STAT 2131
Orly Olbum

FINAL EXAM

Problem 1 $Y = (y_1, \dots, y_n)^T \in \mathbb{R}^n$, $x_i \in \mathbb{R}^p$, $X = (x_1, \dots, x_n)^T \in \mathbb{R}^{n \times p}$ → full rank

Suggested model: $y_i = x_i^T \beta + \epsilon_i$, but some ϵ_i take extreme, outlying values.

(a) Should we use **OLS**?

Because we don't yet know what type of extreme values these errors are, they may influence estimation of β when using OLS. I would recommend a different method to reduce the impact of these outlaying values such as leave one out (press (to find influence), bootstrap, or kernel (among others).

(b) Proposing:

$$\mathcal{L}_\lambda(\beta, \delta) = \frac{1}{2} \|Y - X\beta - \delta\|_2^2 + \lambda \|\delta\|_1$$

$$\{\hat{\beta}^{(\lambda)}, \hat{\delta}^{(\lambda)}\} = \underset{\beta \in \mathbb{R}^p, \delta \in \mathbb{R}^n}{\operatorname{argmin}} \mathcal{L}_\lambda(\beta, \delta)$$

(penalizing δ, not β)

(i) What is $\lim_{\lambda \to \infty} \hat{\beta}^{(\lambda)}$?

$$\underset{\beta \in \mathbb{R}^p, \delta \in \mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2} \|Y - X\beta - \delta\|_2^2 + \lambda \|\delta\|_1$$

Rewrite $\|Y - X\beta - \delta\|_2^2 = (Y - X\beta - \delta)^T (Y - X\beta - \delta)$

$$= Y^TY - Y^TX\beta - Y^T\delta - (X\beta)^TY + (X\beta)^TX\beta + (X\beta)^T\delta - \delta^TY + \delta^TX\beta + \delta^T\delta$$

$$= (Y - X\beta)^T(Y - X\beta) + (Y - \delta)^T(Y - \delta) + (X\beta)^TX\beta + \delta^T\delta$$

$$\underset{\beta, \delta}{\operatorname{argmin}} \underbrace{\frac{1}{2}\|Y - X\beta\|_2^2}_{\text{min at OLS, } \hat{\beta}^{OLS} = (X^TX)^{-1}X^TY} + \underbrace{\frac{1}{2}\|Y - \delta\|_2^2}_{\text{min at } \delta = Y} + \underbrace{\frac{1}{2}\|X\beta\|_2^2}_{\text{min at } \beta = 0} + \underbrace{\frac{1}{2}\|\delta\|_2^2}_{\text{min at } \delta = 0} + \underbrace{\lambda\|\delta\|_1}_{\text{min at } \delta = 0}$$

So, $\lim_{\lambda \to \infty} \hat{\beta}^{(\lambda)} \mathcal{L}_\lambda(\beta, \delta) = \lim_{\lambda \to \infty} [(X^TX)^{-1}X^T + 1_n] Y$

$$= \hat{\beta}^{OLS} = (X^TX)^{-1}X^TY$$

→

(ii) Is $\hat{\beta}^{(0)}$ unique?

$$\hat{\beta}^{(0)} = \underset{\beta,\delta}{\operatorname{argmin}} \ \frac{1}{2}\|Y - X\beta - \delta\|_2^2$$

Since X is full rank, $\hat{\beta}^{(0)}$ (solution to above) is unique because the solution is strictly convex. We will show that in the next problem. If X was not full rank, there would be a number of minimizers that would solve $\hat{\beta}^{(0)}$, which would make it not unique, but only one does the job now.

(iii) Show that $\mathcal{L}_\lambda$ is convex in $x = (\beta^T, \delta^T)^T \in \mathbb{R}^{p+n}$

($f(x)$ is convex if $f(\alpha x_1 + (1-\alpha)x_2) \leq \alpha f(x_1) + (1-\alpha) f(x_2)$ $\alpha \in [0,1]$)

$$\mathcal{L}_\lambda = \frac{1}{2}\|Y - X\beta - \delta\|_2^2 + \lambda\|\delta\|_1$$

$$= \frac{1}{2}\left[\|Y - X\beta\|_2^2 + \|Y - \delta\|_2^2 + \|X\beta\|_2^2 + \|\delta\|_2^2\right] + \lambda\|\delta\|_1$$

$$= \frac{1}{2}\left[(Y - X\beta)^T(Y - X\beta) + (Y-\delta)^T(Y-\delta) + (X\beta)^T(X\beta) + \delta^T\delta\right] + \lambda\|\delta\|_1$$

If $X_1 = \beta$ and $X_2 = \delta$, show that

$$f(\alpha\beta + (1-\alpha)\delta) \leq \alpha f(\beta) + (1-\alpha) f(\delta) \quad \text{for } \forall \alpha \in [0,1]$$

$$f(\alpha\beta + (1-\alpha)\delta) = \frac{1}{2}\left[\|Y - X\beta\|_2^2 + \|X\beta\|_2^2\right] + \frac{1}{2}\left[\|Y-\delta\|_2^2 + \|\delta\|_2^2\right] + \lambda\|\delta\|_1$$

$$= \left\{\frac{1}{2}\|Y - X\beta\|_2^2 + \frac{1}{2}\|X\beta\|_2^2\right\} + \left\{\frac{1}{2}\|Y-\delta\|_2^2 + \frac{1}{2}\|\delta\|_2^2 + \lambda\|\delta\|_1\right\}$$

$$= \frac{1}{2}\left\{\|Y - X\beta\|_2^2 + \|X\beta\|_2^2\right\} + \frac{1}{2}\left\{\|Y-\delta\|_2^2 + \|\delta\|_2^2 + 2\lambda\|\delta\|_1\right\}$$

$$= \frac{1}{2}\left\{Y^TY - Y^TX\beta - (X\beta)^TY + 2(X\beta)^TX\beta\right\} + \frac{1}{2}\left\{Y^TY - Y^T\delta - \delta^TY + 2\delta^T\delta + 2\lambda\|\delta\|_1\right\}$$

