STA 6366, Homework.1

Carson Slater *Baylor University*

1. **Suppose that $X \sim$ Gamma$(\alpha, 1)$ and $Y \sim$ Gamma$(\beta, 1)$. Show that $Z = \frac{X}{X+Y} \sim$ Beta(α, β). Use MGFs at least once in your answer.**

We have that $X \sim$ Gamma$(\alpha, 1)$, and $Y \sim$ Gamma$(\beta, 1)$. Let $Z = \frac{X}{X+Y}$ and $S = X + Y$. By MGFs, $M_S(t) = M_X(t)M_Y(t) = (1-t)^{-\alpha}(1-t)^{-\beta} = (1-t)^{-(\alpha+\beta)}$, so $S \sim$ Gamma$(\alpha + \beta, 1)$. Using the transformation $X = ZS$ and $Y = S(1-Z)$, the Jacobian is:

$$\begin{aligned} J &= \det \begin{bmatrix} \frac{\partial X}{\partial Z} & \frac{\partial X}{\partial S} \\ \frac{\partial Y}{\partial Z} & \frac{\partial Y}{\partial S} \end{bmatrix} \\ &= \det \begin{bmatrix} S & Z \\ -S & 1-Z \end{bmatrix} \\ &= (S)(1-Z) - (Z)(-S) \\ &= S - SZ + SZ \\ &= S. \end{aligned}$$

We have that since $X, Y > 0$, then $S > 0$ and $0 < z < 1$. The joint density is then:

$$\begin{aligned} f_{Z,S}(z,s) &= f_{X,Y}(zs, (1-z)s) \cdot s \\ &= \frac{1}{\Gamma(\alpha)\Gamma(\beta)}(zs)^{\alpha-1}e^{-zs}(s(1-z))^{\beta-1}e^{-s(1-z)} \cdot s \\ &= \underbrace{\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}z^{\alpha-1}(1-z)^{\beta-1}}_{f_Z(z) \sim \text{Beta}(\alpha,\beta)} \cdot \frac{1}{\Gamma(\alpha+\beta)}s^{\alpha+\beta-1}e^{-s} \end{aligned}$$

Since the joint PDF factors into the pdf of of z and the pdf of of s, we have shown $Z \sim$ Beta(α, β).

2. **Suppose a random string of 100 letters from a 4-letter alphabet "A", "B", "C", "D" is generated with each letter equally likely to be placed at each spot. We are interested in knowing if the exact sequence "BDCAC" shows up at least once in the 100-letter string.**

a. **Let X be the number of times the sequence appears in the string. Explain why X does not follow a binomial distribution.**

X does not follow a binomial distribution because of the idea of overlapping windows inducing dependence between 'trials'. For an instance of X, five slots must follow the specified sequence exactly, meaning that for each slot, the probability of success is contingent on the realizations of the prior four slots.

b. **Calculate $P(X \geq 1)$ using (i) the Markov Chain approach and (ii) using a Binomial approximation. Do these results align with what we established in (a)? Explain.**

For the Markov Chain approach given the 4-letter alphabet, each transition occurs with probability $p = 1/4$ if the next letter continues the pattern, and we stay in or reset to a previous state with the remaining probability. The transition matrix $\mathbf{P}$ is defined as:

$$\mathbf{P} = \begin{pmatrix} 3/4 & 1/4 & 0 & 0 & 0 & 0 \\ 2/4 & 1/4 & 1/4 & 0 & 0 & 0 \\ 3/4 & 0 & 0 & 1/4 & 0 & 0 \\ 2/4 & 1/4 & 0 & 0 & 1/4 & 0 \\ 2/4 & 1/4 & 0 & 0 & 0 & 1/4 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

We then apply the transition matrix 100 times to the initial state vector $\mathbf{s} = (1, 0, 0, 0, 0, 0)'$

```r
# QUESTION 2b ################################################

# Define the transition matrix
# fmt: skip
TrM <- matrix(c(
  3/4, 1/4, 0,   0,   0,   0,
  2/4, 1/4, 1/4, 0,   0,   0,
  3/4, 0,   0,   1/4, 0,   0,
  2/4, 1/4, 0,   0,   1/4, 0,
  2/4, 1/4, 0,   0,   0,   1/4,
  0,   0,   0,   0,   0,   1
), nrow = 6, ncol = 6, byrow = TRUE)

# Initial state vector (starting with 0 letters matched)
v0 <- matrix(c(1, 0, 0, 0, 0, 0), nrow = 1)

get_state_n <- function(v, P, n) {
  res <- v
  for (i in 1:n) {
    res <- res %*% P
  }
  res
}

final_probs <- v0 |> get_state_n(TrM, 100)

# The probability of being in State 5 of 0:5
cat("True P(X >= 1) = ", final_probs[6])
```

True P(X >= 1) = 0.084874

Ignoring dependence discussed in (a), we could approximate this with $X \sim \text{Bin}(96, (1/4)^5)$, as there are 96 windows.

