Challenge problems

1. Write $3x^3 + 4x^2 + 5x + 6$ in nested form, & evaluate it @ -2, 3.

   Solution: $p(x) = (3x+4)x+5)x+6$; \[ p(-2) = \frac{2}{3} \quad p(3) = 56 \] So $p(5) = -12$. This is the "synthetic division" approach.

   Another approach: $2x+4 \rightarrow 13x+5 \rightarrow 10x+6 \rightarrow 10x+5 \rightarrow -\frac{2}{3}$. So $p(-2) = 138$.

2. Evaluate $p(x) = 8x^5 - 20x^4 - 9x^3 + 30x^2 - 100x + 16$ mentally @ $x = 2, 3$.

   Solution: the process is $p(2) = 82 - 20(4) - 9(8) + 30(16) - 100(2) + 16 = -200 = p(2)$;
   $p(3) = 83 - 20(27) - 9(27) + 30(81) - 100(3) + 16 = 67 = p(3)$

3. Given $a < b$, n ∈ N, $h = \frac{b-a}{n}$, $x_k = a + kh$, for $k = 0, \ldots, n$, and $f \in C^n([a, b])$, show that $\| f - p_n \|_\infty \leq \frac{h^{n+1} \| f^{(n+1)} \|_\infty}{(n+1) \cdot n!}$.

   Solution: $\| f - p_n \|_\infty \leq \max_{x \in [a, b]} | f(x) - p_n(x) | = \max_{x \in [a, b]} \left| \frac{1}{(n+1) \cdot n!} \int_{a}^{b} (x - x_k)^{n+1} f^{(n+1)}(x) \, dx \right|$

4. Use the error bound & notation in problem 3 to find an index $n \in N$ such that $\| f - p_n \|_\infty \leq \frac{1}{4}$, where $f(x) = \sin(x)$ in $(5, x)$.

   Solution: $\| f^{(n+1)}(x) \|_{\infty} \leq | f^{(n+1)}(x) |_{\infty}$, so $\| f^n \|_\infty \leq | f^n(x) |_{\infty}$.

5. Find the error in estimating $f^n(x)$ with $f^n(x) = \frac{f(x)-2f(x)+f(2x)}{h^2}$.

   Solution: $\left| f^n(x) - \frac{f(x) - 2f(x) + f(2x)}{h^2} \right| = \left| \frac{h^2}{12} \left[ f^{(3)}(x_0) + 2f^{(3)}(x_1) + f^{(3)}(x_2) \right] \right|$

   But $\left| f^n(x) - \frac{f(x) - 2f(x) + f(2x)}{h^2} \right| = \frac{h^2}{12} \left[ f^{(3)}(x_0) + 2f^{(3)}(x_1) + f^{(3)}(x_2) \right]$
6. If \( x_0 < x_1 < ... < x_n = b, S: \mathbb{R}^n \) for \( i = 0, ... , n \). Show that the Hermite polynomial \( p \) satisfying \( p^{(k)}(x_i) = f^{(k)}(x_i) \) for all \( i = 0, ... , n \) and all \( k = 0, ... , n \) is the unique polynomial of degree \( \leq n \) satisfying that property.

Solution: Suppose \( g \) is another degree \( \leq n \) polynomial with \( g^{(k)}(x_i) = f^{(k)}(x_i) \) for all \( i = 0, ... , n \) and all \( k = 0, ... , n \).

Then \( g^{(k)}(x_i) = p^{(k)}(x_i) \) \( \forall i = 0, ... , n \), \( \forall k = 0, ... , n \), so we may view \( p \) as a Hermite polynomial for \( g \), and thus we can use the error formula \( g(x) = p(x) + (x-x_i)^2 + (x-x_i)^4 + \cdots + (x-x_i)^{2n} \cdot \sum_{j=0}^{n} \frac{f^{(j)}(x_i)}{j!} \), where \( N = 2n \). But \( g \) is degree \( \leq 2n \), so \( g^{(n)} = 0 \).

Thus \( g = p \), and the Hermite polynomial is unique.