If $\alpha = \frac{1}{2}$, we have shown that $\mathcal{L}_\lambda$ is convex because we know the OLS solution is convex, and $\lambda\|\delta\|_1$ is strictly positive, $\lambda > 0$

(iv) The inclusion of $\delta = (\delta_1, \dots, \delta_n)^T$ in the loss function in this problem are there to penalize the large errors rather than what Lasso includes, which penalizes to many terms being present. By having $\{Y - X\beta - \delta\}$, as $\delta \to \infty$, large errors are replaced by a controlled value that is also penalized with $\{\lambda \|\delta\|_1\}$, the way coefficients too large are penalized by λ in lasso.

© Show that $\hat{\beta}^{(\lambda)}, \hat{\delta}^{(\lambda)}$ must satisfy

$$\hat{\beta}^{(\lambda)} = (X^TX)^{-1}X^T(Y - \hat{\delta}^{(\lambda)})$$

$$\hat{\delta}_i^{(\lambda)} = S_\lambda(y_i - x_i^T\hat{\beta}^{(\lambda)}), \quad i \in \{1, \dots, n\}$$

$S_\lambda(x) = \text{sign}(x)(|x| - \lambda)_+$ is soft-thresholding function.

$$S_\lambda(y_i - x_i^T\hat{\beta}^{(\lambda)}) = \text{sign}(y_i - x_i^T\hat{\beta}^{(\lambda)})(|y_i - x_i^T\hat{\beta}^{(\lambda)}| - \lambda)_+, \quad i \in \{1, \dots, n\}$$

$\hat{\beta}^{(\lambda)}$ minimizes $L_\lambda = \frac{1}{2}\|Y - X\beta - \delta\|_2^2 + \lambda\|\delta\|_1$

$$\to \tfrac{1}{2}[Y^TY - Y^TX\beta - Y^T\delta - (X\beta)^TY + \beta^TX^TX\beta + (X\beta)^T\delta - \delta^TY + \delta^TX\beta + \delta^T\delta] + \lambda\|\delta\|_1$$

$$= \tfrac{1}{2}[Y^TY - (Y^TX\beta + \beta^TX^TY) - Y^T\delta + \beta^T\beta + \beta^TX^T\delta - \delta^TY + \delta^TX\beta + \delta^T\delta] + \lambda\|\delta\|_1$$

$$= \tfrac{1}{2}[Y^TY - 2X^TY\beta - (Y^T\delta + \delta^TY) + \beta^T\beta + 2X^T\delta\beta + \delta^T\delta] + \lambda\|\delta\|_1$$

$$= \tfrac{1}{2}Y^TY + \sum_{i=1}^{n}\{\tfrac{1}{2}\beta_i^2 + \tfrac{1}{2}\delta_i^2 + \lambda|\delta_i| - X^TY\beta_i + X^T\delta_i\beta_i\}$$

minimize $\{\to\}$, and $-X^T\beta_i(Y - \delta_i)$ extrapolates on the lasso solution to say that

$$\hat{\delta}_i^{(\lambda)} = S_\lambda(y_i - x_i^T\hat{\beta}^{(\lambda)})$$

where $\hat{\beta}^{(\lambda)} = (X^TX)^{-1}X^T(Y - \hat{\delta}^{(\lambda)})$ from above, minimized at every i separately.

(d) Use (c) to derive iterative algorithm to determine $\hat{\beta}^{(n)}$, prove that the sequence is st. $\mathcal{L}_\lambda$ is non-increasing at each iteration.

If we start with $\hat{\beta}^{(0)}$ in (c), we have OLS estimator
- then we weight based on the soft-thresholding function and solve for the new estimate for β
- Move to $\hat{\beta}^{(1)}$ and re-calculate weights to find updated estimator

By nature of $\nabla\mathcal{L}$ => (only β segments)
$$\Rightarrow \sum\{\beta_i - x^T y + x^T \delta_i\} \Rightarrow \sum_i \{\beta_i - x^T(y - \delta_i)\}$$

$\nabla\mathcal{L}$ remains stationary or decreasing as estimates for β in each iteration converge to $\hat{\beta}^{(\lambda)}$ by $\delta_i^{(n)} = S_\lambda(y_i - x_i^T \hat{\beta}^{(n)})$

(e) If $i = 1, \ldots, n$ were outliers, they would remove them. Based on (c), is $\mathcal{L}_\lambda$ congruent with this?

Based on (c), $\mathcal{L}_\lambda$ is not congruent with this claim. While this particular loss function includes δ to "smooth" (sort of) outliers, it does not get rid of them. If we removed outliers, other points may then appear to be outliers and this would not be correct in the context of the initial data. From what we learned about lasso and Huber's loss, it does not seem appropriate to remove outliers if the loss function is taking care of them. Not only that, but (d) shows that $\mathcal{L}_\lambda$ is non-increasing, which really means we should keep the outliers in.

