UN MÉTODO NUMÉRICO CERRADO PARA LA SOLUCIÓN DE SISTEMAS DE ECUACIONES NO LINEALES EN DOS VARIABLES

MAURICIO AREIZA

Estudiante de Ingeniería Civil, Facultad de Minas, Universidad Nacional de Colombia, Medellín.

Recibido para revisar 30 de Noviembre de 2001, aceptado 24 de Junio de 2002, versión final 27 de julio de 2002.

RESUMEN: En este trabajo se presenta un método numérico cerrado para la solución de sistemas de ecuaciones no lineales en dos variables. Dicho método se caracteriza por ser un método cerrado de rápida convergencia y fácil implementación computacional. De igual modo se prueba que el método de posición falsa (Regula Falsi) es un caso particular del método propuesto. Este algoritmo también puede ser utilizado para la solución de una ecuación no-lineal en una variable.

PALABRAS CLAVES: Método Numérico, Ecuaciones No lineales, Algoritmo.

ABSTRACT: In this work a closed numerical method is presented for the solution of systems of no-lineal equations in two variables. This method is characterized as a closed method of quick convergence and easy computer implementation. In a same way it is proven that the method of false position (Regula Falsi) is a a particular case of the proposed method. This algorithm can also be used for the solution of a no-linear equation in a variable.

KEYWORDS: Numeric Method, No-Lineal Equations, Algorithm.

1. INTRODUCCIÓN

Para la solución de un problema físico-matemático algunas veces se llega a la necesidad de solucionar un sistema de ecuaciones no lineales. Este tipo de sistemas no puede ser resuelto por los métodos analíticos algebraicos convencionales y se hace necesario la implementación de un método numérico para encontrar una solución aproximada.

Los métodos numéricos para la solución de ecuaciones no lineales en una variable se clasifican en dos tipos: Métodos cerrados y métodos abiertos. Los métodos cerrados son más confiables que los métodos abiertos, debido a que el usuario tiene más control sobre el problema y su convergencia está asegurada. Los métodos abiertos tienen la desventaja que su convergencia se asegura sólo si la función cumple con ciertos criterios tales como: derivabilidad, continuidad, punto fijo, entre otros.

Para la solución de sistemas de ecuaciones no lineales en dos variables se dispone sólo de métodos abiertos tal como el método de Newton, al igual que para el caso de una variable su convergencia sólo se asegura cuando la función cumple con criterios de derivabilidad, continuidad, punto fijo, entre otros.

El método propuesto puede ser utilizado para la solución de sistemas de ecuaciones no lineales en dos variables y tiene como ventajas ser un método cerrado, talvez su principal característica; de fácil implementación computacional; no necesita criterios de derivabilidad o punto fijo para garantizar su convergencia y además puede también ser utilizado para la solución de ecuaciones no lineales en una variable

2. MODELO MATEMÁTICO

Sean dos funciones $f_1(x)$, $f_2(x)$, continuas en un intervalo $[x_0, x_1]$, al cual la raíz buscada debe pertenecer. El método consiste en generar dos líneas rectas entre los puntos $P_1(x_0, f_1(x_0))$, $P_4(x_1, f_1(x_1))$; $P_2(x_0, f_2(x_0))$, $P_3(x_1, f_2(x_1))$ respectivamente, como se muestra en la Figura 1. La solución del sistema lineal en dos variables será el punto (x_2, y_2); x_2 será la primera aproximación a la raíz buscada. Luego se hallan los puntos $(x_2, f_1(x_2))$, $(x_2, f_2(x_2))$; con este nuevo par de puntos y con $P_1(x_0, f_1(x_0))$, $P_2(x_0, f_2(x_0))$ se crean dos nuevas rectas respectivamente, la solución del nuevo sistema lineal obtendrá el punto (x_3, y_3), x_3 será una nueva y mejor aproximación a la raíz buscada. El mismo procedimiento se repite varias veces hasta encontrar que $|f_1(x_n) - f_2(x_n)| \le \varepsilon$, con ε muy pequeño.

