Exploratory Data Analysis

Transforms and Robust Regression

David B King, Ph.D.

September 27, 2015

Resistant Regression

Resistant Regression

Tukey's Robust-resistant line

How to fit straight lines? $\hat{y} = a + bx$

- Robust-resistant line:
  1. Sort the values of x, $(x_1 \leq x_2 \leq \cdots \leq x_n)$, and divide the n data points (x_i, y_i) into 3 nearly equal groups, according to the x values.
  2. Assume no ties among x_i's

	$n = 3k$	$n = 3k+1$	$n = 3k+2$
L=left group	k	k	$k+1$
M=middle group	k	$k+1$	k
R=right group	k	k	$k+1$

  3. Find summary points (median x-value, median y-value) in outer groups (x_L, y_L), (x_R, y_R)

$$\boxed{\text{slope } b = \frac{y_R - y_L}{x_R - x_L}}$$

  4. Intercept $\boxed{a = \text{median}\{y_i - bx_i\}}$

Tukey's Resistant Line Method

- Iteration 0 we compute b_0, a_0 and residuals $r_i^{(0)}$ by:

$$b_0 = \frac{\tilde{y}_R - \tilde{y}_L}{\tilde{y}_R - \tilde{y}_L}$$

$$a_0 = \frac{1}{3}[(\tilde{y}_L - b_0 X_L) + (\tilde{y}_M - b_0 X_M) + (\tilde{y}_R - b_0 X_R)]$$

$$r_i^{(0)} = y_i - [a_0 + b_0(x_i - M_x)]$$

- Iteration 1 we we treat $r^{(0)}$ as the new "y"-variable and if $\{\delta_1, \gamma_1\}$ are the slope and y-intercept of this new regression line then we update our slope, intercept and residual estimates by

$$b_1 = b_0 + \delta_1, \qquad a_1 = a_0 + \gamma_1$$

$$r_i^{(1)} = y_i - [a_1 + b_1(x_i - M_x)]$$

Tukey's Resistant Line Method

- After the first 2 iterations Tukey, then employs the "safe" iteration method. Otherwise the procedure might not converge.
- Note that after the j^{th} iteration $\Delta r(b) = \tilde{r}_R^{(j)} - \tilde{r}_L^{(j)} = \delta_{j+1}(\tilde{x}_R - \tilde{x}_L)$ which does not depend on the intercept estimates.
- Hence subsequent iterations for $j \geq 2$ are based off of

$$\boxed{b_j = b_{j-1} - \Delta r(b_{j-1}) \frac{b_{j-1} - b_{j-2}}{\Delta r(b_{j-1}) - \Delta r(b_{j-2})}}$$

- Tukey estimates intercept at each iteration using

$$a_j = \frac{1}{3}[(\tilde{y}_L - b_j X_L) + (\tilde{y}_M - b_j X_M) + (\tilde{y}_R - b_j X_R)]$$

- However, the rrline1 and run.rrline programs utilize

$$\boxed{a_j = \text{median}\{y_i - b_j x_i\}}$$

Example 1: Compare fits (residuals and coefficients)

Compare differences in slope and intercept

```
     UT = (67.3, 4.5)

  Min    1Q    Med   3Q    Max
-6.60 -2.22 0.00 2.27 6.95

          Estimate SE   t-stat
Intercept 19.222 3.092   6.22*
HSgrad    -0.223 0.057 -3.87*

R-sq=0.238 Adj R-sq=0.222

RRline:
Int=22.3020 Slope=-0.2847
```

```
     UT = (87.3, 14.5)

  Min    1Q    Med   3Q    Max
-6.51 -2.46 0.04 2.57 10.44

    Estimate SE   t-stat
    13.147 3.129   4.20*
    -0.104 0.058 -1.80

R-sq=0.064 Adj R-sq=0.044


Int=22.2789 Slope=-0.2843
```

LS line vs RR line

Compare:

- Difference in slope and intercept
- LS line: b is a weighted average of y_i's, weights depend on distance of x_i from $\bar{x}$. The further x_i is from $\bar{x}$, the greater its impact on slope. An outlier in y_i also affects the slope.

$$b = \sum_{i=1}^{n} c_i y_i, \quad c_i = \frac{x_i - \bar{x}}{\sum_{i=1}^{n} (x_i - \bar{x})^2}.$$

- RR line: A single x_i or y_i enters slope calculation only as a median, so value of an extreme x_i or y_i is not used.

LS line vs RR line

Compare:

- Difference in slope and intercept
- LS line: b is a weighted average of y_i's, weights depend on distance of x_i from $\bar{x}$. The further x_i is from $\bar{x}$, the greater its impact on slope. An outlier in y_i also affects the slope.

$$b = \sum_{i=1}^{n} c_i y_i, \quad c_i = \frac{x_i - \bar{x}}{\sum_{i=1}^{n}(x_i - \bar{x})^2}.$$

- RR line: A single x_i or y_i enters slope calculation only as a median, so value of an extreme x_i or y_i is not used.
- LS line: An outlier in y_i affects the intercept.

$$a = \begin{cases} \text{mean}\{y_i - bx_i\} - LS \\ \text{median}\{y_i - bx_i\} - RR \end{cases}$$

LS line vs RR line

Compare:

- Difference in slope and intercept
- LS line: b is a weighted average of y_i's, weights depend on distance of x_i from $\bar{x}$. The further x_i is from $\bar{x}$, the greater its impact on slope. An outlier in y_i also affects the slope.

$$b = \sum_{i=1}^{n} c_i y_i, \quad c_i = \frac{x_i - \bar{x}}{\sum_{i=1}^{n}(x_i - \bar{x})^2}.$$

- RR line: A single x_i or y_i enters slope calculation only as a median, so value of an extreme x_i or y_i is not used.
- LS line: An outlier in y_i affects the intercept.

$$a = \begin{cases} \text{mean}\{y_i - bx_i\} - LS \\ \text{median}\{y_i - bx_i\} - RR \end{cases}$$

