
Numerical Analysis and Computational Mathematics

Fall Semester 2024 – CSE Section

Prof. Laura Grigori

Assistant: Israa Fakih

Session 2 – September 18, 2024

Solutions – Nonlinear equations: bisection and Newton methods

Solution I (MATLAB)

a) We consider the following implementation of the MATLAB function `bisection.m`:

```
function [xvect,esterrvect,resvect,nit] = bisection(fun,a,b,tol,nmax)
% BISECTION Find a zero of a nonlinear scalar function inside an interval.
%   XVECT=BISECTION(FUN,A,B,TOL,NMAX) finds a zero of the continuous
%   function FUN in the interval [A,B] using the bisection method and returns
%   a vector XVECT containing the successive approximations of the zero (iterates).
%   FUN accepts real scalar input x and returns a real scalar value;
%   FUN can also be an inline object.
%   TOL is the tolerance on error allowed and NMAX the maximum number of iterations.
%   If the search fails an error message is displayed.
%
%   [XVECT,ESTERRVECT,RESVECT,NIT]=BISECTION(FUN,...) also returns the vector
%   ESTERRVECT of error estimators for each iterate, the vector RESVECT of residual
%   evaluations for each iterate, and NIT the number of iterations.
%   Note: the length of the vectors is equal to ( NIT + 1 ).
%
if a >= b
    error(' b must be greater than a (b > a)');
end

% evaluate f at the endpoints
fa = fun(a);
fb = fun(b);
if sign(fa) * sign(fb) > 0
    error(' The sign of FUN at the extrema of the interval must be different');
end

if fa == 0 % a is the solution
    xvect = a; fx = 0; esterr = 0; nit = 0;
    resvect = fx; esterrvect = esterr;
```

```

    return
elseif fb == 0 % b is the solution
    xvect = b; fx = 0; esterr = 0; nit = 0;
    resvect = fx; esterrvect = esterr;
    return
end

nit = 0;
xvect = []; resvect = []; esterrvect = [];

% initial approximate solution
x = (a + b) / 2;
% initial error estimator is the half of the length of the interval
esterr = (b - a) / 2;
fx = fun(x);
xvect = x;
resvect = fx;
esterrvect = esterr;

% loop until convergence or maximum number of iterations reached
while esterr ≥ tol && nit < nmax

    if fx == 0 % we found the solution
        return;
    end
    if sign(fx) * sign(fa) < 0 % alpha is in (a,x)
        b = x;
    elseif sign(fx) * sign(fb) < 0 % alpha is in (x,b)
        a = x;
    else
        error('Algorithm not operating correctly');
    end
    % calculate mid-point of updated interval
    x = (a + b) / 2;
    % the error estimator is now half of the previous one
    esterr = esterr / 2;
    fx = fun(x);
    xvect = [xvect, x];
    resvect = [resvect, fx];
    esterrvect = [esterrvect, esterr];
    nit = nit + 1;

end

if esterrvect(end) > tol
    warning(['bisection stopped without converging to the desired tolerance ',...
            'because the maximum number of iterations was reached']);
end

return

```

We solve the nonlinear equation $f(x)$ by considering the following commands, taking $tol = 10^{-1}$ as tolerance:

```

fun = @(x) sin(2*x) - 1 + x;
a = -1; b = 3; tol = 1e-1; nmax = 100;
[xvect,esterrvect,resvect,nit] = bisection(fun,a,b,tol,nmax);

```

```

x_nit = xvect( nit+1 ) % last iterate (approximated zero)
nit
% x_nit =
%
%      0.3125
%
% nit =
%
%      5

```

We observe that the bisection method converges in $n = 5$ iterations to the approximate zero $x^{(5)} = 0.3125$.

- b) If we decrease the tolerance to the suggested values ($tol = [10^{-2}, 10^{-3}, 10^{-4}]$), we obtain the approximate solutions $[x^{(8)} = 0.3516, x^{(11)} = 0.3525, x^{(15)} = 0.3522]$, respectively (the numbers in bracket represent again the iterations needed by the algorithm to converge).
- c) Note that, above, we have made the algorithm more robust by adding several checks on the input arguments and on some edge cases, and by displaying warning/error messages that can help the user if something goes wrong.

