

GUÍA BÁSICA DE ESTUDIO DE MATLAB

Mauricio Monsalve M.

Esta guía está orientada a ayudarle a estudiar Matlab. Por favor resuelva la guía usando Matlab o alguna alternativa como Octave o Scilab.

Los ejercicios con asterisco (*) se dejan como propuestos.

Bajando aplicaciones

Para tener **Matlab** usted necesita comprarlo. Es software comercial.

Como **no comprará Matlab** sólo para estudiar, puede descargar las siguientes alternativas: **Scilab** u **Octave**. Si tiene Windows, prefiera Scilab. *Las alternativas no son tan potentes como Matlab.*

- **Scilab:** funciona muy bien tanto en Windows como en Linux. Su instalación es sencilla. Es algo más quisquilloso que Matlab y Octave en la sintaxis; para las constantes hay que anteponer el porcentaje, así: **%pi**. Esto no se da ni Matlab ni en Octave (se usa **pi** a secas). También, para cerrar un bloque *while*, *for*, *if*, *function* es necesario escribir explícitamente **endwhile**, **endfor**, **endif**, **endfunction**. En Matlab y en Octave esto se puede evitar tan sólo con **end**, pero usar la sintaxis end-algo no es incorrecto (de hecho es más fácil de leer). Sitio web de Scilab: <http://www.scilab.org>. Descargable para Windows (98/XP/Vista): <http://www.scilab.org/download/4.1.1/scilab-4.1.1.exe>
- **Octave:** funciona muy bien en Linux. Tiene también instalación en Windows, aunque probablemente no sea sencilla de realizar. Sitio web: <http://www.gnu.org/software/octave/>. Descarga Windows (SourceForge): http://sourceforge.net/project/showfiles.php?group_id=2888&package_id=40078&release_id=404493

Ejercicios generales

Definiendo vectores y matrices

1> Defina los vectores A, B y C como la base canónica de \mathbb{R}^3 . Use la notación vertical:

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

2> Defina en M la siguiente matriz:

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

3> Defina en M la siguiente matriz:

$$\begin{pmatrix} 1 & 2 \\ -2 & -1 \end{pmatrix}$$

4> ¿Cuál es la traspuesta de M?

5> Obtenga $M \cdot M^T$, la multiplicación de M por su traspuesta.

6> Obtenga la inversa de M y multiplíquela por M. Debiera obtener la matriz de identidad en $M_{2 \times 2}$.

7> Cree un vector horizontal de números entre 0 y 1 a pasos de 0,05 (0, 0,05, 0,1, 0,15, ... , 1).

8> Resuelva el siguiente sistema lineal:

$$4x + 5y + 6z = 1$$

$$3x + 5y + 7z = 2$$

$$9x - 2y - 3z = 1$$

9> Cree un vector horizontal con los cosenos de $-\pi$ hasta $+\pi$, en 11 partes. Llámelo C.

10> Cree un vector horizontal con los senos de $-\pi$ hasta $+\pi$, en 11 partes. Llámelo S.

11> Cree el vector CS tal que $CS_i = C_i^2 + S_i^2$. Debería resultar un vector horizontal de unos (o de números muy cercanos a uno).

12* Repita el procedimiento anterior, pero en vez de usar coseno y seno (cos & sin), use coseno hiperbólico y seno hiperbólico (cosh & sinh). Ahora cree CS tal que $CS_i = C_i^2 - S_i^2$. (Se debe verificar la identidad $\cosh^2(x) - \sinh^2(x) = 1$.)

13* ¿Por qué el coseno hiperbólico se llama como tal? Véalo usando: $\cos(.5)$ y $\text{real}(\cosh(.5*i))$.

Soluciones

1> $A=[1;0;0]$, $B=[0;1;0]$, $C=[0;0;1]$

Observaciones: el punto y coma (;) indica que se pasa a la siguiente fila. Para pasar a la siguiente columna, puede usar espacios o comas.

2> $M=[1\ 0\ ;\ 0\ -1]$

3> $M=[1\ 2\ ;\ -2\ -1]$

4> M' (Obs: la apóstrofe (') significa trasposición)

5> $M*M'$

6> $\text{inv}(M)*M$ (Obs: *inv* entrega la inversa de una matriz)

7> Hay dos maneras sencillas:

$0 : 0.05 : 1$ (Recorre de 0 a 1 a intervalos de 0,05)

$\text{linspace}(0,1,21)$ (Observar que 0,05 cabe 20 veces entre 0 y 1, y a esto contarle el principio para obtener 21)

8> Dos soluciones:

$A=[4\ 5\ 6;\ 3\ 5\ 7;\ 9\ -2\ -3]$; $b=[1;2;1]$; $A\b b$

Observación: $A\b b$ resuelve el sistema $Ax=b$.

