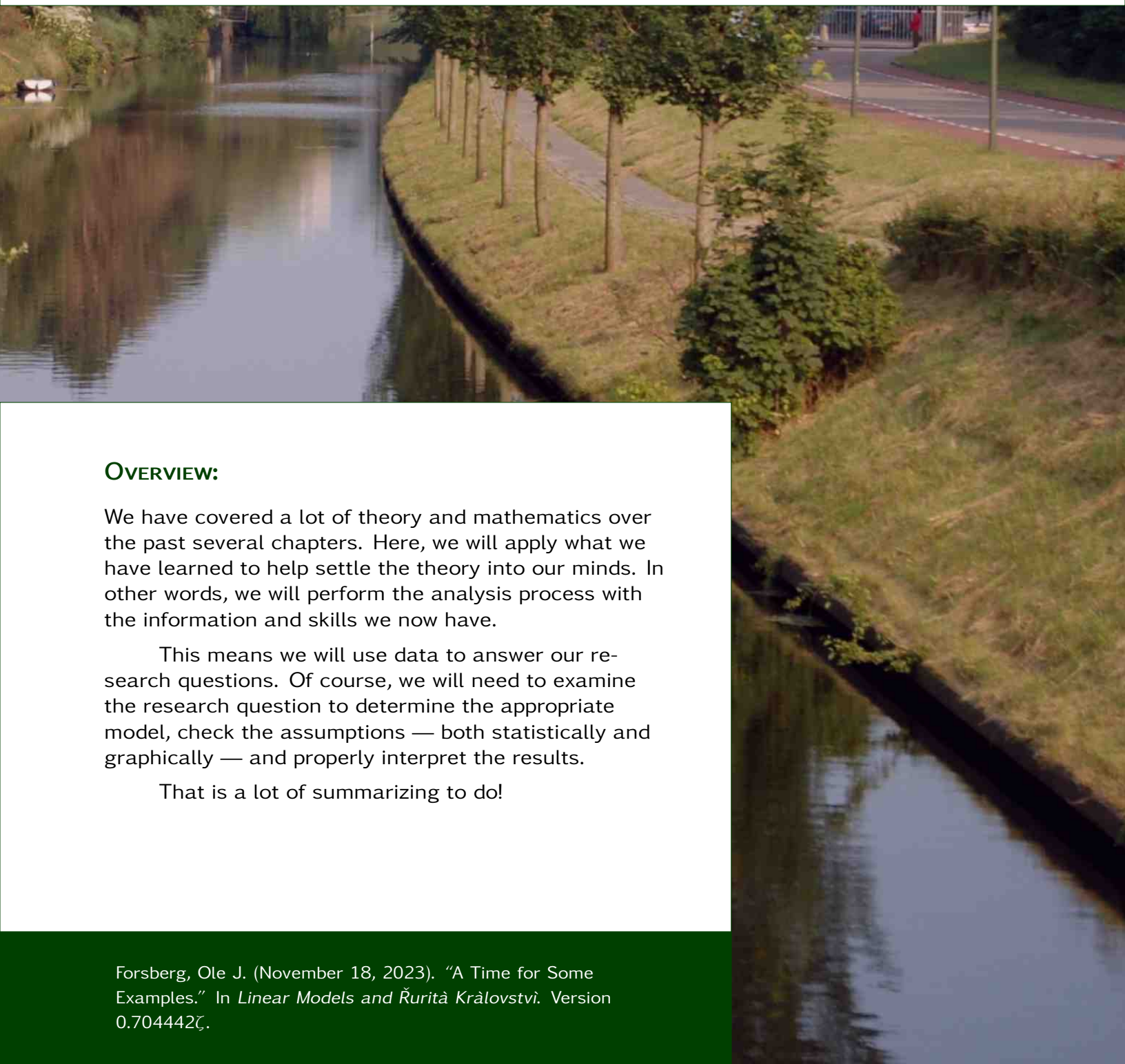
Chapter 5:
A Time for Some Examples



Overview:

We have covered a lot of theory and mathematics over the past several chapters. Here, we will apply what we have learned to help settle the theory into our minds. In other words, we will perform the analysis process with the information and skills we now have.

This means we will use data to answer our research questions. Of course, we will need to examine the research question to determine the appropriate model, check the assumptions — both statistically and graphically — and properly interpret the results.

That is a lot of summarizing to do!

Forsberg, Ole J. (November 18, 2023). "A Time for Some Examples." In *Linear Models and Řuritâ Kràlovstvì*. Version 0.704442ζ.

Chapter Contents

5.1 Full Example: Violent Crime 125
5.2 Full Example: Violent Crime, Wealth, Region 130
5.3 Full Example: Cows in the City of Děčín 138
5.4 Conclusion 153
5.5 End-of-Chapter Materials 154

And so, we have completed a majority of the important mathematics underlying ordinary least squares estimation. Be aware that OLS is how we estimate the parameters. The models itself is referred to as the classical linear model. It makes the usual four assumptions. The observations follow the equation

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_{p-1} x_{p-1} + \varepsilon \tag{5.1}$$

and the residuals follow this distribution

$$\varepsilon_i \overset{\text{iid}}{\sim} \mathcal{N}\left(0;\, \sigma^2\right) \tag{5.2}$$

From those assumptions, we were able to use OLS to calculate formulas for the estimators of $\beta_0, \beta_1, \ldots, \beta_{p-1}$. The next chapter used the distribution to determine the distribution of those estimators. This led to confidence intervals for the parameters and test statistics for testing hypotheses about the parameters. It also led to distributions and intervals and test statistics for estimated and predicted values of y.

All of that from four small assumptions.

This chapter will apply these results to different research questions to illustrate the statistical research process. So, turn the page and begin seeing applications of what we have done.

5.1: Full Example: Violent Crime

To help settle all of this, let's see a simple extended example of modeling the violent crime rate in 2000 using just the violent crime rate in 1990.

The preamble is the part of the code that imports the extra functions, loads the data, and gives us an overview of it. This is a typical preamble

preamble—textbf

```
### Preamble
source("http://rfs.kvasaheim.com/rfs.R")
library(lawstat)
library(lmtest)

dt = read.csv("http://rur.kvasaheim.com/data/crime.csv")
attach(dt)

summary(dt)
```

Note that there are *many* variables in this data set. Since we are modeling the violent crime rate in 2000 using the rate in 1990, we will only use the variables `vcrime00` and `vcrime90`. To fit the model and estimate the parameters using ordinary least squares, run this line:

OLS

```
crimeMod = lm(vcrime00 ~  vcrime90)
```

Nothing gets outputted by this line. `R` just echoes it if you typed it correctly. However, a lot has happened behind the scenes. Inside `R`, the model was fit using ordinary least squares (using matrices). The parameters were estimated. All of this was done behind the scenes.

The next step is to check that the model does not violate any of the assumptions/requirements.

5.1.1 Normality of the Residuals

The first we will check is the Normality of the residuals:

```
e = residuals(crimeMod)

# Normal Residuals?
overlay(e)
shapiroTest(e)
```

The histogram overlaid with the normal curve suggests the residuals are slightly skewed to the right. The Shapiro-Wilk test strongly indicates a lack of Normality (p-value = 0.004424). The sample size of $n = 51$, however, definitely seems large enough to ensure the sums of the residuals closely follows a normal distribution (this is the *actual* requirement). If you would like to check this, run the following code (first think what it does and why it answers this problem):

```
et = numeric()
for(i in 1:1e3) {
  x = sample(e, replace=TRUE)
  et[i] = sum( x )
}
shapiroTest(et)
```

Since the reported p-value is much greater than α, we can conclude that the sample sums are sufficiently Normal. And, it is the sample sums that affect the distribution of b_0 and b_1.

