Mathematical practice final exam 2025

2025-07-02

1.

Find the inverse of the following matrix and verify it using the `all.equal()` function.

$$\begin{pmatrix} 9 & 4 & 12 & 2 \\ 5 & 0 & 7 & 9 \\ 2 & 6 & 8 & 0 \\ 9 & 2 & 9 & 11 \end{pmatrix}$$

```
A <- matrix(
  c(9, 4, 12, 2,
    5, 0, 7, 9,
    2, 6, 8, 0,
    9, 2, 9, 11),
  nrow = 4,
  byrow = TRUE
)

A_inv <- solve(A)
product <- A %*% A_inv
unit_matrix <- diag(4)
is_equal <- all.equal(product, unit_matrix)

cat("Original Matrix A:\n")
```

```
## Original Matrix A:
```

```
print(A)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    9    4   12    2
## [2,]    5    0    7    9
## [3,]    2    6    8    0
## [4,]    9    2    9   11
```

```
cat("\nInverse Matrix A_inv:\n")
```

```
##
## Inverse Matrix A_inv:
```

```
print(round(A_inv, 4))
```

```
##         [,1]  [,2]    [,3]    [,4]
## [1,]  0.0857 -0.26 -0.1229  0.1971
## [2,] -0.1429 -0.30  0.1714  0.2714
## [3,]  0.0857  0.29  0.0271 -0.2529
## [4,] -0.1143  0.03  0.0471  0.0871
```

```
cat("\nResult of Matrix Multiplication (A × A_inv):\n")
```

```
##
## Result of Matrix Multiplication (A × A_inv):
```

```
print(round(product, 4))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```
cat("\nIs it equal to the identity matrix?", is_equal, "\n")
```

```
##
## Is it equal to the identity matrix? TRUE
```

2.

Execute the following lines which create two vectors of random integers which are chosen with replacement from the integers $0, 1, \ldots, 999$. Both vectors have length 250.

```
xVec <- sample(0:999, 250, replace=T)
yVec <- sample(0:999, 250, replace=T)
```

a. Create the vector $(y_2 - x_1, \cdots, y_n - x_{n-1})$.

```
vec_a <- yVec[2:250] - xVec[1:249]
length(vec_a)
```

```
## [1] 249
```

b. Pick out the values in yVec which are > 600.

```
vec_b <- yVec[yVec > 600]
head(vec_b)
```

```
## [1] 975 611 815 727 676 832
```

c. What are the index positions in yVec of the values which are > 600?

```
idx_c <- which(yVec > 600)
head(idx_c)
```

```
## [1]  2  3  4  5  9 11
```

d. Sort the numbers in the vector xVec in the order of increasing values in yVec.

```
x_sorted_by_y <- xVec[order(yVec)]
head(yVec[order(yVec)])
```

```
## [1]  3  7 10 13 15 16
```

```
head(x_sorted_by_y)
```

```
## [1] 889 138 648 963 300 275
```

e. Pick out the elements in yVec at index positions $1, 4, 7, 10, 13, \cdots$

```
idx_e <- seq(from = 1, to = 250, by = 3)
vec_e <- yVec[idx_e]
length(vec_e)
```

```
## [1] 84
```

```
head(vec_e)
```

```
## [1] 226 815 594 446 830 280
```

3.

For this problem we'll use the (built-in) dataset state.x77.

```
data(state)
state.x77 <- as_tibble(state.x77, rownames  = 'State')
```

a. Select all the states having an income less than 4300, and calculate the average income of these states.

```
avg_income_low <- state.x77 %>%
  filter(Income < 4300) %>%
  summarize(avg_income = mean(Income))
print(avg_income_low)
```

```
## # A tibble: 1 × 1
##   avg_income
##        <dbl>
## 1      3831.
```

b. Sort the data by income and select the state with the highest income.

```
highest_income_state <- state.x77 %>%
  arrange(desc(Income)) %>%
  slice(1) %>%
  pull(State)
cat("the state with the highest income: ", highest_income_state, "\n")
```

```
## the state with the highest income:  Alaska
```

c. Add a variable to the data frame which categorizes the size of population: $<= 4500$ is `S` , > 4500 is `L` .

d. Find out the average income and illiteracy of the two groups of states, distinguishing by whether the states are small or large.

