Unit 3: Linear models: Misspecification

Eugene Katsevich

October 18, 2021

In our discussion of linear model inference in Unit 2, we assumed the normal linear model throughout:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_n). \tag{1}$$

In this unit, we will discuss what happens when this model is misspecified:

- Non-normality (Section 1): $\boldsymbol{\epsilon} \sim (0, \sigma^2 \boldsymbol{I}_n)$ but not $N(0, \sigma^2 \boldsymbol{I}_n)$.
- Heteroskedastic errors (Section 2): $\epsilon_i \overset{\text{ind}}{\sim} N(0, \sigma_i^2)$, where it is not the case that $\sigma_1^2 = \cdots = \sigma_n^2$.
- Correlated errors (Section 3): It is not the case that $(\epsilon_1, \ldots, \epsilon_n)$ are independent.
- Model bias (Section 4): It is not the case that $\mathbb{E}[\boldsymbol{y}] = \boldsymbol{X}\boldsymbol{\beta}$ for some $\boldsymbol{\beta} \in \mathbb{R}^p$.
- Outliers (Section 5): For one or more i, it is not the case that $y_i \sim N(\boldsymbol{x}_{i*}^T \boldsymbol{\beta}, \sigma^2)$.

For each type of misspecification, we will discuss its origins, consequences, detection, and fixes (Sections 1-5). We conclude with an R demo (Section 7).

1 Non-normality

1.1 Origin

Non-normality occurs when the distribution of $y|\boldsymbol{x}$ is either skewed or has heavier tails than the normal distribution. This may happen, for example, if there is some discreteness in y.

1.2 Consequences

Non-normality is the most benign of linear model misspecifications. While we derived linear model inferences under the normality assumption, all the corresponding statements hold asymptotically without this assumption. Recall Homework 2 Question 1, or take for example the simpler problem of estimating the mean μ of a distribution based on n samples from it: We can test $H_0 : \mu = 0$ and build a confidence interval for μ even if the underlying distribution is not normal. So if n is relatively large and p is relatively small, you need not worry too much. If n is small and the errors are highly skewed or heavy-tailed, there might be an issue.

1.3 Detection

Non-normality is a property of the error-terms ϵ_i. We do not observe these directly, but we can approximate these using the residuals

$$\widehat{\epsilon}_i = y_i - \boldsymbol{x}_{i*}^T \widehat{\boldsymbol{\beta}}. \tag{2}$$

Recall from Unit 2 that $\text{Var}[\widehat{\boldsymbol{\epsilon}}] = \sigma^2(\boldsymbol{I} - \boldsymbol{H})$. Letting h_i be the ith diagonal entry of $\boldsymbol{H}$, it follows that $\widehat{\epsilon}_i \sim (0, \sigma^2(1 - h_i))$. The *standardized residuals* are defined as

$$r_i = \frac{\widehat{\epsilon}_i}{\widehat{\sigma}\sqrt{1 - h_i}}. \tag{3}$$

Under normality, we would expect $r_i \dot{\sim} N(0, 1)$. We can therefore assess normality by producing a histogram or normal QQ-plot of these residuals (see Figure 1).

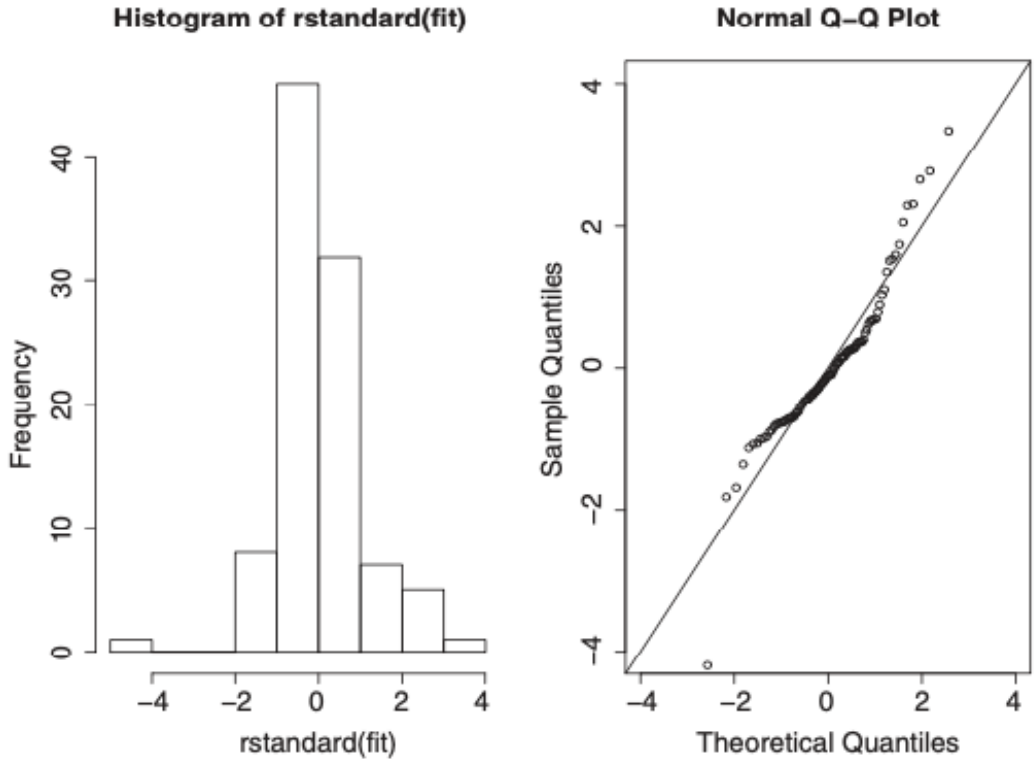


Figure 1: Histogram and normal QQ plot of standardized residuals.

1.4 Fixes

As mentioned in Section 1.2, non-normality is not necessarily a problem that needs to be fixed, except in small samples. In small samples, we can apply the bootstrap (Section 6.2.2) for robust standard error computation and a few different strategies (Section 6.3) for robust hypothesis testing.

2 Heteroskedastic errors

2.1 Origin

Suppose each observation y_i is actually the average of n_i underlying observations, each with variance σ^2. Then, the variance of y_i is σ^2/n_i, which will differ across i if n_i differ. It is also common to see the variance of a distribution increase as the mean increases (as in Figure 2), whereas for a linear model the variance of y stays constant as the mean of y varies.

2.2 Consequences

All normal linear model inference from Unit 2 hinges on the assumption that $\epsilon_i \overset{\text{i.i.d.}}{\sim} N(0, \sigma^2)$. The coverage of confidence intervals and the levels of hypothesis tests may depart from their nominal levels. This is easiest to see if we consider the width of confidence intervals for $\boldsymbol{x}_0^T\boldsymbol{\beta}$; see Figure 2 for intuition.

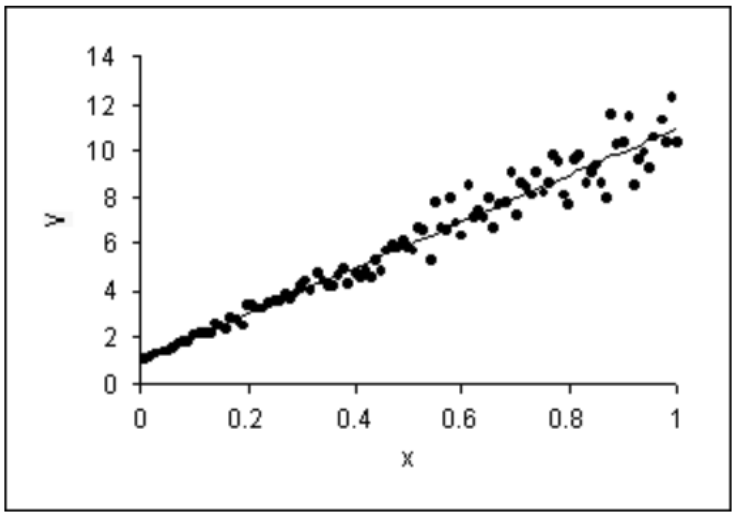


Figure 2: Heteroskedasticity in a simple bivariate linear model (image source).

2.3 Detection

Heteroskedasticity is usually assessed via the *residual plot* (Figure 3). In this plot, the standardized residuals r_i (3) are plotted against the fitted values $\widehat{\mu}_i$. In the absence of heteroskedasticity, the spread of the points around the origin should be roughly constant as a function of $\widehat{\mu}$ (Figure 3(a)). A common sign of heteroskedasticity is the fan shape where variance increases as a function of $\widehat{\mu}$ (Figure 3(c)).

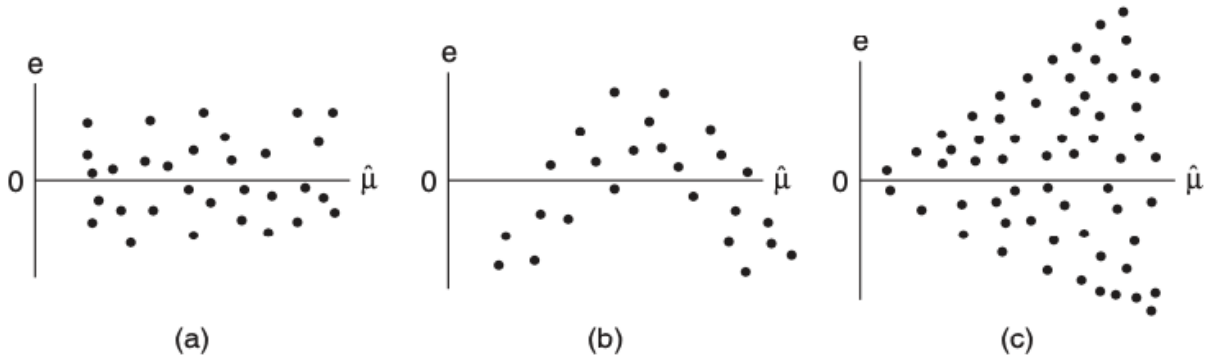


Figure 3: Residuals plotted against linear-model fitted values that reflect (a) model adequacy, (b) quadratic rather than linear relationship, and (c) nonconstant variance(image source: Agresti Figure 2.8).

