Interpolation - cubic spline

wiki: https://en.wikipedia.org/wiki/Spline_(mathematics)#Natural_continuity

given a series pairs of function values $x_0, x_1, \ldots, x_N$ (sorted), $f_0, f_1, \ldots, f_N$, estimate function value at some intermediate point x.

1. continuous in first and second order derivatives

interval: $h_i = x_{i+1} - x_i, \quad i = 0,1,\ldots,N-1$

auxiliary function: $\lambda_i = \dfrac{x_{i+1} - x}{h_i}, \quad \omega_i = 1 - \lambda_i, \quad i = 0,1,\ldots,N-1$

1. $\lambda_i(x_i) = \omega_i(x_{i+1}) = 1$
2. $\lambda_i(x_{i+1}) = \omega_i(x_i) = 0$
3. $\lambda_i'(x_i) = \lambda_i'(x_{i+1}) = -\dfrac{1}{h_i}$
4. $\omega_i'(x_i) = \omega_i'(x_{i+1}) = \dfrac{1}{h_i}$

Interpolation function: $f_{[x_i, x_{i+1}]} = f_i\lambda_i + f_{i+1}\omega_i + \dfrac{h_i^2 f''_i}{6}\left(\lambda_i^3 - \lambda_i\right) + \dfrac{h_i^2 f''_{i+1}}{6}\left(\omega_i^3 - \omega_i\right)$

1. $f(x_i)_{[x_i, x_{i+1}]} = f_i$
2. $f(x_{i+1})_{[x_i, x_{i+1}]} = f_{i+1}$
3. $f'(x_i)_{[x_i, x_{i+1}]} = -\dfrac{f_i}{h_i} + \dfrac{f_{i+1}}{h_i} - \dfrac{h_i}{3} f''_i - \dfrac{h_i}{6} f''_{i+1}$
4. $f'(x_{i+1})_{[x_i, x_{i+1}]} = -\dfrac{f_i}{h_i} + \dfrac{f_{i+1}}{h_i} + \dfrac{h_i}{6} f''_i + \dfrac{h_i}{3} f''_{i+1}$
5. $f''(x_i)_{[x_i, x_{i+1}]} = f''_i$
6. $f''(x_{i+1})_{[x_i, x_{i+1}]} = f''_{i+1}$

equations: $h_{i-1} f''_{i-1} + 2(h_{i-1} + h_i) f''_i + h_i f''_{i+1} = 6\dfrac{f_{i+1} - f_i}{h_i} - 6\dfrac{f_i - f_{i-1}}{h_{i-1}}$

Boundary condition for natural cubic spline: $f''_0 = f''_N = 0$

denote: $M_i = 6\dfrac{f_{i+1} - f_i}{h_i} - 6\dfrac{f_i - f_{i-1}}{h_{i-1}}, \quad i = 1,2,\ldots,N-1$

