Task 20
Implement the Homeier method for finding a root of the polynomial $p(x) = c_0U_0(x) + c_1U_1(x) + ... + c_nU_n(x)$, where $U_k(x)$ denotes the Chebyshev polynomial of the second order

Maciej Sobczynski

June 8, 2018

1 Method description

Homeier's method is a modified version of Newton's method for finding roots of polynomials so to understand it, we need to understand the Newton's method first.

1.1 Newton's method

The main idea of this method is to start with an initial guess for a root and then approximate the function by its tangent line. The next point is chosen at the intersection of the tangent and the x-axis. This point is the next (typically better) approximation of the function's root. This method can be iterated in the following fashion:
Let f be a differentiable function with $x \in \mathbb{R}$ and $f(x) \in \mathbb{R}$. Suppose we already obtained current approximation x_n. Then the next approximation x_{n+1} can be found. The tangent line to $y = f(x)$ at point $x = x_n$ is

$$y = f'(x_n)(x - x_n) + f(x_n).$$

Next, the x-intercept of the tangent is used to find the approximation of the root x_{n+1}. We obtain it by solving the previous equation with $y = 0$ and $x = x_{n+1}$ for x_{n+1} which gives

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

1.2 Hoimier's method

This method is very similar to the previously explained Newton's method. The only key difference is that in this method

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

where

$$y_n = x_n - \frac{f(x_n)}{f'(x_n)}$$

1.3 Chebyshev polynomials

The Chebyshev polynomials are a sequence of polynomials related to the trigonometric multi-angle formulae. One usually distinguishes between Chebyshev polynomials of the first kind which are denoted T_n and Chebyshev polynomials of the second kind which are denoted U_n.

The Chebyshev polynomials of the second kind are defined by the recurrence relation

$$U_0(x) = 1$$

$$U_1(x) = 2x$$

$$U_{n+1}(x) = 2xU_n(x) - U_{n-1}(x)$$

The ordinary generating function for U_n is

$$\sum_{n=0}^{\infty} U_n(x)t^n = \frac{1}{1 - 2tx + t^2}$$

and the exponential generating function is

$$\sum_{n=0}^{\infty} Y_n(x)\frac{t^n}{n!} = e^{tx}(\cosh(t\sqrt{x^2 - 1}) + \frac{x}{\sqrt{x^2 - 1}} \sinh(t\sqrt{x^2 - 1})).$$

The first few Chebyshev polynomials of the second kind are

$$U_0(x) = 1$$

$$U_1(x) = 2x$$
$$U_2(x) = 4x^2 - 1$$
$$U_3(x) = 8x^3 - 4x$$
$$U_4(x) = 16x^4 - 12x^2 + 1$$
$$U_5(x) = 32x^5 - 32x^3 + 6x$$
$$U_6(x) = 64x^6 - 80x^4 + 24x^2 - 1$$

2 Program description

2.1 Homeier.m

A function to implement the Homeier's method. It takes four arguments: f - an input function, x0 - initial approximation/guess, tol - tolerance and max - maximal number of iterations. The function returns the approximation of the root and number of iterations.

2.2 Cheby.m

A function to calculate the value of the polynomial$p(x) = c_0U_0(x)+c_1U_1(x)+ ... + c_nU_n(x)$ and it's derivative. As an input argument, the function takes a vector of coefficients that represents $c_0, c_1, ..., c_n$ and value x at which the polynomial is evaluated. The function returns two values - p -value of the polynomial and p2 - value of it's derivative)

2.3 Scripts

The scripts contain the numerical examples presented in the next section

3 Numerical examples

In the examples the function will be graphed and the result of the Homeier method x_0 and number of iterations k will be presented as well as $diff$ - the value of the polynomial at x_0

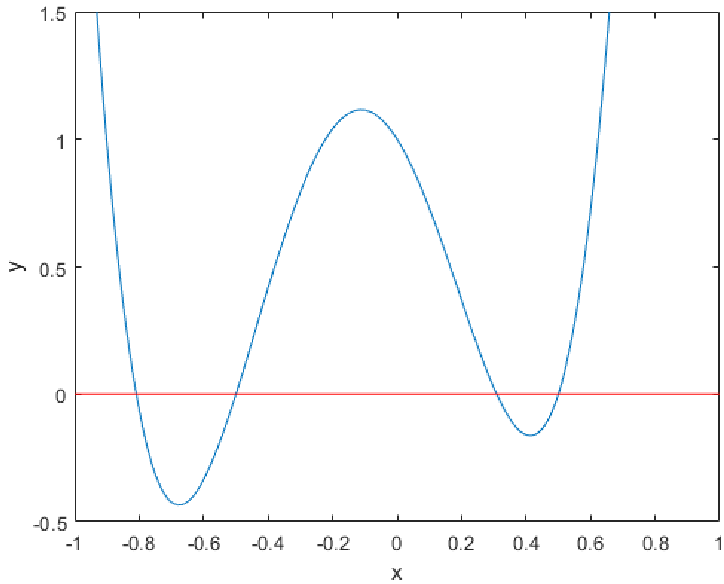

Figure 1: Example 1 graph

3.1 Example 1

x0 = 0.3090, k = 5, diff = -2.2204e-16

```
%16*x^4 + 8*x^3 - 8*x^2 - 2*x + 1
x=linspace(-1,1);
p=[0,0,0,1];
for i = 1:100
    t=x(i);
    [y(i),~]=Cheby(p,t);
end
plot(x,y);
%plot(t,subs(fr,x,t));
hold on;
hline = refline(0);
hline.Color = 'r';
axis([-1 1 -2.5 2.5])
%axis('normal')
xlabel(' x ')
```

```
ylabel(' y ')
hold off;
[x0,k]=Homeier(p,1,0.0001,100)
diff=polyval(f,x0)
print -deps ex1
```

