum_{i=0}^{\infty}\frac{1}{(i+2)^2}$ เราสามารถใช้ “for … end” คำสั่ง FOR ตามด้วยตัวแปรที่ต้องการจะเปลี่ยนค่า (ในกรณีนี้ คือ i) ตามด้วยค่าที่ต้องการจะเปลี่ยน เช่น (แนะนำให้พิมพ์ใน แฟ้มข้อมูล เพื่อที่จะสะดวกในการแก้ไข)

```
» sum = 0
» for i = 0:1:30,
»   i
»   sum = sum + 1/(i+2)^2
» end
```

เมื่อเรียกใช้ script ที่เขียนขึ้นจะสังเกตว่าบรรทัดที่สาม จะเริ่มตั้งแต่ 0, 1, 2 จนถึง 30 และบรรทัดที่ สี่จะเป็นค่าผลรวมที่เพิ่มขึ้นเรื่อยๆ

ในกรณีที่เราต้องการให้การคำนวณเกิดขึ้นอย่างต่อเนื่องจนกระทั่งเงื่อนไขบางอย่างเกิดขึ้น ตัวอย่างเช่นถ้าเราต้องการหา $\sum_{i=0}^{\infty}\frac{1}{(i+2)^2}$ แต่ใน MATLAB การคำนวณต้องเป็นจำนวนที่จำกัด ถ้าเราต้องการหาค่าโดยประมาณ เราสามารถทำได้โดยการบวกสะสมไปเรื่อยจนกระทั่งค่าที่จะบวกเพิ่มมีค่าน้อยกว่า ค่าน้อยๆ ค่าหนึ่ง เช่น 0.00000005 ดังนี้

```
» sum = 0;
» i = 0;
» while (1/(i+2)^2)> 0.00000005,
»   sum = sum + 1/(i+2)^2
»   i = i+1;
» end
```

เมื่อเรียกใช้ script ที่เขียนขึ้นจะสังเกตว่าบรรทัดที่สี่ จะบวกสะสมไปเรื่อยๆ จนกระทั่งความละเอียดระดับหนึ่ง (วิธีนี้ถูกเฉพาะบางอนุกรมเท่านั้น)

การใช้เงื่อนไข if … end จะใช้ในกรณีต้องการตัดสินใจว่าจะทำอย่างใดอย่างหนึ่ง อย่างเช่น จะหาผลบวกของอนุกรมด้านบน แต่เฉพาะค่า i เป็นเลขจำนวนคู่ ถ้าค่า i เป็นจำนวนคี่เราจะนำมาลบออก เราสามารถทำได้โดยใช้ script

```
» sum = 0;
» i = 0;
» while (1/(i+2)^2)> 0.00005,
»   if mod(i,2)==0
```



```
»     sum = sum + 1/(i+2)^2
»   else
»     sum = sum - 1/(i+2)^2
»   end
»   i = i+1;
» end
```

ในกรณีนี้เราใช้ 0.00005 เพื่อที่จะประหยัดเวลาในการทำงาน จะสังเกตว่าเงื่อนไขของการตัดสินใจ จะตามหลังคำสั่ง if ถ้าเงื่อนไขที่กำหนดเป็นจริงจะทำบรรทัดถัดมา ถ้าเงื่อนไขไม่เป็นจริงจะทำบรรทัดที่ต่อจากคำสั่ง else

สร้างฟังก์ชั่นใน MATLAB

ฟังก์ชั่นทำให้การเขียนโปรแกรมภายใน MATLAB เป็นไปอย่างสมบูรณ์ ฟังก์ชั่นต่างจาก script ในลักษณะของการใช้งาน ฟังก์ชั่นจะมีการส่งค่าเข้าและออกจากฟังก์ชั่นได้ เหมาะกับการแบ่งการคำนวนให้เป็นสัดส่วน และ นำกลับมาใช้ใหม่ได้สะดวก

ฟังก์ชั่นภายใน MATLAB จะมีลักษณะคล้าย script หนึ่งฟังก์ชั่นจะอยู่ภายในหนึ่งแฟ้มข้อมูล ชื่อของแฟ้มข้อมูลควรจะเป็นชื่อเดียวกับฟังก์ชั่น (เพื่อป้องกันการสับสนในการใช้งาน)

ข้อแตกต่างของฟังก์ชั่น และ script คือส่วนของการรับค่า และส่งค่ากลับ ซึ่งหคือส่วนเริ่มต้น ของแฟ้มข้อมูล ต้องมีคำสั่ง `function` ตามด้วย

