MMA307

Labb1

Magnus Sörensen and Natnael Zekiewos

20160924

1 Part1

1.1 Objectives

Objective of this part is to implement two versions of the same numerical method in MATHLAB and lock at the difference in time it takes to calculate equation.

1.2 details

The function $f(x)$ can be written like 1 and like the Honer's method 2.

$$f(x) = 6x^5 + 5x^4 + 4x^3 + 3x^2 + 2x \tag{1}$$

$$P_b(x) = x(2 + x(3 + x(4 + x(5 + 6x)))) \tag{2}$$

1.3 Code

1.3.1 Native code

The native version of the polynomial is written like this in MATLAB.

```
function y = native_polynomial_calc(x)
    % Calculate the polynyomial
    %   f(x)=6x^5 + 5x^4 + 4x^3.+ 3x^2 + 2+ 3x^2 + 2+ 3x^2 + 2x
    y = 6.*x.^5 + 5.*x.^4 + 4.*x.^3 + 3.*x.^2 + 2.* x;
end
```

And the execution of $x = -\pi$ gives the result.

```
>> native_polynomial_calc(-pi)
ans =
    -1.449772132366787e+03
```

1.3.2 Honer optimized code

The Honer's version of the polynomial is written like this in MATLAB.

```
function y = Honer_polynomial_calc(x)
    % Honer version of the polynomial.
    % f(x) = 6x^5 + 5x^4 + 4x^3 + 3x^2 + 2x
    % f(x) = x.*(2 + x.*(3 + x.*(4 + x.*(5+  6.*x))))
    y = x.*(2 + x.*(3 + x.*(4 + x.*(5+  6.*x))));
end
```

And the output of $x = -\pi$ gives the result with is same as 1.3.1 on the preceding page.

```
>> Honer_polynomial_calc(-pi)
ans =
    -1.449772132366787e+03
```

1.4 Time consumption

The time consumption reported by the script provided in the laboratory instructions.

```
Naivemethod:
Elapsed time is 1.329380 seconds.
Horner smethod:
Elapsed time is 0.343049 seconds.
```

1.4.1 Why honer's function is faster.

The difference in time is do to the number of calculations the computer needed to calculate the equations. In the native version of the equations the computer required 20 calculations while in the Honer version the computer only reacquired 10 calculations to derive the correct answers. Also take in consideration that the error the computer makes while doing those 10 extra steps.

2 Part2

2.1 Objective

Investigate the formulae for numerical differentiation.

2.2 Details

The derivative in a point of an equation that is continuous in that point can be approximated with the central dividing derivative (CDD).

$$f'_{cdd}(x) \approx \begin{matrix} lim \\ h \to 0 \end{matrix} \frac{f(x+h) - f(x-h)}{2 * h} \tag{3}$$

That limit can be acquired by expanding the series 4 around $k = 1$.

$$f'(x) \approx \sum_{\substack{i=0 \\ i \neq k}}^{2k} \left(f(x-(k-i)h) \frac{(-1)^{k+i}(k!)^2}{(k-i)i!(2k-i)!h} \right) \tag{4}$$

Proof for expansion of the series 4 around $k = 1$ gives equation 3 on the previous page

$$f'(x) \approx \sum_{\substack{i=0 \\ i \neq k}}^{2k} \left(f(x-(k-i)h) \frac{(-1)^{k+i}(k!)^2}{(k-i)i!(2k-i)!h} \right) = f(x-h) * \frac{-1}{2h} + f(x+h) * \frac{1}{2*h}$$

$$= \frac{-f(x-h)}{2h} + \frac{f(x+h)}{2*h} = \frac{f(x+h) - f(x-h)}{2h} \qquad \clubsuit$$

For a even more precise central dividing derivative we set $k = 2$ and get the flowing expansion of 4.

$$f(x-(2-0)h)\frac{(-1)^{2+0}(2!)^2}{(2-0)*0!*(2*2-0)!*h} + f(x-(2-1)h)\frac{(-1)^{2+1}(2!)^2}{(2-1)*1!*(2*2-1)!*h}$$

$$+f(x-(2-3)h)\frac{(-1)^{2+3}(2!)^2}{(2-3)*3!*(2*2-3)!*h} + f(x-(2-4)h)\frac{(-1)^{2+4}(2!)^2}{(2-4)*4!*(2*2-4)!*h}$$

$$\Rightarrow f(x-2h)\frac{4}{48h} + f(x-h)\frac{-4}{6h} + f(x+h)\frac{4}{-6h} + f(x+2h)\frac{4}{48h}$$

$$= \frac{f(x-2h) + f(x+2h)}{12h} + \frac{2*(f(x+h) - f(x-h))}{3*h}$$

2.3 code.

