Chapter 8: heteroskedasticity

We start again by loading the data and filtering it so that we restrict it to properties that host at most 6 people. Also, as in Chapter 3, we again create a variable `review_scores_rating_standardized` with the standardized review score.

```
# Load the datasets from the RData file
load("../dataCreated/listings_clean.RData")

listings_clean_filtered <- listings_clean %>%
  filter(accommodates <= 6)

# Standardize review_scores_rating
listings_clean_filtered <- listings_clean_filtered %>%
  mutate(review_scores_rating_standardized =
           (review_scores_rating - mean(review_scores_rating, na.rm = TRUE)) /
           sd(review_scores_rating, na.rm = TRUE))
```

Our point of departure for Chapter 8 is, as for Chapter 4, the richer model from Chapter 3 where we regress the log price on `review_scores_rating`, `accommodates`, and 4 neighborhood characteristics.

```
estimatesFullModel <- lm(log(price) ~ review_scores_rating_standardized + accommodates + Centrality + Qu
summary(estimatesFullModel)
```

```
##
## Call:
## lm(formula = log(price) ~ review_scores_rating_standardized +
##     accommodates + Centrality + Quietness + Coolness + Fanciness,
##     data = listings_clean_filtered)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.48845 -0.24997  0.01026  0.23530  0.94151
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        3.10015    0.49275   6.292    8e-10 ***
## review_scores_rating_standardized  0.05000    0.01887   2.650  0.00837 **
## accommodates                       0.21641    0.01480  14.625  < 2e-16 ***
## Centrality                         0.08591    0.04406   1.950  0.05186 .
## Quietness                          0.12896    0.04451   2.897  0.00397 **
## Coolness                           0.09606    0.07834   1.226  0.22079
## Fanciness                         -0.14765    0.05382  -2.744  0.00634 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3834 on 414 degrees of freedom
## Multiple R-squared:  0.3494, Adjusted R-squared:   0.34
## F-statistic: 37.05 on 6 and 414 DF,  p-value: < 2.2e-16
```

The reported point estimates are unbiased (Chapter 3)/consistent (Chapter 5) under MLR1-MLR4. The standard errors are valid when we make the additional assumption of homoskedasticity.

Next, we test the null of homoskedasticity.

```
bptest(estimatesFullModel)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  estimatesFullModel
## BP = 19.232, df = 6, p-value = 0.00379
```

The null of homoskedasticity is rejected.

Next, we do weighted least squares.

```
# Obtain residuals
residuals_full_model <- residuals(estimatesFullModel)

# Step 2: Regress squared residuals on predictors to model heteroskedasticity
squared_residuals <- residuals_full_model^2
model_resid_squared <- lm(squared_residuals ~ review_scores_rating_standardized + accommodates + Centra

# Obtain fitted values from this regression
fitted_values <- fitted(model_resid_squared)

# Step 3: Calculate weights as inverse of the square root of fitted values
weights <- 1 / sqrt(fitted_values)
```

```
## Warning in sqrt(fitted_values): NaNs produced
```

```
# Step 4: Run the WLS regression with these weights
wls_model <- lm(log(price) ~ review_scores_rating_standardized + accommodates + Centrality + Quietness +
                data = listings_clean_filtered, weights = weights)

# Summary of the WLS model
summary(wls_model)
```

```
##
## Call:
## lm(formula = log(price) ~ review_scores_rating_standardized +
##     accommodates + Centrality + Quietness + Coolness + Fanciness,
##     data = listings_clean_filtered, weights = weights)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -2.84538 -0.41093  0.01522  0.37911  1.47203
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        3.26967    0.46508   7.030 8.64e-12 ***
## review_scores_rating_standardized  0.05635    0.01872   3.011 0.002766 **
## accommodates                       0.20550    0.01377  14.919  < 2e-16 ***
## Centrality                         0.10411    0.03976   2.618 0.009164 **
## Quietness                          0.12621    0.04266   2.959 0.003266 **
## Coolness                           0.07941    0.07390   1.075 0.283179
## Fanciness                         -0.17061    0.04430  -3.851 0.000136 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6209 on 411 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.3722, Adjusted R-squared:  0.363
## F-statistic: 40.61 on 6 and 411 DF,  p-value: < 2.2e-16
```

Observe the error message. It appears because some fitted values are negative. This can in principle happen, as we use a linear regression. Therefore, we "lose" some observations. In practice, one would fine-tune things now to avoid this. One can do that by using a more flexible specification for the explanatory variables, with interactions, square terms, and so on.

We won't do that now, for the time being.

Finally, we use the stargazer package to show OLS results side-by-side OLS results with robust standard errors and weighted least squares results without and with robust standard errors. For the WLS estimates, in theory, one does not need them if one gets the weighting function right.

```
# Calculate robust standard errors for both OLS and WLS models
robust_se_ols <- sqrt(diag(hccm(estimatesFullModel, type = "hc0")))
robust_se_wls <- sqrt(diag(hccm(wls_model, type = "hc0")))

# Compare OLS with standard errors, OLS with robust standard errors, WLS, and WLS with robust SE
stargazer(estimatesFullModel, estimatesFullModel, wls_model, wls_model,
          se = list(NULL, robust_se_ols, NULL, robust_se_wls),
          column.labels = c("OLS", "OLS (Robust SE)", "WLS", "WLS (Robust SE)"),
          column.separate = c(1, 1, 1, 1),
          type = "text", keep.stat = c("n", "rsq"))
```

```
##
## ========================================================================================
##                                                  Dependent variable:
##                                     ----------------------------------------------------
##                                                      log(price)
##                                       OLS    OLS (Robust SE)    WLS    WLS (Robust SE)
##                                       (1)         (2)           (3)          (4)
## ----------------------------------------------------------------------------------------
## review_scores_rating_standardized 0.050***     0.050***     0.056***     0.056***
##                                    (0.019)      (0.019)      (0.019)      (0.018)
##
## accommodates                      0.216***     0.216***     0.206***     0.206***
##                                    (0.015)      (0.015)      (0.014)      (0.015)
##
## Centrality                         0.086*       0.086**     0.104***     0.104***
##                                    (0.044)      (0.040)      (0.040)      (0.034)
##
## Quietness                         0.129***     0.129***     0.126***     0.126***
##                                    (0.045)      (0.048)      (0.043)      (0.045)
##
## Coolness                            0.096        0.096        0.079        0.079
##                                    (0.078)      (0.073)      (0.074)      (0.071)
##
## Fanciness                        -0.148***    -0.148***    -0.171***    -0.171***
##                                    (0.054)      (0.054)      (0.044)      (0.045)
##
```

```
## Constant                           3.100***     3.100***     3.270***     3.270***
##                                     (0.493)      (0.476)      (0.465)      (0.449)
##
## ------------------------------------------------------------------------------------
## Observations                          421          421          418          418
## R2                                   0.349        0.349        0.372        0.372
## ====================================================================================
## Note:                                               *p<0.1; **p<0.05; ***p<0.01
```

This shows that standard errors are almost unchanged. If anything, they get smaller. This can in principle happen, as it does here. More often, they get bigger.