CSE 541 - Interpolation

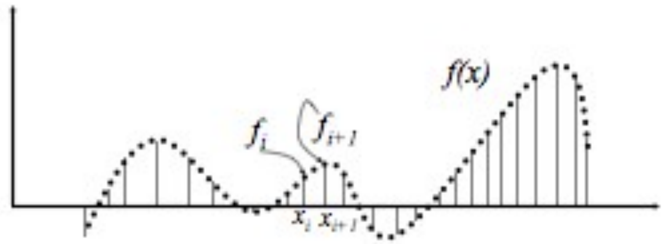
Roger Crawfis

Taylor's Series and Interpolation

- Taylor Series interpolates at a specific point:
  - The function
  - Its first derivative
  - …
- It may not interpolate at other points.
- We want an interpolant at several *f(c)*'s.

Basic Scenario

- We are able to *prod* some **function**, but do not know what it really is.
- This gives us a list of data points: $[x_i, f_i]$



Interpolation & Curve-fitting

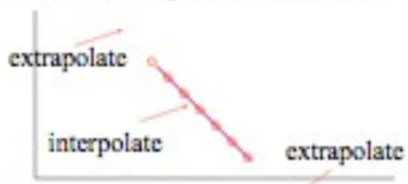
- Often, we have data sets from experimental/observational measurements
  - Typically, find that the data/dependent variable/output varies…
  - As the control parameter/independent variable/input varies. Examples:
    - Classic gravity drop: location changes with time
    - Pressure varies with depth
    - Wind speed varies with time
    - Temperature varies with location
- Scientific method: Given data identify underlying relationship
- Process known as curve fitting:

Interpolation & Curve-fitting

- Given a data set of n+1 points (x_i,y_i) identify a function f (x) (the curve), that is in some (well-defined) sense the best fit to the data
- Used for:
  - Identification of underlying relationship (modelling/ prediction)
  - Interpolation (filling in the gaps)
  - Extrapolation (predicting outside the range of the data)

Interpolation Vs Regression

- Distinctly different approaches depending on the quality of the data
- Consider the pictures below:



Pretty confident:
there is a polynomial relationship
Little/no scatter
Want to find an expression
that passes exactly through all the points

Unsure what the relationship is
Clear scatter
Want to find an expression
that captures the trend:
minimize some measure of the error
Of all the points…

Interpolation

- Concentrate first on the case where we believe there is no error in the data (and round-off is assumed to be negligible).
- So we have $y_i=f(x_i)$ at n+1 points $x_0, x_1 \ldots x_i, \ldots x_n$: $x_j > x_{j-1}$
- (Often but not always evenly spaced)
- In general, we do not know the underlying function $f(x)$
- Conceptually, interpolation consists of two stages:
  - Develop a simple function $g(x)$ that
    - Approximates $f(x)$
    - Passes through all the points x_i
  - Evaluate $f(x_t)$ where $x_0 < x_t < x_n$

Interpolation

- Clearly, the crucial question is the selection of the simple functions $g(x)$
- Types are:
  - Polynomials
  - Splines
  - Trigonometric functions
  - Spectral functions…Rational functions etc…

Curve Approximation

- We will look at three possible approximations (time permitting):
  - Polynomial interpolation
  - Spline (polynomial) interpolation
  - Least-squares (polynomial) approximation
- If you know your function is periodic, then trigonometric functions may work better.
  - Fourier Transform and representations

Polynomial Interpolation

- Consider our data set of n+1 points $y_i=f(x_i)$ at n+1 points $x_0, x_1 \ldots x_i, \ldots x_n$: $x_j > x_{j-1}$
- In general, given n+1 points, there is a unique polynomial $g_n(x)$ of order n:

$$g_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

- That passes through all n+1 points

Polynomial Interpolation

- There are a variety of ways of expressing the same polynomial
- Lagrange interpolating polynomials
- Newton's divided difference interpolating polynomials
- We will look at both forms

Polynomial Interpolation

- Existence – does there exist a polynomial that **exactly** passes through the n data points?
- Uniqueness – Is there more than one such polynomial?
  - We will assume uniqueness for now and prove it latter.