Thus, the model passes the normality requirement.

5.1.2 Constant Expected Value (Functional Form)

The second assumption to test is constant expected value (proper model form):

```
# Constant Expected Value of Residuals
plot(vcrime90,e)
runs.test(e, order=vcrime90)
```

The residuals plot seems a bit inconclusive to me. This is mainly due to the single point far to the right (the District of Columbia). The runs test, however, indicates that there is no significant evidence the residuals follow anything other than a horizontal line (p-value = 0.6732).

Thus, the model does not violate the second assumption.

5.1.3 Constant Variance

The third assumption is that the variance of the residuals is constant:

```
# Constant Variance of Residuals
plot(vcrime90,e)
bptest(crimeMod)
```

For me, the graphic is inconclusive because of DC. The Breusch-Pagan test did not detect significant heteroskedasticity (p-value = 0.1041).

Thus, the model passes the third and final requirement.

5.1.4 The Final Model

This model seems appropriate, and we can now see the estimates:

```
Call:
lm(formula = vcrime00 ~  vcrime90)

Residuals:
    Min      1Q  Median      3Q     Max
-241.32  -42.84  -18.04   40.97  208.41

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 109.52716   21.42679   5.112 5.27e-06 ***
vcrime90      0.58065    0.03107  18.689  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.10   1

Residual standard error: 85.55 on 49 degrees of freedom
Multiple R-squared:  0.877, Adjusted R-squared:  0.8745
F-statistic: 349.3 on 1 and 49 DF,  p-value: < 2.2e-16
```

The first section reports the model you presented. Use it to double-check you typed things in correctly or to remind you what the model is examining. The second part produces the five-number summary of the residuals. Since the mean (0) is greater than the median (−18.04), there is evidence of a positive skew to the residuals. This, we discovered above.

The third section is the "regression table." Each row corresponds to a different independent variable (or the intercept). The columns are the estimates, the standard errors, the test statistic (estimate divided by standard error), and the p-value.

In this example, there is very strong evidence that the relationship between the violent crime rate in 1990 and in 2000 is positive ($b_1 = 0.58065$). If State A had a higher violent crime rates in 1990 than State B, then it also tended to have a higher violent crime rates in 2000.

The intercept, $b_0 = 109.52716$ indicates that for a state with 0 violent crime in 1990, the expected violent crime rate in 2000 is 109.52716 crimes

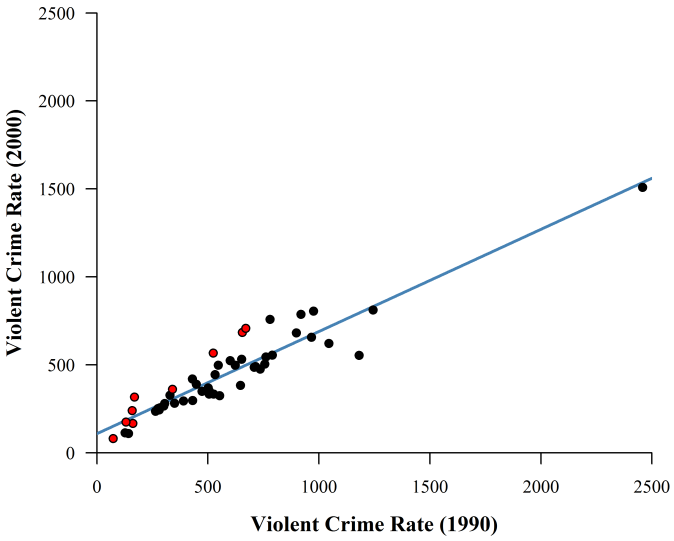


Figure 5.1: *Plot of the violent crime rate in 2000 against that in 1990. The ordinary least squares line of best fit is included. Red-colored states are those whose violent crime rate increased from 1990 to 2000.*

per 100,000 people. However, since no state was close to having a violent crime rate in 1990 of 0, this interpretation does not make statistical sense.

Remember that we should only use our models to predict and estimate for values of the 1990 violent crime rate within the domain of the `vcrime90` variable in our data.

interpolation

5.1.5 Graphic

The following lengthy code produces a graphic like that at the top of the page:

```
plot.new()
plot.window( xlim=c(0,2500), ylim=c(0,2500) )

axis(1); axis(2)

title(xlab="Violent Crime Rate (1990)", line=2.75)
title(ylab="Violent Crime Rate (2000)", line=3.25)

xx = seq(0,2500)
yy = predict(crimeMod, newdata=data.frame(vcrime90=xx))
lines(xx,yy, col="steelblue", lwd=2)

points(vcrime90,vcrime00, pch=21, bg=1+(vcrime00>vcrime90))
```

Note that the graphic also indicates which states had their violent crime rate increase. It comes from this line:

```
points(vcrime90,vcrime00, pch=21, bg=1+(vcrime00>vcrime90))
```

The first two slots are the x- and y-values. The third slot specifies the plotting character. A `pch` of 21 is a dot with its insides colorable. The fourth slot, `bg`, specifies the color to fill the inside of the dots (`bg` = "background").

The part `(vcrime00>vcrime90)` takes on value 1 if the violent crime rate increased and 0 otherwise. Adding 1 to each ensures that the two colors are 1 and 2 — black and red.

5.1.6 Confidence Interval for β_1

In addition to calculating the point estimates of the slope and intercept, we can also calculate confidence intervals:

```
confint(crimeMod)
```

From this output, we are 95% confident that the effect of the violent crime rate in 1990 on 2000 is between 0.518 and 0.643.

5.1.7 Confidence Interval for Y

estimation

We can estimate the value of Y for a given value of x:

```
predict(crimeMod, newdata=data.frame(vcrime90=100), interval="
    confidence")
```

We are 95% confident that the expected value of Y when $x = 100$ is between 129.5 and 205.6, with a point estimate of 167.6.

5.1.8 Prediction Interval for Y

prediction

Finally, we can predict the value of Y for a new value of x:

```
predict(crimeMod, newdata=data.frame(vcrime90=100), interval="
    prediction")
```

We are 95% sure that the next *observation* of the violent crime rate in 2000 for a state with a violent crime rate in 1990 of 100 is between −8.5 and 343.7, with a prediction of 167.6.

5.2: Full Example: Violent Crime, Wealth, Region

That was fun! Let's now try this with two independent variables. We will model the violent crime rate in 2000 using the GSP per capita in 1990 *and* the region of the state. This will give us the opportunity to reiterate and emphasize that these methods are not constrained to numeric independent variables. As in Example 2.2 on page 24, we can represent categorical independent variables appropriately and model using ordinary least squares estimation.

The following creates the interaction model between a numeric and a categorical variable. This particular type of interaction analysis is referred to as the Analysis of Covariance, ANCOVA:

interaction

```
modEd1 = lm(vcrime00 ~  gspcap90 * census9)
summary.aov(modEd1)
```

The interaction model allows for the effect to vary between the levels. In terms of this problem, the interaction model allows the effect of the 1990 violent crime rate on the 2000 to be different for the Midwest, the Northeast, the South, and the West.

It does not *force* it to be different. It only allows it to be.

Occam's Razor

Because of the writings of a 14th-Century monk by the name of William of Ockham, there is a bias in science to create models that are as simple as possible, without being too simple (his doctrine of efficient reasoning).[1]

Non sunt multiplicanda entia sine necessitate.

