Cálculo Simbólico con MATLAB. Introducción a la Programación.

Bloque III: Funciones. Límites. Resolución de ecuaciones. Resolución de sistemas de ecuaciones. Derivadas. Integración.

Cristina Solares

Universidad de Castilla-La Mancha

9 de septiembre de 2022

MATLAB proporciona cientos de funciones propias.

Funciones trigonométricas:

- `acos(x)`: Angle in radians whose cosine equals x
- `acot(x)`: Angle in radians whose cotangent equals x
- `asin(x)`: Angle in radians whose sine equals x
- `atan(x)`: Angle in radians whose tangent equals x
- `atan2(y,x)`: Four-quadrant angle in radians whose tangent equals y/x
- `cos(x)`: Cosine of x (x in radians)
- `cot(x)` : Cotangent of x (x in radians)
- `sin(x)`: Sine of x (x in radians)
- `tan(x)`: Tangent of x (x in radians)

Funciones exponenciales:

- `exp(x)`: e raised to the x power
- `log(x)`: Natural logarithm x
- `log2(x)`: Base-2 logarithm of x
- `log10(x)`: Base-10 logarithm of x
- `sqrt(x)`: Square root of x

Funciones que trabajan con números complejos:

- `abs(z)`: Absolute value of z
- `angle(z)`: Phase angle of z
- `conj(z)`: Complex conjugate of z
- `imag(z)`: Imaginary part of z
- `real(z)`: Real part of z

Funciones de redondeo, resto y signo:

- `fix(x)`: Round x towards zero
- `floor(x)`: Round x towards minus infinity
- `ceil(x)`: Round x towards plus infinity
- `round(x)`: Round x towards nearest integer
- `rem(x,n)`: Remainder of x/n (see help for case of noninteger n)
- `sign(x)`: 1 if $x > 0$; 0 if x equals 0; -1 if $x < 0$

Definición de Funciones

En MATLAB podemos definir nuestras propias funciones como funciones anónimas, escribiendo la función en un fichero externo con la extensión `.m` o definiendo la función como una expresión simbólica.
Podemos definir nuestra propia **función anónima** como sigue

```
function_name = @(list_of_argument_names)(one_line_of_code)
```

Una vez que hemos definido una función anónima, puede utilizarse escribiendo su nombre y un valor para el argumento entre paréntesis.
A continuación definimos la función $h(x) = \sin x + \cos x$

```
>> h=@(x) sin(x)+cos(x)
h =
  function_handle with value:
    @(x)sin(x)+cos(x)
>> h(pi) %valor de h en pi
ans =
   -1.0000
>> syms a; h(a) %valor de h en a
ans =
cos(a) + sin(a)
```

Se puede comprobar que MATLAB no ha creado una variable `x` cuando se define la función `h(x)`. Si queremos calcular un límite, integrar o derivar la función `h(x)`, tenemos que declarar `x` como una variable simbólica. Lo mismo ocurre si queremos dibujar la función de forma simbólica. En los casos anteriores, hacemos `>> syms x`. Para dibujar la función de forma simbólica, utilizamos el comando `fplot` de dos formas

```
>> fplot(h,[-pi,pi])
```

o

```
>> syms x
>> fplot(h(x),[-pi,pi])
```

Si trabajamos de forma numérica, utilizamos el comando `plot`

```
>> x=-pi:pi/10:pi;
>> y=h(x);
>> plot(x,y)
```

Se puede definir una **función anónima de dos variables** `g(x,y)` haciendo

```
>> g=@(x,y) x^2+y^2
g =
  function_handle with value:
    @(x,y)x^2+y^2
>> g(2,3)
ans =
    13
>> syms a b
>> g(a,b)
ans =
a^2 + b^2
```

Definición de Funciones

Se puede definir una **función simbólica de una o varias variables**

```
>> syms x
>> syms h(x)
>> h(x)=sin(x)*cos(x);
>> h(pi/4)
ans =
1/2
>> syms a
>> h(a)
ans =
cos(a)*sin(a)
```

y se puede definir una **función como una expresión simbólica**. Para evaluar la función en un punto se utiliza el comando `subs`

```
>> syms x
>> h=sin(x)*cos(x);
>> subs(h,x,pi/4)
ans =
1/2
```

