MLE HW 1

Lindley Slipetz

6/27/2021

Before jumping into code, I just want let you know that I'm not taking this course for a grade. Now let's load the data. There seems to be an issue with loading dta data from this particular version of STATA with the foreign package, so I'm going to use haven instead.

```
library(haven)
Normalxy<-read_dta("C:\\Users\\Owner\\Documents\\ICPSR\\MLE\\HW 1\\normalxy.dta")
```

Here's the summary of the data.

```
summary(Normalxy)
```

```
##        x                y
##  Min.   :-174.2   Min.   :-58.01
##  1st Qu.:  88.1   1st Qu.: 29.95
##  Median : 148.6   Median : 49.94
##  Mean   : 149.6   Mean   : 50.00
##  3rd Qu.: 209.5   3rd Qu.: 70.14
##  Max.   : 498.6   Max.   :167.17
```

Estimate the regression model

```
ols <- lm(Normalxy$y ~ Normalxy$x)
summary(ols)
```

```
##
## Call:
## lm(formula = Normalxy$y ~ Normalxy$x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.685  -8.679  -0.072   8.800  46.909
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.319726   0.252976   21.03   <2e-16 ***
## Normalxy$x  0.298736   0.001448  206.31   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.08 on 9998 degrees of freedom
## Multiple R-squared:  0.8098, Adjusted R-squared:  0.8098
## F-statistic: 4.256e+04 on 1 and 9998 DF,  p-value: < 2.2e-16
```

Estimate a likelihood function using a normal distribution

```
#install.packages("maxLik")
library(maxLik)
```

```
## Loading required package: miscTools
```

```
##
## Please cite the 'maxLik' package as:
## Henningsen, Arne and Toomet, Ott (2011). maxLik: A package for maximum likelihood estimation in R. C
##
## If you have questions, suggestions, or comments regarding the 'maxLik' package, please use a forum o
## https://r-forge.r-project.org/projects/maxlik/
```

```
ml.in <- function(par, x=Normalxy$x, y=Normalxy$y){
n <- length(y)
xb <- par[1] + par[2]*x
e <- y-xb
sigma <- sqrt(sum(e^2)/n)
sum(-(1/2)*log(sigma^2) - (1/(2*sigma^2))*((y-xb)^2))
}
ml.out<-maxBFGS(ml.in, start=c(1,1), x=Normalxy$x, y=Normalxy$y)
summary(ml.out)
```

```
## --------------------------------------------
## BFGS maximization
## Number of iterations: 49
## Return code: 0
## successful convergence
## Function value: -30707.2
## Estimates:
##       estimate      gradient
## [1,] 5.3197261 -3.637979e-06
## [2,] 0.2987359 -8.039933e-04
## --------------------------------------------
```

Hessian

```
ml.out$hessian
```

```
##             [,1]         [,2]
## [1,]   -65.48362    -8752.977
## [2,] -8752.97701 -1785301.720
```

Calculate SE's from Hessian

```
ml.se <- sqrt(diag(solve(-ml.out$hessian)))
ml.se
```

```
## [1] 0.210493553 0.001274821
```

Calculate SE's from VCOV

```
ml.se.vcov <- sqrt(diag(ml.out$vcov))
ml.se.vcov
```

```
## <0 x 0 matrix>
```

Table of results

```
res <-matrix(NA, nrow = 5, ncol = 2)
res[, 1] <-c(format(coef(ols)[1], digits = 3),paste0("(",format(sqrt(diag(vcov(ols)))[1], digits = 3),
res[, 2] <- c(format(ml.out$estimate[1], digits = 3), paste0("(", format(ml.se[1], digits = 3), ")"), fo
rownames(res) <- c("Intercept", "", "x", "", "N")
colnames(res) <-c("OLS", "MLE")
res
```

```
##           OLS         MLE
## Intercept "5.32"      "5.32"
##           "(0.253)"   "(0.21)"
## x         "0.299"     "0.299"
##           "(0.00145)" "(0.00127)"
## N         "10000"     "10000"
```

When rounded to three significant figures, the coefficient estimates are the same for OLS and MLE. The intercept is 5.32 and the x-coefficient is 0.299. They do differ in standard error, meaning that the OLS data are slightly further from the regression line than the MLE estimates. Thus, we say that the MLE model has a better fit.