Unit 6:

Least Squares

How can we use many observations of a phenomenon to infer a model?

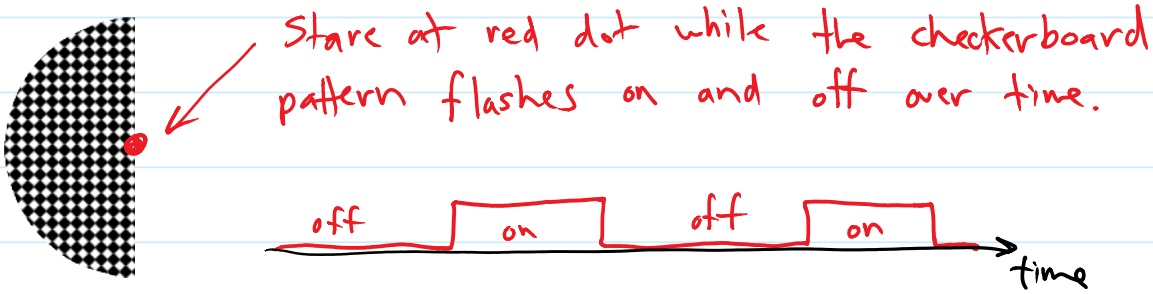
Goal: To see how the problem of least-squares crops up in data analysis.

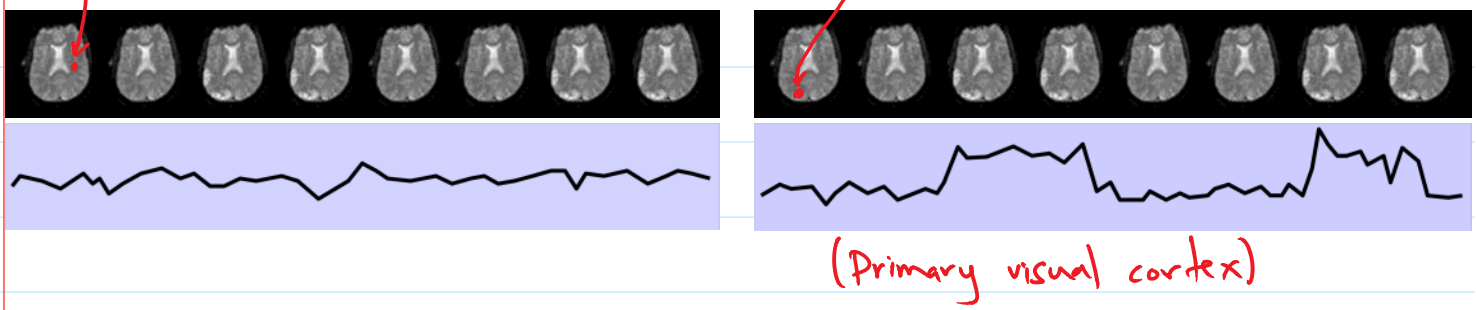
Functional Magnetic Resonance Imaging (FMRI)

Functional MRI is able to pick up on slight brightness changes due to the concentration of oxygenated blood caused by neurological activity. Active brain regions appear between 1% and 4% brighter.

To find out what part of the brain is responsible for different types of processing, give the desired stimulus in an on/off manner and look for corresponding fluctuations in the brightness.

eg. Stimulus



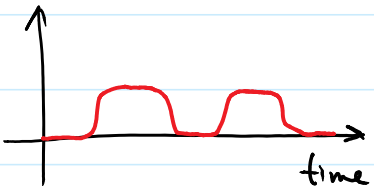


The FMRI signal for a given pixel can be decomposed as

signal = **brain activation + baseline intensity + known artifacts + noise**

Brain Activation:

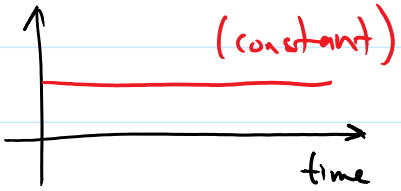
The time-course is chosen by the experimenter. Usually a smoothed on-off function.



