# MATH3091 Computer Lab 2

7 Feb 2022

In this lab, we will code up the likelihood function, find MLE, and revise the linear model. Please take your time and make sure you understand each of the `R` command.

## Log-likelihood function

Recall the log-likelihood function for Bernoulli($p$) i.i.d data,

$$\ell(p) = n\bar{y}\log(p) + n(1-\bar{y})\log(1-p),$$

where $\bar{y} = \sum_{i=1}^n y_i/n$. We have seen the log-likelihood function in the Lecture 2, where we are taking two input arguments: `y` for the $n$-vector of observed data and `p` for the value of parameters.

```r
bernoulli.lfun <- function(p, y) {
ybar <- mean(y)
n <- length(y)

# return the log-likelihood
n * ybar * log(p) + n * (1 - ybar) * log(1 - p)
}
```

We can plot the curve of this function for a range of $p$, given the data. Say,

```r
y <- c(1, 0, 0, 0, 1, 0, 0, 0, 1, 0)
```

The the log-likelihood function

```r
curve(bernoulli.lfun(x, y), from = 0, to = 1, ylim = c(-12, -6),
      xlab = "p", ylab = "log L(p) ", main="Bernoulli log-likelihood")
```

**Question:** Could you plot the log-likelihood for another sample `y <- c(rep(1,100), rep(0,120))`?

Now let us look at the normal $N(\mu, \sigma^2)$ log-likelihood function, which is

$$\ell(\mu, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2}\frac{\sum_{i=1}^n (y_i-\mu)^2}{\sigma^2}$$

We can program it simply as:

```
normal.lfun<-function(mu,  sigma2,  y){
n <- length(y)

#return the log-likelihood
ret <- NULL
for (i in 1:length(mu)){
lik<- -.5*n*log(2*pi) -.5*n*log(sigma2) - (1/(2*sigma2))*sum((y-mu[i])**2)
ret <- c(ret,lik)
}
return(ret)
}
```

**Question:** Can you plot the log-likelihood function versus $\mu$, with a fixed $\sigma^2 = 1$ and a data set of 20 random numbers from $N(2, 1)$?

## Find the MLE

Now let us think about how to find MLEs by optimising the log-likelihood function. Of course, we can derive them directly if there is a close-form expression (like $\hat{p} = \bar{y}$ for the Bernoulli($p$) data). If not, numerical methods is needed.

By default, the `optim` command can be invoked.

```
#help file of optim
help(optim)
```

A minimal working example of this command is `optim(starting values, negative log-likelihood, data)`. Here `starting values` is a vector of starting values, `negative log-likelihood` is the name of the function that we seek to minimise, and `data` declares the dataset for the get the MLE.

```
#data of 10 samples
y <- c(1, 0, 1, 0, 1, 0, 0, 0, 1, 0)


bernoulli.lfun1 <- function(p, y) {
ybar <- mean(y)
n <- length(y)

# return the negative log-likelihood
-n * ybar * log(p) - n * (1 - ybar) * log(1 - p)
}
#here we are taking an extra minus sign
#because optim is used to minimise an object

#the MLE by hand calculation should be .4
```

```r
optim(0.3, bernoulli.lfun1, y=y)
```

**Question:** Can you numerically find the MLE of $\mu$ given $\sigma^2 = 1$ using the normal data that we generated above? Comparing your results to the MLE by expression.

**Question:** We can also code the New-Raphson method introduced in Lecture 4 to do the optimisation, which is more complex, you can try it in your spare time.

## Linear regression model

Next, we will go over the `R` multiple regression commands, which you should have seen before. As the later material on generalised linear models depends on you understanding this.

Let us import the nitric acid data again

Recall the `read.csv` command to read in the file `nitric.csv`.

```r
nitric <- read.csv("nitric.csv")
```

This data set relates to 21 successive days of operation of a plant oxidising ammonia to nitric acid. The response `yield` is ten times the percentage of ingoing ammonia that is lost as unabsorbed nitric acid (an indirect measure of the yield). The aim here is to study how the yield depends on flow of air to the plant (`flow`), temperature of the cooling water entering the absorption tower (`temp`) and concentration of nitric acid in the absorbing liquid (`conc`).

