20 May 2020

Emilia Wróblewska
Group 4

Program implementing the King method for finding a root of the polynomial

$$p_n(x) = c_0T_0(x) + c_1T_1(x) + ... + c_nT_n(x)$$

where $T_k(x)$ denotes the Chebyshev polynomial of the first order.

1 Method Description

In real world problems, cases where zeros of the function can be found in a finite number of arithmetic operations are not so common, that is why in order to get $\alpha \in \mathbb{R}$ such that $f(\alpha) = 0$ we use iterative methods to find a sequence of approximations x_{k} such that $x_{\mathrm{k}} \to \alpha$ as $k \to \infty$ starting from the initial guess x_0. Our iterative method will be the King's method, also known as the secant method, that uses a succession of roots of secant lines to better approximate a root of a function.

Starting with initial values $x_0, x_1 \in \mathbb{R}$, we construct a line through the points $(x_0, f(x_0))$ and $(x_1, f(x_1))$. In slope–intercept form, the equation of this line is:

$$y = \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_1) + f(x_1)$$

The root x of this function satisfies $y = 0$, so

$$x = x_1 - f(x_1)\frac{x_1 - x_0}{f(x_1) - f(x_0)}$$

Then we use this new value of x as x_2 and repeat the process, using x_1 and x_2 instead of x_0 and x_1. We continue this calculation until we reach a sufficiently high level of precision (a sufficiently small difference between x_n and $x_{n'1}$):

$$\begin{aligned}
x_2 &= x_1 - f(x_1)\frac{x_1 - x_0}{f(x_1) - f(x_0)} \\
x_3 &= x_2 - f(x_2)\frac{x_2 - x_1}{f(x_2) - f(x_1)} \\
&\vdots \\
x_n &= x_{n-1} - f(x_{n-1})\frac{x_{n-1} - x_{n-2}}{f(x_{n-1}) - f(x_{n-2})}
\end{aligned}$$

We obtain the final recurrence relation:

$$x_n = x_{n-1} - f(x_{n-1})\frac{x_{n-1} - x_{n-2}}{f(x_{n-1}) - f(x_{n-2})} = \frac{x_{n-2}f(x_{n-1}) - x_{n-1}f(x_{n-2})}{f(x_{n-1}) - f(x_{n-2})}$$

2 Program description

After You ran the program, You will see the Menu:

Menu
Change initial guess x_0
Change initial guess x_1
Change degree n of polynomial
Change tolerance
Change maximum number of iterations
Display variables
Find root
Plot function
FINISH

- Pressing any „Change" button will change the value of corresponding variable.
- Option „Display variables" will simply display previously input or default variables.
- After clicking „Find root" the program will calculate the root of the function using King's method and output its value as well as number of performed iterations.
- Option „Plot function" will display the graph of the function in range $[-1, 1]$
- At last „FINISH" will end the program.

MATLAB functions used:

1. Menu_King.m - script for graphic interface for the rest of the functions.
2. king_method.m - function used for computing the approximation of root for the Chebyshev polynomial of the first kind

$$p_n(x) = c_0T_0(x) + c_1T_1(x) + ... + c_nT_n(x)$$

using formulas described at the beginning.

(**Note.** *All source codes can be found in section 5.*)