The usual translation is "Things are not to be multiplied without necessity." In other words, simpler models tend to be more helpful than complicated ones. Realize that they are more "helpful" and not more "correct." To drive this point home, allow me to quote George E. P. Box (1976):

> Since all models are wrong the scientist cannot obtain a "correct" one by excessive elaboration. On the contrary following William of Occam he should seek an economical description of natural phenomena. Just as the ability to devise simple but evocative

[1] Note, however, that this doctrine/belief did not originate with William. It goes back to — at least — Aristotle in his *Posterior Analytics*: "We may assume the superiority *ceteris paribus* of the demonstration which derives from fewer postulates or hypotheses."

> models is the signature of the great scientist so overelaboration and overparameterization is often the mark of mediocrity.

The results from the above code are given here

```
                   Df  Sum Sq Mean Sq F value   Pr(>F)
gspcap90            1  816163  816163  26.149 1.32e-05 ***
census9             8  794820   99353   3.183   0.0087 **
gspcap90:census9    8  273868   34233   1.097   0.3901
Residuals          33 1029986   31212
```

Note that the p-value (last column) for the interaction term is greater than our usual $\alpha = 0.05$ ($p-value = 0.3901$). This tells us that the interaction term is not statistically significant. In other words, we can remove it from our model without adversely affecting our model.

When the model has no interaction terms, it is called an additive model. The following code fits the additive model. **additive**

```
modEd2 = lm(vcrime00 ~  gspcap90 + census9)
```

As usual, the next step is to test the assumptions. Note that this process is the same, even if the particulars differ. The `census9` variable is categorical, not numeric. That requires we think a bit more about how to perform the assumption testing.

5.2.1 Normality

Again, the first assumption to test is the Normality of the residuals:

```
e = residuals(modEd2)

# Normality checking
overlay(e)
shapiroTest(e)
```

According to the Shapiro-Wilk test there is no significant evidence that the residuals come from a non-Normal distribution ($p-value = 0.1732$). Thus, the model passes this test.

5.2.2 Constant Expected Value (Functional Form)

The second requirement we test is that the expected value of the residuals is constant against *each of* the independent variables.

```
# Expected Value
plot(gspcap90, e)
runs.test(e, order=gspcap90)
```

The runs test indicates that there is no evidence the expected values are not constant (p – value = 0.2038). Thus, this test is passed, too. Yippee!

But wait! This only tested for constant expected value of the residuals against one of the two independent variables. It is required that the expected value is constant against *all* of them.

Here is the problem with just using the runs test when the independent variable is categorical: The ordering within each level is not uniquely defined. For a hint[2] on what to do, look at the graphic:

```
plot(census9,e)
```

Holy side-by-side box-and-whiskers plot, Batman! This makes sense, because we are plotting a numeric variable (e) across nine levels. We need to test that the expected value (mean) is the same in each group.

From your prior statistics class, this screams ANOVA!

```
summary(aov(e~ census9))
```

The p-value returned is 1, which is greater than α. So we fail to reject the null hypothesis of equal expected values.

Well, this result should not be surprising. Because of the mathematics of OLS, the means in each group will be centered at zero. Thus, you should expect p-values of 1 whenever doing this test.

5.2.3 Constant Variance

The last requirement is that the residuals have a constant variance against each of the independent variables. For the numeric variable, this is not a problem:

```
# Heteroskedasticity
plot(gspcap90,e)
hetero.test(e,gspcap90)
```

With a p-value of 0.7008, this test is passed for the numeric independent variable.

[2] When not sure what to do, plotting things frequently helps. It seems to force the researcher into determining what needs to be examined. When in doubt: Graph and Interpret.

For the categorical variable, remember that we need to test equality of variance *across several groups*. From your previous statistics course, you may recall that the Fligner-Killeen test does just this:

```
plot(census9,e)
fligner.test(e~ census9)
```

According the the Fligner-Killeen test, there is no evidence of heteroskedasticity (p – value = 0.7869).

Thus, this model passes the homoskedastic requirement. Vahooo!

Note: We could have taken care of both tests for heteroskedasticity using the Breusch-Pagan test. However, if the model fails that test, we have no clue as to how to fix it. Breaking it into two parts allows us the additional information of which variable caused the issues.

Personally, I run the Breusch-Pagan test to determine *if* there is a violation, then the separate tests to get information on *where* it issue lies.

5.2.4 The Final Model

This model is appropriate, and we can now see the estimates. However, if we are in the 'model creation' or 'model selection' mode, we need to determine if both variables are statistically significant. If either is not, then that variable needs to be dropped and the new model tested.

To get p-values for the variables, just run

```
summary.aov(modEd2)
```

Yeppers, that is `summary.aov` that you are using. It provides statistical significance of the *variables*, while `summary` and `summary.lm` provide the statistical significance of the levels of the categorical variables.

The output from the `summary.aov(modEd2)` command is

```
             Df  Sum Sq Mean Sq F value   Pr(>F)
gspcap90      1  816163  816163  25.664 9.07e-06 ***
census9       8  794820   99353   3.124  0.00753 **
Residuals    41 1303854   31801
```

The p-value for the gspcap90 variable is less than alpha, so that variable has a significant effect on the violent crime rate. The p-value for the census9 variable is also less than alpha, so it too needs to be included in the final model.

In short, this is the model we need to use. The abbreviated regression table from this model is

```
Coefficients:
                              Estimate        Pr(>|t|)
(Intercept)                  1.125e+02           0.308
gspcap90                     1.550e-02         6.7e-05 ***
census9East South Central    7.518e+01           0.535
census9Middle Atlantic      -6.777e+01           0.608
census9Mountain             -1.707e+02           0.101
census9New England          -1.708e+02           0.122
census9Pacific              -1.316e+02           0.263
census9South Atlantic        1.590e+02           0.125
census9West North Central   -4.953e+01           0.638
census9West South Central    1.197e+02           0.323
```

From this, we can conclude that there is a statistically significant, and positive (!), effect of average state wealth on the violent crime rate. We get that conclusion from the gspcap90 line in the table.

base category

The rest of the table compares the effect of each level of the census9 variable to the base category, East North Central. As no p-values is less than alpha, we can conclude that none of the regions is statistically different in its effect from the East North Central region.

What about when compared to the Mountain region?

First, we have to specify that we want the Mountain region to be the base category against which everything else is calculated. Then, we need to re-fit the model with the new base.

```
census9 = set.base(census9, "Mountain")
modEd3 = lm(vcrime00 ~  gspcap90 + census9)
summary(modEd3)
```

The regression table now indicates that the violent crime rate in the Mountain region is significantly lower than that in the East South Central, South Atlantic, and West South Central regions.

How does the violent crime rate in the different regions compare to the Pacific region?

```
census9 = set.base(census9, "Pacific")
modEd3 = lm(vcrime00 ~  gspcap90 + census9)
summary(modEd3)
```

The violent crime rate in the Pacific region is significantly lower than that in the South Atlantic and the West South Central regions.

How do the regions compare to the South Atlantic region?

```
census9 = set.base(census9, "South Atlantic")
modEd3 = lm(vcrime00 ~  gspcap90 + census9)
summary(modEd3)
```

The violent crime rate in the South Atlantic region is significantly higher than in the Mountain, New England, Pacific, and West South Central regions.

> ***Note***: Be aware of the multiple comparisons issue (see Section S.6.2). Remember that these individual analyses only work if you perform only one of them. Multiple comparisons require adjustment of the alpha-level. For a reminder, see Appendix Section S.6.2.