$A=[4\ 5\ 6;\ 3\ 5\ 7;\ 9\ -2\ -3]$; $b=[1;2;1]$; $\text{inv}(A)*b$

Observación: $Ax=b \Rightarrow A^{-1}Ax=A^{-1}b \Rightarrow x=A^{-1}b$.

9> $C=\cos(\text{linspace}(-\pi,\pi,11))$

Observación: el coseno de un vector entrega como

resultado el vector evaluado por componentes.

10> $S=\sin(\text{linspace}(-\pi,\pi,11))$

Observación: el seno de un vector entrega como resultado el vector evaluado por componentes.

11> $CS = C.^2 + S.^2$

Observación: la potencia (^) está definida para matrices (y los vectores con matrices). Por ello, es necesario forzar su evaluación coordenada a coordenada. Esto se hace anteponiendo un punto (.) al operador.

Lo anterior no afecta a las operaciones coordenada a coordenada como la suma. Pruebe con la definición del ejercicio 1 que: $A + B = A .+ B$

Dibujando con PLOT

1> Dibuje un segmento de recta entre el punto (0, 5) y (2, 3).

2> Dibuje la curva coseno entre 0 y 6,29. Use 10 puntos.

3> Dibuje la curva seno entre 0 y 6,29. Use 100 puntos.

4> Dibuje la curva $(1-p)^p$ sujeto a $0 < p < 1$. Use 300 puntos.

5> Dibuje un triángulo usando plot.

6> Dibuje un círculo usando plot. Use 50 puntos.

Soluciones

1> $\text{plot}([0\ 2], [5\ 3])$;

Observación: plot recibe un primero un vector horizontal de coordenadas en el eje x y luego un vector vertical de coordenadas en el eje y.

2> $x=\text{linspace}(0,6.29,10)$; $\text{plot}(x, \cos(x))$;

3> $x=\text{linspace}(0,6.29,100)$; $\text{plot}(x, \sin(x))$;

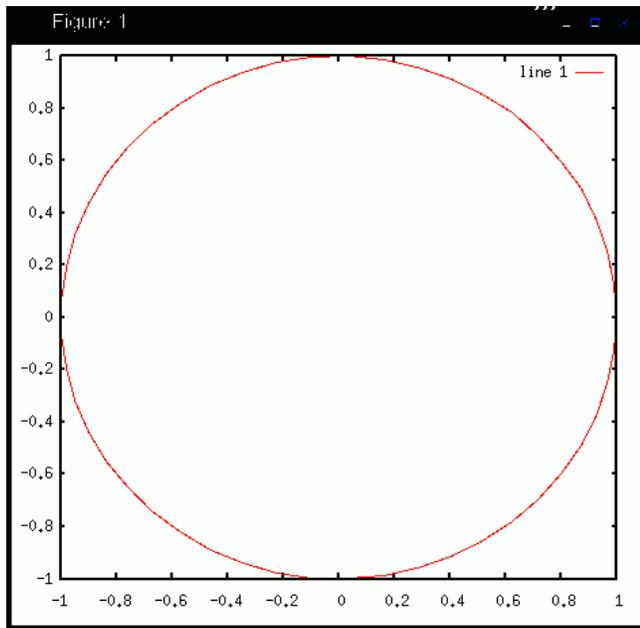
4> $p=\text{linspace}(0,1,300)$; $\text{plot}(p, (1-p).^p)$;

Observación: debo hacer multiplicación coordenada a coordenada, por lo que **debo** usar .* para ello.

5> $\text{plot}([0\ 1\ 2\ 0], [0\ 2\ 1\ 0])$;

Observación: dibujé (0,0), (1,2), (2,0) y (0,0) nuevamente para cerrar la figura. Con los cuatro puntos hago los tres lados del triángulo.

```
6> x=linspace(0,6.29,50); plot( cos(x), sin(x) );
```



(Ejemplo de círculo. Generado con Octave.)