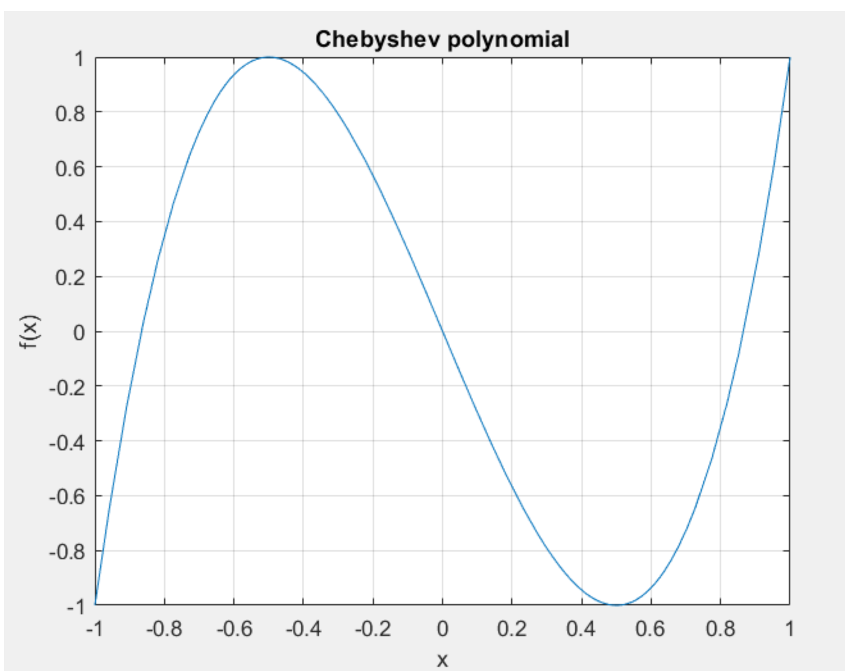
3 Examples

Here are couple of exemplary results - both simple and more interesting ones.
x_0 and x_1 are the initial guesses, n is the current degree of the Chebyshev Polynomial and max_iter and $tolerance$ are number of maximum iterations and maximum error respectively. Then x is a final answer obtained after i iterations of using $king\_method$ function and $p(x)$ is a value of the function for previously calculated x.

