[next](http://www.fim.utp.ac.pa/Revista/Matlab/node28.html)[up](http://www.fim.utp.ac.pa/Revista/Matlab/node23.html)[previous](http://www.fim.utp.ac.pa/Revista/Matlab/node26.html)  
**Next:** [Archivos de datos.](http://www.fim.utp.ac.pa/Revista/Matlab/node28.html) **Up:** [Otras órdenes en programación.](http://www.fim.utp.ac.pa/Revista/Matlab/node23.html) **Previous:** [Órdenes de ruptura.](http://www.fim.utp.ac.pa/Revista/Matlab/node26.html)

**Instrucciones eval y feval.**

Al programar ciertos algoritmos, como por ejemplo métodos numéricos, nos damos cuenta que si queremos aplicarlos a distintas funciones hemos de editarlos y cambiar el nombre a la función cada vez que esta aparezca en el programa. Esto se puede simplificar en MATLAB si utilizamos la orden **feval**, que se escribe con el siguiente formato,

**feval**(' *función*', *arg1*, ... , *argn*)

y equivale a

*función(arg1, ... ,argn)*.

La utilidad principal de esta instrucción, que puede ser observada en el ejemplo posterior, radica en que podemos introducir el nombre de una función en un programa como una cadena de caracteres, mediante los argumentos de entrada de dicho programa.

Como muestra implementamos el método de Euler con paso fijo $h$para el problema de valores iniciales

\begin{displaymath}\left \{ \begin{array}{l}
y'(t)=f(t,y(t)), \\
y(t_0)=y_0,
\end{array} \right .\end{displaymath}

donde $f$es una función cualquiera. Recordemos que el método de Euler nos suministra la sucesión definida por recurrencia

\begin{displaymath}\left \{ \begin{array}{l}
t_{n+1}=t_n+h, \\
y_{n+1}=y_n+hf(t_n,y_n),
\end{array} \right .\end{displaymath}

a partir de los valores $t_0$e $y_0$.

function [t,y]=euledo(fun,t0,t1,y0,h)   
% La orden euledo permite resolver numéricamente el   
% problema de valores iniciales   
% y'(t)=fun(t,y(t)),   
% y(t0)=y0,   
% la orden euledo tiene el siguiente formato   
% [t y]=euledo(fun,t0,t1,y0,h)   
% donde   
%     t0         es el punto donde damos la condición inicial.   
%     t1         es el extremo del intervalo donde calculamos la   
% solución.   
%     t            es un vector que almacena los puntos de la   
% partición del intervalo [t0,t1].   
%     y            almacena las aproximaciones a la solución en   
% cada punto de la partición.   
%     y0         valor de y en la condición inicial.   
%     h            paso del método de Euler.   
%   
if ((t1-t0)\*h<=0)|($\sim$isstr(fun))   
$\mathstrut$error ('Error en los datos.')   
end   
t(1)=t0;   
it=1;   
y(1)=y0;   
while (t1-t(it))\*h>0   
$\mathstrut$y(it+1)=y(it)+h\*feval(fun,t(it),y(it));   
$\mathstrut$t(it+1)=t(it)+h;   
$\mathstrut$it=it+1;   
end

La instrucción **eval** es similar a **feval**. Su formato es

**eval**(' *str* ')

y el resultado que se tiene al ejecutar dicha orden, es el que se obtiene al ejecutar la instrucción *str*. Por ejemplo, la instrucción

$\gg$eval('a=2');

es equivalente a

$\gg$a=2;

y si a la variable **fun** se le asigna una cadena con el nombre de una función, por ejemplo **'atan2'**, entonces

$\gg$eval([fun '(3,a)']);

es equivalente a

$\gg$feval(fun,3,a);

[next](http://www.fim.utp.ac.pa/Revista/Matlab/node28.html)[up](http://www.fim.utp.ac.pa/Revista/Matlab/node23.html)[previous](http://www.fim.utp.ac.pa/Revista/Matlab/node26.html)  
**Next:** [Archivos de datos.](http://www.fim.utp.ac.pa/Revista/Matlab/node28.html) **Up:** [Otras órdenes en programación.](http://www.fim.utp.ac.pa/Revista/Matlab/node23.html) **Previous:** [Órdenes de ruptura.](http://www.fim.utp.ac.pa/Revista/Matlab/node26.html)   
[I:\ARCHIVOS MATLAB\Archivos HTML\Instrucciones eval y feval_archivos\es.gif](http://es.nedstat.net/cgi-bin/viewstat?name=fimutp)