Gov 2001: Section 5

I. A Normal Example
II. Uncertainty

Gov 2001

Spring 2010

A roadmap

- We started by introducing the concept of likelihood in the simplest univariate context – one observation, one variable.
- Then we moved forward with more than one observation – and multiplied likelihoods together.
- Now, we are introducing covariates.

A roadmap (ctd.)

Key to all of this is the distinction between stochastic and systematic components:

- *Stochastic* - the probability distribution of the data; key to identifying what model (Poisson, binomial, etc.) you should use., E.g., $Y_i \sim f(y_i|\gamma)$.
- *Systematic* - how the parameters of the probability distribution vary over your covariates; key to incorporating covariates into your model. E.g., $\gamma = g(X_i, \theta)$.

You'll need both parts to model the likelihood, and you'll need a more sophisticated systematic component to include interesting covariates.

Normal Example

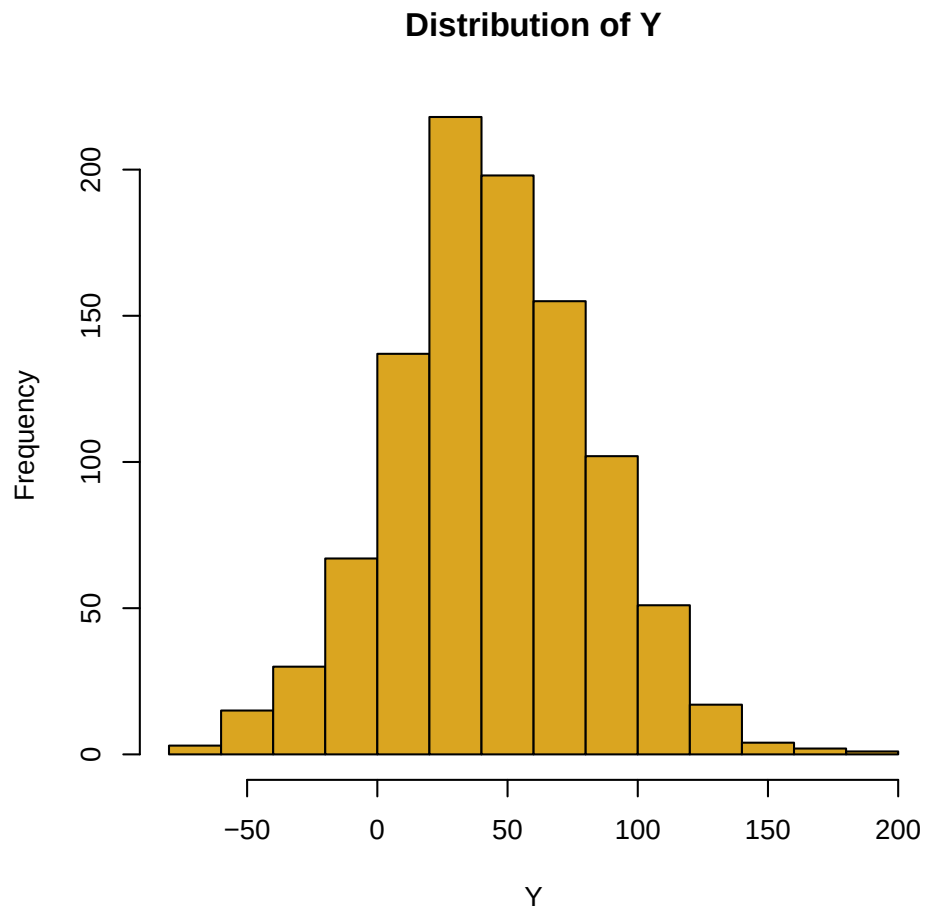
Let's work through an example of how this all works. I'm going to create some fake data:

```
> x <- rnorm(1000,.5,6)
> z <- rnorm(1000,100,.5)
> Y <- 14 + 6.4*x + .25*z + rnorm(1000,0,1)
```

Normal Example (ctd.)