The code written in the listing is using a for loop to generate the general solution for al values of k.

```
function y = deriv(f, x, k, h)
% Calculates the k derivitive of function f = f(x) at point x.
    y = 0;
    for i = 0:2*k
        if i ~= k
            y = y + f(x - (k - i).*h).*((-1).^(k+i).*(factorial(k)).^2)/((k
    -i).*factorial(i).*factorial(2.*k - i).*h);
        end
    end

end
```

2.4 Plot of the log error.

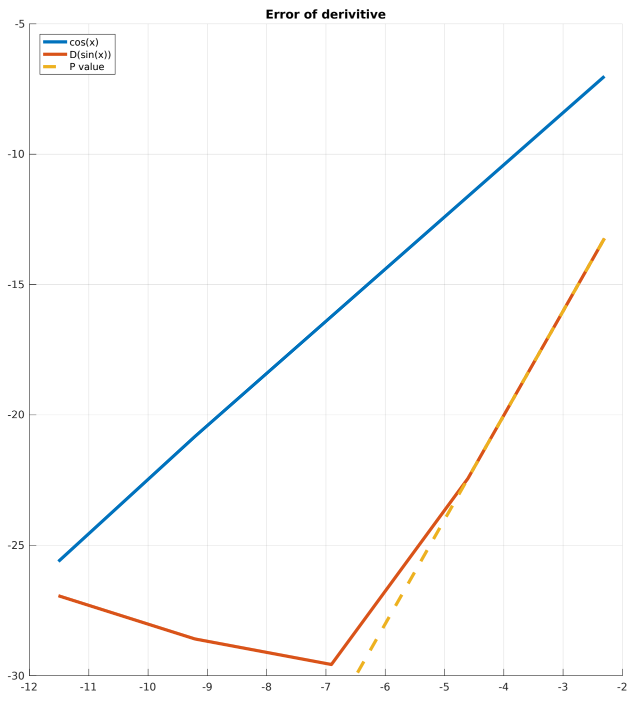

Figure 1: Plot of the error of derivative when $k = 1$ and $cos(x)$ as a reference

The figure 1 shows the error for the error for both $cos(x)$ and $\frac{d}{dx}sin(x)$ where a strait line is the most wanted result. The knee at -7 for $\frac{d}{dx}sin(x)$ is do to the computer in ability to calculate small numbers. So for the $\frac{d}{dx}sin(x)$ our p value lined by doted lines in the graph have a value around:

$$p \approx 4$$

.

Code to draw the error plot 1 on the previous page

```matlab
function plot_error(f)
    % Plots the error of dx

    f = @(x) sin(x);
    h = 1./10.^(1:5);
    x = 1;
    yE = cos(x);
    for i = 1:5
        y(i, 1) = deriv(f, x, 1, h(i));
        y(i, 2) = deriv(f, x, 2, h(i));
        log_err1(i) = log(abs(yE - y(i, 1)));
        log_err2(i) = log(abs(yE - y(i, 2)));
        log_h(i) = log(h(i));
    end
    p1 = 1; p2=2;
    for px = 1:4
        delta_x(px) = log_h(px) - log_h(px+1);
        delta_y(px) = log_err2(px) - log_err2(px+1);
        p(px) = delta_y/delta_x;
    end
    disp('p(1)='); disp(p(1));
    m = log_err2(1) - p(1) .* log_h(1)
    disp('p=');
    disp(transpose(p));
    disp('medium p');
    disp(sum(p)/4)
    disp('log_h');
    disp(transpose(log_h));
    axis([-12 -2 -30 -5]);
    hold on;
    grid on;
    plot(log_h, log_err1, 'LineWidth', 3)
    plot(log_h, log_err2, 'LineWidth', 3)
    plot(log_h, p(1)*(log_h)+m, '--', 'LineWidth' , 3)
    legend('cos(x)','D(sin(x))','P value', 'Location','northwest');
    title('Error of derivitive');
end
```