Definiendo funciones

1> Cree una función que calcule el seno de un ángulo en grados sexagesimales (0° a 360°). Llámelo “seno” (sí, en castellano). Báse en la equivalencia π *radiantes* = 180 *grados*. Use esta función para dibujar el seno entre 0 y 360°.

2> Cree una función recursiva que obtenga los números de Fibonacci: 1, 1, 2, 3, 5, 8, 13, 21, etc. La sucesión está definida por $f(n+2)=f(n+1)+f(n)$, considerando $f(0)=f(1)=1$. Dibuje los 30 primeros elementos de esta sucesión.

3> Una función contractante¹ es una función con derivada menor a 1 en norma (entre 1 y -1). Si una función es contractante en un conjunto compacto (un intervalo cerrado en R, por ejemplo), entonces tiene un “punto fijo” tal que $x=f(x)$. Más aún, la sucesión $s(n+1)=f(s(n))$ converge a $s=f(s)$, el punto fijo, dentro del compacto. Por ejemplo, el punto fijo

1 Se está haciendo alusión al *Teorema del punto fijo de Banach*, que deberían ver en el curso de cálculo en varias variables.

de la función seno es 0 ($\sin(0) = 0$). ¿Cuál es el punto fijo del coseno? Defina la función recursiva $F(x)$ que llame a $F(\cos(x))$ si $|x-\cos(x)|>0,001$ y que devuelva x si no.

4* Aproxime la función exponencial a través de una serie de Mc Lauren (expansión de Taylor en torno a cero). Use los primeros 50 términos de la sucesión:

$$\sum_{n=0}^{\infty} \frac{x^n}{n!}$$

O sea, defínase la función expo como sigue:

$$\text{expo}(x) = \sum_{n=0}^{49} \frac{x^n}{n!}$$

¿Cómo aproxima? Comparar con $\text{exp}(.5)$ y $\text{exp}(10)$. Nota: use *for* para realizar la sumatoria.

5* Aproxime la función exponencial a través de la definición de sucesión, así:

$$\text{expo}(x, n) = \left(1 + \frac{x}{n}\right)^n \approx \text{exp}(x), n \gg 0$$

¿Cómo aproxima? Comparar el término 100, 10000 y 1000000 de la sucesión para aproximar $\text{exp}(10)$.

6* La aproximación de Stirling para un número factorial, usada ampliamente en Física Estadística y Astronomía, dice que:

$$\log(n!) \approx n \log(n) - n$$

La función factorial $n!$ se escribe como *gamma(n)*, pues gamma es la función factorial (sin embargo, es continua). *log* evalúa el logaritmo natural. Cree una función *error_stirling(n)* que indique el error absoluto de la aproximación de Stirling para un n dado. Grafique el error entre 1 y 100.

Soluciones

1>

```
function s=seno(x)
```

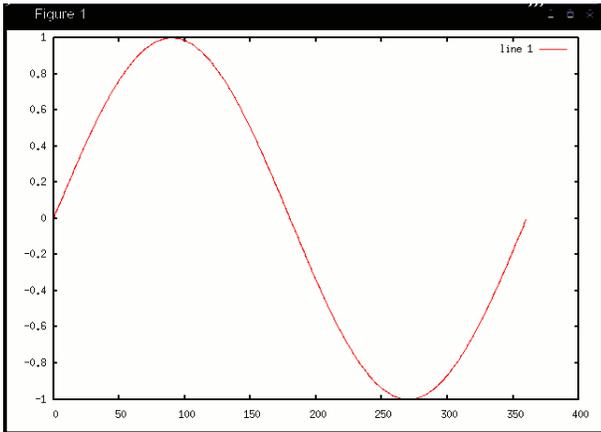
```
    s=sin( x*pi/180 );
```

```
endfunction
```

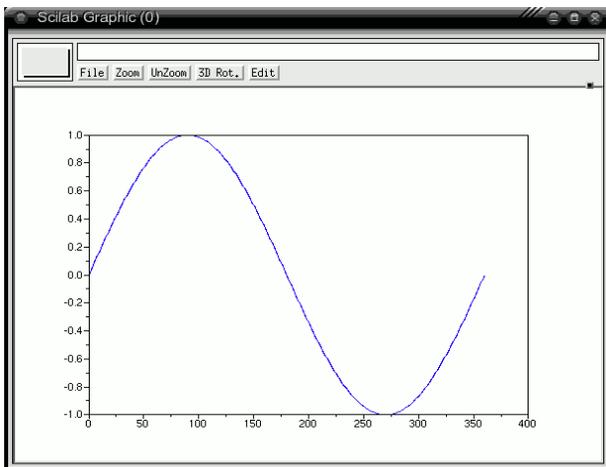
```
x=linspace(0,360,360);
```

```
plot( x , seno(x) );
```

Observación: *sin* evalúa coordenada a coordenada el vector *x*, así que esta función no garantiza problemas de evaluación. **Pero esto no significa que siempre ocurra automáticamente** (vea el siguiente ejercicio).