- RR may need to iterate: fit line to residuals

Example 2: Brain and Body Weights for 28 Species

An example from R data library **MASS**

```
> library(MASS)
> data(Animals)
> help(Animals)
```

```
Description
Average brain and body weights for 28 species of land animals.

Format
body weight in kg.
brain weight in g.

Source: Rousseeuw and Leroy (1987) Robust Regression and
Outlier Detection. Wiley, p. 57.
```

Example 2: Data

	Mountain beaver	Cow	Grey wolf	Goat	Guinea pig
body	1.35	465	36.33	27.66	1.04
brain	8.10	423	119.50	115.00	5.50

	Dipliodocus	Asian elephant	Donkey	Horse	Potar monkey	Cat
body	11700	2547	187.1	521	10	3.3
brain	50	4603	419.0	655	115	25.6

	Giraffe	Gorilla	Human	African elephant	Triceratops
body	529	207	62	6654	9400
brain	680	406	1320	5712	70

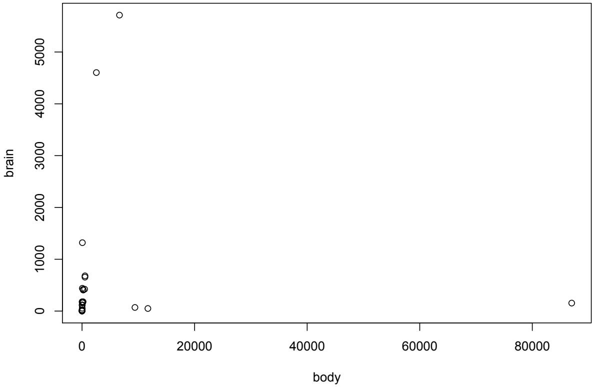
	Rhesus monkey	Kangaroo	Golden hamster	Mouse	Rabbit	Sheep
body	6.8	35	0.12	0.023	2.5	55.5
brain	179.0	56	1.00	0.400	12.1	175.0

	Jaguar	Chimpanzee	Rat	Brachiosaurus	Mole	Pig
body	100	52.16	0.28	87000.0	0.122	192
brain	157	440.00	1.90	154.5	3.000	180

Example 2: Plot data

```
Animals.abb <- c("Bvr","Cow","Wolf","Goat","Gpig","Dipl",
"AsEl","Dnky","Hors","Pmky","Cat","Grff","Grll","Humn",
"AfEl","Tric","Rmky","Kngr","Hmst","Mous","Rbbt","Shep",
"Jagr","Cmpz","Rat","Brac","Mole","Pig")

body <- Animals[,1]

brain <- Animals[,2]

plot(body,brain)
```

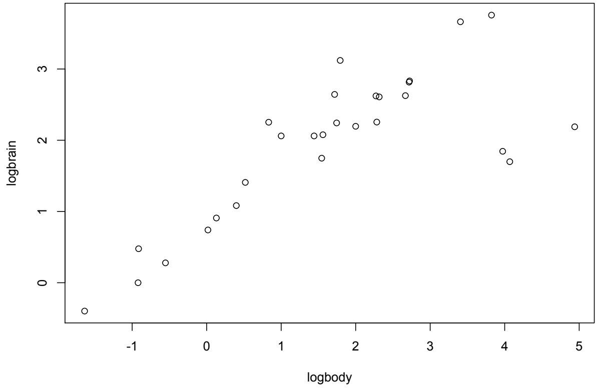
Example 2: Scatterplot



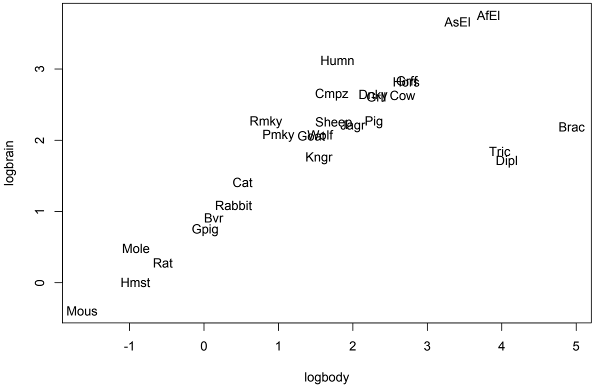
Example 2: Log Transformation

```
logbody <- log10(body)
logbrain <- log10(brain)

plot(logbody, logbrain)

plot(logbody, logbrain, type='n')
text(logbody, logbrain, Animals.abb)
```

Example 2: Plot 1 of transformed data



Example 2: Plot 2 of transformed data



Example 2: Sort transformed data

```
sort.logbb <- cbind(sort(logbody), logbrain[order(logbody)])

dimnames(sort.logbb) <- list(Animals.abb[order(logbody)],
c("logbody","logbrain"))

round(t(sort.logbb), 3)
```

Example 2: Sorted data

	Mous	Hmst	Mole	Rat	Gpig	Bvr	Rabbit
logbody	-1.638	-0.921	-0.914	-0.553	0.017	0.130	0.398
logbrain	-0.398	0.000	0.477	0.279	0.740	0.908	1.083

	Cat	Rmky	Pmky	Goat	Kngr	Wolf	Cmpz
logbody	0.519	0.833	1.000	1.442	1.544	1.560	1.717
logbrain	1.408	2.253	2.061	2.061	1.748	2.077	2.643

	Sheep	Humn	Jagr	Dnky	Pig	Grll	Cow
logbody	1.744	1.792	2.000	2.272	2.283	2.316	2.667
logbrain	2.243	3.121	2.196	2.622	2.255	2.609	2.626

	Hors	Grff	AsEl	AfEl	Tric	Dipl	Brac
logbody	2.717	2.723	3.406	3.823	3.973	4.068	4.940
logbrain	2.816	2.833	3.663	3.757	1.845	1.699	2.189