2.4 Fixes

Heteroskedasticity-robust standard errors for hypothesis testing and confidence intervals can be obtained using a number of strategies, including the Huber-White sandwich estimator 6.2.1, the bootstrap 6.2.2, and permutation tests 6.3.1.

3 Correlated errors

3.1 Origin

Correlated errors can arise when observations have group, spatial, or temporal structure. Below are examples:

- Group/clustered structure: We have 10 samples $(\boldsymbol{x}_{i*}, y_i)$ each from 100 schools.
- Spatial structure: We have 100 soil samples from a 10×10 grid on a 1km×1km field.
- Temporal structure: We have 366 COVID positivity rate measurements, one from each day of the year 2020.

The issue arises because there are common sources of variation among sample that are in the same group or spatially/temporally close to one another.

3.2 Consequences

Like with heteroskedastic errors, correlated errors can cause invalid standard errors. In particular, positively correlated errors typically cause standard errors to be smaller than they should be, leading to inflated Type-I error rates. For intuition, consider estimating the mean of a distribution based on n samples. Consider the cases when these samples are independent, compared to when they are perfectly correlated. The effective sample size in the former case is n and in the latter case is 1.

3.3 Detection

Residual plots once again come in handy to detect correlated errors. Instead of plotting the standardized residuals against the fitted values, we should plot the residuals against whatever variables we think might explain variation in the response that the regression does not account for. In the presence of group structures, we can plot residuals versus group (via a boxplot); in the presence of spatial or temporal structure, we can plot residuals as a function of space or time. If the residuals show a dependency on these variables, this suggests they are correlated.

3.4 Fixes

There are a few approaches to addressing correlated errors:

1. Estimate the covariance matrix $\boldsymbol{\Sigma}$ of the observations, so that $\boldsymbol{y} \sim N(\boldsymbol{X}\boldsymbol{\beta}, \Sigma)$. This is a *generalized least squares* problem for which inference can be carried out. The generalized least squares estimate is $\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{y}$, which is distributed as $\widehat{\boldsymbol{\beta}} \sim N(\boldsymbol{\beta}, (\boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{X})^{-1})$. We can carry out inference based on the latter distributional result analogously to how we did so in Unit 2. A special case of this is the *linear mixed effects model*, which hopefully we will have time to discuss in Unit 6.
2. Use the Liang-Zeger variance estimator; see Section 6.2.1.
3. Apply a clustered or block bootstrap; see Section 6.2.2.

4 Model bias

4.1 Origin

Model bias arises when predictors are left out of the regression model:

$$\text{assumed model: } \boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}; \quad \text{actual model: } \boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}. \tag{4}$$

We may not always know about or measure all the variables that impact a response $\boldsymbol{y}$.

Model bias can also arise when the predictors do not impact the response on the linear scale. For example:

$$\text{assumed model: } \mathbb{E}[\boldsymbol{y}] = \boldsymbol{X}\boldsymbol{\beta}; \quad \text{actual model: } g(\mathbb{E}[\boldsymbol{y}]) = \boldsymbol{X}\boldsymbol{\beta}. \tag{5}$$

4.2 Consequences

In cases of model bias, the parameters $\boldsymbol{\beta}$ in the assumed linear model lose their meanings. The least squares estimate $\widehat{\boldsymbol{\beta}}$ will be a biased estimate for the parameter we probably actually want to estimate. In the case (4) when predictors are left out of the regression model, these additional predictors $\boldsymbol{Z}$ will act as confounders and create bias in $\widehat{\boldsymbol{\beta}}$ as an estimate of the $\boldsymbol{\beta}$ parameters in the true model, unless $\boldsymbol{X}^T\boldsymbol{Z} = 0$. As discussed in Unit 2, this can lead to misleading conclusions.

4.3 Detection

Similarly to the detection of correlated errors, we can try to identify model bias by plotting the standardized residuals against predictors that may have been left out of the model. A good place to start is to plot standardized residuals against the predictors $\boldsymbol{X}$ (one at a time) that are in the model, since nonlinear transformations of these might have been left out. In this case, you would see something like Figure 3(b).

It is possible to formally test for model bias in cases when we have repeated observations of the response for each value of the predictor vector. In particular, suppose that $\boldsymbol{x}_{i*} = \boldsymbol{x}_c$ for $c = c(i)$ and predictor vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C \in \mathbb{R}^p$. Then, consider testing the following hypothesis:

$$H_0 : y_i = \boldsymbol{x}_{i*}^T\boldsymbol{\beta} + \epsilon_i \quad \text{versus} \quad H_1 : y_i = \beta_{c(i)} + \epsilon_i. \tag{6}$$

The model under H_0 (the linear model) is nested in the model for H_1 (the saturated model), and we can test this hypothesis using an F-test called the *lack of fit F-test*.

4.4 Fixes

To fix model bias in the case (4), ideally we would identify the missing predictors $\boldsymbol{Z}$ and add them to the regression model. This may not always be feasible or possible. To fix model bias in the case (5), it is sometimes advocated to find a transformation g (e.g. a square root or a logarithm) of $\boldsymbol{y}$ such that $\mathbb{E}[g(\boldsymbol{y})] = \boldsymbol{X}\boldsymbol{\beta}$. However, a better solution is to use a *generalized linear model*, which we will discuss starting in Unit 4.

5 Outliers

5.1 Origin

Outliers often arise due to measurement or data entry errors. An observation can be an outlier in $\boldsymbol{x}$, in y, or both.

5.2 Consequences

An outlier can have the effect of biasing the estimate $\widehat{\boldsymbol{\beta}}$. This occurs when an observation has outlying $\boldsymbol{x}$ as well as outlying y.

5.3 Detection

There are a few measures associated to an observation that can be used to detect outliers, though none are perfect. The first quantity is called the *leverage*, defined as

$$\text{leverage of observation } i \equiv \text{corr}(y_i, \widehat{\mu}_i)^2. \tag{7}$$

This quantity measures the extent to which the fitted value $\widehat{\mu}_i$ is sensitive to the (noise in the) observation y_i. It can be derived that

$$\text{leverage of observation } i = h_{ii}, \tag{8}$$

which is the ith diagonal element of the hat matrix $\boldsymbol{H}$. This is related to the fact that $\text{Var}[\widehat{\epsilon}_i] = \sigma^2(1 - h_{ii})$. The larger the leverage, the smaller the variance of the residual, so the closer the line passes to the ith observation. The leverage of an observation is larger to the extent that $\boldsymbol{x}_{i*}$ is far from $\bar{\boldsymbol{x}}$. For example, in the bivariate linear model $y_i = \beta_0 + \beta_1 x_i + \epsilon_i$,

$$h_{ii} = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i'=1}^{n}(x_{i'} - \bar{x})^2}.$$

Note that the leverage is not a function of y_i, so a high-leverage point might or might not be an outlier in y_i and therefore might or might not have a strong impact on the regression. To assess more directly whether an observation is *influential*, we can compare the least squares fits with and without that observation. To this end, we define the *Cook's distance*

$$D_i = \frac{\sum_{i'=1}^{n}(\widehat{\mu}_{i'} - \widehat{\mu}_{i'}^{-i})^2}{p\widehat{\sigma}^2}, \tag{9}$$

where $\widehat{\mu}_{i'}^{-i} = \boldsymbol{x}_{i*}^T \widehat{\boldsymbol{\beta}}^{-i}$ and $\widehat{\boldsymbol{\beta}}^{-i}$ is the least squares estimate based on $(\boldsymbol{X}_{-i,*}, \boldsymbol{y}_{-i})$. An observation is considered influential if it has Cooks distance greater than one.

There is a connection between Cook's distance and leverage:

$$D_i = \left(\frac{y_i - \widehat{\mu}_i}{\widehat{\sigma}\sqrt{1 - h_{ii}}}\right)^2 \cdot \frac{h_{ii}}{p(1 - h_{ii})}. \tag{10}$$

We recognize the first term as the standardized residual; therefore a point is influential if its residual and leverage are large.

Note that Cook's distance may not successfully identify outliers. For example, if there are groups of outliers, then they will *mask* each other in the calculation of Cook's distance.

5.4 Fixes

If outliers can be detected, then the fix is to remove them from the regression. But, we need to be careful. Definitively determining whether observations are outliers can be tricky. Outlier detection can even be used as a way to commit fraud with data, as now-defunct blood testing start-up Theranos is alleged to have done.

As an alternative to removing outliers, we can fit estimators $\widehat{\boldsymbol{\beta}}$ that are less sensitive to outliers; see Section 6.1.

6 Robust inference

There are a number of strategies designed to address one or more of the misspecification issues listed above. These fall into the categories of robust estimation (to get better estimates of $\widehat{\boldsymbol{\beta}}$ in the presence of outliers; see Section 6.1), robust standard error computation (to get more reliable standard errors in the presence of heteroskedasticity or correlated errors; see Section 6.2), and robust hypothesis testing (to get more reliable hypothesis tests in the presence of heteroskedasticity, correlated errors, and sometimes even model bias; see Section 6.3).

