Problem 1 (13 credits)

HW1

Michael Brucek (mbrucek), Eduardo Chavez (echavezh), Kevin Grady (grady133), Danny Moncada (monca016), Tian Zhang (zhan7003)

November 03, 2020

```
suppressWarnings(suppressPackageStartupMessages({
  library(TSA)
  library(forecast)
  library(ggplot2)
  library(dplyr)
}))
```

Binary Random Walk

Assume Y_t is a ***binary*** random walk as we illustrated in the class, such that

$$\begin{cases} Y_1 = e_1 \\ Y_2 = e_1 + e_2 \\ \vdots \\ Y_t = \sum_{i=1}^{t} e_i \end{cases}$$

where $e_i \in \{-1, 1\}$ i.i.d

Note that this is not the Gaussian random walk where $e_t \sim N(0, 1)$ but a binary one where e_t can only take two values: -1 and 1 each with probability 50%.

Question 1

a) (1 credit)

Please compute analytically:

$E[Y_3] =$

```
E_Y3 <- 0
```

(please include only the numerical answer above not the computation. An acceptable format for your answer is such as `E_Y3 <- 5`. Please do not rename this variable)

b) (1 credit)

Please compute analytically:

$\text{Var}[Y_3] =$

```
Var_Y3 <- 3
```

c) (1 credit)

Please compute analytically:

$\text{Cov}(Y_2, Y_3) =$

```
Cov_Y2Y3 <- 2
```

d) (1 credit)

Please compute analytically:

$\text{Cov}(Y_5, Y_{10}) =$

```
Cov_Y5Y10 <- 5
```

Question 2

a) (1 credit)

Please generate one sample path of length $T = 100$ for this binary random walk.

- Please save it into a data.frame `df2a` column `df2a$Y`

```
set.seed(42) # Please do not change the seed

T <- 100L

df2a <- data.frame(Y = rep(sample(c(-1, 1), T, TRUE, prob = c(0.5, 0.5))))
```

b) (2 credits)

Please generate $N = 100$ sample paths of length $T = 100$ for this binary random walk.

- Please save the results into a data.frame `df2b` where:
  - column `df2b$Y` has the values of the process
  - column `df2b$id` has the id of the sample path
  - column `df2b$t` has the time

```
set.seed(42) # Please do not change the seed

N <- 100L
T <- 100L

df2b <- data.frame(e = rep(sample(c(-1, 1), N*T, TRUE, prob = c(0.5, 0.5))),
                   id = rep(1:N, each = T),
                   t = rep(1:T, N))
```

```
df2b <- df2b %>% group_by(id) %>%
            mutate(Y = cumsum(e)) %>% select(Y, id, t)

head(df2b)
```

```
## # A tibble: 6 x 3
## # Groups:   id [1]
##       Y    id     t
##   <dbl> <int> <int>
## 1    -1     1     1
## 2    -2     1     2
## 3    -1     1     3
## 4    -2     1     4
## 5    -3     1     5
## 6    -4     1     6
```

c) (1 credit)

