Laboratorium przedmiotu Metody numeryczne

Sprawozdanie nr 3: Rozwiązywanie równań nieliniowych

Data: 11.04.2022

Ćwiczenie wykonał: Dawid Chmielewski 311188, grupa dziekańska 3

Ćwiczenie prowadził: mgr inż. Marek Wdowiak

Część 1. Implementacja dwóch metod rozwiązywania równań: bisekcji (w ramach zajęć) oraz Newtona-Raphsona.

Kod samych algorytmów zamieściłem na końcu sprawozdania. Poniżej zaś prezentuję odnalezione miejsca zerowe przez oba algorytmy- punkty nakładają się na wykresie. Tutaj za punkt startowy dla Newtona-Raphsona przyjąłem -1 (jest on przekazywany jako jeden z argumentów wywołania).

```
clc
clear
close all
format compact

f = @(x)(-3.55*x.^3 + 1.1*x.^2 + 0.765*x - 0.74);
df = @(x)(-10.65*x.^2 + 2.2*x + 0.765);

xLim = [-1 1];
x = xLim(1):1e-3:xLim(2);
y = f(x);

[x0, y0] = Bisection(f, xLim, eps, 30)
```

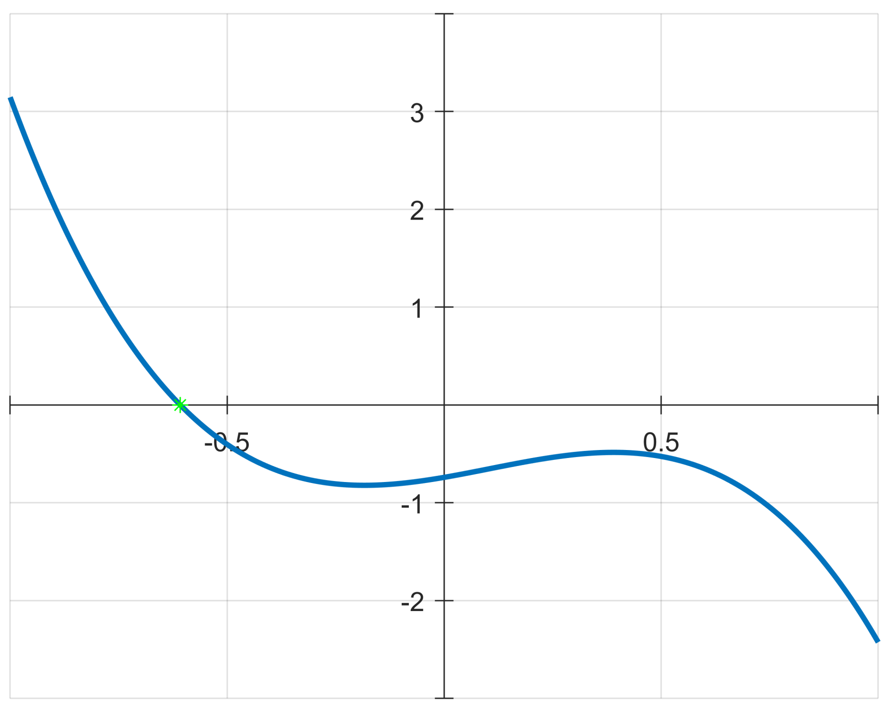
```
it = 30
x0 = -0.6081
y0 = 1×30
    0.7400    0.4037    0.8027    0.0783    0.1904    0.0633    0.0056    0.0293 ···
```

```
[x1, y1] = NewtonRaphson(f, df, xLim, eps, 30, -1)
```

```
it = 6
x1 = -0.6081
y1 = 1×6
    0.7332    0.1032    0.0035    0.0000    0.0000    0.0000
```

```
figure;
plot(x,y,'LineWidth',2);
hold on;
plot(x0,f(x0),'r*');
plot(x1,f(x1),'g*');
hAx = gca;
hAx.XAxisLocation = 'origin';
```

