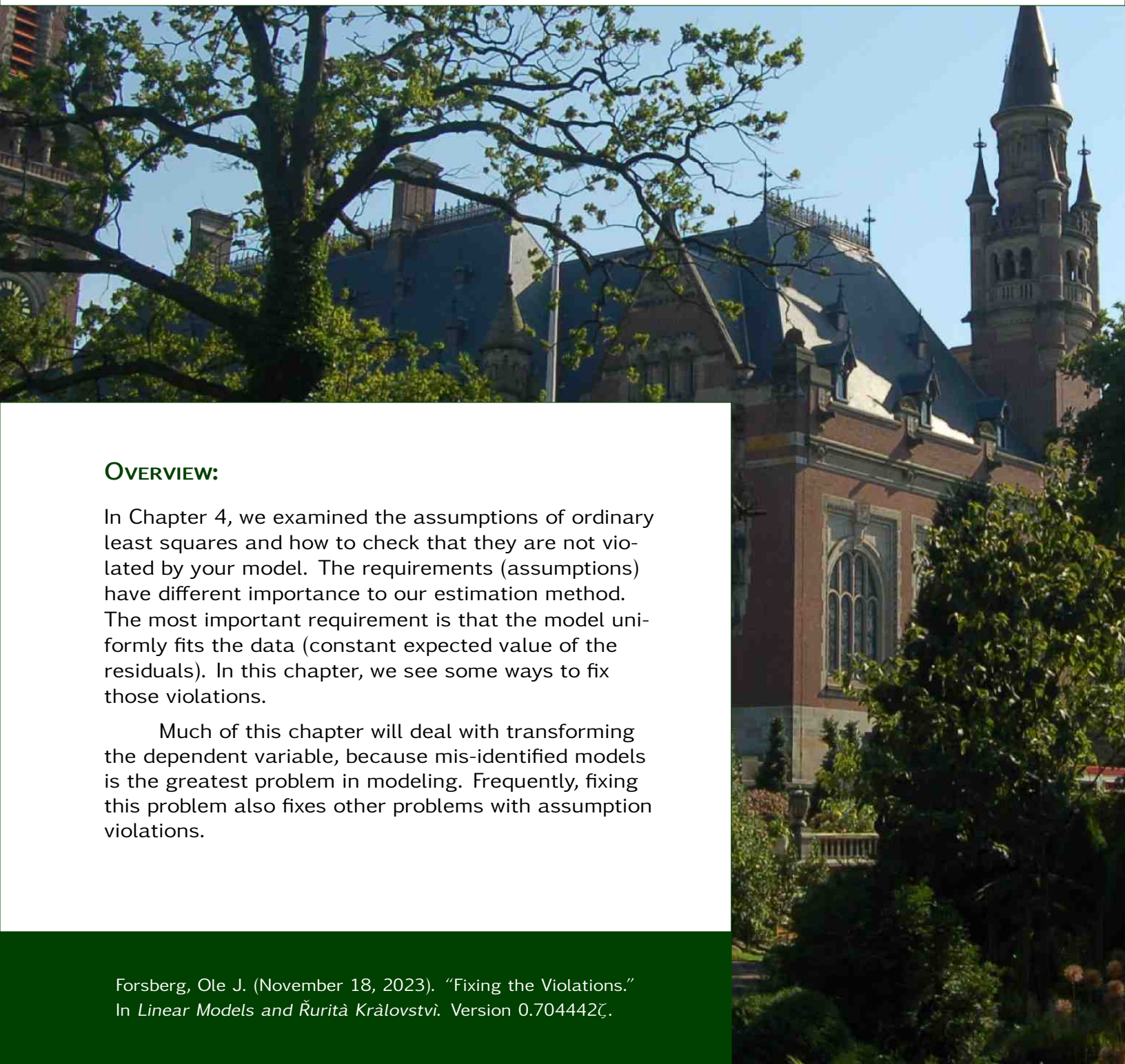
Chapter 6:
Fixing the Violations



Overview:

In Chapter 4, we examined the assumptions of ordinary least squares and how to check that they are not violated by your model. The requirements (assumptions) have different importance to our estimation method. The most important requirement is that the model uniformly fits the data (constant expected value of the residuals). In this chapter, we see some ways to fix those violations.

Much of this chapter will deal with transforming the dependent variable, because mis-identified models is the greatest problem in modeling. Frequently, fixing this problem also fixes other problems with assumption violations.

Forsberg, Ole J. (November 18, 2023). "Fixing the Violations." In *Linear Models and Řurità Kràlovstvì*. Version 0.704442ζ.

Chapter Contents

In the previous chapters, we introduced ordinary least squares (OLS) estimation for the classical linear model (CLM) and its assumptions (requirements). Last chapter, we looked at how to test that the requirements are sufficiently met in our data and model. We also looked at the importance of the assumptions. In this chapter, we determine some methods for dealing with *some* violations of those requirements. Hopefully, this extends the usefulness of this simple and straight-forward estimation method.

The ordinary least squares estimation method (OLS) requires that the error terms have a constant expected value, have a constant variance, and are generated from a Normal (Gaussian) process. But, what happens when these requirements are not met?

There are essentially three ways of handling violations, depending on the type and the severity: First, you can ignore it. Ignoring the violations is usually not *too* bad when you are dealing with predicting within the domain of the observed data, as the increase in bias and the loss of efficiency are usually minor. However, if it is important to estimate parameters, you definitely should not ignore this violation. Furthermore, if the assumption of a constant expected value is practically violated, you *need* to fix it.

Second, we can use other methods (and modeling paradigms) for performing regression. Two popular alternatives to the Classical Linear Model paradigm are the Generalized Linear Model (GLM) and the Generalized Additive Model (GAM). The former paradigm will be covered in Chapters 10 through 14. The latter is well examined in Wood (2006). The strength of these models (and estimation methods) is that they extend the CLM to include (for instance) discrete dependent variables and non-linear relationships (Nelder and Wedderburn 1972; Wood 2006). These unified paradigms allow the computer to estimate the effect coefficients using a very powerful method (called Maximum Likelihood Estimation). The drawback is that not

all problems lend themselves to fitting using Maximum Likelihood Estimation (MLE; Chapter 9). Luckily, most do. Even more luckily, new estimation methods are developed frequently.

However, if we desire to stay within the realm of the classical linear model, estimating the parameters using ordinary least squares, we can fix many violations simply by transforming the dependent variable — especially if the violations are minor.

These transformations are very flexible. Once you get used to working in two different systems of units, you can easily use transformation methods to 'Normalize' many restricted dependent variable. Unfortunately, one cannot transform an arbitrary dependent variable; there are types that cannot be fit using this technique, such as categorical. To handle these types of dependent variables, we will need to introduce a new modeling paradigm (Chapter 10).

two systems

Finally, you can make adjustments to the estimates and their standard errors to "fix" or "adjust for" the violation. This is a common practice in the presence of heteroskedasticity (Section 6.3) and multicollinearity (Section 4.4.2). It does *not* work for violations of model fit (non-constant expected residuals), however.

6.1: The Issue of Boundedness

We finished Chapter 5 with a model of vote proportions for ballot measures concerning keeping cows in the city (Section 5.3). We applied that model to an upcoming vote in Děčín to predict the outcome. Finally, we used Monte Carlo methods to estimate the probability that the ballot measure would pass. In the end, we predicted that the ballot measure had a 20% chance of passing, with a point-prediction of 42% of the voters in favor of the bill.

