Lecture 11 - *Least-Squares Problems*

OBJECTIVE:
The idea of least-squares data fitting has been around since Gauss and Legendre invented it around 1800.
It is still the method of choice when it comes to general parameter estimation.

In terms of linear algebra, we want to "solve" an overdetermined system of equations;
i.e., $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A}$ has more rows than columns.

The idea of a least-squares solution is to find the $\mathbf{x}$ that minimizes the 2-norm of the residual $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$.

◇ THE LINEAR LEAST-SQUARES PROBLEM

Consider an *overdetermined* system of linear equations with n unknowns and $m > n$ equations
i.e., we want to find a vector $\mathbf{x} \in \mathbb{R}^n$ that satisfies $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$.

In general, this problem has no solution!

A solution $\mathbf{x}$ can only exist if $\mathbf{b} \in \mathrm{range}(\mathbf{A})$.

But, $\mathbf{b} \in \mathbb{R}^m$ and $\mathrm{range}(\mathbf{A})$ has dimension at most n.

Because $m > n$, we can see how only exceptional vectors $\mathbf{b}$ might lie in $\mathrm{range}(\mathbf{A})$.

We can try to quantify by how much the equations fail to be satisfied by defining a *residual* vector

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$$

So, $\mathbf{r} = \mathbf{0}$ if $\mathbf{A}\mathbf{x} = \mathbf{b}$.

But, for a general overdetermined system $\mathbf{r} \neq \mathbf{0}$, we try and minimize $\|\mathbf{r}\|$ for some choice of norm.

If we choose the 2-norm, the problem can be stated as

$$\min_{\mathbf{x} \in \mathbb{R}^n} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m, m \geq n$ are given.

For reasons which we will see later, this is called a *linear* least-squares problem because eventually the unknowns appear as the solution to a linear system (the $\|\cdot\|_2$ is quadratic in its arguments, so when you set its derivative to zero in order to find the min, the arguments appear linearly).

The 2-norm also corresponds to Euclidean distance, so there is a simple geometric interpretation of what we are doing: find a vector $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{Ax} \in \mathbb{R}^m$ is closest to $\mathbf{b} \in \mathbb{R}^m$.

Example 11.1 POLYNOMIAL INTERPOLATION

Suppose we have m distinct data points of the form $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$.

Then, we can connect these m points by a *unique* polynomial of degree at most $m-1$:

$$p(x) = c_0 + c_1 x + \ldots + c_{m-1} x^{m-1} \qquad (1)$$

i.e.,

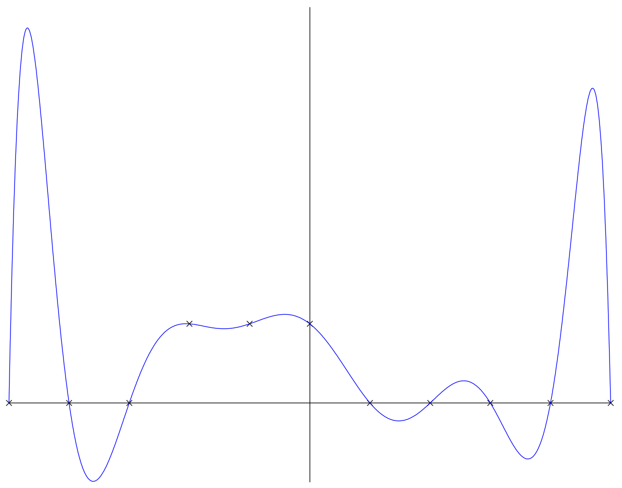
$$p(x_i) = y_i \qquad \text{for } i = 1, 2, \ldots, m \qquad (2)$$

But how do we find the coefficients c_i?

We use the m conditions in (2) and the form (1) to create a linear system

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{m-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^{m-1} \end{bmatrix} \begin{bmatrix} c_o \\ c_1 \\ \vdots \\ c_{m-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

which can be solved for the c_i provided $x_i \neq x_j$ when $i \neq j$ (this is called a *Vandermonde* system).



Note 1.

1. *$p(x)$ passes through all the data points (as it should!)*

2. *near the ends of the interval, $p(x)$ exhibits large* spurious *oscillations*
   i.e., oscillations that are likely not a reasonable reflection of the underlying data.

