CENG 235 ALGORİTMALARLA SAYISAL ÇÖZÜMLEME

Prof. Dr. Tufan TURACI
tturaci@pau.edu.tr

- Pamukkale Üniversitesi
- Mühendislik Fakültesi
- Bilgisayar Mühendisliği Bölümü

- Hafta 6

6. Hafta Konular

- **Lineer Olmayan Denklem Sistemlerinin Yaklaşık Çözüm Yöntemleri**

  --- Lineer Olmayan Denklem Sistemleri İçin Sabit Nokta İterasyonu

  --- Lineer Olmayan Denklem Sistemleri İçin Newton-Raphson Yöntemi

Lineer Olmayan Denklem Sistemleri

$f_1, f_2, ---, f_n$ lineer olmayan denklemler olmak üzere

$$\left.\begin{aligned} f_1(x_1, x_2, ---, x_n) &= 0 \\ f_2(x_1, x_2, ---, x_n) &= 0 \\ &\vdots \\ f_n(x_1, x_2, ---, x_n) &= 0 \end{aligned}\right\}$$

non-lineer denklem sistemi

sistemine "Lineer Olmayan Denklem Sistemi" denir.

Non-lineer denklem sistemlerini çözmek için en çok kullanılan iki yöntem:

1- Basit iterasyon yöntemi

2- Newton-Raphson yöntemi

Lineer Olmayan Denklem Sistemleri İçin Sabit Nokta İterasyonu

$f(x,y)=0$
$g(x,y)=0$ $\Big\} \Rightarrow$ iki bilinmeyenli iki denklemli lineer olmayan denklem sistemi olsun.

$x=F(x,y)$

$y=G(x,y)$ ve (x_0,y_0) başlangıç noktası olmak üzere iterasyon:

$$x_{n+1}=F(x_n,y_n)$$

$$y_{n+1}=G(x_n,y_n)$$

şeklindedir.

Yakınsama Şartı:

F ve G fonksiyonları (x_0, y_0) başlangıç noktasının komşuluğundaki bir D bölgesinde birinci kısmi türevleri sürekli olsunlar.

$\forall x, y \in D$ için $|F_x| + |F_y| < 1$

$$|G_x| + |G_y| < 1$$

şartları gerçekleşirse köke yakınsama olur.

– Üç bilinmeyenli bir sistem olsaydı:

$$f(x,y,z)=0 \qquad\qquad x=F(x,y,z)$$
$$g(x,y,z)=0 \quad \Rightarrow \quad y=G(x,y,z)$$
$$h(x,y,z)=0 \qquad\qquad z=H(x,y,z)$$

şeklinde yazılır.

İterasyon:

$$x_{n+1}=F(x_n,y_n,z_n)$$
$$y_{n+1}=G(x_n,y_n,z_n)$$
$$z_{n+1}=H(x_n,y_n,z_n)$$

şeklindedir.

Yakınsama şartı:

$\forall x,y,z \in D$ için

$$|F_x|+|F_y|+|F_z|<1$$
$$|G_x|+|G_y|+|G_z|<1$$
$$|H_x|+|H_y|+|H_z|<1$$

şeklindedir.

Örnek:
$$0.1x^2 + 0.1y^2 - x + 0.8 = 0$$
$$0.1xy^2 + 0.1x - y + 0.8 = 0$$

sisteminin çözümünü $x_0 = 0.5$ ve $y_0 = 0.5$ başlangıç değerleri kullanarak $\varepsilon = 5.10^{-3}$ hata ve 4 ondalık kullanarak basit iterasyon yöntemi ile bulunuz.

$$x = 0.1x^2 + 0.1y^2 + 0.8 = F(x,y)$$
$$y = 0.1xy^2 + 0.1x + 0.8 = G(x,y)$$

şeklinde çözelim.

Yakınsama şartı kontrol:

- $|F_x| + |F_y| < 1$?

$$|0.2x| + |0.2y| = 0.2(|x| + |y|) \overset{?}{<} 1$$

$$= 0.2(0.5 + 0.5) < 1$$

$$\boxed{= 0.2 < 1}$$

- $|G_x| + |G_y| < 1$?

$$|0.1y^2 + 0.1| + |0.2xy| < 1 \ ?$$

$$0.1(y^2+1) + 0.2(x \cdot y) < 1$$

$x_0 = 0.5$ ve $y_0 = 0.5$ için

$$\boxed{0.175 < 1}$$

Yakınsama şartları sağlandı..

