Chapter 5

The Method of Maximum Likelihood for Simple Linear Regression

5.1 Recapitulation

We introduced the method of maximum likelihood for simple linear regression in Chapter 3. Let's review.

We start with the statistical model, which is the Gaussian-noise simple linear regression model, defined as follows:

1. The distribution of X is arbitrary (and perhaps X is even non-random).

2. If $X = x$, then $Y = \beta_0 + \beta_1 x + \epsilon$, for some constants ("coefficients", "parameters") β_0 and β_1, and some random noise variable ϵ.

3. $\epsilon \sim N(0, \sigma^2)$, and is independent of X.

4. ϵ is independent across observations.

A consequence of these assumptions is that the response variable Y is independent across observations, conditional on the predictor X, i.e., Y_1 and Y_2 are independent given X_1 and X_2 (Exercise 1).

As you'll recall, this is a special case of the simple linear regression model: the first two assumptions are the same, but we are now assuming much more about the noise variable ϵ: it's not just mean zero with constant variance, but it has a particular distribution (Gaussian), and everything we said was uncorrelated before we now strengthen to independence[1].

[1] See Chapter 1 for a reminder, with an explicit example, of how uncorrelated random variables can nonetheless be strongly statistically dependent.

Because of these stronger assumptions, the model tells us the conditional pdf of Y for each x, $p(y|X=x;\beta_0,\beta_1,\sigma^2)$. (This notation separates the random variables from the parameters.) Given any data set $(x_1,y_1),(x_2,y_2),\ldots(x_n,y_n)$, we can now write down the probability density, under the model, of seeing that data:

$$\prod_{i=1}^{n} p(y_i|x_i;\beta_0,\beta_1,\sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i-(\beta_0+\beta_1 x_i))^2}{2\sigma^2}}$$

In multiplying together the probabilities like this, we are using the conditional independence of the Y_i (given the X_i), which follows from the independence of the ϵ_i.

When we see the data, we do not *known* the true parameters, but any guess at them, say (b_0,b_1,s^2), gives us a probability density:

$$\prod_{i=1}^{n} p(y_i|x_i;b_0,b_1,s^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{(y_i-(b_0+b_1 x_i))^2}{2s^2}}$$

This is the **likelihood**, a function of the parameter values. It's just as informative, and much more convenient, to work with the **log-likelihood**,

$$L(b_0,b_1,s^2) = \log \prod_{i=1}^{n} p(y_i|x_i;b_0,b_1,s^2) \quad (5.1)$$

$$= \sum_{i=1}^{n} \log p(y_i|x_i;b_0,b_1,s^2) \quad (5.2)$$

$$= -\frac{n}{2}\log 2\pi - n\log s - \frac{1}{2s^2}\sum_{i=1}^{n}(y_i-(b_0+b_1x_i))^2 \quad (5.3)$$

In the **method of maximum likelihood**, we pick the parameter values which maximize the likelihood, or, equivalently, maximize the log-likelihood. After some calculus (see Chapter 3, this gives us the following estimators:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sum_{i=1}^{n}(x_i-\overline{x})^2} = \frac{c_{XY}}{s_X^2} \quad (5.4)$$

$$\hat{\beta}_0 = \overline{y} - \hat{beta}_1\overline{x} \quad (5.5)$$

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i-(\hat{\beta}_0+\hat{\beta}_1 x_i))^2 \quad (5.6)$$

As you will recall, the estimators for the slope and the intercept exactly match the least squares estimators. This is a special property of assuming independent Gaussian noise. Similarly, $\hat{\sigma}^2$ is exactly the in-sample mean squared error.

5.2 Sampling Distributions

We may seem not to have gained much from the Gaussian-noise assumption, because our point estimates are just the same as they were from least squares. What makes the

Gaussian noise assumption important is that it gives us an exact conditional distribution for each Y_i, and this in turn gives us a distribution — the **sampling distribution** — for the estimators. Remember, from the notes from last time, that we can write $\hat{\beta}_1$ and $\hat{\beta}_0$ in the form "constant plus sum of noise variables". For instance,

$$\hat{\beta}_1 = \beta_1 + \sum_{i=1}^{n} \frac{x_i - \overline{x}}{n s_X^2} \epsilon_i$$

Now, in the Gaussian-noise model, the ϵ_i are all independent Gaussians. Therefore, $\hat{\beta}_1$ *is also Gaussian*. Since we worked out its mean and variance last time, we can just say

$$\hat{\beta}_1 \sim N(\beta_1, \sigma^2 / n s_X^2)$$

Again, we saw that the fitted value at an arbitrary point x, $\hat{m}(x)$, is a constant plus a weighted sum of the ϵ:

$$\hat{m}(x) = \beta_0 + \beta_1 x + \frac{1}{n} \sum_{i=1}^{n} \left(1 + (x - \overline{x}) \frac{x_i - \overline{x}}{s_X^2}\right) \epsilon_i$$

Once again, because the ϵ_i are independent Gaussians, a weighted sum of them is also Gaussian, and we can just say

$$\hat{m}(x) \sim N\left(\beta_0 + \beta_1 x, \frac{\sigma^2}{n}\left(1 + \frac{(x - \overline{x})^2}{s_X^2}\right)\right)$$

Slightly more complicated manipulation of the ϵ_i makes it possible to show that

$$\frac{n\hat{\sigma}^2}{\sigma^2} \sim \chi^2_{n-2}$$

These are all important, because when we come to doing statistical inference on the parameters — forming confidence intervals, or testing hypotheses — we need to know these sampling distributions. When we come to making predictions of new Y's, these sampling distributions will let us give confidence intervals for the expected values, $\hat{m}(x)$, as well as give prediction intervals (of the form "when $X = 5$, Y will be between l and u with 95% probability") or full distributional forecasts. We will derive these inferential formulas in later chapters.