First Example

x_0	x_1	n	max_iter	tolerance
-20	-10	3	100	e^{-10}

$$p(x) = 4x^3 - 3x$$

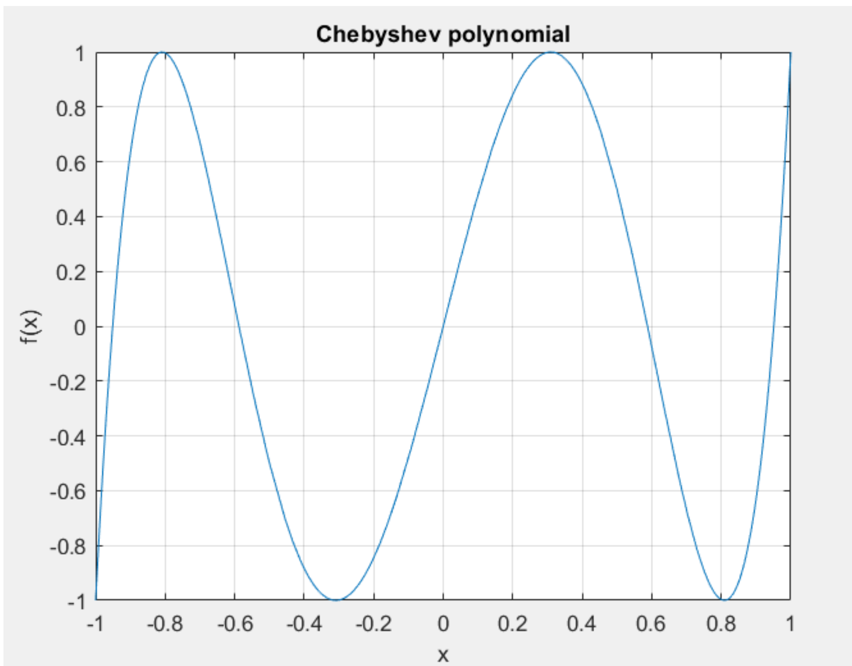


x	i	$p(x)$
-0.8660254038	17	$-2.7456e^{-24}$

Second Example

x_0	x_1	n	max_iter	tolerance
0	1	5	100	e^{-10}

$$p(x) = 16x^5 - 20x^3 + 5x$$

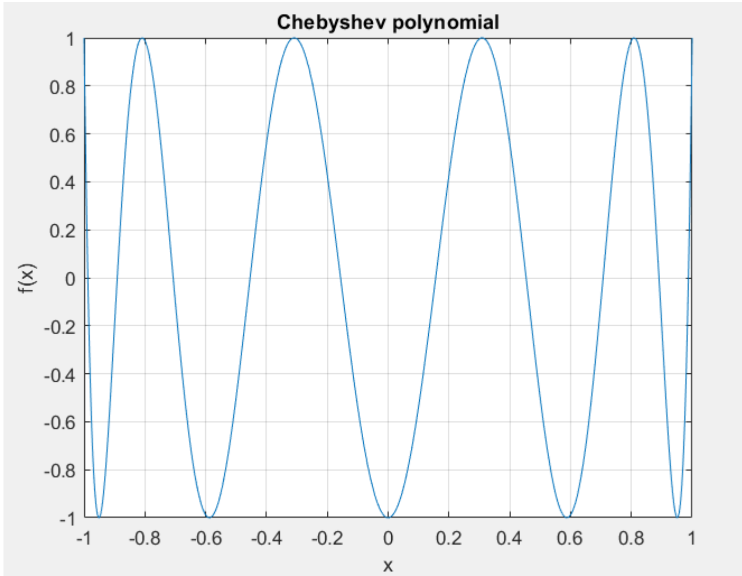


x	i	$p(x)$
0.0	2	0.0

Third Example

x_0	x_1	n	max_iter	tolerance
4	5	10	100	e^{-10}

$$p(x) = 512x^{10} - 1280x^8 + 1120x^6 - 400x^4 + 50x^2 - 1$$

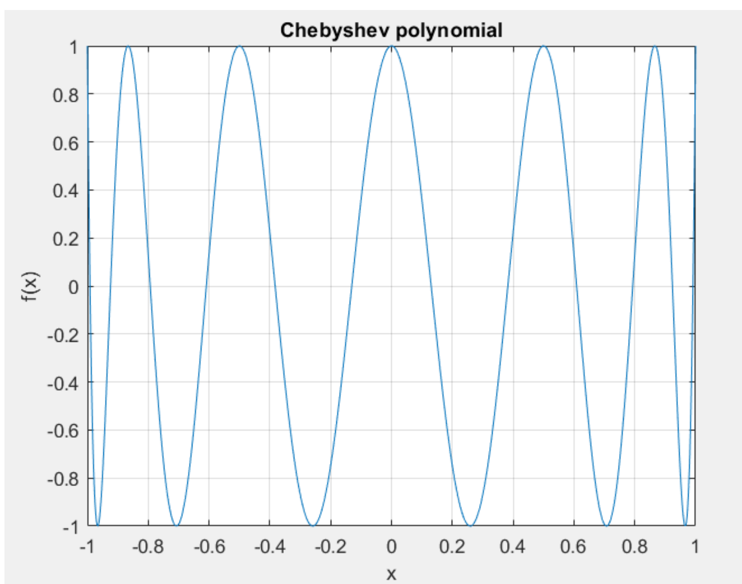


x	i	$p(x)$
0.9876883406	34	$2.0359e^{-18}$

Fourth Example

x_0	x_1	n	max_iter	tolerance
1	2	12	100	e^{-10}

$$p(x) = 2048x^{12} - 6144x^{10} + 6912x^8 - 3584x^6 + 840x^4 - 72x^2 + 1$$

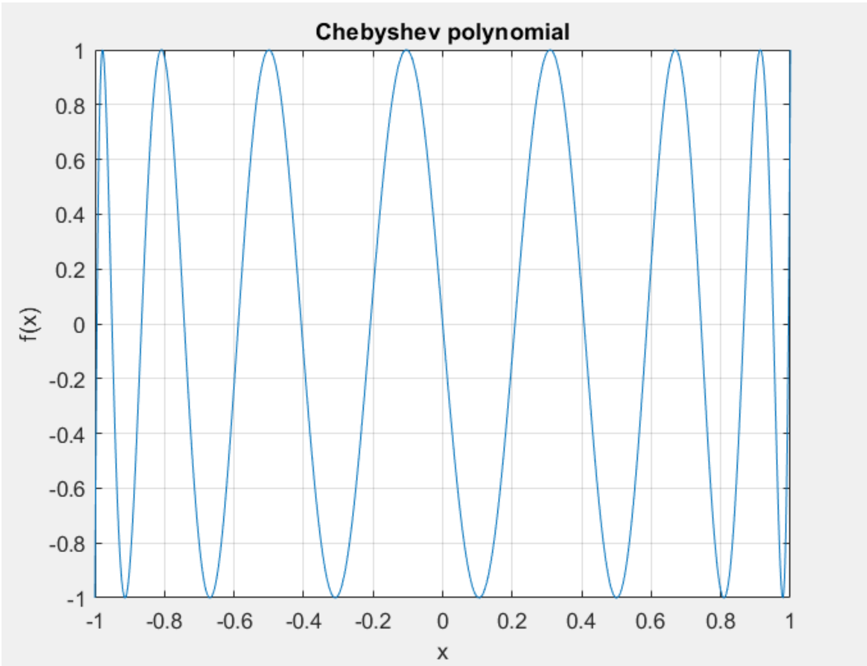


x	i	$p(x)$
0.9914448614	8	$2.550482e^{-14}$

Fifth Example

x_0	x_1	n	max_iter	tolerance
7	8	15	100	e^{-10}

$$p(x) = 16384x^{15} - 61440x^{13} + 92160x^{11} - 70400x^{9} + 28800x^{7} - 6048x^{5} + 560x^{3} - 15x$$



x	i	$p(x)$
0.9945218954	60	$5.77515e^{-15}$

4 Conclusions

We can observe that the King's method is not that efficient since the number of steps needed to find the root (accepted within specified range of error) depends on initial guesses x_0 and x_1. The farther from the actual root our guesses are, the more steps it takes to find it. However, Chebyshev polynomials are special and their roots are always relatively close to 0, so choosing right x_0 and x_1 is very easy and guarantees higher efficiency of our program. Just like in second and fourth example - when our guesses were close to 0, the number of iterations did not exceed 10.
We can also observe that this method is very accurate since even for the most complicated example - which is the last one with 60 iterations - we obtained a result for which an error was in 10^{-14} range. Hence We can come to conclusion that the King's method is a good way for finding the root of a function provided we can properly establish our initial guesses. Otherwise it can appear slow and inefficient.

5 Source Codes

Menu_King.m:

```
% MENU
clear
clc
finish=9;
control=1;