6.1 Robust estimation

The squared error loss $\sum_{i=1}^n (y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta})^2$ is sensitive to outliers in the sense that a large value of $y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta}$ can have a significant impact on the loss function. The least squares estimate, as the minimizer of this loss function, is therefore sensitive to outliers. One way of addressing this challenge is to replace the squared error loss by a different loss that does not grow so quickly in $y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta}$. A popular choice for such a loss function is the Huber loss:

$$L_\delta(y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta}) = \begin{cases} \frac{1}{2}(y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta})^2, & \text{if } |y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta}| \le \delta; \\ \delta(|y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta}| - \delta), & \text{if } |y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta}| > \delta. \end{cases} \tag{11}$$

This function is differentiable, like the squared error loss, but grows linearly as opposed to quadratically. We can then define

$$\widehat{\boldsymbol{\beta}}^{\text{Huber}} \equiv \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^n L_\delta(y_i - \boldsymbol{x}_{i*}^T\boldsymbol{\beta}).$$

This is an *M-estimator*; it is consistent and has an asymptotic normal distribution that can be used for inference.

6.2 Robust standard error computation

When the error terms in a regression are not homoskedastic and independent, the usual standard errors are invalid. There are several strategies to computing valid standard errors in such situations.

6.2.1 Huber-White and Liang-Zeger sandwich estimators

Let's say that $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \boldsymbol{\Sigma})$. Then, we can compute that the covariance matrix of the least squares estimate $\widehat{\boldsymbol{\beta}}$ is

$$\text{Var}[\widehat{\boldsymbol{\beta}}] = (\boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{\Sigma}\boldsymbol{X})(\boldsymbol{X}^T\boldsymbol{X})^{-1}. \tag{12}$$

Note that this expression reduces to the usual $\sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ when $\boldsymbol{\Sigma} = \sigma^2\boldsymbol{I}$. It is called the sandwich variance between we have the $(\boldsymbol{X}^T\boldsymbol{\Sigma}\boldsymbol{X})$ term sandwiched between two $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ terms. If we have some estimate $\widehat{\boldsymbol{\Sigma}}$ of the covariance matrix, we can construct

$$\widehat{\text{Var}}[\widehat{\boldsymbol{\beta}}] \equiv (\boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\widehat{\boldsymbol{\Sigma}}\boldsymbol{X})(\boldsymbol{X}^T\boldsymbol{X})^{-1}. \tag{13}$$

Different estimates $\widehat{\boldsymbol{\Sigma}}$ are appropriate in different situation. Below we consider two of the most common choices: one for heteroskedasticity (due to Huber-White) and one for correlated errors (due to Liang-Zeger).

Huber-White standard errors. Now, suppose $\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2, \ldots, \sigma_n^2)$ for some variances $\sigma_1^2, \ldots, \sigma_n^2 > 0$. The Huber-White sandwich estimator is defined by (12), with

$$\widehat{\boldsymbol{\Sigma}} \equiv \text{diag}(\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2), \quad \text{where} \quad \widehat{\sigma}_i^2 = (y_i - \boldsymbol{x}_{i*}^T \widehat{\boldsymbol{\beta}})^2. \tag{14}$$

While each estimator $\widehat{\sigma}_i^2$ is very poor, Huber and White's insight was that the resulting estimate of the (averaged) quantity $\boldsymbol{X}^T \widehat{\boldsymbol{\Sigma}} \boldsymbol{X}$ is not bad.

Liang-Zeger standard errors. Next, let's consider the case of correlated errors. Specifically, suppose that the observations are *clustered*, with correlated errors among clusters but not between clusters (recall Section 3.1). Suppose there are C clusters of observations, with the ith observation belonging to cluster $c(i) \in \{1, \ldots, C\}$. Suppose for the sake of simplicity that the observations are ordered so that clusters are contiguous. Let $\widehat{\boldsymbol{\epsilon}}_c$ be the vector of residuals in cluster c, so that $\widehat{\boldsymbol{\epsilon}} = (\widehat{\boldsymbol{\epsilon}}_1, \ldots, \widehat{\boldsymbol{\epsilon}}_C)$. Then, the true covariance matrix is $\boldsymbol{\Sigma} = \text{block-diag}(\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_C)$ for some positive definite $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_C$. The Liang-Zeger estimator is then defined by (12), with

$$\widehat{\boldsymbol{\Sigma}} \equiv \text{block-diag}(\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{1}}, \ldots, \widehat{\boldsymbol{\Sigma}}_{\boldsymbol{C}}), \quad \text{where} \quad \widehat{\boldsymbol{\Sigma}}_{\boldsymbol{c}} \equiv \widehat{\boldsymbol{\epsilon}}_c \widehat{\boldsymbol{\epsilon}}_c^T. \tag{15}$$

Note that the Liang-Zeger estimator is a generalization of the Huber-White estimator. Its justification is similar as well: while each $\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{c}}$ is a poor estimator, the resulting estimate of the (averaged) quantity $\boldsymbol{X}^T \widehat{\boldsymbol{\Sigma}} \boldsymbol{X}$ is not bad as long as the number of clusters is large. Liang-Zeger standard errors are sometimes referred to as "clustered standard errors."

6.2.2 Bootstrap

A completely different approach to constructing robust standard errors is the *bootstrap*. The core idea of the bootstrap is to use the data to construct an approximation to the data-generating distribution, and then to approximate the sampling distribution of any test statistic by simulating from this approximate data-generating distribution. This approach, pioneered by Brad Efron in 1979, replaces mathematical derivations with computation. The bootstrap is extremely flexible, and can be adapted to apply in a variety of settings.

Parametric bootstrap. The parametric bootstrap proceeds by fitting a parametric model, and then by resampling from this model. In the linear regression case, we use the original data to fit $(\widehat{\boldsymbol{\beta}}, \widehat{\sigma}^2)$. Then, we sample new response vectors

$$y_i^b = \boldsymbol{x}_{i*}^T \widehat{\boldsymbol{\beta}} + \epsilon_i^b, \quad \epsilon_i^b \overset{\text{i.i.d.}}{\sim} N(0, \widehat{\sigma}^2) \quad \text{for } b = 1, \ldots, B. \tag{16}$$

We then fit a least squares coefficient vector $\widehat{\boldsymbol{\beta}}^b$ to $(\boldsymbol{X}, \boldsymbol{y}^b)$ for each b, and then get variance estimates by treating $\{\widehat{\boldsymbol{\beta}}^b\}_{b=1}^B$ as though it were the sampling distribution of $\widehat{\boldsymbol{\beta}}$. For example, we could use the sample standard deviation of $\widehat{\beta}_j^b$ as the standard error for β_j.

This the most model-based of the bootstrap variants. It assumes a completely well-specified model, and gives equivalent results to traditional parametric inference. It is typically not applied in regression settings, and presented here mainly for pedagogical purposes.

Residual bootstrap. We can weaken the assumptions of the parametric bootstrap by assuming only that $y_i = \boldsymbol{x}_{i*}^T \boldsymbol{\beta} + \epsilon_i$, where $\epsilon_i \overset{\text{i.i.d.}}{\sim} F$ for some distribution F. Then, the data-generating distribution is specified by $(\boldsymbol{\beta}, F)$, which we approximate by substituting $\widehat{\boldsymbol{\beta}}$ for $\boldsymbol{\beta}$ and the empirical

distribution of the residuals $\widehat{\epsilon}_i$ (call it $\widehat{F}$) for F. We can then sample new response vectors based on this approximate data-generating distribution:

$$y_i^b = \boldsymbol{x}_{i*}^T \widehat{\boldsymbol{\beta}} + \epsilon_i^b, \quad \epsilon_i^b \overset{\text{i.i.d.}}{\sim} \widehat{F} \quad \text{for } b = 1, \ldots, B. \tag{17}$$

Note that i.i.d. sampling ϵ_i^b from $\widehat{F}$ amounts to sampling $(\epsilon_1^b, \ldots, \epsilon_n^b)$ with replacement from $(\widehat{\epsilon}_1, \ldots, \widehat{\epsilon}_n)$. Then, as with the parametric bootstrap, we fit a least squares coefficient vector $\widehat{\boldsymbol{\beta}}^b$ to $(\boldsymbol{X}, \boldsymbol{y}^b)$ for each b and obtain standard errors by treating $\{\widehat{\boldsymbol{\beta}}^b\}_{b=1}^B$ as though it were the sampling distribution of $\widehat{\boldsymbol{\beta}}$.

The residual bootstrap corrects for non-normality, but not heteroskedasticity or correlated errors, since it assumes that the noise terms are i.i.d. from some distribution.

Pairs bootstrap. Weakening the assumptions further, let's assume only that $(\boldsymbol{x}_{i*}, y_i) \overset{\text{i.i.d.}}{\sim} F$ for some joint distribution F. We then resample our observations by sampling with replacement from the original observations.

Note that, unlike the parametric or residual bootstrap, the pairs bootstrap treats the predictors $\boldsymbol{X}$ as random rather than fixed. The benefit of the pairs bootstrap is that it does not assume homoskedasticity, since the error variance is allowed to depend on $\boldsymbol{x}_{i*}$. Therefore, the pairs bootstrap addresses both non-normality and heteroskedasticity, though it does not address correlated errors (though variants of the pairs bootstrap do; see below). Note that the pairs bootstrap does not even assume that $\mathbb{E}[y_i] = \boldsymbol{x}_{i*}^T \boldsymbol{\beta}$ for some $\boldsymbol{\beta}$. However, in the presence of model bias, it is unclear for what parameters we are even doing inference. While the pairs bootstrap assumes less than the residual bootstrap, it may be somewhat less efficient in the case when the assumptions of the latter are met.