(f) Let

$$\rho_\lambda(x) = \begin{cases} \frac{1}{2}x^2 & , |x| \le \lambda \\ \lambda|x| - \frac{1}{2}\lambda^2 & , |x| > \lambda \end{cases}$$

be Huber's loss. Show that $\hat{\beta}^{(\lambda)}$ is a minimizer of $\sum_i \rho_\lambda(y_i - x_i^T\beta)$. Given (e), what does this imply about robustness of Huber's loss with outliers? Should it be used if the goal is to eliminate impact of outliers on β estimate?

$$\begin{cases} \hat{\beta}^{(\lambda)} = (X^TX)^{-1}X^T(Y - \hat{\delta}^{(\lambda)}) \\ \hat{\delta}_i^{(\lambda)} = S_\lambda(y_i - x_i^T\hat{\beta}^{(\lambda)}), i \in \{1,\dots,3\} \end{cases}$$

Minimizer: $\frac{\partial \rho_\lambda(x)}{\partial x}$

$|x| \le \lambda \rightarrow \frac{\partial \rho_\lambda(x)}{\partial x} = x$

$|x| > \lambda \rightarrow \frac{\partial \rho_\lambda(x)}{\partial x} = \text{sign}(x) \cdot \lambda$

Minimize $\sum_i \rho_\lambda(y_i - x_i^T\beta)$

$$\rho_\lambda = \begin{cases} \frac{1}{2}(y_i - x_i^T\beta)^2 & , |y_i - x_i^T\beta| \le \lambda \\ \lambda|y_i - x_i^T\beta| - \frac{1}{2}\lambda^2 & , |y_i - x_i^T\beta| > \lambda \end{cases}$$

$$\frac{\partial}{\partial\beta}\rho_\lambda = \begin{cases} -2x_i^T(y_i - x_i^T\beta) & , |y_i - x_i^T\beta| \le \lambda \\ \text{sign}(x_i^T\beta)|y_i - x_i^T\beta| - \lambda & , |y_i - x_i^T\beta| > \lambda \end{cases}$$

which brings us to the equations from (c), and from (b) we know $\hat{\beta}^{(\lambda)}$ minimizes ρ_λ.

From (e), we can conclude that Huber's loss is robust to outliers, and should not be used if we simply want to eliminate outliers to estimate β. Being robust means Huber's loss essentially has a predetermined method for dealing with outliers, and will produce a "good" estimate for β with them kept in. →

(9) General outlier-corrected estimate

$$\{\tilde{\beta}^{(\lambda)}, \tilde{\delta}^{(\lambda)}\} = \underset{\beta\in\mathbb{R}^p, \delta\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2}\|Y - X\beta - \delta\|_2^2 + \sum_{i=1}^n P_\lambda(\delta_i) \right\}$$

where $P_\lambda(x)$ penalizes large x and $\lambda \geq 0$ is a tuning parameter.

Threshhold rule

$$S_{\lambda,P}(x) = \underset{u\in\mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{2}(x-u)^2 + P_\lambda(u) \right\}$$

(i) If $P_\lambda(x) = \frac{\lambda^2}{2} 1\{x \neq 0\}$, show that $S_{\lambda,P}$ is the hard-thresholding function ($S_{\lambda,P}(x) = x1\{|x| > \lambda\}$).

If $P_\lambda(x) = \frac{\lambda^2}{2}$, $S_{\lambda,P}(x) = \underset{u\in\mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{2}(x-u)^2 + \frac{\lambda^2}{2} \right\}$, $x \neq 0$

The min of $\left\{ \frac{1}{2}(x-u)^2 + \frac{\lambda^2}{2} \right\}$ occurs at

$(x-u) = 0$, or $x = u$,

making $S_{\lambda,P}(x) = x$, given $|x| > \lambda$

which takes the form of hard-thresholding function.

(ii) Show that like $\hat{\beta}^{(\lambda)}$ & $\hat{\delta}^{(\lambda)}$, $\tilde{\beta}^{(\lambda)}$ & $\tilde{\delta}^{(\lambda)}$ must satisfy

$$\tilde{\beta}^{(\lambda)} = (X^TX)^{-1}X^T(Y - \tilde{\delta}^{(\lambda)})$$

$$\tilde{\delta}_i^{(\lambda)} = S_{\lambda,P}(y_i - x_i^T\tilde{\beta}^{(\lambda)}), \quad i \in \{1, \ldots, n\}$$

If $\tilde{\delta}_i^{(\lambda)} = S_{\lambda,P}(y_i - x_i^T\tilde{\beta}^{(\lambda)}) = \underset{u\in\mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{2}(y_i - x_i^T\beta - u)^2 + P_\lambda(u) \right\}$

$S_{\lambda,P}$ has essentially turned into the outlier-corrected estimate, minimized by $\tilde{\beta}^{(\lambda)}$

$$\rightarrow \underset{u\in\mathbb{R}}{\operatorname{argmin}} \left\{ \frac{1}{2}(y_i - x_i(x_i^Tx_i)^{-1}x_i^T(y_i - \tilde{\delta}_i^{(\lambda)}) - u)^2 + P_\lambda(u) \right\}$$

$$\frac{1}{2}(y_i - x_i^T\beta - u)^2 = \frac{1}{2}[y_i^2 - y_ix_i^T\beta - y_iu + x_i^T\beta y_i + (x_i^T\beta)^2 + x_i^T\beta u - uy_i + ux_i^T\beta + u^2]$$

$$= \frac{1}{2}[y_i^2 + u^2 - 2uy_i + 2ux_i^T\beta + (x_i^T\beta)^2]$$

derive wrt $u \rightarrow \frac{1}{2}(2u - 2y_i + 2x_i^T\beta) = -(y_i - x_i^T\beta - u)$

which has a solution of $\beta = \tilde{\beta}^{(\lambda)}$

(iii) $\rho_\lambda(x)$ is any continuous differentiable and robust loss function (like Huber).