Baseline Intensity:

Baseline Intensity:

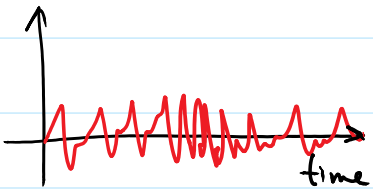
This is how bright the pixel would be without any other influences.



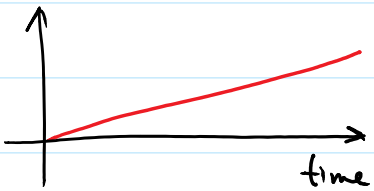
Known Artifacts:

Patient breathing introduces a slow oscillation. We can measure their lung volume during the scan, so we know the basic form of this component.

The heart beat causes a similar oscillation, but at a higher frequency. We can also monitor the patient's heart beat.



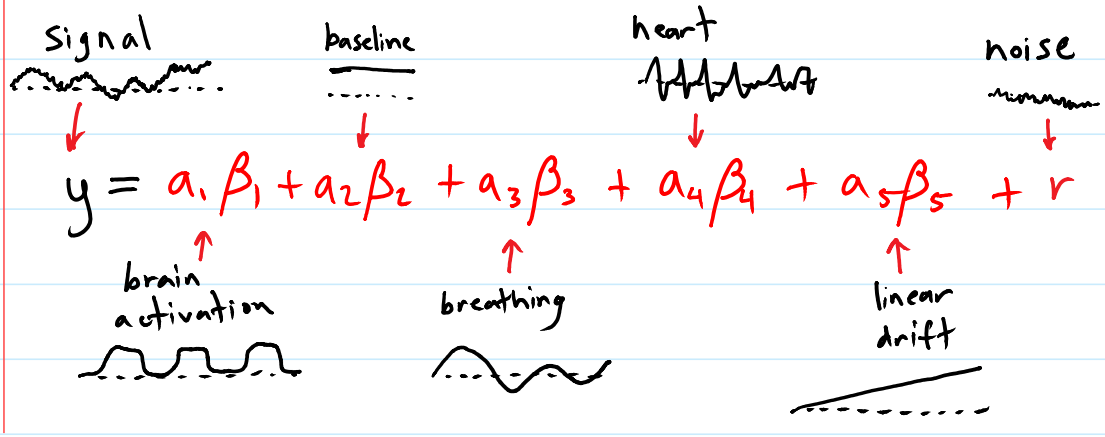
During the scan, the scanner tends to heat up, changing the strength of the magnetic fields, and thus the signal strength. We can model this as a linear trend.



Noise:

Everything else, including electronic circuit noise, quantum noise, etc.

Hence, we can model our signal as a linear combination of known components, plus noise:



In this formulation, $\beta_1, \ldots, \beta_5$ determine how much of each component is present in the signal y.
For a given pixel:

- If β_1 is small ==> no activation there
- If β_1 is large ==> activation there

Goal: We want to know the strength of the activation component for each pixel.

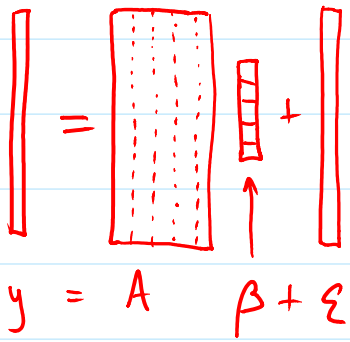
Problem: We can't isolate the activation component unless we know how much of each of the other components to subtract off.

Solution: Find a multiplier for each component so that the error between the measured signal and the model is minimized.