Note: Instead of using the error formula for Hermite polynomials, we can use a direct argument:

Put \( g(x) = p(x) - f(x) \). Then \( g^{(k)}(x_i) = 0 \) \( \forall i = 0, ... , n \), \( \forall k = 0, ... , n \). Then \( g \) has \( \sum_{i=0}^{n} \) zeros (including multiplicities) and is degree \( \leq 2n \).

So \( g^{(n)}(x) = 0 \) for some \( x_i \), \( 0 < x_i \), and \( g \) has a zero of multiplicity \( n+1 \) at \( x_i \).

So \( g^{(n)}(x) = 0 \) for some \( x_i \), \( 0 < x_i \), and \( g \) has a zero of multiplicity \( n+1 \) at \( x_i \).

Graphically: Zeros of \( g^{(n)} \).

7. If \( f \) is the Lagrange polynomial for \( f \) using nodes \( a = x_0 < ... < x_n = b \), find an expression for the error \( (f - p) \) in an approximate manner by \( p \) over \( [a,b] \).

Solution: \( \forall x \in [a,b] \) we have \( f(x) = p(x) + (x-x_0)(x-x_1)(x-x_2)...(x-x_n) \cdot f[x_0, ... , x_n, x] \). Differentiating \( \forall x \in [a,b] \) yields \( f'(x) = p'(x) + (x-x_0)(x-x_1)(x-x_2)...(x-x_n) \cdot f[x_0, ... , x_n, x] \cdot \text{linear combo of } f(x_0), f(x_1), f(x_2), \ldots, f[x_0, ... , x_n, x] \).

In particular, assuming \( f \) is at least \( C_2 \), we get \( f'(x) = p'(x) + (x-x_0)(x-x_1)(x-x_2)...(x-x_n) \cdot f[x_0, ... , x_n, x] \).

8. If \( f[1] = 16, f[1,2] = 13, \) and \( f[1,2,3] = 7 \), find \( f[3] \).

Solution: \( x_i \):
\[
\begin{array}{c|c|c|c}
1 & 16 & 6 \times 7 & 6 \\
2 & 13 & 4 \times 3 & 4 \\
3 & 7 & 2 \times 4 & 2 \\
\end{array}
\]

By filling in the missing entries:
\[
\begin{array}{c}
7 = \frac{f[1,2] - 13}{2 - 1} \\
-13 = \frac{f[2] - 16}{2 - 1} \\
1 = \frac{f[3] - 7}{2 - 1} \\
\end{array}
\]

Thus \( f[3] = 4 \).

9. Let \( p_{x_0, ..., x_n}(x) \) denote the Lagrange interpolating polynomial matching \( f \) at \( x_0, ... , x_n \). If \( p_5(10) = 56, p_{6}(x) = 8 - 2x, \) find \( p_{5}(3) \).

Solution: Use Neville's method.
\[
p_{3,6}(3) = \frac{8 - 2 \cdot 3}{3 - 1} = \frac{2}{2} = 1
\]

10. Let \( f \) denote the piecewise linear spline \( S \) with knots at \( a = x_0 < x_1 < ... < x_n = b \), and find an expression for the error \( \Delta f \) in approximating \( f \) by \( S \) over \( [x_0, x_n] \). Assume \( f \) is the first order Lagrange interpolant for \( f \) over \( [x_{i-1}, x_i] \), \( f(x_n) = f[x_0, x_1, ..., x_{n-1}, x_n] \). This defines \( f \) over \( [x_0, x_n] \). Over each \( [x_{i-1}, x_i] \), \( f \) is the first order Lagrange interpolant for \( f \) over \( [x_{i-1}, x_i] \), \( f(x_n) = f[x_0, x_1, ..., x_{n-1}, x_n] \). The definition of \( f \) over \( [x_0, x_n] \) gives the spline \( S(x) \) as \( f(x) = (x-x_{i-1}) \cdot f(x_i) + \frac{f(x_{i-1}) + f(x_i)}{2} \cdot \text{error} \).

