Rozwiązywanie układu równań nieliniowych metodą Newtona

Moduł NSInterval

1. Zastosowanie:

Procedura NewtonsysInterval rozwiązuje układ równań nieliniowych w arytmetyce przedziałowej postaci:

$$f_k(x_1, x_2, \dots, x_n) = 0,\ k = 1,2, \dots, n. \qquad (1)$$

2. Opis metody:

Układ równań jest rozwiązywany metodą iteracyjną newtona określoną wzorem:

$$x^{(x+1)} = x^{)i)} - [Df(x^{(i)})]^{-1} * f(x^{(i)}) \qquad (2)$$

lub

$$Df(x^{(i)})x^{(x+1)} = Df(x^{(i)}) * x^{)i)} - f(x^{(i)}),\ i = 0, 1, \dots, n \qquad (3)$$

gdzie

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}, \quad f(x) = \begin{pmatrix} f_1(x_1, x_2, \dots, x_n) \\ f_2(x_1, x_2, \dots, x_n) \\ \dots \\ f_n(x_1, x_2, \dots, x_n) \end{pmatrix}, \quad Df(x) = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \dots & \frac{\partial f_1(x)}{\partial x_n} \\ \frac{\partial f_2(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_2} & \dots & \frac{\partial f_2(x)}{\partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_n(x)}{\partial x_1} & \frac{\partial f_n(x)}{\partial x_2} & \dots & \frac{\partial f_n(x)}{\partial x_n} \end{pmatrix},$$

i gdzie wektor $x^{(0)}$ jest dany. W każdym kroku iteracyjnym należy rozwiązać układ równań liniowych (2). W procedurze NewtonsysInterval stosuje się do tego celu metodę eliminacji Gaussa-Jordana z pełnym wyborem elementu podstawowego. Proces iteracyjny przerywa się, gdy dla każdego k = 1, 2, … , n zachodzi nierówność:

$$\frac{\left|x_k^{(i+1)} - x_k^{(i)}\right|}{\max\left(\left|x_k^{(i+1)}\right|, \left|x_k^{(i)}\right|\right)} < \mathbf{E}, \quad x_k^{(i+1)} \neq 0 \text{ lub } x_k^{(i)} \neq 0, \qquad (4)$$

Gdzie E oznacza dokładność zadaną z góry lub gdy $x_k^{(i+1)} = x_k^{(i)} = 0$.

3. Wywołanie procedury:

NewtonsysInterval(n, x, f, df, mit, eps, it, st).

4. Dane:

n - liczba równań układu (równa liczbie niewiadomych),

x - tablica, której elementy x[i] (i = 1, 2, … , n) zawierają początkowe przybliżenia n składowych rozwiązania,

f - funkcja języka Turbo Pascal, która dla danej wartości i oraz $x_{1,}\, x_{2,} \ldots, x_n$ oblicza wartość funkcji $f_k(x_1, x_2, \ldots, x_n)$,

df - procedura języka Turbo Pascal, która dla danej wartości i oraz $x_{1,}\, x_{2,} \ldots, x_n$ oblicza wartości pochodnych $\frac{\partial f_1}{\partial x_1}$ dla j = 1, 2, … , n,

mit - maksymalna dozwolona liczba iteracji w metodzie Nwetona,

eps - błąd względny rozwiązania.

Po wykonaniu procedury NewtonsysInterval wartości z tablicy x są zmienione.

5. Wyniki:

x - tablica zawierająca przybliżone rozwiązanie układu równań (1),

it - liczba wykonanych iteracji.

6. Inne parametry:

st - zmienna, której w procedurze NewtonsysInterval jest przypisywana jedna z następujących wartości:

1, gdy n < 1 lub mit < 1,

2, jeżeli podczas obliczeń (w którejś iteracji) macierz układu równań liniowych (3) jest osobliwa, wystąpiło dzielenie przez przedział zawierający zero,

3, jeżeli liczba iteracji jest większa niż wartość parametru mit,

0, w przeciwnym przypadku.

Jeżeli st = 1, to po wykonaniu procedury NewtonsysInterval wartości elementów tablicy x nie są zmienione. Gdy st = 2 lub 3, to tablica x zawiera ostatnio obliczone przybliżenia składowych rozwiązania.

7. Typy parametrów:

Integer: it, mit, n, st

Extended: eps

vector4: x

fxi: fi

dfxi: dfi

8. Identyfikatory nielokalne:

vector4 - nazwa typu tablicy dynamicznej $[q_1 .. q_n]$ o elementach typu **interval**, gdzie $q_1 \leq 1$ oraz $q_n \geq n$,

vector5 - nazwa typu tablicy dynamicznej $[q_1 .. q_{n-1}]$ o elementach typu **interval**, gdzie $q_1 \leq 1$ oraz $q_{n+1} \geq n + 1$,

vector6 - nazwa typu tablicy dynamicznej $[q_1 .. q_{n-1}]$ o elementach typu **Integer**, gdzie $q_1 \leq 1$ oraz $q_{n+1} \geq n + 1$,

vector7 - nazwa typu tablicy dynamicznej $[q_1 .. q_{(n+2)(n+2)/4}]$ o elementach typu **interval**, gdzie $q_1 \leq 1$ oraz $q_{(n+2(n+2)/4} \geq (n + 2)(n + 2)/4$,

fxi - identyfikator typu funkcyjnego zdefiniowany następująco:

```
type fxi = function (i, n : Integer;
                     x    : vector4) : interval;
```

dfxi - identyfikator typu proceduralnego zdefiniowany następująco:

```
type dfxi = procedure (i, n    : Integer;
                       x       : vector4;
                       var dfatx : vector4);
```