$\psi_\lambda(x) = \frac{d}{dx}\rho_\lambda(x)$

Show that if $\psi_\lambda(x) + S_{\lambda,\rho}(x) = x$, $\tilde{\beta}^{(\lambda)}$ is a stable point of $\sum_i \rho_\lambda(y_i - x_i^T\beta) \rightarrow \sum_i \rho(\epsilon_i)$

That is, $\frac{d}{d\beta}\{\sum_i \rho_\lambda(y_i - x_i^T\beta)\}|_{\beta=\tilde{\beta}^{(\lambda)}} = 0_p$

$$\frac{\partial}{\partial\beta}\sum_i \rho_\lambda(y_i - x_i^T\beta) = \begin{cases} -2x_i^T(y_i - x_i^T\beta) & , |y_i - x_i^T\beta| \leq \lambda \\ \text{sign}(x_i^T\beta)\,|y_i - x_i^T\beta| - \lambda & , |y_i - x_i^T\beta| > \lambda \end{cases}$$

$\psi_\lambda(x) \Rightarrow \sum_i (y_i - x_i^T b)x_i = 0_p$

expansion: $\sum_i \epsilon_i x_i + \sum_i -\dot{\psi}(y_i - x_i^T\beta)\, x_i x_i^T H(\beta)(\tilde{\beta}^{(\lambda)} - \beta)$

where $\tilde{\beta}^{(\lambda)}$ makes $\psi_\lambda(x) = 0_p$,

and $\psi_\lambda(x) + S_{\lambda,\rho}(x) = x$

because $S_{\lambda,\rho}(x) = x$ for minimum u (see (i))

(iv) Use (iii) to suggest a <u>continuous</u> shrinkage function $S_{\lambda,\rho}$ that would satisfy the goal of removing data points with potentially extreme ϵ_i values.

Instead of penalizing extreme ϵ_i by having some function of them, can we have a constant (flat line)?

Something along the lines of...

$$\rho(x) = \begin{cases} \frac{1}{2}x^2 & , |x| \leq \lambda \\ \text{sgn}(x)\cdot\lambda & , |x| > \lambda \end{cases}$$

This essentially removes data points past the designated "extreme"

→

(9) (iv) continued...

we could additionally suggest some weights that begin with the OLS solution for β where

$$\psi(x) = \rho'(x) = \begin{cases} x, & |x| \leq \lambda \\ 0, & |x| > \lambda \end{cases}$$

and weights are

$$w_i = \psi(x)/x = \begin{cases} 1, & |x| \leq \lambda \\ 0, & |x| > \lambda \end{cases}$$

That serves to keep values within $|\lambda|$ (chosen how?) and eliminate extreme values of $\epsilon_i = y_i - x_i^T \beta$.

Problem 2

An airfoil is the cross-sectional shape of a wing or propeller, and is the object that helps generate lift (the force that allows airplanes to fly, for example). Here you will use the data “airfoil.dat” to model “pressure” (the sound pressure level, in decibels) as a function of “frequency”, “angle”, “chordLength”, “velocity”, and “thickness”.

(a) Use ordinary least squares to regress pressure onto all of the other variables.

```
##
## Call:
## lm(formula = pressure ~ ., data = air)
##
## Coefficients:
## (Intercept)    frequency        angle  chordLength     velocity    thickness
##   1.328e+02   -1.282e-03   -4.219e-01   -3.569e+01    9.985e-02   -1.473e+02
```

(i) Write down the assumed mathematical model for pressure, define all coefficients in your model, and clearly list all assumptions.

The model:

pressure = 132.8 - 0.001(frequency) - 0.422(angle) - 35.69(chordLength) + 0.099(velocity) - 147.3(thickness)

(plus some error)