So y will be normally distributed. Why?

```
> hist(Y, col = "goldenrod", main = "Distribution of y")
```



Normal Example (ctd.)

Since Y is continuous and normally distributed, we could use OLS:

```
> Y <- 14 + 6.4*x + .25*z + rnorm(1000,0,1)
> my.lm <- lm(Y ~ x + z)
> my.lm

Call:
lm(formula = Y ~ x + z)

Coefficients:
(Intercept)            x            z
    16.6723       6.3997       0.2228
```

Normal Example (ctd.)

- But we can also calculate the same results using likelihood techniques. How?

$$\begin{aligned} Stochastic: Y_i &\sim N(\mu, \sigma^2) \\ Systematic: \mu &= B_0 + B_1 X_1 + ... + B_k X_k \end{aligned}$$

This leaves us with the following likelihood for the ith observation:

$$\begin{aligned} L(\mu_i, \sigma^2 | y) &\propto N(y_i | \mu_i, \sigma^2) \\ &\propto (2\pi\sigma^2)^{-\frac{1}{2}} e^{\frac{-(y_i - \mu_i)^2}{2\sigma^2}} \end{aligned}$$

Normal Example (ctd.)

- To calculate the full log likelihood, we assume independence among observations and multiply;
- then take the natural log;
- then introduce our parameterization.

$$
\begin{aligned}
L(\beta, \sigma^2|y) &= \prod L(y_i|\mu_i, \sigma^2) \\
lnL(\beta, \sigma^2|y) &= \sum lnL(y_i|\mu_i, \sigma^2) \\
&= \sum -\frac{1}{2}[ln\sigma^2 + \frac{(y_i - \mu)^2}{\sigma^2}] \\
&= \sum -\frac{1}{2}[ln\sigma^2 + \frac{(y_i - X_i\beta)^2}{\sigma^2}]
\end{aligned}
$$

Normal Example (ctd.)

- This log likelihood is too complicated to analyze analytically.
- So we aim for a numeric solution.
- We can implement the log likelihood in R using the commands from Monday's lecture notes:

```
ll.normal <- function(par,y,X){
    beta <- par[1:ncol(X)]
    sigma2 <- exp(par[ncol(X)+1])
    -1/2 * (sum(log(sigma2) + (y -(X%*%beta))^2/sigma2))
}
```

Normal Example (ctd.)

The `Zelig` package will calculate the MLE estimates automatically.

```
> install.packages("Zelig")
> library(Zelig)
> ex <- data.frame(Y,x,z)
> my.z <- zelig(Y ~ x + z, model = "normal", data = ex)
> my.z
Coefficients:
(Intercept)           x           z
     13.079       6.394       0.259
```

Normal Example (ctd.)

But we will tackle this manually:

```
ll.normal <- function(par,y,X){
    beta <- par[1:ncol(X)]
    sigma2 <- exp(par[ncol(X)+1])
    -1/2 * (sum(log(sigma2) + (y -(X%*%beta))^2/sigma2))
}
```

where the inputs will be

- `par` - a vector of parameters you want the likelihood for
- `y` - a vector for the dependent variable
- `X` - a matrix of covariates plus a row of 1's for the intercept

(Why do you need a vector of 1's? Because $\mu = X_i\beta$.)

Normal Example (ctd.)

Note: X must be in matrix form so that you can do the matrix multiplication.

```
> ll.normal(par = c(14,6.4,.25, 40),
  +  y = Y, X = cbind(1,x,z))
[1] -20000
```

```
> ll.normal(par = c(0,0,0,0), y = Y, X = cbind(1,x,z))
[1] -1591275
```

Which potential parameters are better? Why?

Normal Example (ctd.)

- At the end of the day, we don't want the absolute value of the likelihood.
- We want to optimize the likelihood across different values of the parameters and check which values maximize the likelihood.
- We have four parameters: an intercept, a coefficient on x, a coefficient on z, and a value of σ^2.
- To calculate automatically the likelihood across different possible values of these, we use `optim`.