$x_0 = 0.5$

$y_0 = 0.5$

$$x_1 = F(x_0, y_0) = F(0.5, 0.5) = 0.85$$

$$y_1 = G(x_0, y_0) = G(0.5, 0.5) = 0.8625$$

$$x_2 = F(x_1, y_1) = F(0.85, 0.8625) = 0.9466$$

$$y_2 = G(x_1, y_1) = G(0.85, 0.8625) = 0.9482$$

$|x_2 - x_1| > \varepsilon = 0.005$

$|y_2 - y_1| > \varepsilon = 0.005$

$$x_3 = F(x_2, y_2) = 0.9795$$

$$|x_3 - x_2| > \varepsilon = 0.005$$

$$y_3 = G(x_2, y_2) = 0.9798$$

$$|y_3 - y_2| > \varepsilon = 0.005$$

$$x_4 = F(x_3, y_3) = 0.9919$$

$$|x_4 - x_3| > \varepsilon = 0.005$$

$$y_4 = G(x_3, y_3) = 0.9920$$

$$|y_4 - y_3| > \varepsilon = 0.005$$

$x_5 = F(x_4, y_4)$

$= 0.9968$

$y_5 = G(x_4, y_4)$

$= 0.9968$

$|x_5 - x_4| = 0.0043 < \varepsilon$

$|y_5 - y_4| = 0.0048 < \varepsilon$

Her ikisinde de $\varepsilon = 0.005$'den küçük hata olduğundan iterasyonlar biter.

$5 \cdot 10^{-3}$ hata ile yaklaşık kök:

$$(0.9968, 0.9968)$$

elde edilir.

Algoritma:

```
#include<stdio.h>
#include<conio.h>
#include<locale.h>
#include<math.h>
#define hata 0.005

float f(float x,float y)
{return ((0.1)*pow(x,2))+((0.1)*pow(y,2))+(0.8)-x;}

float g(float x,float y)
{return ((0.1)*x*pow(y,2))+((0.1)*x)+(0.8)-y;}

float F(float x,float y)
{return ((0.1)*pow(x,2))+((0.1)*pow(y,2))+(0.8);}

float G(float x,float y)
{return ((0.1)*x*pow(y,2))+((0.1)*x)+(0.8);}
```

```
int main()
{setlocale(LC_ALL, "Turkish");
float x0=0.5,x1,y0=0.5,y1; int i=0;
printf("Yönteme başladığımız nokta= (%.4f,%.4f)\n",x0,y0);
do
  { x1=x0;
    y1=y0;
    x0=F(x1,y1);
    y0=G(x1,y1);
    i++;
    printf("%d. adımda yaklaşık değer= (%.4f,%.4f)\n",i,x0,y0);
    } while ((fabs(x1-x0)>hata)||(fabs(y1-y0)>hata));

printf("yaklaşık kök x=%.4f\n",x0);
printf("yaklaşık kök y=%.4f\n",y0);
printf("f(%.4f,%.4f)= %.4f\n",x0,y0,f(x0,y0));
printf("g(%.4f,%.4f)= %.4f\n",x0,y0,g(x0,y0));
getch ();
return 0;
}
```

