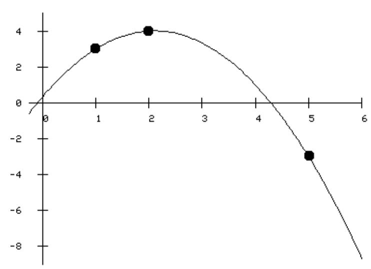
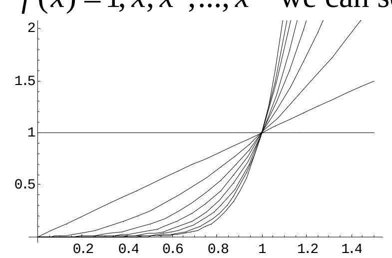
CS205 – Class 13

Covered in class: 1, 3, 5
Readings: 6.7, 7.2 to 7.3.3

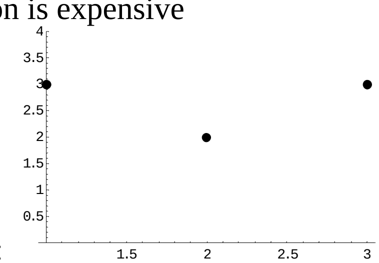
1. Interpolation
   a. **polynomial of degree n** $y = c_1 + c_2 x + c_3 x^2 + \cdots + c_{n+1} x^n$
      i. **Monomial basis** – $y = c_1\phi_1 + c_2\phi_2 + c_3\phi_3 + \cdots + c_{n+1}\phi_{n+1}$ where the basis function are $\phi_j(x) = x^{j-1}$ for $j = 1, 2, \cdots, n+1$
         1. **Polynomial interpolation** given a set of $n+1$ points (x_i, y_i), find the unique n degree polynomial $y = c_1 + c_2 x + c_3 x^2 + \cdots + c_{n+1} x^n$ that interpolates them.
         2. Solve Ax = y where A is the $(n+1)\times(n+1)$ **Vandermonde matrix** with rows $(1, x_i, x_i^2, \cdots, x_i^n)$ for each data point (x_i, y_i)
         3. Example: i.e. for (1,3), (2,4), (5,-3) for quadratic we would have $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 5 & 25 \end{pmatrix}\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ -3 \end{pmatrix}$ so we get $\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1/3 \\ 7/2 \\ -5/6 \end{pmatrix}$ and thus the equation is $f(x) = \frac{1}{3} + \frac{7}{2}x - \frac{5}{6}x^2$ which looks like:

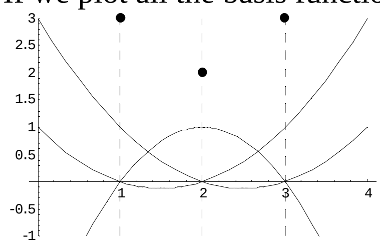


         4. But this is not an ideal basis, because as polynomials get higher, the functions have lots of overlap. Plotting $f(x) = 1, x, x^2, \ldots, x^8$ we can see this effect:



      ii. **Lagrange interpolation** $y = c_1\phi_1 + c_2\phi_2 + c_3\phi_3 + \cdots + c_{n+1}\phi_{n+1}$ with basis functions $\phi_j(x) = \dfrac{\prod_{k \neq j}(x - x_k)}{\prod_{k \neq j}(x_j - x_k)}$ for $j = 1, 2, \cdots, n+1$

1. $\phi_j(x_j) = 1$ and $\phi_j(x_k) = 0$ where $k \neq j$ – therefore the coefficients are $c_j = y_j$ (easy to compute)
2. No "overlap" problem
3. Evaluation is expensive
4. Example:  and we get $\phi_1(x) = \frac{(x-2)(x-3)}{(1-2)(1-3)} = \frac{1}{2}(x-2)(x-3)$ and we have $\phi_1(1) = 1, \phi_1(2) = 0, \phi_1(3) = 0$. We have that the polynomial is zero at each of the other data points. And it is one at the point. If we plot all the basis functions we see



iii. **Newton interpolation** $y = c_1\phi_1 + c_2\phi_2 + c_3\phi_3 + \cdots + c_{n+1}\phi_{n+1}$ with basis functions $\phi_j(x) = \prod_{k=1}^{j-1}(x - x_k)$ for $j = 1, 2, \cdots, n+1$

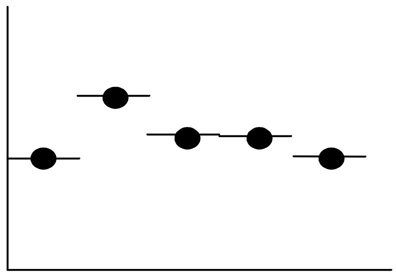
1. No "overlap" problem
2. **Divided differences** We initially set $f[x_k] = y_k$ at the first level. Then the higher levels are based on
$$f[x_1, x_2, \cdots, x_k] = \frac{f[x_2, x_3, \cdots, x_k] - f[x_1, x_2, \cdots, x_{k-1}]}{x_k - x_1}$$
3. The coefficients are given by $c_j = f[x_1, x_2, \cdots, x_j]$
4. Represents a compromise between Lagrange and monomial basis.

