Point sets, Maps and Navigation - II

D.A. Forsyth

Robustness is a serious problem

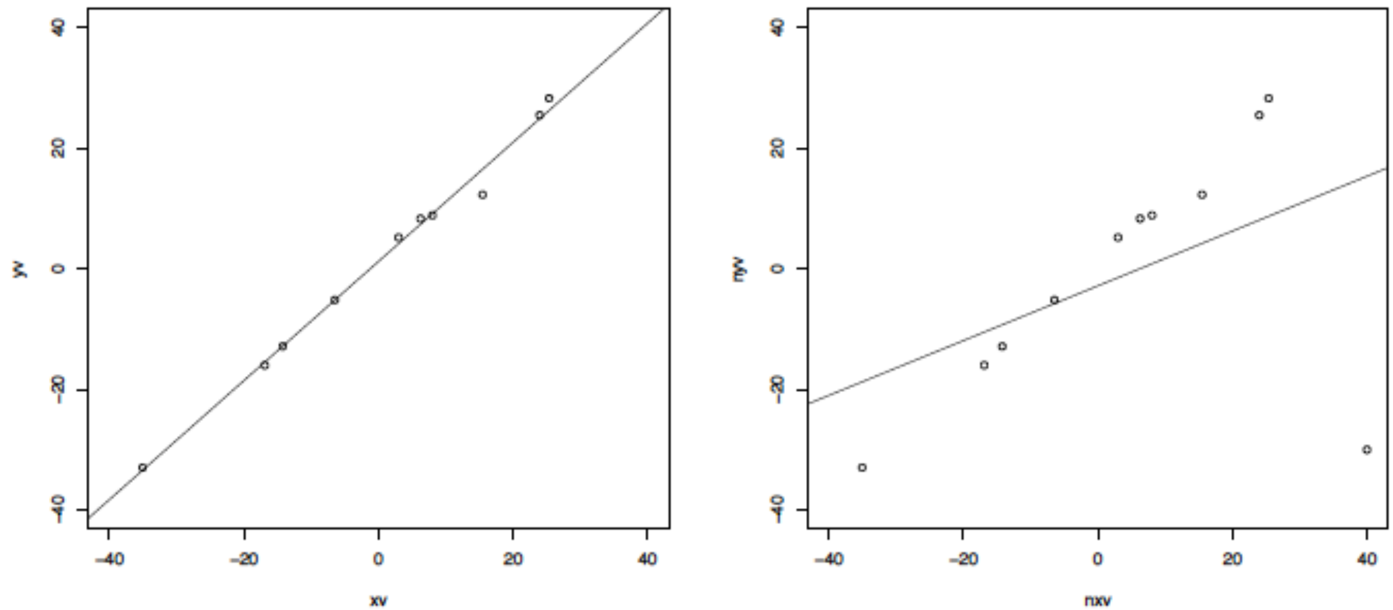


FIGURE 10.6: *On the* **left**, *a synthetic dataset with one independent and one explanatory variable, with the regression line plotted. Notice the line is close to the data points, and its predictions seem likely to be reliable. On the* **right**, *the result of adding a single outlying datapoint to that dataset. The regression line has changed significantly, because the regression line tries to minimize the sum of squared vertical distances between the data points and the line. Because the outlying datapoint is far from the line, the squared vertical distance to this point is enormous. The line has moved to reduce this distance, at the cost of making the other points further from the line.*

