Time series and interactive visualization

23 marks

You will need the interactive visualization system called `loon`. If you have not already installed it, do so from CRAN.

a. In this part, you will be using three different `R` functions, `paste()`, `rev()`, and `rep()`, to construct different character vectors (vectors whose elements are `"strings"`. Look at the `help()` for each of these to better understand how to use them. You will also use some existing vectors, namely `LETTERS`, `letters`, and `month.name` (just print these to see their values).

   i. *2 marks* Using the above functions and data structures, produce a vector with the following contents:

```
##  [1] "A.z" "B.y" "C.x" "D.w" "E.v" "F.u" "G.t" "H.s" "I.r" "J.q" "K.p" "L.o"
## [13] "M.n" "N.m" "O.l" "P.k" "Q.j" "R.i" "S.h" "T.g" "U.f" "V.e" "W.d" "X.c"
## [25] "Y.b" "Z.a"
```

   Show your code.

```
paste(LETTERS,rev(letters),sep='.')
```

```
##  [1] "A.z" "B.y" "C.x" "D.w" "E.v" "F.u" "G.t" "H.s" "I.r" "J.q" "K.p" "L.o"
## [13] "M.n" "N.m" "O.l" "P.k" "Q.j" "R.i" "S.h" "T.g" "U.f" "V.e" "W.d" "X.c"
## [25] "Y.b" "Z.a"
```

   ii. *2 marks* Using the above functions and data structures, produce a vector with the following contents:

```
## [1] "1(a)" "1(b)" "1(c)" "2(a)" "2(b)" "2(c)" "3(a)" "3(b)" "3(c)"
```

   Show your code.

```
paste(rep(1:3, each=3),'(', rep(letters[1:3], times=3), ')', sep='')
```

```
## [1] "1(a)" "1(b)" "1(c)" "2(a)" "2(b)" "2(c)" "3(a)" "3(b)" "3(c)"
```

   iii. *4 marks* Recall the `getYears()` function from the lecture slides which was used to extract the years from the `co2` and `sunspot.month` time series. Write a function

   The function should be using `paste()`, `rep()`, etc. You will also want to use `frequency()`. Show your code.

   Answer:

```
getTSlabels <- function(ts, freqNames) {
  l <- length(ts)
  f <- frequency(ts)
  paste(rep(freqNames, times=l/f), rep(unique(floor(time(ts))), each=f))
}

head(getTSlabels(co2, month.abb))
```

```
## [1] "Jan 1959" "Feb 1959" "Mar 1959" "Apr 1959" "May 1959" "Jun 1959"
```

```
tail(getTSlabels(co2, month.abb))
```

```
## [1] "Jul 1997" "Aug 1997" "Sep 1997" "Oct 1997" "Nov 1997" "Dec 1997"
```

b. In this question you are going to interactively explore the `co2` data.

Construct the interactive plot

```
library(loon)
pointlabels <- getTSlabels(co2, month.abb)
decomp <- stl(co2, s.window = 6, s.degree = 1)
p_stl <- l_plot(decomp, linkingGroup = "co2",
                itemLabel = pointlabels,
                showItemLabels = TRUE)
```

i. *2 marks* Click on the plot of residuals. Using only the mouse, identify the month and year of the most negative residual. Identify the month and year of the most positive residual. Report your answers. (It might be helpful to zoom in on the residuals using CMD/ALT and scrolling; select scale to plot in the inspector to get back to original.)

The most negative: April 1971. The most positive: March 1985.

ii. *1 mark* You can select a large number of points by clicking on the plot background and moving the mouse (while the left mouse button is held down). This is called *sweep selection*. Try it on any plot. What happens to the points in the other plots?

The corresponding points in other plots are automatically selected.

iii. *2 marks* Use sweep selection on the plot of residuals to select only the most positive 10-20 residuals. These are places where the fit of the model is worst. Ideally, there should be no obvious pattern in where these poorer fits occur in our model. For example, the poor fit might be only at the beginning of each season, or at the seasonal extremes. Comment on whether there are any such obvious patterns here.

There are no bad points in the middle of the plot (or around 1980). All bad points are in both ends.

iv. Click on the plot of the seasonal component; its picture should now appear in the inspector window. In the inspector window, click on the "Layers" tab and once there select the label "line". With "line" selected, go to the bottom ans click on a button that looks like an eye with a stroke through it; this will turn off the lines in the plot. Select the "Analysis" tab to get back to where you were.