3.2 Example 2

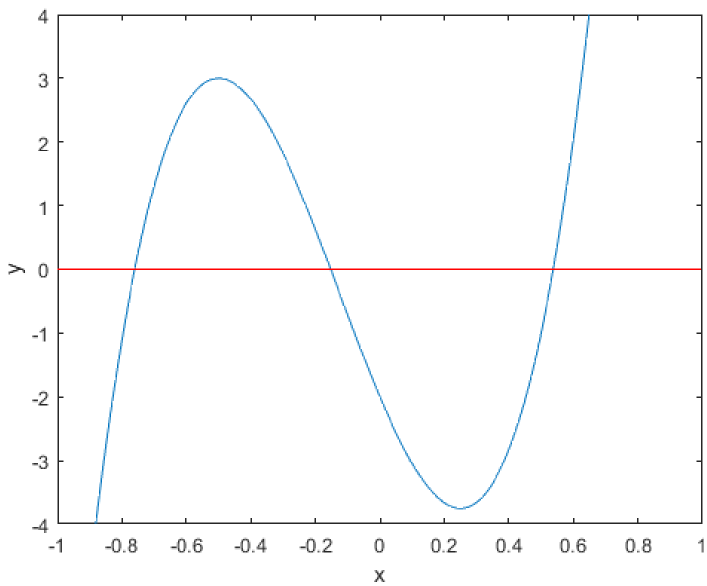

Figure 2: Example 2 graph

x0 = -0.1528, k = 3, diff = 4.3350e-06

```
%32*x^3 + 12*x^2 - 12*x - 2
x=linspace(-1,1);
p=[1,2,3,4];
for i = 1:100
    t=x(i);
    [y(i),~]=Cheby(p,t);
end
plot(x,y);
```

```
hold on;
hline = refline(0);
hline.Color = 'r';
axis([-1 1 -4 4])
%axis('normal')
xlabel(' x ')
ylabel(' y ')
hold off;
[x0,k]=Homeier(p,0.2,0.01,100)
diff=polyval(f,x0)
print -deps ex2
```

3.3 Example 3

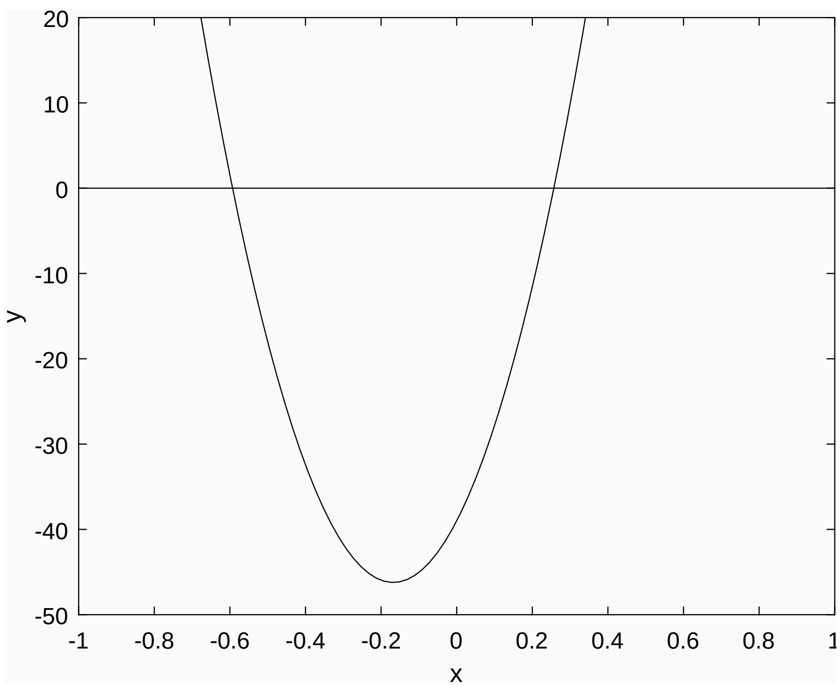

Figure 3: Example 3 graph

x0 = 0.2570, k = 3, diff = 0

```
%32*x^3 + 12*x^2 - 12*x - 2
```

```
x=linspace(-1,1);
p=[25,43,64];
for i = 1:100
    t=x(i);
    [y(i),~]=Cheby(p,t);
end
plot(x,y);
hold on;
hline = refline(0);
hline.Color = 'r';
axis([-1 1 -50 20])
%axis('normal')
xlabel(' x ')
ylabel(' y ')
hold off;
[x0,k]=Homeier(p,0,0.001,100)
diff=polyval(f,x0)
print -deps ex3
```