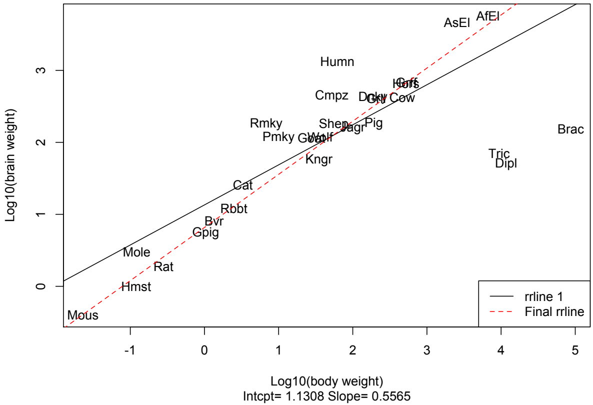
Example 2: rrline1

```
rrline1 <- function(x,y) {
  n <- length(x); nmod3 <- n%%3;
  if(nmod3 == 0) n3 <- n/3;
  if(nmod3 == 1) n3 <- (n-1)/3;
  if(nmod3 == 2) n3 <- (n+1)/3;
#  n3 <- floor((length(x)+1.99)/3)
  x.order <- order(x)
  medxL <- median(x[x.order][1:n3])
  medxR <- median(rev(x[x.order])[1:n3])
  medyL <- median(y[x.order][1:n3])
  medyR <- median(rev(y[x.order])[1:n3])
  slope1 <- (medyR - medyL)/(medxR - medxL)
  int1 <- median(y - slope1 * x)
  newy <- y - slope1*x - int1
  sumres <- sum(abs(newy))
  list(a=int1, b=slope1, sumres = sumres, res=newy)
}
```

Example 2: rrline1

```
rr1 <- rrline1(logbody, logbrain)

plot(logbody, logbrain, xlab="Log10(body weight)",
ylab="Log10(brain weight)", type="n", sub=
paste(paste("Intcpt=", format(round(rr1$a,4))),
      paste("Slope=",format(round(rr1$b,4)))))


text(logbody,logbrain,Animals.abb)
abline(rr1$a,rr1$b)

rr.bb <- run.rrline(logbody, logbrain)
abline(rr.bb$coef[6,1], rr.bb$coef[6,2], col=2, lty=2)
```

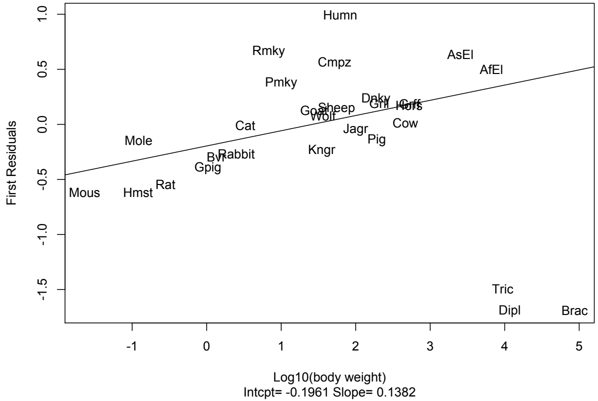
Example 2: rrline1



Example 2: rrline 2 fits first residuals

```
rr2 <- rrline1(logbody, rr1$res)

plot(logbody, rr1$res, xlab="Log10(body weight)",
ylab="First Residuals", type="n", sub=
paste(paste("Intcpt=", format(round(rr2$a,4))),
      paste("Slope=",format(round(rr2$b,4)))))

text(logbody,rr1$res,Animals.abb)
abline(rr2$a,rr2$b)
```

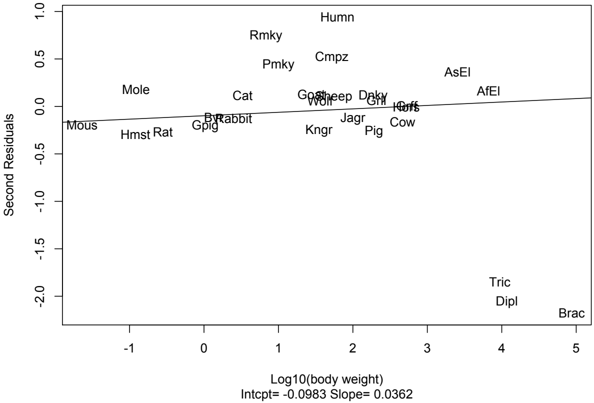
Example 2: rrline 2 fits first residuals



Example 2: rrline 3 fits second residuals

```
rr3 <- rrline1(logbody, rr2$res)

plot(logbody, rr2$res, xlab="Log10(body weight)",
ylab="Second Residuals", type="n", sub=
paste(paste("Intcpt=", format(round(rr3$a,4))),
      paste("Slope=",format(round(rr3$b,4)))))

text(logbody,rr2$res,Animals.abb)
abline(rr3$a,rr3$b)
```

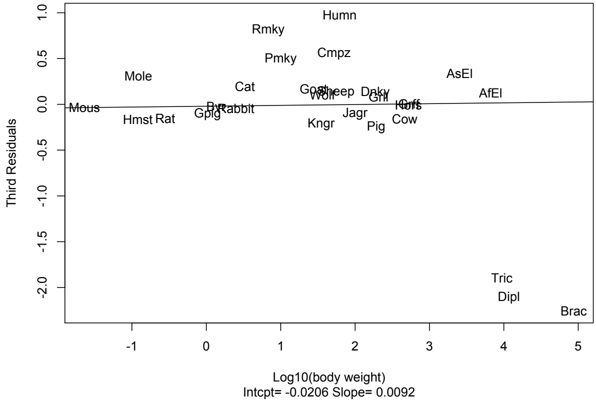
Example 2: rrline 3 fits second residuals



Example 2: rrline 4 fits third residuals

```
rr4 <- rrline1(logbody, rr3$res)

plot(logbody, rr3$res, xlab="Log10(body weight)",
ylab="Third Residuals", type="n", sub=
paste(paste("Intcpt=", format(round(rr4$a,4))),
      paste("Slope=",format(round(rr4$b,4)))))

text(logbody,rr3$res,Animals.abb)
abline(rr4$a,rr4$b)
```