This is typical of high-degree polynomial interpolation (i.e., fitting a high-degree polynomial to many data points) and the problems get <u>worse</u> as you take more data points!
$\longrightarrow$ polynomial interpolation is ill-conditioned.
If a data point is perturbed by a little, the interpolating polynomial changes by a lot.

Example 11.2 LEAST-SQUARES POLYNOMIAL FITTING

Rather than forcing a high-degree polynomial to exactly go through all the data points, it is more stable to reduce the degree of the polynomial and construct a least-squares fit
(now we have an overdetermined system: more data points than polynomial coefficients c_i)

So, given data points $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$ let

$$p(x) = c_o + c_1 x + \ldots + c_{n-1} x^{n-1}$$

for some $n < m$.

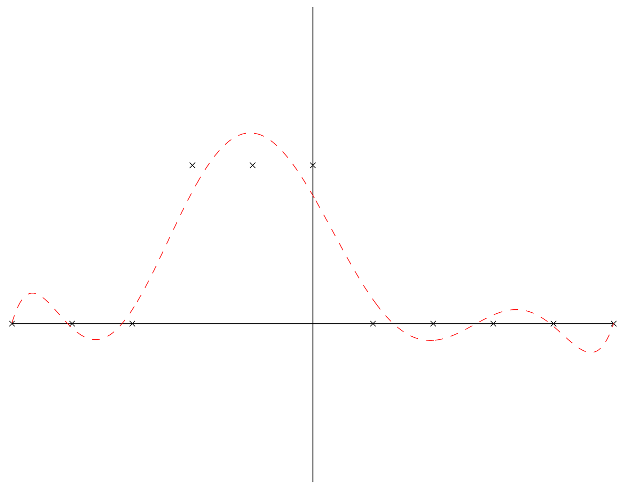
This polynomial will be a least-squares fit to the data if it minimizes

$$\sum_{i=1}^{m} (p(x_i) - y_i)^2 \tag{3}$$

i.e., the sum of the squares of the deviation of the fit from the measured data points.

The expression (3) equals $\|\mathbf{r}\|_2^2$ for the rectangular Vandermonde system

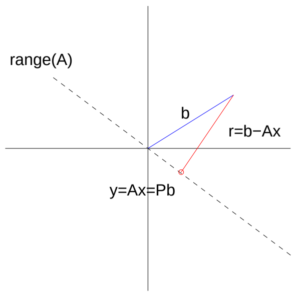
$$\begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^{n-1} \end{bmatrix} \begin{bmatrix} c_o \\ c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$



A much better fit!

◇ ORTHOGONAL PROJECTION AND THE NORMAL EQUATIONS

One can think of using least squares to “solve” an overdetermined system $\mathbf{Ax} = \mathbf{b}$ as finding the magical $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{Ax} \in \mathrm{range}(\mathbf{A}) \in \mathbb{R}^m$ is closest to the vector $\mathbf{b} \in \mathbb{R}^m$
$\longrightarrow$ this will minimize the 2-norm of the residual $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$



From the figure, the magical $\mathbf{x}$ that minimizes $\|\mathbf{r}\|$ satisfies

$$\mathbf{Ax} = \mathbf{Pb}$$

where $\mathbf{P} \in \mathbb{R}^{m \times m}$ is the *orthogonal projector* (Lecture 6) of $\mathbf{b}$ onto $\mathrm{range}(\mathbf{A})$
i.e., *the residual* $\mathbf{r}$ *must be orthogonal to* $\mathrm{range}(\mathbf{A})$.

Theorem 1. *Let* $\mathbf{A} \in \mathbb{R}^{m\times n}(m \geq n)$ *and* $\mathbf{b} \in \mathbb{R}^m$ *be given.*
Then a vector $\mathbf{x} \in \mathbb{R}^n$ *that minimizes*

$$\|\mathbf{r}\|_2 = \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$$

(i.e., $\mathbf{x}$ *is the least-squares solution) if and only if*

$$\mathbf{r} \perp \mathrm{range}(\mathbf{A})$$

$\Leftrightarrow$

$$\mathbf{A}^T\mathbf{r} = \mathbf{0}$$

$\Leftrightarrow$

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b} \tag{4}$$

$\Leftrightarrow$

$$\mathbf{P}\mathbf{b} = \mathbf{A}\mathbf{x}$$

where $\mathbf{P} \in \mathbb{R}^{m\times m}$ *is the orthogonal projector onto* $\mathrm{range}(\mathbf{A})$*.*

Note 2. *Equations (4) are known as the* normal equations.
This is an $n \times n$ system of equations that has a unique solution if and only if $\mathbf{A}$ has full rank.
Thus, the solution $\mathbf{x}$ to the least-squares problem is unique if and only if $\mathbf{A}$ has full rank.

