Last lecture: Least Squares

• Consider the overdetermined linear system:

\[ A \vec{x} = \vec{b} \]

\[ \vec{a}_1 x_1 + \vec{a}_2 x_2 + \ldots + \vec{a}_n x_n = \vec{b} \]

• If \( \vec{b} \) is not in colspan(\( A \)), there is no solution
• the least-squares solution “minimally perturbs” \( \vec{b} \)
Last lecture: Least Squares

\[
\begin{align*}
\min_{\hat{x}} \| \bar{e} \|^2 &= \| \bar{b} - \hat{b} \|^2 = \| \bar{b} - A\hat{x} \|^2 \\
\rightarrow \quad \langle A^T, \bar{e} \rangle &= 0 \\
\hat{x} &= (A^T A)^{-1} A^T \bar{b}
\end{align*}
\]
Linear regression

Least squares solution

\[ \hat{w} = (A^T A)^{-1} A^T b \]
Least squares is a building block for all of signal processing/machine learning/pattern matching.

**Example: Linear regression**

Fit a line (i.e., project data onto an allowed line) but don't know the line!

$$y = mx + c$$

We measure $y$ for multiple $x$ values:

- Known: $(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)$
- Unknown: \[
\begin{bmatrix}
m \\
c
\end{bmatrix}
\]

Treat as $\vec{w}$ in $A\vec{w} = \vec{b}$

$$A = \begin{bmatrix}
x_1 & 1 \\
x_2 & 1 \\
x_3 & 1 \\
\vdots & \vdots \\
x_n & 1
\end{bmatrix}, \quad \begin{bmatrix}
ym \\
c
\end{bmatrix} = \begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
\vdots \\
y_n
\end{bmatrix}$$

Solve for $\vec{w}$

Best estimate for $\widehat{\vec{w}} = (A^T A)^{-1} A^T \vec{b}$ least squares soln

* go to Python notebook demo 'Linear-regression-Facebook.ipynb'

- Fits fairly well to line
- But that may not hold for extrapolation
Demo: fitting Facebook stock data to a line

```python
a1 = (fb_stock_data[:,0]).reshape([number_of_data_points,1])
a2 = np.ones([number_of_data_points,1])
A = np.hstack((a1,a2))
b = (fb_stock_data[:,1]).reshape([number_of_data_points,1])

x = least_squares(A,b)
m = x[0]
c = x[1]

y_new = m*(fb_stock_data[:,0]) + c

plt.plot(fb_stock_data[:,0], fb_stock_data[:,1], 'g')
plt.plot(fb_stock_data[:,0], y_new, 'r', linewidth=2.0)
plt.show()
```
BUT, not everything fits to a line!?!
BUT, not everything fits to a line!?!?

“Using a term like nonlinear science is like referring to the bulk of zoology as the study of non-elephant animals.” – S. Ulam
How did Gauss find Ceres? fit to Kepler's Laws (elliptical orbits)

Let's put unknowns into a vector:

\[
\begin{bmatrix}
  a \\
  b \\
  c \\
  d \\
  e
\end{bmatrix}
\]

Write some equations for measured position (x,y):

\[ax^2 + by^2 + cxy + dx + ey = 1\]

There are 22 measurements in dataset, so let's put in a matrix:

\[
\begin{bmatrix}
  x_1 & y_1 & x_1 & y_1 \\
  x_2 & y_2 & x_2 & y_2 \\
  \vdots & \vdots & \vdots & \vdots \\
  x_{22} & y_{22} & x_{22} & y_{22}
\end{bmatrix}
\begin{bmatrix}
  a \\
  b \\
  c \\
  d \\
  e
\end{bmatrix} = \begin{bmatrix}
  1 \\
  1 \\
  \vdots \\
  1
\end{bmatrix}
\]

22 equations, 5 unknowns

'Overdetermined'

in machine learning, cols are called 'features'

*Gauss did this by hand. We have Jupyter notebooks so can be lazy 😊 yay!*

Demo: 'Ceres-orbit.ipynb'

Now we can guess the missing measurements!

→ go to demo 'Linear-regression_Snapchat_new.ipynb', load 'SNAP.csv'

→ the fit gets better for higher-order polynomials (y_new=predict_with_degree)

→ BUT, you might be "fitting to noise"

→ want to use the simplest model that most accurately represents the real world! KISS!

"Everything should be made as simple as possible, and no simpler." - Einstein?