Ekran Çıktısı:

```
Yönteme başladığımız nokta= (0,5000,0,5000)
1. adımda yaklaşık değer= (0,8500,0,8625)
2. adımda yaklaşık değer= (0,9466,0,9482)
3. adımda yaklaşık değer= (0,9795,0,9798)
4. adımda yaklaşık değer= (0,9919,0,9920)
5. adımda yaklaşık değer= (0,9968,0,9968)
yaklaşık kök x=0,9968
yaklaşık kök y=0,9968
f(0,9968,0,9968)= 0,0019
g(0,9968,0,9968)= 0,0019


--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

Örnek! $x^2 - 2x - y + 0.5 = 0$

$x^2 + 4y^2 - 4 = 0$

lineer olmayan denklem sistemini

$x_0 = 0$ ve $y_0 = 1$ başlangıç değerleri ve

$\varepsilon = 10^{-4}$ hata ve 4 ondalık kullanarak

Sabit nokta iterasyonu ile bulunuz.

$$x = \frac{x^2 - y + 0.5}{2} = F(x,y)$$

$$y = \frac{-x^2 - 4y^2 + 8y + 4}{8} = G(x,y) \text{ olsun.}$$

$\left(x^2 + 4y^2 - 4 + 8y - 8y = 0 \right.$, buradan $G(x,y)$ elde edildi. $\left. \right)$

Yakınsama Şartı:

- $|F_x| + |F_y| < 1$?

$$|x| + \left|-\frac{1}{2}\right| = |x| + \frac{1}{2} < 1$$

$x_0 = 0$ old. dan

$\boxed{|F_x| + |F_y| < 1 \text{ 'dir.}}$

- $|G_x| + |G_y| < 1$?

$$\left|-\frac{x}{4}\right| + |-y+1| < 1$$

$x_0 = 0$, $y_0 = 1$ için $\boxed{|G_x| + |G_y| < 1 \text{ 'dir.}}$

iterasyon:

$x_0 = 0$ $\quad y_0 = 1$

$x_1 = F(x_0, y_0)$

$= F(0, 1)$

$= -0.25$

$y_1 = G(x_0, y_0)$

$= G(0, 1)$

$= 1$

$|x_1 - x_0| > \varepsilon$ $\quad |y_1 - y_0| < \varepsilon$

devam ediyoruz...

$x_2 = F(x_1, y_1)$

$= -0.2188$

$y_2 = G(x_1, y_1)$

$= 0.9922$

$|x_2 - x_1| > \varepsilon$ $\quad |y_2 - y_1| > \varepsilon$

$x_3 = F(x_2, y_2)$

$= -0.2222$

$|x_3 - x_2| > \varepsilon$

$y_3 = G(x_2, y_2)$

$= 0.9940$

$|y_3 - y_2| > \varepsilon$

$x_4 = F(x_3, y_3)$

$= -0.2223$

$y_4 = G(x_3, y_3)$

$= 0.9938$

$x_5 = F(x_4, y_4)$

$= -0.2222$

$y_5 = G(x_4, y_4)$

$= 0.9938$

$x_6 = F(x_5, y_5)$

$= -0.2222$

$y_6 = G(x_5, y_5)$

$= 0.9938$

$|x_6 - x_5| < \varepsilon$ $|y_6 - y_5| < \varepsilon$

$\varepsilon = 10^{-4}$ hata ile yaklaşık kök;

$(-0.2222, \; 0.9938)$

elde edilir.

Algoritma:

```
#include<stdio.h>
#include<conio.h>
#include<locale.h>
#include<math.h>
#define hata 0.0001

float f(float x,float y)
{return pow(x,2)-2*x-y+0.5;}

float g(float x,float y)
{return pow(x,2)+4*pow(y,2)-4;}

float F(float x,float y)
{return (pow(x,2)-y+0.5)/2;}

float G(float x,float y)
{return (-1*pow(x,2)-4*pow(y,2)+8*y+4)/8;}
```

```
int main()
{setlocale(LC_ALL, "Turkish");
float x0=0,x1,y0=1,y1; int i=0;
printf("Yönteme başladığımız nokta= (%.4f,%.4f)\n",x0,y0);
do
  { x1=x0;
    y1=y0;
    x0=F(x1,y1);
    y0=G(x1,y1);
    i++;
    printf("%d. adımda yaklaşık değer= (%.4f,%.4f)\n",i,x0,y0);
    } while ((fabs(x1-x0)>hata)||(fabs(y1-y0)>hata));

printf("yaklaşık kök x=%.4f\n",x0);
printf("yaklaşık kök y=%.4f\n",y0);
printf("f(%.4f,%.4f)= %.4f\n",x0,y0,f(x0,y0));
printf("g(%.4f,%.4f)= %.4f\n",x0,y0,g(x0,y0));
getch ();
return 0;
}
```

Ekran Çıktısı:

```
Yönteme başladığımız nokta= (0,0000,1,0000)
1. adımda yaklaşık değer= (-0,2500,1,0000)
2. adımda yaklaşık değer= (-0,2188,0,9922)
3. adımda yaklaşık değer= (-0,2222,0,9940)
4. adımda yaklaşık değer= (-0,2223,0,9938)
5. adımda yaklaşık değer= (-0,2222,0,9938)
6. adımda yaklaşık değer= (-0,2222,0,9938)
yaklaşık kök x=-0,2222
yaklaşık kök y=0,9938
f(-0,2222,0,9938)= 0,0000
g(-0,2222,0,9938)= 0,0000


--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

Çalışma Sorusu:

$$y^2 - x = 0$$
$$x^2 - y = 0$$

Sistemini $x_0 = 0.25$, $y_0 = -0.25$ başlangıç değerleri $\varepsilon = 10^{-4}$ hata ve 4 ondalık kullanarak Sabit nokta iterasyonu ile bulunuz.