Get a plot of the data.

```r
pairs(nitric)
```

or with the help of `ggplot2` and `GGally`

```r
GGally::ggpairs(data=nitric)
```

**Fit a linear model**. We fit the linear model

$$Y_i = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \epsilon_i, \quad i = 1, \ldots, 21,$$

where $Y_i$ is the `yield`, $x_{i1}$ is the `flow`, $x_{i2}$ is the `temp` and $x_{i3}$ is the `conc` on the $i$-th day. To fit this model to the `nitric` data, we can use

```r
nitric.lm1 <- lm(yield ~ flow + temp + conc, data = nitric)
summary(nitric.lm1)
```

**Interpretation**. It looks like `conc` is not required in the model because its $p$-value is 0.344. The other parameters are all significant. The model explained about 91.36% of total variation in the data.

**Remove a term from the model**. We can refit by omitting `conc`.

```r
nitric.lm2 <- lm(yield ~ flow + temp, data = nitric)
```

Alternatively, we could use `update` to drop a term from `nitric.lm1`:

```r
nitric.lm2.alt <- update(nitric.lm1, . ~ . - conc)
```

This updates the linear model object `nitric.lm1` by removing `conc` from the formula. Check that `nitric.lm2` and `nitric.lm2.alt` are the same.

We can write this second model mathematically as

$$Y_i = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i, \quad i = 1, \ldots, 21.$$

We can get a summary of the fitted model with

```r
summary(nitric.lm2)
```

**Question:** Do you think you should drop any other variables?

**Find confidence intervals**. We could find a 95% confidence interval for each regression coefficient $\beta_i$ by using the Estimate and standard error columns of the summary output. From those we can obtain a confidence interval using

$$\text{Estimate} \pm \text{Std. Error} \times \text{critical value}.$$

For a 95% confidence interval the critical value is

```r
qt(0.975, 18)
```

which is 2.1. Therefore $0.671 \pm 0.127 \times 2.1$ or $(0.404, 0.937)$ is a 95% confidence interval for $\beta_1$ under model 2. Find 95% confidence intervals for the intercept $\alpha$ and regression coefficient $\beta_2$.

`R` has a built in function `confint` to find confidence intervals. You can find out more about the arguments of `confint` by checking the help

```r
?confint
```

Try

```r
confint(nitric.lm2)
```

**Question:** How does this compare to the confidence intervals you found "by hand" above?

**Question:** Can you find 90% confidence intervals for the parameters of `nitric.lm2` using `confint`?

**Check the diagnostic plots**. For the selected model we should check the assumptions. We do this by issuing the command

```r
plot(nitric.lm2)
```

It produces four plots. We will only consider the first two plots.

If the model is good, the first plot should not show any systematic pattern: the points should be randomly scattered above and below the $x$-axis.

The second plot is the normal qqplot. This is used to check normality. If the residuals are normally distributed then all the points should lie close to a straight line. In our case observation 21 shows some outlying behaviour. Otherwise the plot seems to be fine.

**How can we predict using a fitted linear model?**. We can predict using the linear model output very easily using the predict function. Look at the help file

```r
?predict.lm
```

If we just wanted to predict the means at the values for the explanatory variables we have used to fit, we just use

```r
predict(nitric.lm2, se.fit = TRUE, interval = "confidence")
```

This gives the predicted means, standard errors and the default 95% confidence interval.

If we want to predict future actual observations (NOT the means) at the values for the explanatory variables which we have used to fit, we use

```r
predict(nitric.lm2, interval = "prediction")
```

This gives the predicted observations and the default 95% confidence interval.

If we want to predict at new values for the explanatory variables then we first create a data set which has the column names exactly as the original data set. The response `yield` column is not needed. Then put the values for the explanatory variables for which we want to predict as rows. Then use the above commands, but the second argument should be the name of the new data set.

Suppose that we want to predict the `yield` value at `flow = 60`, `temp = 20` using the model with two covariates `flow` and `temp`. We create a new data frame, `newdata` for example, by using

```r
newdata <- data.frame(flow = 60, temp = 20)
```

Then issue the command

```r
predict(nitric.lm2, newdata = newdata, se.fit = TRUE, interval = "confidence")
```

This gives the predicted mean, standard errors and the default 95% confidence interval. The command

```r
predict(nitric.lm2,  newdata = newdata, se.fit = TRUE, interval= "prediction")
```

gives the predicted observation and the default 95% prediction interval.

**Question:** What is the difference between the "predition interval" and the "confidence interval"?