The model can be written using matrix-vector notation,

$$y = \left[a_1 \middle| a_2 \middle| a_3 \middle| a_4 \middle| a_5\right] \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix} + r \quad \Leftrightarrow \quad y = A\beta + r$$

Pictorially,



$y = A \quad \beta + \varepsilon$

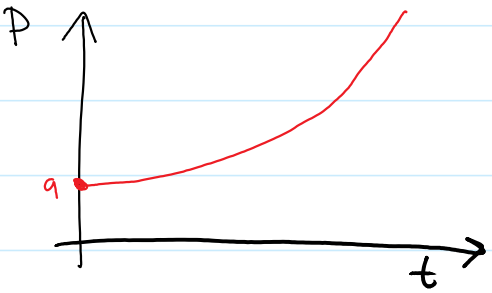
Our task is to find parameter values, $\hat{\beta}$, so that $A\hat{\beta}$ is as close to our observations y as possible.
ie. we want $y - A\hat{\beta}$ to be small.

Example 2: Canada's Population

Example 2: Canada's Population

A common model for population growth is the exponential growth model,

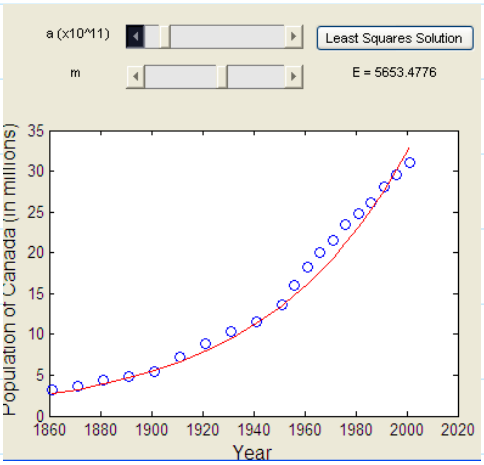
$$P(t) = ae^{tm}$$



We'd like to fit this model to our Canadian census data. That is, calculate values for a and m so that the model fits the data.

(Matlabe demo ls_gui.m)

One way to find the best a and m values is to find the least squares solution. This is a least-squares parameter fitting problem, similar to the FMRI problem above.



$$P(t_i) \approx ae^{t_i m}$$

$$\begin{aligned} 3230 &= ae^{1861m} \\ 3689 &= ae^{1871m} \\ &\vdots \\ 31021 &= ae^{2001m} \end{aligned}$$

20 equations,
2 unknowns
(over determined)

However, the population model is nonlinear, so we can't formulate it as a matrix equation.

TRICK: Take the logarithm of both sides.

$$\ln P(t) = \ln(ae^{tm})$$

$$\ln P(t) = \ln a + \ln e^{tm}$$

$\ln(y) = \ln a + \ln e$

$y = b + tm$

So, if we're willing to work with the logarithm of the population, then the resulting model is linear. This type of model is called a **log-linear** model.

$$\left.\begin{array}{l} \ln 3236 \rightarrow 8.0802 = b + 1861m \\ \vdots \\ \ln 31021 \rightarrow 10.3424 = b + 2001m \end{array}\right\}$$ 20 equations, 2 unknowns

In matrix form,

$$\begin{bmatrix} 8.0802 \\ \vdots \\ 10.3424 \end{bmatrix} = \begin{bmatrix} 1 & 1861 \\ \vdots & \vdots \\ 1 & 2001 \end{bmatrix} \begin{bmatrix} b \\ m \end{bmatrix} + \begin{bmatrix} ? \\ \\ \end{bmatrix}$$

$$y = A\beta + r$$

<u>Visually, as intersection of lines</u>

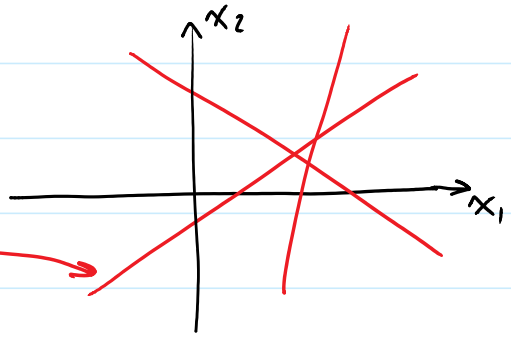
Consider this overdetermined system.

eg. $\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$

3 equations
2 unknowns

Each line is a set of (x_1, x_2) values consistent with one observation.

ie. $a_{i1}x_1 + a_{i2}x_2 = b_i$



Normally, there is no solution. But if the 3 lines really do come from a process with only 2 degrees of freedom, then we should be able to find (x_1, x_2) that approximately solves for the intersection of all 3 lines.

