Mathematical practice final exam 2025

2025-07-02

1.

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

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

```
# 创建矩阵 A
# byrow = FALSE 表示按列填充数据
A <- matrix(c(9, 5, 2, 9, 4, 0, 6, 2, 12, 7, 8, 9, 2, 9, 0, 11),
            nrow = 4, byrow = FALSE)

# 使用 solve() 函数计算矩阵 A 的逆矩阵
A_inv <- solve(A)

# 打印原始矩阵和其逆矩阵
print("Matrix A:")
```

```
## [1] "Matrix A:"
```

```
kable(A)
```

9	4	12	2
5	0	7	9
2	6	8	0

9	2	9	11

```
print("Inverse of Matrix A:")
```

```
## [1] "Inverse of Matrix A:"
```

```
kable(A_inv)
```

0.0857143	-0.26	-0.1228571	0.1971429
-0.1428571	-0.30	0.1714286	0.2714286
0.0857143	0.29	0.0271429	-0.2528571
-0.1142857	0.03	0.0471429	0.0871429

```
# 验证：A 乘以其逆矩阵应该得到一个单位矩阵
# 创建一个 4x4 的单位矩阵用于比较
I <- diag(4)

# 使用 all.equal() 检查 A %*% A_inv 是否约等于单位矩阵 I
# 这个函数可以处理浮点数计算带来的微小误差
verification <- all.equal(A %*% A_inv, I)

# 打印验证结果
print(paste("Verification result (TRUE means success):", verification))
```

```
## [1] "Verification result (TRUE means success): 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)
n <- length(xVec)
```

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

```
# yVec 从第二个元素取到最后一个
# xVec 从第一个元素取到倒数第二个
result_a <- yVec[2:n] - xVec[1:(n-1)]
print("First 6 elements of the resulting vector:")
```

```
## [1] "First 6 elements of the resulting vector:"
```

```
head(result_a)
```

```
## [1]  156  566  526 -394 -277  590
```

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

```
# 使用逻辑索引来筛选 yVec 中大于 600 的元素
result_b <- yVec[yVec > 600]
print("Values in yVec > 600 (first 6):")
```

```
## [1] "Values in yVec > 600 (first 6):"
```

```
head(result_b)
```

```
## [1] 783 757 969 687 830 965
```

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

```
# which() 函数返回满足条件的元素的索引
result_c <- which(yVec > 600)
print("Index positions of values > 600 in yVec (first 6):")
```

```
## [1] "Index positions of values > 600 in yVec (first 6):"
```

```
head(result_c)
```

```
## [1]  1  3  4  7  9 15
```

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

```
# order(yVec) 返回 yVec 按升序排列的原始索引
# 然后用这个索引顺序来排列 xVec
result_d <- xVec[order(yVec)]
print("xVec sorted by yVec (first 6):")
```

```
## [1] "xVec sorted by yVec (first 6):"
```

```
head(result_d)
```

```
## [1] 119 813 979 283  44 648
```

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

```
# seq() 生成一个从 1 开始，步长为 3 的序列
indices <- seq(from = 1, to = n, by = 3)
result_e <- yVec[indices]
print("Elements from yVec at positions 1, 4, 7, ... (first 6):")
```

```
## [1] "Elements from yVec at positions 1, 4, 7, ... (first 6):"
```

```
head(result_e)
```

```
## [1] 783 969 687  91  33 204
```

3.

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

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

state_data <- state.x77
# 为了方便后续操作，将列名中的空格替换为下划线
colnames(state_data) <- make.names(colnames(state_data))
```

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

```
# 使用 filter() 筛选收入低于 4300 的州
low_income_states <- state_data %>% filter(Income < 4300)

# 使用 summarise() 计算平均收入
avg_income <- low_income_states %>% summarise(AverageIncome = mean(Income))

print("Average income for states with income < 4300:")
```

```
## [1] "Average income for states with income < 4300:"
```

```
kable(avg_income)
```

AverageIncome
3830.6

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

