2019 Class Test 2: DD80

This test paper is only for students on the DD80 programme.

The Moodle password for DD80 Class Test 2 is: tracks

Formula 1

Formula One is the highest class of single-seater auto racing sanctioned by the Federation Internationale de l'Automobile (FIA). The file `tracks.csv` contains the location for all tracks where a Formula 1 Grand Prix race has been held (at least once) since 1956. The file has the following columns

`tracks.csv`	
`location`	town/city the track is in
`country`	country the track is in
`lat`	latitude of the track
`lng`	longitude of the track
`continent`	continent the track is in

Q1 [**2 marks**] Correctly read in the file `tracks.csv` in to R and save it as a dataframe called `tracks`.

Q2 [**2 marks**] Define a variable `spanish` which contains the number of race tracks in the dataframe `tracks` which are located in Spain.

```
spanish <- sum(tracks$country=="Spain")
```

Q3 [**2 marks**] Sort the rows of the `tracks` dataframe in decreasing order according to `lat`. The ordered dataframe should be called `tracks`

```
tracks <- tracks[order(tracks$lat, decreasing=TRUE),]
```

Q4 [**4 marks**] Define a 6×2 column matrix `central` which contains for each continent the average latitude and longitude of the race tracks within that continent. i.e. each row of `central` corresponds to a single `continent` and the columns correspond to the average `lat` and `lon` of the race tracks within each continent.

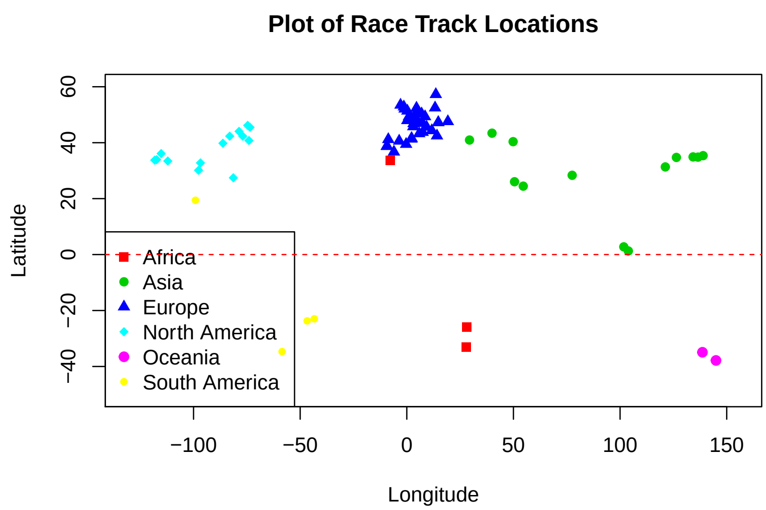
```
central <- matrix(NA, nrow=6, ncol=2)
for (i in 1:nlevels(continent))
{
s <- tracks[continent==levels(continent)[i],]
central[i,] <- c(mean(s$lng), mean(s$lat))
}
```

Q5 [**4 marks**] Produce a plot of the location of each of the tracks. Your plot should . . .

- have the track locations coloured according to the continent they are in. You can choose your own colours but in the plot shown the colours used are `2,...,7`
- have a different plotting symbol for each continent. You can choose your own plotting symbols but in the plot shown the plotting symbols used are `15, ..., 20`
- include a red dashed reference line at the Equator (`lat=0`)
- have a legend, a title and meaningful axis labels.

Your plot should look similar to the one below.

```
par(mfrow=c(1,1))
plot(lat ~ lng, col=unclass(continent)+1, xlab="Longitude",xlim=c(-130,155), ylim=c(-50,60), ylab="Lati
legend("bottomleft", pch=15:20, col=2:7,legend=levels(continent))
abline(h=0, col=2 , lty=2)
```

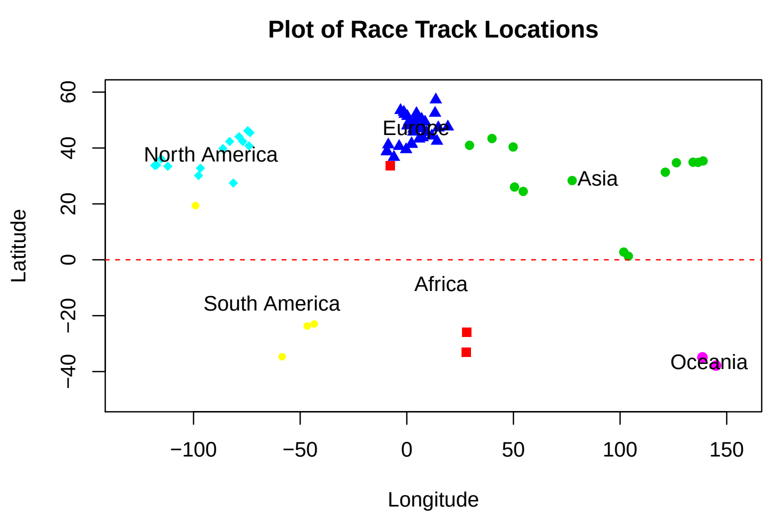


Q6 [**2 marks**] Add to your plot produced in Question 5. text labels representing the name of each continent at the points given in the matrix `central` (created in question Question 4).