4.

a. Write a function to simulate `n` observations of (X_1, X_2) which follow the uniform distribution over the square $[0, 1] \times [0, 1]$.

```
sim_unif_square <- function(n) {
  X1 <- runif(n, min = 0, max = 1)
  X2 <- runif(n, min = 0, max = 1)
  data.frame(X1 = X1, X2 = X2)
}
set.seed(123)
n <- 10000
data <- sim_unif_square(n)
head(data)
```

	X1 <dbl>	X2 <dbl>
1	0.2875775	0.3105917
2	0.7883051	0.3245201
3	0.4089769	0.8702542
4	0.8830174	0.3286738
5	0.9404673	0.1257012
6	0.0455565	0.3562214

6 rows

b. Write a function to calculate the proportion of the observations that the distance between (X_1, X_2) and the nearest edge is less than 0.25, and the proportion of them with the distance to the nearest vertex less than 0.25.

```
calculate_proportions <- function(data) {
  if (!all(c("X1", "X2") %in% colnames(data))) {
    stop("Input data must contain 'X1' and 'X2' columns (coordinates of points).")
  }
  x <- data$X1
  y <- data$X2
  dist_to_left <- x              # Distance to x=0 (left edge)
  dist_to_right <- 1 - x         # Distance to x=1 (right edge)
  dist_to_bottom <- y            # Distance to y=0 (bottom edge)
  dist_to_top <- 1 - y           # Distance to y=1 (top edge)
  min_edge_dist <- pmin(dist_to_left, dist_to_right, dist_to_bottom, dist_to_top)
  prop_edge <- mean(min_edge_dist < 0.25)
  dist_to_00 <- sqrt(x^2 + y^2)                  # Distance to (0,0)
  dist_to_01 <- sqrt(x^2 + (1 - y)^2)            # Distance to (0,1)
  dist_to_10 <- sqrt((1 - x)^2 + y^2)            # Distance to (1,0)
  dist_to_11 <- sqrt((1 - x)^2 + (1 - y)^2)      # Distance to (1,1)
  min_vertex_dist <- pmin(dist_to_00, dist_to_01, dist_to_10, dist_to_11)

  prop_vertex <- mean(min_vertex_dist < 0.25)
  list(
    prop_edge_less_0.25 = prop_edge,
    prop_vertex_less_0.25 = prop_vertex
  )
}
proportions <- calculate_proportions(data)
print(proportions)
```

```
## $prop_edge_less_0.25
## [1] 0.7493
##
## $prop_vertex_less_0.25
## [1] 0.1948
```

5.

To estimate π with a Monte Carlo simulation, we draw the unit circle inside the unit square, the ratio of the area of the circle to the area of the square will be $\pi/4$. Then shot K arrows at the square, roughly $K * \pi/4$ should have fallen inside the circle. So if now you shoot N arrows at the square, and M fall inside the circle, you have the following relationship $M = N * \pi/4$. You can thus compute π like so: $\pi = 4 * M/N$. The more arrows N you throw at the square, the better approximation of π you'll have.

```
n <- 10000

set.seed(1)
points <- tibble("x" = runif(n), "y" = runif(n))
```

Now, to know if a point is inside the unit circle, we need to check whether $x^2 + y^2 < 1$. Let's add a new column to the points tibble, called `inside` equal to `1` if the point is inside the unit circle and `0` if not:

```
points <- points |>
    mutate(inside = map2_dbl(.x = x, .y = y, ~ifelse(.x**2 + .y**2 < 1, 1, 0))) |>
    rowid_to_column("N")
```

a. Compute the estimation of π at each row, by computing the cumulative sum of the 1's in the `inside` column and dividing that by the current value of `N` column:

```
points <- points |>
  mutate(pi_est = 4 * cumsum(inside) / N)
head(points)
```