Results, however, suggest that there may be something gravely wrong with this model (Section 5.3.8). To see this more clearly, let us predict the proportion of voters in support of a hypothetical 1994 ballot measure in Venkovský (religious percent = 85) that also banned chickens (the results table from our Cow-Vote model is in Table 5.4 on page 147).

From the results summarized in the table, the point-prediction for this 1994 Venkovský ballot measure is

$$\hat{p} = 0.1512 + -0.0201(\texttt{yearPassed}) + -0.0373(\texttt{chicken}) + 0.0095(\texttt{religPct}) \tag{6.1}$$

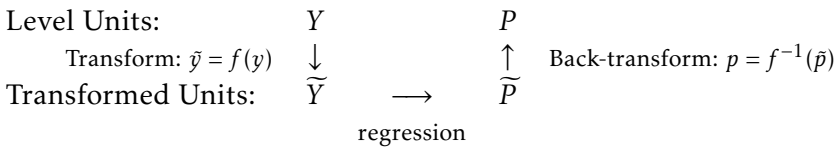


Figure 6.1: *Schematic of a variable transformation procedure, such as described in the text. Here, Y is the original values of the dependent variable, $\widetilde{Y}$ is the transformed values of the dependent variable, $\widetilde{P}$ is the result from the regression in transformed units, and P is the result in the original (level) units.*

$$= 0.1512 + -0.0201(-6) + -0.0373(1) + 0.0095(85) \tag{6.2}$$

$$= 1.0379 \tag{6.3}$$

Thus, this model predicts that the ballot measure will pass with over 103% of the vote — a physically impossible outcome. What went wrong? How can we fix this model so that this cannot happen?

First, nothing "went wrong," *per se*. The model did *exactly* what it was supposed to do. The prediction, however, is based on assuming the effect (slope) is *constant*. If the slope is constant, one can find large enough (or small enough) values for the independent variables to make the prediction arbitrarily large or small. When we are predicting a bounded dependent variable, this will necessarily lead to an impossible prediction, such as a 103.79% support rate. Thus, the issue is with the linear (constant slope) aspect of the prediction equation *or* with the bounded nature of the dependent variable (bounded below by 0 and above by 1).

Thus, to improve the model, we can either model using non-linear coefficient functions (Chapter 10) or eliminate the boundedness. At this point, the easier of the two is to eliminate this boundedness; that is, we need to change the dependent variable so that *all* values make physical sense. This is done through the process of variable transformation. There are three steps: First, transform the dependent variable from a restricted range to an unrestricted range. Second, perform the analysis on this transformed variable. Finally, back-transform the estimated values (not estimated effects) into the original units. The overview of this plan is shown in Figure 6.1.

The key is the transformation. It must change the range of Y from its current limited version to an unlimited version, denoted $\widetilde{Y}$. Luckily, there are two transformations that take care of most of our needs, in general: the logit (*LOH-jit*) and the logarithm transformations.

6.1.1 Data Bounded by 0 and 1

One type of data you may come across in your research is proportion data, data where the values are bounded below and above (by 0 and 1, respectively); that is, if Y is the dependent variable, then $0 < Y < 1$. One appropriate function that transforms this bounded domain into an unbounded range is the logit function:

$$\tilde{y} = \text{logit}(y) := \log\left(\frac{y}{1-y}\right) \tag{6.4}$$

The logit function transforms (maps) variables bounded by 0 and 1 into unbounded variables; in symbols,

$$\text{logit} : (0,1) \mapsto \mathbb{R} \tag{6.5}$$

The logit's inverse, which maps it from logit units back into level units is called the logistic function:

$$y = \text{logistic}(\tilde{y}) := \frac{1}{1+\exp(-\tilde{y})} = \frac{\exp(\tilde{y})}{1+\exp(\tilde{y})} \tag{6.6}$$

The logistic function transforms unbounded variables into variables bounded by 0 and 1:

$$\text{logistic} : \mathbb{R} \mapsto (0,1) \tag{6.7}$$

Other transforms are available, but the logit is frequently used for the following three reasons:

1. The transformation *and* its inverse are both functions (the transform is a bijective function). This means that the results are always commensurate to the original problem.
2. The transformation is symmetric. This means that the 'stretching' is the same for values near 0 as they are for values near 1.
3. The function is exact, as opposed to the probit transform which requires numerical approximations. This increases the speed and accuracy of your predictions.

A careful reader will note that the domain of Y includes neither 0 nor 1. This is because there is no way of transforming a closed (or a half-closed) interval into an open interval such as $\mathbb{R}$ while ensuring that the inverse is also a function. This is a provable fact of mathematics (Strichartz 2000).

But, what do we do if there are y-values that *are* zero or one? One solution is to add (subtract) an extremely small number, δ, to the zero (one). A second solution is to completely drop those data from the analysis.

Delta Adjustment

> ***Note***: None of these solutions is perfect. *If* you insist on using linear regression, then you should use both methods and see how much your answer changes. A general rule of thumb is that if your underlying research model is correct then the results should not vary wildly based on similar models. That is, if we know Y depends on X_1 and X_2, then all appropriate modeling techniques should give *approximately* the same results. If they do not, then there is something seriously wrong with our assumptions about the underlying relationships — the model.

rule of thumb

A third solution is to change the proportion into a bounded count and use a different paradigm (Chapter 12). While this is the best option, it requires more background before we can cover it.

Example 6.1: Let us return to the `cows` data file and the example of Section 5.3. The voters of Děčín are being sent to the polls to vote on a constitutional referendum that proposes to limit the number of cows kept in the city. This was not the first time that Ruritanians were sent to the polls to vote on this or a closely related issue. Given the information from previous votes, what is the estimated probability that this ballot measure will pass in Děčín?

•

Solution: Let us now answer this question more correctly. Recall that without performing a transformation of the dependent variable, there existed predictions which fell outside possible reality. To fix this, let us transform the dependent variable using the logit function, repeat the analysis, back-transform these transformed results to the original units, and compare results.

steps

	Estimate	Std. Error	t-value	p-value
Constant Term	-1.8909	0.2898	-6.53	$\ll 0.0001$
Year Passed (after 2000)	-0.0885	0.0157	-5.64	$\ll 0.0001$
Contains a Chicken Ban	-0.2318	0.0878	-2.64	0.0134
Percent Religious in Kraj	0.4750	0.0047	10.06	$\ll 0.0001$

Table 6.1: *Results table of the results of regression on the dependent variable, using a logit transformation of the dependent variable.*

The first step is to transform the dependent variable. As the dependent variable is a proportion, let us use the `logit` transform (from the `RFS` package). If we decide to call the new variable `logitWin`, then the command will be

```
logitWin = logit(propWin)
```

Now, this is our new dependent variable. As such, we perform the same analysis as in Chapter 5:

```
modLgt = lm(logitWin ~  yearPassed + chickens + religPct)
```

The `summary(modLgt)` command provides the results summarized in Table 6.1. Note that all three independent variables are more statistically significant than in the non-transformed model, Table 5.4. Also note that the effect directions are the same as before. ♦

How shall we interpret the results? There are a few ways. The graphic is the best. However, an older manner relies on the "log odds ratio." The odds ratio is frequently used to illustrate the strength of the association between two variables. For every increase of 1 in the percent religious in Kraj, the log of the odds of the vote passing increases by 0.4750. Said another way, the odds of it passing increases by approximately $\exp 0.4750 = 1.6080$ for each increase of 1.