Note: The definition of the error is the difference between the actual function \( f \) and the approximating spline \( S \).
If \( s(a) \) and \( s''(a) \) are specified, then there are 4 constraints imposed on \( p_i \), which may be viewed as a certain Hermite polynomial, so all 4 coefficients of \( p_i \) can be uniquely determined. Then \( p_i(x), p_i'(x), p_i''(x), \) and \( p_i'''(x) \) are specified (as \( s \) must be \( C^2 \)), so \( p_i \) is uniquely determined, etc. So it seems reasonable to assume that we can impose 2 extra conditions on our spline to uniquely specify it. This leads to the two most common cubic splines: natural & clamped.

**Natural:** \( s''(a) = 0 = s''(b) \)

**Clamped:** \( s(a) = f(a), s'(b) = f'(b) \)

Let’s track down the \( c_i \)’s & get a recursive relationship:

\[
\begin{align*}
(p_i(x_i)) &= f(x_i) & \text{for } i = 1, \ldots, n \\
p_i'(x_i) &= f'(x_i) & \text{for } i = 1, \ldots, n \\
p_i''(x_i) &= f''(x_i) & \text{for } i = 1, \ldots, n \\
p_i'''(x_i) &= f'''(x_i) & \text{for } i = 1, \ldots, n
\end{align*}
\]

\[
\begin{align*}
a_i &= f(x_i) \\
b_i + 2c_i + 3d_i &= f'(x_i) \\
b_i + 2c_i + 3d_i &= f''(x_i)
\end{align*}
\]

\[
\begin{align*}
-2b_i - c_i + 3d_i &= f'''(x_i) \\
-3b_i - 2c_i + 3d_i &= f''''(x_i)
\end{align*}
\]

\[
\begin{align*}
2b_i &= -c_i + 3d_i \\
\frac{1}{3}c_i + \frac{1}{3}b_i &= -c_i + \frac{2}{3}a_i \\
\frac{1}{3}b_i + \frac{1}{3}c_i &= -c_i + \frac{2}{3}a_i
\end{align*}
\]

\[
\begin{align*}
c_{i-1} - 3b_i - 2c_i + 3d_i &= f'''(x_i) \\
c_{i-1} - 3b_i - 2c_i + 3d_i &= f''''(x_i)
\end{align*}
\]

\[
\begin{align*}
c_{i-1} - 3b_i - 2c_i + 3d_i &= f'''(x_i) \\
c_{i-1} - 3b_i - 2c_i + 3d_i &= f''''(x_i)
\end{align*}
\]

\[
\begin{align*}
c_{i-1} - 3b_i - 2c_i + 3d_i &= f'''(x_i) \\
c_{i-1} - 3b_i - 2c_i + 3d_i &= f''''(x_i)
\end{align*}
\]

Thus, writing a linear system for the \( c_i \)’s, we get:

\[
\begin{pmatrix}
-2 & 1 & 0 & \cdots & 0 \\
1 & -2 & 1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & \cdots & \cdots & -2 & 1
\end{pmatrix}
\begin{pmatrix}
c_1 \\
c_2 \\
\vdots \\
c_n
\end{pmatrix}
= 
\begin{pmatrix}
f_1 - f_{n-1} \\
f_2 - f_{n-1} \\
\vdots \\
f_n - f_{n-1}
\end{pmatrix}
\]

Thus, the linear system is:

\[
\begin{pmatrix}
-a_1 & 0 & \cdots & 0 \\
1 & -a_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & \cdots & \cdots & 1
\end{pmatrix}
\begin{pmatrix}
b_1 \\
b_2 \\
\vdots \\
b_n
\end{pmatrix}
= 
\begin{pmatrix}
f_1 \\
f_2 \\
\vdots \\
f_n
\end{pmatrix}
\]

\[
\begin{pmatrix}
1 & \cdots & 0 \\
0 & \cdots & 1 \\
\vdots & \ddots & \vdots \\
0 & \cdots & 0
\end{pmatrix}
\begin{pmatrix}
c_1 \\
c_2 \\
\vdots \\
c_n
\end{pmatrix}
= 
\begin{pmatrix}
\frac{b_1}{2} + 2c_1 + 3d_1 \\
\frac{b_2}{2} + 2c_2 + 3d_2 \\
\vdots \\
\frac{b_n}{2} + 2c_n + 3d_n
\end{pmatrix}
\]