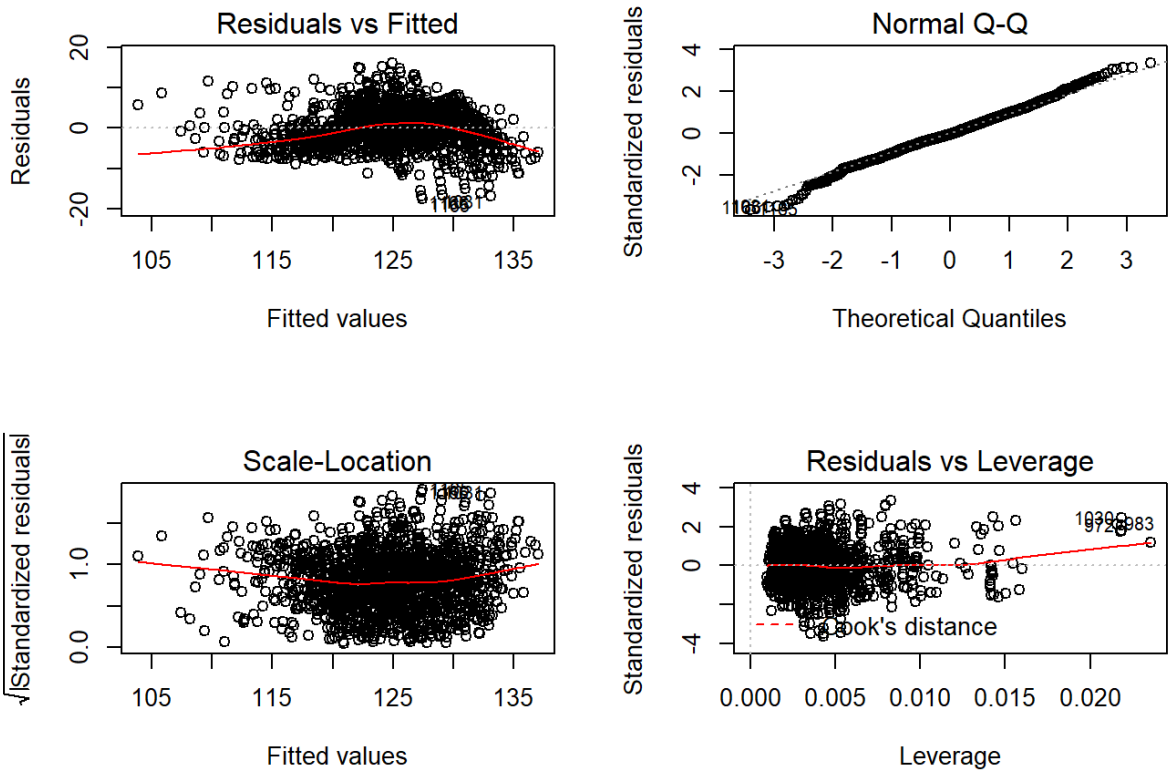
Coefficients:

- the negative coefficient for frequency, angle, chordLength, and thickness suggest that all else constant, a unit increase in any of these covariates result in a decrease in pressure
- the positive coefficient for velocity suggests that all else constant, a unit increase in velocity will result in an increase in pressure

OLS Assumptions:

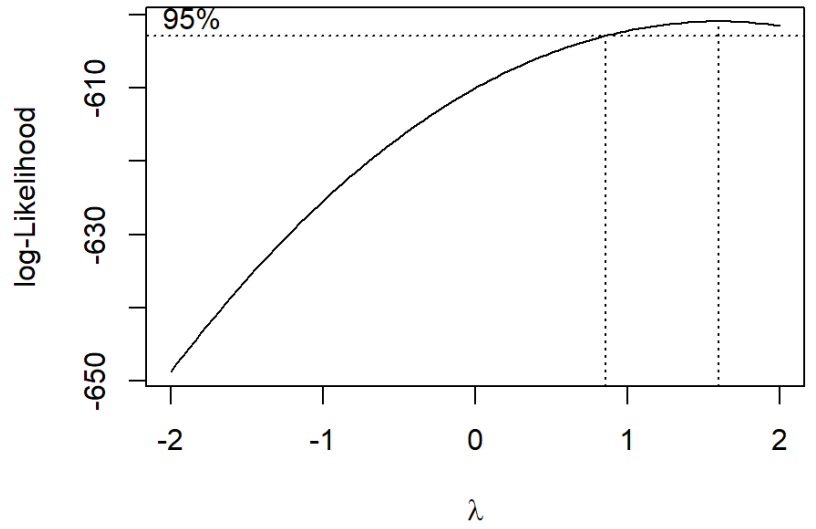
1. Data is linear (here, pressure is assumed to be linear in each covariate)
2. Residuals are normal (errors are normally distributed)
3. Residual variance is constant
4. Residuals are independent (if they are not, the x’s may not really be explaining y)

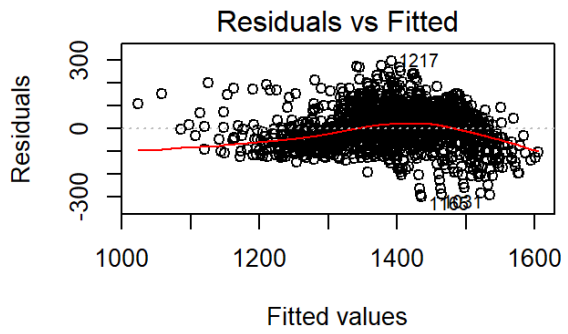
(ii) In order to satisfy modelling assumptions, decide whether the dependent variable requires a transformation.

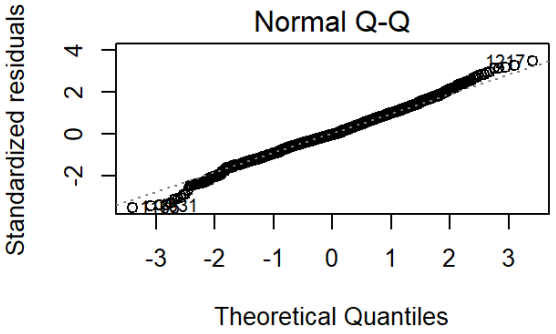


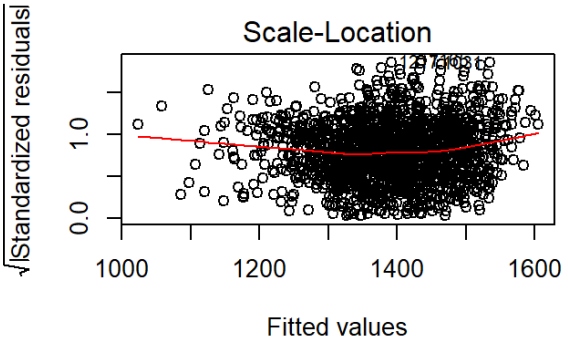
We can see in the plots that we do need a transformation. The first plot shows a curve rather than the desired horizontal line, and the fourth plot shows some influential points in relation to Cook's distance. We might try box-cox.