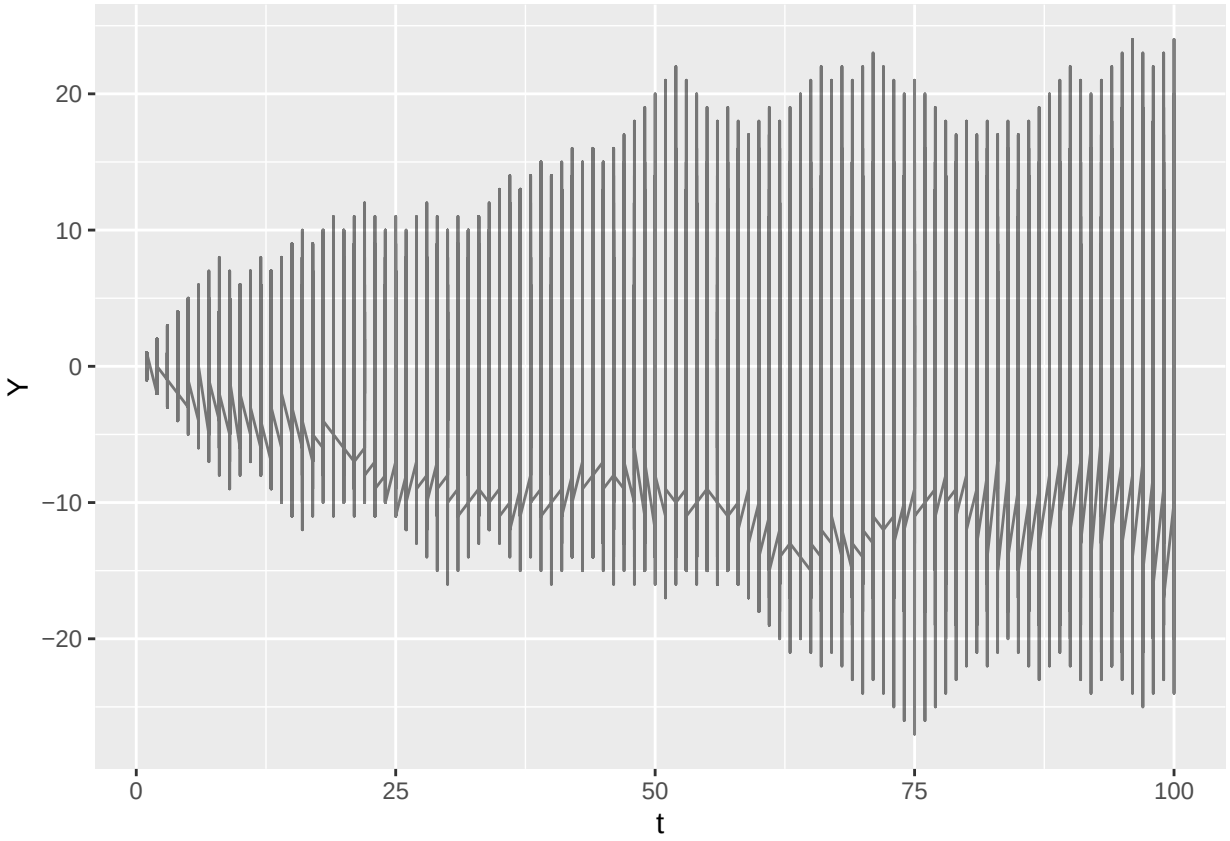
Please plot the sample paths that you generated in the previous question

- Please save your plot into variable `p2c` and plot it

Hints:

- use `ggplot` and take advantage of the long format of the data
- please don't change the color (keep the lines black) but do put `alpha=0.5` into your `geom_line` to make sample paths somewhat transparent.
- do not use `geom_points` just `geom_line` is fine
- As you will see from your plot:
  - the fainter the line the less likely the random walk would reach this spot

```
p2c <- ggplot(data = df2b, aes(x = t, y = Y)) + geom_line(alpha = 0.5)
p2c
```



d) (1 credit)

Please use the code that you wrote before to generate $N = 5000L$ sample paths.

- Please save the result into `df2d` data.frame following conventions of the prior question
  - column `df2d$Y` has the values of the process
  - column `df2d$id` has the id of the sample path
  - column `df2d$t` has the time

```
set.seed(42) # Please do not change the seed

N <- 5000L
T <- 100L

df2d <- data.frame(e = rep(sample(c(-1, 1), N*T, TRUE, prob = c(0.5, 0.5))),
                   id = rep(1:N, each = T),
                   t = rep(1:T, N))

df2d <- df2d %>% group_by(id) %>%
            mutate(Y = cumsum(e)) %>% select(Y, id, t)

head(df2d)
```

```
## # A tibble: 6 x 3
```

```
## # Groups:   id [1]
##       Y    id     t
##   <dbl> <int> <int>
## 1    -1     1     1
## 2    -2     1     2
## 3    -1     1     3
## 4    -2     1     4
## 5    -3     1     5
## 6    -4     1     6
```

e) (2 credits)

