The Idea: Given the data set \((\tilde{x}, \tilde{f})\), where \(\tilde{x} = \{x_0, x_1, \ldots, x_n\}^T\) and \(\tilde{f} = \{f_0, f_1, \ldots, f_n\}^T\) we want to fit a simple model (usually a low degree polynomial, \(p_m(x)\)) to this data.

We seek the polynomial, of degree \(m\), which minimizes the residual:

\[
    r(\tilde{x}) = \sum_{i=0}^{n} [p_m(x_i) - f(x_i)]^2.
\]

We find the polynomial by differentiating the sum with respect to the coefficients of \(p_m(x)\). — If we are fitting a fourth degree polynomial \(p_4(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4\), we must compute the partial derivatives wrt. \(a_0, a_1, a_2, a_3, a_4\).

In order to achieve a minimum, we must set all these partial derivatives to zero. — In this case we get 5 equations, for the 5 unknowns; the system is known as the normal equations.
The Normal Equations — Second Derivation

Last time we showed that the normal equations can be found with purely a Linear Algebra argument. Given the data points, and the model (here $p_4(x)$), we write down the over-determined system:

$$
\begin{align*}
0 + a_1 x_0 + a_2 x_0^2 + a_3 x_0^3 + a_4 x_0^4 &= f_0 \\
0 + a_1 x_1 + a_2 x_1^2 + a_3 x_1^3 + a_4 x_1^4 &= f_1 \\
& \vdots \\
0 + a_1 x_n + a_2 x_n^2 + a_3 x_n^3 + a_4 x_n^4 &= f_n.
\end{align*}
$$

We can write this as a matrix-vector problem:

$$
X \tilde{a} = \tilde{f},
$$

where the Vandermonde matrix $X$ is tall and skinny. By multiplying both the left- and right-hand-sides by $X^T$ (the transpose of $X$), we get a “square” system — we recover the normal equations:

$$
X^T X \tilde{a} = X^T \tilde{f}.
$$

Example: Fitting $p_i(x)$, $i = 0, 1, 2, 3, 4$ Models.

Figure: We revisit the example from last time; and fit polynomials up to degree four to the given data. The figure shows the best $p_0(x)$, $p_1(x)$, and $p_2(x)$ fits.

Below: the errors give us clues when to stop.

<table>
<thead>
<tr>
<th>Model</th>
<th>Sum-of-squares-error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_0(x)$</td>
<td>205.45</td>
</tr>
<tr>
<td>$p_1(x)$</td>
<td>52.38</td>
</tr>
<tr>
<td>$p_2(x)$</td>
<td>51.79</td>
</tr>
<tr>
<td>$p_3(x)$</td>
<td>51.79</td>
</tr>
<tr>
<td>$p_4(x)$</td>
<td>51.79</td>
</tr>
</tbody>
</table>

Table: Clearly in this example there is very little to gain in terms of the least-squares-error by going beyond 1st or 2nd degree models.


Given the data set $(\tilde{x}, \tilde{f})$, where $\tilde{x} = \{x_0, x_1, \ldots, x_n\}$ and $\tilde{f} = \{f_0, f_1, \ldots, f_n\}$, we can quickly find the best polynomial fit for any specified polynomial degree!

Notation: Let $\tilde{x}^j$ be the vector $\{x_0^j, x_1^j, \ldots, x_n^j\}$.

E.g. to compute the best fitting polynomial of degree 4, $p_4(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4$, define:

$$
X = \begin{bmatrix}
1 & \tilde{x}^2 & \tilde{x}^3 & \tilde{x}^4 \\
\end{bmatrix}, 
\quad \text{and compute } \tilde{a} = (X^T X)^{-1}(X^T \tilde{f}).
$$

Not like this! See math 543!

Introduction: Defining the Problem.

Up until now: Discrete Least Squares Approximation applied to a collection of data.

Now: Least Squares Approximation of Functions.

We consider problems of this type: —

Suppose $f \in C[a, b]$ and we have the class $\mathcal{P}_n$ which is the set of all polynomials of degree at most $n$. Find the $p(x) \in \mathcal{P}_n$ which minimizes

$$
\int_a^b [p(x) - f(x)]^2 \, dx.
$$
Finding the Normal Equations...