An increase of 2 percent religious increases the odds by $\exp(2 \times 0.4750) = 2.5857$. [As an aside, this is also the same as 1.6080^2.]

Note: Beyond this, one *cannot* directly compare the magnitudes of these coefficients with the magnitudes of the previous coefficients; these effect

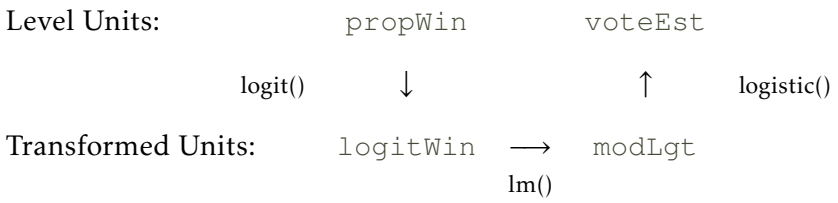


Figure 6.2: *Schematic of the variable transformation procedure used in Example 6.1. Note that the results table, Table 6.1, displays the coefficients of* `modLgt`*, which is in the transformed units, not the original units. As such you cannot compare these magnitudes with the magnitudes in Table 5.4.*

estimates are in different units. The coefficients seen in Table 5.4 predict in the original units (proportions). The coefficients in Table 6.1 predict in logit (of proportions) units. Furthermore, merely taking the logistic of the coefficients will not put them in level units; the transform is non-linear, as we designed, thus the effect of *any* depends on the values of *all*. In order to compare the two models, we need to perform predictions (remembering to back-transform them). Refer to Figure 6.2 for the steps we use in this particular example.

Predicting the proportion of the vote for the Děčín ballot measure is almost as easy as it was before. The only additional step is that we need to back-transform the prediction to get it in proportion units.

So, according to this transformed model, what is the expected vote in Děčín? To answer this, we need the Děčín information: `yearPassed` = 9, `chickens` = 0, `religPct` = 48. With this information, and under the assumption that the model is correct, we have our prediction of −0.4091 logits. Back-transforming this value gives a prediction of logistic(−0.4091) = 40% of the population will vote in favor of this ballot measure — just slightly different from our original prediction of 42%.

```
DECIN = data.frame(yearPassed=9, chickens=0, religPct=48)
voteLgt = predict(modLgt, newdata=DECIN)
voteEst = logistic(voteLgt)
```

However, remember that the original question was not this point estimate, it was a *probability* of the ballot measure passing. To determine this probability, we just need to repeat the same steps as we did answering this question before (Section 5.3.7), but remembering to back-transform the results.

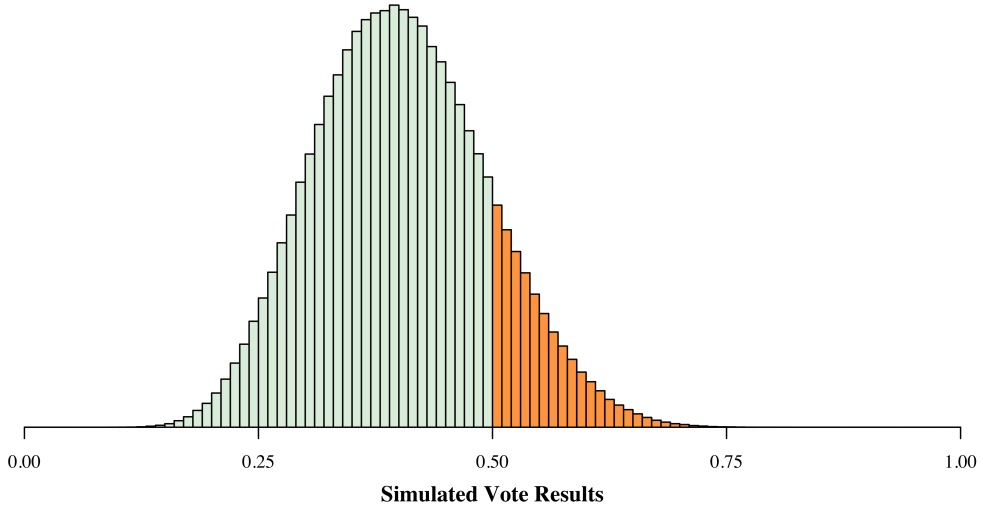


Figure 6.3: *Histogram of the results of the Monte Carlo experiment described in the text. Note that the distribution has a slight right-skew as a result of the transformation process. Also note that there are no predicted vote outcomes less than 0 or greater than 1, as compared to the original untransformed model of Section 5.3.8. In fact, the lowest prediction is 9.0%, while the largest is 81.6%.*

The Monte Carlo results of the transformed model indicate that there is a 15% chance that the ballot measure will pass in Děčín. The histogram of a million predictions is presented as Figure 6.3. From this information, we can conclude that there is a definite possibility that the cow ballot measure will pass in Děčín (15%), with a predicted 40% vote in favor.

If we were into betting, we could also conclude that this model predicts that the odds of this ballot measure passing is $\frac{1.00-0.15}{0.15}$, 5.67-to-1 *against*.

Thus, a 'fair' bet would pay \$5.67 for every \$1.00 bet in favor of the ballot measure and \$1/5.67 = \$0.176 for every dollar bet against the ballot measure passing.

Regardless, since the probability of the measure passing is 15%, a pass would not be wholly unexpected. Its probability is more likely than flipping a fair coin three times and having it come up heads all three times — definitely not unheard of.

The 95% prediction interval for the Děčín referendum outcome, according to our model, is from 23.5% to 59.0%. The observed value of 53% is well-within that interval.

Note: From this past discussion, we were able to estimate success probabilities and fair betting odds. This is yet another use of statistical modeling.

Note that we are estimating the probability of an event. Unless that probability is 0 or 1, there is always a chance the event will (or will not) happen. Thus, the passing of the Děčín referendum in 2009 does not directly detract from our model. There was a 15% chance it would pass, according to our model.

Warning: *Stay aware of what statistical model says* and does not say — *the choice is humility or humiliation.*

6.1.2 Data Bounded Below by 0

When the dependent variable represents a proportion (bounded by 0 and 1), we can use the logit function to transform it into an unbounded variable, perform the usual analysis, and back-transform those results into level units (the previous section). However, not all bounded variables fit this bounding, e.g., age, height, income. These variables are bounded below by 0 and have no theoretical upper bound. For such variables, we may want to use the logarithm transform.[1]

The logarithm function transforms variables bounded below by 0 into unbounded variables; in symbols, $\log : (0,\infty) \mapsto \mathbb{R}$. Its inverse is the exponential function, $\exp : \mathbb{R} \mapsto (0,\infty)$. Both functions are bijections and strictly increasing and so are appropriate functions for transforming our variables.

Note that values of 0 are problematic for the logarithm in much the same way that values of 0 and 1 were problematic for the logistic function. Solutions are similar (Section 6.1.1, page 164).

Example 6.2: The gross domestic product (GDP) per capita is one of many measures of average wealth in countries. If extant theory is correct, then the wealth in the country is directly affected by the level of honesty in the government — countries with high levels of honesty (low levels of corrup-

[1] By "theoretical upper bound," we mean there exists a limit (a single value) such that the variable can get sufficiently close to *that* limit, but no greater.

tion) should be wealthier than those with low levels of honesty (high levels of corruption). Furthermore, if theory is correct, the level of democracy in a country should *also* influence the country's level of wealth — countries with higher levels of democracy should be wealthier than countries with low levels of democracy.

