10:38 Friday 27th November, 2015
See updates and corrections at `http://www.stat.cmu.edu/~cshalizi/mreg/`

Lecture 24–25: Weighted and Generalized Least Squares

36-401, Fall 2015, Section B

19 and 24 November 2015

Contents

1 Weighted Least Squares

When we use ordinary least squares to estimate linear regression, we (naturally) minimize the mean squared error:

$$MSE(\mathbf{b}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_{i\cdot}\beta)^2 \tag{1}$$

The solution is of course

$$\widehat{\beta}_{OLS} = (\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T\mathbf{y} \tag{2}$$

We could instead minimize the *weighted* mean squared error,

$$WMSE(\mathbf{b}, w_1, \ldots w_n) = \frac{1}{n}\sum_{i=1}^{n} w_i(y_i - \mathbf{x}_{i\cdot}\mathbf{b})^2 \tag{3}$$

This includes ordinary least squares as the special case where all the weights $w_i = 1$. We can solve it by the same kind of linear algebra we used to solve the ordinary linear least squares problem. If we write $\mathbf{w}$ for the matrix with the w_i on the diagonal and zeroes everywhere else, then

$$\begin{aligned} WMSE &= n^{-1}(\mathbf{y} - \mathbf{x}\mathbf{b})^T\mathbf{w}(\mathbf{y} - \mathbf{x}\mathbf{b}) && (4) \\ &= \frac{1}{n}\left(\mathbf{y}^T\mathbf{w}\mathbf{y} - \mathbf{y}^T\mathbf{w}\mathbf{x}\mathbf{b} - \mathbf{b}^T\mathbf{x}^T\mathbf{w}\mathbf{y} + \mathbf{b}^T\mathbf{x}^T\mathbf{w}\mathbf{x}\mathbf{b}\right) && (5) \end{aligned}$$

Differentiating with respect to $\mathbf{b}$, we get as the gradient

$$\nabla_{\mathbf{b}} WMSE = \frac{2}{n}\left(-\mathbf{x}^T\mathbf{w}\mathbf{y} + \mathbf{x}^T\mathbf{w}\mathbf{x}\mathbf{b}\right)$$

Setting this to zero at the optimum and solving,

$$\widehat{\beta}_{WLS} = (\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\mathbf{y} \tag{6}$$

But why would we want to minimize Eq. 3?

1. *Focusing accuracy.* We may care very strongly about predicting the response for certain values of the input — ones we expect to see often again, ones where mistakes are especially costly or embarrassing or painful, etc. — than others. If we give the points near that region big weights, and points elsewhere smaller weights, the regression will be pulled towards matching the data in that region.

2. *Discounting imprecision.* Ordinary least squares minimizes the squared error when the variance of the noise terms ϵ is constant over all observations, so we're measuring the regression function with the same precision elsewhere. This situation, of constant noise variance, is called **homoskedasticity**. Often however the magnitude of the noise is not constant, and the data are **heteroskedastic**.

When we have heteroskedasticity, ordinary least squares is no longer the optimal estimate — we'll see presently that other estimators can be unbiased and have smaller variance. If however we know the noise variance σ_i^2 at each measurement i, and set $w_i = 1/\sigma_i^2$, we get minimize the variance of estimation.

To say the same thing slightly differently, there's just no way that we can estimate the regression function as accurately where the noise is large as we can where the noise is small. Trying to give equal attention to all values of X is a waste of time; we should be more concerned about fitting well where the noise is small, and expect to fit poorly where the noise is big.

3. *Sampling bias.* In many situations, our data comes from a survey, and some members of the population may be more likely to be included in the sample than others. When this happens, the sample is a biased representation of the population. If we want to draw inferences about the population, it can help to give more weight to the kinds of data points which we've under-sampled, and less to those which were over-sampled. In fact, typically the weight put on data point i would be inversely proportional to the probability of i being included in the sample (exercise 1). Strictly speaking, if we are willing to believe that linear model is exactly correct, that there are no omitted variables, and that the inclusion probabilities p_i do not vary with y_i, then this sort of survey weighting is redundant (DuMouchel and Duncan, 1983). When those assumptions are not met — when there're non-linearities, omitted variables, or "selection on the dependent variable" — survey weighting is advisable, if we know the inclusion probabilities fairly well.

   The same trick works under the same conditions when we deal with "covariate shift", a change in the distribution of X. If the old probability density function was $p(x)$ and the new one is $q(x)$, the weight we'd want to use is $w_i = q(x_i)/p(x_i)$ (Quiñonero-Candela *et al.*, 2009). This can involve estimating both densities, or their ratio (topics we'll cover in 402).

4. *Doing something else.* There are a number of other optimization problems which can be transformed into, or approximated by, weighted least squares. The most important of these arises from **generalized linear models**, where the mean response is some nonlinear function of a linear predictor; we will look at them in 402.

In the first case, we decide on the weights to reflect our priorities. In the third case, the weights come from the optimization problem we'd really rather be solving. What about the second case, of heteroskedasticity?

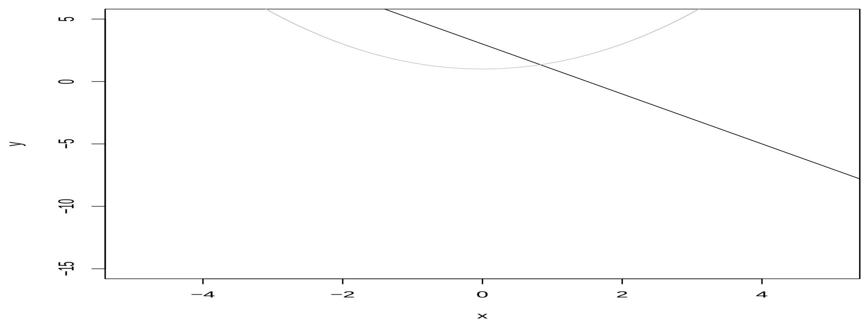


FIGURE 1: *Black line: Linear response function ($y = 3 - 2x$). Grey curve: standard deviation as a function of x ($\sigma(x) = 1 + x^2/2$). (Code deliberately omitted; can you reproduce this figure?)*

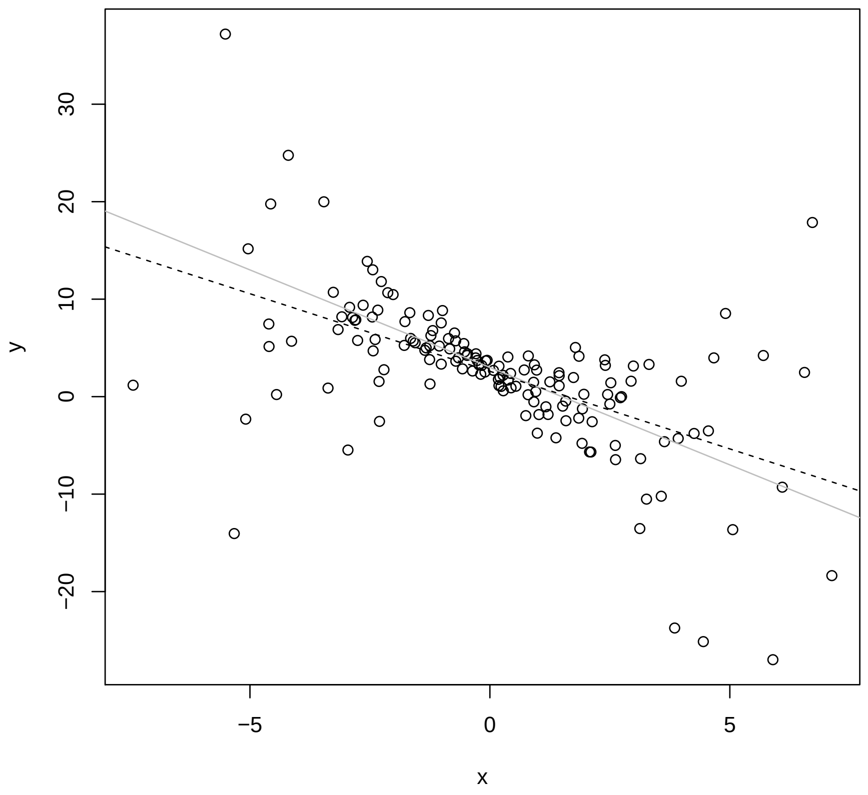
2 Heteroskedasticity

Suppose the noise variance is itself variable. For example, Figure 1 shows a simple linear relationship between the predictors X and the response Y, but also a nonlinear relationship between X and $\text{Var}[Y]$.