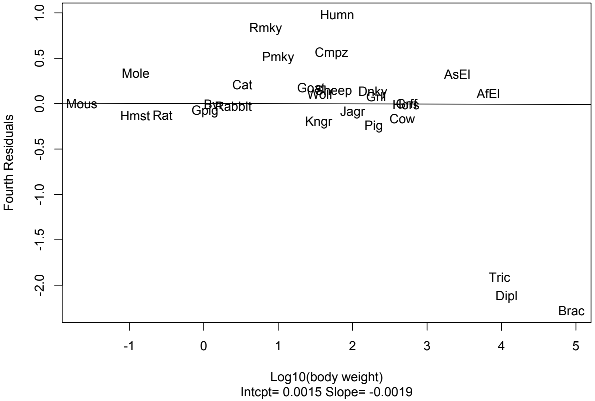
Example 2: rrline 4 fits third residuals



Example 2: rrline 5 fits fourth residuals

```
rr5 <- rrline1(logbody, rr4$res)

plot(logbody, rr4$res, xlab="Log10(body weight)",
ylab="Fourth Residuals", type="n", sub=
paste(paste("Intcpt=", format(round(rr5$a,4))),
      paste("Slope=",format(round(rr5$b,4)))))

text(logbody,rr4$res,Animals.abb)
abline(rr5$a,rr5$b)
```

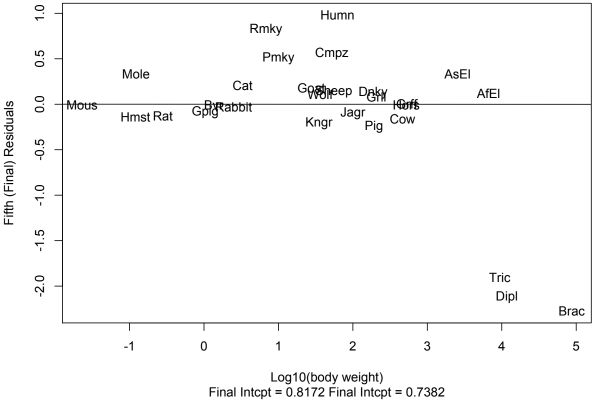
Example 2: rrline 5 fits fourth residuals



Example 2: final residuals

```
plot(logbody, rr5$res, xlab="Log10(body weight)",
   ylab="Fifth (Final) Residuals", type="n", sub=
   paste(
paste("Final Intcpt =",format(round(rr.bb$coef[6,1],4))),
paste("Final Intcpt =",format(round(rr.bb$coef[6,2],4)))))

text(logbody,rr.bb$res,Animals.abb)
abline(0.00027,-0.00019)
```

Example 2: final residuals



Example 2: rrline

```
run.rrline <- function(xx,yy,iter=5) {
  out.coef <- matrix(0,iter,3)
  l <- (1:length(xx))[!is.na(xx) & !is.na(yy)]; n <- length(l)
  x <- xx[l];  y <- yy[l]; newy <- y
  for (i in 1:iter) {
    rr <- rrline1(x,newy)
    out.coef[i,] <- c(rr$a,rr$b,rr$sumres)
    newy <- rr$res
  }
  dimnames(out.coef) <- list(format(1:iter),c("a","b","|res|"))
  aa <- sum(out.coef[,1])
  bb <- sum(out.coef[,2])
  cc <- sum(abs(y - aa - bb*x))
  res <- y - aa - bb*x
  out.coef <- rbind(out.coef,c(aa,bb,cc))
  print(round(out.coef,5))
  list(a = aa, b = bb, res = res, coef=out.coef)
}
```

Example 2: rrline, coefficients and residuals

```
run.rrline(logbody, logbrain)

$res
 [1] -0.00493 -0.15990  0.10842  0.17916 -0.08942 -2.12125
 [7]  0.33161  0.12783 -0.00645  0.50532  0.20828  0.00493
[13]  0.08174  0.98028  0.11750 -1.90495  0.82111 -0.20880
[19] -0.13748 -0.00582 -0.02817  0.13825 -0.09765  0.55856
[25] -0.13036 -2.27448  0.33434 -0.24740
$coef
             a            b    |res|
1  1.130794115  0.556500479 13.07018
2 -0.196111359  0.138201888 12.02777
3 -0.098333309  0.036150002 11.93348
4 -0.020620942  0.009217754 11.90944
5  0.001476957 -0.001899684 11.91439
   0.817205463  0.738170440 11.91439
```

Some notes

- Difference between book and above in estimating intercept!

$$a = \frac{1}{3}\left[(y_L - bx_L) + (y_M - bx_M) + (y_R - bx_R)\right]$$

$$a = \text{median}\{y_i - bx_i\}$$

  Very similar performance – use whichever is easiest.

- RRline in transformation plots

Measures of Resistance

Operationally, we can think about what would happen to regression line estimates if we take data points $\{(x_i, y_i)\}$ and "send them to infinity"

Definition of Breakdown Bound

The *breakdown bound* of a procedure for fitting a line to n pairs of data $\{(x_i, y_i)\}$ is k/n, if k is the greatest number of data points that can be replaced by arbitrary values while always leaving the slope and intercept estimates bounded.

Other Robust Lines

1. The Brown-Mood line: $\hat{y} = a_{BM} + b_{BM}x$
   - Two groups (spilt at median)
   - b_{BM} and a_{BM} are chosen to yield zero median residual in each of the two groups

$$\operatorname*{med}_{x_i \leq M_x} \{y_i - a_{BM} - b_{BM}x_i\} = 0$$

$$\operatorname*{med}_{x_i > M_x} \{y_i - a_{BM} - b_{BM}x_i\} = 0$$

   - Calculate b_{BM} using an iterative procedure

$$a_{BM} = \text{med}\{y_i - b_{BM}x_i\}$$

Other Robust Lines

2. Bartlett's method
    - 3 groups
    - 3 summary points: mean
    - $$b_B = \frac{\bar{y}_U - \bar{y}_L}{\bar{x}_U - \bar{x}_L}, \quad a_B = \bar{y} - b_B \bar{x}$$

Other Robust Lines

3. Wald's method
   - Two groups
   - 2 summary points: mean
   - $$b_W = \frac{\bar{y}_U - \bar{y}_L}{\bar{x}_U - \bar{x}_L}, \quad a_W = \bar{y} - b_W \bar{x}$$

	Summary measure	
# groups	Mean	Median
2	Wald	Brown and Mood
3	Barlett	3-group RR line

Breakdown bound[1]: $LS = 0$, $RR = 1/6$

[1] The *breakdown bound* of a procedure for fitting a line to n pairs of y-versus-x data is k/n, where k is the greatest number of data points that can be replaced by arbitrary values while always leaving the slope and intercept bounded.

