
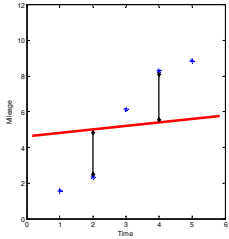


COMPUTER SCIENCE 51
 Spring 2009
 cs51.seas.harvard.edu


Prof. Greg Morrisett
Prof. Ramin Zabih

computer science 51 

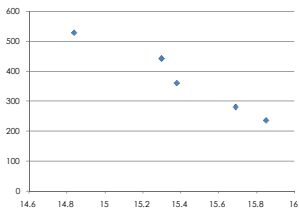
Summary of Least Squares




$$\text{Err}(m, b) = \sum_i [y_i - (mx_i + b)]^2$$



Nice example graph

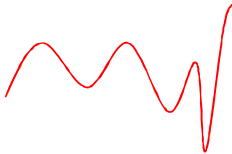



- From: <http://pubs.acs.org/doi/abs/10.1021/ci700332k>
 - c/o The Atlantic via Cindy Cheng



Why do hillclimbing?

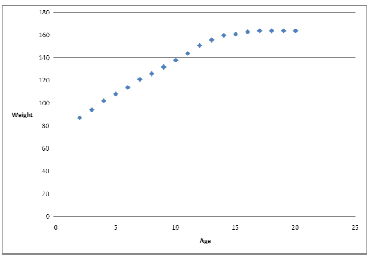
- It's rare to have an exact solution
 - Even for LS, some variant of hillclimbing is used in practice for big problems
- For non-convex functions, it's the most sensible idea (yields local minimum)






Recall: exact formula for LS

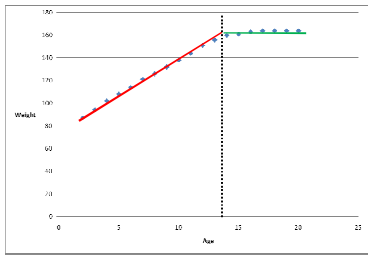
- But, suppose that our data did not come from a single line?






Let's pretend

- Suppose we knew how many lines there were, and the break locations





New error function

- Write break point between line 1 and line 2 as x_{12} , and data as $\{(x_i, y_i)\}$.
- Then our error function is

$$\text{Err}(m_1, b_1, m_2, b_2) = \sum_{x_i < x_{12}} (y_i - (m_1 x_i + b_1))^2 + \sum_{x_i > x_{12}} (y_i - (m_2 x_i + b_2))^2$$

- If we know there are 2 lines and also know the break point, this is convex
 - Just two convex subproblems

Find the break point?

- Suppose we don't know the break?
 - Try all, pick the best one

$$\text{Err}(m_1, b_1, m_2, b_2, x_{12}) = \sum_{x_i < x_{12}} (y_i - (m_1 x_i + b_1))^2 + \sum_{x_i > x_{12}} (y_i - (m_2 x_i + b_2))^2$$

- What if there are more than 2 lines?
- With N data points, at most $N - 1$ breaks
 - For the solution where each point has a line

Segmented least squares

- We don't know how many lines there are, or where the breaks occur
 - There are exponentially many places to put the breaks
 - Counting them is a cute math problem
- Lots of possibilities, non-convex error
 - Looks like big trouble
- Can we efficiently find best answer?
 - Yes we can™
 - Like Dijkstra's algorithm

What are we minimizing?

- Need penalty to introduce a new line
 - If this is free, you always get the answer where each point is its own line
 - This has 0 overall error (can't beat it)
- The total error will be the sum of:
 - each line's individual LS error, plus
 - c times the number of lines
- Think about very large or small c

Dijkstra analogy

- Suppose you need to visit, in order, cities $1, 2, \dots, N$
 - Cost of c to break for a night
 - Cost $e(i, j)$ to travel $i, i+1, \dots, j$ on same day
- We can travel optimally to city $N-1$, break overnight, then travel to city N
- Or: optimally to $N-2$, break, then $N, N-1$
- Or: optimally to $N-3$, etc.

Solution

- To compute $\text{OPT}(j)$, cheapest to p_j :
 - Last segment is p_i, p_{i+1}, \dots, p_j for some i .
 - Cost = $e(i, j) + c + \text{OPT}(i-1)$
 - Last break before p_i , cost depends on i
- So we have:
 - $\text{Opt}(1) = 0$
 - Consider a single point!
 - $\text{Opt}(j) = \min_{i < j} (e(i, j) + c + \text{OPT}(i-1))$
 - Find best place for the last break and recurse
 - Can compute the break locations as well

Least squares can fail

- Let's look at an amusing special case
 - Given a set of numbers x_i , find the "best guess" in the least squares sense
 - I.e. minimize the sum of squared errors

$$\text{Err}(x) = \sum (x_i - x)^2$$

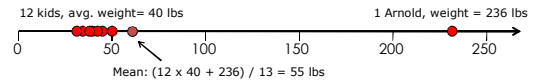
```
(define (sse l)
  (lambda (guess)
    (foldr + 0 (map (lambda (x) (square (- guess x))) l))))

(define (try-sse nums)
  (min-bin (sse nums)))
```



Mean can fail

- Average weight in kindergarten?