Let us determine if reality (in the form of the data in the `gdp` data file) supports the current theory or if current theory needs to explain the severe discrepancies. Furthermore, let us estimate the GDP per capita for Ruritania and provide a 95% confidence interval for that estimate. •

Solution: For this section, recall that the level of honesty in government for Ruritania is 5.1 and the level of democracy is -7. With that information, I leave it as an exercise for you to model the data *without* transforming the dependent variable and discovering the predicted GDP per capita for Ruritania is $26,795.64. This seems awesome for Ruritania. The 95% prediction interval is from $5232 to $48,360. That's rather wide. It is a function of the high level of variation in the data.

However, to see a problem with the model, let us estimate the GDP per capita for Papua New Guinea (democracy=10, hig=2.1). According to the model, the predicted GDP per capita is -$2337, which is not physically possible. If nothing else, this prediction should suggest to you that the data needs transformation before being modeled.

The process to estimate the GDP per capita in Ruritania using a transformed model is formulaic for us by now: transform the dependent variable by applying the logarithm function, model the transformed variable, estimate in the transformed units, back-transformed into level units — here, dollars.

One feature of `R` that is shared by few other statistical packages is that you do not have to actually create a new variable; you can perform the transformation within the modeling command; *e.g.*,

```
modLog = lm(log(gdpcap) ~  democracy + hig)
```

The results table for this model is provided in Table 6.2. Again, as we have transformed the dependent variable, the coefficients are not in units of dollars. As such, their magnitudes cannot be directly compared to those in the untransformed model. Their *directions*, however, can be compared because the transformation we used was strictly increasing. Thus, this model tells

us that higher levels of honesty in government correspond to countries with higher GDPs per capita (in this sample). Additionally, countries with higher democracy scores correspond to countries with *lower* GDPs per capita (in this *sample*).

The first finding is so strong in this sample that we can conclude that there is evidence of this relationship *in the population*. This second finding, which conflicts with current theory, is not statistically significant at the usual $\alpha = 0.05$ level. Thus, we cannot conclude that the effect in the population is negative, positive, or null (zero). All we can conclude is that we did not detect an effect *with this data*. Whether this is due to a lack of effect in the population, the sample selected, the sample size, no one can tell.

With this model, we can estimate the GDP per capita in Ruritania using the standard method, but remembering that we must back-transform the final estimate. That is, if we used the commands

```
RUR = data.frame(hig=5.1, democracy=-7)}
estLog = predict(modLog, newdata=RUR)
```

then we would report Ruritania's GDP per capita as an estimated value of \$11,508 (using `exp(estLog)`). ♦

Note: From your mathematics course, you may recall that $\log(1+x) \approx x$ for small values of x. This means we can interpret the coefficients in the log-model as percent increases/decreases. For instance, the coefficient for the level of democracy in the country is -0.0028. We can interpret this as "one increase in the level of democracy decreases the GDP per capita by 0.28%, on average." The coefficient of the level of honesty in government is 0.4702. We could interpret this as "one increase in the level of honesty

	Estimate	Std. Error	t-value	p-value
Constant term	6.9333	0.1479	46.89	$\ll 0.0001$
Level of Democracy	-0.0028	0.0113	-0.25	0.8055
Honesty in Government	0.4702	0.0359	13.11	$\ll 0.0001$

Table 6.2: *Results table for the GDP per capita modeling exercise. As the model is a transformed model, these effects estimates are not in units of dollars.*

in the government increases the GDP per capita by approximately 47%, on average."

However, what do we mean by "small values of x"? Anything less than 0.2 is usually fine. Our interpretation of the honesty-in-government coefficient probably should not have been done. A log-coefficient value of 0.4702 really corresponds to a percent increase of only 38.5%. It is more accurate, but less spiffy.

Here is my code to explore the relationship $\log(1+x) \approx x$:

```
x = seq(0,1, length=1e4)
y = log(1+x)
plot(x,y, col="blue1")
abline(0,1, col="orange")
```

❧ ❧ ❧

The question asked us to calculate the estimate, but to also provide a 95% confidence interval. One way of doing this is to use Monte Carlo methods. The steps are all the same, with the additional step of back-transforming the estimates (last line).

```
b.int   =  6.933298
b.dem   = -0.002776
b.hig   =  0.470225

s.int   = 0.147873
s.dem   = 0.011253
s.hig   = 0.035855

e.int   = rnorm(trials, m=b.int, s=s.int)
e.dem   = rnorm(trials, m=b.dem, s=s.dem)
e.hig   = rnorm(trials, m=b.hig, s=s.hig)

outcome = e.int + e.dem*-7 + e.hig*5.1
est     = exp(outcome)
```

The assignments in the second and third group are the coefficient estimates and standard errors from the model (Table 6.2). The histogram of these results are provided in Figure 6.4. To calculate a 95% confidence interval, we merely find the values of `est` for which 2.5% and 97.5% of the data are less.

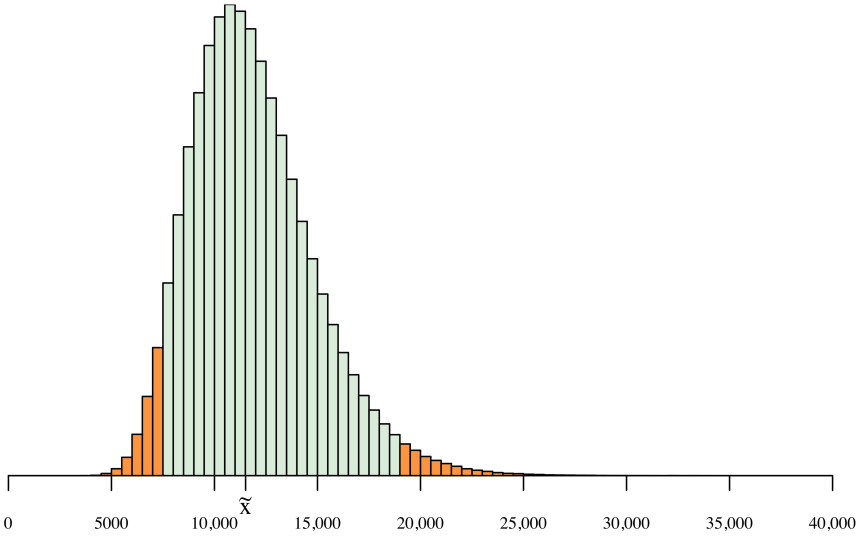


Figure 6.4: *Results of the Monte Carlo experiment estimating the GDP per capita for Ruritania and its 95% confidence interval. Note that 5% of the estimates fall in the rejection (tan) region, 2.5% above and 2.5% below. The median of this distribution is designated by $\tilde{x}$.*

```
quantile(est, c(0.025,0.975))
```

From this, we can conclude that our model estimates the GDP per capita for Ruritania is $11,508, with a 95% confidence interval being from $7075 to $18,733. It is interesting to note that the actual GDP per capita in Ruritania is $55,000, which is well above our confidence interval. Thus, our question is this: Is our model that weak, or is Ruritania doing that well?

