Numerical Analysis
Assignment 7

Chuan Lu

October 7, 2017

Problem 1. Problem 3.28, Page 190.

(a). We know from the last two conditions (the first derivative),

$$p'(x) = \frac{y_1' - y_0'}{x_1 - x_0}(x - x_0) + y_0'.$$

Integrate this polynomial we get

$$p(x) = \frac{1}{2}\frac{y_1' - y_0'}{x_1 - x_0}(x - x_0)^2 + y_0' x + c.$$

with boundary conditions we get $c = y_0 - y_0' x_0$, then

$$p(x) = y_0 + y_0'(x - x_0)\left(-\frac{(x - x_0)^2}{2(x_1 - x_0)} + 1\right) + y_1'\left(\frac{(x - x_0)^2}{2(x_1 - x_0)}\right).$$

(b). We assume $p'(x_1) = y_1'$ is known. Then using Hermite interpolation we can get the fifth-order polynomial interpolation

$$H(x) = \sum_{i=0}^{2} h_i(x) + \sum_{i=0}^{2} \tilde{h}_i(x),$$

which satisfies

$$H(x_i) = y_i,\ 0 \le i \le 2,$$
$$H(x_i)' = y_i',\ 0 \le i \le 2.$$

In order to get a fourth-order polynomial, we need to cancel out the coefficient of the fifth-order term, which means

$$\sum_{i=0}^{2}(-2l_i'(x_i))\left(\frac{1}{\prod_{j\neq i}(x_i - x_j)}\right)^2 y_i + \sum_{i=0}^{2}\left(\frac{1}{\prod_{j\neq i}(x_i - x_j)}\right)^2 y_i' = 0.$$

It shows

$$-2\left(\frac{(x_0 - x_1) + (x_0 - x_2)}{(x_0 - x_1)^3(x_0 - x_2)^3}y_0 + \frac{(x_1 - x_0) + (x_1 - x_2)}{(x_1 - x_0)^3(x_1 - x_2)^3}y_1 + \frac{(x_2 - x_0) + (x_2 - x_1)}{(x_2 - x_0)^3(x_2 - x_1)^3}y_2\right)$$
$$+\left(\frac{1}{(x_0 - x_1)^2(x_0 - x_2)^2}y_0' + \frac{1}{(x_1 - x_0)^2(x_1 - x_2)^2}y_1' + \frac{1}{(x_2 - x_0)^2(x_2 - x_1)^2}\right) = 0.$$

with $x_i = x_0 + ih$, this equation becomes

$$y_1' = \frac{3}{4h}(y_2 - y_0) - \frac{1}{4}(y_0' + y_2').$$

Then the polynomial is

$$H(x) = \sum_{i=0}^{2} h_i(x)y_i + \sum_{i=0}^{2} \tilde{h}_i(x)y_i',$$

with

$$h_i(x) = (1 - 2l_i'(x_i)(x - x_i))(l_i(x))^2,$$
$$\tilde{h}_i(x) = (x - x_i)(l_i(x))^2,$$

and $l_i(x)$ is the Lagrange basis.

Problem 2. Problem 3.29, Page 190

Solution. Assume the derivative of this quadratic polynomial is

$$p'(x) = a(x - x_1) + y_1'.$$

Then integrate this we get

$$p(x) = \frac{a}{2}(x - x_1)^2 + y_1'x + c.$$

with boundary conditions we know a, c are the solutions of the linear equation system

$$\begin{cases} \dfrac{(x_0 - x_1)^2}{2}u + v = y_0 - y_1'x_0 \\ \dfrac{(x_2 - x_1)^2}{2}u + v = y_0 - y_1'x_2 \end{cases}$$