Normal Example (ctd.)

Here's how we can use `optim`:

```
> my.optim <- optim(par = c(0,0,0,0), fn = ll.normal,
+   y = Y, X = cbind(1,x,z),
+   method = "BFGS", control=list(fnscale=-1), hessian=T)
```

- The inputs to `optim` include a `par` argument. These should be your proposed starting values.
- Choose starting values that substantively make sense – otherwise, the optimizing algorithm might get lost!
- Also remember to include starting values for your intercept and for ancillary parameters.

Normal Example (ctd.)

So let's look at the `optim` output:

```
> my.optim$par
[1] 16.67015681  6.39974960  0.22284211 -0.02824721
```

We can cross-check our answers with the `lm` function.

```
> my.lm

Coefficients:
(Intercept)                x                z
    16.6723           6.3997           0.2228
```

Look good!

Gov 2001: Section 5

I. A Normal Example
II. Uncertainty

Gov 2001

Spring 2010

Intro to Uncertainty

- Once an ML estimates are calculated, we'll want to know how good they are.
- How much information does the MLE contain about the underlying parameter?
- How good a summary of the entire likelihood is this one point?
- The MLE alone isn't satisfying – we need a way to quantify uncertainty.

Intro to Uncertainty (ctd.)

Common ways to think about uncertainty:

- Likelihood ratio tests – useful for comparing restricted versus unrestricted models. (UPM p. 84-86)
- Estimating standard errors – Use normal approximation to get the standard errors of the coefficients; may be calculated by estimating only the unrestricted model (more like what Gary was talking about in class). (UPM 87-92)

Likelihood Ratio Tests

- Useful for when you are comparing two models.
- We'll call these restricted and unrestricted:

$$\textit{Unrestricted} : \beta_0 + \beta_1 X_1$$
$$\textit{Restricted} : \beta$$

- We want to test the usefulness of the parameters in the unrestricted model but omitted in the restricted model.

Likelihood Ratio Tests (ctd.)

Here's how to operationalize this:

- Let L^* be the maximum of the unrestricted likelihood, and let L_r^* the maximum of the restricted likelihood.
- But adding more variables can only increase the likelihood.
- Thus, $L^* \geq L_r^*$, or $\frac{L_r^*}{L} \leq 1$.
- If the likelihood ratio is exactly 1, then there's no effect of the extra parameters at all.

Likelihood Ratio Tests (ctd.)

Now, let's define a test statistic:

$$\begin{aligned} define: \Re &= -2ln\frac{L_r^*}{L^*} \\ &= 2(lnL^* - lnL_r^*) \end{aligned}$$

- $\Re$ will always be greater than zero.
- By definition it follows a χ^2 distribution with m degrees of freedom, where m is the number of restrictions.
- Key question: how much greater than zero does $\Re$ have to be in order to convice us that the difference is due to systematic differences between the two models?

Likelihood Ratio Test Example

Let's go back to our example.

```
> unrestricted <- optim(par = c(0,0,0,0), fn = ll.normal,
+   y = Y, X = cbind(1,x,z),
+   method = "BFGS", control=list(fnscale=-1), hessian=T)
> unrestricted$value
[1] -485.8741
```

versus

```
> restricted <- optim(par = c(0,0,0), fn = ll.normal,
+   y = Y, X = cbind(1,x),
+  method = "BFGS", control=list(fnscale=-1), hessian=T)
> restricted$value
[1] -492.2747
```

Likelihood Ratio Test Example (ctd.)

Under the null that the restrictions are valid, the test statistic would be distributed χ^2 with one degree of freedom:

```
> r <- 2*(unrestricted$value-restricted$value)
```

```
> 1-pchisq(r, df = 1)
[1] 0.0003464178
```

So the probability of getting this test statistic under the null is extremely small. We reject.