iv. High order polynomials tend to be oscillatory

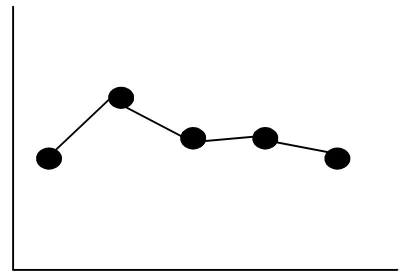
v. Using unequal data points can help, e.g. Chebyshev points

2. A better solution is to use **piecewise polynomials** – a different polynomial in each subinterval $[x_i, x_{i+1}]$
   a. Defined using **control points** (x_i, y_i)
   b. **Piecewise linear** – connect the control points with straight lines

3. A better solution is to use **piecewise polynomials** – a different polynomial in each subinterval $[x_i, x_{i+1}]$
   a. Simplest is piecewise constant, next is linear. As order increases number of points required increases but the accuracy also increases.
   b. One could argue you should go to even higher order.



O(Δx) Piecewise constant
Piecewise constant

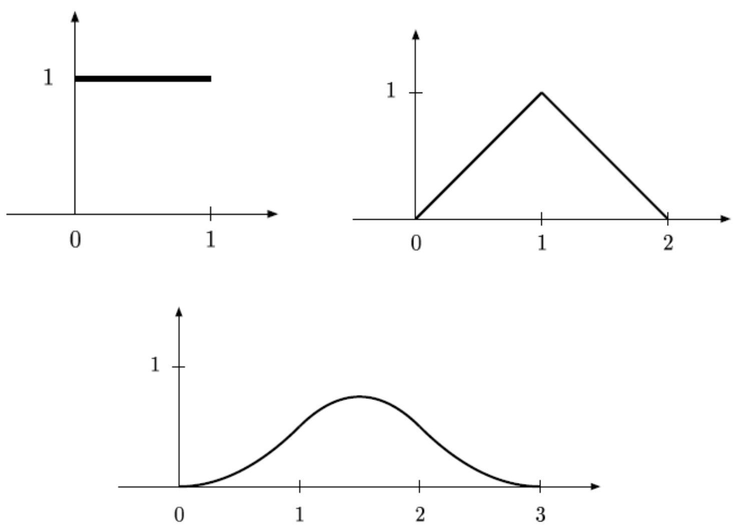


O(Δx^2) Piecewise constant
Linear

   c. Higher order is not necessarily better
      i. Once you go to spectral you get infinite accuracy, better than any polynomial, but you need to pay with requiring more data and Gibb's phenomena. There is some smoothness assumption about the function you are interpolating.
      ii. In practice higher order methods overly smooth discontinuous phenomena. So they are good for smoother phenomena like simulating tree sap. However, if you had a turbulent flow the interesting and important discontinuities would get destroyed.
4. Spline interpolation, little detail here, but for more information Prof. Guibas teaches a course on it. (not covered in class)
   a. Defined using **control points** (x_i, y_i)
   b. **Piecewise linear** – connect the control points with straight lines
   c. **Hermite interpolation** – specify the function values y_i and the derivatives y'_i at each control point
      i. **Hermite cubic** – cubic polynomial on each subinterval $[x_i, x_{i+1}]$
      ii. If there are n control points and $n-1$ intervals, then there are $n-1$ cubics
      iii. We need to specify $4(n-1)$ parameters, i.e. 4 parameters for each cubic
      iv. Interpolating the function values y_i gives $2(n-1)$ conditions, i.e. 2 for each subinterval
      v. Requiring the derivative to be continuous is $n-2$ conditions, one for each interior control point
      vi. $4(n-1)-2(n-1)-(n-2)=n$ more conditions need to be specified
   d. **Spline** a piecewise polynomial of degree k that is differentiable k-1 times
   e. **Cubic spline** continuous 1st and 2nd derivatives at the control points

        i. $2(n-1)$ conditions to interpolate the y_i
        ii. $n-2$ conditions for continuous 1st derivatives
        iii. $n-2$ conditions for continuous 2nd derivatives
        iv. Total of $4n-6$ conditions – we need 2 more conditions
            1. **Hermite cubic spline** specify the 1st derivative at x_1 and x_n (endpoints)
            2. **Periodic cubic spline** forcing the 1st and 2nd derivatives to match at x_1 and x_n
            3. **Natural cubic spline** set the 2nd derivative to zero at x_1 and x_n
            4. Set up the equations and solve

f. **B splines** the basis function B_i^k is a piecewise polynomial of degree k
    i. Piecewise constant $B_i^0(x)=1$ for $x \in [x_i, x_{i+1})$ and 0 otherwise
    ii. Recursively $B_i^k(x)=v_i^k(x)B_i^{k-1}(x)+(1-v_{i+1}^k(x))B_{i+1}^{k-1}(x)$ with linear functions $v_i^k(x)=\dfrac{x-x_i}{x_{i+k}-x_i}$
    iii. B_i^1 are piecewise linear, B_i^2 are piecewise quadratic, B_i^3 are piecewise cubic, etc.



The B^0, B^1, B^2 polynomials