The approximation would be:

```
cat("Binomial approximation P(X >= 1) = ", 1 - dbinom(0, 96, (1 / 4)5))
```

Binomial approximation P(X >= 1) = 0.08953134

This result is comparable to the Markov Chain result, but not exact.

c. **Simulate X in R (or something similar) with at least 100k samples (do not try to create all permutations). Compare your simulated results to the binomial approximation for $X = 0, 1, 2, 3, 4, 5$ and $X > 5$.**

Table 1: Comparison of Probabilities for 'BDCAC' Occurrences

X_val	Binomial	Simulated
0	0.91047	0.91202
1	0.08544	0.08427
2	0.00397	0.00360
3	1.2151e-04	1.1000e-04
4	2.7616e-06	0.00000
5	4.9671e-08	0.00000
>5	7.4576e-10	0.0000e+00

3. **Suppose that X, Y are *iid* random variables with CDF $F(\cdot)$. Do not use the order statistic density formula for these problems.**

a. **Let $V = \max(X, Y)$. Show that the CDF of V is $F_V(t) = F(t)^2$.**

We know X and Y, have the same CDF, $F(t) = P(X \leq t)$. Then, we have that

$$
\begin{aligned}
F_V(t) &= P(V \leq t) \\
&= P(\max(X, Y) \leq t) \\
&= P(X \leq t \cap Y \leq t) \quad \text{(Because saying the max is less than } t \text{ is saying both are less than } t) \\
&= P(X \leq t)P(Y \leq t) \quad \text{(by independence)} \\
&= F(t)F(t) \quad (\text{since } X, Y \overset{iid}{\sim} F) \\
&= F(t)^2
\end{aligned}
$$

b. **If $U = \min(X, Y)$, find the CDF of U.**

$$\begin{aligned} F_U(t) &= P(U \leq t) \\ &= 1 - P(U > t) \\ &= 1 - P(\min(X, Y) > t) \\ &= 1 - P(X > t \cap Y > t) \\ &= 1 - P(X > t)P(Y > t) \quad \text{(by independence)} \\ &= 1 - [1 - F(t)][1 - F(t)] \\ &= 1 - (1 - F(t))^2 \\ &= 2F(t) - F(t)^2 \end{aligned}$$

4. **Suppose n random variables (*iid*) are drawn from a normal distribution with mean μ and variance $\sigma^2 = 1$. Suppose that $\mu = 0$ under H_0, and $\mu > 0$ under H_a. Using the maximum value of these random variables as the test statistic, how could you calculate the p-value for this hypothesis without running a simulation? Use this to calculate a p-value for $X_{(n)} = 4.2$ with $n = 1,000$, $n = 10,000$ and $n = 1,000,000$. (Hint: your answer to (3) might come in handy for this problem.)**

Under H_0, let $X_1, \ldots, X_n \overset{iid}{\sim} \mathcal{N}(0, 1)$. Let $\Phi(t)$ be the CDF of $\mathcal{N}(0, 1)$. The p-value is defined as $P(X_{(n)} \geq 4.2 \mid H_0)$. We generalize the result from (3a) to find the CDF of the maximum as $F_{X_{(n)}}(t) = \Phi(t)^n$. Then

$$\begin{aligned} p\text{-value} &= 1 - P(X_{(n)} < 4.2) \\ &= 1 - F_{X_{(n)}}(4.2) \\ &= 1 - [\Phi(4.2)]^n. \end{aligned}$$

The results are shown as code output:

```
## # A tibble: 3 x 2
##         n p_value
##     <dbl>   <dbl>
## 1    1000  0.0133
## 2   10000  0.125
## 3 1000000  1.000
```

5. **Let $X \sim$ Exp(λ).**

a. **For any $s, t > 0$, show that $P(X > s + t | X > s) = P(X > t)$.**

We show this result:

$$\begin{aligned} P(X > s+t \mid X > s) &= \frac{P(X > s+t \cap X > s)}{P(X > s)} \\ &= \frac{P(X > s+t)}{P(X > s)} \quad \text{(since } s+t > s\text{)} \\ &= \frac{e^{-\lambda(s+t)}}{e^{-\lambda s}} \\ &= \frac{e^{-\lambda s}e^{-\lambda t}}{e^{-\lambda s}} \\ &= e^{-\lambda t} \\ &= P(X > t). \end{aligned}$$

b. **Explain intuitively what this means.**

This is the fabled 'memoryless property' of the exponential distribution. In temporal settings, it means that the probability for X occuring in the future is unaffected by the how much time has already passed.

c. **Can you think of a discrete distribution that has this property?**

The Geometric distribution has this property. Suppose $Y \sim \text{Geom}(p)$, then

$$\begin{aligned} P(Y > s+t \mid X > s) &= \frac{P(Y > s+t \cap Y > s)}{P(Y > s)} \\ &= \frac{P(Y > s+t)}{P(Y > s)} \quad \text{(for integers } s, t \geq 0\text{)} \\ &= \frac{(1-p)^{s+t}}{(1-p)^s} \\ &= (1-p)^t \\ &= P(Y > t). \end{aligned}$$

