Programación en lenguaje MATLAB

Clase 4

Dr. Ing. Rodrigo Gonzalez
rodralez@frm.utn.edu.ar

Universidad Tecnológica Nacional,
Facultad Regional Mendoza.

Resumen

Funciones anónimas (FA)

- MATLAB permite declarar funciones sin la necesidad de crear un archivo .m.
- Se declara como:
  `nombre_fun = @ (ia1,ia2,...)  ( expresion )`,
  tanto en un programa como en consola.
- Las FA suelen usarse para declarar funciones cortas.
  Ej: **`parabola = @ (x,a,b,c) (a*x^2+b*x+c)`**

Ejercicio 1

Convierta la función `int_compuesto` en una función anónima. Verifique su correcto funcionamiento.

```
function Cf = int_compuesto (Ci, int, n)
```

```
nombre_fun = @ (ia1,ia2,...)   ( expresion )
```

Respuesta

```
» int_compuesto = @ (Ci,int,n) (Ci.*(1 + int).^n)
» Cf = int_compuesto(cap_ini, interes, periodo);
```

Manejador de funciones (MF, function handle)

- A veces una función necesita recibir una función como argumento.
- Un MF es un tipo de dato en MATLAB y puede ser pasado como argumento a una función.
- Se declara como:

  `fun_handle = @ fun_nombre`

- Se puede usar con funciones estándares o creadas por el usuario.

  Ej: `sin_h = @ sin`

- En el caso de FA, su nombre se considera un MF.

  `nombre_fun = @ (ia1,ia2,...)  expresion`

- En otros lenguajes, un MF se conoce como *puntero a función*.

Manejador de funciones

Ejercicio 2

Cree la función `graficar` que grafique cualquier tipo de función en el intervalo `v = [a b]`. Verifique su correcto funcionamiento con las funciones seno y coseno. Para graficar use las funciones `figure` y `plot()`.
Prototipo de la función: **`function graficar (fcn, v)`**

Respuesta 1/2

```
function graficar (fcn, v)
  t = v(1):0.01:v(2);
  y = fcn(t);
  figure;
  plot(t,y);
end
```

Manejador de funciones

Ejercicio 2

Cree la función `graficar` que grafique cualquier tipo de función en el intervalo `v = [a b]`. Verifique su correcto funcionamiento con las funciones seno y coseno. Para graficar use las funciones `figure` y `plot()`.

Prototipo de la función: **`function graficar (fcn, v)`**

Respuesta 2/2

```
» fsin = @ sin; fcos = @ cos;
» a = 0; b = 2*pi;
» graficar (fsin, [a b]);
» graficar (fcos, [a b]);
```

Declaración de variables simbólicas

Las variables simbólicas se definen con los comandos `sym` y `syms`

- `obj = sym ('nombre' )`
- `obj = sym (4) % Numero simbólico`
- `syms a b % Con syms es posible definir números`
- `syms c d real % Tags modifican rango`
- `syms var1 var2 positive`

Ejemplo

```
1 » syms x y b real
2 » a = sym(1/3) % Número racional exacto
3 » M = sym([1 2; 3 4;])
4 » c = sym('c' , 'positive' ) % Tags modifican
  rango
5 » d = a * b
6 » Ms = inv(M)
7 » e = a * b + c
```

Operación

- Cuando se opera con variables simbólicas el resultado es exacto.
- Además, el resultado es otra variable simbólica.
- Las variables simbólicas se muestran en consola sin tabular.

Ejemplo

```
» syms x real
» a = x/3
» b = x/4
» c = a + b % El resultado es 7x/12, no 0.583x
» d = a * b
» e = sym(1/3)
» f = sym(4)
» g = e + f
» h = 1/3 + 4 % Comparar con el valor de g
```

Declaración de expresiones simbólicas

- Son expresiones matemáticas definidas a partir de variables simbólicas.
- Si una expresión aproximada es parte de una expresión simbólica, se fuerza su representación simbólica.

Ejemplo

```
1 » syms a b c x y real
2 » f = a*x^2 + b*x + c
3 » S1 = (26*a)/21 - (37*x)/6 + 31/6
4 » S2 = (26*a)/21 - (37*x)/6 + 5.1666 % ¿Qué observa?
```

Funciones para manejo de variables simbólicas

- `findsym(S)`, muestra variables simbólicas en S.
- `collect(S)`, agrupa variables simbólicas de mayor a menor potencia.
- `expand(S)`, expande expresión simbólica en sumas.
- `factor(S)`, contrae expresión en productos.
- `simplify(S)`, simplifica expresión según reglas de simplificación.