N <int>	x <dbl>	y <dbl>	inside <dbl>	pi_est <dbl>
1	0.2655087	0.06471249	1	4
2	0.3721239	0.67661240	1	4
3	0.5728534	0.73537169	1	4
4	0.9082078	0.11129967	1	4

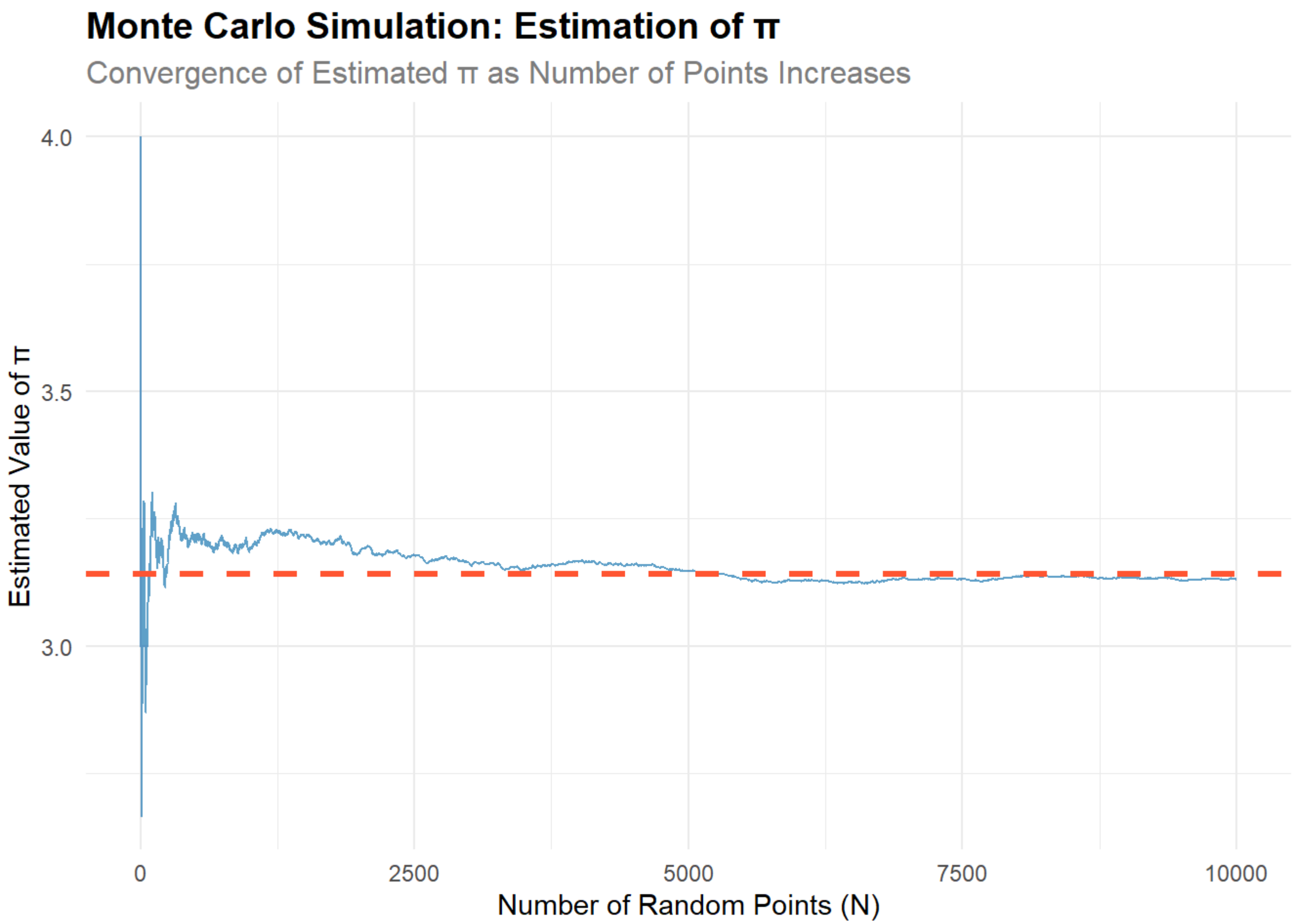
N <int>	x <dbl>	y <dbl>	inside <dbl>	pi_est <dbl>
5	0.2016819	0.04665462	1	4
6	0.8983897	0.13091031	1	4