If \( p(x) \in \mathcal{P}_n \) we write \( p(x) = \sum_{k=0}^{n} a_k x^k \). The sum-of-squares-error, as function of the coefficients, \( \tilde{a} = \{a_0, a_1, \ldots, a_n\} \) is:

\[
E(\tilde{a}) = \int_a^b \left[ \sum_{k=0}^{n} a_k x^k - f(x) \right]^2 \, dx.
\]

Differentiating with respect to \( a_j \) \((j = \{0, 1, \ldots, n\})\) gives:

\[
\frac{\partial E(\tilde{a})}{\partial a_j} = 2 \sum_{k=0}^{n} a_k \int_a^b x^{j+k} \, dx - 2 \int_a^b x^j f(x) \, dx.
\]

At the minimum, we require \( \frac{\partial E(\tilde{a})}{\partial a_j} = 0 \), which gives us a system of equations for the coefficients \( a_k \), the normal equations.

The Normal Equations.

The \((n+1)\)-by-(\(n+1\)) system of equations is:

\[
\sum_{k=0}^{n} a_k \int_a^b x^{j+k} \, dx = \int_a^b x^j f(x) \, dx, \quad j = 0, 1, \ldots, n.
\]

Some notation, let:

\[
\langle f(x), g(x) \rangle = \int_a^b f(x) g(x)^* \, dx,
\]

where \( g(x)^* \) is the complex conjugate of \( g(x) \) (everything we do in this class is real, so it has no effect...)

This is known as an inner product on the interval \([a, b]\). (But, if you want, you can think of it as a notational shorthand for the integral...)

More Notation, Defining the Discrete Inner Product.

In inner product notation, our normal equations:

\[
\sum_{k=0}^{n} a_k \int_a^b x^{j+k} \, dx = \int_a^b x^j f(x) \, dx, \quad j = 0, 1, \ldots, n.
\]

become:

\[
\sum_{k=0}^{n} a_k \langle x^j, x^k \rangle = \langle x^j, f(x) \rangle, \quad j = 0, 1, \ldots, n.
\]

Recall the Discrete Normal Equations:

\[
\sum_{k=0}^{n} \left[ a_k \sum_{i=0}^{N} x_i^{j+k} \right] = \sum_{i=0}^{N} x_i^j f_i, \quad j = 0, 1, \ldots, n.
\]

Hmmm, looks quite similar!

If we have two vectors

\[
\tilde{v} = \{v_0, v_1, \ldots, v_N\}
\]

\[
\tilde{w} = \{w_0, w_1, \ldots, w_N\},
\]

we can define the discrete inner product

\[
[v, w] = \sum_{i=0}^{N} v_i w_i^*,
\]

where, again \( w_i^* \) is the complex conjugate of \( w_i \).

Equipped with this notation, we revisit the Normal Equations...
We need some new language, and tools!

The Condition Number of a Matrix

The condition number of a matrix is the ratio of the largest eigenvalue and the smallest eigenvalue:

$$\text{cond}(A) = \frac{|\lambda_n|}{|\lambda_1|}$$

The condition number is one important factor determining the growth of the numerical (roundoff) error in a computation.

The only thing that changed is the inner product — we went from summation to integration!

A matrix with these entries is known as a Hilbert Matrix; — classical examples for demonstrating how numerical solutions run into difficulties due to propagation of roundoff errors.

Hey! It’s really the same problem!!!

The Condition Number for Our Example

Figure: Ponder, yet again, the example of fitting polynomials to the data (RIGHT). The plot on the left shows the condition numbers for 0th, through 4th degree polynomial problems. Note that for the 5-by-5 system (Hilbert matrix) corresponding to the 4th degree problem the condition number is already $\sim 10^7$. 
Least Squares & Orthogonal Polynomials — (17/28)

Linearly Independent Functions.

Definition (Linearly Independent Functions)

The set of functions \{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\} is said to be 
linearly independent on \([a, b]\) if, whenever
\[
\sum_{i=0}^{n} c_i \Phi_i(x) = 0, \quad \forall x \in [a, b],
\]
then \(c_i = 0, \forall i = 0, 1, \ldots, n\). Otherwise the set is said to be 
linearly dependent.

Theorem

If \(\Phi_j(x)\) is a polynomial of degree \(j\), then the set
\{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\} is linearly independent on any interval
\([a, b]\).

Proof.