5.2.5 The Graphic

The following code generates the graphic at the top of the next page:

```
par(mar=c(4,4,0,1)+0.5, family="serif", las=1)
par(xaxs="i", yaxs="i")
par(cex.lab=1.2, font.lab=2)

plot.new()
plot.window( xlim=c(0,75), ylim=c(0,2500) )

axis(1); axis(2)

title(xlab="GSP per Capita (1990) [$000]", line=2.75)
title(ylab="Violent Crime Rate (2000)", line=3.5)

xx = seq(15,70)*1000
```

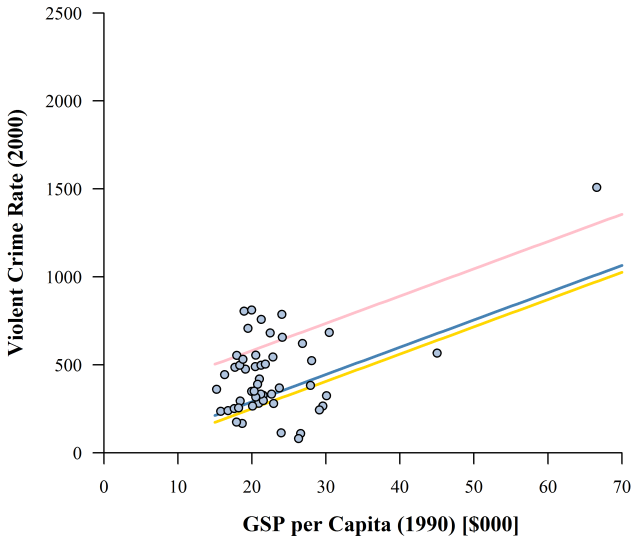


Figure 5.2: *Plot of the violent crime rate in 2000 against the GSP per capita in 1990 (in thousands of dollars). The ordinary least squares line of best fit is included. The red-colored line is the estimate for the South Atlantic states; blue, Pacific states; and gold, Mountain states.*

```
yyPac = predict(modEd2, newdata=data.frame(gspcap90=xx, census9
    ="Pacific"))
yyMtn = predict(modEd2, newdata=data.frame(gspcap90=xx, census9
    ="Mountain"))
yySAt = predict(modEd2, newdata=data.frame(gspcap90=xx, census9
    ="South Atlantic"))

lines(xx/1000,yyPac, col="steelblue", lwd=2)
lines(xx/1000,yyMtn, col="gold", lwd=2)
lines(xx/1000,yySAt, col="pink", lwd=2)

points(gspcap90/1000,vcrime00, pch=21, bg="lightsteelblue")
```

It would be helpful to have a legend, but let us leave that for another day!

Also, you need to be able to determine what each line of this script does.

5.2.6 Confidence Interval for Slope

We can obtain confidence intervals for the effects using the same method as before. The interpretation follows the same rules, but the table is *much* bigger:

```
confint(modEd2)
```

We are 95% confident that the effect of the GSP per capita on the violent crime rate is between 8 and 23 additional violent crimes (per 100,000 population) for every $10,000 increase in GSP per capita.

The confidence intervals for the effects of each of the levels is *as compared to* the base level. Thus, we are 95% confident that the average violent crime rate in the West North Central region is between 21 and 396 lower than that in the South Atlantic region (the base level).

5.2.7 Confidence Interval for $\hat{y}$

Again, we can estimate the expected value of a violent crime rate, given the GSP per capita and the region.

```
predict(modEd2, newdata=data.frame(gspcap90=50000, census9="
    Pacific"), interval="confidence")
```

We are 95% confident that the expected violent crime rate in a Pacific-region state with a GSP per capita of $50,000 is between 537 and 975, with a point estimate of 756 violent crimes per 100,000 people.

5.2.8 Prediction Interval for y_{new}

Finally, we can also calculate a prediction interval for a new observation:

```
predict(modEd2, newdata=data.frame(gspcap90=50000, census9="
    Pacific"), interval="prediction")
```

We are 95% sure that the violent crime rate for a new observation of a Pacific-region state with a GSP per capita of $50,000 is between 334 and 1177, with a best guess of 756 violent crimes per 100,000 people.

Note: As is always the case, the width of the prediction interval is larger than the width of the confidence interval. Remember why this is the case.

5.3: Full Example: Cows in the City of Děčín

To illustrate the process of model selection, let us examine Děčín's ballot measure of 2016. That ballot measure sought to constitutionally restrict the number of cows that can be housed within the city limits. While this extended example seems rather dated, it does cover some interesting issues in statistical modeling and questions we can answer with our model.

Example 5.1: The voters of Děčín are being sent to the polls to vote on a constitutional referendum that proposes to limit the number of cows that could be housed within the city limits. 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, and the demographics of Děčín voters, what is the probability that this ballot measure will pass?

•

Before attempting any analysis, there needs to be a search of the literature to inform us as to which variables should be present, and which directions those variables should affect the dependent variable. From that literature review, we hypothesize that the vote in favor of such ballot measures depends on three variables: age of the population, religiosity of the population, and whether the ballot measure also restricts chickens. The effect direction for each is that kraj that are more religious should vote against cow-housing at a higher rate; Measures that also ban chickens should have a harder time passing; Measures passed more recently should have a more difficult chance of passing, as the young tend to support cows, and the elderly tend to oppose them (wanting quiet, dung-free neighborhoods).

directional hypotheses

With this theory and the resulting hypotheses, we can take our next step: Getting to know the data.

	Year Passed	Chicken Ban	Religious Percent
Minimum	1998	0	51.00
Maximum	2008	1	85.00
Median	2004	1	67.50
Mean	2004	0.5938	66.75
Variance	6.0650	0.2490	88.1935
Coefficient of Variation	0.5794	0.8404	0.1407

Table 5.1: *Descriptive statistics on the variables in the* `cows` *dataset.*

5.3.1 Get to Know the Data

Before we begin trying to answer this question, we must get to know our data. There are several functions available to us to visualize the data: histogram, scatter plots, and quantile-quantile plots. In addition to visualizing the data, we should calculate several of the descriptive statistics for the variables of interest.

```
source("http://rfs.kvasaheim.com/rfs.R")

cows = read.csv("http://rur.kvasaheim.com/data/cows.csv")
summary(cows)
```

Variability: Since we have multiple independent variables, we should calculate both univariate and bivariate descriptive statistics. Table 5.1 provides the univariate descriptive statistics. The primary univariate question to ask about the independent variables here is whether there is sufficient variation. The two measures we need to examine are the variance and the coefficient of variation. If both of these numbers are small, then there may be an issue.

variation

In this data, the variance of the Chicken Ban variable is small and potentially worrisome; however, its coefficient of variation (a scaled standard deviation, $c_v = \left| \frac{s}{\bar{x}} \right|$) indicates that there is no serious issue (the value is close to 1).[3] None of the three variables have small enough variation to cause us concern.

coefficient of variation

[3] As this is a dichotomous variable, the mean is the percent of the values equal to 1. Thus, there are about 60% of the values 1 and 40% of the values 0 — more than sufficient variation.

	Year Passed	Chicken Ban	Religious Percent
Year Passed		0.1903	0.2399
Chicken Ban	0.1903		0.5146
Religious Percent	0.2399	0.5146	

Table 5.2: *The correlations between the variables in the* `cows` *data. The correlation between Chicken Ban and Percent Religious is statistically significant ($t = 3.2869; \nu = 30; p = 0.0026$). This is the sole statistically significant correlation.*

Relationships: After getting to know the variables individually, it is important to get to know the relationships between the variables. This can be done through correlation tests and bivariate scatter plots. Independent variables with strong correlations with the dependent variable should be considered for inclusion in the model. Independent variables with strong correlations with other independent variables should be of concern. Remember that one of the assumptions of OLS regression is that the independent variables are statistically independent of each other. If independent variables are highly correlated, the statistical properties of the method weaken.

correlated

The pairwise correlations are provided in Table 5.2. Of the three independent variables, only Chicken Ban and Religious Percent have a statistically significant correlation ($t = 3.2869; \nu = 30; p = 0.0026$). Should the level of correlation be a concern? Perhaps. While their correlation is $r = 0.5146$, this corresponds to an R^2 value of just 0.2648. As such, the correlation may not be large enough to severely affect our coefficient estimates (see Sections 2.2.1 and 2.5). Let us just remember this relationship for the future.