Ejemplo

```
» syms x y a b c
» f = a*x^2 + b*x + c, findsym(f)
» e1 = expand((x-2)*(x-4))
» e2 = expand(cos(x+y))
» c1 = collect((x+y)*(x^2+y^2+1))
» f1 = factor([a^2 - b^2, a^3 + b^3])
» f2 = factor(sym('1024' ))
» s1 = simplify(sin(x)^2 + cos(x)^2)
» S = [(x^2 + 5*x + 6)/(x + 2), sqrt(16)]
» s2 = simplify(S)
```

Funciones para manejo de variables simbólicas (2)

- `solve(S,var)`, resuelve (despeja) expresión respecto a una variable.
- `solve(S1,S2,S3...Sn)`, resuelve sistema de ecuaciones.
- `subs(S,var,valor)`, sustituye en expresión variable por número.
- `double(var)`, calcula el valor aproximado de una variable exacta.

Ejemplo

```
syms a b c x y z
so1 = solve(x^2 - 1)
so2 = solve(a*x^2 + b*x + c == 0, a)
so3 = solve(a*x^2 + b*x + c == 0, b)
so4 = solve(4*x - 2*y + 6*z == 8, ...
            2*x + 8*y + 2*z == 4, ...
            6*x + 10*y + 3*z == 0)
double(so4.x), double(so4.y), double(so4.z)
su1 = subs(x + y, y, a)
su2 = subs(sin(x + 1) + 1 == x, sin(x + 1), a)
su3 = subs(sin(a) + cos(b), [a, b], [x + y, x - y])
su4 = subs(sin(a) + cos(b), [a, b], [pi/2 pi/3])
```

Ejercicio 3

La ecuación que describe un círculo de radio r, centrado en el punto (a, b) es

$$(x - a)^2 + (y - b)^2 = r^2$$

Resuelva usando `solve()` para `x` e `y`.

Respuesta

```
syms a b r x y
solve((x-a)^2+(y-b)^2 == r^2, x)
solve((x-a)^2+(y-b)^2 == r^2, y)
```

Ejercicio 4

La ecuación que describe un círculo de radio r en el plano x-y está dada por

$$(x-2)^2 + (y-4)^2 = r^2$$

.

La ecuación de una recta en el plano está dada por

$$y = \frac{x}{2} + 1$$

.

Encuentre los puntos donde la recta intersecta a la circunferencia para $r = 10$. Resuelva usando `solve()` y `subs()` para las coordenadas en `x` e `y`.

Respuesta

```
syms x y R
[px, py] = solve( (x-2)^2+(y-4)^2 == R^2, y == x/2+1)
subs(px,R,10)
subs(py,R,10)
```

Declaración de polinomios

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \cdots + a_1 x + a_0$$

- a_n son números reales,
- n es entero no negativo, orden del polinomio.
- Un polinomio se representa con un vector fila.

Ejemplos:

Polimonio	**En MATLAB**
$8x + 5$	[8 5]
$2x^2 - 4x + 10$	[2 -4 10]
$5x^5 + 6x^2 - 7x$	[5 0 0 6 -7 0]

Resolución de polinomios

- `polyval(p,x)`, evalua un polinonio en x.
- x puede ser un valor, un vector o una matriz.

Ejercicio 5

Para el polinomio $f(x) = x^5 - 12.1x^4 + 40.59x^3 - 17.015x^2 - 71.95x + 35.88$

1. Calcule $f(9)$.
2. Luego, grafique $f(x)$ para $-1.5 \leq x \leq 6.7$.

Respuesta

1. `» p = [1 -12.1 40.59 -17.015 -71.95 35.88];`
2. `» polyval(p,9)`
3. `» x = -1.5:0.01:6.7;`
4. `» y = polyval(p,x);`
5. `» plot(x,y)`

Raíces de un polinomio

$$f(x) = (x - r_n)(x - r_{n-1})(x - r_{n-2}) \cdots (x - r_1)$$

- Las raíces de un polinomio son los valores de las variables que hacen que el polinomio valga cero.
- `roots(p)`, calcula las raíces de un polinomio.