In this particular case, the ordinary least squares estimate of the regression line is $2.6 - 1.59x$, with R reporting standard errors in the coefficients of ± 0.53 and 0.19, respectively. Those are however calculated under the assumption that the noise is homoskedastic, which it isn't. And in fact we can see, pretty much, that there is heteroskedasticity — if looking at the scatter-plot didn't convince us, we could always plot the residuals against x, which we should do anyway.

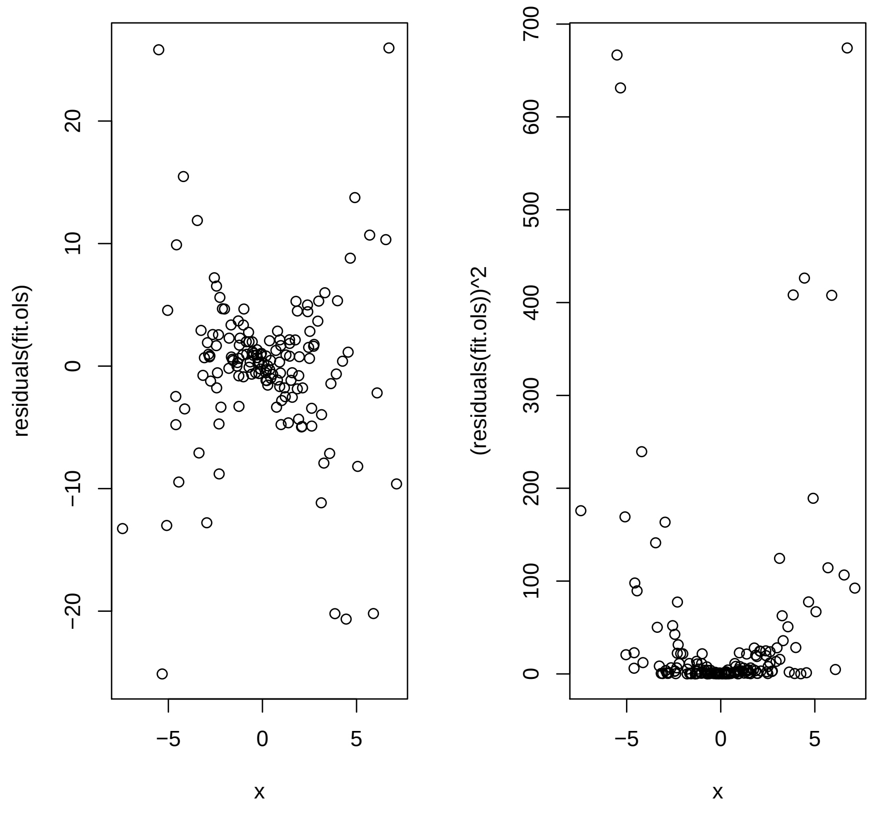
To see whether that makes a difference, let's re-do this many times with different draws from the same model (Figure 4).

Running `ols.heterosked.error.stats(100)` produces 10^4 random samples which all have the same x values as the first one, but different values of y, generated however from the same model. It then uses those samples to get the standard error of the ordinary least squares estimates. (Bias remains a non-issue.) What we find is the standard error of the intercept is only a little inflated (simulation value of 0.57 versus official value of 0.53), but the standard error of the slope is much larger than what R reports, 0.42 versus 0.19. Since the intercept is fixed by the need to make the regression line go through the center of the data, the real issue here is that our estimate of the slope is much less precise than ordinary least squares makes it out to be. Our estimate is still consistent, but not as good as it was when things were homoskedastic. Can we get back some of that efficiency?



```
# Plot the data
plot(x,y)
# Plot the true regression line
abline(a=3,b=-2,col="grey")
# Fit by ordinary least squares
fit.ols = lm(y~x)
# Plot that line
abline(fit.ols,lty="dashed")
```

FIGURE 2: *Scatter-plot of $n = 150$ data points from the above model. (Here X is Gaussian with mean 0 and variance 9.) Grey: True regression line. Dashed: ordinary least squares regression line.*



```
par(mfrow=c(1,2))
plot(x,residuals(fit.ols))
plot(x,(residuals(fit.ols))^2)
par(mfrow=c(1,1))
```

FIGURE 3: *Residuals (left) and squared residuals (right) of the ordinary least squares regression as a function of x. Note the much greater range of the residuals at large absolute values of x than towards the center; this changing dispersion is a sign of heteroskedasticity.*

```
# Generate more random samples from the same model and the same x values,
# but different y values
# Inputs: number of samples to generate
# Presumes: x exists and is defined outside this function
# Outputs: errors in linear regression estimates
ols.heterosked.example = function(n) {
  y = 3-2*x + rnorm(n,0,sapply(x,function(x){1+0.5*x^2}))
  fit.ols = lm(y~x)
  # Return the errors
  return(fit.ols$coefficients - c(3,-2))
}

# Calculate average-case errors in linear regression estimates (SD of
# slope and intercept)
# Inputs: number of samples per replication, number of replications (defaults
#         to 10,000)
# Calls: ols.heterosked.example
# Outputs: standard deviation of intercept and slope
ols.heterosked.error.stats = function(n,m=10000) {
  ols.errors.raw = t(replicate(m,ols.heterosked.example(n)))
  # transpose gives us a matrix with named columns
  intercept.sd = sd(ols.errors.raw[,"(Intercept)"])
  slope.sd = sd(ols.errors.raw[,"x"])
  return(list(intercept.sd=intercept.sd,slope.sd=slope.sd))
}
```

FIGURE 4: *Functions to generate heteroskedastic data and fit OLS regression to it, and to collect error statistics on the results.*

FIGURE 5: *Statistician (right) consulting the Oracle of Regression (left) about the proper weights to use to overcome heteroskedasticity.* (Image from http://en.wikipedia.org/wiki/Image:Pythia1.jpg.)

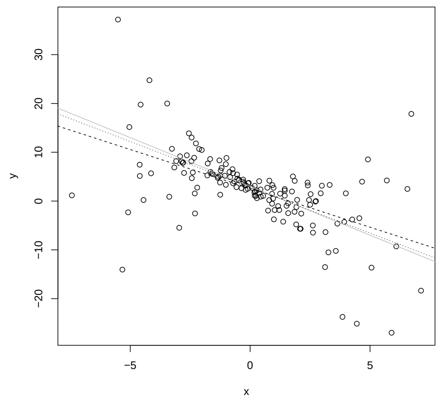
2.1 Weighted Least Squares as a Solution to Heteroskedasticity

Suppose we visit the Oracle of Regression (Figure 5), who tells us that the noise has a standard deviation that goes as $1 + x^2/2$. We can then use this to improve our regression, by solving the weighted least squares problem rather than ordinary least squares (Figure 6).