Side note: this linear system will generally be better conditioned if we instead use \( p_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \), but for the case of equispaced knots \( (h = b - a) \), it is not matter.
The resulting linear system for either natural or clamped cubic splines is diagonally dominant: \[ A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}, \quad |a_{ii}| > \sum_{j \neq i} |a_{ij}|. \]

Suppose \( Ax = 0 \) for some \( x \neq 0 \). Set \( k \) so that \( |x_k| = \|x\|_{\infty} \). Then \( |x_k| > 0 \), and looking at the \( k \)-th row of \( Ax = 0 \) we get:

\[
0 = \sum_{j \neq i} a_{ij} x_j \geq |a_{ik} x_k| - \sum_{j \neq i, j \neq k} |a_{ij} x_j| - \sum_{j \neq i} |a_{ij} x_j| = |a_{ik} x_k| - \sum_{j \neq i} |a_{ij} x_j| = |x_k| - \sum_{j \neq i} |a_{ij} x_j| = 0,
\]

contradiction by the triangle inequality.

Hence \( \ker(A) = \{0\} \), so \( A \) is invertible.

This proves existence & uniqueness of both natural & clamped cubic splines, as we've just shown that every diagonally dominant matrix is invertible, so we can uniquely solve all the coefficients \( \phi_i \) of the spline.

Also, the system for the \( c_i \)'s is tridiagonal in addition to being diagonally dominant. Thus it admits a handy LU-factorization as follows

\[
A = LU,
\]

where

\[
L = \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
l_2 & 1 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
l_n & l_{n-1} & l_{n-2} & \cdots & 1 \\
\end{pmatrix},
\]

\[
U = \begin{pmatrix}
\alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n} \\
0 & \alpha_{22} & \cdots & \alpha_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \alpha_{nn} \\
\end{pmatrix}.
\]

Note: If \( k \) is least such that \( d_k = 0 \), then \( A_k := \begin{pmatrix}
\alpha_{11} & \alpha_{12} & \cdots & \alpha_{1k} \\
0 & \alpha_{22} & \cdots & \alpha_{2k} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \alpha_{kk} \\
\end{pmatrix}\) is not invertible; but \( A_k \) is diagonally dominant, so it must be invertible; contradiction. Thus no \( d_k \) is \( 0 \), and the above algorithm for finding the LU-factorization works.

Then the linear system for the \( c_i \) coefficients of the cubic spline is of the form \( LU \boldsymbol{x} = \boldsymbol{b} \).

Set \( \tilde{\boldsymbol{x}} = \boldsymbol{U} \boldsymbol{x} \). Solve \( \tilde{\boldsymbol{L}} \tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}} \) for \( \tilde{\boldsymbol{x}} \) by back-substitution:

\[
\begin{pmatrix}
l_1 & 0 & 0 & \cdots & 0 \\
l_2 & l_1 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
l_n & l_{n-1} & l_{n-2} & \cdots & 1 \\
\end{pmatrix}
\begin{pmatrix}
w_1 \\
w_2 \\
\vdots \\
w_n \\
\end{pmatrix} =
\begin{pmatrix}
1 \\
l_2 \\
\vdots \\
l_n \\
\end{pmatrix}
\begin{pmatrix}
b_1 \\
b_2 \\
\vdots \\
b_n \\
\end{pmatrix},
\]

so

\[
w_k = b_k - l_{k-1} w_{k-1} \quad (\text{for } k = 2, \ldots, n).
\]

Then solve \( \boldsymbol{U} \boldsymbol{x} = \tilde{\boldsymbol{x}} \) for \( \boldsymbol{x} \) by back-substitution:

\[
\begin{pmatrix}
\alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n} \\
0 & \alpha_{22} & \cdots & \alpha_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \alpha_{nn} \\
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n \\
\end{pmatrix} =
\begin{pmatrix}
w_1 \\
w_2 \\
\vdots \\
w_n \\
\end{pmatrix},
\]

so

\[
x_k = w_k - d_k x_{k-1} \quad (\text{for } k = n-1, n-2, \ldots, 1).
\]

Normally, Gaussian elimination takes \( O(n^3) \) time (multiplications/divisions). But by the above process, solving a diagonally dominant tridiagonal system only takes about \( 5n \) multiplications/divisions, so the system is solved in \( O(n) \) time, a remarkable improvement!