> ***Note***: The issue is actually more than a statistics issue. If two independent variables are highly correlated with each other, it is logically impossible to determine *which* affects the dependent variable or how much of the effect to partition to each independent variable. Statistics is, however, able to tease out the independent relationships better than not. As a rule of thumb, if the correlation is greater than $r = 0.90$, there may be a serious logical issue. If two variables are so highly correlated, which of the two is the "correct" independent variable? How can one tell? Can both be good? Is the commonality between them the real independent variable?

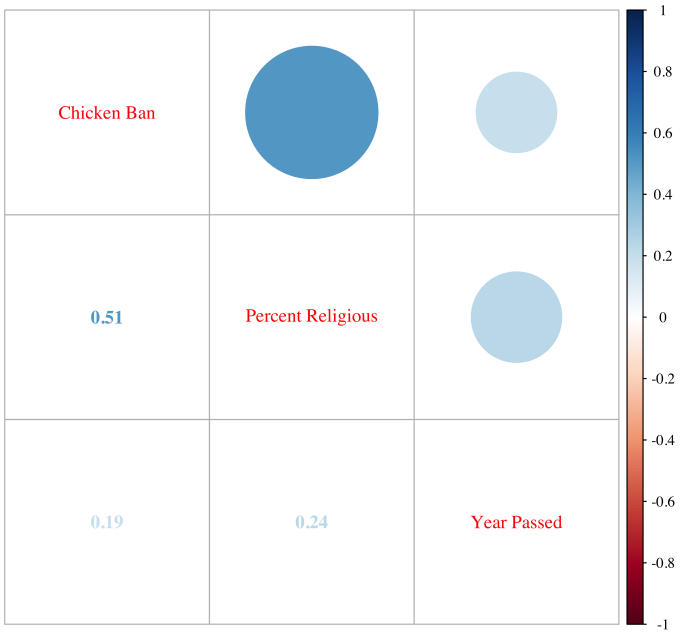


Figure 5.3: *Correlation plots between the three independent variables. The correlation between* `Civil Ban` *and* `Percent Religious` *was statistically significant according to the Pearson product-moment correlation test. This is evident in this graph, as well.*

VIF

5.3.2 Variance Inflation Factor

This problem is a bit more extensive than suggested above. Recall that one of the mathematical requirements is that the rank of the design matrix equals p (the number of parameters to be estimated). This can happen if one variable is perfectly correlated with (linear function of) another variable. It can also happen if a variable is a linear function of the other variables.

Thus, while checking the bivariate correlations is helpful, it is not the answer we need. We need to check if the independent variable is a linear function of (or close to) a combination of the others.

To determine the level of multicollinearity we can use the "variance inflation factor" (VIF). Recall from Section 4.4.2 how to calculate the VIF for a given independent variable: Regress all other independent variables on it, calculate the R_i^2, and calculate the VIF from

$$\text{VIF}_i = \frac{1}{1 - R_i^2} \tag{5.3}$$

Once you have calculated the VIF for each independent variable, compare the values to the "usual" Rule of Thumb.[4]

There is nothing magical about using the R^2 value to calculate your variance inflation factor. One could advocate using the adjusted-R^2. However, in doing so, the Rules of Thumb may need to be adjusted.

Of course, if we think about the effects of multicollinearity, rather than just detecting "severe levels," then we may wish to eschew such tests, assume multicollinearity is an issue, and adjust for it. On the other hand, if there is no reason to think that the model should suffer from multicollinearity, we will want to avoid such adjustments.

These are science questions, not statistics questions.

Know the science behind your theory.

[4] This Rule of Thumb depends on the discipline. The three typical boundaries are 5, 8, and 10. If all of your VIF scores are greater than 10, then there is an issue with multicollinearity. If all are less than 5, then there is no issue. If it is between those extremes, then you should think about the effect of multicollinearity on your estimators. What do those VIF values correspond to? A VIF of 5 means the R^2 value is 0.80. A VIF of 10 means the R^2 is 0.90. Keep that in mind. In other words, the *other* independent variables explain 90% of the variation in this independent variable.

~	Separates the dependent variable (left-hand side) and the independent variables (right-hand side)
+	Indicates the following variable is added to the formula
−	Indicates the following variable is removed from the formula
:	Indicates the following and the preceding variable are multiplied in the formula
*	Indicates the following and the preceding variable are crossed in the formula
^	Includes the specified level of interactions.
I()	Replaces the formula grammar of what is in the parentheses with algebraic grammar.

Table 5.3: *The symbols and their meanings in the grammar of formulas. I sure wish I could locate the book that created these, but I cannot find it anymore. It is in the Oklahoma State University library... somewhere.*

5.3.3 Model the Data The example asked us to determine the probability that the ballot measure will pass. Before we can answer that question, we need to model the proportion of the vote in favor of the ballot measure using our independent variables; that is, we need to be able to predict the proportion of the vote in favor of the ballot measure with the information we have. **prediction**

Thus, the dependent variable will be `propWin` and the independent variables will be `yearPassed`, `chickens`, and `religPct`. For now, let us assume a linear relationship between the independent variables and the dependent variable.

Model Selection: Unless you have a lot of independent variables, I recommend you start with the interaction model.[5] The interaction model includes the effects of each independent variable singly (main effects) as well as all possible combinations of those variables (interaction effects). **interaction model**

`R` uses the usual formula grammar (Table 5.3). Its use takes prac- **grammar**

[5]Some will disagree and recommend starting with the simplest model and building complexity from that. There tends to be little difference between the two model-building methods. On either case, one has to worry about the multiple comparisons issue (Appendix S.6.2). How we should address it in the realm of model building is still unknown. We are certain of two things, however. First, the Bonferroni procedure is far too conservative. Second, doing nothing is not an acceptable option.

tice. For instance, if you wish to fit the model $y = \beta_0 + \beta_1 x + \varepsilon$, you would use `y ~ x`. If you wish to fit the model $y = \beta_1 x + \varepsilon$, you would use either `y ~ x - 1` (my usual) or `y ~ x + 0`.

Some other examples include:

Algebraic form	Formula form
$y = \beta_0 + \beta_1 x1 + \beta_2 x2 + \beta_3 x1x2$	`y ~ x1*x2`
$y = \beta_0 + \beta_1 x1 + \beta_2 x2 + \beta_3 x1x2$	`y ~ x1 + x2 + x1:x2`
$y = \beta_0 + \beta_1 x1 + \beta_2 x2 + \beta_3 x1x2$	`y ~ (x1 + x2)^2`
$y = \beta_0 + \beta_1 x1x2$	`y ~ x1:x2`
$y = \beta_0 + \beta_1 x1^3 + \beta_2 \sin(x2)$	`y ~ I(x1^3) + I(sin(x2))`
$y = \beta_0 + \beta_1 x1 + \beta_2 x2 + \beta_3 x3 + \beta_4 x1x2 + \beta_5 x1x3 + \beta_6 x2x3 + \beta_7 x1x2x3$	`y ~ x1*x2*x3`

With this brief introduction to the grammar of formulas, we can return to our example. We have three independent variables; the formula to give a full interaction model is

asterisk

```
propWin ~ yearPassed * chickens * religPct
```

As we will use this model a bit, we save the linear regression results into a variable. Thus, the two lines to run are

```
mod1 = lm(propWin ~  yearPassed * chickens * religPct)
summary(mod1)
```

These lines give the following output (well the first, fourth, and fifth column of that output):

```
                                t value   Pr(>|t|)
(Intercept)                       1.148      0.262
yearPassed                       -0.901      0.377
chickens                         -1.084      0.289
religPct                          1.557      0.133
yearPassed:chickens               0.950      0.352
yearPassed:religPct               0.510      0.615
chickens:religPct                 0.979      0.338
yearPassed:chickens:religPct     -0.895      0.379
```

The line starting `yearPassed:chickens:religPct` is the three-way interaction term. As it is the highest interaction, it is the *only one* we can interpret here. Note that it is not statistically significant ($p = 0.379$). Thus, removing that term will do two things. First, it will simplify the model.

three-way

Occam

Second, it will not significantly harm the model's descriptive (or predictive) ability.