Para dibujar una función o expresión simbólica se utiliza el comando `fplot`, en el primer caso

```
>> fplot(h(x),[-2*pi,2*pi])
```

y en el segundo caso

```
>> fplot(h,[-2*pi,2*pi])
```

La toolbox Symbolic Math Toolbox dispone de comandos para calcular **límites de funciones**. Se define la función `f(x)` y posteriormente se calcula el límite mediante el comando `limit(f(x),x,a)`. En el siguiente ejemplo calculamos el límite de la función $f(x) = 5x^2 + x - 7$ cuando x tiende a -2 y a $-\infty$.

```
>> f=@(x) 5*x^2+x-7;
f =
  function_handle with value:
    @(x)5*x^2+x-7
>> syms x
>> lim=limit(f(x),x,-2)
lim =
11
>> limit(f(x),x,-Inf)
ans =
Inf
```

También podemos definir la función $f(x)$ de forma simbólica y calcular su límite

```
>> syms x
>> f(x)= 5*x^2+x-7;
>> limit(f(x),x,-2)
```

Utilizando 'right' y 'left' indicamos si el límite se calcula por la derecha o izquierda.

```
>> g=@(x) 1/(x-1);
>> syms x
>> limit(g(x),x,1,'right')
ans =
Inf
>> limit(g(x),x,1,'left')
ans =
-Inf
```

También podemos calcular límites de expresiones sin definir una función (pueden incluir símbolos)

```
>> syms x a
>> limit((a*x^2-1)/(x^2+1),x,1)
ans =
a/2 - 1/2
```

Se puede definir una variable que sea igual a la expresión anterior, para calcular el límite

```
>> syms x a
>> f=(a*x^2-1)/(x^2+1);
>> limit(f,x,1)
ans =
a/2 - 1/2
```

El comando `solve(eqn,var)` **resuelve la ecuación** `eqn` en la variable `var`

```
>>syms x a
>> sol1=solve(2*x^2+a*x+1,x) %resolvemos en la variable x
sol1 =
 - a/4 - (a^2 - 8)^(1/2)/4
   (a^2 - 8)^(1/2)/4 - a/4
```

Si queremos la solución particular en el caso `a=0`, debemos utilizar el comando `subs`

```
>> subs(sol1,a,0) %Sustituimos a por el valor 0
ans =
 -(8^(1/2)*1i)/4
  (8^(1/2)*1i)/4
```

En el ejemplo anterior el comando `solve` devuelve dos soluciones, para acceder a cada solución hacemos

```
>> sol1(1) %Obtenemos la primera
ans =
- a/4 - (a^2 - 8)^(1/2)/4
>> sol1(2) %Obtenemos la segunda
ans =
(a^2 - 8)^(1/2)/4 - a/4
```

También se puede utilizar el comando `solve(f(x)-g(x),x)` para resolver la ecuación `f(x)=g(x)` en la variable `x`

```
>>syms x
>> solve(sin(x)-(1/2),x)
ans =
     pi/6
 (5*pi)/6
```

El comando `solve(f(x,y)-g(x,y),h(x,y)-w(x,y),x,y)` **resuelve el sistema de ecuaciones** {`f(x,y)=g(x,y), h(x,y)=w(x,y)`} en las variables `x` e `y`.

```
>>syms x y
>> [x1,y1]=solve(x+sqrt(y)-1,(x+y)^2-y,x,y)
x1 =
      0
 1 - 1i
y1 =
  1
 -1
```

MATLAB devuelve la solución exacta del sistema de ecuaciones. El valor aproximado se puede obtener con el comando `double`

```
>>syms x y
>> [x1,y1]=solve(2*x+y-1,3*x+2*y-(7/5),x,y)
x1 =
3/5
 y1 =
-1/5
>> double(x1)
ans =
    0.6000
```

Si no incluimos `[x1,y1]`, el programa devuelve

```
>>syms x y
>> solve(2*x+y-1,3*x+2*y-(7/5),x,y)
ans =
    x: [1x1 sym]
    y: [1x1 sym]
```

La solución se puede visualizar haciendo `ans.x` y `ans.y`. También se pueden escribir las ecuaciones como sigue

```
>>syms x y
>> [x1,y1]=solve(2*x+y==1,3*x+2*y==(7/5),x,y)
```