Using Standard Errors

We can also move forward using the curvature of the likelihood curve around the MLE, which is a measure of the precision of the ML estimate.

- Measure of curvature: Fisher Information Matrix

$$I(\hat{\theta}) = -\frac{\partial^2 lnL(\theta)}{\partial^2 \theta}(\hat{\theta})$$

- Inverse of the Fisher Information gives us $Var(\hat{\theta})$

$$[I(\hat{\theta})]^{-1} \approx Var(\hat{\theta})$$

- Square root of $Var(\hat{\theta})$ gives us $SE(\hat{\theta})$

$$SE(\hat{\theta}) = \sqrt{Var(\hat{\theta})}$$

Using Standard Errors (ctd.)

- $I(\hat{\theta})$ is based on a quadratic approximation of $lnL(\theta|y)$ at $\hat{\theta}$
- If $\hat{\theta}$ is normal, then the quadratic approximation will be exactly true
- If $\hat{\theta}$ is not exactly normal, then the quadratic approximation holds as n $\rightarrow \infty$
- Why? Central limit theorem and sampling distribution of $\hat{\theta}$

Using Standard Errors (ctd.)

- We can use the standard errors in a variety of ways, including to do hypothesis testing and to calculate confidence intervals.
- Wald's test is a generalization from t-tests from regression analysis.
- Here's how it works
  - Choose a null hypothesis, H_0: $\theta = \theta_0$;
  - Use that to calculate a test statistic, Z:

$$Z = \frac{\hat{\theta} - \theta_0}{SE(\hat{\theta})} \sim N(0, 1)$$

  - Then see how likely it is to see that test statistic given that the null is true.

Using Standard Errors (ctd.)

Let's go back to our example:

```
> my.opt <- optim(par = c(0,0,0,0), fn = ll.normal,
+         y = Y, X = cbind(1,x,z),
+ method = "BFGS", control=list(fnscale=-1), hessian=T)
```

Let's get the Hessian matrix out:

```
> my.opt$hessian
              [,1]          [,2]          [,3]          [,4]
[1,] -1.069522e+03 -8.922910e+02 -1.069601e+05  3.008466e-03
[2,] -8.922910e+02 -3.939036e+04 -8.900873e+04  8.031144e-02
[3,] -1.069601e+05 -8.900873e+04 -1.069708e+07  3.145044e-01
[4,]  3.008466e-03  8.031144e-02  3.145044e-01 -4.999940e+02
```

Using Standard Errors (ctd.)

To calculate the variance-covariance matrix:

```
> opt.vcv <- solve(-1*my.opt$hessian)
> opt.vcv
                [,1]           [,2]           [,3]           [,4]
[1,]  3.663626e+01 -2.173033e-03 -3.663080e-01 -1.032219e-05
[2,] -2.173033e-03  2.600229e-05  2.151180e-05  4.632738e-09
[3,] -3.663080e-01  2.151180e-05  3.662629e-03  1.032319e-07
[4,] -1.032219e-05  4.632738e-09  1.032319e-07  2.000024e-03
```

Using Standard Errors (ctd.)

To calculate the variances and standard errors:

```
> vars <- diag(opt.vcv)
> vars
[1] 3.663626e+01 2.600229e-05 3.662629e-03 2.000024e-03

> ses <- sqrt(vars)
> ses
[1] 6.052789312 0.005099244 0.060519658 0.044721627
```

Using Standard Errors (ctd.)

And, lastly, to compare this with the `lm` output:

```
> results <- data.frame(my.opt$par, ses)
> results
   my.opt.par          ses
1 17.92300722 6.052789312
2  6.39448686 0.005099244
3  0.21106796 0.060519658
4 -0.06721196 0.044721627

> summary(my.lm)
Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept) 17.928169   6.061845    2.958  0.00317 **
x            6.394485   0.005107 1252.132  < 2e-16 ***
z            0.211016   0.060610    3.482  0.00052 ***
```

Rock and Roll!