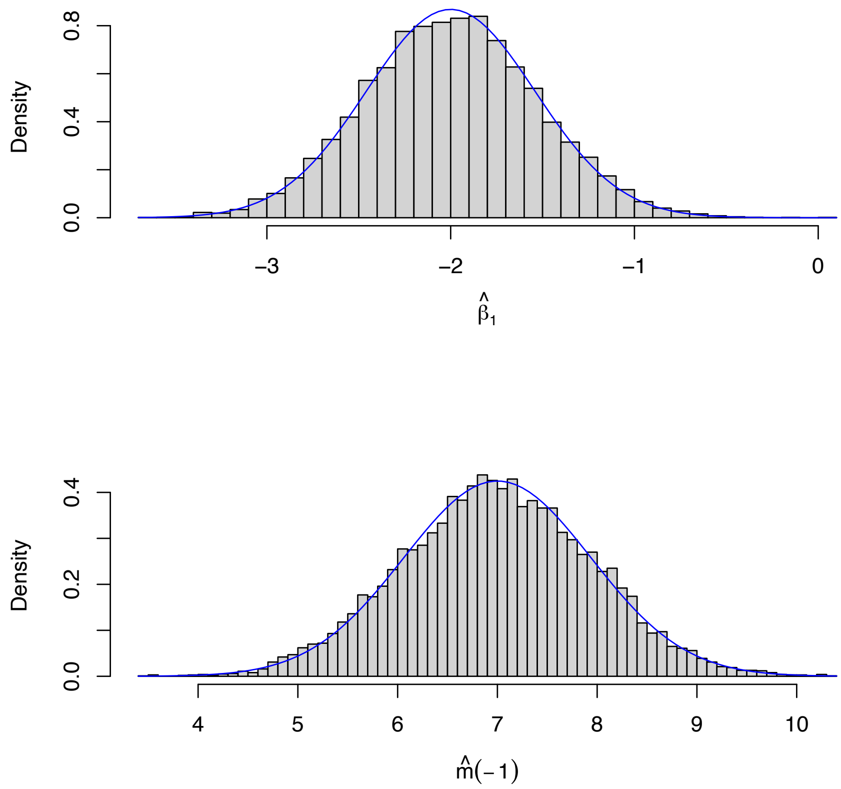
5.2.1 Illustration

To make the idea of these sampling distributions more concrete, I present a small simulation. Figure 5.1 provides code which simulates a particular Gaussian-noise linear model: $\beta_0 = 5$, $\beta_1 = -2$, $\sigma^2 = 3$, with twenty X's initially randomly drawn from an exponential distribution, but thereafter held fixed through all the simulations. The theory above lets us calculate just what the distribution of $\hat{\beta}_1$ should be, in repeated simulations, and the distribution of $\hat{m}(-1)$. (By construction, we have no *observations* where $x = -1$; this is an example of using the model to extrapolate beyond the data.) Figure 5.2 compares the theoretical sampling distributions to what we actually get by repeated simulation, i.e., by repeating the experiment.

```
# Fix x values for all runs of the simulation; draw from an exponential
n <- 20  # So we don't have magic #s floating around
beta.0 <- 5
beta.1 <- -2
sigma.sq <- 3
fixed.x <- rexp(n = n)

# Simulate from the model Y=\beta_0+\beta_1*x+N(0,\sigma^2) Inputs: intercept;
# slope; variance; vector of x; return sample or estimated linear model?
# Outputs: data frame with columns x and y OR linear model fit to simulated y
# regressed on x
sim.lin.gauss <- function(intercept = beta.0, slope = beta.1, noise.variance = sigma.sq,
    x = fixed.x, model = FALSE) {
    # Make up y by adding Gaussian noise to the linear function
    y <- rnorm(length(x), intercept + slope * x, sd = sqrt(noise.variance))
    # Do we want to fit a model to this simulation and return that model?  Or
    # do we want to just return the simulated values?
    if (model) {
        return(lm(y ~ x))
    } else {
        return(data.frame(x = x, y = y))
    }
}
```