> ***Note***: Here, I use the original estimate as the point estimate for the GDP per capita of Ruritania ($11,508). It would have also been appropriate to use the mean of the Monte Carlo trials ($11,870) *or* the median of the Monte Carlo trials ($11,510). All three are acceptable measures of the center. It is usual, however, to use the original prediction.

Here is an interesting question. In the previous example, we estimated a confidence interval. How could we estimate a prediction interval?

To answer this, we need to remember the only difference between confidence and prediction intervals. In a confidence interval, we are estimating an expected value. In a prediction interval, we are predicting a new outcome. That new outcome is a combination of the expected value *and* the σ^2 from the ε term.

And so, to get a prediction interval, we use the following. Check to see the difference between this and the previous script.

```
b.int   =  6.933298
b.dem   = -0.002776
b.hig   =  0.470225
b.err   =  0

s.int   = 0.147873
s.dem   = 0.011253
s.hig   = 0.035855
s.err   = 0.8841

e.int   = rnorm(trials, m=b.int, s=s.int)
e.dem   = rnorm(trials, m=b.dem, s=s.dem)
e.hig   = rnorm(trials, m=b.hig, s=s.hig)
e.err   = rnorm(trials, m=b.err, s=s.err)

outcome = e.int + e.dem*-7 + e.hig*5.1 + e.err
est     = exp(outcome)
```

From this, the 95% prediction interval is from \$1907 to \$69,345. Note that it is *much* wider than the confidence interval. Also note that this should not surprise us at all. Prediction intervals are always wider than the corresponding confidence interval.

6.1.3 Additional Bounds

Thus far, we have looked at transformation of a dependent variable when it is bounded above and below by 0 and 1 (two bounds), and when it is only bounded below by 0 (one bound). Other bounds are possible.[2] In this section, we figure out how to handle all types of bounds. The basic steps are to determine if the variable is bounded on one side or two. If one, then perform an *algebraic* transformation so that the new variable is

[2]While other bounds are possible, the number of bounds can only be 0, 1, or 2. This makes this section so important.

bounded below by 0, then use the log transform. If two, then perform an algebraic transformation so that the new variable is bounded by 0 and 1, then use the logit transform. In either case, you will need to remember to back-transform the predictions with this algebraic transformation.

> ***Note***: The only bounds I frequently come across in my own research are those bounded by 0 and 1, bounded by 0 and 100 (percentages), bounded by 0 and 4 (GPAs), and bounded below by 0. The quick solution for percentages is to divide them by 100 to make them proportions, then multiply the predictions by 100 to turn the predictions back into percentages.

Bounded by L and U: What if our data has a theoretic lower bound L and a theoretic upper bound U? As it is bounded above *and* below, we will change it into a proportion and using the logit transform as in Section 6.1.1, remembering to back-transform with the additional transformation. The algebraic transformation is

$$a(y) = p = \frac{y - L}{U - L} \tag{6.8}$$

The back-transform is

$$a^{-1}(p) = y = p(U - L) + L \tag{6.9}$$

Example 6.3: The scores on the quantitative portion of the Graduate Record Examination (GRE) range from $L = 200$ to $U = 800$. If we wished to properly model a person's GRE quantitative score, we would first subtract 200 from each score, then divide by $800 - 200 = 600$. The new variable would range from 0 to 1, a proportion. •

Example 6.4: The grade point averages (GPAs) are bounded below by $L = 0$ and above by $U = 4$. To appropriately model GPAs, we would have to subtract 0, then divide by 4. This new variable would now be a proportion. •

Bounded Below by L: It may be that your dependent variables is bounded below by a specific value, L, but not bounded above. As it is bounded on only *one* side, we will transform it into a variable bounded below by 0 and then apply the logarithm transform as in Section 6.1.2, remembering to back-transform with the additional transformation. The algebraic transformation is

$$a(y) = p = y - L \tag{6.10}$$

The back-transform is

$$a^{-1}(p) = y = p + L \tag{6.11}$$

Example 6.5: Hourly workers make at least \$7.25 per hour. To model excess hourly wage, we would subtract off $L = 7.25$ from each hourly wage. This new variable is bounded below by 0, so we can apply the log transformation to it. •

Bounded Above by U: It may be that your dependent variable is theoretically bounded *above* by U. As there is only *one* bound, we will perform an algebraic transformation so that it is bounded below by 0 and then apply the log transform as in Section 6.1.2, remembering to back-transform with the additional transformation. The algebraic transformation is

$$a(y) = p = U - y \tag{6.12}$$

The back-transform is

$$a^{-1}(p) = y = U - p \tag{6.13}$$

Example 6.6: In the ocean, different species live at different depths. In fact, we can predict the depth based solely on the species observed. Ocean depth is bounded above by 0 and has no theoretic lower bound (although it certainly has a genuine lower bound at the Challenger Deep in the Mariana Trench, which has a depth of -35,994 ft). To transform the depths into a variable upon which we can perform a log transform, we subtract each value from $U = 0$. After we predict, we will have to back-transform by again subtracting each prediction from $U = 0$. •

Of course, the transformation in this last example is equivalent to measuring depth in terms of 'distance below the surface', which is a positive number requiring no additional transformation.

6.2: Full Example: The South Sudanese Referendum

Free and fair elections are one of the requirements for a legitimate democratic system; furthermore, being a legitimate democratic State is necessary for some forms of external assistance. As such, many not-so-democratic States wish to appear democratic. They hold elections, but the elections are either fraudulent or the electoral system (rules governing the elections) is unfair.

There are many definitions for fairness in an election, but they all contain the same requirement that a person's vote has the same probability of being counted as anyone else's. In other words, the probability of a vote being invalidated is independent of the characteristics of the person casting the vote — including who the vote was for. This aspect of fairness can actually be tested in elections where the number of invalidated votes is counted: If the proportion of the vote for a specific candidate or position is not independent of the proportion of the vote invalidated in the electoral division, then there is evidence against the assumption of fairness.

Does the 2011 independence referendum in southern Sudan indicate an issue with fairness?

Narrative Solution: As one of the conditions to the 2005 Naivasha Agreement, which ended the civil war in Sudan, the South was allowed to vote on independence from the North. That referendum was held January 9–15, 2011. Official results stated that 98.83% of the South Sudanese voted against unity and in favor of independence.

The `xsd2011referendum` data contains the number of votes in favor of independence (`Secession`), the number of votes declared invalid (`Invalid`), and the total number of votes cast (`Votes`). Load it and save it into the `xsd` variable without attaching the data. Because we need to determine if there is a (linear) relationship between the proportion of the vote for a specific side and the proportion of the vote invalidated in the electoral division, and because we just have vote counts, we need to create those proportions. The proportion of the vote for the candidate is the number of votes