Other Robust Lines

4. Least absolute residuals (LAR)

$$\min \sum |y_i - a - bx_i|$$

- No explicit formulas for $\hat{\alpha}$, $\hat{\beta}$, solve computationally
- May not even be unique
- Not robust, but less sensitive to moderate disturbance than LS

Other Robust Lines

5. Median of pairwise slopes (Theil 1950)

   median $b_{ij} = (y_j - y_i)/(x_j - x_i), \quad 1 \leq i < j \leq n, \quad b_T = \text{med}\{b_{ij}\}$

   - $n(n-1)$ possible b_{ij}
   - If k points are "wild" $\Rightarrow k(k-1)/2 + k(n-k)$ "wild" slopes (affected by 2 or 1 of k wild points)
   - Need $k(k-1)/2 + k(n-1) < 0.5n(n-1)/2$ $(k/n \approx 0.29)$; Otherwise, b_T is wild

Other Robust Lines

6. Repeated Median Line

$$b_{RM} = \text{med}_i\{\text{med}_{j\neq i}\{b_{ij}\}\}$$

- For each point $\{x_i, y_i\}$, find $s_i \equiv$ median of all slopes through it
- Take median of s_i's as slope
- $a_{RM} = \text{median}_i\{y_i - b_{RM}x_i\}$
- Use each b_{ij} twice, think of a matrix of slopes (first find medians of rows, and then take median; # 5: median of the matrix)

$$\begin{pmatrix} & b_{12} & b_{13} & \cdots & b_{1n} \\ b_{21} & & b_{23} & \cdots & b_{2n} \\ \vdots & \ddots & \ddots & \cdots & \vdots \\ b_{n1} & \cdots & \cdots & \cdots & \end{pmatrix}$$

- $k \leq n/2$ wild points

Other Robust Lines

7. Least median of squares $b_{LMS} = \text{argmin}\{\text{median}(y_i - bx_i)\}$
   - minimize median, not mean/sum
   - very robust, breakdown $\approx 1/2$
   - computationally cumbersome
   - requires two sorts on n observations, $2O(n \log n)$ operations

Other Robust Lines

8. General M-estimation

$$b_M = \operatorname{argmin} \sum_{i=1}^{n} \rho(y_i - \beta x_i)$$

- $\rho(\cdot) = (\cdot)^2 \Rightarrow$ LS too sensitive to outliers
- $\rho(\cdot) = |\cdot| \Rightarrow$ LAR too sensitive to the middle observations
- Huber estimator: quadratic in the center and linear in the tails (Ch11, P371)

$$\rho(u) = u^2 I_{[-k,k]} + (2k u \operatorname{sign}(u) - k^2) I_{[k,\infty]}(|u|)$$

- Redescending estimators (or Hampels), very robust, breakdown: 0.5
- Find iteratively (w-iteration)

Resistance and efficiency

How to choose from available methods?

1. Degrees of resistance
2. Break Down Bound
3. Relative precision of the slope estimates
4. Increased resistance requires more computational effort (sort)

Resistant Multiple Regression

Resistant Multiple Regression
Chapter 7 of EDTTS

Resistant Multiple Regression

Goal: seek a resistant multiple-regression fit

$$\hat{y} = b_0 + b_1 x_1 + \cdots + b_k x_k$$

- Proper interpretation of b_j: how y depends, on average, to a change in x_j, after allowing for simultaneous linear change in other x_j's.
- Improper interpretation of b_j: how y responds, on average, to a change in x_j, after controlling for (holding fixed) other x_j's.

Sweeping Out

Tukey described a one variable at a time procedure involving multiple uses of the action “sweeping out”.

Sweeping Out

Suppose we have two columns of numbers corresponding to variable A and variable B. We sweep the variable A out of B when we fit a resistant line using B as the response and A as the predictor and then remove the estimated slope $\hat{c}$ times the variable A from the variable B. The operation of sweeping a variable out of another, removes the linear dependence and leaves behind the intercept plus the residual.

When X_1 is swept out of Y it yields the variable

$$\boxed{Y_{\cdot 1} = Y - \hat{c}X_1}$$

Tukey's Procedure For 2 Variables

We seek a resistant multiple regression of the form

$$\hat{Y} = b_0 + b_1 X_1 + b_2 X_2$$

Procedure:

1. Sweep X_1 out of X_2 to obtain

$$X_{2 \cdot 1} = X_2 - d_1 X_1$$

2. Sweep X_1 out of Y to obtain

$$Y_{\cdot 1} = Y - c_1 X_1$$

Tukey's Procedure For 2 Variables

3. Sweep $X_{2.1}$ out of $Y_{.1}$ to obtain
$$Y_{.12} = Y_{.1} - c_2 X_{2.1}$$
4. (Refinement step) If the slope of $Y_{.12}$ against X_1 is not "small enough", remove this additional multiple of X_1 from $Y_{.12}$ and adjust c_1; otherwise go to step 6.
5. (Refinement step) If the slope of $Y_{.12}$ against $X_{2.1}$ is not "small enough", remove this additional multiple of $X_{2.1}$ from $Y_{.12}$ and adjust c_2 and return to step 4; otherwise go to step 6.
6. Using the final c_1 and c_2, calculate
$$\begin{aligned} b_1 &= c_1 - c_2 d_1 \\ b_2 &= c_2 \\ b_0 &= \text{median}\{Y_i - b_1 X_{i1} - b_2 X_{i2}\} \end{aligned}$$
7. Form the residuals
$$r_i = Y_i - \hat{Y}_i = Y_i - \{b_0 + b_1 X_{i1} + b_2 X_{i2}\}$$

Multiple Regression

Multiple regression by iterative fittings: $y = b_0 + b_1 x_1 + b_2 x_2$

(1a) Fit y using x_1

(2a) Fit residuals from (1a) to x_2

(1b) Fit residuals from step (2a) to x_1

(2b) Fit residuals from step (1b) to x_2

(1c) Fit residuals from step (2b) to x_1

(2c) Fit residuals from step (1c) to x_2, ...
until coefficeints are effectively zero

When fitting by least squares, trial residuals will be the same as the least square residuals. A very slow way of obtaining the LS fit.