Ejercicio 6

Encuentre las raíces del polinomio
$f(x) = x^5 - 12.1x^4 + 40.59x^3 - 17.015x^2 - 71.95x + 35.88$

Respuesta

1. `» p = [1 -12.1 40.59 -17.015 -71.95 35.88];`
2. `» r = roots(p)`

Funciones que operan con polinomios

- Suma y resta, mismas reglas que suma y resta de vectores.
- `c = conv(p1,p2)`, multiplicación de polinomios.
- `[q,r] = deconv(p1,p2)`, división de polinomios.
- `k = polyder(p)`, derivada.
- `k = polyder(p1,p2)`, multiplica los polinomios y deriva.
- `[q,r] = polyder(p1,p2)`, divide los polinomios y deriva, donde $\frac{q(x)}{r(x)} = \frac{d}{dx}\left[\frac{p1(x)}{p2(x)}\right]$.

Ejercicio 7

Calcule la derivada del polinomio $f_1(x) = 3x^2 - 2x + 4$. Luego, siendo $f_2(x) = x^2 + 5$, calcule la derivada de $f_1(x) * f_2(x)$ y de $f_1(x)/f_2(x)$.

Respuesta

1. `f1 = [3 -2 4]; f2 = [1 0 5];`
2. `k1 = polyder(f1)`
3. `k2 = polyder(f1,f2)`
4. `[q,r] = polyder(f1,f2)`

Integración simbólica

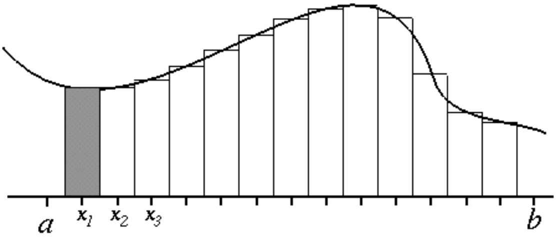
- `int(S)`, integral indefinida respecto a la variable por defecto.
- `int(S,var)`, integral indefinida respecto a una variable determinada.
- `int(S,a,b)`, integral definida en el intervalo [a,b].
- `int(S,var,a,b)`, integral definida en el intervalo [a,b] respecto a una variable determinada.

Ejemplos

```
1 » syms x y t
2 » S = 2*cos(x)-6*x;
3 » int(S)
4 » R = 5*y^2*cos(3*t);
5 » int(R)
6 » int(R,t)
7 » int(sin(y)-5*y^2,0,pi)
```

Resolución numérica de integrales

$$q = \int_a^b f(x)\, dx \approx \sum_{i=1}^{n} f(x_i)\, dx_i$$



- Se resuelve $f(x)$ a partir de ecuaciones o valores discretos.
- MATLAB ofrece diferentes métodos para el calcular integrales.
- Cada método representa la superficie bajo la curva de forma diferente.
- Se utilizan diferentes funciones según el caso que $f(x)$ sea una ecuación o vector.

Integración de expresiones matemáticas

- `i = integral(fd,a,b)`, método de cuadratura adaptativa global $\Diamond$.
- `i = quad(fd,a,b)`, método de Simpson $\Diamond$.
- `i = quadl(fd,a,b)`, método de Lobatto.
- `i = quadgk(fd,a,b)`, método de Gauss-Kronrod.

  `a` y `b` pueden valer `-Inf` y/o `Inf`.
- `fd` es un manejador de funciones (function handle).
- $f(x)$ debe ser escrita para operar elemento a elemento.

Ejercicio 8

Calcule $S = \int_0^8 (x\, e^{-x^{0.8}} + 0.2)\, dx$ por los cuatro métodos mencionados.

Respuesta

```
1 » S = @(x) (x .* exp(-x.^(0.8)) + 0.2)
2 » I1 = integral(S,0,8) , I2 = quad(S,0,8)
3 » I3 = quadl(S,0,8) , I4 = quadgk(S,0,8)
4 » d1 = abs(I1-I2), d2 = abs(I3-I4)
```

Integración discreta

- `i = sum(v) .* dt`
- `i = trapz(v) .* dt`, integrada numérica trapezoidal.
- `dt` es el paso entre elementos de `v`.