Now click again on the plot of the seasonal component. While holding the CTRL key down, scroll so that the series shrinks and expands only **horizontally**.

Shrink the plot horizontally until separate curves are distinguishable.

It might be helful to construct the following plot *as well*:

```
seasonal <- decomp$time.series[, "seasonal"]
p <- l_plot(seasonal, linkingGroup = "co2", ylabel = "seasonal estimate",
            itemLabel = pointlabels, showItemLabels = TRUE)
```

- *2 marks* What does the highest curve represent? The lowest? What does each curve represent?

  The highest curve represents the CO_2 level in April and May. The lowest represents the CO_2 level in September and October. Each curve prepresents a certain CO_2 level in particular months in a year.

- *2 marks* Recall how the `stl` decomposition is constructed. Explain how each of these curves was constructed and how the seasonal pattern is constructed. Horizontal zooming will help you understand these ideas.

  These curves are constructed by taking points that are close to each other because close points having the same CO_2 level. So, we are basically decomposing the curve and using points that are close to each other to provide estimate and construct curves.

c. The sunspot data. Consider the following interactive plot:

```
p <- l_plot(sunspot.month,
        ylabel="Mean sqrt monthly sunspots",
        xlabel="Time",
```

```
        title="Sunspot activity",
        linkingGroup="sunspots",
        size=1,
        showGuides=TRUE)
l_layer_line(p, sunspot.month)
```

There are 24 *complete* cycles in this data if we measure a cycle as being peak to peak. The objective of this question is to capture the lengths of these cycles.

Using *sweep selection*, an entire cycle can be selected at once (from peak to peak). To record the values, we construct a numeric vector in `R`

```
cycle_lengths <- numeric()
```

and note that

```
sum(p["selected"])
```

will at anytime return the count of the number of selected points.

You could do these for each cycle and record the count, OR, you could accumulate all 24 by executing the following

```
cycle_lengths <- c(cycle_lengths, sum(p["selected"]))
```

immediately after each new cycle has been selected on the plot `p`.

Accumulate the 24 cycle lengths (each being a "V" in the plot `p`) in this way by selecting the "V"s left to right one after the other in the plot `p`.

When you have all 24 cycle lengths,

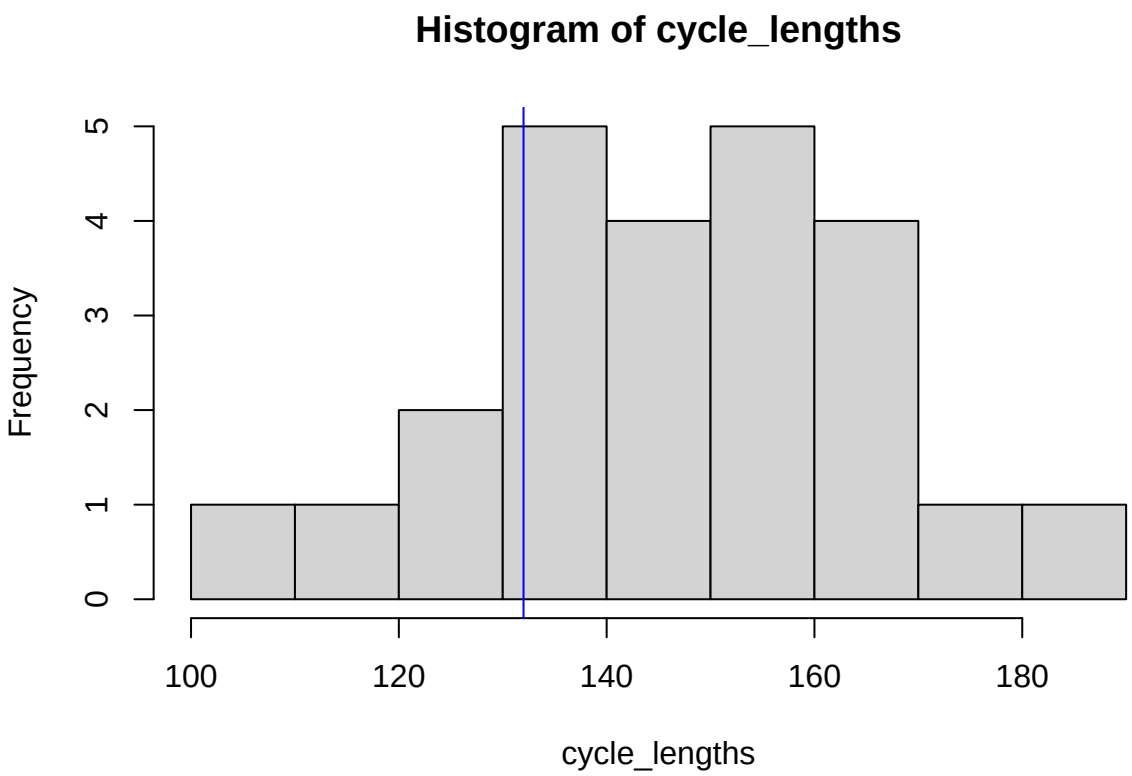
- *1 mark* print them;

  ```
  cycle_lengths
  ```

  ```
  ##  [1] 153 138 169 150 157 169 158 106 138 170 163 176 139 144 132 139 150 113 157
  ## [20] 145 125 153 183 130
  ```

