Algorithms for Cubic Spline Interpolation

Algorithm for finding z_i, $i = 0 \ldots n$

```
% Compute the h_i and b_i
for i = 0 : n-1
      h_i = x_{i+1} - x_i
      b_i = (y_{i+1} - y_i)/h_i
end
% Gaussian Elimination
u_1 = 2(h_0 + h_1)
v_1 = 6(b_1 - b_0)
for i = 2 : n-1
      u_i = 2(h_{i-1} + h_i) - h_{i-1}^2/u_{i-1}
      v_i = 6(b_i - b_{i-1}) - h_{i-1}v_{i-1}/u_{i-1}
end
% Back-substitution
z_n = 0
for i = n-1 : -1 : 1
      z_i = (v_i - h_i z_{i+1})/u_i
end
z_0 = 0
```

How many flops are required to compute the z_i?

Evaluating $S(x)$

Remember that once you have the z_i, you can evaluate $S(x)$ as follows:

$$S_i(x) = \frac{z_i}{6h_i}(x_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - x_i)^3 + C_i(x - x_i) + D_i(x_{i+1} - x)$$

with $C_i = \frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}$ and $D_i = \frac{y_i}{h_i} - \frac{h_i}{6}z_i$.

This, however, is not the most efficient computational form. We would like to use the idea of nested multiplication, so write:

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

Notice that this is just the infinite Taylor expansion $S_i(x) = \sum\limits_{n=1}^{\infty} \frac{1}{n!}(x - x_i)^n S_i^{(n)}(x_i)$ (with $S_i^{(n)} = 0$ for $n \geq 4$ since S_i is a cubic polynomial).

Therefore,

$$\begin{aligned}
a_i &= S_i(x_i) = y_i \\
b_i &= S_i'(x_i) = -\frac{h_i}{6}z_{i+1} - \frac{h_i}{3}z_i + \frac{y_{i+1} - y_i}{h_i} \\
c_i &= \frac{1}{2}S_i''(x_i) = \frac{z_i}{2} \\
d_i &= \frac{1}{6}S_i'''(x_i) = \frac{z_{i+1} - z_i}{6h_i}
\end{aligned}$$

Algorithm for Evaluating $S(x)$

```
for i = 0 : n-1
    if x <= x_{i+1}
        break;
    end
end
h = x_{i+1} - x_i
Compute a, b, c and d as above.
S = a + (x - x_i)(b + (x - x_i)(c + (x - x_i)d))
```

How many flops are required to for each spline function evaluation?