(El gráfico de la función seno sexagesimal. Generado con Octave.)



(El gráfico de la función seno sexagesimal. Generado con Scilab.)

2>

```
function f=fibo(n)
  if (n<2) f=1; else f=fibo(n-1)+fibo(n-2); endif
endfunction
```

```
x=0:1:19;
```

```
for(i=0:19) y(i+1)=fibo(i); endfor
```

```
plot(x, y);
```

Observación: evaluar `fibo(0:1:19)` es un error. La función no está diseñada para recibir un vector, sólo debe recibir un escalar. Por eso se evalúa con un `for` para poder dibujarla.

Observación: note `y(i+1)=fibo(i);`. Como `y` no es una función definida, se asume como un vector (recibe sólo un índice). Note además que **el vector no está indexado como en Java; las coordenadas del vector van de 1 a *n*, no de 0 a *n*-1 como en los arreglos de Java.**

3>

```
function s=F(x)
```

```
  if (abs( x-cos(x) )<0.001) s=x;
```

```
  else s=F( cos(x) ); endif
```

```
endfunction
```

Ahora el punto fijo se puede encontrar como $F(0)$, $F(1)$, $F(100)$... El punto fijo del coseno es único, y se nota con esta función. (Es algo inferior a 0,74.)

Conversión a Java

1> Traduzca la función de Fibonacci a Java:

```
function f=fibo(n)
```

```
  if (n<2)
```

```
    f=1;
```

```
  else
```

```
    f=fibo(n-1)+fibo(n-2);
```

```
  endif
```

```
endfunction
```

2> Traduzca la función del punto fijo del coseno a Java. Asuma `x` como escalar.

```
function s=F(x)
```

```
  if (abs( x-cos(x) )<0.001) s=x;
```

```
else s=F( cos(x) ); endif
```

```
endfunction
```

3> Traduzca la siguiente función a Java. Note que x es un vector.

```
function s=acum(x)
```

```
    s(1)=x(1);
```

```
    for(i=2:length(x))
```

```
        s(i)=s(i-1)+x(i);
```

```
    endfor
```

```
endfunction
```

Soluciones

1> Usar int es optativo. Perfectamente se pudo usar double o float.

```
static int fibo(int n) {
```

```
    int f;
```

```
    if (n<2)
```

```
        f=1;
```

```
    else
```

```
        f=fibo(n-1)+fibo(n-2);
```

```
    return f;
```

```
}
```

2> Aquí se **debe** usar double o float para los tipos.

```
static double F(double x) {
```

```
    double s;
```

```
    if ( Math.abs( x - Math.cos(x) ) < 0.001 )
```

```
        s=x;
```

```
    else
```

```
        s=F( Math.cos(x) );
```

```
    return s;
```

```
}
```

3> Se devuelve un vector. En Java devolveremos un arreglo.

```
static double[] acum(double[] x) {
```

```
    double[] s=new double[ x.length ];
```

```
    s[0]=x[0];
```

```
    for(int i=1;i<x.length;i++) {
```

```
        s[i]=s[i-1]+x[i];
```

```
    }
```

```
    return s;
```

```
}
```

Ejemplo de aplicación

Usted es dueño de una pizzería y usará Matlab para analizar su producción y optimizarla de manera de volverse lo más rico que pueda.

1> Cree la función **Gente(D)**. Esta función entrega el número de trabajadores necesarios para satisfacer una cierta demanda diaria de pizzas. Suponga que para hacer 100 pizzas en un día necesita 5 trabajadores. Use esto como referencia para definir la función lineal.

2> Usando estadística, usted ha descubierto que la función de demanda diaria **D(P)** es una función del precio tal que $D(P)=1000000/(1+P^2)$. Esta función es decreciente ya que, a más cara la pizza, menos gente la compra. Escriba la función **D(P)**.