Ejercicio 9

Resuelva $S = (x\, e^{-x^{0.8}} + 0.2)$ para $0 \leq x \leq 8$ con pasos de 0.01 usando la función anónima del ejercicio anterior. Luego integre la curva resultante con `trapz()`. Compare con los resultados del ejercicio anterior.

Respuesta

```
1  » S = @(x) (x .* exp(-x.^(0.8)) + 0.2)
2  » dt = 0.01;
3  » t = 0:dt:8;
4  » y = S(t);
5  » I5 = trapz(y) .* dt
```

Derivación simbólica

- `diff(S)`, derivada respecto a la variable por defecto.
- `diff(S,var)`, derivada respecto a una variable determinada.
- `diff(S,var,n)`, *n* indica el orden de la derivada.

Ejemplos

```
» syms x y t
» S = exp(x^4);
» d1 = diff(S)
» d2 = diff((1-4*x)^3)
» R = 5*y^2*cos(3*t);
» d3 = diff(R,t)
» d4 = diff(S,2)
% Derivada del producto
» f = sin(x); g = 2*x^2 + 3*x + 1;
» y = f*g; d5 = diff(y)
% Regla de la cadena
» u = 2*x+3; f = sin(u); d6 = diff(f)
```

Derivación de expresiones matemáticas

Se obtiene combinando las funciones `d=diff(S)` y `subs(d,v)`.

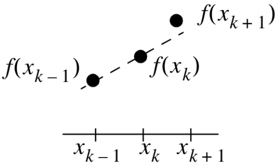
Ejemplos

1. `» syms x y`
2. `» y = sin(x); t1 = 0:0.01:2*pi;`
3. `» d1 = diff(y,'x' ); da1 = subs(d1, t1)`
4. `» y = exp(2*x); t2 = 0:0.01:1;`
5. `» d2 = diff(y); da2 = subs(d2, t2)`

- Si `y` es un vector, `d = diff(y)` devuelve un vector con la resta entre elementos contiguos.
- `d` contiene 1 elemento menos que `y`.
- `x` e `y` deben tener igual dimensión.

Backward Difference

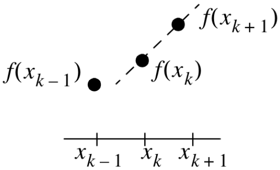
$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$



`dydx = diff(y) ./ diff(x)` para `x(2:end)`

Forward Difference

$$f\,(x_k) \approx \frac{f(x_{k+1}) - f(x_k)}{x_{k+1} - x_k}$$

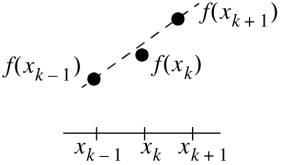


`dydx = diff(y) ./ diff(x)` para `x(1:end-1)`

Derivación discreta centrada

Central Difference

$$f'(x_k) \approx \frac{f(x_{k+1}) - f(x_{k-1})}{x_{k+1} - x_{k-1}}$$



`dydx_par = diff(y(1:2:end)) ./ diff(x(1:2:end))`
para `x(2:2:end)`

`dydx_impar = diff(y(2:2:end)) ./ diff(x(2:2:end))`
para `x(3:2:end)`

Derivación discreta centrada (2)

Implementación en MATLAB

```
function dcen = diff_central (x, y)
[n,m] = size(x);
% Derivada de elementos pares
dcen_par = diff(y(1:2:end)) ./ diff(x(1:2:end));
% Derivada de elementos impares
dcen_impar = diff(y(2:2:end)) ./ diff(x(2:2:end));
dcen = zeros(size(x));
if( mod(n,2) == 0) % Si n es par...
 dcen (2:2:end-2) = dcen_par;
 dcen (3:2:end) = dcen_impar;
else%  Si n es impar...
 dcen (2:2:end) = dcen_par;
 dcen (3:2:end-2) = dcen_impar;
end
% Se pierden 2 elementos, el primero y el ultimo
dcen = dcen (2:end-1);
end
```

Derivación discreta

Ejercicio 10

Encuentre el valor de la derivada $f(x) = 5\cos(10x) + x^3 - 2x^2 - 6x + 10$ para $0 \leq x \leq 4$ para los 3 métodos. Grafique.