6 rows

b. Plot the estimates of π against `N`.

```
library(ggplot2)
ggplot(points, aes(x = N, y = pi_est)) +
  geom_line(color = "#1F77B4", alpha = 0.7) +
  geom_hline(yintercept = pi, color = "#FF5733", linetype = "dashed", size = 1.2) +
  labs(
    title = "Monte Carlo Simulation: Estimation of π",
    subtitle = "Convergence of Estimated π as Number of Points Increases",
    x = "Number of Random Points (N)",
    y = "Estimated Value of π"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    plot.subtitle = element_text(size = 12, color = "gray50")
  )
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



6.

Mortality rates per 100,000 from male suicides for a number of age groups and a number of countries are given in the following data frame.

```
suicrates <- tibble(Country = c('Canada', 'Israel', 'Japan', 'Austria', 'France', 'Germany',
'Hungary', 'Italy', 'Netherlands', 'Poland', 'Spain', 'Sweden', 'Switzerland', 'UK', 'USA'),
Age25.34 = c(22,  9, 22, 29, 16, 28, 48,  7,  8, 26,  4, 28, 22, 10, 20),
Age35.44 = c(27, 19, 19, 40, 25, 35, 65,  8, 11, 29,  7, 41, 34, 13, 22),
Age45.54 = c(31, 10, 21, 52, 36, 41, 84, 11, 18, 36, 10, 46, 41, 15, 28),
Age55.64 = c(34, 14, 31, 53, 47, 49, 81, 18, 20, 32, 16, 51, 50, 17, 33),