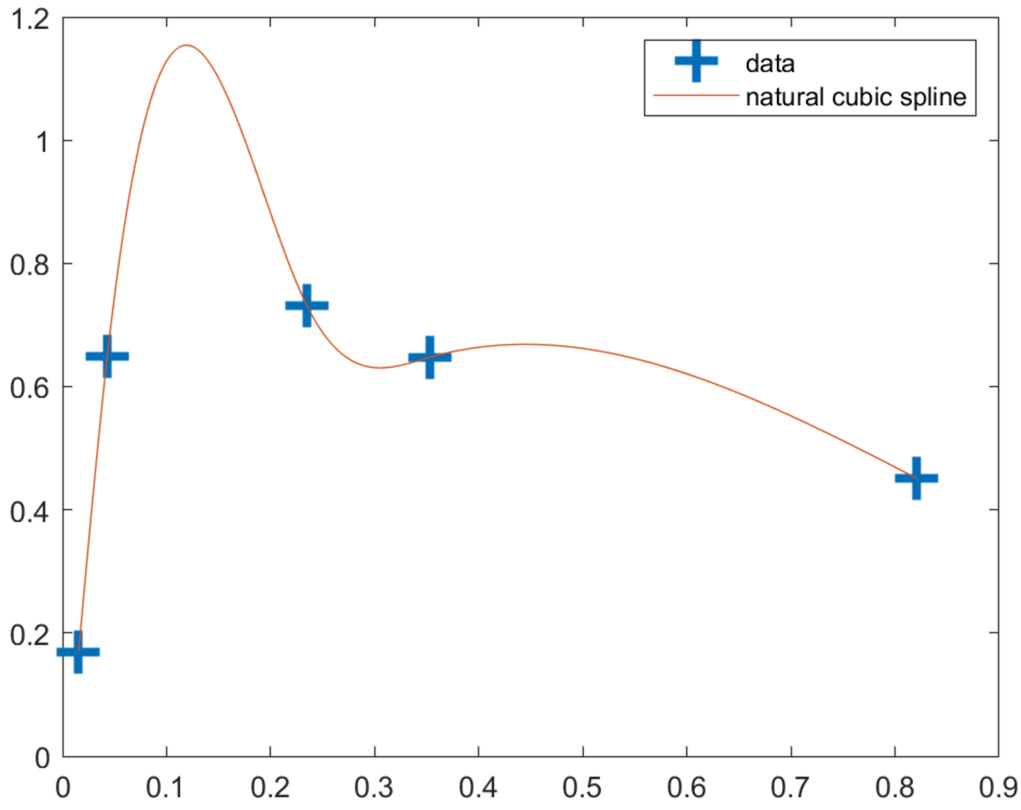
$$
\begin{pmatrix}
1 & & & & & \\
h_0 & 2(h_0+h_1) & h_1 & & & \\
 & h_1 & 2(h_1+h_2) & h_2 & & \\
 & & \ddots & \ddots & \ddots & \\
 & & & h_{N-1} & 2(h_{N-1}+h_N) & h_N \\
 & & & & & 1
\end{pmatrix}
\begin{pmatrix}
f''_0 \\ f''_1 \\ f''_2 \\ \vdots \\ f''_{N-1} \\ f''_N
\end{pmatrix}
=
\begin{pmatrix}
0 \\ M_1 \\ M_2 \\ \vdots \\ M_{N-1} \\ 0
\end{pmatrix}
$$

parameter

```
num_data = 5;
data_x = sort(rand(num_data,1));
data_y = rand(size(data_x));

num_interp_x = 1000;
interp_x = linspace(min(data_x),max(data_x),num_interp_x).';
```

Natural Cubic Spline

```
fig = figure();
ax = axes(fig, 'NextPlot','add','Box','on');
hline0 = plot(ax, data_x, data_y, 'LineStyle', 'none', 'Marker', '+', 'MarkerSize', 15,'LineWi

tmp1 = my_cubic_spline(data_x, data_y, interp_x);
hline1 = plot(ax, interp_x, tmp1);

legend([hline0,hline1],{'data', 'natural cubic spline'})
```



function

```
function ret = my_cubic_spline(x0, y0, x)
% x0(N0,1) y0(N0,1) x(N1,1)
% ret(N1,1)
% reference: https://en.wikipedia.org/wiki/Spline_(mathematics)#Natural_continuity
N0 = size(x0,1);
assert(N0>1, 'cubic_spline requires at least two points');

tmp1 = all(x>=min(x0(:))) && all(x<=max(x0(:)));
assert(tmp1, 'cubic_spline require all x point sit between [min(x), max(x)]');

tmp1 = sortrows([x0,y0]);
x0 = tmp1(:,1);
y0 = tmp1(:,2);
delta_x = x0(2:end) - x0(1:(end-1));

tmp1 = [delta_x(1:end-1),(delta_x(1:(end-1))+delta_x(2:end))*2,delta_x(2:end)];
tmp2 = full(spdiags(tmp1, [0;1;2], N0-2, N0));
matA = [1,zeros(1,N0-1);tmp2;zeros(1,N0-1),1];

tmp1 = (y0(2:end) - y0(1:(end-1)))./delta_x;
matb = [0;6*(tmp1(2:end)-tmp1(1:(end-1)));0];
params = matA\matb;
```

```
ret = zeros(size(x));
for ind1 = 1:numel(x)
    ind2 = min(find(x(ind1)>=x0, 1, 'last'), N0-1);
    h = delta_x(ind2);
    lambda = (x0(ind2+1)-x(ind1))/h;
    omega = 1 - lambda;
    ret(ind1) = y0(ind2)*lambda + y0(ind2+1)*omega + params(ind2)*h^2/6*(lambda^3-lambda) ...
        + params(ind2+1)*h^2/6*(omega^3-omega);
end
end
```