FIGURE 5.1: *Function to simulate a Gaussian-noise simple linear regression model, together with some default parameter values. Since, in this chapter, we'll always be estimating a linear model on the simulated values, it makes sense to build that into the simulator, but I included a switch to control that.*



```
par(mfrow = c(2, 1))
slope.sample <- replicate(10000, coefficients(sim.lin.gauss(model = TRUE))["x"])
hist(slope.sample, freq = FALSE, breaks = 50, xlab = expression(hat(beta)[1]), main = "")
curve(dnorm(x, -2, sd = sqrt(3/(n * var(fixed.x)))), add = TRUE, col = "blue")
pred.sample <- replicate(10000, predict(sim.lin.gauss(model = TRUE), newdata = data.frame(x = -1))))
hist(pred.sample, freq = FALSE, breaks = 50, xlab = expression(hat(m)(-1)), main = "")
curve(dnorm(x, mean = beta.0 + beta.1 * (-1), sd = sqrt((sigma.sq/n) * (1 + (-1 -
    mean(fixed.x))^2/var(fixed.x)))), add = TRUE, col = "blue")
```

FIGURE 5.2: *Theoretical sampling distributions for $\hat{\beta}_1$ and $\hat{m}(-1)$ (blue curves) versus the distribution in 10^4 simulations (black histograms).*

5.3 Virtues and Limitations of Maximum Likelihood

The method of maximum likelihood does not always work; there are models where it gives poor or even pathological estimates. For Gaussian-noise linear models, however, it actually works very well. Indeed, in more advanced statistics classes, one proves that for such models, as for many other "regular" statistical models, maximum likelihood is **asymptotically efficient**, meaning that its parameter estimates converge on the truth as quickly as possible[2]. This is on top of having exact sampling distributions for the estimators.

Of course, all these wonderful abilities come at a cost, which is the Gaussian noise assumption. If that is wrong, then so are the sampling distributions I gave above, and so are the inferential calculations which rely on those sampling distributions. *Before* we begin to do those inferences on any particular data set, and *especially* before we begin to make grand claims about the world on the basis of those inferences, we should really check all those modeling assumptions. That, however, brings us into the topics for next week.

Exercises

1. Let $Y_1, Y_2, \ldots Y_n$ be generated from the Gaussian-noise simple linear regression model, with the corresponding values of the predictor variable being $X_1, \ldots X_n$. Show that if $i \neq j$, then Y_i and Y_j are conditionally independent given (X_i, X_j). *Hint:* If U and V are independent, then $f(U)$ and $g(V)$ are also independent, for any functions f and g.

2. In many practical fields (e.g., finance and geology) it is common to encounter noise whose distribution has much heavier tails than any Gaussian could give us. One way to model this is with t distributions. Consider, therefore, the statistical model where $Y = \beta_0 + \beta_1 X + \epsilon$, and $\epsilon/\sigma \sim t_\nu$, with ϵ independent of X and independent across observations. That is, rather than having a Gaussian distribution, the noise follows a t distribution with ν degrees of freedom (after scaling).

   Note: Most students find most parts after (a) quite challenging.

   (a) Write down the log-likelihood function. Use an explicit formula for the density of the t distribution.

   (b) Find the derivatives of this log-likelihood with respect to the four parameters β_0, β_1, σ (or σ^2, if more convenient) and ν. Simplify as much as possible. (It is legitimate to use derivatives of the gamma function here, since that's another special function.)

[2] *Very* roughly: writing θ for the true parameter, $\hat{\theta}$ for the MLE, and $\tilde{\theta}$ for any other consistent estimator, asymptotic efficiency means $\lim_{n\to\infty} \mathbb{E}\left[n\|\hat{\theta}-\theta\|^2\right] \leq \lim_{n\to\infty} \mathbb{E}\left[n\|\tilde{\theta}-\theta\|\right]$. (This way of formulating it takes it for granted that the MSE of estimation goes to zero like $1/n$, but it typically does in parametric problems.) For more precise statements, see, for instance, Cramér (1945), Pitman (1979) or van der Vaart (1998).

(c) Can you solve for the maximum likelihood estimators of β_0 and β_1 without knowing σ and ν? If not, why not? If you can, do they match the least-squares estimators again? If they don't match, how do they differ?

(d) Can you solve for the MLE of all four parameters at once? (Again, you may have to express your answer in terms of the gamma function and its derivatives.)

3. Refer to the previous problem, and do part (a).

(a) In R, write a function to calculate the log-likelihood, taking as arguments a data frame with columns names `y` and `x`, and the vector of the four model parameters. *Hint:* use the `dt` function.

(b) In R, using `optim`, write a function to find the MLE of this model on a given data set, from an arbitrary starting vector of guesses at the parameters. This should call your function from part (a).

(c) In R, write a function which gives an unprincipled but straight-forward initial estimate of the parameters by (i) calculating the slope and intercept using least squares, and (ii) fitting a t distribution to the residuals. *Hint:* call `lm`, and `fitdistr` from the package `MASS`.

(d) Combine your functions to write a function which takes as its only argument a data frame containing columns called `x` and `y`, and returns the MLE for the model parameters.

(e) Write another function which will simulte data from the model, taking as arguments the four parameters and a vector of x's. It should return a data frame with appropriate column names.

(f) Run the output of your simulation function through your MLE function. How well does the MLE recover the parameters? Does it get better as n grows? As the variance of your x's increases? How does it compare to your unprincipled estimator?