Residual

After some β -values have been chosen, I can plug them into my model and remove modeled components. What I'm left with is called the **"residual".** It's the r (noise) component.

i.e. $y = A\beta + r$

$\Rightarrow \quad r = y - A\beta$

The smaller the values in the residual vector, the better job I've done of modelling my system. A one-number summary to measure the size of the residual is the **"total squared error".**

$$E(\beta) = \sum_{i=1}^{N} r_i^2$$

$E(\beta)$ is the squared length of the residual vector r.
Another way to write it is

$$E(\beta) = r^T r \quad (\text{also written } \|r\|^2)$$

Normal Equations

How do we find the best β that minimizes the total squared error?

$$E(\beta) = r^T r = (y - A\beta)^T (y - A\beta)$$

We need to find the minimum of $E(\beta)$. For this, we return to 1st-year calculus.

$$\Rightarrow \frac{dE}{d\beta} = 0 \quad \text{or} \quad \nabla E = \vec{0}$$

By definition, $\frac{dE}{} = \lim \frac{E(\beta+e) - E(\beta)}{}$ (e is a vector)

By definition, $\frac{dE}{d\beta} = \lim_{e \to 0} \frac{E(\beta+e) - E(\beta)}{\|e\|}$ (e is a vector)

Let's simplify the numerator first.

$$E(\beta+e) - E(\beta) = (y - A(\beta+e))^T (y - A(\beta+e)) - (y - A\beta)^T (y - A\beta)$$

$$= y^Ty - y^TA(\beta+e) - (\beta+e)^TA^Ty + (\beta+e)^TA^TA(\beta+e) - y^Ty + y^TA\beta + \beta^TA^Ty - \beta^TA^TA\beta$$

$$= -y^TAe - e^TA^Ty + e^TA^TA\beta + \beta^TA^Ae + e^TA^TAe$$

($-y^TAe = -e^TA^Ty$, $\beta^TA^Ae = e^TA^TA\beta$)

$$= -2e^TA^Ty + 2e^TA^TA\beta + e^TA^TAe$$

$$= 2e^T(A^TA\beta - A^Ty) + e^TA^TAe$$

$$\therefore \frac{dE}{d\beta} = \lim_{e \to 0} \frac{2e^T(A^TA\beta - A^Ty) + e^TA^TAe}{\|e\|} = 0$$

Observe,

$$\lim_{e \to 0} \frac{e^TA^TAe}{\|e\|} = \lim_{e \to 0} \frac{\|Ae\|^2}{\|e\|} \leq \lim_{e \to 0} \frac{\|e\|^2 \|A\|^2}{\|e\|} = 0$$

matrix norm, see "Condition Numbers and Norms" in notes.

$$\therefore \lim_{e \to 0} \frac{2e^T(A^TA\beta - A^Ty)}{\|e\|} = 0$$

Thus, $A^TA\beta - A^Ty = 0$

$\Rightarrow A^TA\beta = A^Ty$

This system of equations is known as the **"Normal Equations".**

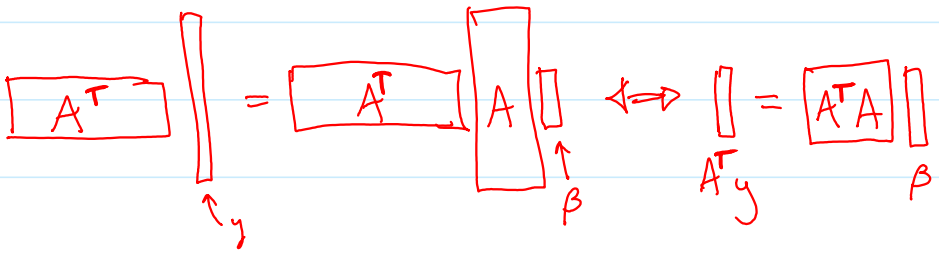
LS-03: Geometrical Interpretation

Goal: To gain an intuitive understanding of least-squares solutions.