The estimated line is now $2.81 - 1.88x$, with reported standard errors of 0.27 and 0.17. Does this check out with simulation? (Figure 7.)

Unsurprisingly, yes. The standard errors from the simulation are 0.27 for the intercept and 0.17 for the slope, so R's internal calculations are working very well.

Why does putting these weights into WLS improve things?



```
# Plot the data
plot(x,y)
# Plot the true regression line
abline(a=3,b=-2,col="grey")
# Fit by ordinary least squares
fit.ols = lm(y~x)
# Plot that line
abline(fit.ols,lty="dashed")
fit.wls = lm(y~x, weights=1/(1+0.5*x^2))
abline(fit.wls,lty="dotted")
```

FIGURE 6: *Figure 2, plus the weighted least squares regression line (dotted).*

```r
### As previous two functions, but with weighted regression

# Generate random sample from model (with fixed x), fit by weighted least
# squares
# Inputs: number of samples
# Presumes: x fixed outside function
# Outputs: errors in parameter estimates
wls.heterosked.example = function(n) {
  y = 3-2*x + rnorm(n,0,sapply(x,function(x){1+0.5*x^2}))
  fit.wls = lm(y~x,weights=1/(1+0.5*x^2))
  # Return the errors
  return(fit.wls$coefficients - c(3,-2))
}

# Calculate standard errors in parameter estiamtes over many replications
# Inputs: number of samples per replication, number of replications (defaults
#         to 10,000)
# Calls: wls.heterosked.example
# Outputs: standard deviation of estimated intercept and slope
wls.heterosked.error.stats = function(n,m=10000) {
  wls.errors.raw = t(replicate(m,wls.heterosked.example(n)))
  # transpose gives us a matrix with named columns
  intercept.sd = sd(wls.errors.raw[,"(Intercept)"])
  slope.sd = sd(wls.errors.raw[,"x"])
  return(list(intercept.sd=intercept.sd,slope.sd=slope.sd))
}
```

FIGURE 7: *Linear regression of heteroskedastic data, using weighted least-squared regression.*

2.2 Some Explanations for Weighted Least Squares

Qualitatively, the reason WLS with inverse variance weights works is the following. OLS cares equally about the error at each data point.[1] Weighted least squares, naturally enough, tries harder to match observations where the weights are big, and less hard to match them where the weights are small. But each y_i contains not only the true regression function $m(x_i)$ but also some noise ϵ_i. The noise terms have large magnitudes where the variance is large. So we should want to have small weights where the noise variance is large, because there the data tends to be far from the true regression. Conversely, we should put big weights where the noise variance is small, and the data points are close to the true regression.

The qualitative reasoning in the last paragraph doesn't explain why the weights should be inversely proportional to the variances, $w_i \propto 1/\sigma_i^2$ — why not $w_i \propto 1/\sigma_i$, for instance? Look at the equation for the WLS estimates again:

$$\widehat{\beta}_{WLS} = (\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\mathbf{y} \tag{7}$$

Imagine holding $\mathbf{x}$ constant, but repeating the experiment multiple times, so that we get noisy values of $\mathbf{y}$. In each experiment, $Y_i = \mathbf{x}_{i\cdot}\beta + \epsilon_i$, where $\mathbb{E}[\epsilon_i|\mathbf{x}] = 0$ and $\mathrm{Var}[\epsilon_i|\mathbf{x}] = \sigma_i^2$. So

$$\begin{aligned} \widehat{\beta}_{WLS} &= (\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\mathbf{x}\beta + (\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\epsilon & (8)\\ &= \beta + (\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\epsilon & (9) \end{aligned}$$

Since $\mathbb{E}[\epsilon|\mathbf{x}] = 0$, the WLS estimator is unbiased:

$$\mathbb{E}\left[\widehat{\beta}_{WLS}|\mathbf{x}\right] = \beta \tag{10}$$

In fact, for the j^{th} coefficient,

$$\begin{aligned} \widehat{\beta}_j &= \beta_j + [(\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\epsilon]_j & (11)\\ &= \beta_j + \sum_{i=1}^{n} k_{ji}(w)\epsilon_i & (12) \end{aligned}$$

where in the last line I have bundled up $(\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}$ as a matrix $\mathbf{k}(w)$, with the argument to remind us that it depends on the weights. Since the WLS estimate is unbiased, it's natural to want it to also have a small variance, and

$$\mathrm{Var}\left[\widehat{\beta}_j\right] = \sum_{i=1}^{n} k_{ji}(w)\sigma_i^2 \tag{13}$$

It can be shown — the result is called the **generalized Gauss-Markov theorem** — that picking weights to minimize the variance in the WLS estimate

[1] Less anthropomorphically, the objective function in Eq. 1 has the same derivative with respect to the squared error at each point, $\frac{\partial MSE}{\partial e_i^2} = \frac{1}{n}$.

has the unique solution $w_i = 1/\sigma_i^2$. It does not require us to assume the noise is Gaussian, but the proof does need a few tricks (see §3).

A less general but easier-to-grasp result comes from adding the assumption that the noise around the regression line is Gaussian — that

$$Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \epsilon,\ \epsilon \sim \mathcal{N}(0, \sigma_x^2) \tag{14}$$

The log-likelihood is then (Exercise 2)

$$-\frac{n}{2}\ln 2\pi - \frac{1}{2}\sum_{i=1}^{n}\log \sigma_i^2 - \frac{1}{2}\sum_{i=1}^{n}\frac{(y_i - \mathbf{x}_{i\cdot}\mathbf{b})^2}{\sigma_i^2} \tag{15}$$

If we maximize this with respect to β, everything except the final sum is irrelevant, and so we minimize

$$\sum_{i=1}^{n}\frac{(y_i - \mathbf{x}_{i\cdot}\mathbf{b})^2}{\sigma_i^2} \tag{16}$$

which is just weighted least squares with $w_i = 1/\sigma_i^2$. So, if the probabilistic assumption holds, WLS is the efficient maximum likelihood estimator.

3 The Gauss-Markov Theorem

We've seen that when we do weighted least squares, our estimates of β are linear in $\mathbf{Y}$, and unbiased (Eq. 10):

$$\widehat{\beta_{WLS}} = (\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\mathbf{y} \tag{17}$$

$$\mathbb{E}\left[\widehat{\beta_{WLS}}\right] = \beta \tag{18}$$

What we'd like to show is that using the weights $w_i = 1/\sigma_i^2$ is somehow optimal. Like any optimality result, it is crucial to lay out carefully the range of possible alternatives, and the criterion by which those alternatives will be compared. The classical optimality result for estimating linear models is the **Gauss-Markov theorem**, which takes the range of possibilities to be *linear, unbiased estimators of* β, and the criterion to be *variance of the estimator*. I will return to both these choices at the end of this section.

Any linear estimator, say $\widetilde{\beta}$, could be written as

$$\widetilde{\beta} = \mathbf{q}\mathbf{y}$$

where $\mathbf{q}$ would be a $(p+1)\times n$ matrix, in general a function of $\mathbf{x}$, weights, the phase of the moon, etc. (For OLS, $\mathbf{q} = (\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T$.) For $\widetilde{\beta}$ to be an unbiased estimator, we must have

$$\mathbb{E}\left[\mathbf{q}\mathbf{Y}|\mathbf{x}\right] = \mathbf{q}\mathbf{x}\beta = \beta$$

Since this must hold for all β and all $\mathbf{x}$, we have to have $\mathbf{q}\mathbf{x} = \mathbf{I}$.[2] (Sanity check: this works for OLS.) The variance is then

$$\text{Var}\left[\mathbf{q}\mathbf{Y}|\mathbf{x}\right] = \mathbf{q}\text{Var}\left[\epsilon|\mathbf{x}\right]\mathbf{q}^T = \mathbf{q}\mathbf{\Sigma}\mathbf{q} \tag{19}$$

[2]This doesn't mean that $\mathbf{q} = \mathbf{x}^{-1}$; $\mathbf{x}$ doesn't have an inverse!

where I abbreviate the mouthful $\mathrm{Var}[\epsilon|\mathbf{x}]$ by $\mathbf{\Sigma}$. We could then try to differentiate this with respect to $\mathbf{q}$, set the derivative to zero, and solve, but this gets rather messy, since in addition to the complications of matrix calculus, we'd need to enforce the unbiasedness constraint $\mathbf{qx} = \mathbf{I}$ somehow.