%default data %
syms x;
x0 = 0;
x1 = 1;
n = 5;
tol = 1e-10;
max_iter = 100;

while control~=finish

    control=menu('Menu', 'Change initial guess x0',...
                 'Change initial guess x1',...
                 'Change degree n of polynomial',...
                 'Change tolerance',...
                 'Change maximum number of iterations',...
                 'Display variables', 'Find root',...
                 'Plot function', 'FINISH');

    switch control
        case 1
            x0=input('x0=');

        case 2
            x1=input('x1=');

        case 3
            n=input('n=');

        case 4
            tol=input('tol=');

        case 5
            max_iter=input('max_iter=');
```

```
        case 6
            disp('---- Variables ----');
            disp('x0 =');disp(x0)
            disp('x1 =');disp(x1)
            disp('Degree n =');disp(n)
            disp('tolerance =');disp(tol)
            disp('max_iteration =');disp(max_iter)

        case 7
             px=chebyshevT(n,x);
            [root, i] = king_method(n,x0,x1,tol,max_iter);
            p0=vpa(subs(px,x,root));
            disp('---- Finding root ----');
            disp('Chebyshev Polynomial p(x) =');disp(px)
            disp('Root x =');disp(root)
            disp('i =');disp(i)
            disp('Value at p(x) =');disp(p0)

        case 8
            f=@(x) chebyshevT(n,x);
            fplot(f,[-1,1])
            ylim([-1 1])
            grid on
            xlabel('x')
            ylabel('f(x)')
            title('Chebyshev polynomial')

        case 9
            disp('FINISH')
            close
    end
end
```

king_method.m:

```
function [root, i] = king_method(n,a,b,maxerr,iterations)

x0 = a;
x1 = b;
syms x;
f=chebyshevT(n,x);
%disp('f = '); disp(f);
```

```
for i=1:iterations

    f0=vpa(subs(f,x,x0)); %value f(x0)
    f1=vpa(subs(f,x,x1)); %value f(x1)
    if (f1-f0)==0
        disp(' You divide by zero! ');
        return;
    end
    root=x1-((x1-x0)/(f1-f0))*f1;
    err=abs(root-x1);

    if err<maxerr
        break
    end
    x0=x1;
    x1=root;
end

end
```