Robustness is a serious problem

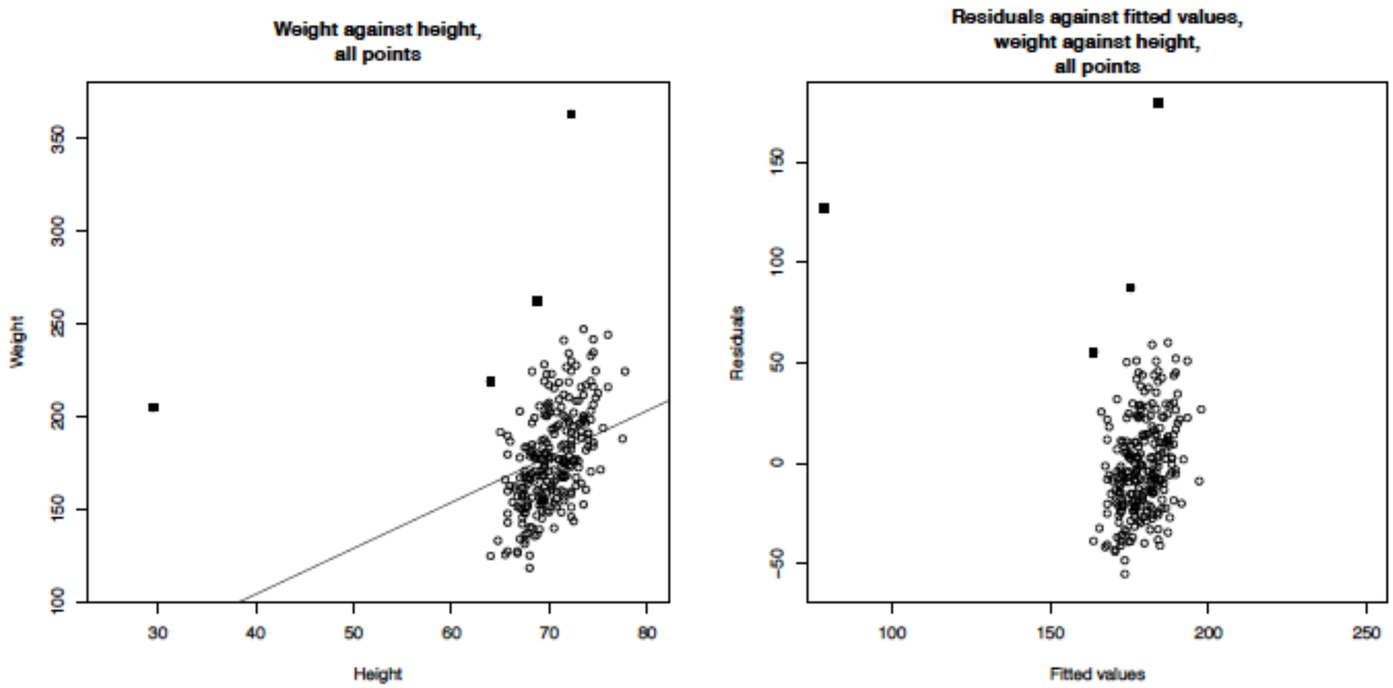


FIGURE 10.7: *On the* **left**, *weight regressed against height for the bodyfat dataset. The line doesn't describe the data particularly well, because it has been strongly affected by a few data points (filled-in markers). On the* **right**, *a scatter plot of the residual against the value predicted by the regression. This doesn't look like noise, which is a sign of trouble.*

Key issue:

- Squaring a large number produces a huge number
- A few wildly mismatch points can throw off R, t
- Fixes:
  - remove matches with “large” distances
    - actually, quite good
    - but what happens if new such pairs emerge?
  - apply an M-estimator
    - deals with new pairs

You should have watched the IRLS movie for regression by now

Some notation, etc.

- We know how to solve

$$\sum_i w_i \left(R\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\right)^T \left(R\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\right)$$

- Which is

$$\sum_i w_i \parallel R\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i \parallel^2$$

- Write as

$$\sum_i w_i f(r_i) \qquad \text{where} \qquad r_i = \parallel R\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i \parallel$$