```
# arrange(desc(Income)) 按收入降序排列
# slice(1) 选取第一行，即收入最高的州
highest_income_state <- state_data %>%
  arrange(desc(Income)) %>%
  slice(1)

print("State with the highest income:")
```

```
## [1] "State with the highest income:"
```

```
kable(highest_income_state)
```

State	Population	Income	Illiteracy	Life.Exp	Murder	HS.Grad	Frost	Area
Alaska	365	6315	1.5	69.31	11.3	66.7	152	566432

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

```r
# 使用 mutate() 和 ifelse() 添加新列 Population.Size
state_data <- state_data %>%
  mutate(
    Population.Size = ifelse(Population <= 4500, "S", "L")
  )

print("Data with new Population.Size column (first 6 rows):")
```

```
## [1] "Data with new Population.Size column (first 6 rows):"
```

```r
kable(head(state_data))
```

State	Population	Income	Illiteracy	Life.Exp	Murder	HS.Grad	Frost	Area	Population.Size
Alabama	3615	3624	2.1	69.05	15.1	41.3	20	50708	S
Alaska	365	6315	1.5	69.31	11.3	66.7	152	566432	S
Arizona	2212	4530	1.8	70.55	7.8	58.1	15	113417	S
Arkansas	2110	3378	1.9	70.66	10.1	39.9	65	51945	S
California	21198	5114	1.1	71.71	10.3	62.6	20	156361	L
Colorado	2541	4884	0.7	72.06	6.8	63.9	166	103766	S

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

```r
# group_by() 按人口规模分组
# summarise() 分别计算每组的平均收入和平均文盲率
summary_by_size <- state_data %>%
  group_by(Population.Size) %>%
  summarise(
    Average.Income = mean(Income),
    Average.Illiteracy = mean(Illiteracy)
  )

print("Summary by population size:")
```

```
## [1] "Summary by population size:"
```

```
kable(summary_by_size)
```

Population.Size	Average.Income	Average.Illiteracy
L	4607.938	1.200000
S	4354.794	1.155882

4.

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

```
# 定义函数，输入参数为 n（观测值数量）
simulate_points <- function(n) {
  # 使用 runif(n) 生成 n 个 [0,1] 区间的随机数
  tibble(
    X1 = runif(n),
    X2 = runif(n)
  )
}

# 调用函数生成 10 个点作为示例
example_points <- simulate_points(10)
print("Example of 10 simulated points:")
```

```
## [1] "Example of 10 simulated points:"
```

```
kable(example_points)
```

X1	X2
0.7609526	0.1733402
0.4610862	0.1893845
0.8561333	0.7899421
0.1767227	0.1771477

X1	X2
0.5002223	0.4423429
0.5060098	0.2904962
0.2486763	0.7895282
0.8206788	0.7888322
0.0464717	0.8846973
0.4279461	0.1149546

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.

```r
# 定义函数，输入为包含 X1, X2 坐标的数据框
calculate_proportions <- function(points_df) {
  n_obs <- nrow(points_df)

  # 计算每个点到最近边缘的距离
  # pmin() 返回每个位置上的最小值
  dist_edge <- pmin(points_df$X1, 1 - points_df$X1,
                    points_df$X2, 1 - points_df$X2)

  # 计算每个点到最近顶点的距离
  # 四个顶点为 (0,0), (1,0), (0,1), (1,1)
  dist_v1 <- sqrt(points_df$X1^2 + points_df$X2^2)
  dist_v2 <- sqrt((1 - points_df$X1)^2 + points_df$X2^2)
  dist_v3 <- sqrt(points_df$X1^2 + (1 - points_df$X2)^2)
  dist_v4 <- sqrt((1 - points_df$X1)^2 + (1 - points_df$X2)^2)
  dist_vertex <- pmin(dist_v1, dist_v2, dist_v3, dist_v4)

  # 计算满足条件的比例
  prop_edge <- sum(dist_edge < 0.25) / n_obs
  prop_vertex <- sum(dist_vertex < 0.25) / n_obs
```

```
  # 返回一个列表包含两个比例
  list(
    prop_near_edge = prop_edge,
    prop_near_vertex = prop_vertex
  )
}