1) ชื่อของตัวแปรที่จะส่งออกจากฟังก์ชั่น และ เครื่องหมายเท่ากับ (ไม่ต้องมีถ้า ไม่มีการส่งค่ากลับ)
2) ชื่อของฟังก์ชั่น (ซึ่งควรจะเหมือนกับชื่อ แฟ้มข้อมูล)
3) เครื่องหมายเปิดวงเล็บ ชื่อของตัวแปรที่รับเข้ามาในฟังก์ชั่น แล้วตามด้วยเครื่องหมายปิดวงเล็บ

ตัวอย่างเช่น ถ้าเราต้องการฟังก์ชั่นที่จะหาค่าของ $\sum_{i=0}^{N}\frac{1}{(i+2)^2}$ ฟังก์ชั่นนี้จะรับค่า N แล้วให้ค่าของผลบวกคืน เราจะสร้างฟังก์ชั่นโดยการพิมพ์คำสั่งข้างล่างใส่ในแฟ้มข้อมูลชื่อ “findsum” (ไม่ต้องพิมพ์ “»”)

```
» function result=findsum(n)
» sum = 0;
» for i = 0:1:n,
»   sum = sum + 1/(i+2)^2;
» end
» result = sum
```

บรรทัดแรก เป็นการแสดงถึงแฟ้มข้อมูลนี้เป็นฟังก์ชั่น โดยที่จะส่งค่าของตัวแปร `result` ออกจากฟังก์ชั่น ค่าของ `n` จะถูกส่งเข้ามาในฟังก์ชั่น บรรทัดสุดท้ายเป็นการบอกว่าค่าที่ส่งกลับจะมีค่าเท่าใด (บรรทัดสุดท้ายจะไม่จำเป็น ถ้าบรรทัดแรกใช้ `sum` แทน `result` เมื่อต้องการใช้งาน เราพิมพ์ `findsum(…)` เช่น

```
» findsum(5)
ans =    0.51179705215420
```


หรือ

```
» a=findsum(20);
» a
ans =    0.60049693331165
```

ถ้าในกรณีที่มีการผ่านค่าเข้ามาในฟังก์ชั่นมากกว่าหนึ่งตัว ให้แยกตัวแปรแต่ละตัวด้วยเครื่องหมายลูกน้ำ (“,”) และในกรณีที่ต้องการส่งค่ากลับมากกว่าหนึ่งค่าให้แยกตัวแปรแต่ละตัวด้วยเครื่องหมายลูกน้ำ (“,”) แล้วรวมกันด้วยเครื่องหมาย square bracket (“[…]”) เช่นฟังก์ชั่นข้างบนถ้าต้องการให้คำนวณสองค่าพร้อมๆ กันจะต้องแก้ฟังก์ชั่นเป็น (ใช้แฟ้มข้อมูลชื่อ “findsum2”)

```
» function [result1, result2]=findsum2(n1,n2)
» sum = 0;
» for i = 0:1:n1,
»   sum = sum + 1/(i+2)^2;
» end
» result1 = sum;
» sum = 0;
» for i = 0:1:n2,
»   sum = sum + 1/(i+2)^2;
» end
» result2 = sum;
```

เมื่อต้องการใช้งาน เราพิมพ์ `findsum2(…)` เช่น

```
» [a, b]=findsum2(5,20);
» a
```

```
ans =    0.51179705215420
» b
ans =    0.60049693331165
```

ภายในฟังก์ชั่น สามารถเรียกใช้ฟังก์ชั่นได้ เช่น ตัวอย่างก่อนหน้านี้สามารถย่อได้เป็น

```
» function [result1, result2]=findsum2(n1,n2)
» result1 = findsum(n1);
» result2 = findsum(n2);
```

หมดสามชั่วโมง

ถึงจุดนี้เราควรที่จะสามารถใช้ MATLAB ได้พอสมควร สิ่งที่เหลือคือ การพลิกแพลง และการหาฟังก์ชั่นและความสามารถใหม่ๆ มาใช้เพิ่มเติม

แหล่งข้อมูลอ้างอิง

แหล่งข้อมูลอ้างอิงที่ดีที่สุด คือ MATLAB help files ในช่วงแรกข้อมูลใน help files อาจจะดูอ่านยาก แต่เมื่อคุ้นเคย จะเป็นแหล่งข้อมูลที่ดีมาก แหล่งข้อมูลอ้างอิงอื่นๆ ได้แก่

1. www.mathworks.com โดย บริษัท The Mathworks, Inc.
2. Robert H. Bishop, “Modern Control Systems Analysis & Design: Using MATLAB & SIMULINK”, Addison Wesley Longman, 1997

กิตติพันธุ์ เตชะกิตติโรจน์