3.4 Example 4

x0 = 0.8464, k = 9, diff = 2.2204e-15

```
%- 32*x^5 - 80*x^4 + 64*x^3 + 48*x^2 - 18*x - 3
x=linspace(-3,2.5);
p=[-1,2,-3,4,-5,6-7];
for i = 1:100
    t=x(i);
    [y(i),~]=Cheby(p,t);
end
plot(x,y);
hold on;
hline = refline(0);
hline.Color = 'r';
axis([-3 2.5 -700 400])
%axis('normal')
xlabel(' x ')
ylabel(' y ')
```

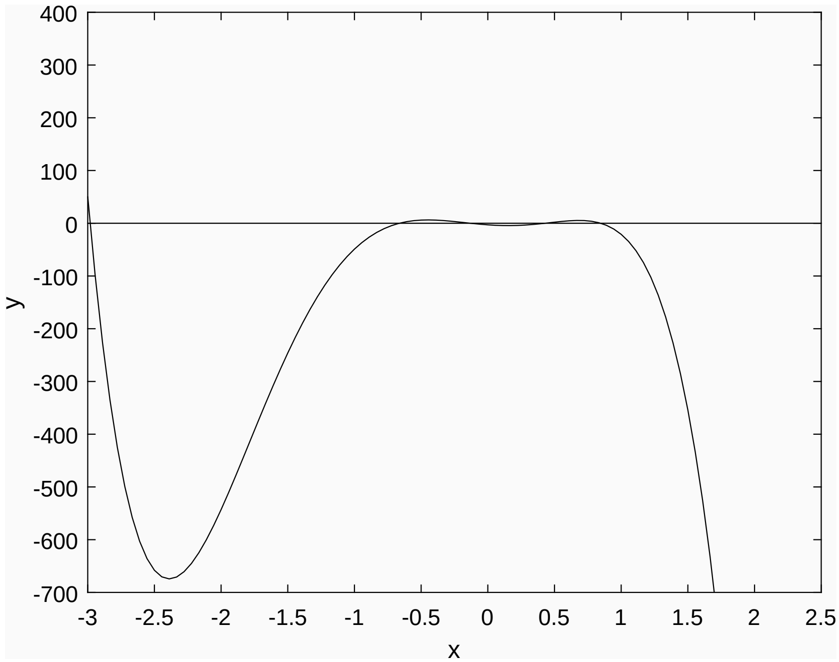

Figure 4: Example 4 graph

```
hold off;
[x0,k]=Homeier(p,10,0.0001,100)
diff=polyval(f,x0)
print -deps ex4
```

4 Analysis of results

Overall the Homeier method provides an extremely fast way of of finding the roots of a polynomial. The method is quite precise and fast - the number of iterations is usually very low.

5 Code

5.1 Homeier.m

```
function [x,k] = Homeier(a,x0,tol,max)
```

```
% Homeier's Method
%   Homeier's method for finding successively better approximations to
%   zeroes of a real-valued function.
%
% Input:
%   f - input funtion
%   x0 - inicial aproximation
%   tol - tolerance
%   max - maximum number of iterations
%
% Output:
%   x - aproximation to root
%   k - number of iterations
%
% Example:
%         [ x, ex ] = newton( 'exp(x)+x', 0, 0.5*10^-5, 10 )
dy=tol+1;
k=0;
xn=x0;
while abs(dy)> tol && k<= max
    [w,dw]=Cheby(a,xn);
    if dw==0
        disp('Division by 0');
        return;
    end
    dy=w/dw;
    yn=xn-dy;
    [~,dwy]=Cheby(a, yn);
    if dwy==0
        disp('Division by 0');
        return;
    end
    x=xn-1/2*w*(1/dw+1/dwy);
    k=k+1;
    xn=x;
end
end
```

5.2 Cheby.m

```
function [p,p2] = Cheby(a,x)
% Value of Polynomial Using Chebyshev Polynomials
%   Produces a value of the polynomial of form p(x)=c0*U0(x)+c1*U1(x)+.
%   where Uk is Chebyshev polynomial of the second kind
%
% Input:
%   a - vector of cooefictients
%   x - the x value for which the value of polynomial is calcualted
%
% Output:
%   p - value of the polynomial
%   p2 - value of the derivative of the polynomial
n=length(a)-1;
u=zeros(1,n);
u(1)=2*x;
u(2)=4*x^2-1;
for i=3:n
    u(i)=2*x*u(i-1)-u(i-2);
end
p=a(1);
for i=2:n+1
    p=p+a(i)*u(i-1);
end

%derivative
n=length(u);
w=zeros(1,n);
w(1)=2;
w(2)=8*x;
for i=3:n
    w(i)=2*x*w(i-1)-w(i-2)+2*u(i-1);
end
p2=0;
for i=2:n+1
    p2=p2+a(i)*w(i-1);
end
```

end