Age65.74 = c(24, 27, 49, 69, 56, 52, 107, 27, 28, 28, 22, 35, 51, 22, 37))
```

a. Transform `suicrates` into *long* form.

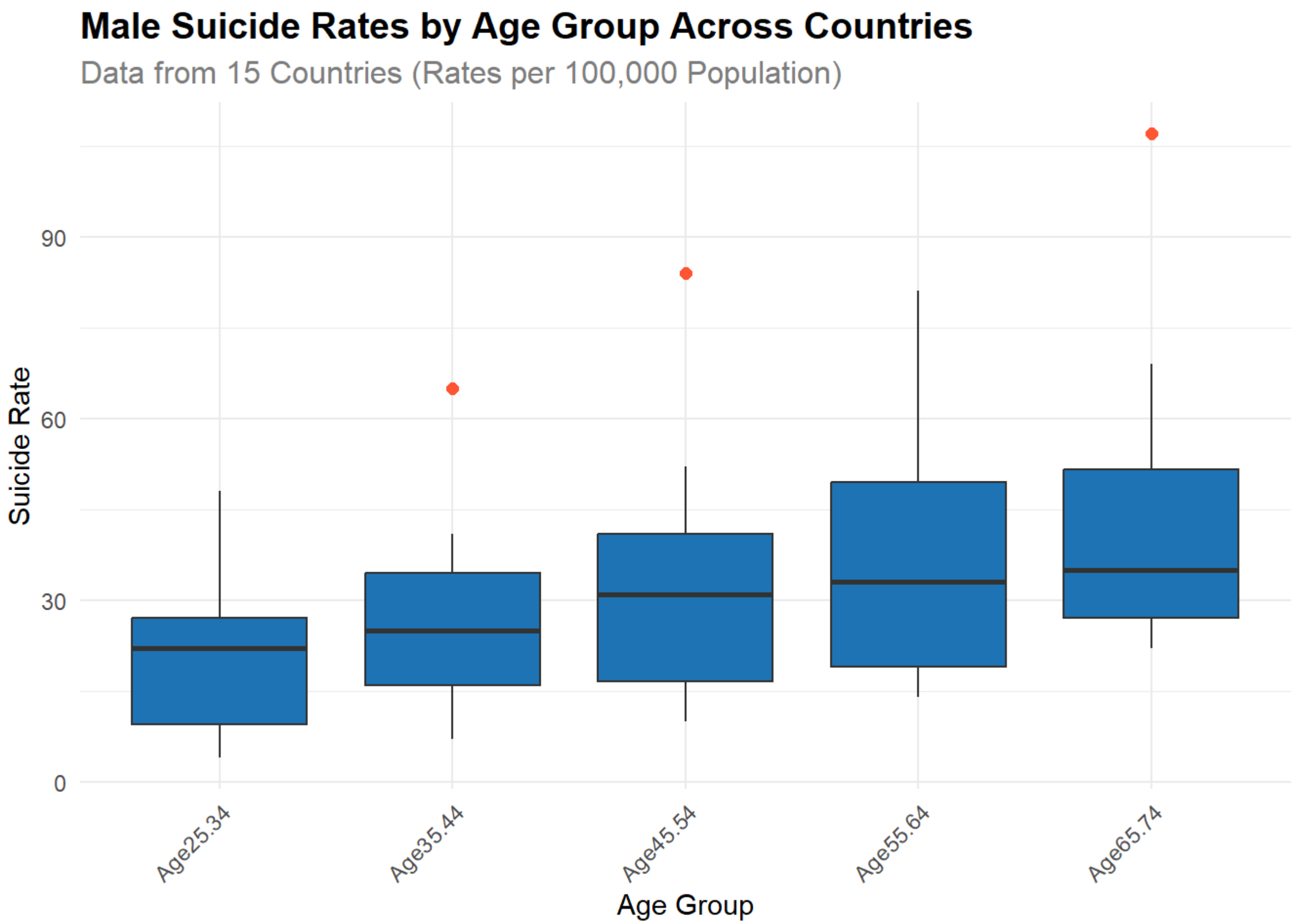
```
library(tidyverse)
suicrates_long <- suicrates %>%
  pivot_longer(
    cols = starts_with("Age"),
    names_to = "AgeGroup",
    values_to = "SuicideRate"
  )
head(suicrates_long)
```

Country <chr>	AgeGroup <chr>	SuicideRate <dbl>
Canada	Age25.34	22
Canada	Age35.44	27
Canada	Age45.54	31
Canada	Age55.64	34
Canada	Age65.74	24
Israel	Age25.34	9

6 rows

b. Construct side-by-side box plots for the data from different age groups, and comment on what the graphic tells us about the data.

```
ggplot(suicrates_long, aes(x = AgeGroup, y = SuicideRate)) +
  geom_boxplot(
    fill = "#1F77B4",
    outlier.color = "#FF5733",
    outlier.shape = 16,
    outlier.size = 2
  ) +
  labs(
    title = "Male Suicide Rates by Age Group Across Countries",
    subtitle = "Data from 15 Countries (Rates per 100,000 Population)",
    x = "Age Group",
    y = "Suicide Rate"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    plot.subtitle = element_text(size = 12, color = "gray50"),
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```



7.