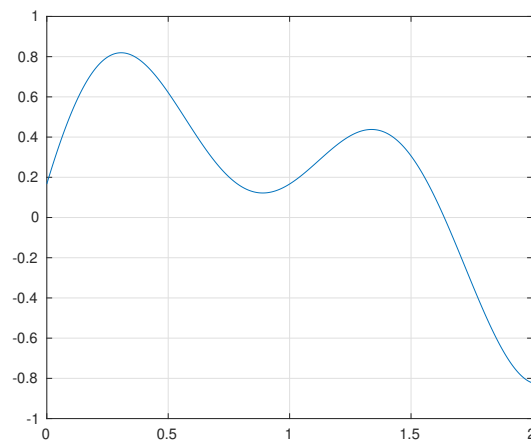
Solution II (Theoretical and MATLAB)

- a) We consider the following MATLAB commands to plot the function $f(x)$:

```

fun = @(x) (1-x) .* sin(4*x) + 1/6;
a = 0; b = 2;
xv = linspace(a,b,1001); % xv=[a:(b-a)/1000:b]
plot( xv, fun(xv) ); grid on

```



The conditions that need to be satisfied in order to apply the bisection method are the continuity of $f(x)$ in $[a, b]$ and $f(a)f(b) < 0$ (i.e. f must change its sign over (a, b)). In our example, these conditions are satisfied, since $f(x) \in C^\infty([0, 2])$, $f(0) = 1/6 > 0$, and $f(2) = -\sin(8) + 1/6 < 0$. The conditions imply that there exists at least one zero in the

interval $(0, 2)$. By referring to the previous plot, we deduce that the zero $\alpha \in (0, 2)$ is also unique.

- b) The error estimator (error bound) at the n th iteration of the bisection method is $\tilde{e}^{(n)} = (b - a)/2^{n+1}$, so that:

$$e^{(n)} = |x^{(n)} - \alpha| \leq \tilde{e}^{(n)} = \frac{b - a}{2^{n+1}}, \quad n = 0, 1, \dots, \infty.$$

Thus, the stopping criterion based on the error estimator implies that n needs to satisfy $\tilde{e}^{(n)} < \epsilon$. Taking the logarithm on both sides yields

$$n_{min} > \log_2 \left(\frac{b - a}{\epsilon} \right) - 1 = \frac{\log(b - a) - \log(\epsilon)}{\log 2} - 1.$$

We obtain the condition $n_{min} > 19.9316$, so $n_{min} = 20$ iterations will suffice.

- c) We consider the following MATLAB commands:

```
tol = 1e-6; nmax = 100;
[xvect, esterrvect, resvect, nit] = bisection(fun, a, b, tol, nmax);
% NOTE: the first entry of the vector xvect contains the initial guess of
% the zero obtained for n=0, i.e. the mid point of the interval [a,b]
resvect( 19 + 1 )
resvect( 20 + 1 )
% ans =
%
% -1.3514e-06
%
% ans =
%
% 1.2434e-06
```

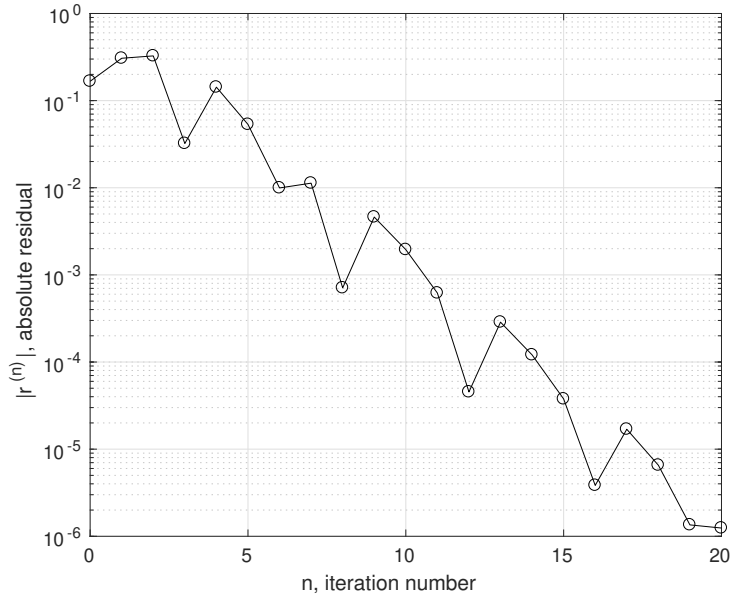
Therefore, we have $r^{(19)} = -1.3514 \cdot 10^{-6}$ and $r^{(20)} = 1.2434 \cdot 10^{-6}$.