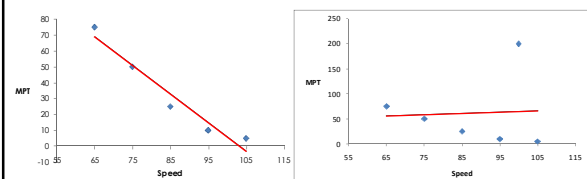


Outliers

- A single bad data point can trash anything based on least squares
 - Mean (see: Kindergarten Cop)
 - LS line (see: next slide)
- Such a point is called an **outlier**
- In small, low-dimensional data can often find them by hand
 - But we really need an algorithm
 - Either find outliers or ignore them



Outlier example



- Our friend gets lucky at 100 MPH
 - Two hours between tickets!



Ideas?

- Any ways to solve this
 - Hint: use something we taught?
- Reduce the weight of "bad" points
 - i.e., use weighted least squares

$$\text{Err}(m, b) = \sum_i w_i [y_i - (mx_i + b)]^2$$

- Make w_i small for bad points

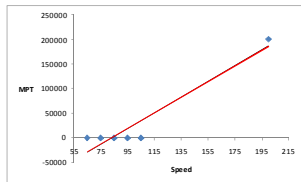


A tempting approach

- Maybe we can do something simpler
- Fit the data via LS
- Declare points with big residuals (i.e. far from the LS line) to be outliers
- Ignore them and recompute LS line
- Can you see any problem with this?
 - If so, you are ahead of some Ph.D.'s



Nice try, but no dice



- Bad data preferred to good data
- Algorithm soundness is black or white
 - Sound if it **always** works

How to do better?

- This feels a bit like segmented LS
 - If we knew the outliers, LS lines are easy
 - If we knew the LS lines, outliers are easy
- But we know neither
- Sometimes called a “chicken and egg” problem

Back to 1D case for insight

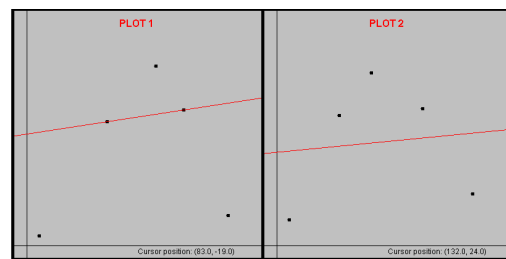
- The problem is that we are adding up squared residuals
 - So Arnold has outsized influence
- Obvious alternative: absolute value

$$\text{Err}(x) = \sum |x_i - x|$$

```
(define (sae l)
  (lambda (guess)
    (foldr + 0 (map (lambda (x) (abs (- guess x))) l))))

(define (try-sae nums)
  (min-bin (sae nums)))
```

Neither is perfect



Least absolute deviation

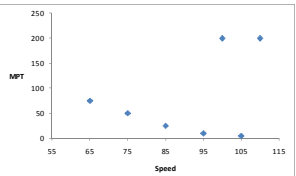
- Nice demo:
 - http://www.math.wpi.edu/Course_Materials/SAS/tablets/7.3/7.3c/index.html
- Much better tolerance of outliers
 - But outliers can still throw you off
- Best fit is not unique!
- How to compute?

Iterative Reweighted LS

- Variant of our bad idea
- If point p_i has residual r_i , weight it by $1/|r_i|$ and refit
 - Note that you need to handle small residuals differently, since they clearly should not get infinite weight
- There is also (sort of) a magic formula
 - It's not quite a closed form
 - But in practice IRLS is used instead

Can we fit either line?

- More generally, can we fit a line with:
 - A bare majority of good data
 - Almost 50% outliers?
- Yes we can, and it's easy to see how



Least Median Squares

- The problem with LS lies more in the **summing** than in the **squaring**
 - Let's replace

$$\text{Err}(m, b) = \sum (y_i - (mx_i + b))^2$$

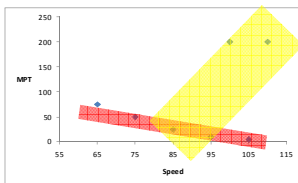
with

$$\text{Err}(m, b) = \text{median}\{(y_i - (mx_i + b))^2\}$$

- Only care about the median residual
 - Suppose 1/3 of the data is completely missed by a line (huge residuals), but 2/3 is very close to the line (tiny residuals). What happens?

Least Median Squares

- Find the “thinnest ruler” that covers more than half the data
 - Non-convex, but there are fast methods



Summary

- LS extensions for data not on 1 line
- Segmented LS is great
 - But special case
- LS doesn't tolerate outliers
- Least absolute deviation is better
 - But not perfect either
 - Compute via IRLS
- Least median squares is alternative