Lagrange Polynomials

- Summation of terms, such that:
  - Equal to *f()* at a data point.
  - Equal to zero at all other data points.
  - Each term is a n^{th}-degree polynomial

Existence!!!

$$p_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i)$$

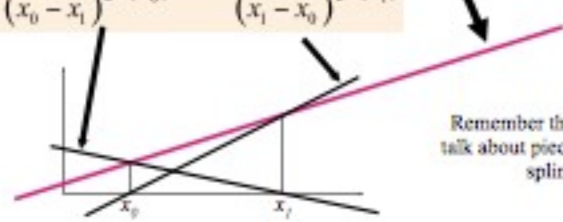
$$L_i(x) = \prod_{k=0, k \neq i}^{n} \frac{(x - x_k)}{(x_i - x_k)}$$

$$L_i(x_j) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

Linear Interpolation

- Summation of two lines:

$$p_1(x) = \sum_{i=0}^{1} L_i(x) f(x_i) = \frac{(x - x_1)}{(x_0 - x_1)} f(x_0) + \frac{(x - x_0)}{(x_1 - x_0)} f(x_1)$$

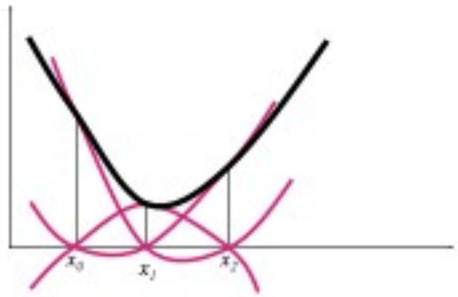


Remember this when we talk about piecewise-linear splines

Lagrange Polynomials

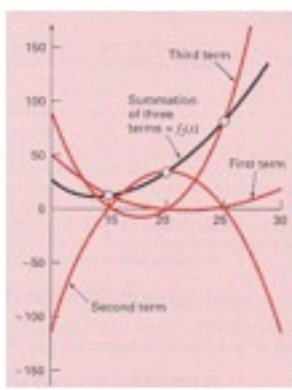
- 2^{nd} Order Case => quadratic polynomials

Adding them all together, we get the interpolating quadratic polynomial, such that:
- $P(x_0) = f_0$
- $P(x_1) = f_1$
- $P(x_2) = f_2$



Lagrange Polynomials

- Sum must be a unique 2^{nd} order polynomial through all the data points.
- What is an efficient implementation?



Newton Interpolation

- Consider our data set of n+1 points $y_i = f(x_i)$ at $x_0, x_1 \ldots x_i, \ldots x_n$: $x_n > x_0$
- Since $p_n(x)$ is the unique polynomial $p_n(x)$ of order n, write it:

$$p_n(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + \ldots + b_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

$$b_0 = f(x_0)$$

$$b_1 = f[x_1, x_0] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

$$b_2 = f[x_2, x_1, x_0] = \frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0}$$

$$\vdots$$

$$b_n = f[x_n, x_{n-1}, \ldots, x_0] = \frac{f[x_n, \ldots, x_1] - f[x_{n-1}, \ldots, x_0]}{x_n - x_0}$$

- $f[x_i, x_j]$ is a first divided difference

Invariance Theorem

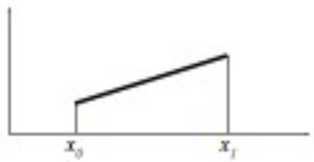
- Note, that the order of the data points does not matter.
- All that is required is that the data points are distinct.
- Hence, the divided difference $f[x_0, x_1, \ldots, x_k]$ is invariant under all permutations of the x_i's.