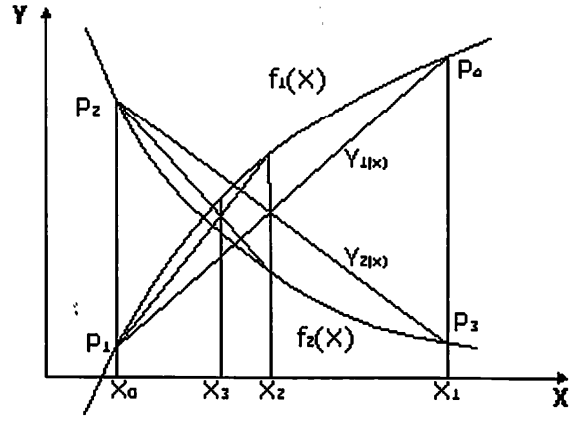

Figura1. Modelo matemático.

3. DERIVACIÓN DE LAS ECUACIONES

La ecuación de iteración demuestra que la solución de un problema que en principio es no lineal, puede ser resuelto por medio de aproximaciones sucesivas que se hacen por medio de la solución de un problema lineal simple de dos ecuaciones y dos incógnitas, donde una de las soluciones es la aproximación a la raíz buscada.

La ecuación de la línea recta $Y_1(x)$ (ver Figura 1) que une los puntos P_1, P_4 es:

$$Y_1 = f_1(x_o) + \frac{\left(f_1(x_o) - f_1(x_1)\right)}{(x_o - x_1)}(x - x_o) \quad (1a)$$

La ecuación de la línea recta $Y_2(x)$ que une los puntos P_2, P_3 es:

$$Y_2 = f_2(x_o) + \frac{\left(f_2(x_o) - f_2(x_1)\right)}{(x_o - x_1)}(x - x_o) \quad (1b)$$

Igualando las ecuaciones 1a,1b y despejando la variable x obtenemos la solución al sistema de ecuaciones. Como x es la primera aproximación a la raíz de las ecuaciones se llamará x_2:

$$x_2 = x_o + (x_o - x_1)\left[\frac{f_1(x_1) - f_2(x_1)}{(x_o - x_1)} - 1\right]^{-1} \quad (2)$$

Haciendo el mismo procedimiento para n-iteraciones se puede llegar a la fórmula:

$$x_n = x_o + (x_o - x_1)\left[\frac{f_1(x_{n-1}) - f_2(x_{n-1})}{f_1(x_o) - f_2(x_o)} - 1\right]^{-1}; n = 1,2,... \quad (3)$$

De la Ecuación 3 se puede observar que si hacemos por ejemplo la función $f_2(x) = 0$; se obtendrá el método de Regula Falsi.

$$x_n = x_o - \frac{(x_{n-1} - x_o)}{f(x_{n-1}) - f(x_o)} f(x_o); n = 1,2,... \quad (4)$$

Lo que demuestra que el algoritmo de punto fijo es un caso particular del algoritmo propuesto.

4. EJEMPLOS COMPARATIVOS

Ejemplo 1:

Supongamos que estamos interesados en encontrar todas las raíces de la ecuación:

$$3x^2 - e^x = 0$$

Para iniciar la búsqueda hacemos la gráfica de la función y así encontrar los intervalos que contengan dichas raíces.

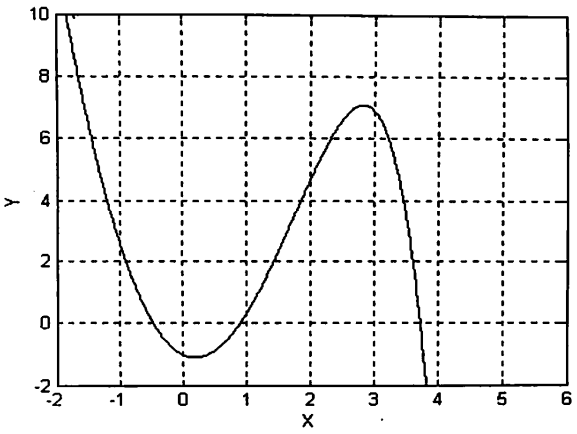

Figura 2. Gráfica de $3x^2 - e^x = 0$

La tolerancia será de: $|Xn\text{-}Xn\text{-}1| < 0.0005$

- **Método de Bisección**

Utilizando bisección en el intervalo [0.5,1]