Instead of the direct approach, we'll use a classic piece of trickery. Set

$$\mathbf{k} \equiv (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{\Sigma}^{-1}$$

which is the estimating matrix for weighted least squares. Now, whatever $\mathbf{q}$ might be, we can always write

$$\mathbf{q} = \mathbf{k} + \mathbf{r} \tag{20}$$

for some matrix $\mathbf{r}$. The unbiasedness constraint on $\mathbf{q}$ translates into

$$\mathbf{rx} = \mathbf{0}$$

because $\mathbf{kx} = \mathbf{I}$. Now we substitute Eq. 20 into Eq. 19:

$$\begin{aligned}
\mathrm{Var}\left[\widetilde{\beta}\right] &= (\mathbf{k}+\mathbf{r})\mathbf{\Sigma}(\mathbf{k}+\mathbf{r})^T && (21)\\
&= (\mathbf{k}+\mathbf{r})\mathbf{\Sigma}^{-1}(\mathbf{k}+\mathbf{r})^T && (22)\\
&= \mathbf{k\Sigma k}^T + \mathbf{r\Sigma k}^T + \mathbf{k\Sigma r}^T + \mathbf{r\Sigma r}^T && (23)\\
&= (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{\Sigma}\mathbf{\Sigma}^{-1}\mathbf{x}(\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1} && (24)\\
&\quad + \mathbf{r}\mathbf{\Sigma}\mathbf{\Sigma}^{-1}\mathbf{x}(\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}\\
&\quad + (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{\Sigma}\mathbf{r}^T\\
&\quad + \mathbf{r\Sigma r}^T\\
&= (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x}(\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1} && (25)\\
&\quad + \mathbf{rx}(\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1} + (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{r}^T\\
&\quad + \mathbf{r\Sigma r}^T\\
&= (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1} + \mathbf{r\Sigma r}^T && (26)
\end{aligned}$$

where the last step uses the fact that $\mathbf{rx} = \mathbf{0}$ (and so $\mathbf{x}^T\mathbf{r}^T = \mathbf{0}^T$).

Since $\mathbf{\Sigma}$ is a covariance matrix, it's positive definite, meaning that $a\mathbf{\Sigma}a^T \geq 0$ for any vector a. This applies in particular to the vector $\mathbf{r}_{i\cdot}$, i.e., the i^{th} row of $\mathbf{r}$. But

$$\mathrm{Var}\left[\tilde{\beta}_i\right] = (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}_{ii} + \mathbf{r}_{i\cdot}\mathbf{w_0}^{-1}\mathbf{r}_{i\cdot}^T$$

which must therefore be strictly larger than $(\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}_{ii}$, the variance we'd get from using weighted least squares.

We conclude that WLS, with the weight matrix $\mathbf{w}$ equal to the inverse variance matrix $\mathbf{\Sigma}^{-1}$, the least variance among all possible linear, unbiased estimators of the regression coefficients.

Notes:

FIGURE 8: *The Oracle may be out (left), or too creepy to go visit (right). What then? (Left, the sacred oak of the Oracle of Dodona, copyright 2006 by Flickr user "essayen", `http://flickr.com/photos/essayen/245236125/`; right, the entrace to the cave of the Sibyl of Cumæ, copyright 2005 by Flickr user "pverdicchio", `http://flickr.com/photos/occhio/17923096/`. Both used under Creative Commons license.)*

1. If all the noise variances are equal, then we've proved the optimality of OLS.
2. The theorem doesn't rule out linear, biased estimators with smaller variance. As an example, albeit a trivial one, $\mathbf{0y}$ is linear and has variance $\mathbf{0}$, but is (generally) very biased.
3. The theorem also doesn't rule out non-linear unbiased estimators of smaller variance. Or indeed non-linear biased estimators of even smaller variance.
4. The proof actually doesn't require the variance matrix to be diagonal.

4 Finding the Variance and Weights

All of this was possible because the Oracle told us what the variance function was. What do we do when the Oracle is not available (Figure 8)?

Sometimes we can work things out for ourselves, without needing an oracle.

- We know, empirically, the precision of our measurement of the response variable — we know how precise our instruments are, or the response is really an average of several measurements so we can use their standard deviations, etc.
- We know how the noise in the response must depend on the input variables. For example, when taking polls or surveys, the variance of the proportions we find should be inversely proportional to the sample size. So we can make the weights proportional to the sample size.

Both of these outs rely on kinds of background knowledge which are easier to get in the natural or even the social sciences than in many industrial applications. However, there are approaches for other situations which try to use the observed residuals to get estimates of the heteroskedasticity; this is the topic of the next section.

4.1 Variance Based on Probability Considerations

There are a number of situations where we can reasonably base judgments of variance, or measurement variance, on elementary probability.

Multiple measurements The easiest case is when our measurements of the response are actually averages over individual measurements, each with some variance σ^2. If some Y_i are based on averaging more individual measurements than others, there will be heteroskedasticity. The variance of the average of n_i uncorrelated measurements will be σ^2/n_i, so in this situation we could take $w_i \propto n_i$.

Binomial counts Suppose our response variable is a count, derived from a binomial distribution, i.e., $Y_i \sim \mathrm{Binom}(n_i, p_i)$. We would usually model p_i as a function of the predictor variables — at this level of statistical knowledge, a linear function. This would imply that Y_i had expectation $n_i p_i$, and variance $n_i p_i(1 - p_i)$. We would be well-advised to use this formula for the variance, rather than pretending that all observations had equal variance.

Proportions based on binomials If our response variable is a proportion based on a binomial, we'd see an expectation value of p_i and a variance of $\frac{p_i(1-p_i)}{n_i}$. Again, this is not equal across different values of n_i, or for that matter different values of p_i.

Poisson counts Binomial counts have a hard upper limit, n_i; if the upper limit is immense or even (theoretically) infinite, we may be better off using a Poisson distribution. In such situations, the mean of the Poisson λ_i will be a (possibly-linear) function of the predictors, and the variance will *also* be equal to λ_i.

Other counts The binomial and Poisson distributions rest on independence across "trials" (whatever those might be). There are a range of discrete probability models which allow for correlation across trials (leadings to more or less variance). These may, in particular situations, be more appropriate.

4.1.1 Example: The Economic Mobility Data

The data set on economic mobility we've used in a number of assignments and examples actually contains a bunch of other variables in addition to the covariates we've looked at (short commuting times and latitude and longitude). While reserving the full data set for later use, let's look one of the additional covariates, namely population.

To see why this might be relevant, recall that our response variable is the *fraction* of children who, in each community, were born into the lowest 20% of the income distribution during 1980–1982 and nonetheless make it into the top 20% by age 30, we're looking at a proportion. Different communities will have had different numbers of children born in the relevant period, generally proportional to their total population. Treating the observed fraction for New York City as being just as far from its *expected* rate of mobility as that for Piffleburg, WI is asking for trouble.