Use the data in `df2d` to numerically verify your analytical results in Question 1 (a,b,c,d)

```
#Please enter your code here first, AND then answer:

## Create Y3 variable by subsetting on t == 3
Y3 = df2d$Y[df2d$t == 3]
## Create Y2 variable by subsetting on t == 2
Y2 = df2d$Y[df2d$t == 2]
## Create Y5 and Y10 vars by subsetting on t == 5 and t == 10
Y5 = df2d$Y[df2d$t == 5]
Y10 = df2d$Y[df2d$t == 10]

E_Y3 <- mean(Y3)
paste("The mean of the random walk process is:", E_Y3)
```

```
## [1] "The mean of the random walk process is: 0.0068"
```

```
Var_Y3 <- var(Y3)
paste("The variance at Y3 is: ", Var_Y3)
```

```
## [1] "The variance at Y3 is:  3.01335643128626"
```

```
Cov_Y2_Y3 <- cov(Y2, Y3)
paste("The covariance of the random walk process at Y2, Y3 is:", Cov_Y2Y3)
```

```
## [1] "The covariance of the random walk process at Y2, Y3 is: 2"
```

```
Cov_Y5_Y10 <- cov(Y5, Y10)
paste("The covariance of the random walk process at Y5, Y10 is:", Cov_Y5Y10)
```

```
## [1] "The covariance of the random walk process at Y5, Y10 is: 5"
```

Hints:

- For Y_3, extract rows corresponding to `[df2d$t==3]`.
- Since 5000 sample paths are used, the results should be very close to the analytical ones

f) (1 credit)

Assume now that you just got a phone call from your friend telling you that this random walk has just been observed to pass through the point $Y_6 = 6$.

- What that means is you know that all first 6 steps of the random walk happened to be in the "up" direction not "down".

In other words, as this new information from your friend became you can refine your forecast: you can remove the sample paths that the process has not taken and only concentrate on the remaining sample paths that it could still take

- This idea is simulation-based forecasting

Please create data.frame `df2e` such that it only has the sample paths (and only those sample paths) from the previous answer that satisfy the condition that $Y_6 = 6$.

Hints:

- use `group_by`
- the cleanest way is to use dplyr's `do()`
- less clean but also doable: you could create a separate data.frame with acceptable sample path ids and then `inner_join` it to the main data.frame to keep only the good sample paths in it.

```
df2e <- df2d %>% group_by(id) %>% do(filter(., Y[6] == 6))

head(df2e)
```

```
## # A tibble: 6 x 3
## # Groups:   id [1]
##       Y    id     t
##   <dbl> <int> <int>
## 1     1    77     1
## 2     2    77     2
## 3     3    77     3
## 4     4    77     4
## 5     5    77     5
## 6     6    77     6
```

g) (1 credit)

Please use the data.frame from the previous question to plot your results.

- Please don't change the color (keep the lines black) but do put `alpha=0.5` into your `geom_line` to make sample paths somewhat transparent.
- Please do plot the first 6 time periods as well. Yes, the first 6 steps will be exactly the same for all sample paths (since all other sample paths have been eliminated).

Key takeaways:

- You should see from your plot why people say that random walks have memory and are not mean reverting.
- This idea of sequentially eliminating sample paths as you observe the process in order to forecast the future better is called *filtration*. This idea is fundamental in continuous-time stochastic processes but we will only touch upon it for discrete-time stochastic processes.

```
p2e <- ggplot(data = df2e, aes(x = t, y = Y)) + geom_line(alpha = 0.5)
p2e
```