# 使用 10000 个模拟点进行计算
many_points <- simulate_points(10000)
proportions <- calculate_proportions(many_points)
print(proportions)
```

```
## $prop_near_edge
## [1] 0.7497
##
## $prop_near_vertex
## [1] 0.1917
```

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:

```
# cumsum(inside) 计算落在圆内的点的累积数量 M
# pi_est = 4 * M / N
pi_estimates <- points %>%
  mutate(
    M = cumsum(inside),
    pi_est = 4 * M / N
  )

print("Pi estimates at each step (first 10):")
```

```
## [1] "Pi estimates at each step (first 10):"
```

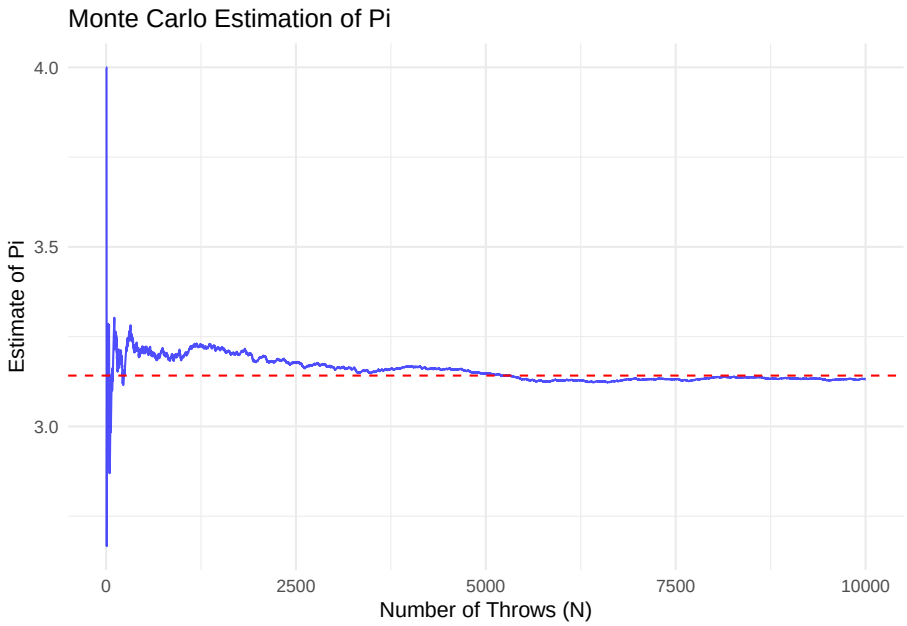
```
kable(head(pi_estimates, 10))
```

N	x	y	inside	M	pi_est
1	0.2655087	0.0647125	1	1	4.000000
2	0.3721239	0.6766124	1	2	4.000000
3	0.5728534	0.7353717	1	3	4.000000
4	0.9082078	0.1112997	1	4	4.000000
5	0.2016819	0.0466546	1	5	4.000000
6	0.8983897	0.1309103	1	6	4.000000
7	0.9446753	0.8809564	0	6	3.428571
8	0.6607978	0.8397255	0	6	3.000000
9	0.6291140	0.8681427	0	6	2.666667

N	x	y	inside	M	pi_est
10	0.0617863	0.0333829	1	7	2.800000

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

```
ggplot(pi_estimates, aes(x = N, y = pi_est)) +
  geom_line(color = "blue", alpha = 0.7) +
  # 添加一条水平红线表示 pi 的真实值
  geom_hline(yintercept = pi, color = "red", linetype = "dashed") +
  # 设置图表标题和坐标轴标签 (英文)
  labs(
    title = "Monte Carlo Estimation of Pi",
    x = "Number of Throws (N)",
    y = "Estimate of Pi"
  ) +
  theme_minimal()
```



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', 'Germ
'Hungary', 'Italy', 'Netherlands', 'Poland', 'Spain', 'Sweden', 'Switzerland', 'UK', 'U
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.