3> Usted es dueño de elegir el precio al que venderá las pizzas. Si las vende caras, nadie las comprará y si las vende muy baratas, no ganará nada. ¿Qué hacer? Comencemos por definir el ingreso, o sea, la plata que entra. Defina **Ingreso(P)** como el precio de las pizzas multiplicado por el número de pizzas vendidas a ese precio ($P \cdot D(P)$).

4> Suponga que paga 1000 a los trabajadores y que cada pizza cuesta, en materiales, 5. Defina **Costo(P)** como el sueldo de sus trabajadores ($1000 \cdot \text{Gente}(D(P))$) más el costo de las pizzas vendidas ($5 \cdot D(P)$).

5> Defina la utilidad como los ingresos excedentes sobre los costos, o sea, **Utilidad(P) = Ingreso(P) - Costo(P)**.

6> Grafique Utilidad(P) usando P de 50 a 350, con paso=1. ¿Cuánto hay que cobrar por pizza para

ganar lo más posible?

7* Use una nueva función de demanda:

$$D(P) = \frac{1000000}{\exp(2 + P/15)}$$

¿Cómo cambia su utilidad?

8* Utilice una nueva función de costos tal que:

- Si la demanda es mayor a 100 pizzas/día, hay un sobrecosto eléctrico de 50.
- Si la demanda es mayor a 1000 pizzas/día, agregar un costo fijo de 1000 de arriendo de otro local.

9* Programe un método de búsqueda para encontrar el precio que maximice la utilidad. Si la utilidad es decreciente, disminuir el precio. Si la utilidad es creciente, aumentarlo. Detenerse cuando la utilidad en cierto precio es mayor a la utilidad de los precios vecinos ($p \pm 1$).

10* Modifique las funciones anteriores para agregar el efecto de la publicidad: por cada 1000 invertidos en publicidad, la demanda se duplica hasta la inversión de 10000. De ahí la demanda no aumenta más. ¿Cuál es la utilidad máxima?

Solución

```
function g=Gente(D)
```

```
g=D/20;
```

```
end
```

```
function d=D(P)
```

```
d=1000000/(1+P*P);
```

```
end
```

```
function I=Ingreso(P)
```

```
I=P*D(P);
```

```
end
```

```
function C=Costo(P)
```

```
C=1000*Gente( D(P) )+5*D(P);
```

```
end
```

```
function U=Utilidad(P)
```

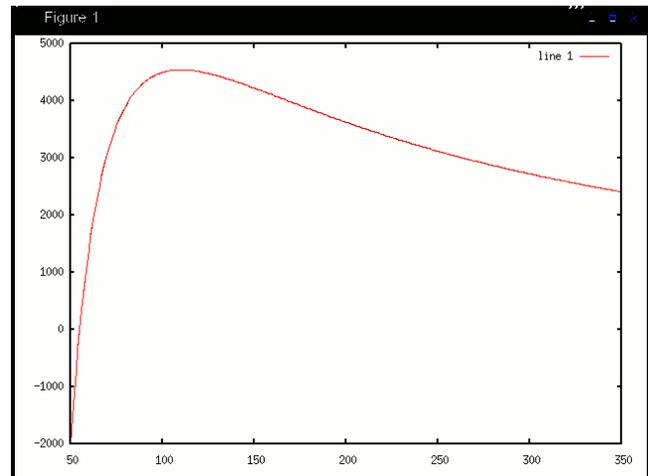
```
U=Ingreso(P)-Costo(P);
```

```
end
```

```
y=[];
```

```
for(z=50:350) y(z-49)=Utilidad(z); endfor
```

```
plot( 50:350 , y);
```



Del gráfico generado, es posible ver que la utilidad es máxima cuando el precio cobrado es 110 (aproximadamente). En ese caso, la utilidad bordea los 4500.

Guía rápida de funciones

abs	Valor absoluto.
cos, sin, tan	Coseno, seno y tangente.
acos, asin, atan	Arcocoseno, arcoseno y arcotangente.
gamma	Factorial.
exp, log	Exponencial y logaritmo natural.
x^n	Potencia: x a la n. Si n=0.5, raíz cuadrada.
linspace(a,b,n)	Intervalo [a,b], en n puntos.
for(i=a:b)	For de a a b a pasos de 1.
endfor	Cierre del for.
while(condición)	While. Cualquier condición lógica sirve.
endwhile	Cierre del while
if (condición)	If. Cualquier condición lógica sirve.
else	Bloque else.
endif	Cierre del if.
plot (x, y)	Dibuja segmentos de recta con los x y los y.