Thus the existence and uniqueness of this polynomial is equavilent to the existence and uniqueness of solution to the linear system. Thus we know the matrix of coefficients is invertible, which means

$$x_0 \neq x_2.$$

Besides, in order to get a non-degenerated quadratic polynomial, the solution of $u \neq 0$. Thus

$$y_1' \neq 0.$$

Problem 3. Problem 3.38, Page

Solution. Let $k(x) = \hat{s}(x) - g(x)$, then

$$\begin{aligned} \int_{x_0}^{x_n} |g''(x)|^2 dx &= \int_{x_0}^{x_n} |\hat{s}''(x) - k''(x)|^2 dx \\ &= \int_{x_0}^{x_n} |\hat{s}''(x)|^2 dx - 2\int_{x_0}^{x_n} \hat{s}''(x)k''(x)dx + \int_{x_0}^{x_n} |k''(x)|^2 dx. \end{aligned}$$

By integration by parts,

$$\int_{x_0}^{x_n} \hat{s}''(x)k''(x)dx = \hat{s}''(x)k'(x)\Big|_{x_0}^{x_n} - \int_{x_0}^{x_n} \hat{s}'''(x)k'(x)dx$$

We know from the boundary conditions,

$$\hat{s}''(x_0) = \hat{s}''(x_n) = 0,$$

thus the first term equals to 0. Besides, since $\hat{s}(x)$ is a cubic polynomial, $\hat{s}'''(x)$ is a constant, denoted by c. Then

$$\int_{x_0}^{x_n} \hat{s}'''(x)k'(x)dx = ck(x)\Big|_{x_0}^{x_n} = c(k(x_n) - k(x_0)).$$

We know from intepolation conditions that

$$k(x_i) = \hat{s}(x_i) - g(x_i) = 0,\ 0 \leq i \leq n$$

Thus the second term also equals to 0. Then we know

$$\int_{x_0}^{x_n} |g''(x)|^2 dx = \int_{x_0}^{x_n} |\hat{s}''(x)|^2 dx + \int_{x_0}^{x_n} |\hat{s}''(x) - g''(x)|^2 dx \geq \int_{x_0}^{x_n} |\hat{s}''(x)|^2 dx.$$

Problem 4. For $n = 1, \cdots, 10$, generate the Newton divided difference polynomials that interpolates $\cos(x)$ for $x \in [0, 2\pi]$, at n equally spaced points. Create an error table.

Solution. The code and result (the graph of error and table of Max error) is listed below.

```
function val = newton_poly(xx, fx, x)
% Newton Polynomial
% xx, fx: intepolation points, and function values;
% x: points for evaluating values;
```

```
5
6  % 1. get the coefficients of Newton Poly,
7  %    the result saved in 'fx'
8  n = length(xx);
9  for i=1:(n-1)
10     fx(i+1:n) = (fx(i+1:n)-fx(i:n-1))./(xx(i+1:n)-xx(i:n-1));
11 end
12
13 % 2. evaluate the function values with points given,
14 %    using Horner's method
15 val = fx(n);
16 for i=(n-1):-1:1
17     val = val.*(x-xx(i))+fx(i);
18 end
```

```
1  % prob4
2  x = 0:1e-3:2*pi;
3  % res = zeros(10, length(x));
4  m = 10;
5  error = zeros(m, 1);
6  for n = 1:m
7      xx = linspace(0, 2*pi, n);
8      val = newton_poly(xx, cos(xx), x);
9      error(n, 1) = max(abs(val-cos(x)));
10     plot(x, abs(val-cos(x)));
11     hold on;
12 end
13 disp(error);
14 legend('n=1', 'n=2', 'n=3', 'n=4', 'n=5',...
15     'n=6', 'n=7', 'n=8', 'n=9', 'n=10', 'Location', 'Best');
16 title('Error with order'); xlabel('x'); ylabel('Error');
```