The pairs bootstrap has several variants that help it overcome correlated errors, in addition to heteroskedasticity. The *cluster bootstrap* is applicable in the case when errors have a clustered/grouped structure. In this case, we sample entire clusters of observations, with replacement, from the original set of clusters. The *moving blocks bootstrap* is applicable in the case of spatially or temporally structured errors. In this variant of the pairs bootstrap, we resample spatially or temporally adjacent blocks of observations together to preserve their joint correlation structure.

6.3 Robust hypothesis testing

In principle, any of the robust standard error constructions from Section 6.2 can be used to construct robust hypothesis tests. In this section, we will discuss a separate set of robust methodologies designed specifically for hypothesis testing. These fall roughly into two main categories: permutation tests (Section 6.3.1) and bootstrap-based tests (Section 6.3.2)). There is a third category of tests based on ranks (e.g. the Wilcoxon test), which we will not discuss in this class.

6.3.1 Permutation tests

Independence testing. Permutation tests are an easy way of testing the null hypothesis of independence between two random variables (or vectors). For our purposes, suppose that $(\boldsymbol{x}_{i*}, y_i)$ are drawn i.i.d. from some joint distribution F (as opposed to the usual assumption that $\boldsymbol{X}$ is fixed). Then, consider the null hypothesis

$$H_0 : \boldsymbol{x} \perp\!\!\!\perp y. \tag{18}$$

This null hypothesis is related to the null hypothesis $H_0 : \boldsymbol{\beta}_{\text{-}0} = 0$ in a linear regression, as formalized by the following lemma.

Lemma 6.1. *Suppose $\boldsymbol{x} \in \mathbb{R}^{p-1}$ has a nondegenerate distribution $F_{\boldsymbol{x}}$ in the sense that there does not exist a vector $c \in \mathbb{R}^{p-1}$ such that $\boldsymbol{c}^T\boldsymbol{x}$ is deterministic. Suppose also that $F_{y|\boldsymbol{x}}$ is a distribution such that $\mathbb{E}[y|\boldsymbol{x}] = \beta_0 + \boldsymbol{x}^T\boldsymbol{\beta}_{\text{-}0}$ and that the distribution $F_{y|\boldsymbol{x}}$ is specified by its mean. Then,*

$$\boldsymbol{x} \perp\!\!\!\perp y \quad \iff \quad \boldsymbol{\beta}_{\text{-}0} = \boldsymbol{0}. \tag{19}$$

Proof. If $\boldsymbol{\beta}_{\text{-}0} = \boldsymbol{0}$, then $\mathbb{E}[y|\boldsymbol{x}] = \beta_0$. Therefore, the mean of y does not depend on $\boldsymbol{x}$. By the assumption on $F_{y|\boldsymbol{x}}$, it follows that the entire distribution $F_{y|\boldsymbol{x}}$ does not depend on $\boldsymbol{x}$, i.e. $y \perp\!\!\!\perp \boldsymbol{x}$. If $\boldsymbol{\beta}_{\text{-}0} \neq \boldsymbol{0}$, then $\mathbb{E}[y|\boldsymbol{x}] = \beta_0 + \boldsymbol{x}^T\boldsymbol{\beta}_{\text{-}0}$, which by assumption is non-constant. Since $\mathbb{E}[y|\boldsymbol{x}]$ depends on $\boldsymbol{x}$, it follows that y is not independent of $\boldsymbol{x}$. □

Therefore, any valid independence test automatically gives a non-normality-robust and heteroskedasticity-robust test of $H_0 : \boldsymbol{\beta}_{\text{-}0} = \boldsymbol{0}$ in a linear regression.

The permutation test. Now, suppose we have n i.i.d. samples $(\boldsymbol{x}_{i*}, y_i)$ from F. Under the independence null hypothesis (18), the distribution of the data is unchanged if we permute the response variables y_i. Formally, let $\boldsymbol{y}_{()}$ be the order statistics of the response variable, let S_n be the permutation group on $\{1, \ldots, n\}$, and let $\boldsymbol{y}_\tau$ denote the permutation of $\boldsymbol{y}$ by $\tau \in S_n$. Then,

$$\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{y}_{()} \sim \frac{1}{n!}\sum_{\tau \in S_n} \delta(\boldsymbol{y}_\tau). \tag{20}$$

Now, let $T(\boldsymbol{X}, \boldsymbol{y})$ be any test statistic measuring the association between $\boldsymbol{y}$ and $\boldsymbol{X}$, e.g. a linear regression F-statistic. Then, the above distributional result implies that

$$T(\boldsymbol{X}, \boldsymbol{y})|\boldsymbol{X}, \boldsymbol{y}_{()} \sim \frac{1}{n!}\sum_{\tau \in S_n} \delta(T(\boldsymbol{X}, \boldsymbol{y}_\tau)). \tag{21}$$

Hence, we can compute the null distribution of T by repeatedly permuting the response $\boldsymbol{y}$ and recomputing $T(\boldsymbol{X}, \boldsymbol{y}_\tau)$. This gives rise to the permutation p-value

$$p^{\text{perm}} \equiv \frac{1}{n!}\sum_{\tau \in S_n} \mathbb{1}(T(\boldsymbol{X}, \boldsymbol{y}_\tau) \geq T(\boldsymbol{X}, \boldsymbol{y})). \tag{22}$$

The uniform distribution of $T(\boldsymbol{X}, \boldsymbol{y})|\boldsymbol{X}, \boldsymbol{y}_{()}$ implies that

$$\mathbb{P}[p^{\text{perm}} \leq t|\boldsymbol{X}, \boldsymbol{y}_{()}] \leq t \quad \implies \quad \mathbb{P}[p^{\text{perm}} \leq t] = \mathbb{E}[\mathbb{P}[p^{\text{perm}} \leq t|\boldsymbol{X}, \boldsymbol{y}_{()}]] \leq t \quad \text{for all } t \in [0, 1].. \tag{23}$$

In practice, p^{perm} is approximated by independently sampling B permutations $\tau_1, \ldots, \tau_B$ from the uniform distribution over S_n. Letting τ_0 be the identity permutation, it follows that

$$\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{y} \in \{\boldsymbol{y}_{\tau_0}, \ldots, \boldsymbol{y}_{\tau_B}\} \sim \frac{1}{B+1}\sum_{b=0}^{B} \delta(\boldsymbol{y}_{\tau_b}). \tag{24}$$

Similar logic as above leads to the approximate permutation p-value

$$\widehat{p}^{\text{perm}} \equiv \frac{1}{B+1}\sum_{b=0}^{B} \mathbb{1}(T(\boldsymbol{X}, \boldsymbol{y}_{\tau_b}) \geq T(\boldsymbol{X}, \boldsymbol{y})) = \frac{1}{B+1}\left(1 + \sum_{b=1}^{B} \mathbb{1}(T(\boldsymbol{X}, \boldsymbol{y}_{\tau_b}) \geq T(\boldsymbol{X}, \boldsymbol{y}))\right). \tag{25}$$

Although $\widehat{p}^{\text{perm}}$ can be viewed as an approximation to $\boldsymbol{p}^{\text{perm}}$, it is also stochastically larger than the uniform distribution in finite samples:

$$\mathbb{P}[\widehat{p}^{\text{perm}} \leq t] \leq t \quad \text{for all } t \in [0, 1]. \tag{26}$$

Warning: A common mistake is to omit the "1+" in the numerator and denominator of the definition (25). The resulting p-value is *not valid* in the sense of equation (26).

Example. A common application of the permutation test is testing for equality of distributions in the two-sample problem, where the permutation test amounts to generating a null distribution for any test statistic (e.g. a difference in means) by pooling together the two samples and randomly reassigning the classes of the samples.

Strengths and weaknesses. The strength of the permutation test is that it is valid under almost no assumptions on the data-generating process. Its main weakness is that it is not applicable to the hypothesis $H_0 : \beta_S = 0$ for any group of predictors $S \neq \{1, \dots, p-1\}$. Intuitively, this would require a fancy kind of permutation that breaks the association between $\boldsymbol{y}$ and $\boldsymbol{X}_{*,S}$ while preserving the association between $\boldsymbol{X}_{*,S}$ and $\boldsymbol{X}_{*,\text{-}S}$. This amounts to a test of *conditional* independence, which requires more assumptions on the joint distribution $F_{\boldsymbol{x},y}$ than an independence test. Another weakness of a permutation test is that it is computationally expensive, although in the 21st century this is not a huge issue.