Yanıt: (0.0039 , 0.0039)

(2. iterasyonda bulundu!)

Lineer Olmayan Denklem Sistemleri İçin Newton-Raphson Yöntemi

Lineer olmayan denklemler için Newton-Raphson Yöntemi:

$$\boxed{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}}$$

şeklindedir.

Yakınsama Şartı: x_0 başlangıç noktası için,

$$f(x_0) \cdot f''(x_0) > 0$$

veya

$$\left| \frac{f(x_0) \cdot f''(x_0)}{[f'(x_0)]^2} \right| < 1 \quad \text{şeklindedir.}$$

Sistemler için de Newton-Raphson yöntemi

benzer şekildedir.

- Çok değişkenli fonksiyonlarda türev

için Jacobien Matrisi kullanılır.

iki bilinmeyenli sistemde

$$\begin{aligned} f(x,y) &= 0 \\ g(x,y) &= 0 \end{aligned} \quad \text{ise} \quad J = \begin{bmatrix} f_x & f_y \\ g_x & g_y \end{bmatrix} \text{ şeklindedir.}$$

Üç bilinmeyenli sistemde

$$f(x,y,z)=0$$

$$g(x,y,z)=0 \quad \text{ise} \quad J=\begin{bmatrix} f_x & f_y & f_z \\ g_x & g_y & g_z \\ h_x & h_y & h_z \end{bmatrix} \text{ şeklinde-dir.}$$

$$h(x,y,z)=0$$

J Jacobien matrisinin tersinir olduğunu kabul edelim. Yani, $|J| \neq 0$.

(x_0, y_0) başlangıç noktası için Taylor

Seri açılımı:

$$f(x,y) = f(x_0,y_0) + (x-x_0) \cdot f_x(x_0,y_0) + (y-y_0) f_y(x_0,y_0) + \cdots$$

$$g(x,y) = g(x_0,y_0) + (x-x_0) g_x(x_0,y_0) + (y-y_0) g_y(x_0,y_0) + \cdots$$

şeklindedir.

Buradan

$$f(x_0,y_0) + (x-x_0)\cdot f_x(x_0,y_0) + (y-y_0) f_y(x_0,y_0) = 0$$

$$g(x_0,y_0) + (x-x_0)\, g_x(x_0,y_0) + (y-y_0)\, g_y(x_0,y_0) = 0$$

sistemini ele alalım ve bu sisteminin (x,y) çözümünü (x_1,y_1) ile gösterelim.

$$X = \begin{bmatrix} x \\ y \end{bmatrix}, \quad \Delta X = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix},$$

$$F(x,y) = \begin{bmatrix} f(x,y) \\ g(x,y) \end{bmatrix} \quad \text{olsun.}$$

Bu sistem:

$$J(x^{(0)})\Delta X = -F(x^{(0)})$$

olarak yazılabilir.

Bu sistem çözülerek elde edilen ΔX ile $x^{(1)} = x^{(0)} + \Delta X$ yaklaşımı ile

$$x^{(1)} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

elde edilir.

Benzer şekilde $n=0,1,2,\ldots$ için

$$J(x^{(n)})\Delta X = -F(x^{(n)})$$

$$X^{(n+1)} = x^{(n)} + \Delta X$$

diğer yaklaşımlar elde edilir.

$f(x)=0$ için $x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$

$f(x,y)=0$
$g(x,y)=0$ sistemi için

$$x^{(n+1)} = x^{(n)} - \frac{f(x,y)}{j(x,y)}\Bigg|_{(x_n,\ y_n)}$$

dir.

Yani: $\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} - \dfrac{f(x_n, y_n)}{j(x_n, y_n)}$

Matrislerde bölme tanımlı olmadığından:

$$X_{n+1} = X_n - \dot{J}^{-1} \cdot F(X_n)$$

yazabiliriz.

$$\dot{J} = \begin{bmatrix} f_x & f_y \\ g_x & g_y \end{bmatrix} \qquad \dot{J}^{-1} = \frac{1}{|\dot{J}|} \cdot \begin{bmatrix} g_y & -f_y \\ -g_x & f_x \end{bmatrix}$$

Böylece:

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} - \begin{bmatrix} \frac{g_y}{|\dot{J}|} & \frac{-f_y}{|\dot{J}|} \\ \frac{-g_x}{|\dot{J}|} & \frac{f_x}{|\dot{J}|} \end{bmatrix} \cdot \begin{bmatrix} f \\ g \end{bmatrix}\Bigg|_{(x_n, y_n)}$$

elde edilir.

