Question 1:

Initial conditions:

$$f(x) = \frac{1}{2}x^T Ax + b^T x,$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and $b \in \mathbb{R}^n$

To find:

$\nabla_x f(x)$ and $\nabla^2 f(x)$

Solution:

1) At the beginning, need to find the gradient:

Take into account that A is symmetric, that is, $A = A^T$.

$$\nabla_x(x^T Ax) = (A + A^T)x$$

So:

$$\nabla_x\left(\frac{1}{2}x^T Ax\right) = \frac{1}{2}\nabla_x(x^T Ax) = \frac{1}{2}(A + A)x = Ax$$

$$\nabla_x(b^T x) = b.$$

So, at the finally:

$$\nabla_x f(x) = Ax + b$$

2) At the beginning, need to find the Hessian:

Need to differentiate the gradient $\nabla_x f(x) = Ax + b$ with respect to x, because the Hessian is the matrix of second derivatives.

So,

$$\nabla^2(Ax) = A$$

$$\nabla^2 b = 0$$

b - const, and A - coefficient, constant before x

Solution:

$$\nabla_x f(x) = Ax + b$$

$$\nabla^2 f(x) = A$$

b)

Initial conditions:

$$f(x) = g(h(x)),$$

where $g : \mathbb{R} \to \mathbb{R}$ and $h : \mathbb{R}^n \to \mathbb{R}$ are both differentiable.

To find:

$$\nabla_x f(x)$$

Solution:

$$\nabla_x f(x) = \nabla_x g(h(x)) = g'(h(x)) \cdot \nabla_x h(x)$$

-because the chain rule applies: the derivative of the outer function with respect to the inner function is multiplied by the derivative of the inner function with respect to the variable of interest

$g'(h(x))$ - scalar value

Gradient $\nabla_x h(x)$is$n$ - vector of partial derivatives

$$\nabla_x h(x) = \begin{bmatrix} \frac{\partial h}{\partial x_1}(x) \\ \frac{\partial h}{\partial x_2}(x) \\ \vdots \\ \frac{\partial h}{\partial x_n}(x) \end{bmatrix}.$$

So, at the finally:

$$\nabla_x f(x) = g'(h(x)) \cdot \nabla_x h(x).$$

c)

Initial conditions:

$$f(x) = g(a^T x)$$

where $a \in \mathbb{R}^n$ and $g : \mathbb{R} \to \mathbb{R}$

To find:

$\nabla_x f(x)$ and $\nabla^2 f(x)$

Solution:

1) the chain rule applies:

$$\nabla_x (a^T x) = a.$$
$$\nabla_x f(x) = g'(a^T x) \cdot \nabla_x (a^T x).$$

At the finally:

$$\nabla_x f(x) = g'(a^T x) \cdot a.$$

2)

$$\nabla_x f(x) = g'(a^T x) \cdot a.$$
$$\nabla^2 f(x) = \nabla_x (g'(a^T x) \cdot a).$$

At the finally:

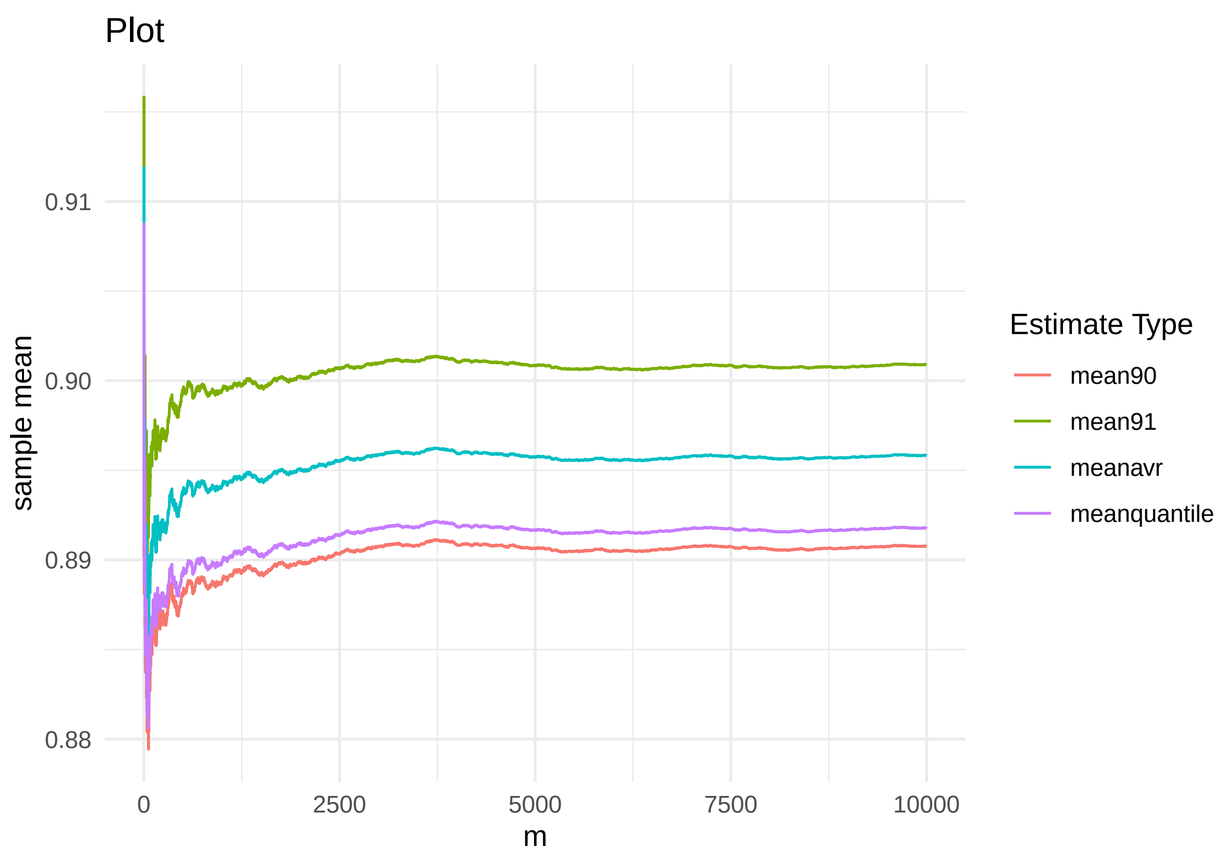
$$\nabla^2 f(x) = g''(a^T x) \cdot \nabla_x (a^T x) \cdot a^T = g''(a^T x) \cdot (aa^T),$$

Question 2:

```r
#a
estimatepercentile90 <- function(x) {
  if (length(x) != 100) {
    stop("Vector must be of length 100")
  }
  sortedx <- sort(x)
  return(sortedx[90])
}

# b
estimatepercentile91 <- function(x) {
  if (length(x) != 100) {
    stop("Vector must be of length 100")
  }
  sortedx <- sort(x)
  return(sortedx[91])
}

avr90and91 <- function(x) {
  return((estimatepercentile90(x) + estimatepercentile91(x)) / 2)
}

# d
n <- 10000
estimatespercentile90 <- numeric(n)
estimatespercentile91 <- numeric(n)
estimatesavr <- numeric(n)
estimatesquantile <- numeric(n)

for (i in 1:n) {
  sample <- runif(100)

  estimatespercentile90[i] <- estimatepercentile90(sample)

  estimatespercentile91[i] <- estimatepercentile91(sample)

  estimatesavr[i] <- avr90and91(sample)

  estimatesquantile[i] <- quantile(sample, 0.9)
}
```

```r
library(ggplot2)

results <- data.frame(
  m = 1:n,
  mean90 = cumsum(estimatespercentile90) / (1:n),
  mean91 = cumsum(estimatespercentile91) / (1:n),
  meanavr = cumsum(estimatesavr) / (1:n),
  meanquantile = cumsum(estimatesquantile) / (1:n)
)
```

```r
library(reshape2)
results_melted <- melt(results, id.vars = "m")

ggplot(results_melted, aes(x = m, y = value, color = variable)) +
  geom_line() +
  labs(title = "Plot",
       x = "m",
       y = "sample mean",
       color = "Estimate Type") +
  theme_minimal()
```



e)

```r
ultimateMeans <- data.frame(
  Method = c("90 Percentile", "91 Percentile", "Average of 90 & 91", "Quantile Function"),
  ultimate_means = c(mean(estimatespercentile90), mean(estimatespercentile91), mean(estimatesavr), mean
)

print(ultimateMeans)
```

```
##               Method ultimate_means
## 1      90 Percentile      0.8907533
## 2      91 Percentile      0.9008927
## 3 Average of 90 & 91      0.8958230
## 4  Quantile Function      0.8917672
```

The estimate obtained from the quantile() function appears to be the best, because it uses a more complex method, which give a more exactly estimate compared to the simpler methods, based on sorting.