Recall that we have an over-determined linear system,

$$y = A\beta + r$$

and we can find the optimal β that minimizes the total squared residual by solving the Normal equations, $A^T y = A^T A \beta$



The Normal equations are a relatively small system.

eg. Suppose we have 1000 observations of a model with 4 unknown parameters

A is 1000×4 but $A^T A$ is 4×4

Canada's Population model

A is 20×2 but $A^T A$ is 2×2

A simple, intuitive way to derive the Normal equations...

$$y = A\beta + r \Rightarrow A^T y = A^T A \beta + A^T r \to 0$$

To solve for β, you could multiply by $(A^T A)^{-1}$.

$$(A^T A)^{-1} A^T y = \beta \Rightarrow \beta = A^+ y$$

A^+ "plus sign" — "matrix pseudo-inverse"

Hence, given the linear least squares problem

$$\min_{\beta} \|y - A\beta\|^2$$

the solution is $\beta = A^+ y$, where $A^+ = (A^T A)^{-1} A^T$

Geometrical Interpretation

Consider the LS problem $\min_{\beta} \|y - A\beta\|^2 \Rightarrow y = A\beta + r$.

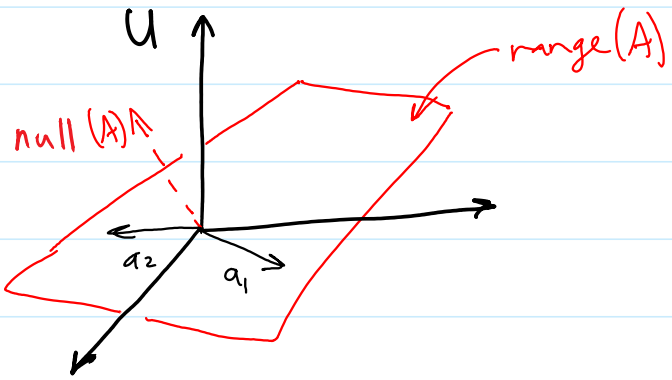
Notice that y, the columns of A, $A\beta$ and r are all vectors in the same common space, call it U.

- For our FMRI signal decomposition, it's the space of all possible sequences of 1000 numbers, $\mathbb{R}^{1000}$.
- For our population model, it's the space of all possible census results, $\mathbb{R}^{20}$.

In that space, U, range(A) is a subspace (Let $A \in \mathbb{R}^{m \times n}$)

$$\text{range}(A) = \{ Ax \in \mathbb{R}^m \mid x \in \mathbb{R}^n \text{ where } A \text{ has } n \text{ columns} \}$$

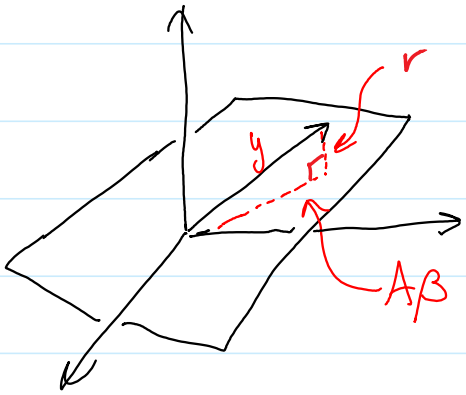
Let $A = [a_1 \mid a_2]$



We have $y = A\beta + r$

↑ this part $\in$ range(A)

The task of least-squares is to find the β such that $A\beta$ is "close" to y.
ie. r is a short vector



The $A\beta$ that is closest gives the shortest r. That is accomplished when

$r \in \text{null}(A)$

$y = A\beta + r \Rightarrow A^+y = A^+A\beta + A^+r$

$A^+y = (A^+A)^{-1}A^+A\beta + A^+r$ LS solution

$A^+y = \beta + A^+r$ $\beta = A^+y$

$\Rightarrow A^+r = 0$

$AA^+y = A\beta + AA^+r$

These are "projection" matrices.