Appendix

```
knitr::opts_chunk$set(
  dev = "cairo_pdf",
  fig.width = 5,
  fig.height = 5,
  fig.align = "center",
  echo = FALSE,
  message = FALSE,
  warning = FALSE,
  error = FALSE,
  results = "markup"
)

# Load required libraries
library("tidyverse")
library("patchwork")
library("glue")
library("scales", warn.conflicts = FALSE)
library("extrafont")
library("tinytex")
library("knitr")
library("tidyr")
library("latex2exp")
library("MASS")
library("kableExtra")

theme_set(theme_minimal(base_family = "Roboto Condensed"))

conflicted::conflicts_prefer(
  readr::col_factor(),
  purrr::discard(),
  dplyr::lag(),
  readr::parse_date(),
  kableExtra::group_rows(),
  dplyr::select
)
# QUESTION 2b ################################################

# Define the transition matrix
# fmt: skip
TrM <- matrix(c(
  3/4, 1/4, 0,   0,   0,   0,
  2/4, 1/4, 1/4, 0,   0,   0,
  3/4, 0,   0,   1/4, 0,   0,
  2/4, 1/4, 0,   0,   1/4, 0,
  2/4, 1/4, 0,   0,   0,   1/4,
```

```
  0,   0,   0,   0,   0,   1
), nrow = 6, ncol = 6, byrow = TRUE)

# Initial state vector (starting with 0 letters matched)
v0 <- matrix(c(1, 0, 0, 0, 0, 0), nrow = 1)

get_state_n <- function(v, P, n) {
  res <- v
  for (i in 1:n) {
    res <- res %*% P
  }
  res
}

final_probs <- v0 |> get_state_n(TrM, 100)

# The probability of being in State 5 of 0:5
cat("True P(X >= 1) = ", final_probs[6])
cat("Binomial approximation P(X >= 1) = ", 1 - dbinom(0, 96, (1 / 4)5))

# QUESTION 2c ##################################################
set.seed(613) # My birthday - I always pick this for the seed

# Parameters
n_sims <- 100000
string_len <- 100
alphabet <- c("A", "B", "C", "D")
target_pattern <- "BDCAC"

p <- (1 / 4)5
n_trials <- string_len - 5 + 1

# Simulation function
run_pattern_sim <- function(n, len, alpha, pattern) {
  replicate(n, {
    # Generate random string
    sample(alpha, len, replace = TRUE) |>
      paste0(collapse = "") |>
      stringr::str_count(paste0("(?=", pattern, ")"))
  })
}

set.seed(123)
sim_results <- run_pattern_sim(n_sims, string_len, alphabet, target_pattern)

# Frequency table
sim_df <- tibble(X = sim_results) |>
  count(X) |>
```

```r
  mutate(Simulated = n / n_sims) |>
  select(-n)


# 1. Create the comparison table with scientific notation
comparison_table <- tibble(
  X_val_num = 0:5
) |>
  mutate(
    # Calculate raw values
    Binomial_raw = dbinom(X_val_num, n_trials, p),
    Simulated_raw = map_dbl(X_val_num, ~ sum(sim_results == .x) / n_sims),

    # Format for precision: Use scientific notation for rare events
    Binomial = ifelse(
      Binomial_raw < 0.001,
      formatC(Binomial_raw, format = "e", digits = 4),
      formatC(Binomial_raw, format = "f", digits = 5)
    ),
    Simulated = ifelse(
      Simulated_raw < 0.001 & Simulated_raw > 0,
      formatC(Simulated_raw, format = "e", digits = 4),
      formatC(Simulated_raw, format = "f", digits = 5)
    ),
    X_val = as.character(X_val_num)
  ) |>
  select(X_val, Binomial, Simulated)

# 2. Add the tail (>5)
gt_5 <- tibble(
  X_val = ">5",
  Binomial_raw = 1 - pbinom(5, n_trials, p),
  Simulated_raw = sum(sim_results > 5) / n_sims
) |>
  mutate(
    Binomial = formatC(Binomial_raw, format = "e", digits = 4),
    Simulated = formatC(Simulated_raw, format = "e", digits = 4)
  ) |>
  select(X_val, Binomial, Simulated)

# 3. Final Bind and Table
final_table <- bind_rows(comparison_table, gt_5)

final_table |>
  kbl(
    format = "latex",
    booktabs = TRUE,
    align = "lcc",
```

```
    caption = "Comparison of Probabilities for 'BDCAC' Occurrences",
    escape = FALSE # Necessary if you have special LaTeX characters in your st
  ) |>
  kable_styling(latex_options = c("striped", "hold_position"))

# QUESTION 4 ################################################
obs_max <- 4.2
n_values <- c(1e3, 1e4, 1e6)

p_value_results <- tibble(n = n_values) |>
  mutate(
    p_value = 1 - pnorm(obs_max)n
  )

print(p_value_results)
```