- d) The residuals of the bisection method are not converging monotonically, and neither are the errors. We verify this property for the function $f(x)$ by plotting the absolute residuals $|r^{(n)}|$ vs. n with the following MATLAB commands:

```
tol = 1e-6; nmax = 100;
[xvect, esterrvect, resvect, nit] = bisection(fun, a, b, tol, nmax);
% NOTE: the first entry of the vector xvect contains the initial guess of
% the zero obtained for n=0, i.e. the mid point of the interval [a,b]
resvect( 19 + 1 )
resvect( 20 + 1 )
nvect = 0 : nit;
resvect_abs = abs( resvect );
semilogy( nvect, resvect_abs, '-ok' ); grid on
```

- e) We consider the following MATLAB commands to obtain the plot representing the sequence a_n vs. the number of iterations n .

```
alpha = xvect( 20 + 1 ); % approximation of exact zero (alpha)
```



```

nv = [ 0 : 18 ];
nv_ind = nv + 1; % Matlab indexes start from 1
err_n_plus_1 = abs( xvect( nv_ind + 1 ) - alpha );
err_n = abs( xvect( nv_ind ) - alpha );
a_n = err_n_plus_1 ./ err_n;
plot( nv, a_n, '--k*' ); grid on

```

As with the residuals, the convergence of the error $e^{(n)} = |x^{(n)} - \alpha|$ is not monotonic wrt n in this case, see also point f), Subfigure (a). Therefore, a coefficient μ representing the asymptotic convergence factor cannot be deduced from the sequence a_n , and we cannot infer that the convergence is linear. However, note that this does not mean that the method does not converge.

- f) Since $\tilde{e}^{(n)} = (b-a)/2^{n+1}$ and $\tilde{e}^{(n+1)} = (b-a)/2^{n+2}$, it follows that $\tilde{a}_n := \tilde{e}^{(n+1)}/\tilde{e}^{(n)} = \nu = 1/2$ for all $n \geq 0$, and the sequence of the error estimators $\{\tilde{e}^{(n)}\}_{n=1}^{\infty}$ converges linearly by definition (the order of convergence is 1). We obtain the plot of the sequence $\tilde{a}_n$ vs. n in Subfigure (b) by using the following commands:

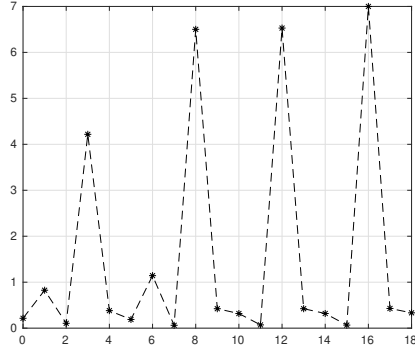
```

esterr_n_plus_1 = ( b - a ) ./ 2.^( nv + 1 );
esterr_n = ( b - a ) ./ 2.^( nv );
a_tilde_n = esterr_n_plus_1 ./ esterr_n;
plot( nv, a_tilde_n, '--k*' ); grid on

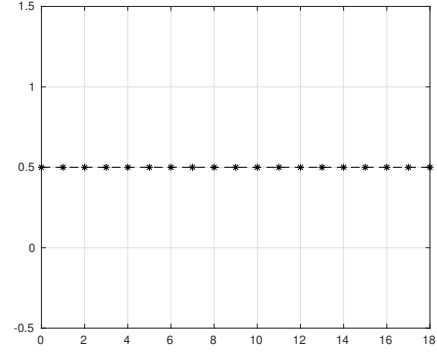
```

We verify graphically that $\nu = 1/2 < 1$. Even if we cannot establish the convergence order of the bisection method according to definition (1) (i.e. for the error $e^{(n)} = |x^{(n)} - \alpha|$), we observe that the error estimators (bounds) $\{\tilde{e}^{(n)}\}_{n=1}^{\infty}$ represent a dominating sequence of the errors $\{e^{(n)}\}_{n=1}^{\infty}$, i.e.:

$$e^{(n)} = |x^{(n)} - \alpha| \leq \tilde{e}^{(n)}, \quad n = 0, 1, \dots, \infty.$$