```
# 使用 pivot_longer 将宽数据转换为长数据
suicrates_long <- suicrates %>%
  pivot_longer(
    cols = -Country,           # 转换除 Country 外的所有列
    names_to = "AgeGroup",     # 新的列名，存储原列名
    values_to = "Rate"         # 新的列名，存储原单元格值
  )

print("Transformed data in long format (first 6 rows):")
```

```
## [1] "Transformed data in long format (first 6 rows):"
```

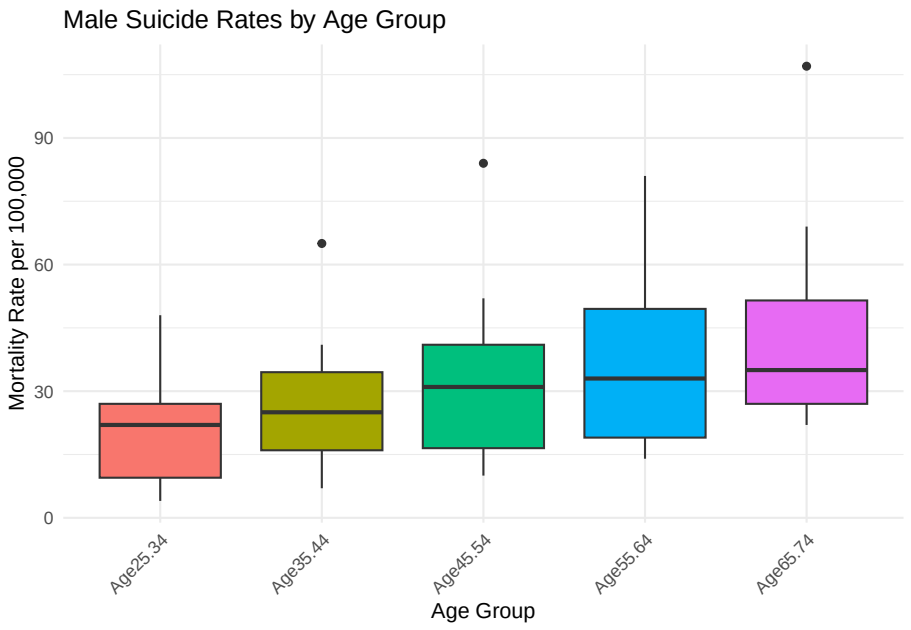
```
kable(head(suicrates_long))
```

Country	AgeGroup	Rate
Canada	Age25.34	22
Canada	Age35.44	27
Canada	Age45.54	31
Canada	Age55.64	34
Canada	Age65.74	24

Country	AgeGroup	Rate
Israel	Age25.34	9

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 = Rate, fill = AgeGroup)) +
  geom_boxplot() +
  # 设置图表标题和坐标轴标签 (英文)
  labs(
    title = "Male Suicide Rates by Age Group",
    x = "Age Group",
    y = "Mortality Rate per 100,000"
  ) +
  theme_minimal() +
  # 旋转 x 轴标签以防重叠
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = "none") # 隐藏图例
```



评论: 从箱线图中可以看出，随着年龄的增长，男性自杀率的中位数和分布范围（即四分位距）整体上呈现上升趋势。Age65-74 年龄组的自杀率最高，且有一个非常高的异常值，表明在某些国家该年龄段的自杀问题尤为严重。具体来说，该箱线图揭示了男性自杀率与年龄的显著关系。主要观察如下：整体趋势：随着年龄组的增长，自杀率的中位数呈现出明显的上升趋势，表明年龄较大的群体面临更高的自杀风险。数据离散度：年龄越大，箱体（即四分位距）也越宽，这说明不同国家在高年龄组的自杀率差异更大，数据变异性增加。关键异常值：Age65-74 年龄组不仅拥有最高的中位数，还包含一个极高的异常值，这说明在至少一个国家，该年龄段的自杀率远超其他国家，是一个需要特别关注的公共卫生问题。

7.