← this is why Gauss' example worked so well - Kepler's laws were correct...
Let's go back to least squares for trilateration:

**Recall:**

Trilateration find my coordinates \( \hat{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \) from known distances \( d_A, d_B, d_C \) to 3 satellites with known positions \( \hat{a}, \hat{b}, \hat{c} \)

\[
\begin{align*}
\| \hat{x} - \hat{a} \|^2 &= d_A^2 \\
\| \hat{x} - \hat{b} \|^2 &= d_B^2 \\
\| \hat{x} - \hat{c} \|^2 &= d_C^2
\end{align*}
\]

\[
\begin{bmatrix}
-2a_1 + 2b_1 \\
-2a_2 + 2b_2 \\
-2a_1 + 2c_1 \\
-2a_2 + 2c_2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= \begin{bmatrix}
d_{\hat{A}}^2 - d_{\hat{B}}^2 - \| \hat{a} \|^2 + \| \hat{b} \|^2 \\
d_{\hat{A}}^2 - d_{\hat{C}}^2 - \| \hat{a} \|^2 + \| \hat{c} \|^2
\end{bmatrix}
\]

knowns  \quad \text{to solve for} \quad \hat{x} = \hat{b}  \quad \text{Linear system}

Now we can use least squares: \( \hat{x} = (A^T A)^{-1} A^T \hat{b} \)

**How do we know if \((A^T A)^{-1}\) exists?**

Recall: invertible \(\Rightarrow\) trivial null space \(\Rightarrow\) lin. ind. cols!

Matrix \(Q\) is invertible if and only if \(\text{Null}(Q) = \{0\}\) (the null space is trivial)

**Theorem:** \(\text{Null}(A^T A) = \text{Null}(A)\)

**Recall:** If \(\| \hat{x} \| = 0\), then \(\hat{x} = \hat{0}\)

**Proof:** \(\| \hat{x} \| = 0\)

\[
x_1^2 + x_2^2 + \ldots + x_k^2 = \langle \hat{x}, \hat{x} \rangle = 0
\]

all terms \(\geq 0\)

so all terms = 0 \(\Rightarrow\) so \(\hat{x} = \hat{0}\) !

Now, back to proof...

**Recall:** Properties of transposes

\( (AB)^T = B^T A^T \)

\[
A = \begin{bmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{bmatrix} \quad B = \begin{bmatrix}
b_1 \\
b_2
\end{bmatrix}
\]

\( (AB)^T = \begin{bmatrix}
a_{11} b_1 + a_{12} b_2 \\
a_{21} b_1 + a_{22} b_2
\end{bmatrix} \]

\( B^T A^T = \begin{bmatrix}
b_1 & b_2
\end{bmatrix} \begin{bmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{bmatrix} = \begin{bmatrix}
\alpha \\
\beta
\end{bmatrix}
\]

same!

\[
= \begin{bmatrix}
(b_1 a_{11} + b_2 a_{12}) & (b_1 a_{21} + b_2 a_{22})
\end{bmatrix} = \begin{bmatrix}
\alpha \\
\beta
\end{bmatrix}
\]

\[
= \begin{bmatrix}
(b_1 a_{11} + b_2 a_{12}) & (b_1 a_{21} + b_2 a_{22})
\end{bmatrix}
\]

\[
= \begin{bmatrix}
\alpha \\
\beta
\end{bmatrix}
\]

\[
= \begin{bmatrix}
\alpha \\
\beta
\end{bmatrix}
\]

\[
= \begin{bmatrix}
\alpha \\
\beta
\end{bmatrix}
\]
Null \( (A^TA) = \text{Null} \( (A) \)

1. Known: \( \mathbf{v} \in \text{Null} \( (A) \) \)
   \[
   A \mathbf{v} = \mathbf{0}
   \]
   want: \( \mathbf{v} \in \text{Null} \( (A^TA) \) \)
   \[
   (A^TA) \mathbf{v} = \mathbf{0}
   \]
   \( \mathbf{v} \) belongs to \( \text{Null} \( (A^TA) \) \)

2. Known: \( \mathbf{v} \in \text{Null} \( (A^TA) \) \)
   \[
   A^TA \mathbf{v} = \mathbf{0}
   \]
   Can't divide by \( A \) to get rid of it
   Consider
   \[
   \|A \mathbf{v}\|^2 = \langle A \mathbf{v}, A \mathbf{v} \rangle = (A^T A) \mathbf{v} = (A \mathbf{v})^T (A \mathbf{v}) = \mathbf{v}^T A^T (A \mathbf{v}) = \mathbf{v}^T (A^T A \mathbf{v}) = \mathbf{v}^T (\mathbf{0}) = 0
   \]
   \( \|A \mathbf{v}\|^2 = 0 \) \( \rightarrow \) \( \|A \mathbf{v}\| = 0 \) \( \rightarrow \) \( A \mathbf{v} = \mathbf{0} \) \( \checkmark \)