$X_{10} = 0.909667969$, $|Xn\text{-}Xn\text{-}1| = 4.882 \times 10^{-4}$

Se observa su lenta convergencia

- **Método de *Regula Falsi***

Utilizando Regula Falsi en el intervalo [0.5,1]

$X_8 = 0.9100300050$, $|Xn\text{-}Xn\text{-}1| = 0.0001866737$

- **Método propuesto**

Utilizando el método propuesto en el intervalo [0.5,1], con una precisión de 0.0005, haciendo:

$$f_1 = 3x^2,\ f_2 = e^x$$

tenemos:

$X_8 = 0.9100300050$, $|Xn\text{-}Xn\text{-}1| = 0.0001866737$

Ejemplo 2:

Utilizando el método propuesto para hallar la tercera raíz positiva de las ecuaciones:

$$f_1 = sin(tan(x)),\ f_2 = \exp(x-3),$$

Las cuales se ilustran en la Figura 3. utilizando el método propuesto encontramos que la solución al problema es:

$X_3 = 1.4179202622$
$f_1(X_3) = 0.205547169142064$
$f_2(X_3) = 0.205547169146523$
$|f_2(X_3)\text{-}f_1(X_3)| = 4.45840586671409 \times 10^{-12}$

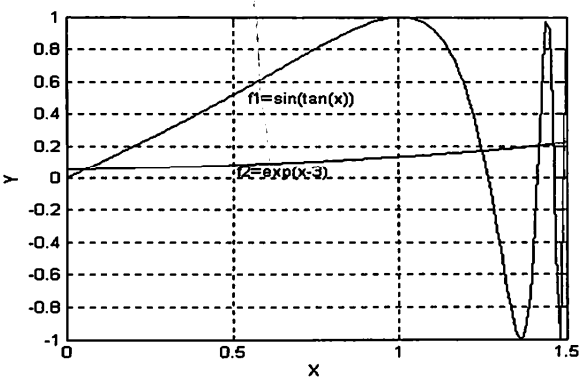

Figura 3. Gráfico de las ecuaciones del ejemplo 2

Ejemplo 3:

El algoritmo propuesto se puede utilizar para solucionar sistemas de dos por dos no lineales con el siguiente procedimiento:

Sean $Z_1 = F_1(X,Y)$; $Z_2 = F_2(X,Y)$

1. Se hallan las trazas de las funciones Z_1, Z_2 con el plano $Z = K$.
2. Cada traza ahora será una relación de dos variables, las cuales se pueden convertir en

un par de funciones de X, así: $Y_1 = f_1(X)$, $Y_2 = f_2(X)$.

3. Encontrar la intersección de estas funciones (X^*, Y^*) con el algoritmo propuesto anteriormente.

4. Los valores de: X^*, $Y^* = Y_1 = Y_2$, $Z = K$ son la solución para el sistema de dos por dos no lineal.

Ejemplo 3.1

Sean

$$Z_1 = F_1(x, y) = \exp(y) + sin(x),$$
$$Z_2 = F_2(x, y) = y * Tan(x)$$

Dos funciones en x y y, encontrar los puntos de intersección de las superficies que pasan por el plano $Z = 1$. Se hallan las trazas de Z_1, Z_2 con el plano $Z = 1$, dando como resultado las siguientes funciones (Figura 4):

$$f_1(x) = Ln(1 - sin(x)) \ , \ f_2(x) = 1/Tan(x)$$

Una vez se conocen las funciones se utiliza el algoritmo propuesto (programado en Matlab® como se presenta en el Apéndice) para hallar algunas de las raíces del sistema

