Math 128A: Programming Assignment 1

Michael Pham

Summer 2024

Problems

Problem 1.1. Implement a MATLAB function `findzero` which implements the following variant of the bisection and the secant method:

- Initialize w = 1
- Iterate for at most 100 times:
  1. Compute $p = a + \frac{wf(a)(a-b)}{f(b)-wf(a)}$
  2. Output $a, b, p, f(p)$ using `fprintf`
  3. If $f(p)f(b) > 0$, set $w = 1/2$, otherwise set $w = 1$ and $a = b$
  4. Set $b = p$
  5. Terminate if $|b - a| < \mathrm{tol}$ or if $|f(p)| < \mathrm{tol}$

Solution. The code for my findzero() function is as follows:

Code: Code for findzero()

```
function p = findzero(f, a, b, tol)
% Initializing w
w = 1;

fprintf(' n         a             b             p            f(p)    \n');

for n = 1:100
        % Calculations
        num = w*f(a)*(a-b);
        denom = f(b)-(w * f(a));
        p = a + (num/denom);

        % Print table
        fprintf('----------------------------------------------------------\n');
        fprintf('%2d  %12.8f  %12.8f  %12.8f  %12.8f\n', n, a, b, p, f(p));

        % Reassignments
        if f(p)*f(b) > 0
                w = 1/2;
        else
                w = 1;
                a = b;
        end
        b = p;

        % End condition
        if or(abs(b - a) < tol, abs(f(p)) < tol)
                break;
        end
end
end
```

∎

Problem 1.2. Test your function `findzero` by solving $f(x) = \cos x - x$ with $a = 0, b = 1$, and $\mathrm{tol} = 10^{-10}$. Include the printed table in your report, and comment on the apparent order of convergence.

Solution. First, we run `findzero` on $f(x)$ to get the following output table:

Code: Output table

```
>> f = @(x) cos(x) - x

f =

  function_handle with value:

    @(x)cos(x)-x

>> findzero(f, 0, 1, 10^(-10))
 n          a              b              p             f(p)
-----------------------------------------------------------
1     0.00000000     1.00000000     0.68507336     0.08929928
-----------------------------------------------------------
2     1.00000000     0.68507336     0.73629900     0.00466004
-----------------------------------------------------------
3     1.00000000     0.73629900     0.74153913    -0.00410926
-----------------------------------------------------------
4     0.73629900     0.74153913     0.73908362     0.00000253
-----------------------------------------------------------
5     0.74153913     0.73908362     0.73908513     0.00000000
-----------------------------------------------------------
6     0.74153913     0.73908513     0.73908513    -0.00000000
-----------------------------------------------------------
7     0.73908513     0.73908513     0.73908513     0.00000000

ans =

  0.7391
```

Thus, we determine that a root within $[0, 1]$ for $f(x) = \cos(x) - x$ is $x \approx 0.7391$.

Looking at the function, we see that whenever $f(p)f(b) \leq 0$, it follows the secant method. This is when $f(p)$ and $f(b)$ have opposite signs.

On the other hand, when they share the same sign, we set $w = 1/2$ and leaves a untouched and it follows more like the bisection method.

This is to ensure that we can take advantage of the speed advantage of the secant method, while also ensuring the bisection method's advantage of guaranteeing convergence.

With this in mind, along with looking at the table generated, we see that its order of convergence appears to superlinear, being better than that of the bisection method, and somewhere close to that of the secant method's. ■

Problem 1.3. Implement a MATLAB function `findmanyzeros` which finds zeros in the interval $[a, b]$ using the following strategy:

1. Compute $n + 1$ equidistant points x_k, $k = 0, \dots, n$, between a and b
2. For $k = 1, \dots, n$, if $f(x_k)$ and $f(x_{k-1})$ have different signs, compute a zero using `findzero`
3. The output vector p should contain all the computed zeros

Solution. The code for `findmanyzeros` is as follows:

