Polynomial interpolants are rarely the end product of a numerical process. Their importance is more as building blocks for other, more complex algorithms in differentiation, integration, solutions of differential equations, approximation theory in the large, . . . . Interpolation is often used both to design a algorithm and to analyze its convergence properties.
Interpolation is a special case of approximation.

- **Data fitting (discrete):** Given data \((x_i, y_i)\) find a reasonable function \(v(x)\) that fits the data. If the data are accurate it may make sense to interpolate the data with \(v(x)\).

- **Approximating functions (continuous):** For a complicated function \(f(x)\) find a simpler function \(v(x)\) that approximates \(f\).
10.1: General approximation and interpolation

Why do we need interpolating functions $v(x)$?

- For prediction: if $x$ is inside the domain of data points, we call $v(x)$ the interpolation value at $x$. Otherwise, we call it extrapolation.
- For manipulation: like finding approximate derivatives or integrals.
- For storage: It is usually easier to store $v(x)$, rather than the data points themselves.
We generally assume a *linear form* for all interpolating functions \( v(x) \). We write

\[
v(x) = \sum_{j=0}^{n} c_j \phi_j(x) = c_0 \phi_0(x) + \ldots + c_n \phi_n(x).
\]

where \( \{c_j\} \) are the unknown coefficients, or parameters, determined by the data, and \( \{\phi_j(x)\} \) are predetermined basis functions.
Our first goal in interpolation is to find the scalars \( \{c_j\} \).

Use the interpolating linear combination and data points to rewrite as a matrix equation.

Note that we assume that the number of basis functions \( n + 1 \) is the same as the number of data points.
There are many types of interpolation depending on the data.

- When \( \{\phi_j(x)\} \) are polynomials, we say polynomial interpolation.
- When \( \{\phi_j(x)\} \) are trigonometric, we say trigonometric interpolation.
- There is also piecewise interpolation.
- We will only consider polynomial interpolation in this course.
10.2: Monomial interpolation

- Use the polynomials $\phi_0(x) = 1$, $\phi_1(x) = x$ to approximate the data $(1, 1), (2, 3)$.

- Use the polynomials $\phi_0(x) = 1$, $\phi_1(x) = x$, $\phi_2(x) = x^2$ to approximate the data $(1, 1), (2, 3), (4, 3)$. 
10.2: Monomial interpolation

# ALGORITHM: Matrix interpolation p. 300
import numpy as np

A = np.array([[1,1,1],
              [1,2,4],
              [1,4,16]], float)

y = np.array([[1],
              [3],
              [3]], float) # initial condition

c = np.linalg.solve(A,y)
print(c)
10.2: Monomial interpolation

This method suggests a general way for obtaining the interpolating polynomial $p(x)$.

- Form the Vandermonde matrix and solve the linear system.
- The advantage of this approach is its simplicity.
- The disadvantage of this approach is that the Vandermonde matrix $X$ is often ill-conditioned.
- Moreover, this approach requires $\frac{2}{3}n^3$ operations. We may be able to do better?
10.3 Lagrange Interpolation

Use the lagrange polynomials

\[ L_j(x) = \frac{(x - x_0) \ldots (x - x_{j-1})(x - x_{j+1}) \ldots (x - x_n)}{(x_j - x_0) \ldots (x_j - x_{j-1})(x_j - x_{j+1}) \ldots (x_j - x_n)} \]

to interpolate in the previous examples.
Use Newton’s divided differences to interpolate in the previous examples (by hand). You will not be required to code the Newton’s divided differences algorithm in this course.
The error in polynomial interpolation is given by

\[ f(x) - p(x) = \frac{f^{(n+1)}(\xi)}{(n + 1)!} (x - x_0)(x - x_1) \ldots (x - x_n). \]

As it stands this formula is more or less useless in computing exact error since we can rarely find \( \xi \) and evaluate \( f^{(n+1)}(\zeta) \). This error formula can however often be used to bound the error in the interpolation. You should understand this error formula and understand its proof.
The error in polynomial interpolation is given by

\[ f(x) - p(x) = \frac{f^{(n+1)}(\xi)}{(n + 1)!} (x - x_0)(x - x_1) \ldots (x - x_n). \]

You may expect that the higher the degree, the more accurate the interpolating polynomial. This is not always true. Low order approximations are often reasonable. High degree interpolants, on the other hand, are expensive to compute and can be poorly behaved, especially near the endpoints. Chapter 11 “Piecewise Polynomial Interpolation” partially resolves these issues by interpolating in pieces with low degree polynomials.
Assume that the polynomial $p_9(x)$ interpolates the function $f(x) = e^{-2x}$ at the 10 evenly spaced points $x = 0, \frac{1}{9}, \frac{2}{9}, \ldots, \frac{8}{9}, 1$. Use the error formula to find an upper bound for the error $e_9(0.5) = |f(0.5) - p_9(0.5)|$. 
Find the Lagrange polynomials $L_j(x)$ and data $y_j$ to interpolate $f(x) = e^x$ at the $x$-values $x = 0$, $x = \frac{1}{2}$, and $x = 1$ by a quadratic polynomial $p(x)$. 
Find the interpolating polynomial for the data points \((0, 1), (2, 2), \) and \((3, 4)\) using

- Lagrange polynomials (section 10.3).
- Monomial interpolation (section 10.2).
- Divided differences (section 10.4).
- Verify that all answers above are the same polynomial.
EXERCISE

You should be able to redo the previous exercise with two data points or four data points. Try it!