Load the `LaborSupply` dataset from the `{Ecdat}` package and answer the following questions:

```
#data(LaborSupply)
LaborSupply <- read_csv("data/LaborSupply.csv")
```

```
## Rows: 5320 Columns: 7
## ── Column specification ────────────────────────────────────────────────────────
────────
## Delimiter: ","
## dbl (7): lnhr, lnwg, kids, age, disab, id, year
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# create hour and wage variables
labor <- LaborSupply |>
  mutate(hour = exp(lnhr), wage = exp(lnwg), .before = kids) |>
  dplyr::select(-lnhr, -lnwg)
```

a. Compute the average annual hours worked and their standard deviations by year.

```
hour_stats_by_year <- labor %>%
  group_by(year) %>%
  summarize(
    mean_hour = mean(hour, na.rm = TRUE),
    sd_hour = sd(hour, na.rm = TRUE)
  ) %>%
  ungroup()
print(hour_stats_by_year)
```

```
## # A tibble: 10 × 3
##     year mean_hour sd_hour
##    <dbl>     <dbl>   <dbl>
##  1  1979     2202.    502.
##  2  1980     2182.    454.
##  3  1981     2185.    460.
##  4  1982     2145.    442.
##  5  1983     2124.    550.
##  6  1984     2149.    492.
##  7  1985     2203.    515.
##  8  1986     2195.    482.
##  9  1987     2219.    529.
## 10  1988     2222.    478.
```

b. What age group worked the most hours in the year 1982?

```
labor_1982 <- labor %>%
  filter(year == 1982) %>%
  mutate(
    age_group = cut(
      age,
      breaks = c(20, 30, 40, 50, 60),
      labels = c("20-29", "30-39", "40-49", "50-59"),
      right = FALSE
    )
  )
age_group_hours <- labor_1982 %>%
  group_by(age_group) %>%
  summarize(mean_hours = mean(hour, na.rm = TRUE)) %>%
  ungroup()
max_age_group <- age_group_hours %>%
  filter(mean_hours == max(mean_hours, na.rm = TRUE)) %>%
  pull(age_group) %>%
  as.character()

cat("The age group that worked the most hours in 1982 was: ", max_age_group, "\n")
```

```
## The age group that worked the most hours in 1982 was:  30-39
```

c. Create a variable, `n_years` that equals the number of years an individual stays in the panel. Is the panel balanced?

```
labor <- labor %>%
  group_by(id) %>%
  mutate(n_years = n()) %>%
  ungroup()
is_balanced <- length(unique(labor$n_years)) == 1
cat("Is the panel balanced?", ifelse(is_balanced, "Yes", "No"), "\n")
```

```
## Is the panel balanced? Yes
```

d. Which are the individuals that do not have any kids during the whole period? Create a variable, `no_kids` , that flags these individuals (1 = no kids, 0 = kids)

```
labor <- labor %>%
  group_by(id) %>%
  mutate(
    no_kids = as.integer(all(kids == 0))
  ) %>%
  ungroup()
head(labor %>% select(id, year, kids, no_kids))
```

id <dbl>	year <dbl>	kids <dbl>	no_kids <int>
1	1979	2	0
1	1980	2	0
1	1981	2	0
1	1982	2	0
1	1983	2	0

id	year	kids	no_kids
<dbl>	<dbl>	<dbl>	<int>
1	1984	2	0

6 rows

e. Using the `no_kids` variable from before compute the average wage, standard deviation and number of observations in each group for the year 1980 (no kids group vs kids group).

```
labor_1980 <- labor %>%
  filter(year == 1980)

wage_stats_1980 <- labor_1980 %>%
  group_by(no_kids) %>%
  summarize(
    mean_wage = mean(wage, na.rm = TRUE),
    sd_wage = sd(wage, na.rm = TRUE),
    n_obs = n()
  ) %>%
  ungroup()

wage_stats_1980 <- wage_stats_1980 %>%
  mutate(
    group = case_when(
      no_kids == 1 ~ "no kids group",
      no_kids == 0 ~ "kids group"
    )
  ) %>%
  select(group, mean_wage, sd_wage, n_obs)

print(wage_stats_1980)
```

```
## # A tibble: 2 × 4
##   group         mean_wage sd_wage n_obs
##   <chr>             <dbl>   <dbl> <int>
## 1 kids group         14.5    6.69   489
## 2 no kids group      15.9    6.71    43
```