

Ciclo while

Vamos a tomar de ejemplo la programación del método de newton(metodo delle tangenti) Por ser un caso sencillo para encontrar soluciones de ecuaciones

Este método requiere de un valor inicial (x_0) y de una función $F(x)$

Aplica la siguiente fórmula

$$x = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Con esta fórmula se halla un nuevo x , si este x no es igual al x_0 asumido entonces este nuevo x se convertirá en x_0 para hallar nuevamente un x , y así sucesivamente, Es decir, que mientras (**WHILE**) que ese x no sea igual a x_0 ($x \neq x_0$) entonces debemos convertir ese nuevo x en x_0 ($x_0 = x$) y luego hallar un nuevo x con la fórmula

Los pasos para este método :

- Hacerle saber a matlab que x la vamos a usar como variable
- Ingresar función ($f(x)$)
- Ingresar un valor inicial de x (x_0)
- Hallar un nuevo valor para x (x)
- Mientras x sea distinto de x_0
- Asignar a x_0 el valor de x
- Hallar un nuevo valor para x (x)
- Fin del ciclo mientras
- Mostrar valor de x

Para **hacerle saber a MATLAB que x la vamos a usar como variable** solo basta colocar :

syms x

Nota: en caso de que tengamos más variables estas pueden ir separadas por un espacio (syms x x1 x2)

Ingresar la función ($f(x)$)

Una vez matlab sabe que x es una variable podemos **ingresar la función** en términos de dicha variable :

$Y=X^2-10*X+16;$

##Nota: si queremos que Matlab nos pida la función cuando se ejecute el programa entonces no se coloca la función sino :

`Y=input( ' ingrese la función(ou quelque chose comme ça) ' ;`
 Ingresar un valor inicial de x :

Puede ser `x0=10;` o como en el caso anterior `x0=input( ' ingrese x inicial ' );` para que lo pida en el momento de ejecutar el código . **##**

Para **Hallar un nuevo valor de x, solo basta con aplicar la fórmula** pero en este caso contiene una derivada y tanto la función como la derivada deben evaluarse en un punto

La función **Y** que ingresamos no puede evaluarse en un punto al menos que le demos un valor a x y la evaluemos (**eval(Y)**) pero también podemos convertirla en una función de tipo f(x) donde x puede ser cualquier número (f(1), f(2) , etc) , para hacer esto se escribe :

f=inline(Y) ;

con esto ya podemos escribir **f(cualquier número)** y nos dará el resultado de evaluar la función en dicho punto ,

##NOTA: Y(cualquier número) no es posible hacerlo pero si queremos que sea posible para no crear otra función como en el caso anterior que se crea una función f , entonces se le asigna el mismo nombre

Y=inline(Y) ;

El problema es que Y que antes era una función de tipo “sym” ahora pasa a ser de tipo “inline”

Y con ella no podemos hacer ciertas operaciones como por ejemplo derivarla .**##**

Ahora calculamos la derivada de la función :

dY=diff(Y);

y se convierte a tipo “inline” para poder evaluarla en un punto

df=inline(dY);

ya con esto si se puede **hallar el valor de x:**

x=x0-f(x0)/df(x0);

Mientras x sea distinto de x_0 :

while $x \neq x_0$

##NOTA: en ocasiones no se requiere que sean totalmente iguales sino aproximadamente iguales , esto con el fin de que se llegue a una respuesta más rápido por lo que se puede poner entonces :

while $\text{abs}(x - x_0) \geq 0.00001$ %un número pequeño

mientras la diferencia de $x - x_0$ sea mayor a un número pequeño y se pone valor absoluto porque no importa quién es mayor no quien es menor **##**

Asignar a x_0 el valor de x :

$x_0 = x$;

Hallar un nuevo valor para x :

$x = x_0 - f(x_0)/df(x_0)$;

Fin del ciclo mientras (hasta aquí llegan las instrucciones que se van a ejecutar mientras x sea distinto de x_0) :

End

Mostrar valor de x :

disp(x)

en total tenemos :

```
syms x
Y=input('ingrese la función');
x0=input('ingrese x inicial');
f=inline(Y);
dY=diff(Y);
df=inline(dY);
x=x0-f(x0)/df(x0);
while x~=x0
    x0=x;
    x=x0-f(x0)/df(x0);
end
disp(x)
```

o si se hace asignando valor y evaluando :

```

syms x
Y=input('ingrese la función');
x0=input('ingrese x inicial');
x=x0;
x0=x-eval(Y)/eval(diff(Y));
while x~=x0
    x=x0;
    x0=x-eval(Y)/eval(diff(Y));
end
disp(x)

```

en este último caso se puede observar que dos líneas se repiten, y esto sucede porque primero necesitamos hacer (quiere decir ejecutar las dos líneas) , luego evaluar y con respecto a eso volver a hacer , es decir, que ese hacer por lo menos se ejecuta una sola vez (en caso de que al ejecutar las dos líneas antes del while, las x den iguales , entonces Matlab se salta el while y procede a mostrar la respuesta de inmediato)

el ciclo while primero evalúa la condición y luego hace (evaluar condición / hacer / evaluar condición / hacer)

y en este primer caso necesitamos que haga y luego evalúe (Java cuenta con la función do-while para esto) , con esto se evitan repetir estas dos líneas

entonces se puede usar una variable que nos mantenga al tanto de la condición (en java esta variable es de tipo boolean y toma valores de true o false) esta variable(non so come chiamarla) tendrá el valor de 1 cuando la condición se cumpla y un valor de 0 cuando la condición no se cumpla

como queremos que el ciclo se ejecute por lo menos una vez la iniciamos con un valor de 1 antes del ciclo y después de ejecutar las acciones del while esta obtendrá el valor de la condición :

```

syms Y(x)
Y=input('ingrese la función');
x0=input('ingrese x inicial');
b=1;
while b
    x=x0;
    x0=x-eval(Y)/eval(diff(Y));
    b=x~=x0;
end
disp(x)

```