That second model can be written as either

```
mod2 = lm(propWin ~  yearPassed * chickens * religPct -
   yearPassed:chickens:religPct)
```

or as

```
mod2 = lm( propWin ~  (yearPassed + chickens + religPct)^2 )
```

The two formulas are equivalent. **formula grammar**

Note that the `summary.aov(mod2)` command indicates that none of the three two-way interactions are statistically significant. Thus, these two-way interactions should be removed from the model.[6] This leaves a model with no interactions—an additive model. Fitting the additive model and checking the statistical significance of the variables is as above **two-ways** **additive model**

```
mod3 = lm(propWin ~  yearPassed + chickens + religPct)
summary.aov(mod3)
```

Note that all three variables are significant according to this output (the chicken variable is statistically significant because we specified an effect direction). Thus, this is our provisional model. **provisional model**

The Additive Model: That is, the equation we will use to fit the data is

$$\texttt{propWin} = \beta_0 + \beta_1(\texttt{yearPassed}) + \beta_2(\texttt{chickens}) + \beta_3(\texttt{religPct}) + \varepsilon \quad (5.4)$$

If $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, then we know

$$\mathbb{E}[\texttt{propWin}] = \beta_0 + \beta_1(\texttt{yearPassed}) + \beta_2(\texttt{chickens}) + \beta_3(\texttt{religPct}) \quad (5.5)$$

Check the Assumptions: But, does this model violate any of the assumptions of OLS regression? All of the usual tests (Shapiro-Wilk, Breusch-Pagan, and runs) pass.

[6] Again, some would alternatively advocate removing just the least significant effect, then refit the new model. Others would suggest refitting with three different models, one for each combination of interaction. There is no "always best" answer, other than the one that your science suggests.

What about multicollinearity? Remember that the effect of multicollinearity is to inflate the standard errors (reduce the t-value, increase the p-value). Thus, if multicollinearity exists, fixing it will make the variables even more statistically significant.

How do we test it? We can do it the hard way or the easy way. The hard way is to estimate three regression equations, calculate the individual R^2 values, and calculate the resulting VIF values.

```
vif1 = lm(yearPassed ~  chickens + religPct)
vif2 = lm(chickens ~  yearPassed + religPct)
vif3 = lm(religPct ~  yearPassed + chickens)

1/(1-summary(vif1)$r.squared)
1/(1-summary(vif2)$r.squared)
1/(1-summary(vif3)$r.squared)
```

Or, you can use the `car` package:

```
library(car)
vif(mod3)
```

The results of these VIF checks are

```
yearPassed   chickens   religPct
  1.067965   1.368963   1.399954
```

None of these three are even close to the lowest "Rule of Thumb." As such, multicollinearity is not an issue in this model.

5.3.4 Results

The regression table for model `mod3`, produced using `summary(mod3)`, is given in Table 5.4. Notice that all three variables of interest are statistically significant at the $\alpha = 0.05$ level.[7] Additionally, the model has an $\overline{R}^2$ of 0.7565, which is a great fit in most of the social sciences. The direction of the coefficients also agrees with theory.

directional hypothesis

research hypotheses

[7] You may claim that the Chicken variable is not statistically significant at the $\alpha = 0.05$ level. However, the provided p-values are two-tailed p-values. Our hypotheses were all directional hypotheses (one-tailed). Thus, to get the one-tailed p-values just halve the two-tailed p-values. With that, all three independent variables are statistically significant.

	Estimate	Std. Error	t-value	p-value
Constant Term	0.1512	0.0659	2.293	0.0295
Year Passed (post 2000)	-0.0201	0.0036	-5.618	$\ll 0.0001$
Banned Chickens	-0.0373	0.0200	-1.868	0.0723
Percent Religious	0.0095	0.0011	8.801	$\ll 0.0001$

Table 5.4: *Results table for the regression of proportion support of a generic ballot limiting the number of cows housed in the city against the three included variables. The R^2 for the model is 0.7801; the $\overline{R}^2$, 0.7565. The p-values calculated are based on two-tailed test. The hypotheses were one-tailed hypotheses. As such, all three explanatory variables are statistically significant at the standard level of significance ($\alpha = 0.05$).*

Thus, the equation for the line of best fit is approximately **prediction line**

$$\begin{aligned}\mathbb{E}[\texttt{propWin}] &= 0.1512 && (5.6)\\ &\quad - 0.0201(\texttt{yearPassed}) && (5.7)\\ &\quad - 0.0373(\texttt{chickens}) && (5.8)\\ &\quad + 0.0095(\texttt{religPct}) && (5.9)\end{aligned}$$

5.3.5 Predicting Děčín

According to this model, what is the expected vote in Děčín? To answer this, we need information about the Děčín ballot measure, specifically the value of the independent variables: `yearPassed` = 9, `chickens` = 0, `religPct` = 48. With this information, and under the assumption that the model is correct, we have our prediction that 42% of the Děčín voters will vote in favor of this ballot measure.

Thankfully, `R` does not require us to do this calculation by hand. The `R` code for predicting the percent of Děčín voters voting in favor of this ballot measure can be

```
DECIN = data.frame(yearPassed=9, chickens=0, religPct=48)
predict(mod3, newdata=DECIN)
```

The first line was used to make the code more readable. It is also helpful to first define the variable `DECIN` if you are going to make predictions for Děčín using several models.

If neither of these appeal to you and you wish to do this in one line, that line would be

```
predict(mod3, newdata=data.frame(yearPassed=9, chickens=0,
    religPct=48))
```

predict

Note the inclusion of the `predict` function, which predicts the dependent variable value given values for each of the independent variables (read the help file on `predict`; we will use this function frequently).

5.3.6 Graphing the Results

Now that we have confidence in our model, we can use it to predict the effects of each of the three independent variables on the vote in favor of these ballot measures. There are three independent variables, so we cannot create a single graph that displays the results. However, as one of the variables is dichotomous, we can show the results in just two graphs (the number of continuous independent variables).

Both of these graphs will have the vote in favor as the dependent variable (vertical axis). One of the two graphs will have percent religious as the primary independent variable, whereas the other will have the year passed as the primary independent variable. The chicken variable will be present in both graphs, signified by two separate curves, one where the ballot measure banned chickens and one where it did not (Figure 5.4).