◇ PSEUDOINVERSE

When $\mathbf{A}$ has full rank, the solution $\mathbf{x}$ to the least-squares problem is unique and formally can be written as

$$\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$$

This allows us to define the so-called *pseudoinverse* of $\mathbf{A}$, denoted by $\mathbf{A}^\dagger$

$$\mathbf{A}^\dagger = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T \in \mathbb{R}^{n\times m}$$

We now look at the 3 most popular algorithms for solving least-squares problems.

◇ NORMAL EQUATIONS

This is the traditional way to solve least-squares problem.

If $\mathbf{A}$ has full rank, then $\mathbf{A}^T\mathbf{A}$ is square, symmetric positive definite (see Lecture 23), and of modest size (n is usually quite small, especially compared to m).

The standard method to solve such systems is by *Cholesky factorization* (Lecture 23) which factors a symmetric positive definite matrix into the form $\mathbf{R}^T\mathbf{R}$, where $\mathbf{R}$ is upper triangular.

So, the normal equations become

$$\mathbf{R}^T\mathbf{R}\mathbf{x} = \mathbf{A}^T\mathbf{b}$$

ALGORITHM 11.1: LEAST SQUARES VIA NORMAL EQUATIONS

1. Form $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{b}$.

2. Compute the Cholesky factorization $\mathbf{A}^T\mathbf{A} = \mathbf{R}^T\mathbf{R}$ to obtain $\mathbf{R}^T\mathbf{R}\mathbf{x} = \mathbf{A}^T\mathbf{b}$.

3. Solve the lower-triangular system $\mathbf{R}^T\mathbf{w} = \mathbf{A}^T\mathbf{b}$ for $\mathbf{w}$ $(= \mathbf{R}\mathbf{x})$.

4. Solve the upper-triangular system $\mathbf{R}\mathbf{x} = \mathbf{w}$ for $\mathbf{x}$.

The cost of this algorithm is dominated by steps 1. and 2.

$$\text{Work for Algorithm 11.1 } \sim mn^2 + \frac{1}{3}n^3 \text{ flops.}$$

◇ QR FACTORIZATION

The modern way (since the 1960s) to solve least-squares problems uses a reduced QR factorization

$$\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}}$$

obtained by Householder triangularization.

The orthogonal projector onto $\mathrm{range}(\mathbf{A})$ can then be written as

$$\mathbf{P} = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T.$$

So,

$$\mathbf{y} = \mathbf{Pb} = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T\mathbf{b}$$

Now since $\mathbf{y} \in \mathrm{range}(\mathbf{A})$, we can solve $\mathbf{Ax} = \mathbf{y}$.

So,

$$\hat{\mathbf{Q}}\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}\hat{\mathbf{Q}}^T\mathbf{b}$$

or

$$\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b} \tag{5}$$

(and if we write $\mathbf{x} = \hat{\mathbf{R}}^{-1}\hat{\mathbf{Q}}^T\mathbf{b}$, we can define

$$\mathbf{A}^\dagger = \hat{\mathbf{R}}^{-1}\hat{\mathbf{Q}}^T).$$

System (5) is upper-triangular and nonsingular if $\mathbf{A}$ has full rank.