Suppose \(c_i \in \mathbb{R}, i = 0, 1, \ldots, n\), and \(P(x) = \sum_{i=0}^{n} c_i \Phi_i(x) = 0 \forall x \in [a, b]\). Since \(P(x)\) vanishes on \([a, b]\) it must be the 
zero-polynomial, i.e. the coefficients of all the powers of \(x\) must be 
zero. In particular, the coefficient of \(x^n\) is zero. \(\Rightarrow c_n = 0\), hence
\(P(x) = \sum_{i=0}^{n-1} c_i \Phi_i(x)\). By repeating the same argument, we find
\(c_i = 0, i = 0, 1, \ldots, n\). \(\Rightarrow \{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\}\) is linearly independent.

More Definitions and Notation... Weight Function

Definition (Weight Function)

An integrable function \(w(x)\) is called a weight function on the interval
\([a, b]\) if \(w(x) \geq 0 \ \forall x \in [a, b]\), but \(w(x) \neq 0\) on any subinterval of
\([a, b]\).

Discrete Least Squares Approximation
Continuous Least Squares Approximation
Orthogonal Polynomials

Linear Independence... Weight Functions... Inner Products
Least Squares, Redux
Orthogonal Functions

Orthogonal Functions

Linearly Independent Functions: Polynomials.

Theorem

If \(\Phi_j(x)\) is a polynomial of degree \(j\), then the set
\{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\} is linearly independent on any interval
\([a, b]\).

Proof.

Suppose \(c_i \in \mathbb{R}, i = 0, 1, \ldots, n\), and \(P(x) = \sum_{i=0}^{n} c_i \Phi_i(x) = 0 \forall x \in [a, b]\). Since \(P(x)\) vanishes on \([a, b]\) it must be the 
zero-polynomial, i.e. the coefficients of all the powers of \(x\) must be 
zero. In particular, the coefficient of \(x^n\) is zero. \(\Rightarrow c_n = 0\), hence
\(P(x) = \sum_{i=0}^{n-1} c_i \Phi_i(x)\). By repeating the same argument, we find
\(c_i = 0, i = 0, 1, \ldots, n\). \(\Rightarrow \{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\}\) is linearly independent.

Weight Function... Inner Product

A weight function will allow us to assign different degrees of 
importance to different parts of the interval. E.g. with
\(w(x) = 1/\sqrt{1-x^2}\) on \([-1, 1]\) we are assigning more weight away 
from the center of the interval.

\[
\langle f(x), g(x) \rangle_{w(x)} = \int_{a}^{b} f(x)g(x)^{*} w(x)dx.
\]
Revisiting Least Squares Approximation with New Notation.

Suppose \( \{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\} \) is a set of linearly independent functions on \([a, b]\), \(w(x)\) a weight function on \([a, b]\), and \(f(x) \in C[a, b]\).

We are now looking for the linear combination

\[
p(x) = \sum_{k=0}^{n} a_k \Phi_k(x)
\]

which minimizes the sum-of-squares-error

\[
E(\tilde{a}) = \int_{a}^{b} [p(x) - f(x)]^2 \, w(x) \, dx.
\]

When we differentiate with respect to \(a_k\), \(w(x)\) is a constant, so the system of normal equations can be written...

The Normal Equations, Revisited for the \(n\)th Time.

\[
\sum_{k=0}^{n} a_k \langle \Phi_k(x), \Phi_j(x) \rangle_w(x) = \langle f(x), \Phi_j(x) \rangle_w(x), \quad j = 0, 1, \ldots, n.
\]

What has changed?

\[
\begin{cases}
    x^k & \rightarrow \Phi_k(x) \\
    \langle \phi, \phi \rangle & \rightarrow \langle \phi, \phi \rangle_w(x)
\end{cases}
\]

New basis functions. New inner product.

Q: — Is he ever going to get to the point?!? Why are we doing this?

We are going to select the basis functions \(\Phi_k(x)\) so that the normal equations are easy to solve!

Orthogonal Functions

**Definition (Orthogonal Set of Functions)**

\(\{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\}\) is said to be an **orthogonal set of functions** on \([a, b]\) with respect to the weight function \(w(x)\) if

\[
\langle \Phi_i(x), \Phi_j(x) \rangle_w(x) = \begin{cases} 
    0, & \text{when } i \neq j, \\
    a_i, & \text{when } i = j.
\end{cases}
\]

If in addition \(a_i = 1, \ i = 0, 1, \ldots, n\) the set is said to be **orthonormal**.

