Numerical Methods – HW8

b04902053 鄭淵仁

June 23, 2018

Environment

- **OS:** Windows
- **CPU:** Intel Core i5-4200H
- **Programming Language:** Matlab R2018a

Problem 8-1: Write a program which can solve general spline problems.

Codes

my_spline

- Return `breaks` and `coefs` so that later we can use Matlab's `ppval` to evaluate the result.
- Use a system to solve the parameter `c`.

```
function [breaks, coefs] = my_spline(x, y)
    breaks = x;
    [n, ~] = size(y);
    n = n - 1;

    %% solve a
    a = y;

    %% solve c
    h = x(2: n+1) - x(1: n);

    tmp_system_A = zeros(n-1, n+1);
    tmp_system_A(1:n-1, 1:n-1) = tmp_system_A(1:n-1, 1:n-1) ...
        + diag(h(1:n-1));
    tmp_system_A(1:n-1, 2:n)   = tmp_system_A(1:n-1, 2:n)   ...
```

```
        + diag(2 * (h(1:n-1) + h(2:n)));
    tmp_system_A(1:n-1, 3:n+1) = tmp_system_A(1:n-1, 3:n+1) ...
        + diag(h(2:n));
    system_A = tmp_system_A(1:n-1, 2:n);

    tmp_system_b = (3 ./ h) .* (a(2:n+1) - a(1:n));
    system_b = tmp_system_b(2: n) - tmp_system_b(1: n-1);

    c = zeros(n+1, 1);
    c(2:n) = system_A \ system_b;

    %% solve b
    b = (1 ./ h) .* (a(2:n+1) - a(1:n)) - (h ./ 3) ...
        .* (2 * c(1:n) + c(2:n+1));

    %% solve d
    d = (c(2:n+1) - c(1:n)) ./ (3 .* h);

    coefs = [d, c(1:n), b, a(1:n)];
end
```

Codes to Compare my_spline and Matlab

- Use `ppval` to evaluate `my_spline` and Matlab's `spline`

```
x = % ...
y = % ...
xx = % ...

[breaks, coefs] = my_spline(x, y);
pp = mkpp(breaks, coefs);
pp_vals = ppval(pp, xx);

[sp] = spline(x, y);
sp_vals = ppval(sp, xx);

hold on
sp_o = plot(x, y, 'o', 'Color', 'k');
pp_line = plot(xx, pp_vals, '-', 'Color', 'b');
sp_line = plot(xx, sp_vals, '-', 'Color', 'r');
hold off

legend([sp_o, pp_line, sp_line], ...
    {'exact value given', 'My spline', "Matlab's spline"});
```

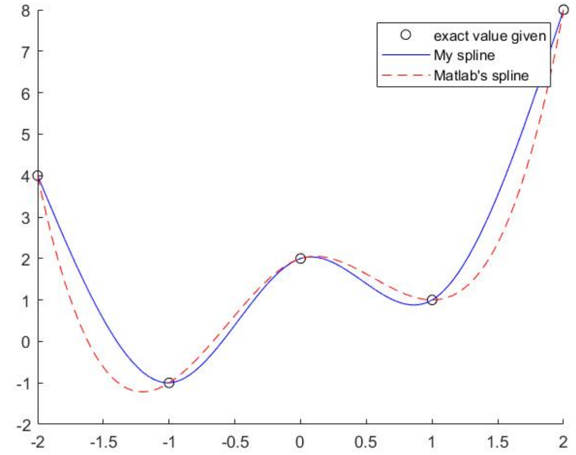
Test your program by using some data and compare with the result of Matlab

Data 1

- Data:

```
x = [-2 -1 0 1 2]';
y = [4 -1 2 1 8]';
xx = linspace(-2, 2, 101);
```

- Graph:

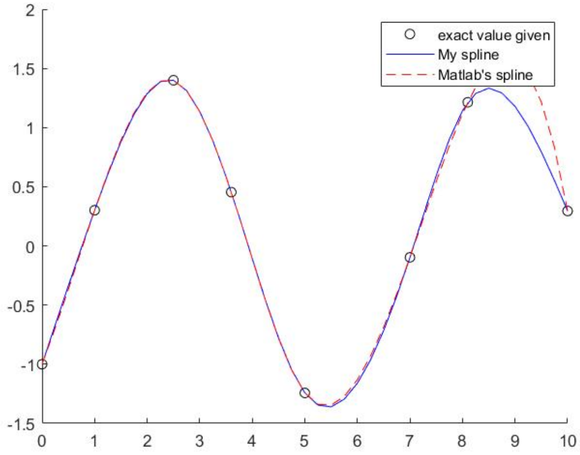


Data 2

- Data:

```
x = [0 1 2.5 3.6 5 7 8.1 10]';
y = sin(x) - cos(x);
xx = 0:.25:10;
```

- Graph:



Problem 8-2

Write down the three equations of $\nabla E = 0$

$$\nabla E = \begin{bmatrix} \frac{\partial E}{\partial a} & \frac{\partial E}{\partial b} & \frac{\partial E}{\partial c} \end{bmatrix}^\mathsf{T} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^\mathsf{T}$$

$$\begin{cases} \dfrac{\partial E}{\partial a} = \displaystyle\sum_{i=1}^{m} 2(y_i - (ax_i^2 + bx_i + c)) \cdot (-x_i^2) & = 0, \\ \dfrac{\partial E}{\partial b} = \displaystyle\sum_{i=1}^{m} 2(y_i - (ax_i^2 + bx_i + c)) \cdot (-x_i) & = 0, \\ \dfrac{\partial E}{\partial c} = \displaystyle\sum_{i=1}^{m} 2(y_i - (ax_i^2 + bx_i + c)) & = 0 \end{cases}$$

$$\begin{cases} \displaystyle\sum_{i=1}^{m} y_i = a\left(\sum_{i=1}^{m} x_i^2\right) + b\left(\sum_{i=1}^{m} x_i\right) + c\left(\sum_{i=1}^{m} 1\right), \\ \displaystyle\sum_{i=1}^{m} x_i y_i = a\left(\sum_{i=1}^{m} x_i^3\right) + b\left(\sum_{i=1}^{m} x_i^2\right) + c\left(\sum_{i=1}^{m} x_i\right), \\ \displaystyle\sum_{i=1}^{m} x_i^2 y_i = a\left(\sum_{i=1}^{m} x_i^4\right) + b\left(\sum_{i=1}^{m} x_i^3\right) + c\left(\sum_{i=1}^{m} x_i^2\right) \end{cases}$$

Write a program doing quadratic least square

quadratic least square

- Use a system to solve the parameters `a`, `b`, `c`.

```
function [coefs] = qua_least_square (x, y)
    [m, ~] = size(x);

    sum_x   = sum(x);
    sum_x_2 = sum(x .^ 2);
    sum_x_3 = sum(x .^ 3);
    sum_x_4 = sum(x .^ 4);

    sum_y     = sum(y);
    sum_x_y   = sum(x .* y);
    sum_x_2_y = sum((x .^ 2) .* y);

    system_A = [sum_x_2, sum_x, m;
        sum_x_3, sum_x_2, sum_x;
        sum_x_4, sum_x_3, sum_x_2];

    system_b = [sum_y; sum_x_y; sum_x_2_y];

    coefs = system_A \ system_b;
end
```

Randomly generate some (x_i, y_i) and draw a figure showing your approximation

Testing Code

- Use `polyval` to evaluate the polynomial.

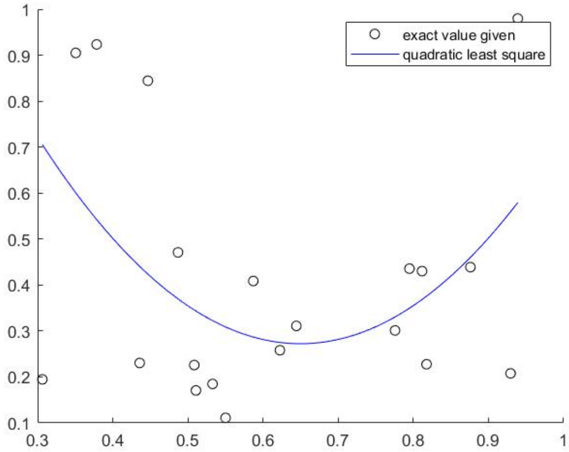
```
x = % ...
y = % ...
xx = % ...

[coefs] = qua_least_square(x, y);
pv = polyval(coefs, xx);

hold on
qua_o    = plot(x, y, 'o', 'Color', 'k');
qua_line = plot(xx, pv, '-', 'Color', 'b');
hold off

legend([qua_o, qua_line], {'exact value given', 'quadratic least square'})
```

Data 1

- Data:

```
x = rand(20, 1);
y = rand(20, 1);
xx = linspace(min(x), max(x), 100);
```

- Graph:

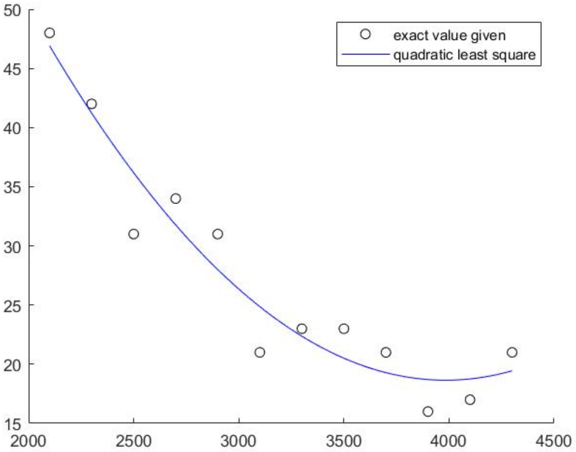


Data 2

- Data:

```
x = [2100 2300 2500 2700 2900 3100 3300 3500 3700 3900 4100 4300]';
y = [48 42 31 34 31 21 23 23 21 16 17 21]';
xx = 2100: 10: 4300;
```

- Graph:


exact value given
quadratic least square