Linear Interpolation

- Simple linear interpolation results from having only 2 data points.

$$p_1(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0)$$

slope



Quadratic Interpolation

- Three data points:

$$p_2(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1)$$

$$= f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0) + \frac{\left[\frac{f(x_2) - f(x_1)}{x_2 - x_1}\right] - \left[\frac{f(x_1) - f(x_0)}{x_1 - x_0}\right]}{x_2 - x_0}(x - x_0)(x - x_1)$$

$$= f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0)$$

$$+ \frac{\left(\left[\frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1)\right](x - x_0)\right) - \left(\left[\frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0)\right](x - x_1)\right)}{x_2 - x_0}$$

Newton Interpolation

- Let's look at the recursion formula:

$$b_n = f[x_n, x_{n-1}, \ldots, x_0] = \frac{f[x_n, \ldots, x_1] - f[x_{n-1}, \ldots, x_0]}{x_n - x_0}$$

where

$$f[x_i] = f(x_i)$$

- For the quadratic term:

$$b_2 = f[x_2, x_1, x_0] = \frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0} = \frac{\frac{f(x_2) - f(x_1)}{x_2 - x_1} - \frac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}$$

$$= \frac{\frac{f(x_2) - f(x_1)}{x_2 - x_1} - b_1}{x_2 - x_0}$$

Evaluating for x_2

$$f(x_2) = b_0 + b_1(x_2 - x_0) + b_2(x_2 - x_0)(x_2 - x_1)$$

$$= f_0 + b_1(x_2 - x_0) + \left(\frac{\frac{f_2 - f_1}{x_2 - x_1} - b_1}{x_2 - x_0}\right)(x_2 - x_0)(x_2 - x_1)$$

$$= f_0 + b_1(x_1 - x_0) + f_2 - f_1$$

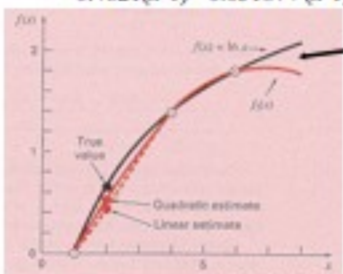
$$= f_0 + \left(\frac{f_1 - f_0}{x_1 - x_0}\right)(x_1 - x_0) + f_2 - f_1$$

$$= f_2$$

Example: ln(x)

- Interpolation of ln(2): given ln(1); ln(4) and ln(6)
  - Data points: {(1,0), (4,1.3863), (6,1.79176)}
  - Linear Interpolation: $0 + \{(1.3863-0)/(4-1)\}(x-1) = 0.4621(x-1)$
  - Quadratic Interpolation: $0.4621(x-1)+((0.20273-0.4621)/5)(x-1)(x-4)$
    $= 0.4621(x-1) - 0.051874\,(x-1)(x-4)$

corrected



Note the divergence for values outside of the data range.

Example: ln(x)

- Quadratic interpolation catches some of the curvature
- Improves the result somewhat
- Not always a good idea: see later…

Calculating the Divided-Differences

- A *divided-difference* table can easily be constructed incrementally.
- Consider the function *ln(x)*.

x	ln(x)	
1	0.000000	[redacted]
2	0.693147	
3	1.098612	
4	1.386294	
5	1.609438	
6	1.791759	
7	1.945910	
8	2.079442	
x	ln(x)	

Calculating the Divided-Differences

x	ln(x)	f[i,i+1]	
1	0.000000		[redacted]
2	0.693147	0.693147	
3	1.098612	0.405465	
4	1.386294	0.287682	
5	1.609438	0.223144	
6	1.791759	0.182322	
7	1.945910	0.154151	
8	2.079442	0.133531	
x	ln(x)	[illegible]	

$$f[i,i+1] = \frac{f(x_{i+1}) - f(x_i)}{(x_{i+1} - x_i)}$$

Calculating the Divided-Differences

x	ln(x)	f[i,i+1]	
1	**0.000000**		
2	0.693147	**0.693147**	
3	1.098612	0.405465	**-0.143841**
4	1.386294	0.287682	-0.058892
5	1.609438	0.223144	-0.032269
6	1.791759	0.182322	-0.020411
7	1.945910	0.154151	-0.014085
8	2.079442	0.133531	-0.010310
x	ln(x)	[illegible]	[illegible]

$$f[i,i+1,i+2]=\frac{f[i+1,i+2]-f[i,i+1]}{(x_{i+2}-x_i)}$$

[redacted]