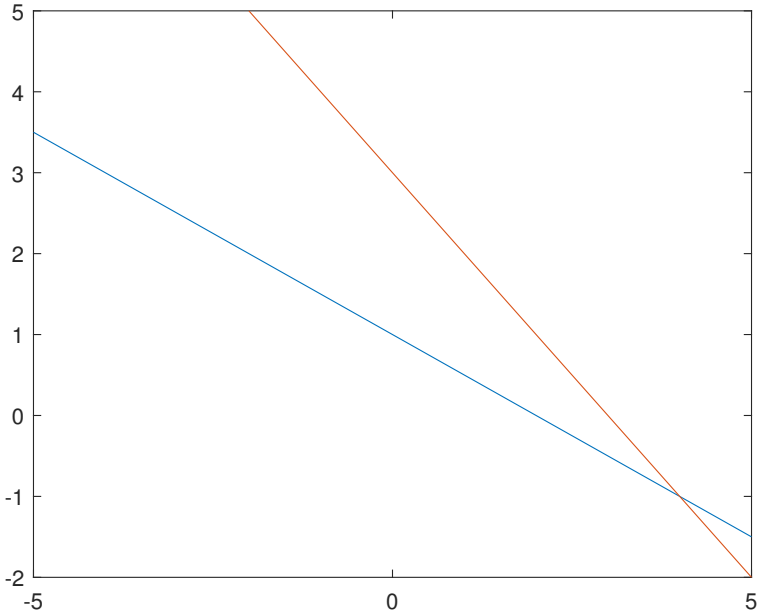
Resolución de un Sistema de Ecuaciones Lineales

Un **sistema se ecuaciones lineales** se puede representar en forma matricial Ax = b. Si A es una matriz cuadrada y no singular, la solución del sistema puede obtenerse mediante la operación A\b.

```
>> A=[3 2; -1 2]; b=[18;2];
>>  det(A)
ans =
     8
>> sol=A\b
sol =
     4
     3
>> A*sol
ans =
    18
     2
```

Resolución de un Sistema de Ecuaciones Lineales

También se puede resolver un sistema de ecuaciones lineales $Ax = b$ utilizando el comando `linsolve(A,b)`.




```
>> A=[1 2; 2 2];
>> b=[2;6];
>> linsolve(A,b)
ans =
     4
    -1
```

Si definimos una función `f(x)` anónima, el comando `diff(f(x),x)` **calcula la derivada de** $f(x)$ respecto a la variable x, previamente declarada como simbólica. Para calcular la derivada n-ésima se utiliza `diff(f(x),x, n)`. También podemos calcular la derivada de una función definida mediante una función o expresión simbólica. En el siguiente ejemplo se calculan las derivadas primera y segunda de la función $f(x) = x^2 + 4x - 1$ respecto de la variable x.

```
>> syms x; f=x^2+4*x-1;
>> df=diff(f,x)
df =
2*x + 4
>> diff(f,x,2)
ans =
2
```

Con el comando `fplot` podemos dibujar la función $f(x)$ y la función primera derivada $\frac{df}{dx}$ en un mismo gráfico, el intervalo de dibujo por defecto es $[-5, 5]$

```
>> fplot(f)
>> hold on
>> title('Grafica de f y su derivada')
>> fplot(df)
```

Se puede incluir directamente en el comando `diff` la función a derivar

```
>> syms y a
>> diff(sin(y)*cos(a),y)
ans =
cos(a)*cos(y)
```

Si definimos una función `f(x)` anónima, el comando `int(f(x),x)` calcula la **integral indefinida** de $f(x)$ respecto a la variable x, previamente declarada como simbólica.
Para calcular una **integral definida** $\int_a^b f(x)\,dx$ se escribe `int(f(x),x, a,b)`. También podemos calcular la integral de una función definida mediante una expresión o función simbólica.

```
>> syms x
>> integral=int(1/sqrt(1+x+x^2),x)
integral =
log(x + (x^2 + x + 1)^(1/2) + 1/2)
>> int(1/sqrt(1+x+x^2),x,0,1)
ans =
-log(2*3^(1/2) - 3)
>> f=cos(x)^2/sin(x);
>> int(f,x)
ans =
log(tan(x/2)) + cos(x)
```