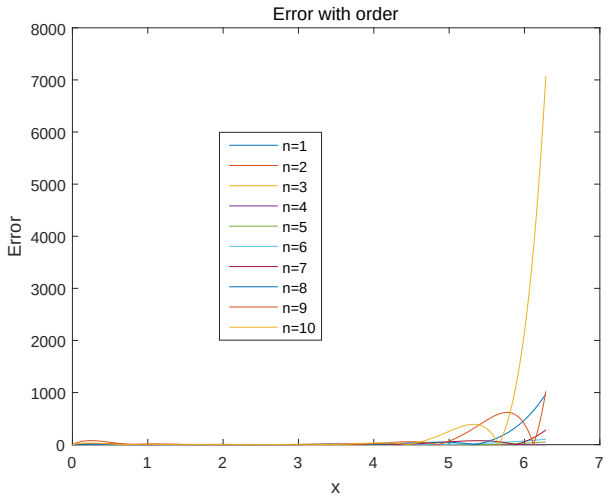

Figure 1: Error of interpolation with different order

```
1  >> prob4
2     1.0e+03 *
3
4      0.0020
5      0.0020
6      0.0040
7      0.0045
8      0.0520
9      0.1047
10     0.2818
```

```
    0.9745
    1.0237
    7.0735
```

Problem 5. For $n = 1, \cdots, 5$, construct Hermite interpolant that interpolates $\cos(x)$ for $x \in [0, 2\pi]$, at n equally spaced points. Create an error table.

Solution. The code and result is as below. It seems Hermite intepolation is more stable then Newton intepolation.

```
function val = Hermite_poly(xx, fx, x)
% Hermite Polynomial;

n = length(xx);
H = zeros(n, n);

% get coefficients
for i=1:n
    H(i, 1) = fx(find(xx == xx(i), 1, 'first'));
end;

for i=2:n
    for j=1:(n-i+1)
        pos1 = find(xx == xx(j), 1, 'first');
        pos2 = find(xx == xx(i+j-1), 1, 'first');
        if pos1 == pos2
            H(j, i) = fx(pos1+i-1)/factorial(i-1);
        else
            H(j , i) = (H(j+1, i-1)-H(j, i-1))/(xx(i+j-1)-xx(j));
        end
    end
end
val = H(1, n);
for i=(n-1):-1:1
    val = val.*(x-xx(i))+H(1, i);
end
```

```
% prob4
x = 0:1e-3:2*pi;
% res = zeros(10, length(x));
m = 5;
error = zeros(m, 1);
for n = 1:m
    xx = linspace(0, 2*pi, 2*n);
    xx(2:2:end) = xx(1:2:end);
    yy = cos(xx);
    yy(2:2:end) = -sin(xx(2:2:end));
    val = Hermite_poly(xx, yy, x);
    error(n, 1) = max(abs(val-cos(x)));
    plot(x, abs(val-cos(x)));
    hold on;
end
disp(error);
legend('n=1', 'n=2', 'n=3', 'n=4', 'n=5', 'Location', 'Best');
title('Error with order'); xlabel('x'); ylabel('Error');
```

```
>> prob5
    2.0000
    4.0801
    0.8723
    0.0900
    0.0056
```

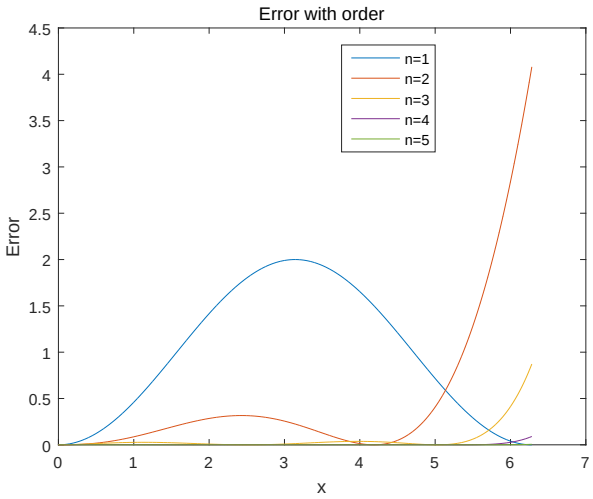

Figure 2: Error of Hermite interpolation with different order