Question 3:

a)

pi2(n) that approximates as a function of n, using the approximation:

$$\pi = \lim_{n\to\infty} \sqrt{6\sum_{k=1}^{n}\frac{1}{k^2}}$$

```
pi2 <- function(n) {
  sumK <- sum(1/(1:n)^2)

  return(sqrt(6 * sumK))
}
```

pi2 for j = 0, 1, 2, ..., 6:

```
solution_pi2 <- sapply(0:6, function(j) pi2(10^j))
names(solution_pi2) <- paste("10^", 0:6, sep="")
solution_pi2
```

```
##     10^0     10^1     10^2     10^3     10^4     10^5     10^6
## 2.449490 3.049362 3.132077 3.140638 3.141497 3.141583 3.141592
```

b)

```
pi3 <- function(n) {
  x <- runif(n, -1, 1)
  y <- runif(n, -1, 1)
  circle <- (x^2 + y^2) <= 1
  fraction <- sum(circle) / n
  return(fraction * 4)
}
```

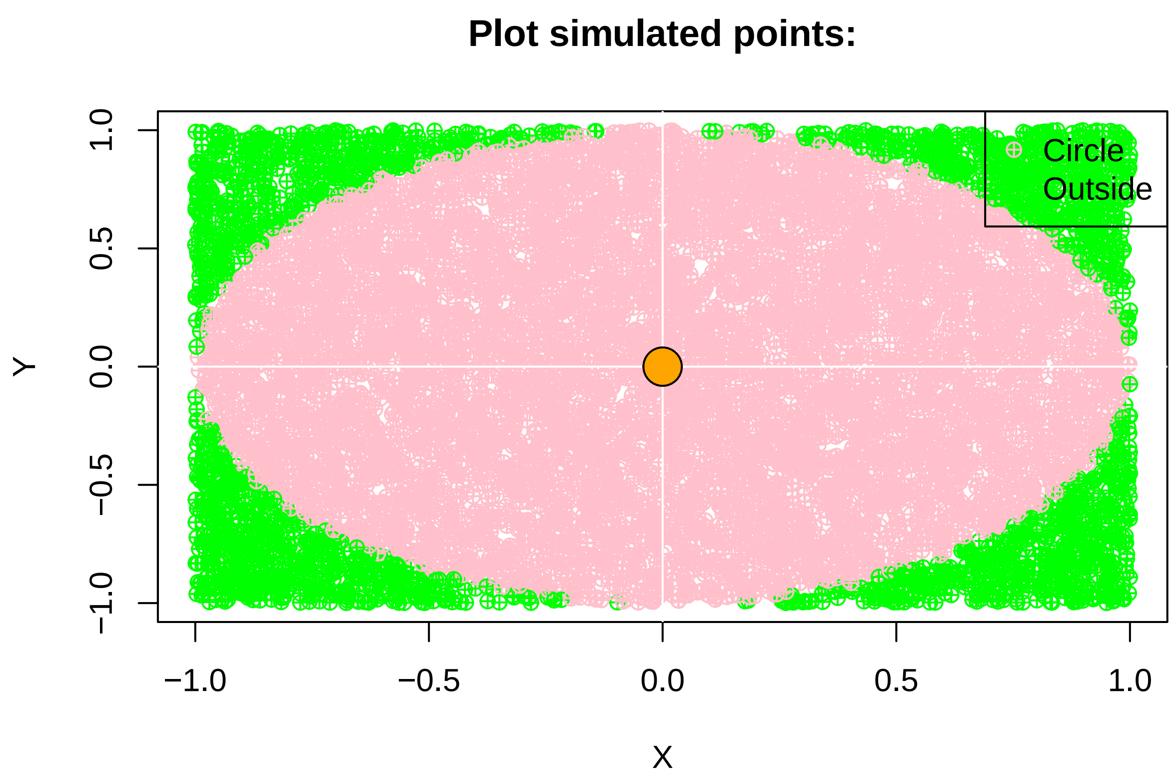
```
solution_pi3 <- sapply(0:4, function(j) pi3(10^j))
names(solution_pi3) <- paste("10^", 0:4, sep="")
solution_pi3
```

```
##   10^0   10^1   10^2   10^3   10^4
## 0.0000 3.2000 2.9600 3.0920 3.1284
```