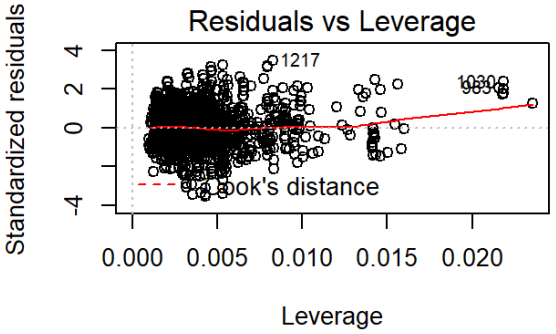
(iii) Fit the new model, and determine if this new model satisfies your assumptions from part (i).











After fitting a boxcox transformation model, we still have residuals with significant leverage, which does not satisfy our assumptions.

(iv) A colleague suggests that because pressure is not normally distributed, the bootstrap is a more appropriate way to do inference in these data. Design a bootstrap procedure to determine a 90% confidence interval for the expected transformed pressure variable at frequency, angle, chordLength, velocity and thickness values given in “Interval.txt”.

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 500 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, conf = 0.9, type = "basic")
##
## Intervals :
## Level      Basic
## 90%   (130.1, 130.9 )
## Calculations and Intervals on Original Scale
```

First, I created a dataframe with the provided values. Then, I wrote a function that takes in data, a formula, and indices and produces the prediction for pressure, transformed from its boxcox format. The boot function provides the inputs to find the confidence interval for pressure (re-transformed) at the given values of all other variables.

(v) How does your interval compare to the confidence interval obtained using standard normal theory? Are the similarities/differences between the two surprising? Explain.

Using the predicted pressure from the boxcox model (no bootstrapping), we obtain the following confidence interval. The first interval is the prediction interval from new data using the original model, and the second interval is the transformed model from boxcox (and transformed back - exp(log(p*lambda + 1/lambda))).

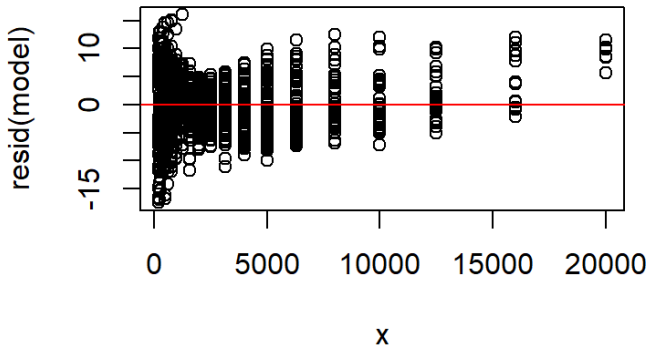
```
##      fit     lwr     upr
## 1 130.54 130.087 130.993
```

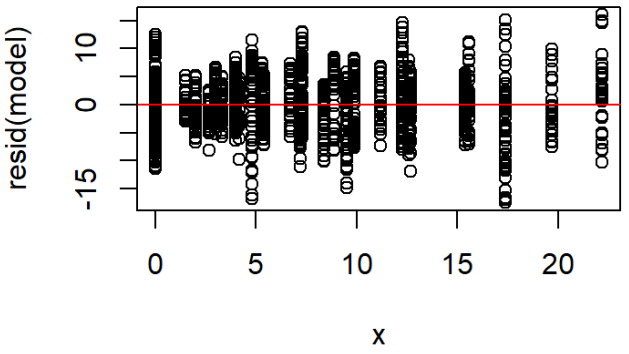
```
##       fit     lwr     upr
## 1 130.512 130.071 130.952
```

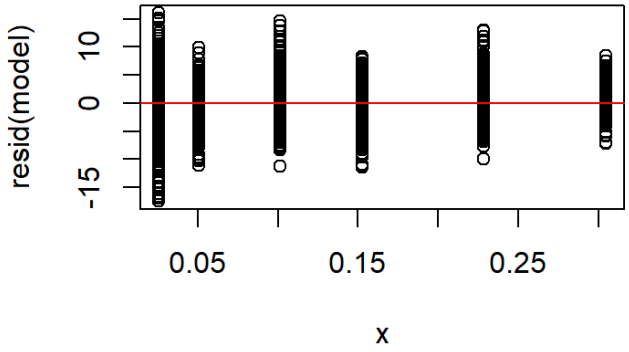
Because of what bootstrapping provides, we can rely on the Central Limit Theorem to satisfy assumptions without requiring normally distributed data - resampling approaches normality. Since bootstrapping by nature reassembles the data/approach to computing the confidence interval, it resembles a standard normal confidence interval, and they turn out to be extremely