Once we have population, there is a very notable pattern: the most extreme levels of mobility are all for very small communities (Figure 9).

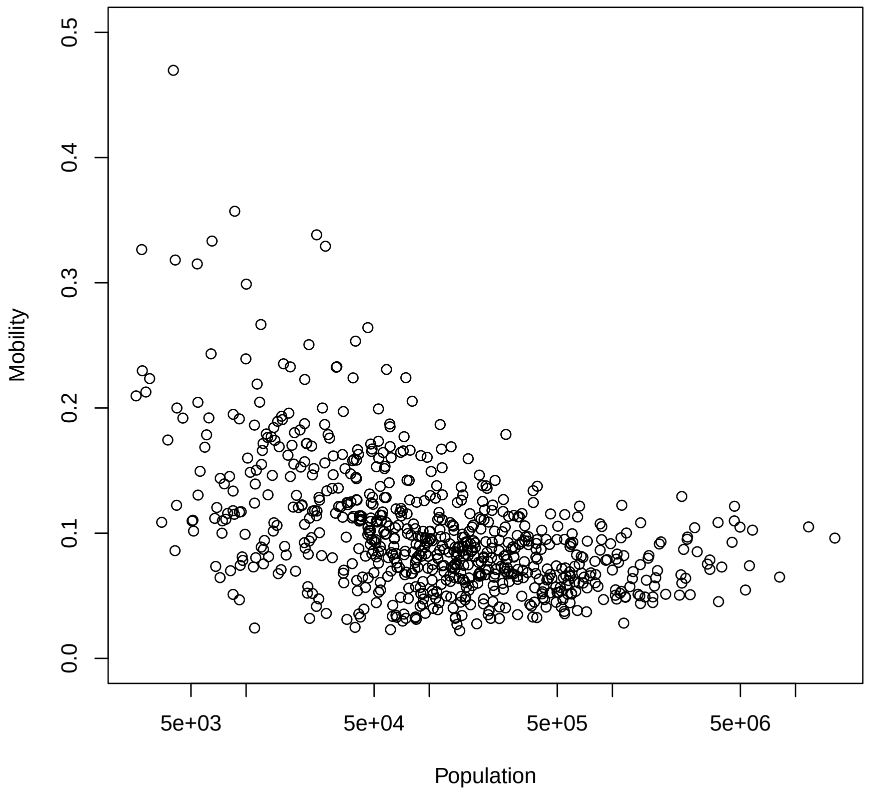
While we do not know the exact number of children for each community, it is not unreasonable to take that as proportional to the total population. The binomial standard error in the observed fraction will therefore be $\propto \sqrt{\frac{p_i(1-p_i)}{n_i}}$.

```
mobility$MobSE <- with(mobility, sqrt(Mobility*(1-Mobility)/Population))
```

Let us now plot the rate of economic mobility against the fraction of workers with short commutes, and decorate it with error bars reflecting these standard errors (Figure 10).

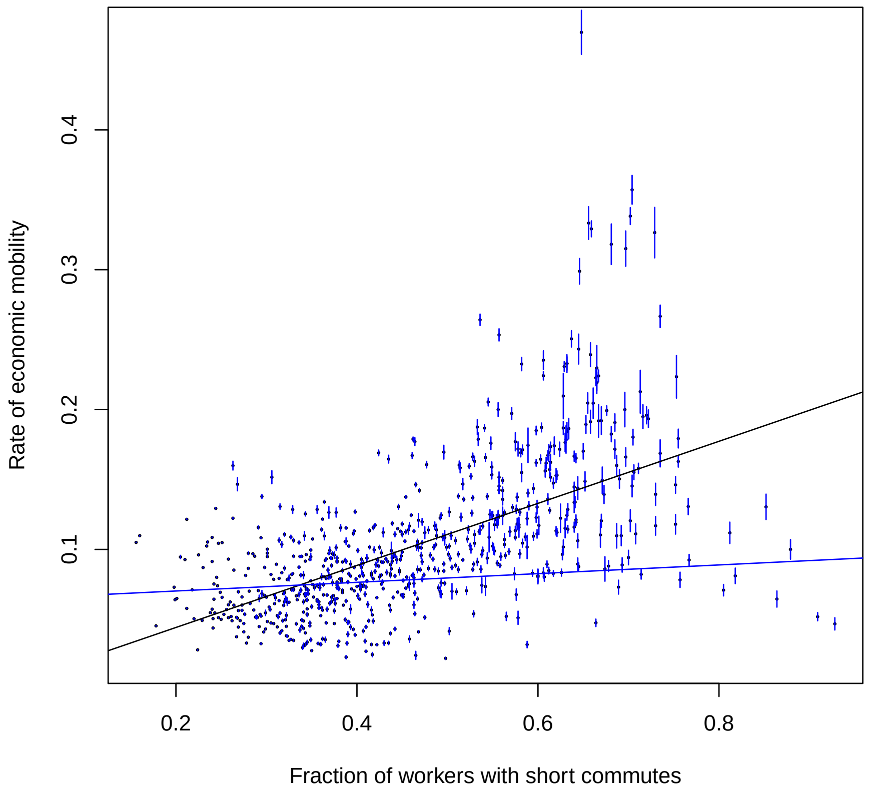
Now, there are reasons why this is not necessarily the last word on using weighted least squares here. One is that if we actually believed our model, we should be using the *predicted* mobility as the p_i in $\sqrt{\frac{p_i(1-p_i)}{n_i}}$, rather than the observed mobility. Another is that the binomial model assumes *independence* across "trials" (here, children). But, by definition, at most, and at least, 20% of the population ends up in the top 20% of the income distribution[3]. It's fairly clear, however, that simply *ignoring* differences in the sizes of communities is unwise.

[3]Cf. Gore Vidal: "It is not enough to succeed; others must also fail."



```
mobility <- read.csv("http://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/24--25/mobility2.csv")
plot(Mobility ~ Population, data=mobility, log="x", ylim=c(0,0.5))
```

FIGURE 9: *Rate of economic mobility plotted against population, with a logarithmic scale on the latter, horizontal axis. Notice decreasing spread at larger population.*



```
plot(Mobility ~ Commute, data=mobility,
     xlab="Fraction of workers with short commutes",
     ylab="Rate of economic mobility", pch=19, cex=0.2)
with(mobility, segments(x0=Commute, y0=Mobility+2*MobSE,
                        x1=Commute, y1=Mobility-2*MobSE, col="blue"))
mob.lm <- lm(Mobility ~ Commute, data=mobility)
mob.wlm <- lm(Mobility ~ Commute, data=mobility, weight=1/MobSE^2)
abline(mob.lm)
abline(mob.wlm, col="blue")
```

FIGURE 10: *Mobility versus the fraction of workers with short commute, with* ± 2 *standard deviation error bars (vertical blue bars), and the OLS linear fit (black line) and weighted least squares (blue line). Note that the error bars for some larger communities are smaller than the diameter of the dots.*

5 Conditional Variance Function Estimation

Remember that there are two equivalent ways of defining the variance:

$$\mathrm{Var}\left[X\right] = \mathbb{E}\left[X^2\right] - \left(\mathbb{E}\left[X\right]\right)^2 = \mathbb{E}\left[\left(X - \mathbb{E}\left[X\right]\right)^2\right] \tag{27}$$

The latter is more useful for us when it comes to estimating variance functions. We have already figured out how to estimate means — that's what all this previous work on smoothing and regression is for — and the deviation of a random variable from its mean shows up as a residual.

There are two generic ways to estimate conditional variances, which differ slightly in how they use non-parametric smoothing. We can call these the **squared residuals method** and the **log squared residuals method**. Here is how the first one goes.