6.3.2 Bootstrap-based tests

While the bootstrap is commonly associated with the construction of standard errors, it can also be used directly for hypothesis testing. Suppose we wish to test the linear regression null hypothesis $H_0 : \boldsymbol{\beta}_S = \boldsymbol{0}$ for some $S \subseteq \{1, \dots, p-1\}$ (which recall we cannot do using a permutation test). We compute some test statistic $T(\boldsymbol{X}, \boldsymbol{y})$ measuring the significance of $\boldsymbol{\beta}_S$ (e.g. an F-statistic but it could be anything else). Then, we can use a variant of the residual bootstrap. We fit the least squares estimate $\widehat{\boldsymbol{\beta}}$ as usual and extract the residuals $\widehat{\epsilon}_i \equiv y_i - \boldsymbol{x}_{i*}^T \widehat{\boldsymbol{\beta}}$ and their empirical distribution $\widehat{F}$. Then, placing ourselves under the null hypothesis, we generate new samples $\boldsymbol{y}^b$ from the null distribution analogously to the usual residual bootstrap (17):

$$y_i^b = \boldsymbol{x}_{i,\text{-}S}^T \widehat{\boldsymbol{\beta}}_{\text{-}S} + \epsilon_i^b, \quad \epsilon_i^b \overset{\text{i.i.d.}}{\sim} \widehat{F} \quad \text{for } b = 1, \dots, B. \tag{27}$$

We can then build a null distribution by recomputing $T(\boldsymbol{X}, \boldsymbol{y}^b)$ for each b and then define the bootstrap-based p-value

$$p^{\text{boot}} \equiv \frac{1}{B+1} \left(1 + \sum_{b=1}^{B} \mathbb{1}(T(\boldsymbol{X}, \boldsymbol{y}^b) \geq T(\boldsymbol{X}, \boldsymbol{y})) \right). \tag{28}$$

This bootstrap-based hypothesis test is not as robust as a permutation test or a heteroskedasticity-robust standard error, since the residual bootstrap implicitly assumes homoskedasticity. However, compared to parametric linear model inference, this bootstrap test affords the additional flexibility of using *any* test statistic T (including one based on, say, machine learning). Note that, while the pairs bootstrap is more robust than the residual bootstrap, the pairs bootstrap does not allow one to create samples under the null distribution and therefore cannot be used for hypothesis testing.

7 R demo

Let's take a look at the crime data from HW2:

```
# read crime data
crime_data = read_tsv("../data/Statewide_crime.dat")

# read and transform population data
```

```
population_data = read_csv("../data/state-populations.csv")
population_data = population_data %>%
  filter(State != "Puerto Rico") %>%
  select(State, Pop) %>%
  rename(state_name = State, state_pop = Pop)

# collate state abbreviations
state_abbreviations = tibble(state_name = state.name,
                             state_abbrev = state.abb) %>%
  add_row(state_name = "District of Columbia", state_abbrev = "DC")

# add CrimeRate to crime_data
crime_data = crime_data %>%
  mutate(STATE = ifelse(STATE == "IO", "IA", STATE)) %>%
  rename(state_abbrev = STATE) %>%
  left_join(state_abbreviations, by = "state_abbrev") %>%
  left_join(population_data, by = "state_name") %>%
  mutate(CrimeRate = Violent/state_pop) %>%
  select(state_abbrev, CrimeRate, Metro, HighSchool, Poverty)

crime_data
```

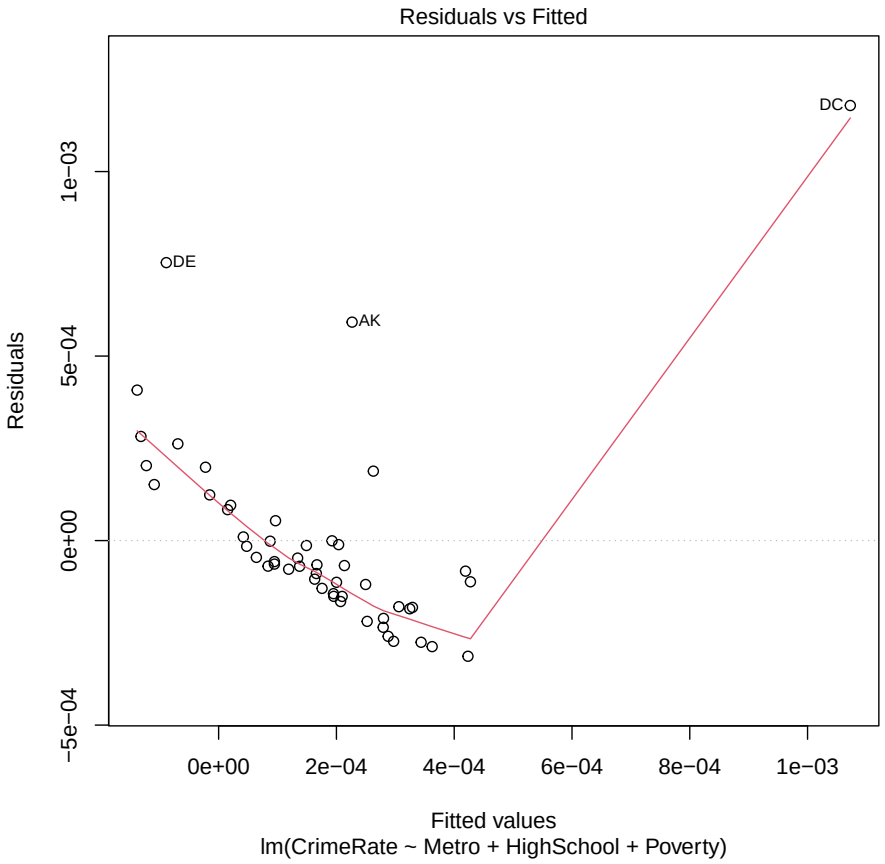
```
## # A tibble: 51 x 5
##    state_abbrev CrimeRate Metro HighSchool Poverty
##    <chr>            <dbl> <dbl>      <dbl>   <dbl>
##  1 AK           0.000819   65.6       90.2     8
##  2 AL           0.0000871  55.4       82.4    13.7
##  3 AR           0.000150   52.5       79.2    12.1
##  4 AZ           0.0000682  88.2       84.4    11.9
##  5 CA           0.0000146  94.4       81.3    10.5
##  6 CO           0.0000585  84.5       88.3     7.3
##  7 CT           0.0000867  87.7       88.8     6.4
##  8 DE           0.000664   80.1       86.5     5.8
##  9 FL           0.0000333  89.3       85.9     9.7
## 10 GA           0.0000419  71.6       85.2    10.8
## # ... with 41 more rows
```

Let's fit the linear regression:

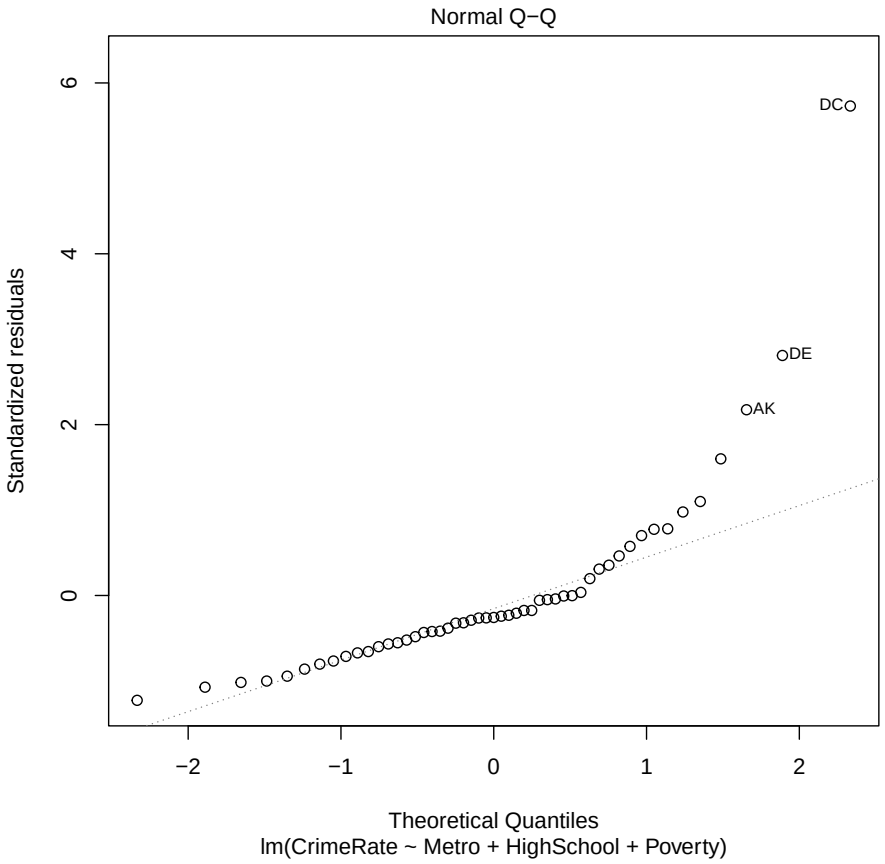
```
# note: we make the state abbreviations row names for better diagnostic plots
lm_fit = lm(CrimeRate ~ Metro + HighSchool + Poverty,
            data = crime_data %>% column_to_rownames(var = "state_abbrev"))
```

We can get the standard linear regression diagnostic plots as follows:

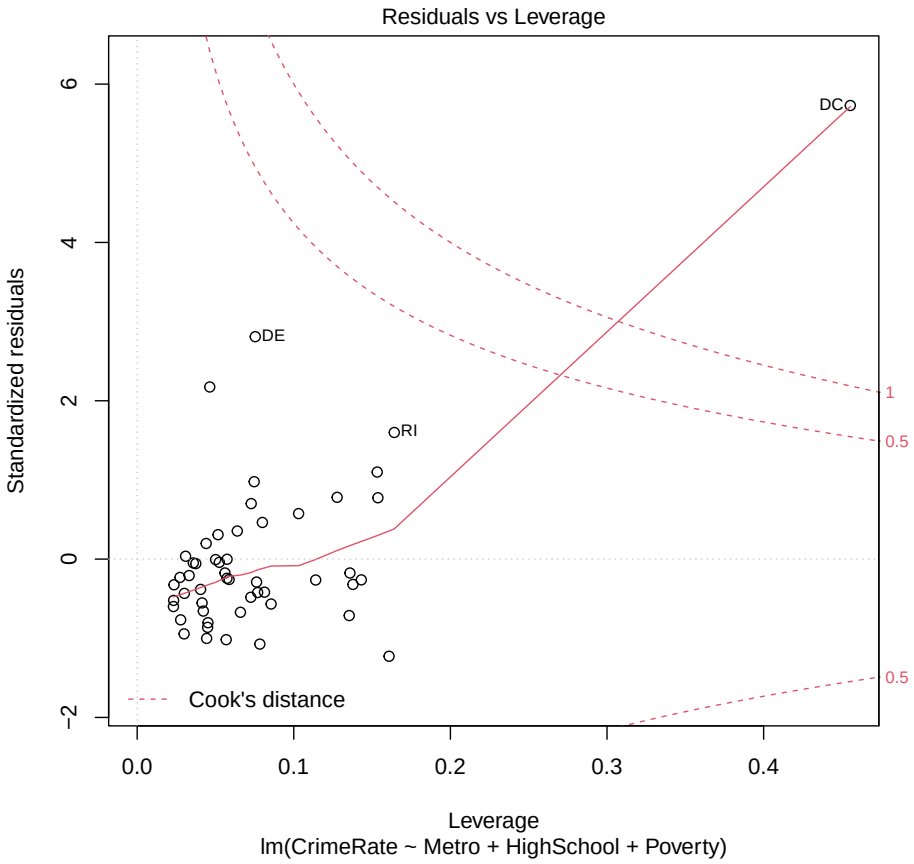
```
# residuals versus fitted
plot(lm_fit, which = 1)
```




```r
# residual QQ plot
plot(lm_fit, which = 2)
```




```r
# residuals versus leverage (with Cook's distance)
plot(lm_fit, which = 5)
```



The information underlying these diagnostic plots can be extracted as follows:

```
tibble(state = crime_data$state_abbrev,
       std_residual = rstandard(lm_fit),
       fitted_value = fitted.values(lm_fit),
       leverage = hatvalues(lm_fit),
       cooks_dist = cooks.distance(lm_fit))
```

```
## # A tibble: 51 x 5
##    state std_residual fitted_value leverage cooks_dist
##    <chr>        <dbl>        <dbl>    <dbl>      <dbl>
##  1 AK           2.17     0.000227    0.0463   0.0574
##  2 AL          -0.422    0.000200    0.0769   0.00371
##  3 AR           1.10    -0.000132    0.153    0.0547
##  4 AZ          -1.02     0.000344    0.0568   0.0156
##  5 CA          -0.264    0.0000839   0.114    0.00224
##  6 CO          -0.383    0.000163    0.0405   0.00155
##  7 CT          -0.175    0.000134    0.0561   0.000456
##  8 DE           2.81    -0.0000888   0.0754   0.161
##  9 FL          -0.804    0.000252    0.0452   0.00764
## 10 GA          -0.599    0.000207    0.0232   0.00213
## # ... with 41 more rows
```

Clearly DC is an outlier. We can either run a robust estimation procedure or we can redo the analysis without DC. Let's try both. First, we try robust regression using `MASS::rlm`:

```
rlm_fit = MASS::rlm(CrimeRate ~ Metro + HighSchool + Poverty, data = crime_data)
summary(rlm_fit)

##
## Call: rlm(formula = CrimeRate ~ Metro + HighSchool + Poverty, data = crime_data)
## Residuals:
##        Min         1Q     Median         3Q        Max
## -8.297e-05 -3.787e-05 -2.249e-05  4.407e-05  2.063e-03
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept) -0.0009  0.0004    -2.2562
## Metro        0.0000  0.0000    -1.2963
## HighSchool   0.0000  0.0000     2.6506
## Poverty      0.0000  0.0000     2.7546
##
## Residual standard error: 6.048e-05 on 47 degrees of freedom
```

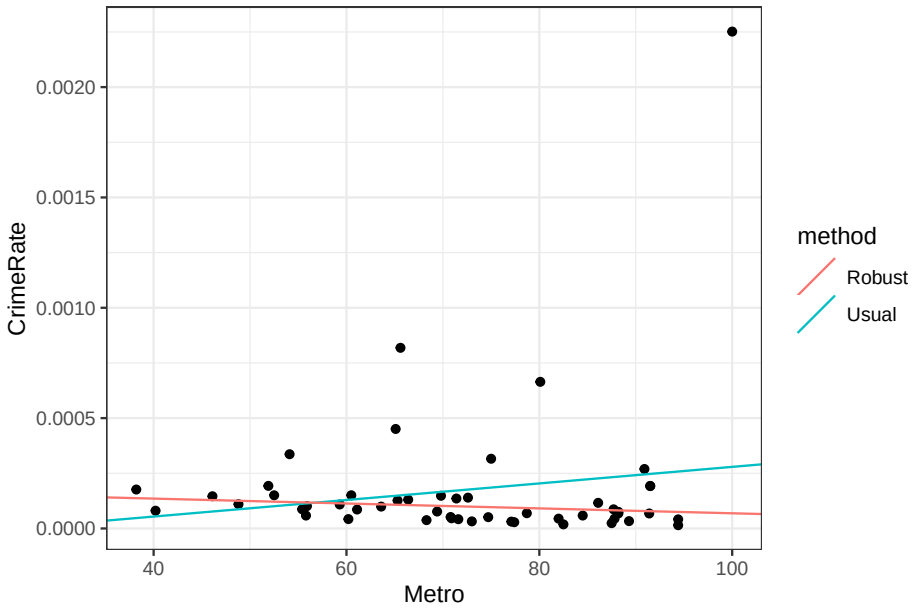
For some reason, the p-values are not computed automatically. We can compute them ourselves instead:

```
summary(rlm_fit)$coef %>%
  as.data.frame() %>%
  rename(Estimate = Value) %>%
  mutate(`p value` = 2*dnorm(-abs(`t value`)))

##                  Estimate   Std. Error   t value    p value
## (Intercept) -8.538466e-04 3.784466e-04 -2.256188 0.06260042
## Metro       -8.639252e-07 6.664623e-07 -1.296285 0.34439400
## HighSchool   1.037849e-05 3.915573e-06  2.650568 0.02378865
## Poverty      1.252839e-05 4.548172e-06  2.754600 0.01795833
```

To see the robust estimation action visually, let's consider a univariate example:

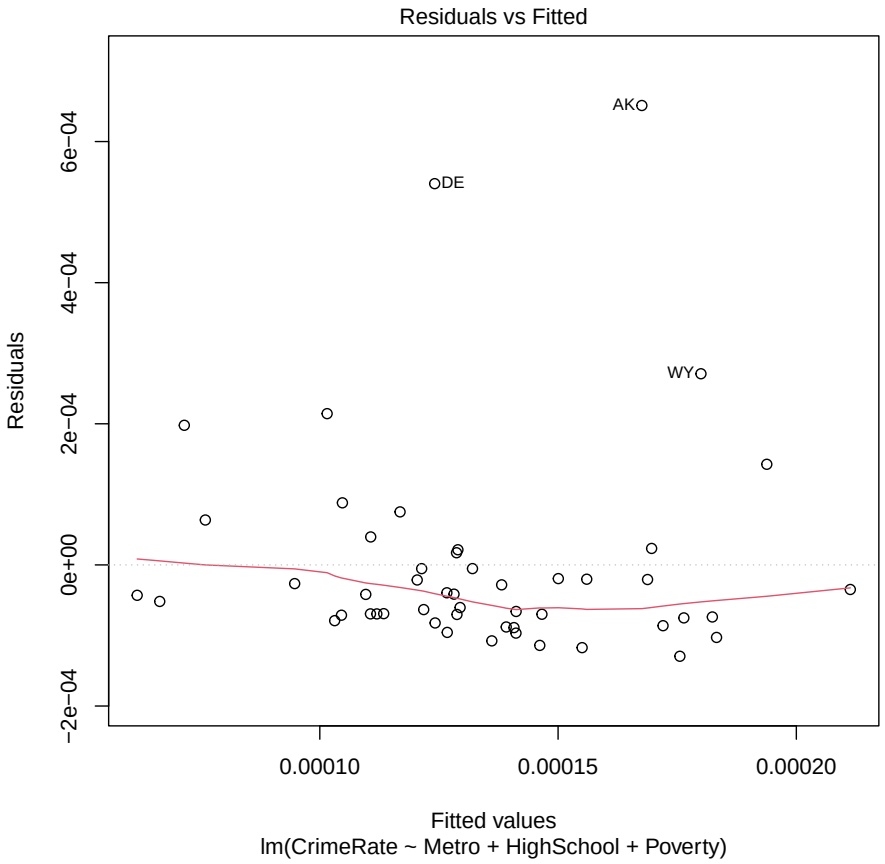
```
# usual and robust univariate fits
lm_fit = lm(CrimeRate ~ Metro, data = crime_data)
rlm_fit = MASS::rlm(CrimeRate ~ Metro, data = crime_data)

# collate the fits into a tibble
line_fits = tibble(method = c("Usual", "Robust"),
                   intercept = c(coef(lm_fit)["(Intercept)"],
                                 coef(rlm_fit)["(Intercept)"]),
                   slope = c(coef(lm_fit)["Metro"],
                                 coef(rlm_fit)["Metro"]))

# plot the fits
ggplot() +
```

```
  geom_point(aes(x = Metro, y = CrimeRate), data = crime_data) +
  geom_abline(aes(intercept = intercept, slope = slope, colour = method),
              data = line_fits) +
  theme_bw()
```