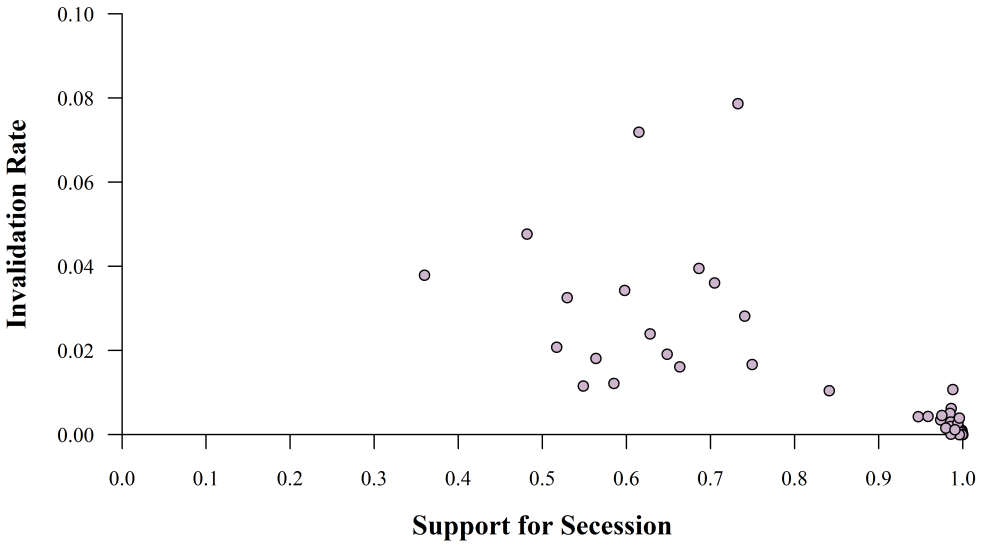


Figure 6.5: *A scatterplot of the results of the 2011 referendum on independence for South Sudan. Note the apparent presence of a relationship between these two variables. As such, there appears to be evidence that the election was not fair for those voting against independence.*

for the candidate divided by the number of valid votes. The invalidation rate is the number of invalid ballots divided by the number of cast ballots (recall Section 6.1.3).

Once that is done, we need to transform these proportions using the logit transformation, perform linear regression, and check for a relationship. If one exists in the transformed variables, then one exists in the untransformed variables. First, however, it is always a good idea to plot the variables to see if there is an obvious answer to the question. Figure 6.5 a the plot of proportion of the vote invalidated against the proportion of the vote in favor of independence.

Suggested by the plot, there appears to be a strong relationship between the two variables, evidence of an election that is not fair. Because of the direction of the slope, it appears as though those areas voting most strongly in favor of independence had a much lower probability of having their votes rejected.

Note: As we are using the logit transform, we must drop any electoral division (here, county) which has zero invalid votes or zero votes in favor of secession. We need to do this because the domain of the logit function is $p \in (0,1)$.

To easily do this in `R`, we can use the `which` function, which determines which entries have the provided condition. Thus,

```
dr = which(xsd$Invalid==0)
```

returns a vector of values $\{15, 19, 23, 24, 28, 46, 47, 49, 50, 57, 72, 73\}$. These numbers correspond to the counties that had zero invalid votes cast. Storing this vector in the variable `dr` allows us to remove those counties from any subsequent calculations. As such, our proportion calculations are:

```
p.ind = xsd$Secession[-dr]/xsd$Votes[-dr]
p.inv = xsd$Invalid[-dr]/xsd$Votes[-dr]
```

The negative signs tells `R` to return values in the vector *other than* these entries.

And so, the two lines to transform the dependent variable and fit the OLS model are

```
l.inv = logit(p.inv)
model.xsd = lm(l.inv ~  p.ind)
```

The results of the linear regression on the transformed dependent variable are given in Table 6.3. There is a very strong relationship between the proportion of the vote invalidated in the county and the proportion of the vote in favor of secession: Those counties with a greater proportion of people voting for independence also had a lower proportion of the vote invalidated. That there is a strong relationship between these two variables is troubling.

To make this relationship more obvious, and to make our point stronger, we can plot the data, the prediction curve, and the 95% Working-Hotelling confidence bands on the same plot.

	Estimate	Std. Error	t-value	p-value
Constant term	1.8978	0.7690	2.468	0.0155
Proportion of Vote for Independence	-9.3991	0.8287	-11.342	$\ll 0.0001$

Table 6.3: *Results table for the South Sudan referendum. The results are in logit units. Note the high level of statistical significance in the effect of the proportion of the vote in favor of independence. This is very indicative of a lack of fairness in the election.*

> ***Note***: What confidence intervals are to univariate data, confidence bands are to bivariate data. We briefly saw the Working-Hotelling confidence bands in Section 3.4.

6.2.1 Graphing Philosophy of R

In R, the philosophy behind graphing is to start with a fresh plot and paint successive layers on top of it. This allows us to create graphs that tell the story and to do so easily. To make the graph described above, we need to

1. Plot the points (displayed in proportion units),
2. Plot the prediction curve (displayed in proportion units, but calculated in logit units),
3. Plot the 95% confidence bands (displayed in proportion units, but calculated in logit units).

The first step has been done already (Figure 6.5).

The second step requires the repeated use of the `predict` function. First, to make things easier, let us define `newX` as a series of "proportion of vote in favor of independence" values for which we want to make predictions: `newX = seq(0, 1, length=1e4)`. This creates a vector containing 10,000 values equally spaced between 0 and 1.

With this, our predict statement will be

```
l.pred = predict(model.xsd,
     newdata=data.frame(p.ind=newX),
     se.fit=TRUE)
```

Note: The `se.fit=TRUE` parameter, which calculates the standard error of the fit at that x-value, will be important for calculating the confidence bands. This is just a courtesy from `R`, as we know how to calculate this value from Theorem 3.14.

Remember that these predictions are in logit units. To get them into level units, we just apply the logistic function to these point predictions:

```
p.pred = logistic(l.pred$fit)
```

Note: The `$fit` selects only the fitted predictions from the `l.pred` variable. This is necessary as we are also using the `se.fit=TRUE` parameter.

Now that we have the predictions in the original units, we merely paint it on the current plot (from Step 1):

```
lines(newX, p.pred)
```

The third step requires us to calculate the 95% confidence bands and paint them on the plot as well. For want of better estimates, let us use the Working-Hotelling bands (Section 3.4). The formula to calculate the upper 95% confidence bands is

```
ucb.l = l.pred$fit+W*l.pred$se.fit
```

the lower,

```
lcb.l = l.pred$fit-W*l.pred$se.fit
```

Here, $W = \sqrt{2F(1-\alpha, 2, n-2)}$, which translates to

```
W = sqrt( 2 * qf(1-0.05, 2, n-2) )
```

Note: The form of these formulas should look vaguely familiar. They are of the same form as when we calculated the upper and lower limits for

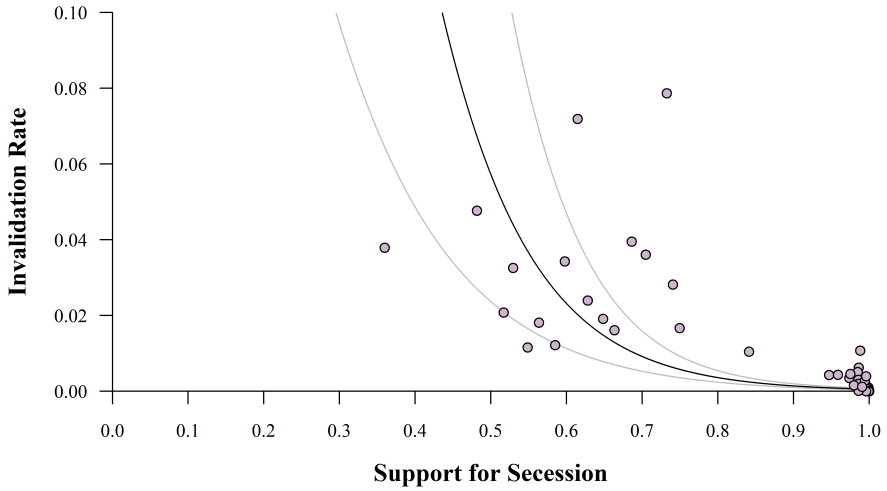


Figure 6.6: *A plot of the results of the South Sudan referendum. Included are the prediction line (in black) and the 95% confidence bands (in grey). Note that a horizontal line cannot fit between the confidence bands. This indicates a statistically significant relationship between the proportion of the votes invalidated and the proportion of the votes in favor of independence. This, in turn, supports the conclusion of an unfair election.*

Normal confidence intervals,

$$u = \overline{x} + 1.96 s_x \qquad \text{and} \qquad l = \overline{x} - 1.96 s_x$$

The W distributional multiplier comes from Working-Hotelling (1929) and its Scheffé extension (1959).