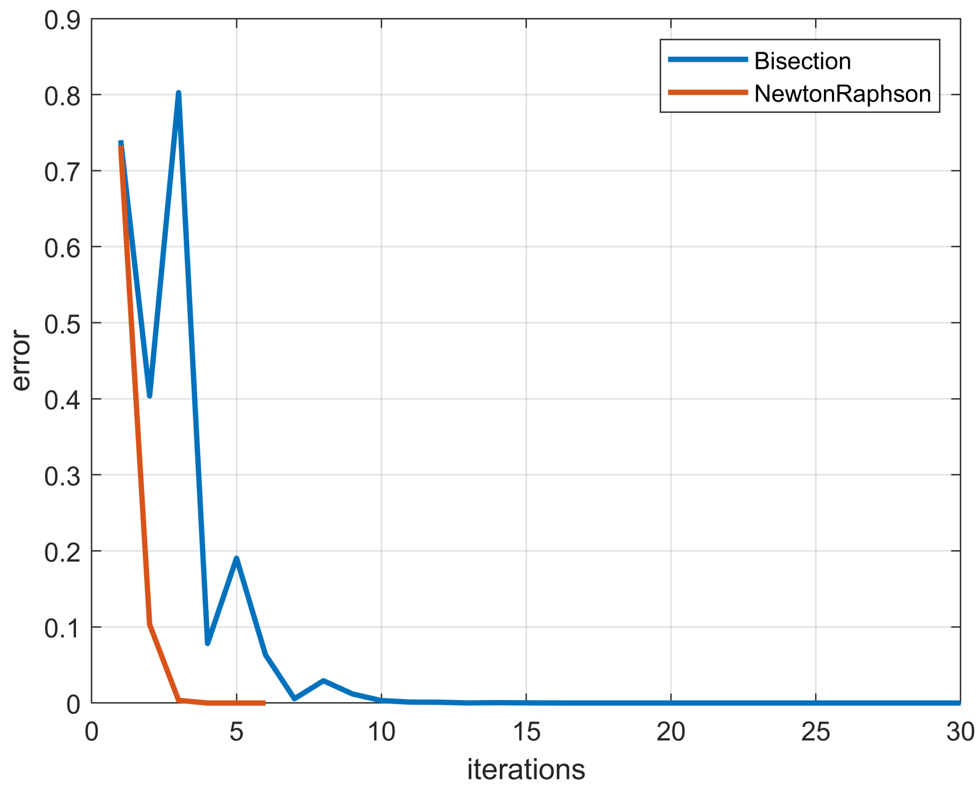
```
hAx.YAxisLocation = 'origin';
hAx.TickDir = 'both';
hAx.Box = 'off';
grid on;
hold off;
```




```
for k = 1:length(y0)
    xx(k) = k;
end

for k = 1:length(y1)
    xx1(k) = k;
end

figure;
plot(xx,y0,'LineWidth',2);
hold on;
plot(xx1,y1,'LineWidth',2);
legend('Bisection','NewtonRaphson');
xlabel('iterations');
ylabel('error');
grid on;
hold off;
```



Wnioski: Oba algorytmy odnalazły miejsce zerowe funkcji. Metoda Newtona-Raphsona szybciej znalazła miejsce zerowe od metody bisekcji, a wartości błędów są zdecydowanie dla niej niższe. Metoda bisekcji nie mogła znaleźć pierwiastka (dla błędu równego/mniejszego od eps) nawet po 30 iteracjach, podczas gdy Newton-Raphson skończył działanie po sześciu.

Część 2. Obserwacja, jak wybór punktu startowego włynie na wyszukiwanie miejsca zerowego wielomianu dla algorytmu Newtona-Raphsona.

```
[xr1, err1] = NewtonRaphson(f, df, xLim, eps, 50, -1/300)
```

```
it = 22
xr1 = -0.6081
err1 = 1×22
    2.2522    0.7917    0.4900   42.9690   12.7464    3.8407    1.2408    0.5351 ···
```

```
[xr2, err2] = NewtonRaphson(f, df, xLim, eps, 50, 0)
```

```
it = 50
xr2 = 0.8467
err2 = 1×50
    2.1839    0.7730    0.4931   23.1930    6.9140    2.1323    0.7589    0.4962 ···
```

