Rozwiązywanie rówania nieliniowego metodą Newtona Raphsona.

Stanisław Fiedler

27 maja 2024

1 Zastosowanie

Funkcja `newton_raphson_i` znajduje przedział zawierający miejsce zerowe równania z zadaną dokładnością.

2 Opis metody

Kolejne przybliżenia wartości miejsca zerowego fukcji za pomocą metody Newtona Raphsona wyznaczane są wzorem:

$$x_{i+1} = x_i - \frac{f'(x_i) + \sqrt{[f'(x_i)]^2 - 2f(x_i)f''(x_i)}}{f''(x_i)}, i = 0, 1, 2, ...$$

gdzie x_0 jest dana. Proces kończy się kiedy szerokość przedziału zawierającego dwa ostatnie przybilżenia jest miejsza od zadanego na początku ε.

3 Wywołanie funkcji

```
newton_raphson_i(x, f, df, d2f ,mit, eps, fatx, it);
```

4 Dane

x - początkowe przybliżenie miejsca zerowego.

f - wskaźnik do funkcji zwracającej wartość $f(x)$.

df - wskaźnik do funkcji zwracającej wartość $f'(x)$.

d2f - wskaźnik do funkcji zwracającej wartość $f''(x)$.

mit - maksymalna liczba iteracji.

eps - błąd wyzaczanego miejsca zerowego.

5 Wynik

x - wyznaczone przybliżenie miejsca zerowego.

fatx - wartość funkcji dla wyznaczonego miejsca zerowego.

it - ilość wykonanych iteracji.

6 Inne parametry

Funkcja `newton_raphson_i` zwraca status wykonia. Poszczególne wartości oznaczają:

- 1 - mit < 1.
- 2 - dla pewnego x: $f''(x) = 0$.
- 3 - w mit iteracjach nie osiągnięto dokładości eps.
- 4 - dla pewnego x: $[f'(x)]^2 - 2f(x)f''(x) < 0$
- 0 - w innych przypadkach.

7 Typy parametrów

Interval : x, fatx;

Interval (*)(Interval) : f, df, d2f;

int : mit, it;

_Float128 : eps;

8 Identyfikatory nielokalne

Interval - klasa wprowadzająca arytmetykę przedziałową.

Interval (*)(Interval) - fukcja przyjmująca jako jedyny argument i zwracająca obiekt klasy Interval.

9 Tekst funkcji

```
int newton_raphson_i(Interval &x, Interval (*f)(Interval),
            Interval (*df)(Interval), Interval (*d2f)(Interval),
            const int mit, const _Float128 eps, Interval &fatx, int &it){
    int st;
    Interval dfatx, d2fatx, p, v, w, xh, x1, x2, ret;

    if(mit < 1){
        return 1;
    }
    st = 3;
    it = 0;
    do{
        it++;
        fatx = f(x);
        dfatx = df(x);
        d2fatx = d2f(x);
        p = dfatx*dfatx - 2*fatx*d2fatx;
        if(con_zero(d2fatx)){
            st = 2;
        }
        else if(neg(p)){
            st = 4;
        }
        else{
            xh = x;
            w = abs(xh);
            p = sqrt(p);
            x1 = x-(dfatx-p)/d2fatx;
            x2 = x-(dfatx+p)/d2fatx;
            if( make_intr(x2, xh).width() > make_intr(x1, xh).width() )
                x = x1;
            else
                x = x2;
            ret = make_intr(x, xh);
            if( ret.width() < eps ){
                st = 0;
            }
        }
    } while(it < mit and st == 3);
    if(st == 0 or st == 3){
        x = ret;
        fatx = f(x);
    }
    return st;
}
```

10 Przykłady

1. Równanie

$$f(x) = x^3 - 3x - 1$$

Dane:

x:	-0.5
mit:	10
eps:	1e-16

Wyniki:

x:	-3.47296355333860697703433253538629587230160763599116e-01
fx:	0.000e+00
it:	4
st:	0

2. Równanie

$$f(x) = x^3 - 3x - 1$$

Dane:

x:	[-0.5, -0.5]
mit:	10
eps:	1e-16

Wyniki:

x:	[-3.47296355333860697703433253782741063236497078931272e-01, -3.47296355333860697703433253538612783491395984966298e-01]
szerokość:	2.44128279745101093964974138431842897246009738490558e-28
fx:	[-8.83808288514330339349461749863251854013475577315218e-29, 7.32340639143079594832094780601746367656068766038846e-28]
szerokość:	8.20721467994512628767040955588071553057416323770368e-28
it:	4
st:	0

3. Równanie

$$f(x) = x^3 - 3x - 1$$

Dane:

x:	[-10, -10]
mit:	10
eps:	1e-16

Wyniki:

st:	4

dla pewnego x: $[f'(x)]^2 - 2f(x)f''(x) < 0$

Spis treści