Once again, we must back-transform these two variables using the logistic function. So, our final confidence bands are

```
ucb = logistic(ucb.l)
lcb = logistic(lcb.l)
```

Finally, we paint this on the current plot with

```
lines(newX, ucb, col="grey")
lines(newX, lcb, col="grey")
```

Putting all this together gives us Figure 6.6. Note that the predictions are curved in these units; they are straight in logit units. Also note the confidence

bands are wider where the value of x is farther from $\overline{x}$ (Theorem 3.14). Lastly, note that no horizontal line can fit between the two confidence bands. This illustrates that there is a statistically significant relationship between the two variables at the $\alpha = 0.05$ level. (Why?) It says the same thing as Table 6.3, but in a graphical manner. Graphs often makes the points more manifest.

6.3: Heteroskedastic Adjustments

The above transformations also work well on fixing problems with heteroskedasticity and non-Normality. Unfortunately, if you perform an appropriate transformation to fix the problem with model fit, further transformations to fix heteroskedasticity may end up creating a new problem with model fit.

Thus, it may happen that you cannot find a way of fixing the heteroskedasticity without breaking something else. In such cases, we can adjust the standard errors using a technique introduced by White in 1980.

Recall from ordinary least squares estimation (page 31) that our estimator for $\mathbf{B}$ is

$$\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \tag{6.14}$$

From this we showed in Theorem 2.10 that this estimator was unbiased; that is, $\mathbb{E}[\mathbf{b}] = \mathbf{B}$. In Theorem 2.11, we also calculated the variance of the estimator as

$$\mathbb{V}[\mathbf{b}] = \sigma^2(\mathbf{X}'\mathbf{X})^{-1} \tag{6.15}$$

That result, however, required that $\mathbb{V}[\mathbf{Y}] = \sigma^2\mathbf{I}$. This is the assumption of homoskedasticity (Section 2.2.2). Under heteroskedasticity, $\mathbb{V}[\mathbf{Y}]$ cannot be reduced. This leaves the variance of our OLS estimators as

$$\mathbb{V}[\mathbf{b}] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \;\; \mathbb{V}[\mathbf{Y} \mid \mathbf{X}] \;\; \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \tag{6.16}$$

So, to better estimate $\mathbb{V}[\mathbf{b}]$, we need to estimate $\mathbb{V}[\mathbf{Y} \mid \mathbf{X}]$ from the data. (Everything else in Equation 6.16 is known.) How do we estimate $\mathbb{V}[\mathbf{Y} \mid \mathbf{X}]$ from the data? We recall that $\mathbb{V}[\mathbf{Y} \mid \mathbf{X}] = \mathbb{V}[\mathbf{E}]$. Thus, we estimate $\mathbb{V}[\mathbf{Y}]$ from the residuals, specifically from how large each residual is. If the i^{th} residual is large, then the value of $\mathbb{V}[Y_i]$ will be large; if e_i is small, then $\mathbb{V}[Y_i]$ will be small.

6.3.1 Having R Do This for Us

Instead of performing the above calculations by hand, we can have R do the adjustments for us. That helps with

	Estimate	Std. Error	t-value	p-value
Constant term	1.8978	0.6704	2.831	0.0045
Proportion of Vote for Independence	-9.3991	0.7420	-12.667	$\ll 0.0001$

Table 6.4: *Results table for the South Sudan referendum using White (heteroskedastic-consistent) standard errors. Compare this to Table 6.3. The results are in logit units. Note the high level of statistical significance in the effect of the proportion of the vote in favor of independence. This is very indicative of a lack of fairness in the election.*

the accuracy and precision. The `summaryHCE` function in the `RFS` package provides the adjustment and presents it in the form of our usual regression table.

Example 6.7: To illustrate the operation of the `summaryHCE` function, let us calculate the White-adjusted standard errors for the South Sudanese model above. •

Solution: We have already calculated the regression table for the logit model (Table 6.3). From looking at the graphic, it seems as though there may be heteroskedasticity. There appears to be a lot more variation in the invalidation rate for smaller values of secession support than for larger values of secession support.

Running the following adjusts the standard errors to reflect the observed heteroskedasticity.

```
summaryHCE(model.xsd)
```

The heteroskedasticity-adjusted regression table is given in Table 6.4. Note that the estimates remain the same. That is because heteroskedasticity does not affect the estimates. The only changes are in the standard errors (and the test statistics and the p-values). ♦

Notice that adjusting the standard errors is rather easy using `R`. It is just a single line. Also notice that we did not model the heteroskedasticity, we merely adjusted for it.

At some level, it is unsettling to adjust for model weaknesses. It is a strong model that does not need fixes. Thus, if you can avoid using White standard errors, I recommend it strongly. Heteroskedasticity is an important part of the data/model. It seems sinful to ignore it.

apple

6.4: Conclusion

In this chapter, we focused on transforming bounded variables so that they did not violate the Normality assumptions as strongly as they did without the transformation. To accomplish this, we noted that there are three basic types of continuous variables: unbounded, bounded on one side, and bounded on two sides. If the dependent variable is unbounded, we do not necessarily need to transform it (although some transforms may reduce the non-Normality of the residuals). If the variable is bounded on one side, we performed an algebraic transformation so that it is bounded below by zero, then applied a log transformation. If the variable is bounded on two sides, we performed an algebraic transformation so that it was bounded by 0 and 1, then applied a logit transformation.

In either case, we needed to ensure that we back-transformed to the original units, first using an exponential or a logistic back-transform, then the inverse of our algebraic transform — order matters.

While this chapter does not exactly mark the end of continuous dependent variables, it does end our view of them in terms of the Classical Linear Model (CLM). This chapter already shows why the CLM needs to be replaced. Here, we were able to stay within the framework, but we had to perform variable transformations to make it work. Once we stray from continuous data, the CLM cannot work; there is no way of transforming a discrete dependent variable into a Normally distributed random variable. As such, we need an new paradigm — Generalized Linear Models (GLMs). The next chapter introduces GLMs, while still using a continuous dependent variable. This is done to show that GLMs can do anything CLMs can do. In fact, if you had used the `glm` function in this and the previous chapter, in lieu of the `lm` function, the results would be *exactly* the same, only the table layout would be different.

6.5: End-of-Chapter Materials

6.5.1 R Functions

In this chapter, we were introduced to several R functions that will be useful in the future. These are listed here.