The graphs illustrate the results of the model — this is their purpose. Although the graphs "illustrate the story," we must still "tell the story" of the graphics, including numbers from the prediction table (Table 5.4). The following paragraphs explain the graphics.

tell the story

> Both graphics show that the effect of adding a chicken ban to the referendum tends to reduce the vote in favor of the referendum. All things being equal, a ballot measure banning chickens will have 3.7% *fewer* people vote for it than a like measure not banning chickens ($s = 1.9988, t = -1.87, p = 0.0723$).
>
> The top graphic illustrates the effect of passing time on the proportion of the vote in favor of these referenda: As the year increases by one, the proportion voting in favor of the referendum decreases by 2% on average ($t = -5.62, p \ll 0.0001$).
>
> The bottom graphic shows the effect of religiosity on the ballot outcome: those kraj with higher levels of religiosity tend to vote in favor of these measures at a higher level than kraj with lower

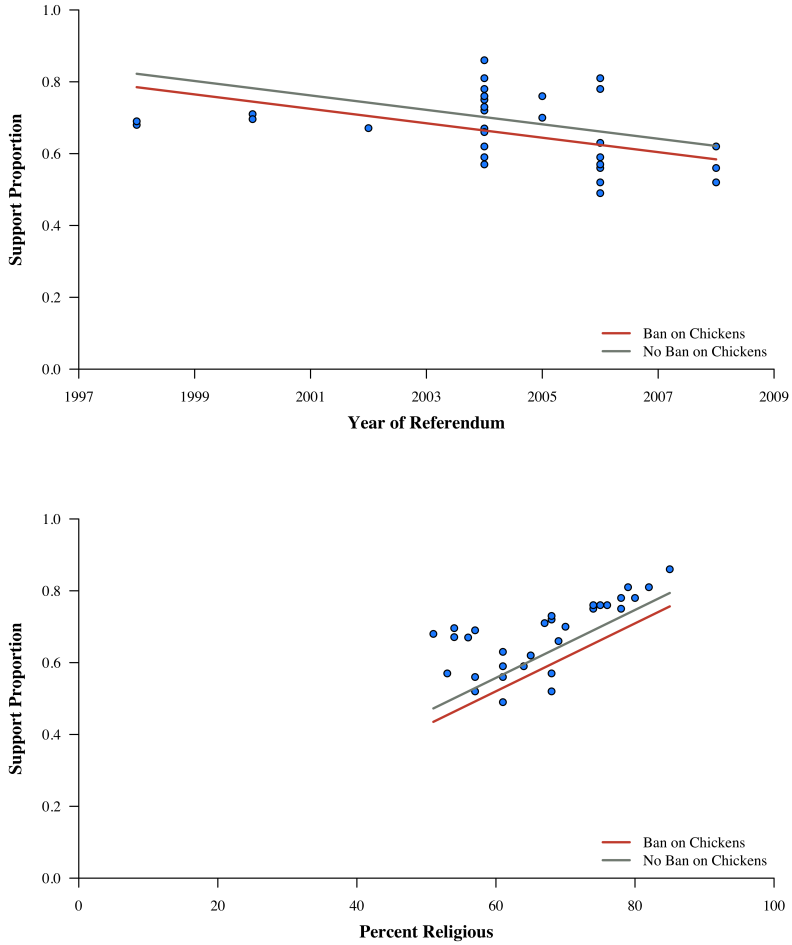


Figure 5.4: *Prediction graphs of our* `cows` *model. These graphs contain two independent variables plotted against the dependent variable, with the dichotomous independent variable included as separate lines. Note that the effect of each of the three independent variables is made manifest by these two graphs.*

> levels of religiosity. In fact, increasing the level of religiosity in the kraj by 1% will tend to increase the vote in favor of the ballot measure by 0.95% ($t = 8.80, p \ll 0.0001$).

Note the interweaving of the graphic discussion with concrete, numerical effects (and statistical significance in parentheses) from the prediction table. This combination aids the reader in interpreting the graphic(s) in terms of statistical language.

regression table

point prediction

5.3.7 Answering the Question*

Thus, we have a prediction of 42% of the voters will support the ballot measure. However, this is *not* an answer to the original question, which asked about the *probability* of the ballot measure passing. From a modeling standpoint, this probability depends on the coefficient estimates, which are just estimates of the true population value, and the standard errors, which are measures of our certainty in those estimates.

random variable

In the ordinary least squares method, those parameter estimates are random variables, since they are *functions of the data*. In other words, if we re-ran human history, the estimated effect would be different, since reality would be different. Furthermore, as these are random variables, they have an associated distribution — the normal distribution. In fact, the distribution of each parameter estimate is normal, with expected value equal to the estimate and standard deviation equal to the standard error. Thus, for example, the effect of `yearPassed` is $\hat{\beta}_1 \sim \mathcal{N}(\mu = -0.0201, \sigma = 0.0036)$; of `chicken`, $\hat{\beta}_2 \sim \mathcal{N}(\mu = -0.0373, \sigma = 0.0200)$; and of `pctRelig`, $\hat{\beta}_3 \sim \mathcal{N}(\mu = 0.0095, \sigma = 0.0011)$.

distribution of estimators

Monte Carlo

Let us leverage these facts to (virtually) re-run human history multiple times, get the parameter estimates for each history, and predict the outcome of the ballot measure in Děčín.[8] In other words, let us perform a Monte Carlo analysis. The steps are the same as with any Monte Carlo analysis we have done (Kennedy 2008). The only difference is what we do within the loop. Here, we draw random numbers from the appropriate distribution and calculate the predicted vote.

Before you look at the following algorithm, write your own and compare it to the one below:

1. Initialize variables
2. Perform loop
    a) Draw from the four distributions
    b) Predict the Děčín outcome
3. Calculate the number of times the ballot measure garnered more than 50% of the vote

[8] Note that this process assumes the parameter estimates are independent of each other. This is not the case. See Theorem 2.11. The effects are dependent on each other, as is the intercept. As such, treat this sub-section as a pedagogical exercise rather than a statistical exercise. There are a lot of questions dealt with here that help better understand things.

One can also store the random numbers inside the loop and predict outside the loop. Also, if the statistical program allows it, you can avoid the loop and just draw all the numbers at once. This last has the advantage of being *very* fast.

It is also the method I use here, in the `R` script:

```
# Initialize variables
  outcome <- numeric()
  trials  <- 1000000

# Coefficient estimates
  b.intc  <-  0.151221
  b.year  <- -0.020095
  b.cban  <- -0.037331
  b.rpct  <-  0.009452

# Coefficient standard errors
  s.intc  <-  0.065938
  s.year  <-  0.003577
  s.cban  <-  0.019988
  s.rpct  <-  0.001074

# Distributions (the "loop")
  e.intc  <- rnorm(trials, m=b.intc, s=s.intc)
  e.year  <- rnorm(trials, m=b.year, s=s.year)
  e.cban  <- rnorm(trials, m=b.cban, s=s.cban)
  e.rpct  <- rnorm(trials, m=b.rpct, s=s.rpct)
  outcome <- e.intc + e.year*9 + e.cban*0 + e.rpct*48
```

At this point, the variable `outcome` holds the proportion of people voting in favor of the ballot measure in one million simulated elections. To answer the question, we just need to determine the proportion of those elections in which the `outcome` is greater than 0.50: `mean(outcome>0.50)` will work.