Code: Code for `findmanyzeros`

```
function p = findmanyzeros(f, a, b, n, tol)

x = linspace(a, b, n+1);
```

```
i = 0;
for k = 1:n
        if f(x(k)) * f(x(k-1)) < 0
                p(i) = findzero(f, x(k-1), x(k), tol);
                i = i + 1;
        end
end
end
```

∎

Problem 1.4. Consider the functions

$$f_1(x) = \sin x - e^{-x}$$
$$f_2(x) = \frac{\sin(x^2)}{10 + x^2} - \frac{1}{50} e^{-x/10}$$

Run your function `findmanyzeros` for these functions and their derivatives, on the interval $[0, 10]$ with $n = 50$ points and $\mathrm{tol} = 10^{-10}$.

Plot the functions with the computed zeros and the local extrema as in the example provided.

Give a brief comment about the results.

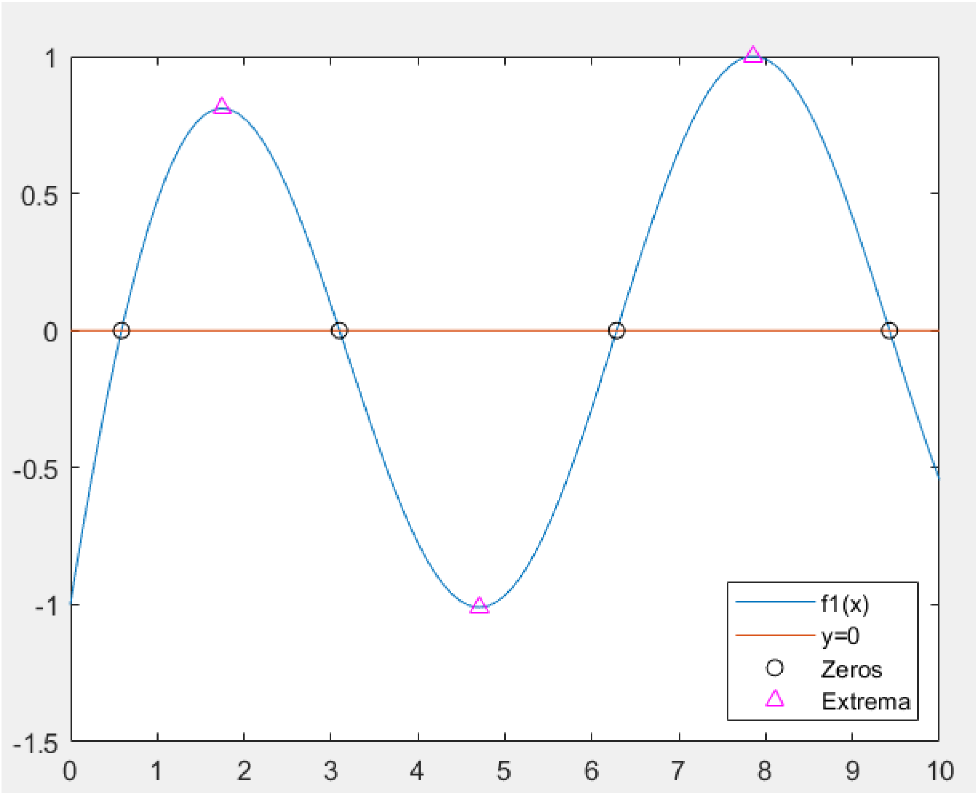
Solution. First, we will have to compute the derivatives of the given functions to plot the extremas with `findmanyzeros`:

$$f_1'(x) = \cos x + e^{-x}$$
$$f_2'(x) = -\frac{2x \sin\left(x^2\right)}{\left(x^2 + 10\right)^2} + \frac{2x \cos\left(x^2\right)}{x^2 + 10} + \frac{\mathrm{e}^{-\frac{x}{10}}}{500}$$

Now, with this in mind, we first look at $f_1(x)$:

Code: Running `findmanyzeros` on $f_1(x)$

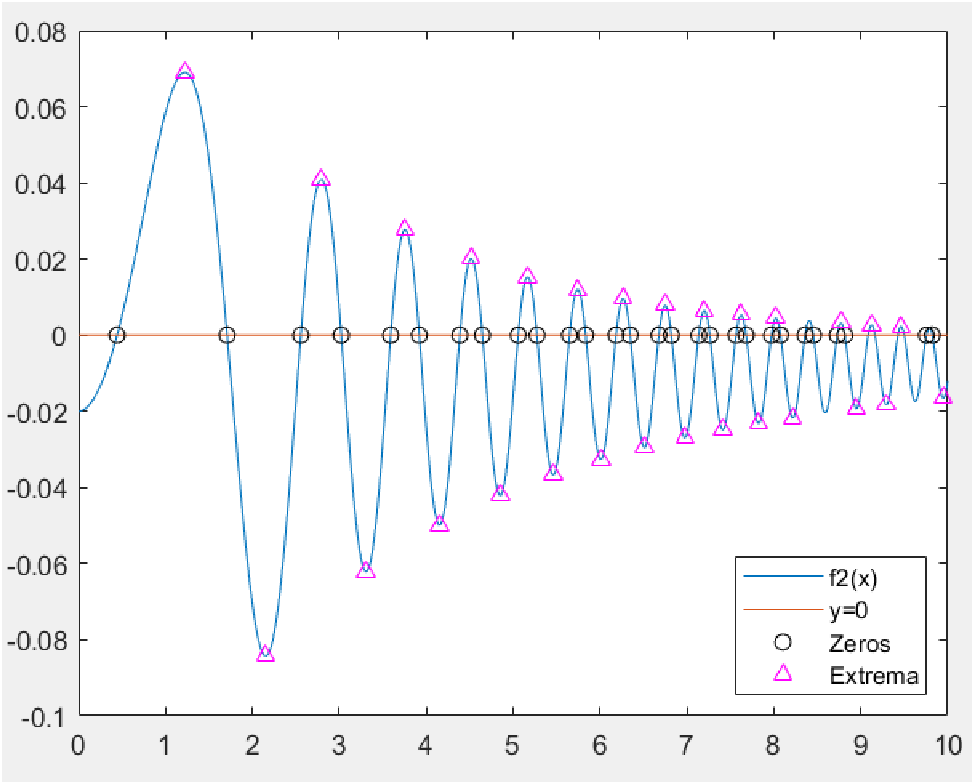
```
f1 = @(x) sin(x) - exp(-x);
df1 = @(x) cos(x) + exp(-x);
p = findmanyzeros(f1, 0, 10, 50, 10^(-10));
ext = findmanyzeros(df1, 0, 10, 50, 10^(-10));
x = 0:0.1:10;
plot(x, f1(x), x, 0*x, p, 0*p, "ko", ext, f1(ext), "m^");
legend("f1(x)", "y=0", "Zeros", "Extrema", "Location", "SouthEast");
```



Next, for $f_2(x)$, we have:

Code: Running `findmanyzeros` on $f_2(x)$

```
df2 = @(x) -((2.*x.*sin(x.^2))./((x.^2+10).^2)) + ( (2.*x.*cos(x.^2))/(x.^2 + 10)) +
↪  (exp(-x./10)./500);
p = findmanyzeros(f2, 0, 10, 50, 10^(-10));
ext = findmanyzeros(df2, 0, 10, 50, 10^(-10));
x = 0:0.01:10;
plot(x, f2(x), x, 0*x, p, 0*p, "ko", ext, f2(ext), "m^");
legend("f2(x)", "y=0", "Zeros", "Extrema", "Location", "SouthEast");
```



We note that for f_2, we miss out on some extremas and zeros near the end; this may be due to the fact that our step size isn't small enough; we should increase n to decrease the step size and increase accuracy. ■