9. Przykłady:

a) Układ równań:

$$3x_1 - \cos x_2 x_3 - \frac{1}{2} = 0,$$

$$x_1^2 - 81(x_2 + 0{,}1)^2 + \sin x_3 + 1{,}06 = 0,$$

$$e^{-x_1 x_2} + 20x_3 + \frac{10\ \prod - 3}{3} = 0.$$

Deklaracje funkcji *fi* oraz *dfi*:

```
function fi (i, n : Integer;
                    x    : vector4) : interval;
begin
    case i of
        1:    fi := 3*x[1] - icos(x[2]*x[3],st) - 0.5;
        2:    fi := isqr(x[1],st) - 81*isqr(x[2]+0.1) + isin(x[3]) + 1.06;
        3:    fi := iexp(-x[1]*x[2]) + 20*x[3] + (10*Pi-3)/3
    end
end;

procedure  dfi (i, n        : Integer;
                x           : vector4;
                var dfatx : vector4);
begin
    case i of
     1 : begin
         dfatx[1]:=3;
```

```
            dfatx[2]:=x[3]*isin(x[2]*x[3],st);
            dfatx[3]:=x[2]*isin(x[2]*x[3],st)
          end;
        2 : begin
            dfatx[1]:=2*x[1];
            dfatx[2]:=-162*(x[2]+0.1);
            dfatx[3]:=icos(x[3],st)
          end;
        3 : begin
            dfatx[1]:=-x[2]*iexp(-x[1]*x[2],st);
            dfatx[2]:=-x[1]*iexp(-x[1]*x[2],st);
            dfatx[3]:=20
          end
      end
end;
```

Reszta danych:

n = 3,

x[1] = 0.1,

x[2] = 0.1,

x[3] = -0.1,

mit = 10,

eps = 1e-16.

Wyniki:

x[1] = [4.9999999999999977E-0001 ; 5.0000000000000023E-0001],

x[2] = [-2.5265298750701271E-0015 ; 2.5265417335313887E-0015],

x[3] = [-5.2359877559829967E-0001 ; -5.2359877559829808E-0001],

it = 10,

st = 3.

Jest to przykład, gdy liczba zadanych iteracji jest niewystarczającaby uzyskać zadaną dokładność.

b) Ten sam układ równań co w podpunkcie a:

Reszta danych:

n = 3,

x[1] = 1,

x[2] = 1,

x[3] = -1,

mit = 5,

eps = 1e-16.

Wyniki:

x[1] = [1.0000000000000000E+0000 ; 1.0000000000000000E+0000]

x[2] = [1.0000000000000000E+0000 ; 1.0000000000000000E+0000]

x[3] = [-1.0000000000000000E+0000 ; -1.0000000000000000E+0000]

it = 1,

st = 3.

Jest to przykład, gdy program został przerwany ze względu na to, że wystąpiło dzielenie przez przedział zawierający zero.

c) Układ równań:

$x_1^2 + 8x_2 - 16 = 0$,

$x_1 - e^{x_2} = 0$.

Deklaracje funkcji *fi* oraz *dfi*:

```
function fi (i, n : Integer;
                x    : vector4) : interval;
begin
 case i of
   1 : fi:=isqr(x[1],st)+8*x[2]-16;
   2 : fi:=x[1]-iexp(x[2],st)
 end
end;

procedure dfi (i, n      : Integer;
                  x        : vector4;
               var dfatx : vector4);
begin
        case i of
        1 : begin
                 dfatx[1]:=2*x[1];
                dfatx[2]:=8
            end;
        2 : begin
                dfatx[1]:=1;
                 dfatx[2]:=-iexp(x[2],st)
            end
```

end

end;

Reszta danych:

n = 2,

x[1] = 0,

x[2] = 0,

mit = 10,

eps = 1e-10.

Wyniki:

x[1] = [2.7908957617642316E+0000 ; 2.7908957617690158E+0000]

x[2] = [1.0263626058665869E+0000 ; 1.0263626058716874E+0000]

it = 7,

st = 0.

Jest to przykład poprawnego wykonania program. Osiągnięto zadaną dokładność, nie wystąpił błąd dzielenia przez przedział zawierający zero oraz macierz w żadnej z iteracji nie była osobliwa.

d) **Układ równań taki, jak w podpunkcie c:**

Reszta danych:

n = 2,

x[1] = 0,

x[2] = 0,

mit = 10,

eps = 1e-16.

Wyniki:

x[1] = 2,79089576176662

x[2] = 1,02636260586914

it = 8,

st = 0.

Jest to przykład poprawnego wykonania programu, gdzie osiągnięto zadaną dokładność oraz wyniki są w zwyczajnej arytmetyce (nieprzedziałowej).