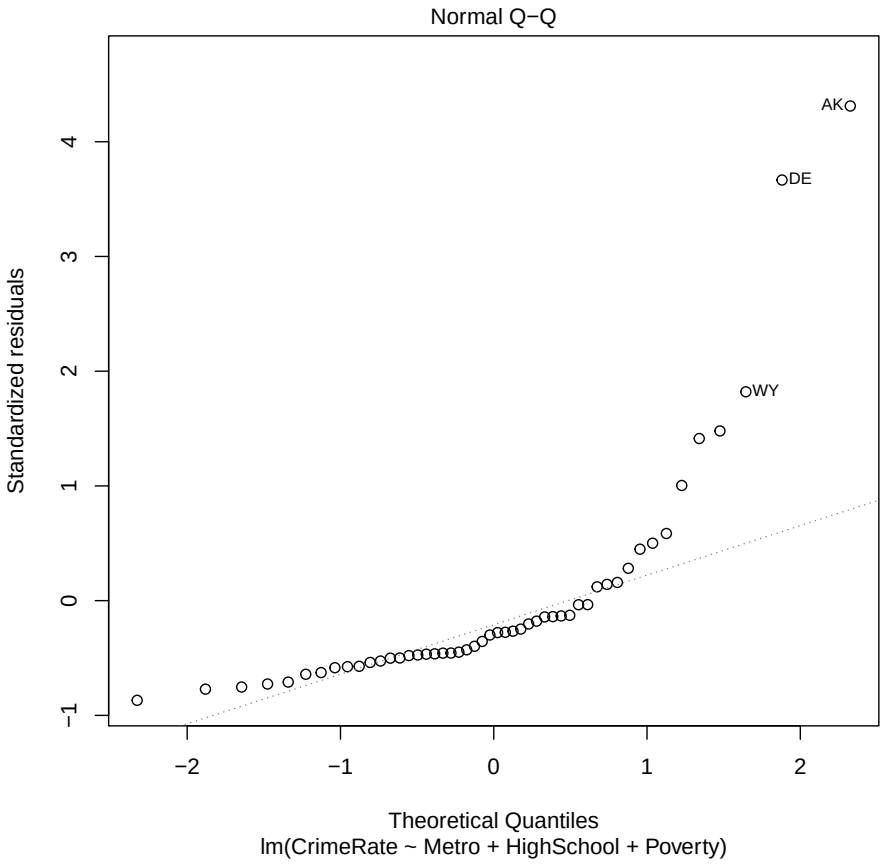


Next, let's try removing DC and running a usual linear regression.

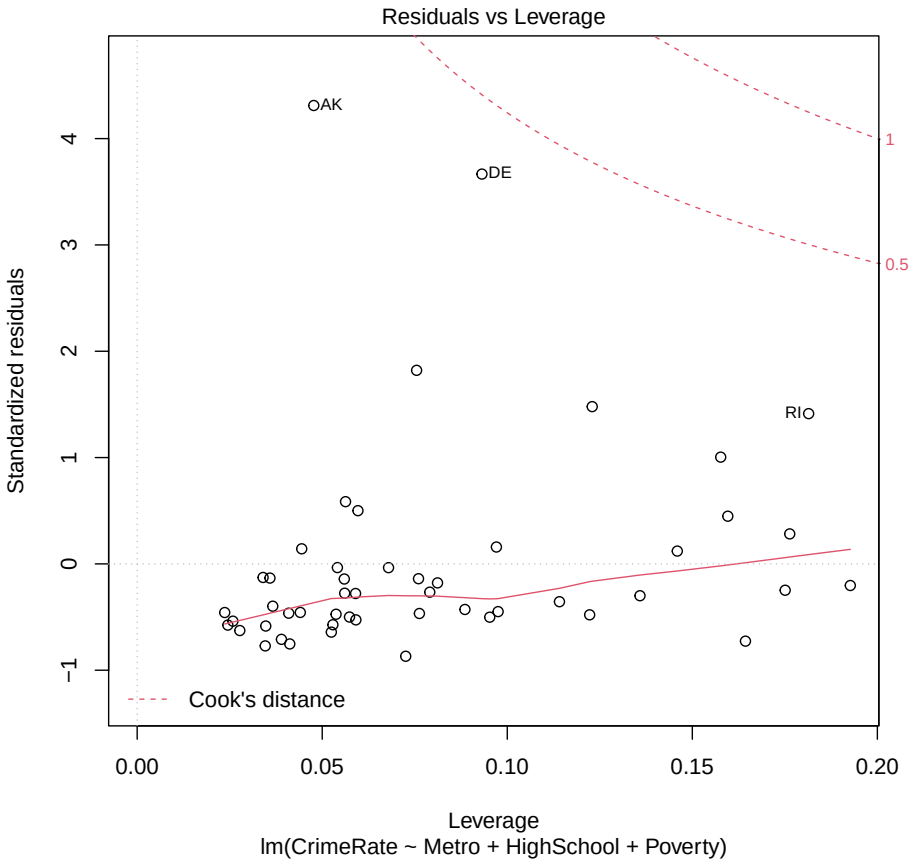
```
lm_fit_no_dc = lm(CrimeRate ~ Metro + HighSchool + Poverty,
             data = crime_data %>%
               filter(state_abbrev != "DC") %>%
               column_to_rownames(var = "state_abbrev"))

# residuals versus fitted
plot(lm_fit_no_dc, which = 1)
```




```r
# residual QQ plot
plot(lm_fit_no_dc, which = 2)
```




```
# residuals versus leverage (with Cook's distance)
plot(lm_fit_no_dc, which = 5)
```



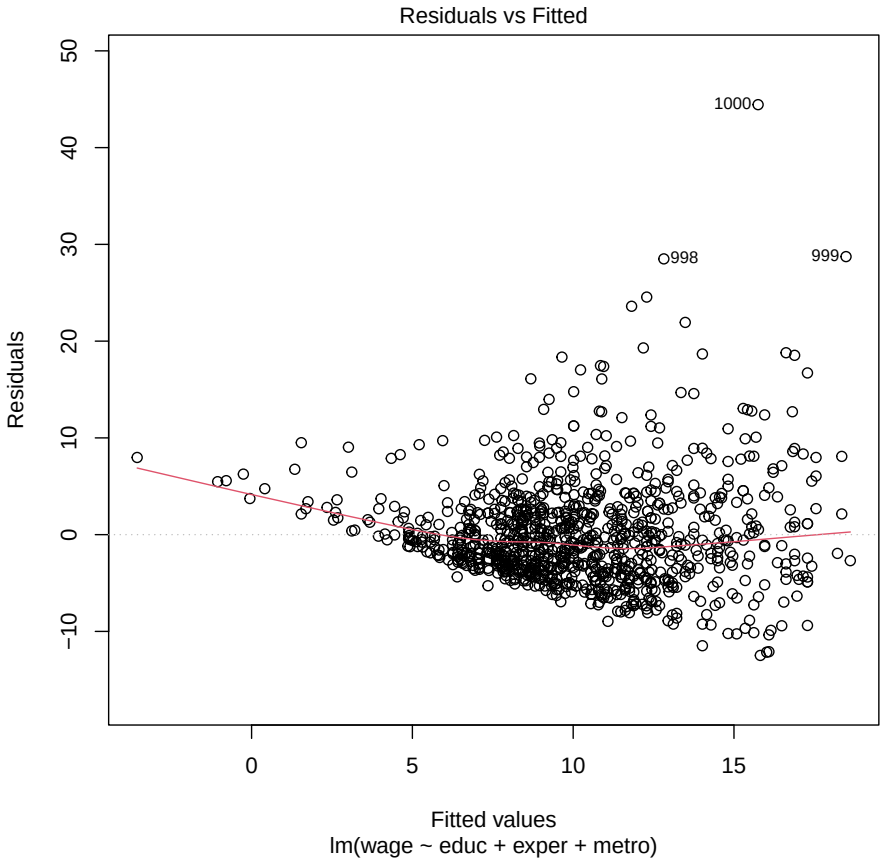
Next let's look at another dataset, from the Current Population Survey (CPS).

```
cps_data = read_tsv("../data/cps2.tsv")

## Rows:  1000 Columns:  10
## - Column specification ----------------------------
## Delimiter:  "\t"
## dbl (10):  wage, educ, exper, female, black, married, union, south, fulltime,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

cps_data

## # A tibble: 1,000 x 10
##     wage  educ exper female black married union south fulltime metro
##    <dbl> <dbl> <dbl>  <dbl> <dbl>   <dbl> <dbl> <dbl>    <dbl> <dbl>
##  1  2.03    13     2      1     0       0     0     1        0     0
##  2  2.07    12     7      0     0       0     0     0        0     1
##  3  2.12    12    35      0     0       0     0     1        1     1
##  4  2.54    16    20      1     0       0     0     1        1     1
##  5  2.68    12    24      1     0       1     0     1        0     1
##  6  3.09    13     4      0     0       0     0     1        0     1
```