1. Estimate $m(x)$ with your favorite regression method, getting $\hat{m}(x)$.
2. Construct the **squared residuals**, $u_i = \left(y_i - \hat{m}(x_i)\right)^2$.
3. Use your favorite *non-parametric* method to estimate the conditional mean of the u_i, call it $\widehat{q}(x)$.
4. Predict the variance using $\widehat{\sigma}^2_x = \widehat{q}(x)$.

The log-squared residuals method goes very similarly.[4]

1. Estimate $m(x)$ with your favorite regression method, getting $\hat{m}(x)$.
2. Construct the **log squared residuals**, $z_i = \log\left(y_i - \hat{m}(x_i)\right)^2$.
3. Use your favorite *non-parametric* method to estimate the conditional mean of the z_i, call it $\hat{s}(x)$.
4. Predict the variance using $\widehat{\sigma}^2_x = \exp\widehat{s}(x)$.

The quantity $y_i - \hat{m}(x_i)$ is the i^{th} residual. If $\widehat{m} \approx m$, then the residuals should have mean zero. Consequently the variance of the residuals (which is what we want) should equal the expected squared residual. So squaring the residuals makes sense, and the first method just smoothes these values to get at their expectations.

What about the second method — why the log? Basically, this is a convenience — squares are necessarily non-negative numbers, but lots of regression methods don't easily include constraints like that, and we really don't want to predict negative variances.[5] Taking the log gives us an unbounded range for the regression.

[4]I learned it from Wasserman (2006, pp. 87–88).

[5]Occasionally people do things like claiming that gene differences explains more than 100% of the variance in some psychological trait, and so environment and up-bringing contribute negative variance. Some of them — like Alford *et al.* (2005) — even say this with a straight face.

Strictly speaking, we don't need to use non-parametric smoothing for either method. If we had a parametric model for σ^2_x, we could just fit the parametric model to the squared residuals (or their logs). But even if you think you know what the variance function should look like it, why not check it?

We came to estimating the variance function because of wanting to do weighted least squares, but these methods can be used more generally. It's often important to understand variance in its own right, and this is a general method for estimating it. Our estimate of the variance function depends on first having a good estimate of the regression function

5.1 Iterative Refinement of Mean and Variance: An Example

The estimate $\widehat{\sigma}^2_x$ depends on the initial estimate of the regression function $\hat{m}(x)$. But, as we saw when we looked at weighted least squares, taking heteroskedasticity into account can change our estimates of the regression function. This suggests an iterative approach, where we alternate between estimating the regression function and the variance function, using each to improve the other. That is, we take either method above, and then, once we have estimated the variance function $\widehat{\sigma}^2_x$, we re-estimate $\hat{m}$ using weighted least squares, with weights inversely proportional to our estimated variance. Since this will generally change our estimated regression, it will change the residuals as well. Once the residuals have changed, we should re-estimate the variance function. We keep going around this cycle until the change in the regression function becomes so small that we don't care about further modifications. It's hard to give a strict guarantee, but *usually* this sort of iterative improvement will converge.

Let's apply this idea to our example. Figure 3b already plotted the residuals from OLS. Figure 11 shows those squared residuals again, along with the true variance function and the estimated variance function.

The OLS estimate of the regression line is not especially good ($\widehat{\beta}_0 = 2.6$ versus $\beta_0 = 3$, $\widehat{\beta}_1 = -1.59$ versus $\beta_1 = -2$), so the residuals are systematically off, but it's clear from the figure that spline smoothing of the squared residuals is picking up on the heteroskedasticity, and getting a pretty reasonable picture of the variance function.

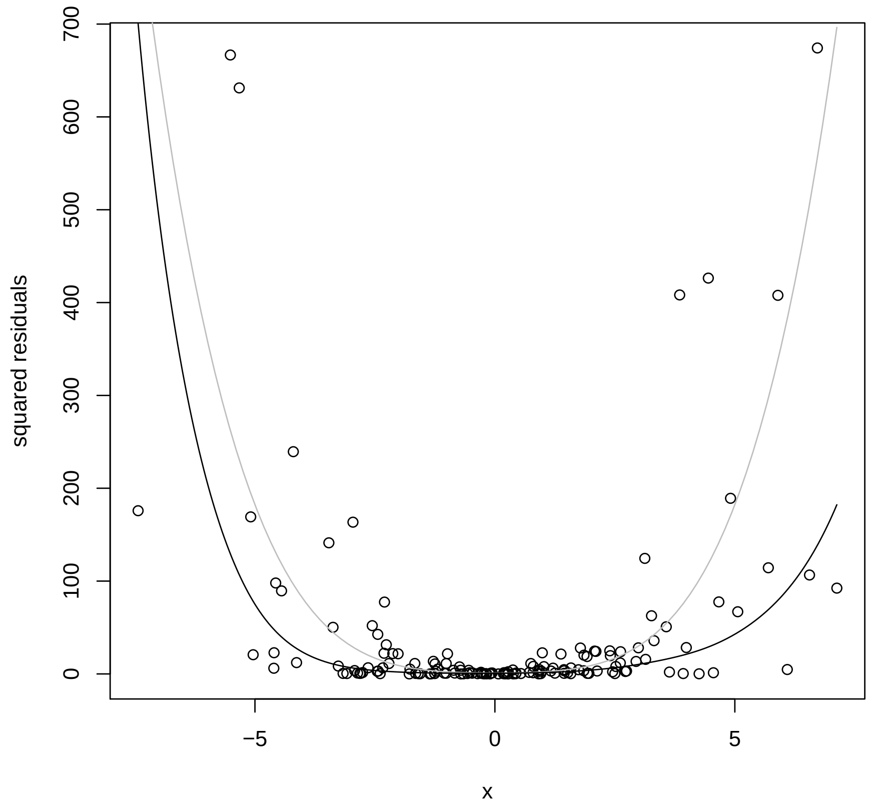
Now we use the estimated variance function to re-estimate the regression line, with weighted least squares.

```
fit.wls1 <- lm(y~x,weights=1/exp(var1$y))
coefficients(fit.wls1)

## (Intercept)           x
##    2.037829   -1.813567

var2 <- smooth.spline(x=x, y=log(residuals(fit.wls1)^2), cv=TRUE)
```

The slope has changed substantially, and in the right direction (Figure 12a). The residuals have also changed (Figure 12b), and the new variance function is



```
plot(x,residuals(fit.ols)^2,ylab="squared residuals")
curve((1+x^2/2)^2,col="grey",add=TRUE)
var1 <- smooth.spline(x=x, y=log(residuals(fit.ols)^2), cv=TRUE)
grid.x <- seq(from=min(x),to=max(x),length.out=300)
lines(grid.x, exp(predict(var1,x=grid.x)$y))
```

FIGURE 11: *Points: actual squared residuals from the OLS line. Grey curve: true variance function,* $\sigma^2_x = (1 + x^2/2)^2$*. Black curve: spline smoothing of the squared residuals.*

closer to the truth than the old one.

Since we have a new variance function, we can re-weight the data points and re-estimate the regression:

```
fit.wls2 <- lm(y~x,weights=1/exp(var2$y))
coefficients(fit.wls2)

## (Intercept)           x
##    2.115729   -1.867144

var3 <- smooth.spline(x=x, y=log(residuals(fit.wls2)^2), cv=TRUE)
```

Since we know that the true coefficients are 3 and -2, we know that this is moving in the right direction. If I hadn't told you what they were, you could still observe that the difference in coefficients between `fit.wls1` and `fit.wls2` is smaller than that between `fit.ols` and `fit.wls1`, which is a sign that this is converging.