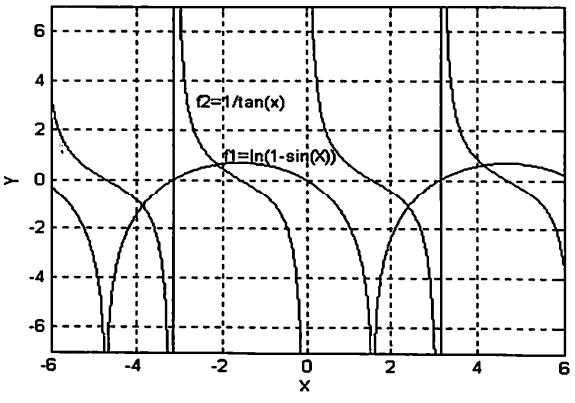

Figura 4. Gráficas de las funciones del ejemplo 3.1

- *Raíces pertenecientes al intervalo [-4,-2]*

Con un punto inicial $Xo = -4$, con tolerancia = 1×10^{-10}, obtenemos:

$X_4 = -3.8756538135,$
$f_1(X_4) = -1.10833055780636$
$f_2(X_4) = -1.10833055,$
$|f_1 - f_2| = 1.9491519509529e\text{-}011$

Se observa su rápida convergencia ($n = 4$) y su pequeño error. Se puede comprobar que:

$Z_1 = exp(f_1) + sin(X_4) = 0.999999999996765$
$Z_2 = f_1 * tan(X_4) = 1.00000000000883$

Con un punto inicial $Xo = -3.0$, con tolerancia = 1×10^{-10}, obtenemos:

$X_5 = -2.1228706365,$
$f_1(X_5) = 0.615962890228079$
$f_2(X_5) = 0.615962890225$
$|f_1 - f_2| = 2.5507373990 \times 10^{-12}$

Se observa su rápida convergencia ($n = 5$) y su pequeño error. Se puede comprobar que:

$Z_1 = exp(f_1) + sin(X_4) = 1.0000000000008$
$Z_2 = f_1 * tan(X_4) = 1.00000000000072$

- *Raíces pertenecientes al intervalo [2,4]*

Con un punto inicial $Xo = 2$, con tolerancia = 1×10^{-10}, obtenemos

$X_4 = 2.4075314936,$
$f_1(X_4) = -1.10833055781752$
$f_2(X_4) = -1.1083305577$
$|f_1 - f_2| = 4.17172962e\text{-}011$

Se observa su rápida convergencia ($n = 4$), su pequeño error. Se puede comprobar que:

$Z_1 = exp(f_1) + sin(X_4) = 0.999999999993072$
$Z_2 = f_1 * tan(X_4) = 1.00000000001888$

Ejemplo 3.2

Considere las funciones:

$$F_1(x, y) = x^2 + 4y^2 - 4$$

$$F_2(x,y) = x^2 - 2x - y + 0.5$$

Se desea encontrar los interceptos de las funciones cuando $F_1(x, y) = F_2(x, y) = 0$. En la Figura 5 se observa la gráfica de las funciones al despejar y = f(x):

$$f_1(x) = \pm\sqrt{(4 - x^2)/4}$$

$$f_2(x) = x^2 - 2x + 0.5$$

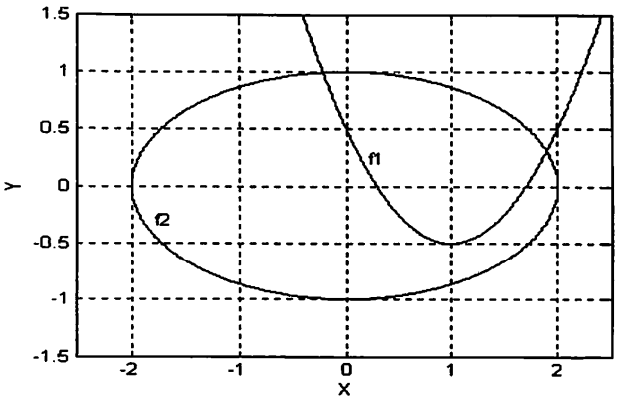

Figura 5. Gráfica de las funciones $f_1(x)$ y $f_2(x)$ del ejemplo 3.2

Se observa que existen dos raíces. Utilizando el algoritmo propuesto hecho en Matlab® (ver Apéndice) se obtiene:

con *Xo= -1:*
$X_5 = -0.2222145551$
$Y_5 = 0.9938084186$

Con *Xo=1:*
$X_4 = 1.9006767264,$
$Y_4 = 0.3112185654$

El mismo ejemplo fue realizado por (*Mathews y Fink*, 2000) utilizando el método de punto fijo para dos dimensiones, obtiendo:

$X_9 = -0.2222146,$
$Y_9 = 0.9938084$
$X_9 = 1.900677$
$Y_9 = 0.3112186$

5. CONCLUSIONES

De los ejemplos presentados se observa que el algoritmo propuesto converge en un intervalo continuo que contenga la raíz que se desea buscar. La rapidez de convergencia es mayor o igual que la de los algoritmos convencionales tratados en este artículo. Para poder implementar el algoritmo con más confiabilidad se recomienda hacer siempre una gráfica de las funciones.

Se hace notar que el algoritmo propuesto es más general que el de *Regula Falsi*, sin embargo, poseen igual rapidez de convergencia.

El algoritmo presentado hecho en Matlab®, demuestra lo fácil que puede ser implementarlo en un computador o calculadora de bolsillo.

La característica principal del algoritmo propuesto es que puede ser utilizado en la solución de un sistema de ecuaciones no lineales de dos por dos de una manera trabajable, garantizando su convergencia y buena rapidez.

Se hace notar que el ejemplo 3 no es fácil de realizar cuando las trazas de las funciones (Z_1,Z_2) no pueden ser expresadas como funciones de X, sin embargo, si éste es el caso, también podría utilizarse el algoritmo propuesto o cualquier otro para encontrar curvas que se aproximen a las trazas con la precisión que el usuario desee. Una vez encontradas las trazas aproximadas se utiliza el algoritmo propuesto para hallar la solución al sistema.

El procedimiento anterior es un poco laborioso, pero la garantía de su convergencia lo hace preferible a los métodos matriciales iterativos (Punto fijo, Newton-Rapson), los cuales garantizan su convergencia en un intervalo si ambas funciones cumplen con los criterios como son: punto fijo, continuidad, derivabilidad.

Los criterios anteriores no son fáciles de probar, además, en los problemas típicos de ingeniería, sólo en algunos casos muy especiales se presentan funciones que cumplan dichos criterios.

REFERENCIAS

Asmar, I.,. *Métodos Numéricos*, 1997.

Nakamura, S., *Análisis numérico y visualización gráfica con Matlab*, 1997.
Matews, J., *Numerical Methods for Mathematics, Science, and, Engineering*, 1997.
Atkinson, L.; Harley, P., *Introducción a los métodos numéricos con pascal*, México: Addison-Wesley Iberoamericana, 1987.
Kreyszing, E., *Matemáticas avanzadas para ingeniería*, Limusa-Wiley, México, 2000.
Mathews, J.; Fink, D., *Métodos Numéricos con Matlab*, Prentice, España, 2000.

Apéndice

El siguiente algoritmo programado en Matlab® es sugerido para la implementación del método numérico.

```
clear
xo=input('Introduzca el número inicial\n');
z=0;
while z>=0
  x1=xo+.01;
  [f1xo,f2xo]=f(xo);
  [f1x1,f2x1]=f(x1);
  z1=f1xo-f2xo;
  z2=f1x1-f2x1;
  z=z1*z2;
  xo=x1;
end
X0=xo-.01;
X1=xo;
n=0;
while abs(f1x1-f2x1)>1E-10
  [f1xo,f2xo]=f(X0);
  [f1x1,f2x1]=f(X1);
  X2=X0+(X0-X1)*(((f1x1-f2x1)/(f1xo-f2xo))-
1)^(-1);
  X1=X2;
  n=n+1;
  if n>1000;
    fprintf('max itera');
    break
  end
end
z=f1x1-f2x1;
disp(' ');
fprintf('la raiz es=%1.10f\n',X1);

function [f1,f2]=f(x);
f1=sin(tan(x));
f2=exp(x-3);
```