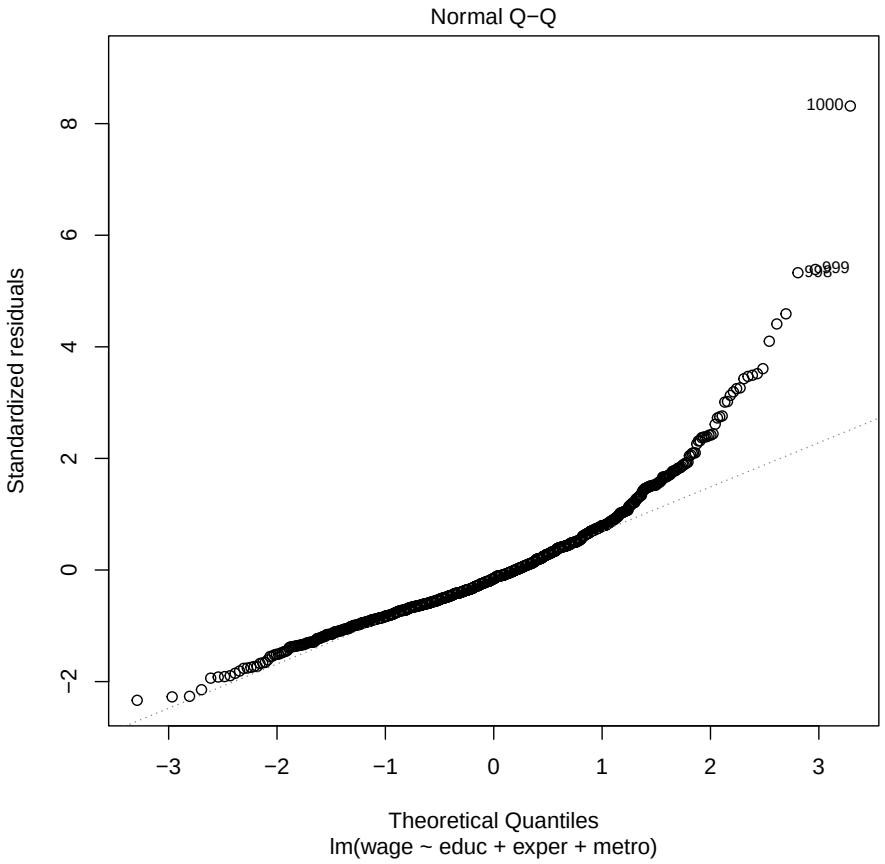
```
##  7  3.16    13     1      0     0       0     0     0       0     0
##  8  3.17    12    22      1     0       1     0     1       0     1
##  9  3.2     12    23      0     0       1     0     1       1     1
## 10  3.27    12     4      1     0       0     0     0       1     1
## # ... with 990 more rows
```

Suppose we want to regress `wage` on `educ`, `exper`, and `metro`. Let's take a look at the diagnostic plots.

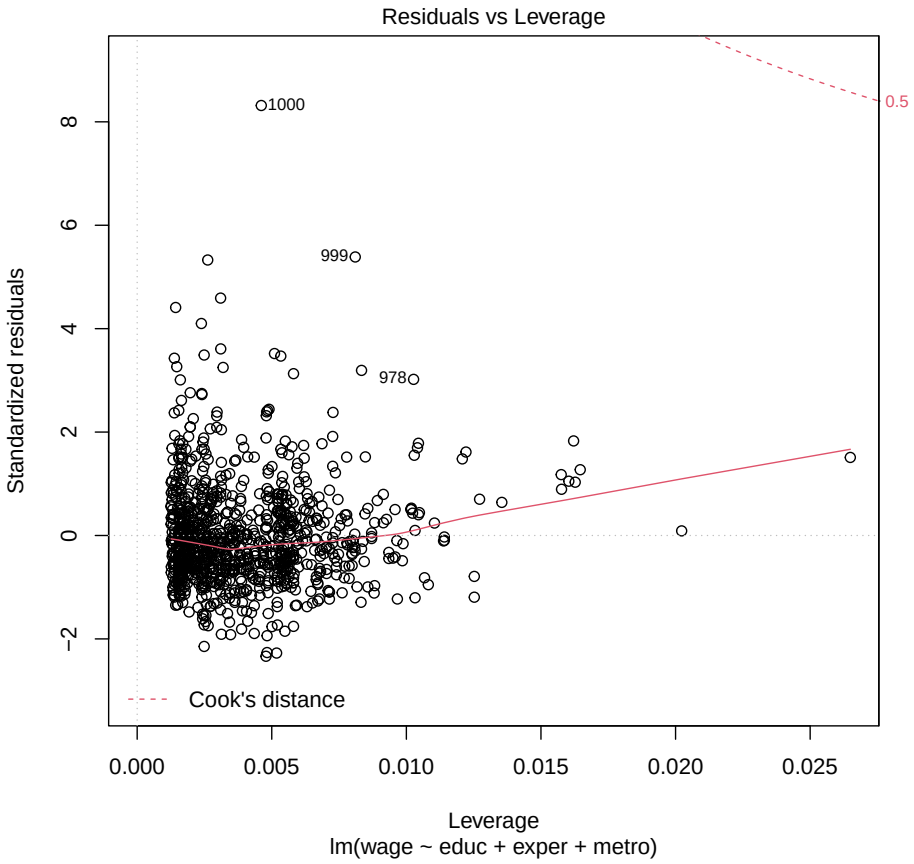
```
lm_fit = lm(wage ~ educ + exper + metro, data = cps_data)

# residuals versus fitted
plot(lm_fit, which = 1)
```




```
# residual QQ plot
plot(lm_fit, which = 2)
```




```r
# residuals versus leverage (with Cook's distance)
plot(lm_fit, which = 5)
```



The residuals versus fitted plot suggests significant heteroskedasticity, with variance growing as a function of the fitted value. To get standard errors robust to this heteroskedasticity, we can use one of the robust estimators discussed in Section 6.2. Most of the robust standard error constructions discussed in that section are implemented in the R package `sandwich`.

```
library(sandwich)
```

For example, Huber-White's heteroskedasticity-consistent estimate $\widehat{\text{Var}}[\widehat{\boldsymbol{\beta}}]$ can be obtain via `vcovHC`:

```
HW_cov = vcovHC(lm_fit)
HW_cov
```

```
##              (Intercept)          educ         exper         metro
## (Intercept)  1.484328645 -0.0967891868 -0.0096871141 -0.1218518012
## educ        -0.096789187  0.0070467982  0.0004037764  0.0018334348
## exper       -0.009687114  0.0004037764  0.0002517826  0.0008369831
## metro       -0.121851801  0.0018334348  0.0008369831  0.1197713348
```

Compare this to the traditional estimate:

```
usual_cov = vcovHC(lm_fit, type = "const")
usual_cov
```

```
##              (Intercept)          educ         exper         metro
## (Intercept)  1.157049852 -0.0671656102 -0.0070323974 -0.1287058354
## educ        -0.067165610  0.0048945781  0.0001924359 -0.0018227782
## exper       -0.007032397  0.0001924359  0.0002320022  0.0001471354
## metro       -0.128705835 -0.0018227782  0.0001471354  0.1858394060

# extract the variance estimates from the diagonal
tibble(variable = rownames(usual_cov),
       usual_variance = diag(usual_cov),
       HW_variance = diag(HW_cov))

## # A tibble: 4 x 3
##   variable    usual_variance HW_variance
##   <chr>                <dbl>       <dbl>
## 1 (Intercept)       1.16        1.48
## 2 educ              0.00489     0.00705
## 3 exper             0.000232    0.000252
## 4 metro             0.186       0.120
```

Bootstrap standard errors are also implemented in `sandwich`:

```
# pairs bootstrap
bootstrap_cov = vcovBS(lm_fit, type = "xy")
tibble(variable = rownames(usual_cov),
       usual_variance = diag(usual_cov),
       HW_variance = diag(HW_cov),
       boostrap_variance = diag(bootstrap_cov))

## # A tibble: 4 x 4
##   variable    usual_variance HW_variance boostrap_variance
##   <chr>                <dbl>       <dbl>             <dbl>
## 1 (Intercept)       1.16        1.48              1.29
## 2 educ              0.00489     0.00705           0.00624
## 3 exper             0.000232    0.000252          0.000219
## 4 metro             0.186       0.120             0.107
```

Note that the bootstrap standard errors are closer to the HW ones than the standard ones.

Other kinds of robust standard errors are implemented in `sandwich`, like clustered standard errors (via `vcovCL`) and many others we have not discussed.

The covariance estimate produced by `sandwich` can be easily integrated into linear model inference using the package `lmtest`.

```
library(lmtest)

# robust t-tests for coefficients
coeftest(lm_fit, vcov. = vcovHC)

##
## t test of coefficients:
##
```

```
##              Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -9.913984   1.218330 -8.1374 1.197e-15 ***
## educ         1.233964   0.083945 14.6996 < 2.2e-16 ***
## exper        0.133244   0.015868  8.3972 < 2.2e-16 ***
## metro        1.524104   0.346080  4.4039 1.178e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# robust confidence intervals for coefficients
coefci(lm_fit, vcov. = vcovHC)
```

```
##                   2.5 %     97.5 %
## (Intercept) -12.3047729 -7.5231954
## educ          1.0692342  1.3986938
## exper         0.1021058  0.1643816
## metro         0.8449747  2.2032337
```

```
# robust F-test
lm_fit_partial = lm(wage ~ educ, data = cps_data) # a partial model
waldtest(lm_fit_partial, lm_fit, vcov = vcovHC)
```

```
## Wald test
##
## Model 1: wage ~ educ
## Model 2: wage ~ educ + exper + metro
##   Res.Df Df      F    Pr(>F)
## 1    998
## 2    996  2 40.252 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```