**Universidad de Puerto Rico**
**Departamento de Matemáticas**
**Humacao, Puerto Rico 00791**

MATE 4061 Análisis Numérico Prof. Pablo Negrón

# Laboratorio 7: Sistemas de Ecuaciones Nolineales

Consideramos ahora el problema de resolver un sistema de ecuaciones nolineales de $n$ ecuaciones en $n$ desconocidas. Sean $f_i : \mathbb{R}^n \to \mathbb{R}$, $1 \leq i \leq n$ funciones (nolineales) suficientemente diferenciables. Un sistema nolineal $n \times n$ se puede escribir de la forma:

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0, \\ f_2(x_1, x_2, \ldots, x_n) = 0, \\ \qquad\qquad \vdots \\ f_n(x_1, x_2, \ldots, x_n) = 0. \end{cases} \tag{1}$$

Si definimos $\vec{F} : \mathbb{R}^n \to \mathbb{R}^n$ por $\vec{F} = (f_1, f_2, \ldots, f_n)^t$, entonces podemos escribir (1) en forma vectorial como:

$$\vec{F}(\vec{x}) = \vec{0} \quad , \quad \vec{x} = (x_1, x_2, \ldots, x_n)^t. \tag{2}$$

Sea $\vec{\alpha} \in \mathbb{R}^n$ tal que $\vec{F}(\vec{\alpha}) = \vec{0}$, i.e., una solución de (2). Definimos la matriz $(n \times n)$ *Jacobiana* del sistema (2) por:

$$\vec{F}'(\vec{x}) = \left( \frac{\partial f_i(\vec{x})}{\partial x_j} \right)_{i,j=1,\ldots,n}. \tag{3}$$

Suponga que $\vec{x}_0$ es una aproximación de $\vec{\alpha}$. Entonces usando el Teorema de Taylor para funciones de varias variables, podemos escribir que

$$\vec{F}(\vec{x}) \approx \vec{F}(\vec{x}_0) + \vec{F}'(\vec{x}_0)(\vec{x} - \vec{x}_0).$$

Definimos ahora la siguiente aproximación $\vec{x}_1$ como la solución de

$$\vec{F}(\vec{x}_0) + \vec{F}'(\vec{x}_0)(\vec{x} - \vec{x}_0) = \vec{0},$$

i.e.,

$$\vec{x}_1 = \vec{x}_0 - \vec{F}'(\vec{x}_0)^{-1}\vec{F}(\vec{x}_0).$$

De esta forma continuamos obteniendo así la versión para sistemas del *Método de Newton* dada por:

$$\begin{cases} \vec{x}_{k+1} = \vec{x}_k - \vec{F}'(\vec{x}_k)^{-1}\vec{F}(\vec{x}_k) \quad , \quad k \geq 0, \\ \vec{x}_0 \ \text{ dado.} \end{cases} \tag{4}$$

Si $\vec{F}'(\vec{\alpha})$ es nosingular, y $\vec{x}_0$ se toma suficientemente cerca de $\vec{\alpha}$, entonces se puede demostrar que las iteraciones $(\vec{x}_k)$ convergen a la raíz $\vec{\alpha}$. Las iteraciones (4) se pueden reescribir para que no haya que calcular el inverso de una matriz en cada iteración. Esto se hace de la forma:

$$\begin{cases} \vec{F}'(\vec{x}_k)\vec{z}_k = -\vec{F}(\vec{x}_k), \\ \vec{x}_{k+1} = \vec{x}_k + \vec{z}_k \quad , \quad k \geq 0, \\ \vec{x}_0 \ \text{ dado.} \end{cases} \tag{5}$$