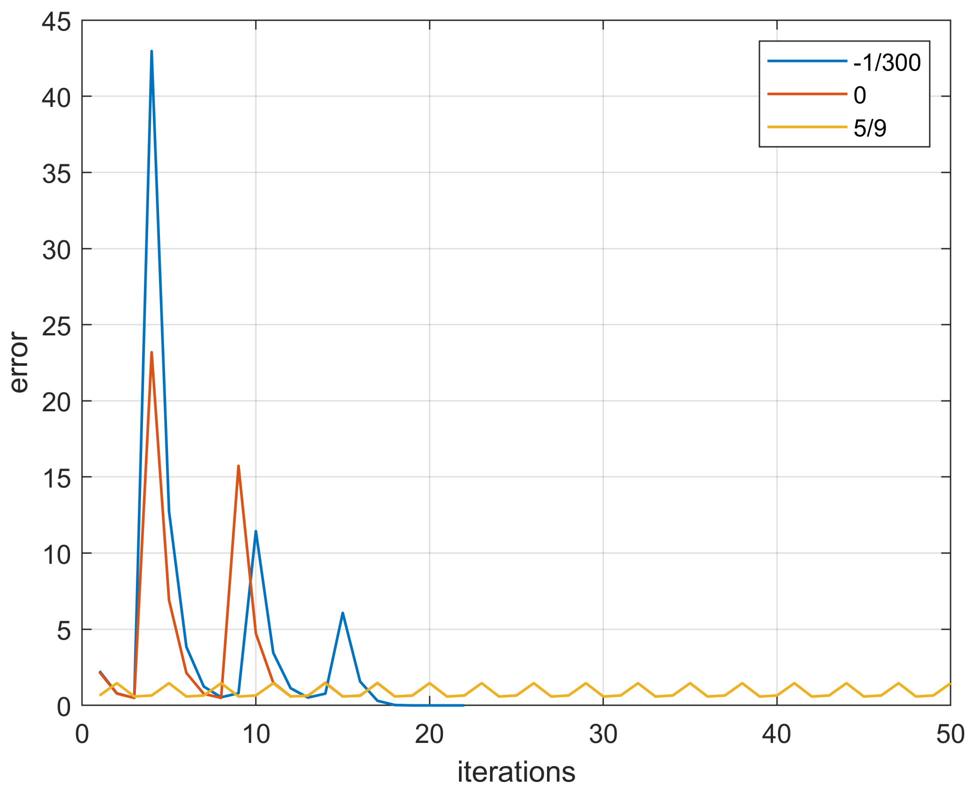
```
[xr3, err3] = NewtonRaphson(f, df, xLim, eps, 50, 5/9)
```

```
it = 50
xr3 = 0.8467
err3 = 1×50
    0.6507    1.4588    0.5841    0.6509    1.4588    0.5841    0.6510    1.4587 ···
```

```
for k = 1:length(err1)
    xi(k) = k;
end

for k = 1:length(err2)
    xxi(k) = k;
end

figure;
plot(xi,err1,'LineWidth',1);
hold on;
plot(xxi,err2,'LineWidth',1);
plot(xxi,err3,'LineWidth',1);
legend('-1/300','0','5/9');
xlabel('iterations');
ylabel('error');
grid on;
hold off;
```



Wnioski: funkcja Newtona-Raphsona miała poważne problemy ze znalezieniem miejsca zerowego dla punktów startowych o współrzędnych odciętych równych 0 oraz 5/9. Algorytm, korzystając z pochodnej badanej funkcji, nie radzi sobie w przypadku napotkania na swojej drodze ekstremów funkcji- zapętla się w pewnym przedziale, w którym usiłuje znaleźć miejsce zerowe (jest to na wykresie widoczne jako "ząbki" dla dalszych iteracji).

Całe ćwiczenie zostało wykonane w oparciu o poniższe funkcje:

```
function [x0, err] = Bisection(f, xLim, eps, n)
currentL = xLim(1);
currentR = xLim(2);

it = 0;
for k = 1:n
    x0 = (currentL + currentR) / 2;
    error = abs(f(x0));
    err(k) = error;
    if error <= eps
        it = k;
        break;
    end

    if (f(x0) > 0)
    %spr czy miejsce zerowe jest na prawo od x0...
        if (f(currentR) < 0)
            currentL = x0;
        elseif (f(currentL) < 0)
            currentR = x0;
        end

    %miejsce zerowe jest na lewo od x0...
    elseif (f(x0) < 0)
        if (f(currentR) > 0)
            currentL = x0;
        elseif (f(currentL) > 0)
            currentR = x0;
        end
    end
    it = k;
end

%it mowi po ilu iteracjach zatrzymal sie program.
it
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x0, err] = NewtonRaphson(f,df,xLim,eps,N,xStart)

%currentX = (xLim(1) + xLim(2)) / 2;
currentX = xStart;

it = 0;
for k = 1:N
    nextX = currentX - ( f(currentX) / df(currentX));
    error = abs(f(nextX));
    err(k) = error;
    if error <= eps
```

```
        it = k;
        break;
    end
    currentX = nextX;
    it = k;
end

x0 = currentX;
%it mowi po ilu iteracjach zatrzymal sie program...
it

end
```