similar. This is not surprising. In bootstrap we are sampling with replacement and creating sub-samples - as we learned in class, the bootstrap is to the sample as the sample is the population. If we increase the number of bootstrap samples, we know by CLT that we approach normal distribution and we have essentially come full circle. The more bootstrap samples we take, the closer the bootstrap estimate will be to the sample estimate (the same way the proportion of heads when flipping a fair coin gets closer and closer to .5 the more times you flip it).

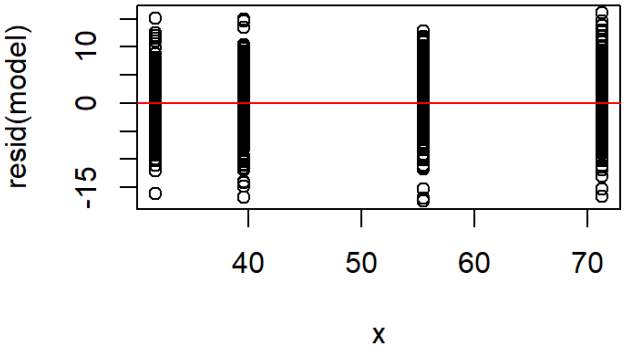
(b) Using the most appropriate model from (a), determine if the dependent variable is nonlinearly related to the five covariates. If so, do your best to fix these non-linearities in the context of classical linear modelling, and determine if the changes to your model from part (a) are significant.

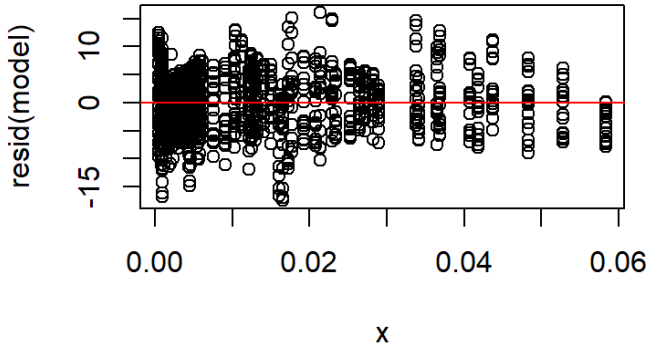
Based on model diagnostics from the two models above, the original OLS model has a better fit (by R sq and F-test), and we can use that one to assess whether further transformations are necessary. Using plot() for this model we can see how the residuals fit (or don’t), as well as the leverage points we discussed earlier. These are a big hint that we will need to transform the variables.



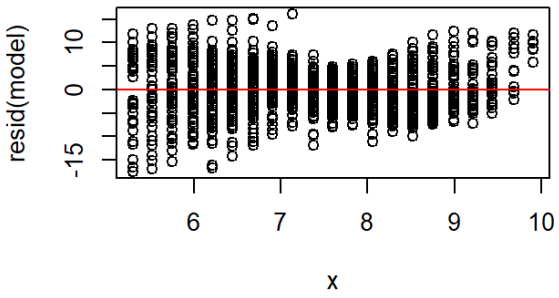


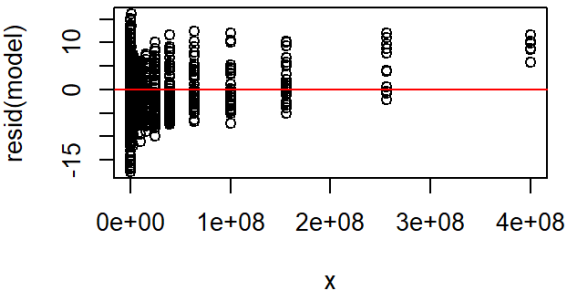


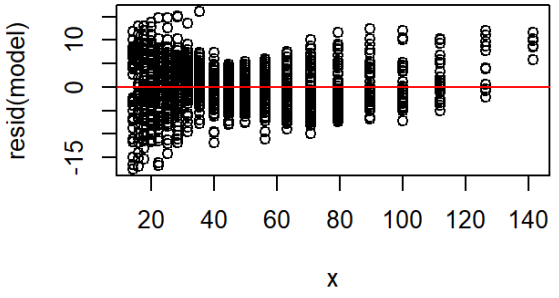


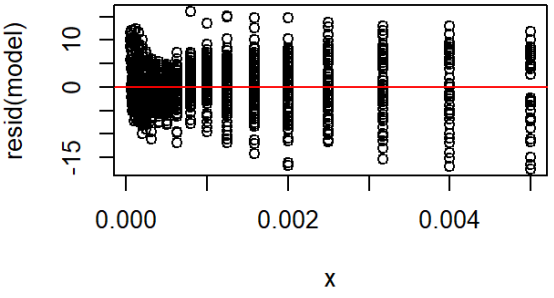


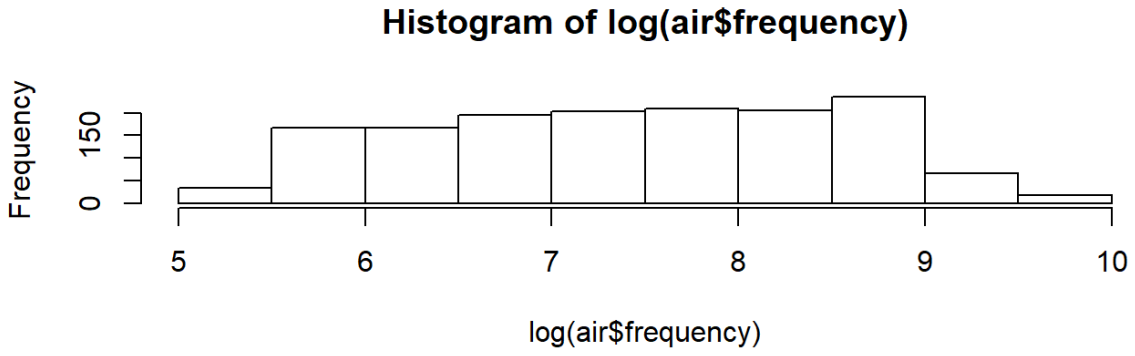
While most of the covariates seem ok, frequency appears to have a slightly increasing relationship with pressure, so we can try a couple transformations and reassess.

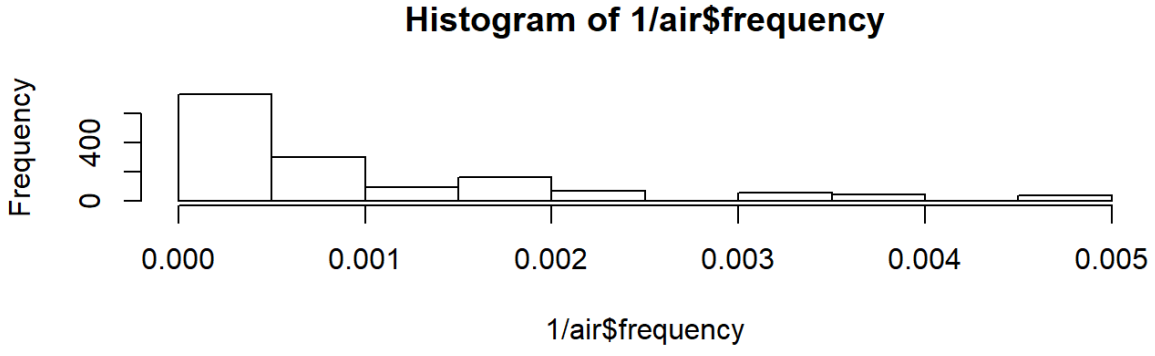












While the first and fourth transformations seem the most promising, it comes down to the logged data since the histogram is so close to normal. Additionally, while viewing the individual plots for each covariate, I saw that there are only 4 levels of velocity, and treating this as a factor variable may help the model as well. Angle has 6 levels - I'll try treating that as a factor variable too.

The first two lines refer to the model with logged frequency and factor(velocity), and the second two lines correspond to the model with logged frequency with factor(velocity) and factor(angle).