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

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

```
# 从提供的 "LaborSupply.csv" 文件加载数据
LaborSupply <- read_csv("LaborSupply.csv")

# 创建 hour 和 wage 变量
labor <- LaborSupply %>%
  mutate(hour = exp(lnhr), wage = exp(lnwg), .before = kids) %>%
  dplyr::select(-lnhr, -lnwg)
```

```r
# 按年份分组，然后计算每年的平均工时和标准差
summary_by_year <- labor %>%
  group_by(year) %>%
  summarise(
    Average.Hours = mean(hour),
    Std.Dev.Hours = sd(hour)
  )

print("Average annual hours worked by year:")
```

```
## [1] "Average annual hours worked by year:"
```

```r
kable(summary_by_year)
```

year	Average.Hours	Std.Dev.Hours
1979	2201.990	502.0461
1980	2182.227	453.9481
1981	2184.521	460.4205
1982	2144.842	441.8772
1983	2123.619	549.8696
1984	2148.749	491.9653
1985	2202.686	514.9282
1986	2194.610	482.2137
1987	2219.321	528.5617
1988	2222.148	477.5253

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

```r
# 筛选 1982 年的数据，并使用 cut() 将年龄分组成段
# 然后按年龄组计算平均工时，并排序找到最高值
most_hours_group <- labor %>%
  filter(year == 1982) %>%
  mutate(
    Age.Group = cut(age, breaks = c(20, 30, 40, 50, 60),
```

```
                    right = FALSE)
  ) %>%
  group_by(Age.Group) %>%
  summarise(Average.Hours = mean(hour)) %>%
  arrange(desc(Average.Hours))

print("Average hours by age group in 1982:")
```

```
## [1] "Average hours by age group in 1982:"
```

```
kable(most_hours_group)
```

Age.Group	Average.Hours
[30,40)	2156.905
[50,60)	2156.660
[40,50)	2134.763
[20,30)	2118.697

```
print(paste("The age group with the most hours is:",
            most_hours_group$Age.Group[1]))
```

```
## [1] "The age group with the most hours is: [30,40)"
```

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

```
# 按个体 id 分组，使用 n() 计算每个 id 出现的次数
labor_with_nyears <- labor %>%
  group_by(id) %>%
  mutate(n_years = n()) %>%
  ungroup()

# 检查所有 n_years 是否都等于最大值
# 如果是，则面板数据是平衡的
is_balanced <- all(labor_with_nyears$n_years == max(labor_with_nyears$n_years))
```

```
print(paste("Is the panel balanced?", is_balanced))
```

```
## [1] "Is the panel balanced? TRUE"
```

```
print("Distribution of n_years:")
```

```
## [1] "Distribution of n_years:"
```

```
kable(table(labor_with_nyears$n_years))
```

Var1	Freq
10	5320

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)

```
# 按 id 分组，如果一个 id 的所有 kids 加起来等于 0，则标记为 1
labor_with_nokids <- labor %>%
  group_by(id) %>%
  mutate(no_kids = ifelse(sum(kids) == 0, 1, 0)) %>%
  ungroup()

print("Individuals with no kids (first 6 rows):")
```

```
## [1] "Individuals with no kids (first 6 rows):"
```

```
kable(head(filter(labor_with_nokids, no_kids == 1)))
```

hour	wage	kids	age	disab	id	year	no_kids
1685.808	10.91349	0	26	1	3	1979	1
1408.105	10.80490	0	27	0	3	1980	1
2038.562	13.19714	0	28	0	3	1981	1
1978.314	13.46374	0	29	0	3	1982	1
2059.050	12.80710	0	30	0	3	1983	1

hour	wage	kids	age	disab	id	year	no_kids
1556.197	13.06582	0	31	0	3	1984	1

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).

```
# 筛选 1980 年的数据，按 no_kids 分组
# 计算每组的平均工资、标准差和观测数 (n())
summary_1980 <- labor_with_nokids %>%
  filter(year == 1980) %>%
  group_by(no_kids) %>%
  summarise(
    Average.Wage = mean(wage),
    Std.Dev.Wage = sd(wage),
    Observations = n()
  )

print("Summary of wage in 1980 by kids status:")
```

```
## [1] "Summary of wage in 1980 by kids status:"
```

```
kable(summary_1980)
```

no_kids	Average.Wage	Std.Dev.Wage	Observations
0	14.49142	6.685356	489
1	15.89679	6.707897	43