We would like to solve

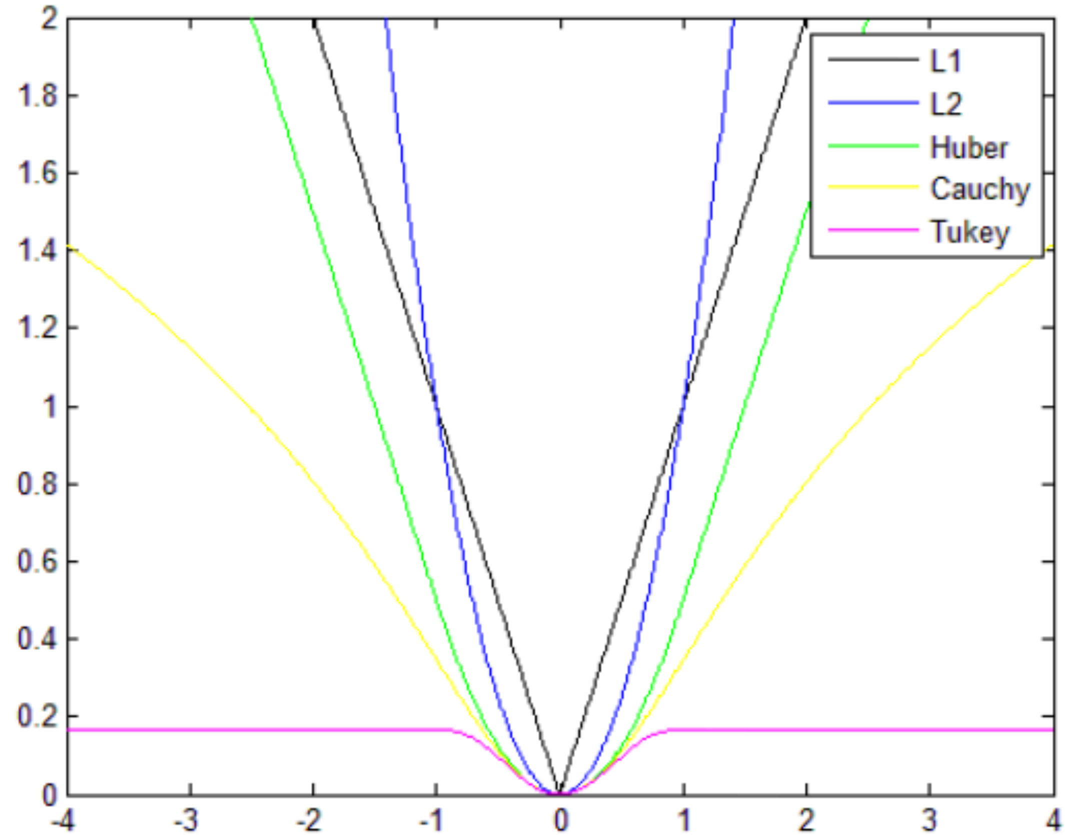
$$\sum_i \rho(r_i) \qquad \text{where} \qquad r_i = \| R\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i \|$$

- where rho could have a variety of useful properties
  - for example:
    - like quadratic for small d, constant for large d
    - like quadratic for small d, linear for large d
    - like absolute value

Non-linear estimation with robust loss function

$$\min_x \sum_i \rho_i(\|f_i(x_i)\|^2)$$

- Non-linear optimization (e.g. Levenberg-Marquard)
- Iteration necessary
- No explicit weight computation necessary
- Loss function should be differentiable
- Jacobian needs to be calculated



A clever trick

- Assume we are optimizing:

$$\sum_i \rho(d_i)$$

$$\sum_i w_i f(r_i)$$

- We are at solution if and only if:

$$\sum_i \frac{d\rho}{dr} \nabla r = 0$$

$$\sum_i w_i 2 r_i \nabla r = 0$$

Choose $$w_i = \frac{\left[\frac{d\rho}{dr}\right]}{r_i}$$

Then w_i are “right” at the true solution

IRLS

- Choose rho
- Iterate
    - find correspondences
    - from these, r_i, and w_i
    - solve (in closed form!)