```
## [1] 0.483
```

```
##    value    numdf    dendf
##  199.754    7.000 1495.000
```

```
## [1] 0.519
```

```
##    value    numdf    dendf
##   49.484   32.000 1470.000
```

Even though it reduces the available degrees of freedom, it appears that treating both angle and velocity as factor variables helps the model (based on R sq value, and the F stat is still significant), and the logged frequency variable remains a significant predictor for pressure in this model.

(c) In your opinion, is the model from part (b) better or worse than that from part (a)? While we have not discussed many alternatives in class, do you think linear modelling is an appropriate way to analyze these data? Explain.

The second model does not appear to really be fixing the problem, which leads me to believe that a linear model might not be the best course of action for this dataset. For chordLength and velocity, there are only 6 and 4 levels respectively, and even when fitted as factor variables the model does not seem to improve. This process also examined a number of ways to address poor model fit, and they do not seem to work in the context of linear regression, which leads me to believe there is a different and potentially better way to fit this data.

R Code for Problem 2

```
library(MASS)
library(boot)

# a
model = lm(pressure ~ ., data = air)
model

# ii
par(mfrow = c(2, 2))
plot(model)

# iii
par(mfrow = c(1, 1))
bc = boxcox(model)
lambda = bc$x[which.max(bc$y)]
lambda
new_model = lm(((pressure^lambda-1)/lambda) ~ frequency + angle + chordLength + velocity +
thickness, data = air)

par(mfrow = c(2, 2))
plot(new_model)

# iv
new.data = data.frame(
  frequency = 1000,
  angle = 1,
  chordLength = 0.2,
  velocity = 70,
  thickness = 0.003
)

# function takes in formula, returns prediction for pressure based on new.data
pres_mean = function(formula, data, indices){
  d = data[indices, ]
  fit = lm(formula, data = d)
  return(exp(log(predict.lm(fit, newdata = new.data)*lambda + 1)/lambda))
}

results = boot(data = air, statistic = pres_mean, R = 500,
           formula = ((pressure^lambda-1)/lambda) ~ frequency + angle + chordLength + velocity +
thickness)

ci = boot.ci(results, conf = 0.90, type = "basic")
ci

# v
```

```
# interval from original model
predict.lm(model, newdata = new.data, interval = "confidence", level = 0.90)
# interval from transformed model, readjusted
p = predict.lm(new_model, newdata = new.data, interval = "confidence", level = 0.90)
exp(log(p*lambda + 1)/lambda)

# b
# function to plot each covariate against the residuals of the model
check = function(x){
  plot(x, resid(model))
  abline(h = 0, col = "red")
}

par(mfrow = c(1, 1))
check(air$frequency)
check(air$angle)
check(air$chordLength)
check(air$velocity)
check(air$thickness)

par(mfrow = c(2, 2))
check(log(air$frequency))
check(air$frequency^2)
check(sqrt(air$frequency))
check(1/air$frequency)

par(mfrow = c(2, 1))
hist(log(air$frequency))
hist(1/air$frequency)

model.fix = lm(pressure ~ log(frequency) + angle + chordLength + factor(velocity) + thickness, data = air)
model.fix2 = lm(pressure ~ log(frequency) + factor(angle) + chordLength + factor(velocity) + thickness,
            data = air)
summary(model.fix)$r.sq; summary(model.fix)$fstatistic
summary(model.fix2)$r.sq; summary(model.fix2)$fstatistic
```