I will spare you the plot of the new regression and of the new residuals. When we update a few more times:

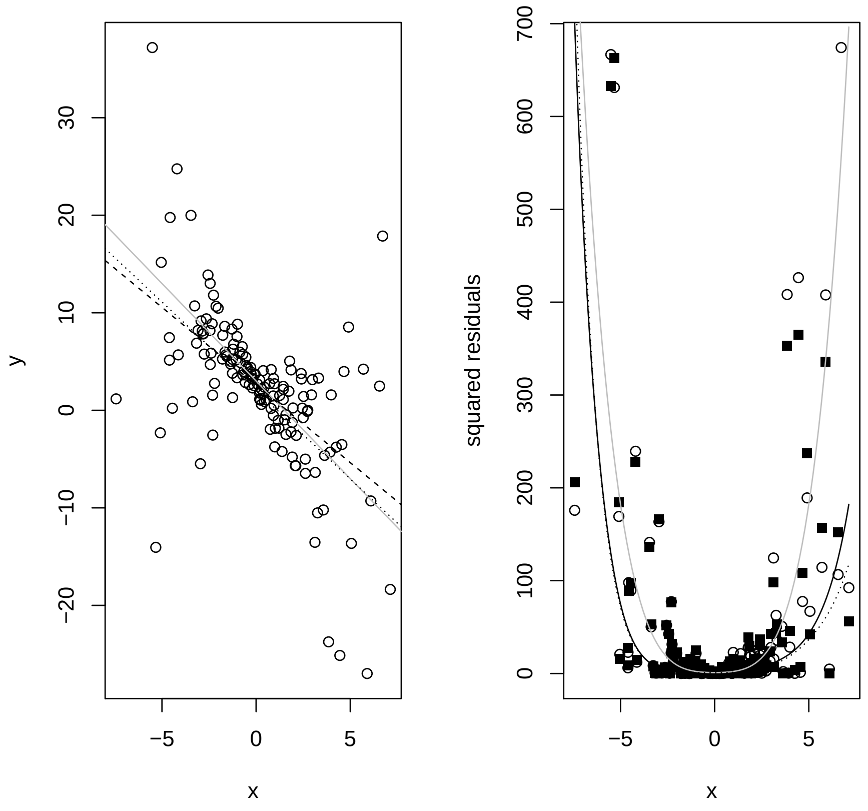
```
fit.wls3 <- lm(y~x,weights=1/exp(var3$y))
coefficients(fit.wls3)

## (Intercept)           x
##    2.113376   -1.857256

var4 <- smooth.spline(x=x, y=log(residuals(fit.wls3)^2), cv=TRUE)
fit.wls4 <- lm(y~x,weights=1/exp(var4$y))
coefficients(fit.wls4)

## (Intercept)           x
##    2.121188   -1.861435
```

By now, the coefficients of the regression are changing relatively little, and we only have 150 data points, so the imprecision from a limited sample surely swamps the changes we're making, and we might as well stop.



```
fit.wls1 <- lm(y~x,weights=1/exp(var1$y))
par(mfrow=c(1,2))
plot(x,y)
abline(a=3,b=-2,col="grey")
abline(fit.ols,lty="dashed")
abline(fit.wls1,lty="dotted")
plot(x,(residuals(fit.ols))^2,ylab="squared residuals")
points(x,residuals(fit.wls1)^2,pch=15)
lines(grid.x, exp(predict(var1,x=grid.x)$y))
var2 <- smooth.spline(x=x, y=log(residuals(fit.wls1)^2), cv=TRUE)
curve((1+x^2/2)^2,col="grey",add=TRUE)
lines(grid.x, exp(predict(var2,x=grid.x)$y),lty="dotted")
par(mfrow=c(1,1))
```

FIGURE 12: *Left: As in Figure 2, but with the addition of the weighted least squares regression line (dotted), using the estimated variance from Figure 11 for weights. Right: As in Figure 11, but with the addition of the residuals from the WLS regression (black squares), and the new estimated variance function (dotted curve).*

Manually going back and forth between estimating the regression function and estimating the variance function is tedious. We could automate it with a function, which would look something like this:

```
iterative.wls <- function(x,y,tol=0.01,max.iter=100) {
  iteration <- 1
  old.coefs <- NA
  regression <- lm(y~x)
  coefs <- coefficients(regression)
  while (is.na(old.coefs) ||
         ((max(abs(coefs - old.coefs)) > tol) && (iteration < max.iter))) {
    variance <- smooth.spline(x=x, y=log(residuals(regression)^2), cv=TRUE)
    old.coefs <- coefs
    iteration <- iteration+1
    regression <- lm(y~x,weights=1/exp(variance$y))
    coefs <- coefficients(regression)
  }
  return(list(regression=regression,variance=variance,iterations=iteration))
}
```

This starts by doing an unweighted linear regression, and then alternates between WLS for the getting the regression and spline smoothing for getting the variance. It stops when no parameter of the regression changes by more than `tol`, or when it's gone around the cycle `max.iter` times.[6] This code is a bit too inflexible to be really "industrial strength" (what if we wanted to use a data frame, or a more complex regression formula?), but shows the core idea.

6 Correlated Noise and Generalized Least Squares

Sometimes, we might believe the right model is (in matrix form)

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon \tag{28}$$

$$\mathbb{E}\left[\epsilon|\mathbf{X}\right] = \mathbf{0} \tag{29}$$

$$\mathrm{Var}\left[\epsilon|\mathbf{X}\right] = \mathbf{\Sigma} \tag{30}$$

where the matrix $\mathbf{\Sigma}$ is *not* diagonal. The off-diagonal entries represent covariance in the noise terms, $\mathrm{Cov}\left[\epsilon_i, \epsilon_j\right] = \Sigma_{ij}$. In fact, we should think this is the right model more often than the "usual" linear regression model, which is the special case where $\mathbf{\Sigma} = \sigma^2\mathbf{I}$. There is, after all, no reason in general considerations of probability theory or mathematical modeling to *expect* that fluctuations around a linear model will be uncorrelated. How might we nonetheless estimate β?

One approach is to try to make the noise disappear, by transforming the variables. Suppose we know $\mathbf{\Sigma}$. (We'll come back to where such knowledge

[6]The condition in the `while` loop is a bit complicated, to ensure that the loop is executed at least once. Some languages have an `until` control structure which would simplify this.

might come from later.) Because $\mathbf{\Sigma}$ is a variance matrix, we know it is square, symmetric, and positive-definite. This is enough to guarantee[7] that there is another square matrix, say $\mathbf{s}$, where $\mathbf{s}\mathbf{s}^T = \mathbf{\Sigma}$, as it were $\mathbf{s} = \sqrt{\mathbf{\Sigma}}$. I bring this fact up because we can use this to make the correlations in the noise go away.

Go back to our model equation, and multiply everything from the left by $\mathbf{s}^{-1}$.

$$\mathbf{s}^{-1}\mathbf{Y} = \mathbf{s}^{-1}\mathbf{X}\beta + \mathbf{s}^{-1}\epsilon$$

This looks like a linear regression of $\mathbf{s}^{-1}\mathbf{Y}$ on $\mathbf{s}^{-1}\mathbf{X}$, with the same coefficients β as our original regression. However, we have improved the properties of the noise. The noise is still zero in expectation,

$$\mathbb{E}\left[\mathbf{s}^{-1}\epsilon|\mathbf{X}\right] = \mathbf{s}^{-1}\mathbf{0} = \mathbf{0}$$

but the covariance has gone away, and all the noise terms have equal variance:

$$\begin{aligned}
\mathrm{Var}\left[\mathbf{s}^{-1}\epsilon|\mathbf{x}\right] &= \mathbf{s}^{-1}\mathrm{Var}\left[\epsilon|\mathbf{x}\right]\mathbf{s}^{-T} && (31)\\
&= \mathbf{s}^{-1}\mathbf{\Sigma}\mathbf{s}^{-T} && (32)\\
&= \mathbf{s}^{-1}\mathbf{s}\mathbf{s}^T\mathbf{s}^{-T} && (33)\\
&= \mathbf{I} && (34)
\end{aligned}$$

(This multiplication by $\mathbf{s}^{-1}$ is the equivalent, for random vectors, of dividing a random variable by its standard deviation, to get something with variance 1.)