Respuesta

```
x = 0:0.01:4;
y = 5*cos(10*x) + x.^3 - 2*x.^2 - 6*x + 10;
dback = diff(y) ./ diff(x);
dfor = diff(y) ./ diff(x);
dcen = diff_central (x, y);

plot( x(2:end), dback, 'b' ), hold on, ...
plot( x(1:end-1), dfor, 'r' )
plot( x(2:end-1), dcen, 'g' )
legend('Backward' , 'Forward' , 'Central' )
```

Introducción a EDO

$$y' = \frac{dy}{dt} = f(t, y), \quad y_0 = f(t_0)$$

- La solución es una función del tipo $y = f(t)$.
- t, variable independiente (tiempo).
- y, variable dependiente.

MATLAB cuenta con varios métodos númericos para resolver EDOs:

- Una EDO es *rígida* cuando su solución varía fuertemente en un intervalo de muestreo.
- Para EDOs no rígidas se usa el solver `ode45`.
- Para EDOs rígidas se usa el solver `ode15s`.
- Hay más *solvers*: `ode23`, `ode113`, `ode23s`, `ode23t` y `ode23tb`.

Uso de solvers

```
[T,Y] = solver(fh,t,y0)
```

- `solver` puede ser `ode45`, `ode15s`, etc.
- `fh`, manejador de función (*function handle*).
- `t`, vector que especifica el intervalo $[t_0 \; t_f]$.
- `y0`, condición inicial.

$$\frac{dy}{dt} = \frac{t^2 - 2\,y}{t}$$

Ejemplo

```
% Function handle
my_ode = @(t,y)((t.^2 - 2*y) ./ t);
t = 1:0.01:3;
y0 = 4.2;
[t2, y2] = ode45(my_ode, t, y0);
```

EDOs de orden n

Se hacen cambios de variables para pasar de 1 ecuación diferencial de orden *n* a *n* ecuaciones diferenciales de primer orden.

Ejemplo: Ecuación de van der Pol:

$$y'' - \mu\,(1 - y^2)\,y' + y = 0$$

Haciendo,

$$\begin{aligned} y_2 &= y' \\ y_1 &= y \end{aligned}$$

Entonces,

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= \mu\,(1 - y_1^2)\,y_2 - y_1 \end{aligned}$$

$$
\begin{aligned}
y_1' &= y_2 \\
y_2' &= \mu\,(1 - y_1^2)\,y_2 - y_1
\end{aligned}
$$

Ejemplo

```
mu = 1;
vdpol=@(t,y) [y(2); mu * (1-y(1).^2) * y(2) - y(1)];
[t,y] = ode45(vdpol,[0 20],[2; 0]);
plot(t,y(:,1),'-' ,t,y(:,2),'-.' )
legend('y_1', 'y_2' )
```

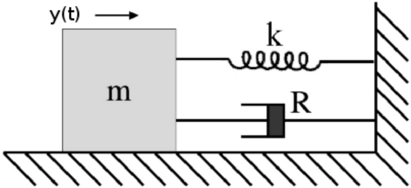
Ejercicio 11

La ecuación general de una masa con resorte responde a la ecuación,

$$m\,y" + c\,y' + k\,y = F(t)$$

Donde,

$y(t)$, posición del resorte.
m, masa.
c, coeficiente de fricción.
k, constante del resorte.
$F(t)$, fuerza externa aplicada al resorte.



Resuelva el sistema para $m = 1$ kg, $c = 0.1$ kg/s, $k = 1$ N/m y $F = 0$ N, para 100 segundos de simulación. Suponga que en el instante inicial la masa está en $y = 10$ m y que la velocidad inicial es nula.

Ejercicio 11, solución

$$m\,y'' + c\,y' + k\,y = 0$$

Haciendo

$$y_2 = y' \;,\;\; y_1 = y$$

Entonces,

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= -\frac{c}{m}\,y_2 - \frac{k}{m}\,y_1 \end{aligned}$$

$$y_1' = y_2 \;,\;\; y_2' = -\frac{c}{m}\,y_2 - \frac{k}{m}\,y_1$$

Ejercicio 11, solución

Respuesta

```
m = 1; c = 0.1; k = 1;
t = 0:0.1:100;
resorte=@(t,y) [y(2); -c./m.*y(2) -k./m.*y(1)];
[t,y] = ode45(resorte, t, [10; 0]);
plot(t,y(:,1),'-' ,t,y(:,2),'-.'  )
legend('y_1', 'y_2' )
```