You can use the `text` function to do this. Your plot should look similar to the one below.

NOTE: *I have not drawn the legend on this plot for clarity. Your answer here should only provide the code to add the labels to your existing plot, you do not need to modify the code for your existing plot.*

```
central <- do.call(rbind,by(tracks[,c("lng", "lat")], tracks$continent, colMeans))
plot(lat ~ lng, col=unclass(continent)+1, xlab="Longitude",xlim=c(-130,155), ylim=c(-50,60), ylab="Lati
#legend("bottomleft", pch=15:20, col=2:7,legend=levels(continent))
abline(h=0, col=2 , lty=2)
text(central[,1], central[,2], levels(continent))
```



Task 2

Q7 [**2 marks**] Define `mat` to be matrix with 2000 rows and 2 columns where each row contains a pair (x, y) randomly drawn from a uniform distribution on the interval $[-1, 1]$.

Hint: You can use the `R` *function* `runif(n, -1,1)` *to generate* n *random draws from a uniform distribution on the interval* $[-1, 1]$.

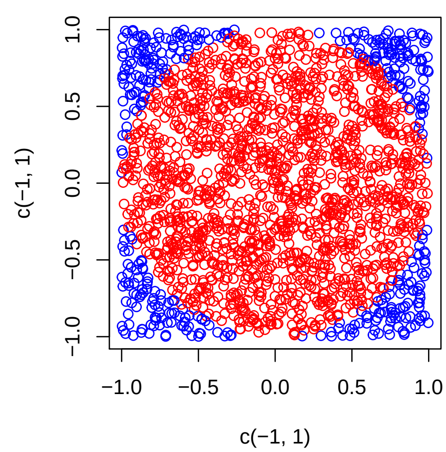
```
mat <- matrix(runif(4000,-1,1),2000,2)
```

Q8 [**4 marks**] Use the following code to set up an empty plotting canvas;

```
plot(c(-1,1), c(-1,1), type="n")
```

Plot on this plotting canvas the points you have simulated in `mat` from Q7. For each of the points (x_i, y_i), $i = 1, ..., 1000$ colour the points where $\sqrt{x_i^2 + y_i^2} <= 1$ in red and those where $\sqrt{x_i^2 + y_i^2} > 1$ in blue. Your plot should look similar to the one below.

```
par(pty="s")
within <- function(u=c(u1, u2) ){
  dist <- sqrt(sum(u^2))
  if (dist <= 1) return(TRUE) else{return(FALSE)}
}
cols <- apply(mat, 1, within)
plot(c(-1,1), c(-1,1), type="n")
points(mat, col=c(4,2)[unclass(cols)+1])
```



Task 3

The file `tempdata.txt` contains a dataframe corresponding to a time series of lake surface water temperature (lswt) recorded in Kelvin. The observations are daily and cover a time period of one year. The file `tempdata.txt` contains the following two columns;

`tempdata.txt`	
`lswt`	the value of lake surface water temperature
`day`	the day of the year the lswt was recorded, i.e. day 1 is January 1^{st}, day 2 is January 2^{nd} ... day 365 is December 31^{st}

Q9 [**2 marks**] Read the file `tempdata.txt` into `R` and save it as a dataframe called `tempdata`.

```
tempdata <- read.table("tempdata.txt", header=TRUE)
attach(tempdata)  ## attach statement not neccessary for marks
                  ## if used in future questions attach should
                  ## be shown somewhere in the script
```

Q10 [**3 marks**] The formula below can be used to convert measurements of temperature recorded in Kelvin (denoted by y_K) to Fahrenheit (denoted by y_F)

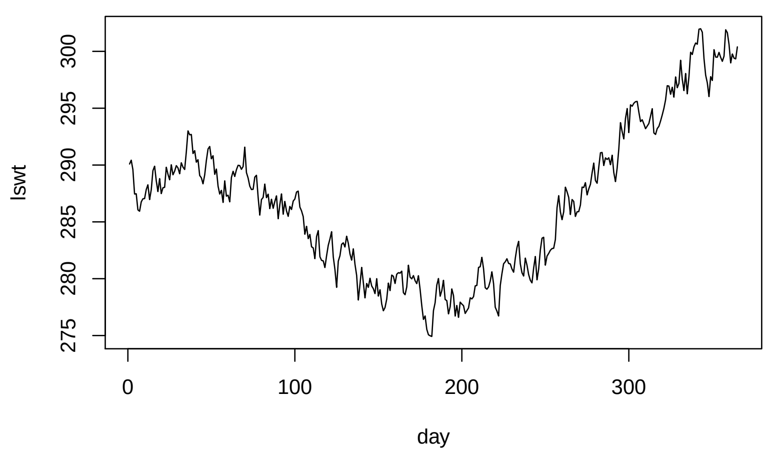
$$y_F = \frac{9\,(y_K - 273.15)}{5} + 32$$

Define a new column of the dataframe `tempdata` called `far` which contains the measurements of `lswt` from the dataframe `tempdata` converted to Fahrenheit.

```
tempdata <- transform(tempdata, far=((9/5)*(lswt-273.15))+32)
```

Q11 [**1 mark**] Produce a line plot of the time series with day of year on the x-axis and lswt on the y-axis.