It can easily be solved by back substitution (Lecture 17)

ALGORITHM 11.2: LEAST SQUARES VIA QR FACTORIZATION

1. Compute the reduced QR factorization $\mathbf{A} = \hat{\mathbf{Q}}\hat{\mathbf{R}}$.

2. Compute the vector $\hat{\mathbf{Q}}^T\mathbf{b}$.

3. Solve the upper-triangular system $\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b}$ for $\mathbf{x}$.

Note 3. *This formulation can be derived from the normal equations*

$$\begin{aligned}(\mathbf{A}^T\mathbf{A})\mathbf{x} &= \mathbf{A}^T\mathbf{b}\\ \hat{\mathbf{R}}^T\hat{\mathbf{Q}}^T\hat{\mathbf{Q}}\hat{\mathbf{R}}\mathbf{x} &= \hat{\mathbf{R}}^T\hat{\mathbf{Q}}^T\mathbf{b}\end{aligned}$$

However, since $\hat{\mathbf{Q}}^T\hat{\mathbf{Q}} = \mathbf{I}$,

$$\hat{\mathbf{R}}\mathbf{x} = \hat{\mathbf{Q}}^T\mathbf{b}$$

The work for Algorithm 11.2 is dominated by the cost of the QR factorization.

Using Householder reflections, we obtain

$$\text{Work for Algorithm 11.2 } \sim 2mn^2 - \frac{2}{3}n^3 \text{ flops}$$

$\diamond$ SVD

We can also use the reduced SVD $\mathbf{A} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\mathbf{V}^T$ to solve the least-squares problem.

Now,

$$\mathbf{P} = \hat{\mathbf{U}}\hat{\mathbf{U}}^T$$

So,

$$\mathbf{y} = \mathbf{P}\mathbf{b} = \hat{\mathbf{U}}\hat{\mathbf{U}}^T\mathbf{b}$$

and,

$$\begin{aligned} \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\mathbf{V}^T\mathbf{x} &= \hat{\mathbf{U}}\hat{\mathbf{U}}^T\mathbf{b} \\ \hat{\mathbf{\Sigma}}\mathbf{V}^T\mathbf{x} &= \hat{\mathbf{U}}^T\mathbf{b} \end{aligned}$$

(and writing $\mathbf{x} = \mathbf{V}\mathbf{\Sigma}^{-1}\hat{\mathbf{U}}^T\mathbf{b}$ implies

$$\mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^{-1}\hat{\mathbf{U}}^T).$$

ALGORITHM 11.3: LEAST SQUARES VIA SVD

1. Compute the reduced SVD $\mathbf{A} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\mathbf{V}^T$.

2. Compute the vector $\hat{\mathbf{U}}^T\mathbf{b}$.

3. Solve the diagonal system $\hat{\mathbf{\Sigma}}\mathbf{w} = \hat{\mathbf{U}}^T\mathbf{b}$
   for $\mathbf{w}$ ($= \mathbf{V}^T\mathbf{x}$).

4. Solve $\mathbf{V}^T\mathbf{x} = \mathbf{w}$ for $\mathbf{x}$; i.e., Set $\mathbf{x} = \mathbf{V}\mathbf{w}$.

If $\mathbf{A}$ has full rank, the diagonal system $\hat{\mathbf{\Sigma}}\mathbf{w} = \hat{\mathbf{U}}^T\mathbf{b}$ is nonsingular.

Starting from the normal equations $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}$

$$\mathbf{V}\hat{\mathbf{\Sigma}}^T\hat{\mathbf{U}}^T\hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\mathbf{V}^T\mathbf{x} = \mathbf{V}\hat{\mathbf{\Sigma}}^T\hat{\mathbf{U}}^T\mathbf{b}$$

$\Leftrightarrow$

$$\hat{\mathbf{\Sigma}}\mathbf{V}^T\mathbf{x} = \hat{\mathbf{U}}^T\mathbf{b}$$

The cost of Algorithm 11.3 is dominated by the computation of the SVD.

Work for algorithm 11.3 $\sim 2mn^2 + 11n^3$ flops

◇ COMPARISON OF ALGORITHMS

1. Solving the normal equations is the cheapest, roughly by a factor of 2.
   However, we will see that it is much less well-conditioned than the other two methods
   i.e., there can be instabilities (accumulation of roundoff errors)

2. QR is cheaper than SVD when $m \approx n$.
   It is the method of choice as long as $\mathbf{A}$ is not too close to being rank deficient.

3. SVD costs about as much as QR when $m \gg n$ (which is often the case!).
   It is also the method of choice if $\mathbf{A}$ is close to being rank deficient (all data are not independent).