Packages:

RFS This package does not yet exist. It is a package that adds much general functionality to R. In lieu of using `library(RFS)` to access these functions, run the following line in R:
`source("http://rfs.kvasaheim.com/rfs.R")`

Statistics:

lm(formula) This function performs linear modeling on the data, with the supplied formula. As there is much information contained in this function, you will want to save the results in a variable, to retrieve the information through the `summary` and `names` functions.

predict(model) The `predict` function calculates the value of the dependent variable in the `model` given the independent variables used to create the model. If new predictions are required, the `newdata=` parameter must be used. This parameter takes a new set of data as its argument. Make sure that all independent variables used in the model are defined in the `newdata=` parameter. If not, an error message will results. Finally, the `se.fit=TRUE` parameter calculates the standard error at each prediction point.

summaryHCE(model) This function, a part of the `RFS` package, allows us to easily calculate the heteroskedastic-consistent standard errors (White 1980).

Probability:

pnorm(x) This function is the cumulative distribution function (CDF) for the Normal distribution. It returns a *p*robability that a Normally-distributed variable will be less than or equal to `x`. This function has two additional parameters that remove the requirement that `x` has undergone the z-transformation, `m` and `s`.

rnorm(n, m, s) This function returns `n` draws from a Normal distribution centered at `m` and with a standard deviation `s`. This function is the cornerstone of much Monte Carlo analysis.

Graphics:

lines(x,y) This is an extremely handy line-generating function, painting a line on the current plot (or returns an error if no plot exists). It first invisibly plots the pairs of points (`x`,`y`) then connects the points with drawn line segments.

If the `col` parameter is not set, then the line will be black. Otherwise, the line will be the color specified. There are three ways of stating the color: using the Windows 1-16 values, using names, and using the rgb values. The following all refer to 'red': `col=2`, `col="red"`, and `col="#ff0000"`.

plot() This function produces a scatterplot of the two-dimensional data. The call can be either `plot(x,y)` or `plot(y~x)`; both give identical results. This function can produce graphs that are very customized. The `R` help file for `par` is invaluable. Some important parameters include `xlab=""` (label for the x-axis), `ylab=""` (label for the y-axis), `xlim=c(min,max)` and `ylim=c(min,max)` (axis limits, min and max, for the x- and y-axis), and `las=1` (makes axis values painted horizontal).

Mathematics:

log(x, b) This returns the logarithm of `x`, with a base of `b`. If you omit the `b`, this function returns the natural logarithm of `x`. To calculate the common logarithm, set `b=10`. The logarithm function is used to transform variables bounded on one side into variables bounded on neither side.

exp(x) This function returns the exponential of the argument, `x`; that is, it returns e^x. The exponential function is the inverse of the logarithm function.

logit(x) This function returns the logit of the provided number. This number must be between 0 and 1, not including either 0 or 1. The logit function is frequently used to transform proportions into unbounded data. It is available through the `RFS` package.

logistic(x) This function returns the logistic of a given number. The range of the logistic function is 0 to 1, exclusive. it is the inverse of the logit function. As such, it is often used to transform predictions from logit units to proportion units. It is available through the `RFS` package.

cloglog(x) The complementary log-log function is a second appropriate transformation for proportion data. It is, however, not a symmetric function. It is available through the `RFS` package.

cloglog.inv(x) This function is the inverse of the complementary log-log function. It is available through the `RFS` package.

Programming:

which(condition) This function returns a vector of indices corresponding to the original vector's values meeting the criteria. Thus, `which(x==4)` returns the indices of all elements in vector `x` that equal 4. Note that equality is checked with a *double* equals, `==`. Other comparisons include: `>`, `<`, `>=`, `<=`, `!=`, `&`, `|`, and `!`. The last four are 'not equal to', 'and', 'or', and 'not'.

6.5.2 Exercises

This section offers suggestions on things you can practice from this chapter.

1. Predict the Venkovský 1994 cow ballot measure vote using the transformed vote model. Is *this* prediction physically possible?

2. Determine a 95% confidence interval, with the *un*transformed cow vote model, for predicting Děčín's vote. Is the actual outcome within the 95% confidence interval?

3. Determine a 95% confidence interval, with the *transformed* cow vote model, for predicting Děčín's vote. Is the actual outcome within the 95% confidence interval?

4. Determine if the assumptions of OLS are violated in the transformed cow vote model.

5. The actual vote share for Děčín was 52.8%. Explain why both models failed in predicting the actual vote outcome. How bad was the error? What can be done to improve the predictions?

6. The logit transformation is not the only possible choice. There is also the asymmetric complementary log-log transformation (`cloglog` in the `RFS` package). Use this function as the transformation to predict Děčín's vote, its 95% confidence interval, and the probability of the cow ballot measure passing. The inverse of the complementary log-log transform has no name, but the `R` function is `cloglog.inv`, also in the `RFS` package.

7. Estimate the GDP per capita for Papua New Guinea using the *un*transformed model, as well as the 95% confidence interval. How close is this estimate to the real answer, and it the real answer within the predicted confidence interval?

6.5.3 Applied Readings

- James M. Avery. (2009) "Political Mistrust among African Americans and Support for the Political System." *Political Research Quarterly* 62(**1**): 132–45.
- Mark Andreas Kayser. (2009) "Partisan Waves: International Business Cycles and Electoral Choice." *American Journal of Political Science* 53(**4**): 950–70.
- Pamela A. Morris. (2008) "Welfare Program Implementation and Parents' Depression." *The Social Service Review* 82(**4**): 579–614.
- Kar Tean Tan, Christopher C. White, and Donald L. Hunston. (2011) "An adhesion test method for spray-applied fire-resistive materials." *Fire and Materials* 35(**4**): 245–59.

6.5.4 Theory Readings

- George Casella and Roger L. Berger. (2001) *Statistical Inference*. New York: Duxbury Press.
- Annette J. Dobson and Adrian Barnett. (2008) *An Introduction to Generalized Linear Models*, Third Edition. New York: Chapman & Hall.
- Friedhelm Eicker. (1967) "Limit Theorems for Regression with Unequal and Dependent Errors." *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*: 59–82.
- Julian J. Faraway. (2004) *Linear Models with R*. New York: Chapman & Hall.
- Julian J. Faraway. (2005) *Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models*. New York: Chapman & Hall.
- Peter J. Huber. (1967) "The Behavior of Maximum Likelihood Estimates under Nonstandard Conditions." *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*: 221–233.
- John A. Nelder and Robert W. M. Wedderburn. (1972) "Generalized Linear Models." *Journal of the Royal Statistical Society. Series A (General)* 135(**3**): 370–84.
- Henry Scheffé. (1959) *The Analysis of Variance*. New York: Wiley.
- Shayle R. Searle. (1997) *Linear Models*. New York: Wiley-Interscience.
- James H. Stapleton. (2009) *Linear Statistical Models*. New York: John Wiley and Sons.
- Robert S. Stritchartz. (2000) *The Way of Analysis*, Revised Edition. Boston: Jones and Bartlett Mathematics.
- Halbert White. (1980) "A Heteroskedasticity-Consistent Covariance Matrix Estimator and a Direct Test for Heteroskedasticity." *Econometrica*. 48(**4**): 817–838.
- Simon N. Wood. (2006) *Generalized Additive Models: An Introduction with R*. New York: Chapman & Hall.

• Holbrook Working and Harold Hotelling. (1929) "Applications of the Theory of Error to the Interpretation of Trends." *Journal of the American Statistical Association* 24(**1**): 73–85.