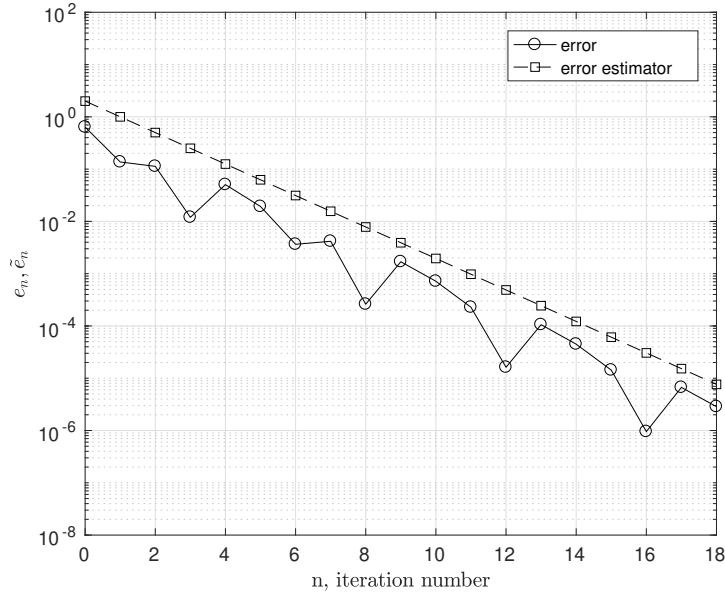
(a) a_n vs. n



(b) $\tilde{a}_n$ vs. n

In this case, the behavior of the error resembles that of the estimator, which converges with order 1. We verify this by plotting the errors $e^{(n)}$ and error estimators $\tilde{e}^{(n)}$ vs. n with the following commands:

```
semilogy( nv, err_n, '-ko', nv, esterr_n, '--ks' ); grid on
```

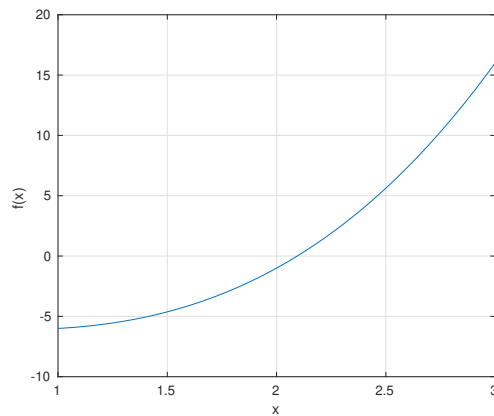


We remark again that the method does not have order 1 because the error is not monotonically decreasing.

Solution III (Theoretical and MATLAB)

- a) Since $f(x) \in C^0([1, 3])$ and $f(1)f(3) < 0$, there exists at least one zero $\alpha \in (1, 3)$. By studying the function $f(x)$ in $(1, 3)$ we deduce that the zero α is also unique since $f(1) < 0$ and the

function is strictly increasing; indeed, the first derivative of $f(x)$ is strictly positive in the interval: $f'(x) = 3x^2 - 2 > 0$ for all $x \in [1, 3]$.



b) The Newton method reads:

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})} = x^{(n)} - \frac{(x^{(n)})^3 - 2x^{(n)} - 5}{3(x^{(n)})^2 - 2} \quad \text{for } n = 0, 1, 2, \dots,$$

until a stopping criterion is satisfied, provided that $f'(x^{(n)}) \neq 0$ for all $n = 0, 1, 2, \dots$

c) We use MATLAB to determine the first three approximate zeros. We consider the following commands:

```
fun = @(x) x.^3 - 2*x - 5;
dfun = @(x) 3*x.^2 - 2;
newton_iterate = @(xn) xn - fun( xn ) / dfun( xn );
x0 = 1.5;
x1 = newton_iterate( x0 )
x2 = newton_iterate( x1 )
x3 = newton_iterate( x2 )
% x1 =
%
%      2.4737
%
% x2 =
%
%      2.1564
%
% x3 =
%
%      2.0966
```

We deduce that the first three approximate zeros, starting from $x^{(0)} = 1.5$, are $x^{(1)} = 2.4737$, $x^{(2)} = 2.1564$ and $x^{(3)} = 2.0966$. Note that $\alpha \simeq 2.0946$.