▶ Data points: (2, 4), (5, 1).
▶ Data points: (−1, 3), (0, −4), (1, 5), (2, −6).
EXERCISE

Let $p(x)$ be the interpolating polynomial of the data points
$(1, 10), (2, 10), (3, 10), (4, 10), (5, 10)$, and $(6, 15)$. Evaluate $p(7)$. 
EXERCISE

- Assume \( p(x) \) interpolates \( f(x) = e^{-2x} \) at the 10 evenly spaced points \( x = 0, \frac{1}{9}, \frac{2}{9}, \ldots, \frac{8}{9}, 1 \). Find an upper bound for the error \( |f\left(\frac{1}{2}\right) - p\left(\frac{1}{2}\right)| \). How many decimal places can you guarantee to be correct if \( p(0.5) \) is used to approximate \( e \)?

- Consider the interpolating polynomial for \( f(x) = \frac{1}{x+5} \) with interpolation nodes \( x = 0, 2, 4, 6, 8, \) and 10. Find an upper bound for the interpolation error at \( x = 1 \) and then at \( x = 5 \).
Write code to compute the interpolation polynomial by divided differences. This is a good exercise in using data structures (the data points) and loops. Although it is not particularly challenging code to write, you may however consider this exercise as extra credit. I will never ask you to code it on an exam or homework quiz. We do not have enough time in the course.
Find the Lagrange polynomials $L_j(x)$ and data $y_j$ to interpolate $f(x) = e^x$ at the $x$-values $x = 0, x = \frac{1}{2},$ and $x = 1$ by a quadratic polynomial $p(x)$.

$\triangleright \quad p(x) = y_0L_0(x) + y_1L_1(x) + y_2L_2(x) = 1L_0(x) + e^{0.5}L_1(x) + eL_2(x)$.

$\triangleright \quad L_0(x) = \frac{(x-\frac{1}{2})(x-1)}{(0-\frac{1}{2})(0-1)}$

$\triangleright \quad L_1(x) = \frac{(x-0)(x-1)}{(\frac{1}{2}-0)(\frac{1}{2}-1)}$

$\triangleright \quad L_2(x) = \frac{(x-0)(x-\frac{1}{2})}{(1-0)(1-\frac{1}{2})}$
Find the interpolating polynomial for the data points \((0, 1), (2, 2),\) and \((3, 4)\) using

- Lagrange polynomials:
  \[
p(x) = 1 \frac{(x-2)(x-3)}{(0-2)(0-3)} + 2 \frac{(x-0)(x-3)}{(2-0)(2-3)} + 4 \frac{(x-0)(x-2)}{(3-0)(3-2)}.
  \]

- Monomial interpolation: solve
  \[
  \begin{bmatrix}
  1 & 0 & 0^2 \\
  1 & 2 & 2^2 \\
  1 & 3 & 3^2 \\
  \end{bmatrix}
  \begin{bmatrix}
  c_1 \\
  c_2 \\
  c_3 \\
  \end{bmatrix}
  =
  \begin{bmatrix}
  1 \\
  2 \\
  4 \\
  \end{bmatrix}.
  \]

- \[p(x) = 1 + \frac{1}{2}(x - 0) + \frac{1}{2}(x - 0)(x - 2)\].
- reduce all polynomials above to form \(y = ax^2 + bx + c\).
You should be able to redo the previous exercise with two data points or four data points. Try it!

- Data points: \((2, 4), (5, 1)\). Then \(p(x) = -x + 6\) in all three methods.

- Data points: \((-1, 3), (0, -4), (1, 5), (2, -6)\). Then \(p(x) = -6x^3 + 8x^2 + 7x - 4\).

- The problem above with four data points illustrates that on an exam there will never be more than four data points. It is too much computation.
Let $p(x)$ be the interpolating polynomial of the data points $(1,10), (2,10), (3,10), (4,10), (5,10)$, and $(6,15)$. Evaluate $p(7)$. 

ANSWER: $p(7) = 40$. 
ANSWERS

- Assume \( p(x) \) interpolates \( f(x) = e^{-2x} \) at the 10 evenly spaced points \( x = 0, \frac{1}{9}, \frac{2}{9}, \ldots, \frac{8}{9}, 1 \). An upper bound for the error \( |f\left(\frac{1}{2}\right) - p\left(\frac{1}{2}\right)| < 7.06 \times 10^{-11} \). How many decimal places can you guarantee to be correct if \( p(0.5) \) is used to approximate \( e \)? ANSWER: 9 places.

- Consider the interpolating polynomial for \( f(x) = \frac{1}{x+5} \) with interpolation nodes \( x = 0, 2, 4, 6, 8, \) and \( 10 \). Find an upper bound for the interpolation error at \( x = 1 \) (ANSWER: \( \frac{3 \cdot 5 \cdot 7 \cdot 9}{5^3} \)) and then at \( x = 5 \) (ANSWER: \( \frac{5^2 \cdot 3^2}{5^3} \)). There are other acceptable answers.