To sum up, if we know $\mathbf{\Sigma}$, we can estimate β by doing an ordinary least squares regression of $\mathbf{s}^{-1}\mathbf{Y}$ on $\mathbf{s}^{-1}\mathbf{X}$. The estimate is

$$\begin{aligned}
\widehat{\beta} &= ((\mathbf{s}^{-1}\mathbf{x})^T\mathbf{s}^{-1}\mathbf{x})^{-1}(\mathbf{s}^{-1}\mathbf{x})^T\mathbf{s}^{-1}\mathbf{y} && (35)\\
&= (\mathbf{x}^T\mathbf{s}^{-T}\mathbf{s}^{-1}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{s}^{-T}\mathbf{s}^{-1}\mathbf{y} && (36)\\
&= (\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{y} && (37)
\end{aligned}$$

This looks just like our weighted least squares estimate, only with $\mathbf{\Sigma}^{-1}$ in place of $\mathbf{w}$.

6.1 Generalized Least Squares

This resemblance is no mere coincidence. We can write the WLS problem as that of minimizing $(\mathbf{y} - \mathbf{x}\beta)^T\mathbf{w}(\mathbf{y} - \mathbf{x}\beta)$, for a diagonal matrix $\mathbf{w}$. Suppose we try instead to minimize

$$(\mathbf{y} - \mathbf{x}\beta)^T\mathbf{w}(\mathbf{y} - \mathbf{x}\beta)$$

for a *non-diagonal*, but still symmetric and positive-definite, matrix $\mathbf{w}$. This is called a **generalized least squares** (GLS) problem. Every single step we went through before is still valid, because none of it rested on $\mathbf{w}$ being diagonal, so

$$\widehat{\beta}_{GLS} = (\mathbf{x}^T\mathbf{w}\mathbf{x})^{-1}\mathbf{x}^T\mathbf{w}\mathbf{y} \tag{38}$$

[7]Here's one way to do it: invoke the "spectral" or "eigendecomposition" theorem, to write $\mathbf{\Sigma} = \mathbf{v}\lambda\mathbf{v}^T$, where $\mathbf{v}$ is the matrix whose columns are the eigenvectors of $\mathbf{\Sigma}$, and λ is the diagonal matrix of the eigenvalues of $\mathbf{\Sigma}$. Then if we set $\mathbf{s} = \mathbf{v}\sqrt{\lambda}$, we'd have $\mathbf{\Sigma} = \mathbf{s}\mathbf{s}^T$, as desired.

What we have just seen is that if we set $\mathbf{w} = \mathbf{\Sigma}^{-1}$, we also get this solution when we transform the variables so as to de-correlate the noise, and then do ordinary least squares. This should at least make it *plausible* that this is a good way to estimate β in the face of correlated noise.

To go beyond plausibility, refer back to §3. At no point in our reasoning did we actually rely on $\text{Var}[\epsilon|\mathbf{x}]$ being diagonal. It follows that if we set $\mathbf{w} = \text{Var}[\epsilon|\mathbf{x}]^{-1}$, we get the linear, unbiased estimator of minimum variance. If we believe that the noise is Gaussian, then this is also the maximum likelihood estimator.

6.2 Where Do the Covariances Come From?

The soundest way to estimate a covariance would be to repeat the experiment many times, under identical conditions. This corresponds to using repeated measurements to estimate variances. It's simple, it works when we can do it, and there is accordingly little to say about it. Except: there are few situations where we can do it.

When we wanted to estimate the variance function, we could take all the squared residuals for values of x_i around a given x and use that as an estimate of $\sigma^2(x)$. This option is not available to us when we are looking at covariances.

If our measurements are spread out over time or space, it's natural to suppose that there is more covariance between nearby observations than between remote ones. A stronger but more delicate assumption is that of **stationarity**, that the covariance between an observation taken at time 0 and time h is the same as the covariance between time t and time $t + h$, whatever t might be. (And similarly for spatial stationarity.) Call the covariance in at this **lag** or **separation** $\gamma(h)$. We can estimate it by taking pairs of observations where the separation is approximately h, and averaging the products of their residuals.

It is common (though perhaps not wise) to make even stronger assumptions, such as that the covariance decays exponentially with distance, $\gamma(h) = \gamma(0)\rho^h$ or $\gamma(h) = \gamma(0)e^{-h/\tau}$. When we can believe such assumptions, they let us estimate the parameters of the covariance function using the sample covariances across all lags. The estimated covariance function, using all of that data, is much more stable than having many separate sample covariances, one for each lag. Even if the assumptions are, strictly, false, the stability that comes from forcing all the covariances to follow a common model can be desirable, on bias-variance grounds.

7 WLS and GLS vs. Specification Errors

When you find that your residuals from an initial model have non-constant variance or are correlated with each other, there are (at least) two possible explanations. One is that the fluctuations around the regression line really are heteroskedastic and/or correlated. In that case, you should try to model that variance and those correlations, and use WLS or GLS. The other explanation is that something is wrong with your model. If there's an important predictor variable which is just missing from your model, for example, then its contribution to the response will be part of your residuals. If that omitted variable is larger in some parts of the data than in others, or if the omitted variable has correlations, then that will make your residuals change in magnitude and be correlated. More subtly, having the wrong functional form for a variable you do include can produce those effects as well.

8 Exercises

1. Imagine we are trying to estimate the mean value of Y from a large population. We observe n members of the population, with individual i being included in our sample with a probability proportional to π_i. Show that the sample mean $n^{-1}\sum_{i=1}^{n} y_i$ is *not* a consistent estimator of $\mathbb{E}\left[Y\right]$ unless all the π_i are equal. Show that $\left(\sum_{i=1}^{n} y_i/\pi_i\right) / \sum_{i'=1}^{n} 1/\pi_{i'}$ *is* a consistent estimator of $\mathbb{E}\left[Y\right]$.
2. Show that the model of Eq. 14 has the log-likelihood given by Eq. 15
3. Do the calculus to verify Eq. 6.
4. Is $w_i = 1$ a necessary as well as a sufficient condition for Eq. 3 and Eq. 1 to have the same minimum?
5. §2.2 showed that WLS gives better parameter estimates than OLS when there is heteroskedasticity, and we know and use the variance. Modify the code for to see which one has better generalization error.

References

Alford, J. R., C. L. Funk and J. R. Hibbibng (2005). "Are Political Orientations Genetically Transmitted?" *American Political Science Review*, **99**: 153–167.

DuMouchel, William H. and Greg J. Duncan (1983). "Using Sample Survey Weights in Multiple Regression Analyses of Stratified Samples." *Journal of the American Statistical Association*, **78**: 535–543. URL `http://www.jstor.org/stable/2288115`.

Quiñonero-Candela, Joaquin, Masashi Sugiyama, Anton Schwaighofer and Neil D. Lawrence (eds.) (2009). *Dataset Shift in Machine Learning*. Cambridge, Massachusetts: MIT Press.

Wasserman, Larry (2006). *All of Nonparametric Statistics*. Berlin: Springer-Verlag.