Multiple Regression

More direct way to obtain LS fit w/o iteration: adjust each variable for successive ones ("sweeping")

1. Fiy y using x_1 only without x_2: $y = a_1 x_1$
2. Residuals $y_{\cdot 1} = y - a_1 x_1$
3. Adjust x_2 for x_1: $x_{2\cdot 1} = x_2 - c_1 x_1$
4. $y_{\cdot 1} = d_0 + d_1 x_{2\cdot 1} + y_{\cdot 12}$
5. b_2 in original regression equation is exactly the slope of y-residuals $y_{\cdot 1}$ on x-residuals $x_{2\cdot 1}$
6. Interpretation of b_2 is ....?
7. What does $x_{2\cdot 1}$ tell you? The "information" in x_2, alone w/o x_1
8. $y_{\cdot 12} = y$ adjusted for both x_1 and x_2

Multiple Regression

See how this process gives usual LS coefficient:

1. Fit y to x, and form residuals
$$\hat{y} = c_0 + c_1 x_1, \quad c_1 = \frac{\sum_{i=1}^n (x_{i1} - \bar{x}_1) y_i}{\sum_{i=1}^n (x_{i1} - \bar{x}_1)^2}, \quad y_{\cdot 1} = y - c_1 x_1$$
2. Fit x_2 to x_1 and form residuals:
$$\hat{x}_2 = d_0 + d_1 x_1, \quad d_1 = \frac{\sum_{i=1}^n (x_{i1} - \bar{x}_1) x_{i2}}{\sum_{i=1}^n (x_{i1} - \bar{x}_1)^2}, \quad x_{2\cdot 1} = x_2 - d_1 x_1$$
3. Fit $y_{\cdot 1}$ to $x_{2\cdot 1}$ and form residuals:
$$\hat{y}_{\cdot 1} = c_0 + c_2 x_{2\cdot 1}, \quad c_2 = \frac{\sum_{i=1}^n (x_{2\cdot 1,i} - \bar{x}_{2\cdot 1}) y_{\cdot 1,i}}{\sum_{i=1}^n (x_{2\cdot 1,i} - \bar{x}_{2\cdot 1})^2}, \quad y_{\cdot 12} = y_{\cdot 1} - c_2 x_{2\cdot 1}$$

Least square fit: $\hat{y} = b_0 + b_1 x_1 + b_2 x_2$, where
$$b_2 = c_2, b_1 = c_1 - d_1 c_2, b_0 = \text{mean}\{y_i - b_1 x_{i1} - b_2 x_{i2}\}$$

Multiple Regression

Note ($\star$): If you plot $y_{\cdot 12}$ against x_1 (or against x_2), the LS slope of the line willbe zero.

Everywhere in the previous steps, replace "LS fit" with "RR line fit", but note ($\star$) no longer applies, so we must iterate

4. plot $y_{\cdot 12}$ against x_1. If slope is not "small", adjust c_1 slope accordingly. forming new c_1, $y_{\cdot 1}$, c_2, $y_{\cdot 12}$.
5. Plot $y_{\cdot 12}$ against $x_{2\cdot 1}$. If slope is not "small", adjust c_2 slope accordingly. And return to (4), otherwise go to (6).
6. Obtain final residuals from step (5), final fit = data - residuals, or from assembling all coefficients from all steps, or form $\hat{y} = b_0 + b_1 x_1 + b_2 x_2$, where $b_2 = c_2$, $b_1 = c_1 - d_1 c_2$, $b_0 = \text{median}\{y_i - b_1 x_{i1} - b_2 x_{i2}\}$

Multiple Regression

Exception: Designed experiments (Orthogonal Columns)

Scenario: How does lung capacity depend on:

1. x_1 = age (14 vs 17, code as -1 vs 1)
2. x_2 = gender (male vs female, code as -1 vs 1)
3. x_3 = training (none vs weekly, code as -1 vs 1)

Measure 10 individuals in each of these 8 classes:

			x_0	x_1	x_2	x_3
14	M	0	1	-1	-1	-1
14	M	W	1	-1	-1	1
14	F	0	1	-1	1	-1
14	F	W	1	-1	1	1
17	M	0	1	1	-1	-1
17	M	W	1	1	-1	1
17	F	0	1	1	1	-1
17	F	W	1	1	1	1

Multiple Regression

Exception: Designed experiments (Orthogonal Columns)
Note:

- $\sum_{i=1}^{8} x_{ij}x_{im} = 0$ for every pair $(j; m)$ of variables: the variables are orthogonal
- For this situation, a "unit" change in $x - j$ corresponds to $b_j/2$ change in volume (if 'age 14' to 'age 17', 'male' to 'female', and 'o' to 'weekly' are considered 'unit changes')
- Designed experiments are wonderful if conductible

Multiple Regression

Exception: Designed experiments (Orthogonal Columns)
Still, the coefficient of any one variable depends on its costock and how it is expressed.

- Fit (1): $y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3$, suppose $x_1^* = c_0 + c_1 x_1 + c_2 x_2 + c_3 x_3$
- Fit (2): $y = B_0 + B_1 x_1^* + B_2 x_2 + B_3 x_3$
- Note: $b_1 = c_1 B_1$
- If we are interested in the coefficient of x_1, we need to compare it with the coefficients of its costock
  Fit (1): compare b_1 with (b_0, b_2, b_3)
  Fit (2): compare B_1 with (B_0, B_2, B_3)
- When interpreting b_j, consider x_j's costock and the entire stock (what variables are included/excluded), not just x_j.