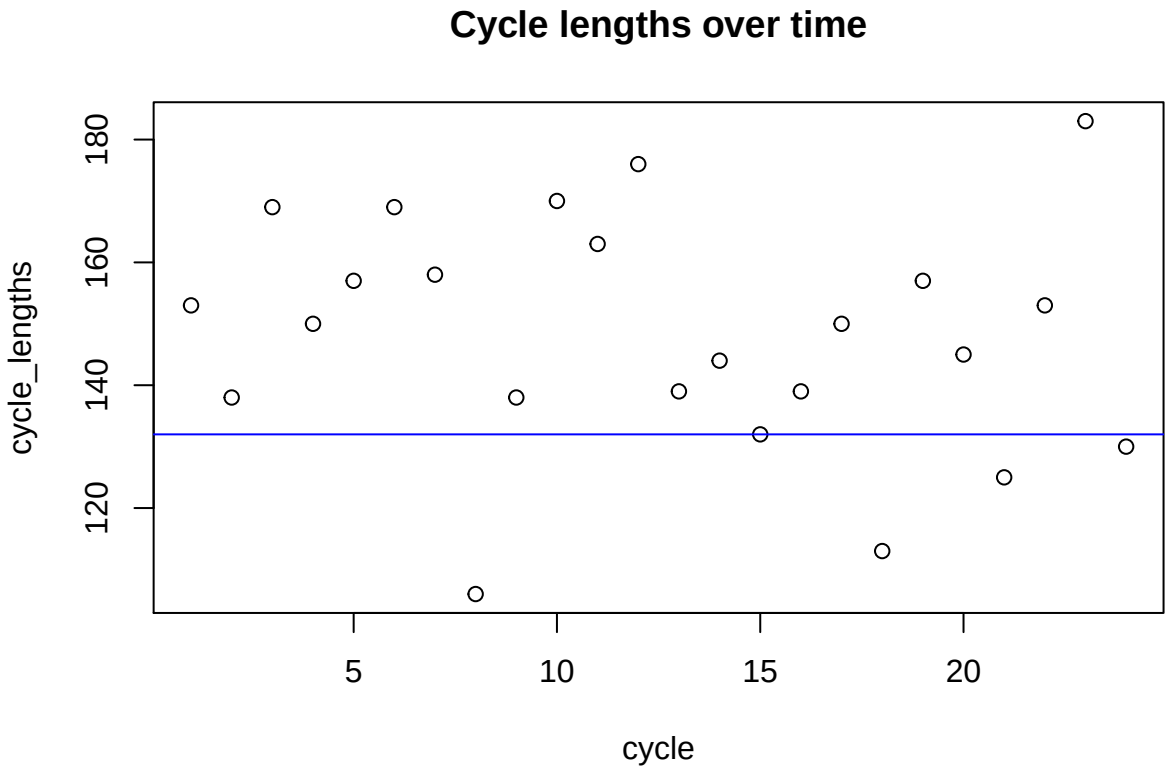
- *1 mark* plot (and hand in) a histogram of the `cycle_lengths` and mark on the histogram with a vertical line (see `abline()`) the 11 year cycle length;

  ```
  hist(cycle_lengths)
  abline(v=11*12, col='blue')
  ```



- *1 mark* plot (and hand in) the cycle lengths over time (again with the 11 year cycle marked, now as a horizontal line)

```
plot(1:24, cycle_lengths, xlab='cycle', main='Cycle lengths over time')
abline(h=11*12, col='blue')
```



- *3 marks* Comment on your findings about the cycle length of sunspots.
  Cycles are sometimes longer and sometimes shorter than 11 years. The dsitribution seems to follow normal distribution.