Calculating the Divided-Differences

$$f[i,\ldots,i+3]=\frac{f[i+1,i+2,i+3]-f[i,i+1,i+2]}{(x_{i+3}-x_i)}$$

x	ln(x)	f[i,i+1]		
1	**0.000000**			
2	0.693147	**0.693147**		
3	1.098612	0.405465	**-0.143841**	
4	1.386294	0.287682	-0.058892	**0.028317**
5	1.609438	0.223144	-0.032269	0.008874
6	1.791759	0.182322	-0.020411	0.003953
7	1.945910	0.154151	-0.014085	0.002109
8	2.079442	0.133531	-0.010310	0.001259
x	ln(x)	[illegible]	[illegible]	[illegible]

[redacted]

Calculating the Divided-Differences

$$f[i,\ldots,i+4] = \frac{f[i+1,\ldots,i+4] - f[i,\ldots,i+3]}{(x_{i+4} - x_i)}$$

x	ln(x)	f[i,i+1]			
1	**0.000000**				
2	0.693147	**0.693147**			
3	1.098612	0.405465	**-0.143841**		
4	1.386294	0.287682	-0.058892	**0.028317**	
5	1.609438	0.223144	-0.032269	0.008874	**-0.004861**
6	1.791759	0.182322	-0.020411	0.003953	-0.001230
7	1.945910	0.154151	-0.014085	0.002109	-0.000461
8	2.079442	0.133531	-0.010310	0.001259	-0.000212
x	ln(x)	[illegible]	[illegible]	[illegible]	[illegible]

[redacted]

Calculating the Divided-Differences

$$f[i,\ldots,i+5] = \frac{f[i+1,\ldots,i+5] - f[i,\ldots,i+4]}{(x_{i+5} - x_i)}$$

x	ln(x)	f[i,i+1]				
1	**0.000000**					
2	0.693147	**0.693147**				
3	1.098612	0.405465	**-0.143841**			
4	1.386294	0.287682	-0.058892	**0.028317**		
5	1.609438	0.223144	-0.032269	0.008874	**-0.004861**	
6	1.791759	0.182322	-0.020411	0.003953	-0.001230	**0.00072**
7	1.945910	0.154151	-0.014085	0.002109	-0.000461	0.000154
8	2.079442	0.133531	-0.010310	0.001259	-0.000212	0.00005
x	ln(x)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

[redacted]

Calculating the Divided-Differences

$$f[i,\ldots,i+6]=\frac{f[i+1,\ldots,i+6]-f[i,\ldots,i+5]}{(x_{i+6}-x_i)}$$

x	ln(x)	f[i,i+1]						[redacted]
1	**0.000000**							
2	0.693147	**0.693147**						
3	1.098612	0.405465	**-0.143841**					
4	1.386294	0.287682	-0.058892	**0.028317**				
5	1.609438	0.223144	-0.032269	0.008874	**-0.004861**			
6	1.791759	0.182322	-0.020411	0.003953	-0.001230	**0.000726**		
7	1.945910	0.154151	-0.014085	0.002109	-0.000481	0.000154	**-0.000096**	
8	2.079442	0.133531	-0.010310	0.001259	-0.000212	0.000050	-0.000017	
x	ln(x)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	

Calculating the Divided-Differences

- Finally, we can calculate the last coefficient.

$$f[i,\ldots,i+7]=\frac{f[i+1,\ldots,i+7]-f[i,\ldots,i+6]}{(x_{i+7}-x_i)}$$

x	ln(x)	f[i,i+1]						f[i,i+1,…,i+7]
1	**0.000000**							
2	0.693147	**0.693147**						
3	1.098612	0.405465	**-0.143841**					
4	1.386294	0.287682	-0.058892	**0.028317**				
5	1.609438	0.223144	-0.032269	0.008874	**-0.004861**			
6	1.791759	0.182322	-0.020411	0.003953	-0.001230	**0.000726**		
7	1.945910	0.154151	-0.014085	0.002109	-0.000481	0.000154	**-0.000096**	
8	2.079442	0.133531	-0.010310	0.001259	-0.000212	0.000050	-0.000017	**0.000011**
x	ln(x)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	(g10-g9)/(a10-a2)