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} - \begin{bmatrix} \dfrac{f \cdot g_y}{|J|} - \dfrac{g \cdot f_y}{|J|} \\ \dfrac{-f \cdot g_x}{|J|} + \dfrac{g \cdot f_x}{|J|} \end{bmatrix} \Bigg|_{(x_n, y_n)}$$

$$|J| = f_x \cdot g_y - f_y \cdot g_x$$

Böylece, iterasyon denklemleri şekildedir:

$$x_{n+1} = x_n - \left(\frac{f \cdot g_y - g \cdot f_y}{f_x \cdot g_y - f_y \cdot g_x} \right) \Bigg|_{(x_n, y_n)}$$

$$y_{n+1} = y_n + \left(\frac{f \cdot g_x - g \cdot f_x}{f_x \cdot g_y - f_y \cdot g_x} \right) \Bigg|_{(x_n, y_n)}$$

Örnek:

$$x^2 + xy - 10 = 0$$

$$y + 3xy^2 - 57 = 0$$

non-lineer denklem sistemini $x_0 = 1.5$, $y_0 = 3.5$ başlangıç değerleri $\varepsilon = 10^{-3}$ hata ve 3 ondalık kullanarak Newton-Raphson yöntemi ile çözünüz.

$f(x,y) = x^2 + xy - 10$

$g(x,y) = y + 3xy^2 - 57$

$f_x = 2x + y \qquad g_x = 3y^2$

$f_y = x \qquad g_y = 1 + 6xy$

$$J = \begin{bmatrix} 2x+y & x \\ 3y^2 & 1+6xy \end{bmatrix}$$

$x_0 = 1.5 \qquad y_0 = 3.5$ için

$$x_1 = x_0 - \left(\frac{f(x_0,y_0) \cdot g_y(x_0,y_0) - g(x_0,y_0) \cdot f_y(x_0,y_0)}{f_x(x_0,y_0) \cdot g_y(x_0,y_0) - g_x(x_0,y_0) \cdot f_y(x_0,y_0)} \right)$$

$$y_1 = y_0 + \left(\frac{f(x_0,y_0) \cdot g_x(x_0,y_0) - g(x_0,y_0) \cdot f_x(x_0,y_0)}{f_x(x_0,y_0) \cdot g_y(x_0,y_0) - g_x(x_0,y_0) \cdot f_y(x_0,y_0)} \right)$$

$x_1 = 2.036$

$y_1 = 2.844$ elde edilir.

$$x_2 = x_1 - \left(\frac{f(x_1,y_1) \cdot g_y(x_1,y_1) - g(x_1,y_1) \cdot f_y(x_1,y_1)}{f_x(x_1,y_1) \cdot g_y(x_1,y_1) - g_x(x_1,y_1) \cdot f_y(x_1,y_1)} \right)$$

$$y_2 = y_1 + \left(\frac{f(x_1,y_1) \cdot g_x(x_1,y_1) - g(x_1,y_1) \cdot f_x(x_1,y_1)}{f_x(x_1,y_1) \cdot g_y(x_1,y_1) - g_x(x_1,y_1) \cdot f_y(x_1,y_1)} \right)$$

$x_2 = 1.999$

$y_2 = 3.002$ elde edilir.

$$x_3 = x_2 - \left(\frac{f(x_2, y_2) \cdot g_y(x_2, y_2) - g(x_2, y_2) \cdot f_y(x_2, y_2)}{f_x(x_2, y_2) \cdot g_y(x_2, y_2) - g_x(x_2, y_2) \cdot f_y(x_2, y_2)} \right)$$

$$y_3 = y_2 + \left(\frac{f(x_2, y_2) \cdot g_x(x_2, y_2) - g(x_2, y_2) \cdot f_x(x_2, y_2)}{f_x(x_2, y_2) \cdot g_y(x_2, y_2) - g_x(x_2, y_2) \cdot f_y(x_2, y_2)} \right)$$