Let's Practice on Stack Loss

Let's take Y = stack.loss, X_1 = Air.Flow and, X_2 = Water.Temp.

```
> stackloss
   Air.Flow Water.Temp Acid.Conc. stack.loss
1        80         27         89         42
2        80         27         88         37
3        75         25         90         37
4        62         24         87         28
5        62         22         87         18
6        62         23         87         18
7        62         24         93         19
8        62         24         93         20
9        58         23         87         15
10       58         18         80         14
11       58         18         89         14
12       58         17         88         13
13       58         18         82         11
14       58         19         93         12
15       50         18         89          8
16       50         18         86          7
17       50         19         72          8
18       50         19         79          8
19       50         20         80          9
20       56         20         82         15
21       70         20         91         15
```

Steps 1 & 2

```
y=stack.loss
x1=Air.Flow
x2=Water.Temp
# Step 1
fit1=run.rrline(x2,x1)
   a       b     |res|
1 46 0.66667 107.6667
2  0 0.00000 107.6667
3  0 0.00000 107.6667
4  0 0.00000 107.6667
5  0 0.00000 107.6667
  46 0.66667 107.6667

d1 = 0.66667
x2.1 = x2 - d1 * x1

# Step 2
fit2=run.rrline(x1,y)
         a       b   |res|
1 -20.8000  0.6000 79.8000
2 -13.2000  0.2400 57.1600
3  -5.0320  0.0960 53.4720
4  -2.3808  0.0384 52.5888
5   0.7192 -0.0116 52.8556
  -40.6936  0.9628 52.8556

c1 = 0.9628
y.1= y - c1*x1
```

Steps 3 & 4

```
# step 3
fit3=run.rrline(x2.1,y.1)
          a         b     |res|
1 -37.64101 0.17611 50.80101
2   0.89571 0.06073 50.24944
3   0.30015 0.02094 50.06796
4   0.10350 0.00722 50.00538
5   0.03569 0.00249 49.98380
  -36.30596 0.26749 49.98380
c2=0.26748
y.12=y.1-c2*x2.1

#step 4
fit4=run.rrline(x1,y.12)
          a         b     |res|
1 -39.92866 0.06244 48.29884
2  -1.45610 0.02498 47.81685
3  -0.52435 0.00999 47.68214
4  -0.19982 0.00400 47.63818
5  -0.07993 0.00160 47.62059
  -42.18886 0.10301 47.62059
y.12 = y.12 - 0.10301*x1 # adjust Y variable
c1 = c1 + 0.10301 # adjust slope for X1
```

Steps 5, 6 & 7

```
# Step 5
fit5=run.rrline(x2.1,y.12)
          a       b     |res|
1 -40.50932 0.08611 46.80302
2   0.61369 0.02969 46.55557
3   0.21162 0.01024 46.47024
4   0.07297 0.00353 46.44082
5   0.02516 0.00122 46.43067
  -39.58588 0.13080 46.43067
y.12 = y.12 - 0.1308*x2.1
c2 = c2 + 0.1308

#back to step 4
fit6=run.rrline(x1,y.12)
          a       b     |res|
1 -40.76068 0.02026 46.04576
2  -0.46996 0.00810 45.89180
3  -0.18798 0.00324 45.83022
4  -0.07519 0.00130 45.80559
5  -0.03008 0.00052 45.79574
  -41.52389 0.03342 45.79574
# Let's say this is close to zero

# Step 6 Final Model coefficients \& residuals
b1 = c1 - c2*d1
b2 = c2
b0 = median(y - b1*x1 + b2*x2)
r = y - b0 - b1*x1 + b2*x2
```

Multiple Regression Comparison

Final resistant regression model is

$$Y = b_0 + b_1 X_1 + b_2 X_2 = -23.04883 + 0.8002887 X_1 + 0.39828 X_2$$

Compare this to OLS regression

```
lmfit = lm( y ~ x1 + x2)
> lmfit = lm( y ~ x1 + x2)
> lmfit$coef
(Intercept)          x1          x2
-50.3588401   0.6711544   1.2953514
```

Reiteration

same decomposition as before

$$\begin{aligned} data &= fit_1 + residual_1 \\ residual_1 &= fit_2 + residual_2 \\ residual_2 &= fit_3 + residual_3 \end{aligned}$$

with final decomposition

$$data = fit_1 + fit_2 + fit_3 + residual_3$$

Note: order matters, unless x_1 is uncorrelated with x_2 (but usually not by much)

Multiple Regression with More than 2 Variables

General procedure, more than 2 carriers

1. Fit y to x_1: $y = c_{10} + c_1 x_1 + y_{\cdot 1}$
2. Fit x_2 to x_1: $x_2 = d_{20} + d_{21} x_1 + x_{2\cdot 1}$
3. Fit $y_{\cdot 1}$ to $x_{2\cdot 1}$: $y_{\cdot 1} = c_{20} + c_2 x_{2\cdot 1} + y_{\cdot 12}$
4. Fit x_3 to x_1 and $x_{2\cdot 1}$: $x_3 = d_{30} + d_{31} x_1 + d_{32} x_{2\cdot 1} + x_{3\cdot 12}$
5. Fit $y_{\cdot 12}$ to $x_{3\cdot 12}$: $y_{\cdot 12} = c_{30} + c_3 x_{3\cdot 12} + y_{\cdot 123}$
6. Fit x_4 to x_1, $x_{2\cdot 1}$, $x_{3\cdot 12}$:
   $x_4 = d_{40} + d_{41} x_1 + d_{42} x_{2\cdot 1} + d_{43} x_{3\cdot 12} + x_{4\cdot 123}$
7. Fit $y_{\cdot 123}$ to $x_{4\cdot 123}$: $y_{\cdot 123} = c_{40} + c_4 x_{4\cdot 123} + y_{\cdot 1234}$
8. Fit x_5 to $x_{2\cdot 1}$, $x_{3\cdot \text{`}12}$, $x_{4\cdot 123}$: (etc)

Assemble coefficients (intercepts, x_1, x_2, ...)