Calculating the Divided-Differences

- All of the coefficients for the resulting polynomial are in bold.

b_0 b_4 b_7

x	ln(x)	f[i,i+1]						f[i,i+1,...,i+7]
1	**0.000000**							
2	0.693147	**0.693147**						
3	1.098612	0.405465	**-0.143841**					
4	1.386294	0.287682	-0.058892	**0.028317**				
5	1.609438	0.223144	-0.032269	0.008874	**-0.004861**			
6	1.791759	0.182322	-0.020411	0.003953	-0.001230	**0.000726**		
7	1.945910	0.154151	-0.014085	0.002109	-0.000461	0.000154	**-0.000095**	
8	2.079442	0.133531	-0.010310	0.001259	-0.000212	0.000050	-0.000017	**0.000011**
x	ln(x)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Polynomial Form for Divided-Differences

- The resulting polynomial comes from the divided-differences and the corresponding product terms:

$$
\begin{aligned}
p_7(x) = 0 \\
&+0.693(x-1) \\
&-0.144(x-1)(x-2) \\
&+0.28(x-1)(x-2)(x-3) \\
&-0.0049(x-1)(x-2)(x-3)(x-4) \\
&+7.26\bullet 10^{-4}(x-1)(x-2)(x-3)(x-4)(x-5) \\
&-9.5\bullet 10^{-5}(x-1)(x-2)(x-3)(x-4)(x-5)(x-6) \\
&+1.1\bullet 10^{-5}(x-1)(x-2)(x-3)(x-4)(x-5)(x-6)(x-7)
\end{aligned}
$$

Many polynomials

- Note, that the order of the numbers (x_i,y_i)'s only matters when writing the polynomial down.
  - The first column represents the set of linear splines between two adjacent data points.
  - The second column gives us quadratics thru three adjacent points.
  - Etc.

Adding an Additional Data Point

- Adding an additional data point, simply adds an additional term to the existing polynomial.
  - Hence, only n additional divided-differences need to be calculated for the $n+1^{st}$ data point.

x	ln(x)	f[i,i+1]						f[i,i+1,...,i+7]	
1.0000000	**0.0000000**								
2.0000000	0.6931472	**0.6931472**							
3.0000000	1.0986123	0.4054651	**-0.1438410**						
4.0000000	1.3862944	0.2876821	-0.0588915	**0.0283165**					
5.0000000	1.6094379	0.2231436	-0.0322990	0.0088741	**-0.0048806**				
6.0000000	1.7917595	0.1823216	-0.0204110	0.0039528	-0.0012303	**0.0007261**			
7.0000000	1.9459101	0.1541507	-0.0140854	0.0021085	-0.0004611	0.0001539	**-0.0000954**		
8.0000000	2.0794415	0.1335314	-0.0103096	0.0012586	-0.0002126	0.0000467	-0.0000174	**0.0000111**	
1.5000000	0.4054651	0.2575348	-0.0225461	0.0027192	-0.0004173	0.0000819	-0.0000215	0.0000065	-0.0000050

b_8

Adding More Data Points

- Quadratic interpolation:
  - does linear interpolation
  - Then add higher-order correction to catch the curvature
- Cubic, …
- Consider the case where the data points are organized such the the first two are the endpoints, the next point is the mid-point, followed by successive mid-points of the half-intervals.
  - Worksheet: $f(x)=x^2$ from -1 to 3.