Of course the numbers are nice, but a histogram may tell a better story. The following code will give a histogram simliar that in Figure 5.5.

```
hist(outcome, main="", xlab="Proportion Vote for Ballot Measure
    ", breaks=-1:99/100)
hist(outcome[outcome>0.50], main="", yaxt="n", breaks=-1:99/
    100, col=2, add=TRUE)
axis(1, at=0.50, labels="50%")
```

The histogram of the Děčín predictions is presented in Figure 5.5. Note that the expected outcome is still 42%, which we found above, but that

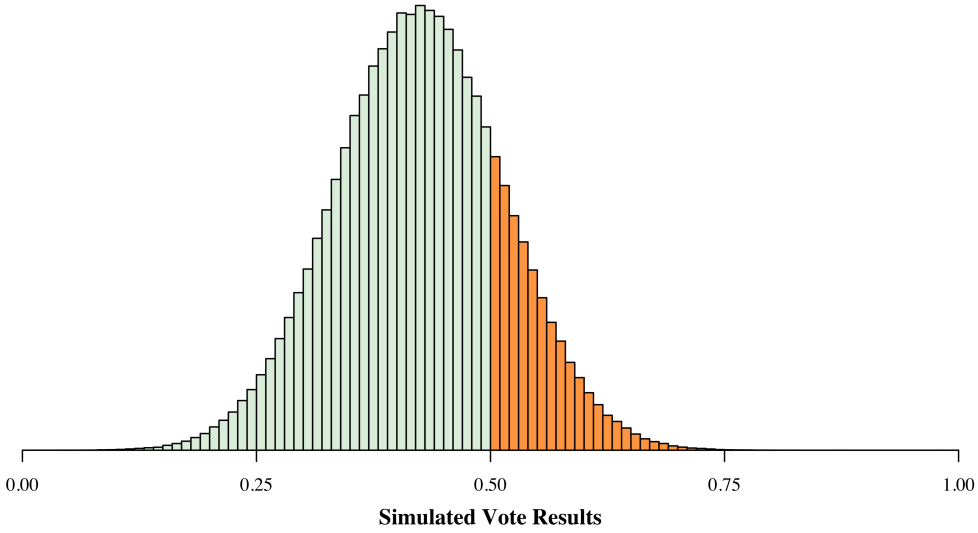


Figure 5.5: *Plot of the predicted vote outcomes from the Monte Carlo experiment described in the text. Note that, while the expected proportion of the vote in favor of the ballot measure is* 42%*, there is still a* 20% *chance of the ballot measure passing, given that our model is correct.*

confidence interval

there is a spread to that prediction the histogram makes manifest, which the single prediction did not. In fact, prior to this analysis, we may have concluded that there was no possibility that the ballot measure would pass in Děčín based on our model; now, we see that there is a 20% chance of the ballot measure passing.[9]

❧ ❧ ❧

Thus, we have an estimated answer to our original question. Given that our model is correct, there is approximately a 20% chance that the ballot measure to limit the number of cows in the city will pass in Děčín, with a point prediction of 42% in favor of the bill.

point estimate

The actual results of the 2009 ballot measure in Děčín was that the ballot measure passed with 53% of the vote. This result is well within the 95% prediction interval suggested by Figure 5.5. Also, the fact that the ballot measure passed should not be too surprising, since this model gave it a

[9] As with all statistical analysis, the *caveat* is that the model and the assumptions must be correct.

20% probability of passing. 20% is not a rare event by *any* stretch of the imagination.

5.3.8 A Fundamental Problem

There is a really big problem with these results, however. Run the following code and interpret it.

```
mean(outcome>1) + mean(outcome<0)
```

This is an important exercise: Always check that predictions make sense.

5.4: Conclusion

In this chapter, we performed full analyses, demonstrating the entire process.

Warning: *Again, be aware of the multiple comparisons issue discussed in Appendix S.6.2. It explains why you need to either adjust your p-value or your alpha level when performing multiple tests, such as when you are testing* both $\beta_0 = 4$ and $\beta_1 = 0$.

5.5: End-of-Chapter Materials

5.5.1 R Functions

In this chapter, we were introduced to many, many, many R functions that will be useful in regression. In fact, this chapter uses more R functions than any other chapter in this book. Here are the many.

Packages:

car This package provides several statistical tests used in the book "An R Companion to Applied Regression" by J. Fox and S. Weisberg. It is a great package that provides a lot of additional functionality for R.

lawstat This package provides several statistical tests used in law and public policy analysis. It provides the `runs.test` function for us.

lmtest This package provides many tests related to linear models. It provides an implementation of the Breusch-Pagan test, `bptest`, which tests for heteroskedasticity in the residuals.

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:

source(filename) This function runs an R script from a separate file. That file may be local or on the Internet.

runs.test(E, order) This alteration to the `lawstat` function tests whether the variable `E`, as ordered by `order` exhibits fit issues.

shapiroTest(E) This tests the null hypothesis that the variable `E` comes from a Normal distribution. It is based on the `shapiro.test` function in the basic R installation. It adds capabilities to test Normality in several groups.

lm(formula) This is the function that performs ordinary least squares estimation on linear models.

bptest(mod) This function from the `lmtest` package performs the Bresuch-Pagan test for heteroskedasticity.

confint(mod) This calculates confidence intervals for the parameters in ordinary least squares regression.

mean(x) This calculates the mean of a sample.

summary(x) This produces the six-number summary or a frequency table of the provided variable, depending on the type of variable.

summary.lm(mod) When applied to a linear model fit using either the `aov` function or the `lm` function, provides estimates of the effects of the numeric variables and the levels of the categorical variables in the model.

summary.aov(mod) When applied to a linear model fit using either the `aov` function or the `lm` function, provides estimates of the statistical significance of the variables in the model.

predict(mod) This predicts the values of the dependent variable at each point in the dataset *or* for the values specified.

fligner.test(formula) This tests for heteroskedasticity when the independent variable is categorical.

aov(formula) This function performs ordinary least squares estimation on linear models.

vif(model) This function calculates the variance inflation factor (VIF) for each of the independent variables in the model.

set.base(var,level) This `RFS` package function redefines the base category in the provided level. By default, the base category is the first according to the alphabet.

Probability:

set.seed(x) This sets the random number seed.

rexp(n, rate) This generates *n* random values from an Exponential distribution with the specified rate parameter.

rnorm(n, mean, sd) This generates *n* random values from a Normal distribution with specified mean and standard deviation. By default the mean is 0 and the standard deviation is 1.

runif(n, min, max) This generates *n* random values from a Uniform distribution with specified minimum and maximum values. By default, the minimum is 0 and the maximum is 1.

Mathematics:

head(x) This returns the first six values in the variable.

foot(x) This returns the last six values in the variable.

seq(from, to, by, length) This returns a vector of sequential values, where `by` indicates the step size and `length` specifies the vector length.

length(x) This calculates the length of a vector (variable), which is the sample size, *n*.

residuals(mod) This calculates the residuals in the model, which is the difference between the observed and the predicted.

Graphics:

qqnorm(x) This creates a Normal quantile-quantile plot for the given values.

qqline(x) This adds the diagonal line to the quantile-quantile plot.

overlay(x) This, from the `RFS` package, produces a histogram with a Normal curve overlaying it.

par(...) This sets parameters on the next graphic started. Look through the help page for this function to see all you can specify.

plot(x,y) This produces a scatter plot of the y-values against the x-values.

axis(side) When a plot is already drawn, this adds values along axis number `side`.

title(...) When a plot is already drawn, this adds the x- and y-labels.

lines(x,y) When a plot is already drawn, this draws lines between each subsequent (x, y) pair.

points(x,y) When a plot is already drawn, this draws points at each (x, y) pair.

Programming:

attach(dataframe) This allows you to access the variables in the `datarame` without having to prefix each with `dataframe$`.

library(package) This loads an external package that you have already installed on your computer. It allows access to all functions and data sets in the `package` package.

as.character(x) This changes the values in variable `x` to be characters.

as.numeric(x) This changes the values in variable `x` to be numbers.

5.5.2 Exercises

1. In the two panels in Figure 5.4, the lines of best fit do not go beyond the data. Why?

2. Section 5.3.8 mentioned that there was a really big problem with this analysis. Run the following code.

   ```
   mean(outcome>1) + mean(outcome<0)
   ```

   What value is given, what does it mean, and why does it imply there is something fundamentally wrong with the analysis?