Note: instead of doing this LU factorization, you can equivalently do Gaussian elimination on \( A \) and still solve the system in \( O(n) \) time, as \( A \) is tridiagonal.
Error in approximating functions by splines: if the cubic spline instead satisfied \( s(x) = f(x) \) (breaking continuity of the 2\textsuperscript{nd} derivative) is then the spline \( S \) over \([x_0, x_5]\) can be viewed as a Hermite interpolating polynomial, so using the Hermite interpolation error bound, we get:

\[
|\text{error}| \leq \frac{h^4}{16} \left( \max_{x \in [x_0, x_5]} |f^{(4)}(x)| \right)
\]

where \( h = \max_{i} x_i \). The error for natural \& clamped cubic splines is harder to show, but for clamped it's \( O(h^4) \) everywhere, and for natural it's \( O(h^3) \) near endpoints a & b and \( O(h^4) \) away from a & b.

The two types of cubic splines mentioned also have a “least curvature” property:

Suppose \( g \in C^2([a,b]) \), \( g(x_i) = f(x_i) \) \( i = 0, \ldots, n \), and \( s \) is the natural cubic spline for \( f \) with knots \( x_0, \ldots, x_n \). Then

\[
\int_a^b (s''(x))^2 \, dx \leq \int_a^b (g''(x))^2 \, dx,
\]

with equality if \( g = s \).

Proof: Set \( e(x) = g(x) - s(x) \). Then

\[
\int_a^b (e''(x))^2 \, dx = \sum_{i=1}^n \int_{x_{i-1}}^{x_i} (e''(x))^2 \, dx = \sum_{i=1}^n \left[ e(x_i) s''(x_i) - e(x_{i-1}) s''(x_{i-1}) \right] = e'(b)s''(b) - e'(a)s''(a) - \sum_{i=1}^n e(x_i) s''(x_i)
\]

\[
\implies -\sum_{i=1}^n e(x_i) s''(x_i) = -\sum_{i=1}^n \left[ e(x_i) s''(x_i) - e(x_{i-1}) s''(x_{i-1}) \right] = 0
\]

for natural cubic splines.

Thus

\[
\int_a^b (g''(x))^2 \, dx = \int_a^b (e''(x))^2 \, dx + \int_a^b (s''(x))^2 \, dx + \int_a^b (e''(x))^2 \, dx = \int_a^b (s''(x))^2 \, dx + \int_a^b (g''(x))^2 \, dx
\]

If \( e''(x) \neq 0 \) for some \( x \), then \( g \) is not the same as \( s \), and \( \int_a^b (e''(x))^2 \, dx \geq \int_a^b (s''(x))^2 \, dx \).

If \( e''(x) = 0 \), then by integrating, \( g(x) = ax + b + s(x) \). But then

\[
\int_a^b (g''(x))^2 \, dx < \int_a^b (s''(x))^2 \, dx
\]

If \( g''(x) = 0 \), then by integrating, \( g(x) = ax + b + s(x) \). But then

\[
\int_a^b (g''(x))^2 \, dx = \int_a^b (s''(x))^2 \, dx
\]

We can get a similar result for clamped splines by forcing \( g \) to also satisfy

\[
g'(a) = f'(a) \quad \text{and} \quad g'(b) = f'(b),
\]

since this makes \( e(x) \) true as well, since then

\[
e'(a) = g'(a) - s'(a) = f'(a) - f'(a) = 0
\]

and similarly for \( e'(b) = 0 \). There are many other types of splines, but we won't cover them here. They generalize Hermite polynomial interpolation, which already generalizes Lagrange interpolation & Taylor polynomials, and they don't suffer from problems with high-order polynomial interpolation associated with Runge's phenomenon. They are a rich and useful class of functions to study.