c) For j=4:

```
points_col <- 10^4
x <- runif(points_col, -1, 1)
y <- runif(points_col, -1, 1)
circle <- (x^2 + y^2) <= 1
```

```r
plot(x, y, col = ifelse(circle, "pink", "green"), pch = ifelse(circle, 10, 10),
     xlim = c(-1, 1), ylim = c(-1, 1), main = "Plot simulated points:",
     xlab = "X", ylab = "Y")
abline(h = 0, v = 0, col = "white")
symbols(0, 0, circles = 1, inches = 0.1, add = TRUE, bg = "orange", fg = "black")
legend("topright", legend = c("Circle", "Outside"), col = c("pink", "green"), pch = c(10, 10))
```



Question 4:

a)

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
cat("\n")
```

```r
books <- read.csv("amazonbooks.csv")
book_with_min_pages <- books[which.min(books$NumPages), ]
name_of_min_book <- book_with_min_pages$Title
col_pages_of_min_book <- book_with_min_pages$NumPages
cat("The book with the minimum number of pages: ", name_of_min_book," ", col_pages_of_min_book,"pages\n
```

```
## The book with the minimum number of pages:  Big Dog . . . Little Dog   24 pages
```

b)

```r
library(dplyr)
summary_table <- books %>%
    group_by(Author) %>%
    summarise(
        all_books = n(),
        mean_pages = mean(NumPages),
        variance_pages = var(NumPages)
    )
print(summary_table)
```

```
## # A tibble: 256 x 4
##    Author              all_books mean_pages variance_pages
##    <chr>                   <int>      <dbl>          <dbl>
##  1 ""                          1        432             NA
##  2 "Abraham Verghese"          1        667             NA
##  3 "Adam Goodheart"            1        460             NA
##  4 "Adam Hochschild"           1        480             NA
##  5 "Adam Mansbach"             1         32             NA
##  6 "Alaa Aswany"               1        255             NA
##  7 "Alice Munro"               2        320           2048
##  8 "Alice Schroeder"           1        832             NA
##  9 "Allen, Toorawa"            1        200             NA
## 10 "Andrea Warren"             1        160             NA
## # i 246 more rows
```

c)

```r
library(ggplot2)
ggplot(summary_table, aes(x = Author, y = mean_pages, fill = Author)) +
  geom_bar(stat = "identity") +
  labs(title = "Average number of pages by author",
       x = "Author",
       y = "Average number of pages") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
```

Hurston
J.M. Barrie
Jamaica Kincaid
James B. Maas
James Napoli
James Patterson
James Patterson & Maxine Paetro
James Patterson & Michael Ledwidge
Jane Austin
Jay Asher
JD Salinger
Jean−Dominique Bauby
Jean−Paul Sartre
Jeffrey Toobin
Jerome Lawrence and Robert E. Lee
Jim Glastonbury
Jimmy Fallon
John Kennedy Toole
John Steinbeck
John Tosh
John Traphagan
Joseph Conrad
Joseph Heller
Joseph M. Williams
Jospeh Heller
Judi Barrett
Judith M. Heimann
Justin Halpern
Kate Chopin
Kate Morton
Kathleen Norris
Kathy Griffin
Kawabata
Kawasaki
Kurt Vonnegut
Lahiri
Lalita Tademy
Laura Hillenbrand
Lauren Leto
Lawrence
Levy
Lewis
Lewis Carroll
Lois Lowry
Lore, Pittacus
Lorraine Hansberry
Louis Sachar
Ludwig Bemelmans
M. Jimmie Killingsw
Marcus Aurelius
Mark Hart