Multiple Regression

Strategies for order in choosing variables:

1. Plot y vs x_k in turn: choose that x_k for which linear relationship looks strongest. Note: some plots may suggest re-expressing x_k, e.g., 1/income, log(rate)
2. Regress (rank of y_i) on (rank of x_{ik}) and choose that x_k for which slope coefficient is largest. (use of ranks reduces impact of outliers on LS regression, but there could be many ties for "largest" if we have only a few data points and several variables have monotonic relationships with y).

Multiple Regression

Some important facts to remember:

1. The fit depends on what other variables are in the equation. Omission of a variable can make drastic changes in fit-even reverse sign of coefficients!
2. The LS coefficient b_k can be obtained by regressing $y_{\cdot(\text{all but } k)}$ on $x_{k\cdot(\text{all but } k)}$, so a plot of $y_{\cdot(\text{all but } k)}$ on $x_{k\cdot(\text{all but } k)}$ can be more informative than a plot of y on x_k, especially when considering several correlated carriers.

Proxy Variables

Illustration:

1. Fit $y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3$
   y: lung cancer incidence in county;
   x_1: # coffee houses;
   x_2: # fitness clubs;
   x_3: age
2. x_1 is very highly correlated with x_1^* (tobacco sales)
   $x_1 \approx d_0 + d_1 x_1^*$
3. The fit would be almost as good with $b_1 x_1$ as with $b_1 d_1 x_1^*$ (in terms of "percent of variation explained" $= R^2$)
4. We say that x_1 is a proxy for x_1^*

Proxy Variables

Now suppose that

- x_1^* is highly related to y (tobacco & lung cancer)
- x_1 is not related to y (coffee & lung cancer ?)
- But x_1 is related to x_1^* (coffee prompts smoking)

Consequence:

x_1 (coffee) looks relevant to the fit for y—but the relevance should be with x_1^* (tobacco sales), not with x_1 (coffee houses)

Proxy Variables

Proxies can be especially misleading if both are in the fit. Suppose $x_1 \equiv x_1^*$

1. Fit
   $y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 \Rightarrow y = b_0 + b_1 x_1 + b_2 x_1^* + b_3 x_3$
2. Substitute $x_1 \approx (d_0 + d_1 x_1^*$:

   $y = b_0 + b_1(d_0 + d_1 x_1^*) + b_2 x_1^* + b_3 x_3 = (b_0 + b_1 d_0) + (b_1 d_1 + b_2) x_1^* + b_3 x_3$
3. Likewise, $x_1^* = (x_1 - d_0)/d_1 = x_1/d_1 - d_0/d_1$

   $$y = (b_0 - d_0 b_2/d_1) + (b_1 + b_2/d_1) x_1 + b_3 x_3$$
4. Implications: b_1 could be big and b_2/d_1 small, or b_1 small and b_2/d_1 big, or half-and-half, or...
5. If $b_1 + b_2/d_1 = 10$, then $b_1 = 1$, $b_2/d_1 = 9$, or $b_1 = 20$, $b_2/d_1 = -10$, or $b_1 = 5$, $b_2/d_1 = 5$, or .....

Proxy Variables

With more variables, more potential for correlation:

$$x_1 = u + v + (small)_1$$

$$x_2 = v + w + (small)_2$$

$$x_3 = w - u + (small)_3$$

correlation b/w $(x_2 - x_1)$ and x_3 is very high!
Standard errors:

- Suppose $SE(b_1) = SE(b_2) = 5$, $cor(b_1, b_2) \approx 0.9$
- Then $SE(b_1 + b_2) \approx 2.3$, i.e., the sum of the coefficients is far better determined than a single coefficient.

Iteratively Reweighted LS

Robust regression: "iteratively re-weighted LS"

$$y = x\beta + \epsilon = \mathit{fit} + \mathit{rough}$$

$$\hat{\beta} = (X'X)^{-1}X'y, \quad \text{linear function of } y$$

weighted LS: $\hat{\beta}_W = (X'WX)^{-1}X'Wy$, linear function of y, where weight matrix $W = \mathrm{diag}(w_1, \cdots, w_n)$.
w_i = Big if residual is small (point i is "consistent" with model")
w_i = Small if residual is large (point i is "inconsistent" with model)
S_r= measure of spread in residuals
Big residual if $|resid| > 6S_r$

$$(*) \quad w_i = \begin{cases} [1 - (resid/(6S_r)^2]^2, & \text{if } |resid| \le 6S_r \\ 0, & \text{else} \end{cases}$$

Iteratively Reweighted LS

Iteratively re-weighted LS

1. Fit $OLS \rightarrow \hat{\beta}^{(0)} \rightarrow r^{(0)} = y - X\hat{\beta}^{(0)}$
2. Calculate $W^{(1)}$ using $r_i^{(0)}$ (weights from (*))
3. Fit $\hat{\beta}^{(1)}$ vis WLS $\rightarrow \hat{\beta}^{(1)} \rightarrow r^{(1)} = y - X\hat{\beta}^{(1)}$
4. Calculate $W^{(2)}$ using $r_i^{(1)}$ (weights from (*))
5. Fit $\hat{\beta}^{(2)}$ via WLS $\rightarrow \hat{\beta}^{(2)} \rightarrow r^{(2)} = y - X\hat{\beta}^{(2)}$
6. Until $|\hat{\beta}^{(k+1)} - \hat{\beta}^{(k)}| < tolerance$

Iteratively Reweighted LS

Notes on robust regression:

1. Starting with OLS is not robust
2. Eqn (*) is called bisque weight function
3. The "6" is a "tuning parameter" often b/w 4–6
4. Any weight function that falls smoothly to 0 yields robust estimate of β. Biweight regression is especially efficient
5. Easy to do computationally. See library (MAS5), help (flu).