Uniqueness

- Suppose that two polynomials of degree n (or less) existed that interpolated to the n+1 data points.
- Subtracting these two polynomials from each other also leads to a polynomial of at most n degree.

$$r_n(x) = p_n(x) - q_n(x)$$

Uniqueness

- Since p and q both interpolate the $n+1$ data points,
- This polynomial r, has at least n+1 roots!!!
- This can not be! A polynomial of degree-n can only have at most n roots.
- Therefore, ***r(x)* = 0**

$$p_n(x) = a_n \prod_{i=1}^{n} (x - r_i)$$

$$p_{n+1}(x) = a_{n+1} \prod_{i=1}^{n+1} (x - r_i)$$

Example

- Suppose f was a polynomial of degree m, where $m<n$.
- Ex: $f(x) = 3x-2$
- We have evaluations of $f(x)$ at five locations: (-2,-8), (-1,-5), (0,-2), (1,1), (2,4)

Error

- Define the error term as:

$$\varepsilon_n(x) = f(x) - p_n(x)$$

- If $f(x)$ is an n^th order polynomial $p_n(x)$ is of course exact.
- Otherwise, since there is a perfect match at $x_0, x_1, \ldots, x_n$
- This function has at least $n+1$ roots at the interpolation poir

$$\therefore \varepsilon_n(x) = (x - x_0)(x - x_1)\cdots(x - x_n)h(x)$$

Interpolation Errors

$$\varepsilon_n(x) = f(x) - p_n(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi) \prod_{i=0}^{n} (x - x_i)$$

$$x \in [a,b], \xi \in (a,b)$$

- Proof is in the book.
- Intuitively, the first $n+1$ terms of the Taylor Series is also an n^{th} degree polynomial.

Interpolation Errors

- Use the point x, to expand the polynomial.

$$x \notin \{x_0, x_1, \ldots x_n\}$$

$$\varepsilon_n(x) = f(x) - p_n(x) = f[x_0, x_1, \ldots x_n, x] \prod_{i=0}^{n} (x - x_i)$$

- Point is, we can take an arbitrary point x, and create an $(n+1)^{th}$ polynomial that goes thru the point x.

Interpolation Errors

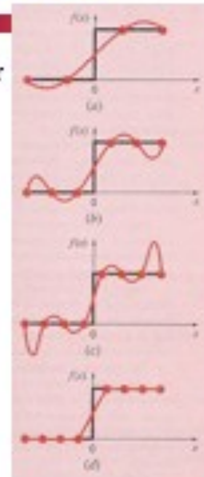
- Combining the last two statements, we can also get a feel for what these divided differences represent.

$$f[x_0, x_1, \ldots x_n] = \frac{1}{n!} f^{(n)}(\xi)$$

- Corollary 1 in book – If $f(x)$ is a polynomial of degree $m<n$, then all $(m+1)^{st}$ divided differences and higher are zero.

Problems with Interpolation

- Is it always a good idea to use higher and higher order polynomials?
- Certainly not: 3-4 points usually good: 5-6 ok:
- See tendency of polynomial to "wiggle"
- Particularly for sharp edges: see figures



Chebyshev nodes

- Equally distributed points may not be the optimal solution.
- If you could select the x_i's, what would they be?
- Want to minimize the $\prod_{i=0}^{n}(x-x_i)$ term.
- These are the Chebyshev nodes.
  - For x=-1 to 1: $x_i = \cos\left[\left(\frac{i}{n}\right)\pi\right],\ (0 \le i \le n)$

Chebyshev nodes

- Let's look at these for n=4.
- Spreads the points out in the center.

$$x_0 = \cos\left[\left(\frac{0}{4}\right)\pi\right] = 1$$

$$x_1 = \cos\left[\left(\frac{1}{4}\right)\pi\right] = \frac{\sqrt{2}}{2} = 0.707$$

$$x = \cos\left[\left(\frac{2}{4}\right)\pi\right] = 0$$

$$x_3 = \cos\left[\left(\frac{3}{4}\right)\pi\right] = -\frac{\sqrt{2}}{2} = -0.707$$

$$x_4 = \cos\left[\left(\frac{4}{4}\right)\pi\right] = -1$$

Polynomial Interpolation in Two-Dimensions

- Consider the case in higher-dimensions.

Finding the Inverse of a Function

- What if I am after the inverse of the function *f(x)*?
  - For example arccos(x).
- Simply reverse the role of the x_i and the f_i.