$x_3 = 2.000$

$y_3 = 3.000$ elde edilir.

$$x_4 = x_3 - \left(\frac{f(x_3,y_3)\cdot g_y(x_3,y_3) - g(x_3,y_3)\cdot f_y(x_3,y_3)}{f_x(x_3,y_3)\cdot g_y(x_3,y_3) - g_x(x_3,y_3)\cdot f_y(x_3,y_3)} \right)$$

$$y_4 = y_3 + \left(\frac{f(x_3,y_3)\cdot g_x(x_3,y_3) - g(x_3,y_3)\cdot f_x(x_3,y_3)}{f_x(x_3,y_3)\cdot g_y(x_3,y_3) - g_x(x_3,y_3)\cdot f_y(x_3,y_3)} \right)$$

$x_4 = 2.000$

$y_4 = 3.000$

$|x_4 - x_3| = 0.000 < \varepsilon = 0.001$

$|y_4 - y_3| = 0.000 < \varepsilon = 0.001$

iterasyon biter!

Yaklaşık kök: $(2.000, 3.000)$

Algoritma:

```
#include<stdio.h>
#include<conio.h>
#include<locale.h>
#include<math.h>
#define hata 0.001
float F(float x,float y)
{ return pow(x,2)+x*y-10;}

float G(float x,float y)
{ return y+3*x*pow(y,2)-57;}

float FX(float x,float y)
{return 2*x+y;}

float FY(float x,float y)
{return x;}

float GX(float x,float y)
{return 3*pow(y,2);}

float GY(float x,float y)
{return 1+6*x*y;}
```

```
int main()
{setlocale(LC_ALL, "Turkish");
float x0=1.5,x1,y0=3.5,y1,x,y; int i=0;
printf("Yönteme başladığımız nokta= (%.3f,%.3f)\n",x0,y0);
do
{ x=x0;
  y=y0;
  x1=x0-((F(x0,y0)*GY(x0,y0)-G(x0,y0)*FY(x0,y0))/(FX(x0,y0)*GY(x0,y0)-GX(x0,y0)*FY(x0,y0)));
  y1=y0+((F(x0,y0)*GX(x0,y0)-G(x0,y0)*FX(x0,y0))/(FX(x0,y0)*GY(x0,y0)-GX(x0,y0)*FY(x0,y0)));
  x0=x1;
  y0=y1;
  i++;
  printf("%d. adımda yaklaşık değer= (%.3f,%.3f)\n",i,x1,y1);
  }while ((fabs(x1-x)>hata)||(fabs(y1-y)>hata));
printf("yaklaşık kök =(%.3f,%.3f)\n",x1,y1);
printf("f(%.3f,%.3f)= %.3f\n",x1,y1,F(x1,y1));
printf("g(%.3f,%.3f)= %.3f\n",x1,y1,G(x1,y1));
getch ();
return 0;
}
```

Ekran Çıktısı:

```
Yönteme başladığımız nokta= (1,500,3,500)
1. adımda yaklaşık değer= (2,036,2,844)
2. adımda yaklaşık değer= (1,999,3,002)
3. adımda yaklaşık değer= (2,000,3,000)
4. adımda yaklaşık değer= (2,000,3,000)
yaklaşık kök =(2,000,3,000)
f(2,000,3,000)= 0,000
g(2,000,3,000)= 0,000

--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

Örnek!

$$x^2 - 2x - y + 0.5 = 0$$

$$x^2 + 4y^2 - 4 = 0$$

non-lineer denklem sistemini $x_0 = 2$, $y_0 = 0$ başlangıç değerleri $\varepsilon = 10^{-3}$ hata ve 4 ondalık kullanarak Newton-Raphson yöntemi ile çözünüz.

$$f(x,y) = x^2 - 2x - y + 0.5 \qquad f_x = 2x-2 \qquad f_y = -1$$

$$g(x,y) = x^2 + 4y^2 - 4 \qquad g_x = 2x \qquad g_y = 8y$$

$$J = \begin{bmatrix} 2x-2 & -1 \\ 2x & 8y \end{bmatrix} \text{ olur.}$$

$$x_{n+1} = x_n - \left(\frac{f \cdot g_y - g \cdot f_y}{f_x \cdot g_y - g_x \cdot f_y} \right) \Bigg|_{(x_n, y_n)}$$

$$y_{n+1} = y_n + \left(\frac{f \cdot g_x - g \cdot f_x}{f_x \cdot g_y - g_x \cdot f_y} \right) \Bigg|_{(x_n, y_n)}$$

iterasyonları kullanıldığında aşağıdaki değerler elde edilir.

$x_0 = 2.0000 \quad y_0 = 0.0000$

$x_1 = 2.0000 \quad y_1 = 0.5000$

$x_2 = 1.9167 \quad y_2 = 0.3333$

$x_3 = 1.9010 \quad y_3 = 0.3116$

$x_4 = 1.9007 \quad y_4 = 0.3112$

$|x_4 - x_3| < \varepsilon = 10^{-3} \quad |y_4 - y_3| < \varepsilon = 10^{-3}$ old. dan

Yaklaşık kök : $(1.9007, 0.3112)$

Algoritma:

```
#include<stdio.h>
#include<conio.h>
#include<locale.h>
#include<math.h>
#define hata 0.001
float F(float x,float y)
{ return pow(x,2)-2*x-y+0.5;}

float G(float x,float y)
{ return pow(x,2)+4*pow(y,2)-4;}

float FX(float x,float y)
{return 2*x-2;}

float FY(float x,float y)
{return -1;}

float GX(float x,float y)
{return 2*x;}

float GY(float x,float y)
{return 8*y;}
```

```
int main()
{setlocale(LC_ALL, "Turkish");
float x0=2,x1,y0=0,y1,x,y; int i=0;
printf("Yönteme başladığımız nokta= (%.4f,%.4f)\n",x0,y0);
do
{ x=x0;
  y=y0;
  x1=x0-((F(x0,y0)*GY(x0,y0)-G(x0,y0)*FY(x0,y0))/(FX(x0,y0)*GY(x0,y0)-GX(x0,y0)*FY(x0,y0)));
  y1=y0+((F(x0,y0)*GX(x0,y0)-G(x0,y0)*FX(x0,y0))/(FX(x0,y0)*GY(x0,y0)-GX(x0,y0)*FY(x0,y0)));
  x0=x1;
  y0=y1;
  i++;
  printf("%d. adımda yaklaşık değer= (%.4f,%.4f)\n",i,x1,y1);
}while ((fabs(x1-x)>hata)||(fabs(y1-y)>hata));

printf("yaklaşık kök =(%.4f,%.4f)\n",x1,y1);
printf("f(%.4f,%.4f)= %.4f\n",x1,y1,F(x1,y1));
printf("g(%.4f,%.4f)= %.4f\n",x1,y1,G(x1,y1));
getch ();
return 0;
}
```

Ekran Çıktısı:

```
Yönteme başladığımız nokta= (2,0000,0,0000)
1. adımda yaklaşık değer= (2,0000,0,5000)
2. adımda yaklaşık değer= (1,9167,0,3333)
3. adımda yaklaşık değer= (1,9010,0,3116)
4. adımda yaklaşık değer= (1,9007,0,3112)
yaklaşık kök =(1,9007,0,3112)
f(1,9007,0,3112)= 0,0000
g(1,9007,0,3112)= 0,0000


--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

Çalışma Sorusu :

$$x^2 + y^2 - 1 = 0$$
$$x - y = 0$$

non-lineer denklem sistemini $x_0 = 1$, $y_0 = 2$ başlangıç değerleri $\varepsilon = 10^{-2}$ hata ve 4 ondalık kullanarak Newton-Raphson yöntemi ile çözünüz.

Yanıt: $(0.7071, 0.7071)$

4. iterasyonda elde edilmiştir.

Kaynaklar

- Numerical Analysis, Richard L. Burden, Brooks/Cole Cengage Learning, Boston., 2009.
- Numerical Methods for Mathematics, Science, and Engineering, 2nd Edition, John H. Mathews, Prentice Hall International Edition, 1992.
- Nümerik Analiz, (Numerical Analysis, D. Kincaid, W. Cheney, 3rd ed.(2002)), Nuri Özalp, Elif Demirci, Gazi Kitabevi Yayınları, 2012.
- Sayısal Analiz ve Mühendislik Uygulamaları, İrfan Karagöz, Nobel Yayıncılık, 2011.
- Sayısal Çözümleme, Recep Tapramaz, Literatür yayıncılık, 2002.
- Bilgisayar Uygulamalı Sayısal Analiz Yöntemleri, Eyüp Sabri Türker, Engin Can, II. Baskı, Değişim Yayınları.