Your plot should look similar to the one below.



Q12 [**4 marks**] We can model the relationship between `lswt` and `day` using harmonic regression.

For a covariate vector $x = (x_1, ..., x_n)$ and period p then the design matrix X for the harmonic regression takes the form

$$X = \begin{bmatrix} 1 & x_1 & \cos\frac{2\pi(x_1-1)}{p} & \sin\frac{2\pi(x_1-1)}{p} \\ 1 & x_2 & \cos\frac{2\pi(x_2-1)}{p} & \cos\frac{2\pi(x_2-1)}{p} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & \cos\frac{2\pi(x_n-1)}{p} & \cos\frac{2\pi(x_n-1)}{p} \end{bmatrix}$$

Define a matrix X which is the design matrix for fitting this harmonic regression model to describe the relationship between lswt and day of year (i.e. the data stored in `tempdata`). For this matrix `day` is the covariate vector x and the period is the number of observations in a year, so $p = 365$.

```
p <- 365
day <- tempdata$day
X = cbind(rep(1, p), day, cos((2*pi*(day-1))/p), sin((2*pi*(day-1))/p))
```

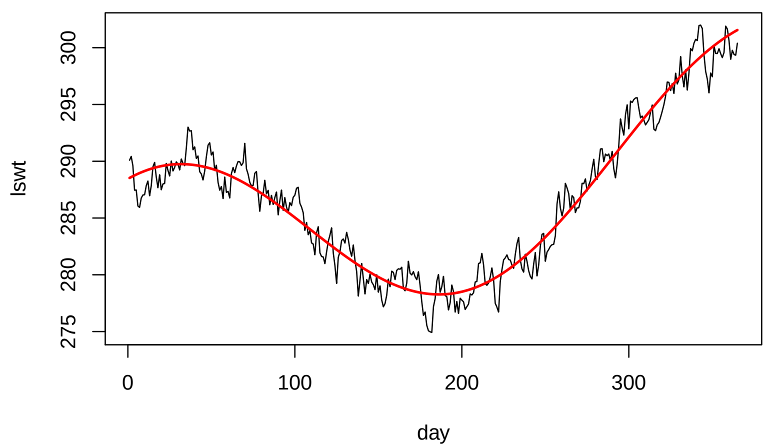
Q13 [**2 marks**] Define a vector `y.hat` which contains the fitted values of a harmonic regression fitted using the design matrix X from question Q12 and the `lswt` as the vector of responses y.

The fitted values can be computed using

$$\hat{y} = X(X^T X)^{-1} X^T y$$

```
y.hat <- X%*%solve(t(X)%*%X, t(X)%*%tempdata$lswt)
```

Q14 [**2 marks**] Add a solid red line representing the fitted values `y.hat` on the plot you produced in Question 11. Your plot should look similar to the one below.



NOTE: *If you have not managed to sucessfully compute the vector* `y.hat` *in Question 13 you may create the vector* `y.hat2` *using the code below and use this as a substitute for* `y.hat` *in Questions 14 and 15. No marks will be awarded for Question 13 if this substitute is used.*

```
library(splines)
mod <- lm(tempdata$lswt~ bs(tempdata$day, degree=6))
y.hat2 <- predict(mod)
```

Q15 [**1 mark**] Define a vector of residuals named `res` by subtracting the vector `y.hat` (defined in Question 13) from the vector of responses `lswt`. The mean of these residuals should be 0 after rounding.

```
res <- tempdata$lswt-y.hat
```

Q16 [**3 marks**] Consider a sequence of observations $r = r_1, ..., r_n$. The Durbin-Watson statistic

$$d = \frac{\sum_{i=2}^{n}(r_i - r_{i-1})^2}{\sum_{i=1}^{n} r_i^2}$$

measures how correlated subsequent observations are.

Define a variable d which contains the Durbin-Watson statistic for the time series of residuals stored in `res` (defined in Q15).

```r
n <- 365
d <- sum((res[-1]-res[-n])^2)/sum(res^2)
```

NOTE: *If you have not managed to successfully define the vector of residuals* `res` *then you can enter the line of code below to generate a subsitute for* `res` *named* `res2` *that can be used to answer Q16.*

```r
set.seed(10);res2 <- arima.sim(list(order=c(1,0,0), ar=0.7), n=365)
```