Veamos ahora una implementación en MATLAB de estas ecuaciones. De paso ilustramos como trabajan las funciones en MATLAB con un numero variable de argumentos de entrada ó salida. El programa es como sigue:

```
function [x,iter]=newton(f,fp,x0,tol,itermax)
%NEWTON   Metodo de Newton para sistemas nolineales.
%         Los datos de entrada son:
%         f:        nombre de la funcion que representa el sistema.
%         fp:       nombre de la funcion que calcula el Jacobiano.
%         x0:       el punto inicial (vector columna).
%         tol:      tolerancia para el error relativo en la solucion
%                   calculada.
%         itermax:  numero maximo de iteraciones que se repiten las
%                   iteraciones.
%
%         Los ultimos dos argumentos son opcionales. Ejemplos de como
%         llamar esta funcion son:
%                   [X,ITER]=NEWTON('FUNC','DFUNC',x0,1.0e-6,100)
%         o
%                   X=NEWTON('FUNC','DFUNC',x0)
if nargin<4
   tol=1.0e-4;
end
if nargin<5
   itermax=20;
end
x=x0;
normx=0;
normz=inf;
iter=0;
while (normz>tol*normx)&(iter<=itermax)
    f0=feval(f,x);
    fp0=feval(fp,x);
    z=-fp0\f0;
    normz=norm(z,2);
    normx=norm(x,2);
    x=x+z;
    iter=iter+1;
end
```

Esta función se debe invocar con al menos tres argumentos. Si se omite alguno de los últimos dos argumentos, la función tiene unos valores que se asignan por omisión a estas variables.

**Ejemplo 1.** Considere el problema de aproximar una solución del sistema:

$$\begin{cases} x^3 - xy^2 + y^3 = 0, \\ x\,\mathrm{sen}(xy) + 1 = 0. \end{cases}$$

Tenemos con $\vec{x} = (x, y)^t$ que

$$\vec{F}(\vec{x}) = \begin{bmatrix} x^3 - xy^2 + y^3 \\ x\,\mathrm{sen}(xy) + 1 \end{bmatrix} \quad , \quad \vec{F}'(\vec{x}) = \begin{bmatrix} 3x^2 - y^2 & -2xy + 3y^2 \\ \mathrm{sen}(xy) + xy\cos(xy) & x^2\cos(xy) \end{bmatrix}.$$

Estas dos expresiones las calculamos en MATLAB mediante las siguientes funciones:

```
function z=func1(w)
z=zeros(2,1);
x=w(1);
y=w(2);
z(1)=x^3-x*y^2+y^3;
```

```
z(2)=x*sin(x*y)+1;

function z=dfunc1(w)
z=zeros(2,2);
x=w(1);
y=w(2);
z(1,1)=3*x^2-y^2;
z(1,2)=-2*x*y+3*y^2;
z(2,1)=sin(x*y)+x*y*cos(x*y);
z(2,2)=x^2*cos(x*y);
```

Tomando como $\vec{x}_0 = (1,0)^t$, podemos invocar la función `newton` para resolver el sistema como sigue:

```
[x,iter]=newton('func1','dfunc1',[1,0]')
```

lo cual produce los siguientes resultados:

```
x =

    1.1674
   -0.8812


iter =

     5
```

Es decir, $(1.1674, -0.8812)^t$ es una raíz aproximada del sistema y esta se calculó en cinco iteraciones. Podemos cotejar que tenemos una solución aproximada evaluando `func1` en el punto calculado:

```
func1(x)

ans =

  1.0e-008 *

   -0.1467
    0.0397
```

Note que esta no es la única solución del sistema. De hecho con $\vec{x}_0 = (3,4)^t$ obtenemos la solución aproximada $(-4.0392, 3.0491)^t$.

**Ejercicio 1.** *Use el Método de Newton para aproximar una solución cerca del punto* $(0.5, 1.0, 0.0)$ *con un error relativo de* $10^{-6}$ *para el sistema*

$$\begin{cases} 2x^2 - x + y^2 - z = 0, \\ 32x^2 - y^2 - 20z = 0, \\ y^2 - 14xz = 0. \end{cases}$$

**Ejercicio 2.** *Use el Método de Newton para aproximar los parámetros* $(\alpha, \beta, \gamma)$ *tal que la función* $f(x) = \alpha \exp(\beta x) + \gamma x$ *interpola a los datos* $(1,10), (2,12), (3,18)$, *i.e.,*

$$f(1) = 10 \quad , \quad f(2) = 12 \quad , \quad f(3) = 18.$$