Let $P_A = AA^+$

$\Rightarrow P_Ay = A\beta + P_Ar$

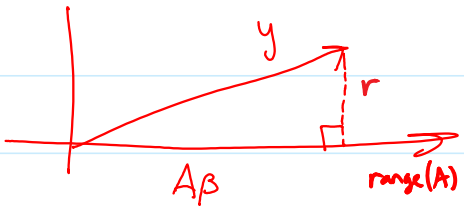
In the LS solution, $P_Ar = 0$

$y^Ty = (A\beta+r)^T(A\beta+r) = \beta^TA^TA\beta + \beta^TA^Tr + r^TA\beta + r^Tr$

If β is chosen so that $r \in \text{null}(A)$, then $\beta^TA^Tr = r^TA\beta = 0$.

$\Rightarrow \|y\|^2 = \|A\beta\|^2 + \|r\|^2$

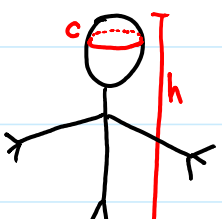
This is Pythagoras' Theorem.



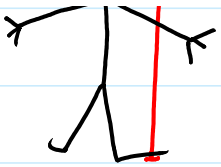
<u>Example Problem</u>

Scientists expect there to be a relationship between height and head circumference (each measured in cm).

The model we will use is



measured in cm).

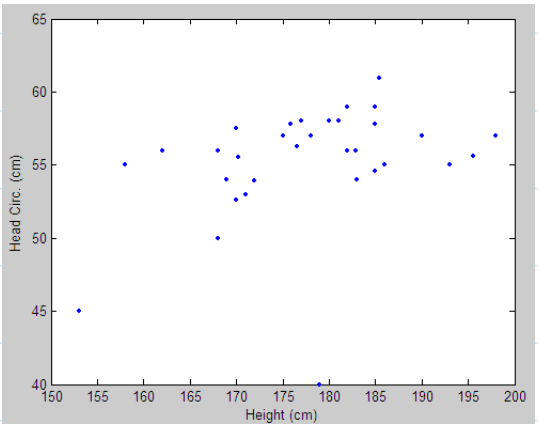


The model we will use is

$$c = a_0 + a_1 h + a_2 h^2 + a_3 h^3$$

32 people were measured, yielding the data

$$\{(h_1, c_1), (h_2, c_2), \ldots, (h_{32}, c_{32})\}$$



We formulate the problem as

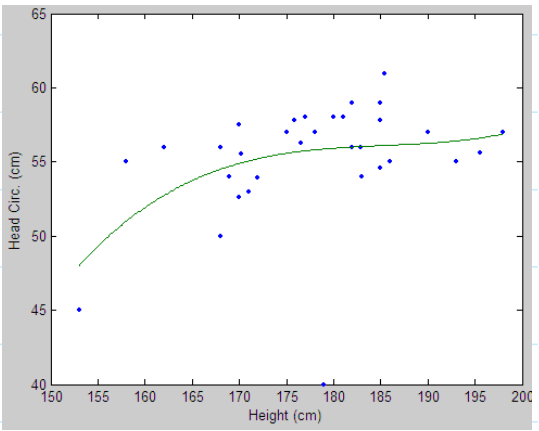
$$c = A\beta + r$$

where $A = [h^0 | h^1 | h^2 | h^3]$

Then compute $A^+ = (A^T A)^{-1} A^T$

Then, compute the optimal parameters, $\beta = A^+ c$

We can plot the curve fit over the data points.



In Matlab, there are 2 options:

1. `Ap = (A'*A)\A';` (Pseudo-inverse)
   `β = Ap*c;` (Solve)

2. `β = A\c;` (If A is not a square matrix, Matlab finds the LS solution for you.)

END