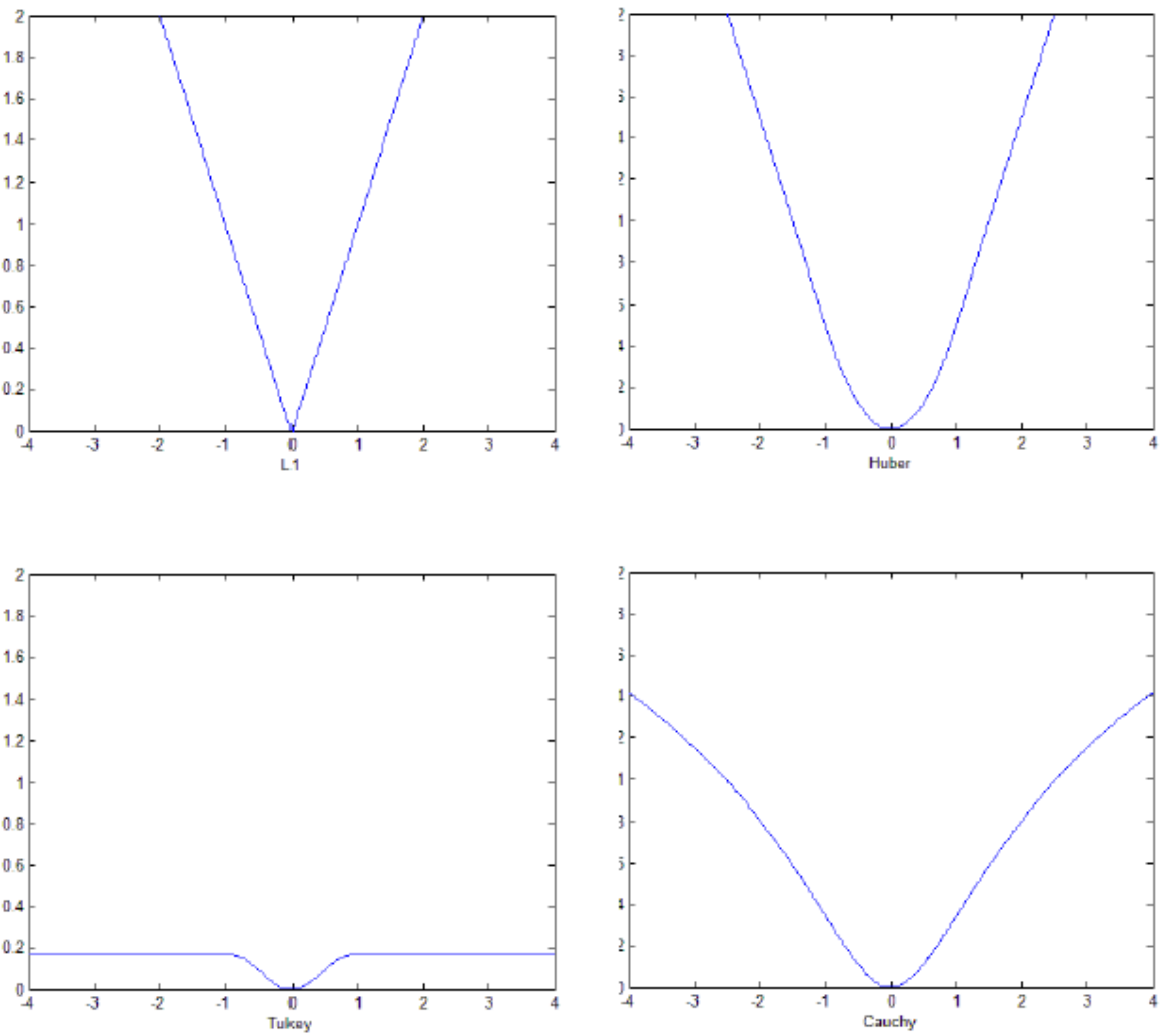
Examples

- rho is L1

$$\text{Choose} \qquad w_i = \frac{\left[\frac{d\rho}{dr}\right]}{r_i} = \frac{1}{r_i}$$

  - effect: make small residuals very important
  - what we expect - L1 minimization really likes zeros

More robust loss functions



loss function	p(x)
L_1	$\|x\|$
$Huber \begin{cases} if\ \|x\| \leq k \\ if\ \|x\| \geq k \end{cases}$	$\begin{cases} x^2/2 \\ k(\|x\| - \frac{k}{2}) \end{cases}$
$Tukey \begin{cases} if\ \|x\| \leq k \\ if\ \|x\| > k \end{cases}$	$\begin{cases} k^2/6(1 - \left(1 - \left(\frac{x}{c}\right)\right. \\ k^2/6 \end{cases}$
$Cauchy$	$\frac{k^2}{2}\log(1 + (x/k)^2)$

Examples

- Huber

$$\text{Choose} \qquad w_i = \frac{\left[\frac{d\rho}{dr}\right]}{r_i} = \begin{cases} 1 & r_i \leq k \\ \frac{k}{r_i} & \text{otherwise} \end{cases}$$

- recall r_i is absolute value of residual
- if point is close (r_i small) this behaves like least squares
- if point is far (r_i large) behaves like L1 - weight error down

Some important cautions

- Notation creates some confusion
  - some authors use $r_i=(Rx_i+t-y_i)^T(R x_i+t-y_i)$, etc….
    - which changes equations but nothing significant
    - as long as you're consistent…
- This is effective, but it isn't magic
  - these problems must have many local minima
    - and you could get one of those
  - it is *really* helpful to have a good start point