Lecture 17 - Least Square Fitting

* Prayer / Quiz / AMA.

* Go over results from Exam 2

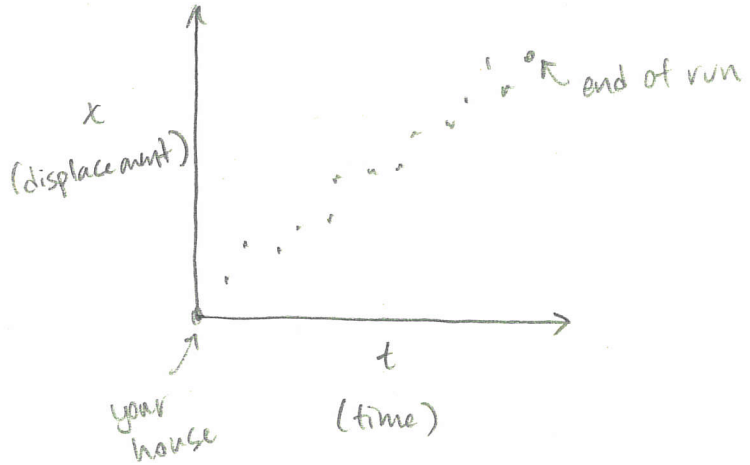
0. Numerical Calculus

* Last part of the class!
  - Fitting & Interpolation → How to turn data into functions/curves
  - Integration (need interpolation to do)
  - Simple differential equations (Don't need Diff Eq. class!)

* On the home stretch. Keep doing your homework. It will payoff.

I. Least Square Fitting

* Suppose you were wearing a fitbit that tracked your position with time while you went for a run. The data it reports might look like this



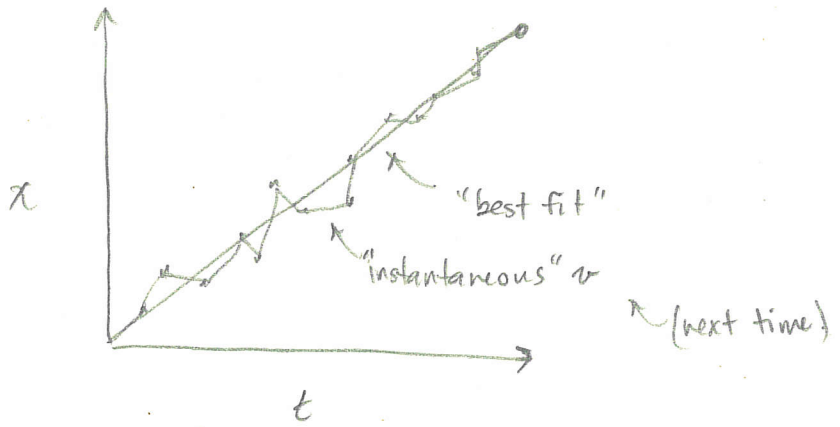
* If we assume a mathematical model where you ran at a constant speed,

$$x = v\,t$$

↑ displacement ← time ← velocity

how could we find your velocity?

* If we somehow found the "best" slope for v, that would tell us the velocity.



* we could get little lines to connect every point, but we don't think that would tell us very much. we would have to take an average or something to find v then. Plus a lot of the spread is not "real." It is just noise, e.g measurement error or small variations in speed we don't care about.

* Our model could also be wrong (here we assumed a constant velocity). Lets ignore that possibility for now

* If our model has noise, let's add that:

$$x_i = v t_i + \varepsilon_i$$

ε_i: noise

i: data point counter

$0, 1, \ldots, n-1$ points

* Least square fitting

- Vary the free parameter (in this case v) to minimize the amount of noise.

$$\varepsilon_i = x_i - v t_i$$

$$SSE = \sum_{i=0}^{n-1} (\varepsilon_i)^2 = \sum_{i=0}^{n-1} (x_i - v t_i)^2$$

SSE is a function of v. Minimize it to find v!

- This is like solving a non-linear equation with an optimization method. Here we don't expect SSE to go to zero, just that it will be as small as possible

- General formulation for $y = f(x)$:

$$\varepsilon_i = y_i - f(x_i)$$

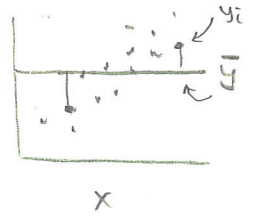
$$SSE = \sum_{i=0}^{n-1} (\varepsilon_i)^2 = \sum_{i=0}^{n-1} (y_i - f(x_i))^2$$

* Goodness of fit

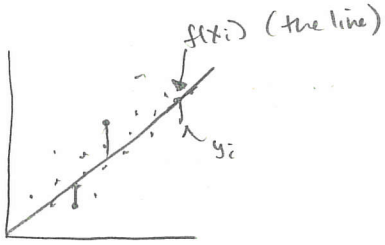
- We can tell how good the fit is by calculating the fraction of the "spread" of the data (variance) is noise or model.

Total spread: $SST = \sum_i (y_i - \bar{y})^2$

↑ average.



Spread from noise: $SSE = \sum_i (y_i - f(x_i))^2$



Fraction of spread from the model

$$R^2 = \frac{SST - SSE}{SST} \quad \left(\frac{\text{model} = \text{total} - \text{noise}}{\text{total}}\right)$$

↑ called "R-squared" or the coefficient of determination.

$$\boxed{R^2 = 1 - \frac{SSE}{SST}}$$

II. Curve Fitting in Excel

Activity

* Demonstration & Practice of curve fitting in Excel
  (1) via a trendline in a scatter plot
  (2) via minimization of SSE using solver
* Calculate R^2 using RSQ

III. Curve Fitting in Python

Activity

* Demonstration & practice of curve-fitting in Python
  (1) via `numpy.polyfit()` & `numpy.polyval()`
  (2) via `scipy.optimize.curve_fit()`

* Calculate R^2 by hand.

If Time

Trick: Linearizing Equations

* Suppose I want to fit

$$y = A e^{Bx}$$

This isn't on my "easy list". It isn't a polynomial.

* what if I take the log of both sides:

$$\ln y = \ln\left(A e^{Bx}\right)$$
$$= \ln A + \ln e^{Bx} = \ln A + Bx$$

let $z = \ln y \Rightarrow z = A' + Bx$
$A' = \ln A$

This is linear!

* Now, I can do a fit to find A' & B, then find A:

$$A = \exp(A')$$

trend line or np.polyfit()