The Payoff — No Matrix Inversion Needed.

\[
\begin{align*}
\text{Theorem} & \quad \text{If } \{\Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x)\} \text{ is a set of orthogonal functions on an interval } [a, b], \text{ with respect to the weight function } w(x), \text{ then the least squares approximation to } f(x) \text{ on } [a, b] \text{ with respect to } w(x) \text{ is} \\
& \quad \quad \quad p(x) = \sum_{k=0}^{n} a_k \Phi_k(x),
\end{align*}
\]

where, for each \(k = 0, 1, \ldots, n,\)

\[
a_k = \frac{\langle \Phi_k(x), f(x) \rangle_w(x)}{\langle \Phi_k(x), \Phi_k(x) \rangle_w(x)}.
\]

We can find the coefficients without solving \(X^T \bar{X} = X^T \bar{b}!!!\)

Where do we get a set of orthogonal functions???
### Building Orthogonal Sets of Functions — The Gram-Schmidt Process

**Theorem (Gram-Schmidt Orthogonalization)**

The set of polynomials \( \{ \Phi_0(x), \Phi_1(x), \ldots, \Phi_n(x) \} \) defined in the following way is orthogonal on \([a, b]\) with respect to \(w(x)\):

\[
\Phi_0(x) = 1, \quad \Phi_1(x) = (x - b_1)\Phi_0,
\]

where

\[
b_1 = \frac{\langle x\Phi_0(x), \Phi_0(x) \rangle_{w(x)}}{\langle \Phi_0(x), \Phi_0(x) \rangle_{w(x)}},
\]

for \(k \geq 2\),

\[
\Phi_k(x) = (x - b_k)\Phi_{k-1}(x) - c_k\Phi_{k-2}(x),
\]

where

\[
b_k = \frac{\langle x\Phi_{k-1}(x), \Phi_{k-1}(x) \rangle_{w(x)}}{\langle \Phi_{k-1}(x), \Phi_{k-1}(x) \rangle_{w(x)}}, \quad c_k = \frac{\langle x\Phi_{k-2}(x), \Phi_{k-2}(x) \rangle_{w(x)}}{\langle \Phi_{k-2}(x), \Phi_{k-2}(x) \rangle_{w(x)}}.
\]

---

### Example: Legendre Polynomials

The set of Legendre Polynomials \( \{ P_n(x) \} \) is orthogonal on \([-1, 1]\) with respect to the weight function \(w(x) = 1\).

\[
P_0(x) = 1, \quad P_1(x) = (x - b_1) \circ 1
\]

where

\[
b_1 = \frac{\int_{-1}^{1} x \, dx}{\int_{-1}^{1} 1 \, dx} = 0
\]

i.e. \(P_1(x) = x\).

\[
b_2 = \frac{\int_{-1}^{1} x^3 \, dx}{\int_{-1}^{1} x^2 \, dx} = 0, \quad c_2 = \frac{\int_{-1}^{1} x^2 \, dx}{\int_{-1}^{1} 1 \, dx} = 1/3,
\]

i.e. \(P_2(x) = x^2 - 1/3\).

The first six Legendre Polynomials are

\[
\begin{align*}
P_0(x) &= 1, \\
P_1(x) &= x, \\
P_2(x) &= x^2 - 1/3, \\
P_3(x) &= x^3 - 3x/5, \\
P_4(x) &= x^4 - 6x^2/7 + 3/35, \\
P_5(x) &= x^5 - 10x^3/9 + 5x/21.
\end{align*}
\]

We encountered the Legendre polynomials in the context of numerical integration. It turns out that the **roots** of the Legendre polynomials are used as the nodes in Gaussian quadrature.

Now we have the machinery to manufacture Legendre polynomials of any degree.

---

### Example: Laguerre Polynomials

The set of Laguerre Polynomials \( \{ L_n(x) \} \) is orthogonal on \((0, \infty)\) with respect to the weight function \(w(x) = e^{-x}\).

\[
L_0(x) = 1,
\]

\[
b_1 = \frac{\langle x, 1 \rangle_{e^{-x}}}{\langle 1, 1 \rangle_{e^{-x}}} = 1
\]

\[
L_1(x) = x - 1,
\]

\[
b_2 = \frac{\langle x(x-1), x-1 \rangle_{e^{-x}}}{\langle x-1, x-1 \rangle_{e^{-x}}} = 3, \quad c_2 = \frac{\langle x(x-1), 1 \rangle_{e^{-x}}}{\langle 1, 1 \rangle_{e^{-x}}} = 1,
\]

\[
L_2(x) = (x-3)(x-1) - 1 = x^2 - 4x + 2.
\]