98

# Survival Analyis HW2

313657003 周佳萱

2025-03-31

## Problem 1

**(a) Record the truncated and censoring probability based on 1000 simulation data for both scenarios. (Attached Code)**

```
library(knitr)

set.seed(123)
m <- 1000
n <- 10000
theta <- 0.1
beta1_list <- c(5, 10)
beta2 <- 5

trunc_prob <- numeric(length(beta1_list))
censor_prob <- numeric(length(beta1_list))

for (i in seq_along(beta1_list)) {
  beta1 <- beta1_list[i]
  trunc_total <- 0
  censor_total <- 0

  for (j in 1:m) {
    T <- rexp(n, theta)
    W <- runif(n, 0, beta1)
    V <- runif(n, 0, beta2)
    C <- W + V

    trunc_total <- trunc_total + sum(T < W)
    censor_total <- censor_total + sum(T > C & T >= W)
  }

  trunc_prob[i] <- trunc_total / (n * m)
  censor_prob[i] <- censor_total / (n * m)
}


prob_table <- data.frame(
  Scenario = c("Scenario 1 (beta1 = 5)", "Scenario 2 (beta1 = 10)"),
  Truncation_Probability = trunc_prob,
```

W should not be involved when calculating censoring prob. (-2)

```
  Censoring_Probability = censor_prob
)


kable(prob_table,
      caption = "Truncation and Censoring Probabilities (based on 1000 simulations × 10000 samples)",
      align = "c")
```

Table 1: Truncation and Censoring Probabilities (based on 1000 simulations × 10000 samples)

| Scenario | Truncation_Probability | Censoring_Probability |
|---|---|---|
| Scenario 1 (beta1 = 5) | 0.2130613 | 0.6191647 |
| Scenario 2 (beta1 = 10) | 0.3679662 | 0.4973954 |

**(b) Based on simulation data in (a), using Kaplan-Meier (only right-censoring version), plot true and the average of estimated survival curve and 95% condence interval of estimated survival curve for both scenarios. (Please write down your own codes and compare to the result by using package)(Attached Code)**

```
library(survival)
library(progress)

sim_censor <- function(n, beta1, beta2, theta) {
  T <- rexp(n, theta)
  W <- runif(n, 0, beta1)
  V <- runif(n, 0, beta2)
  C <- W + V

  idx <- T >= W
  T <- T[idx]
  W <- W[idx]
  C <- C[idx]

  status <- as.numeric(T <= C)
  T_obs <- pmin(T, C)

  df <- data.frame(T = T, W = W, C = C, T_obs = T_obs, status = status)
  df <- df[order(df$T_obs), ]

  # ------------KM no package------------------
  time_points <- unique(df$T_obs[df$status == 1])
  d <- numeric(length(time_points))
  n_risk <- numeric(length(time_points))

  for (i in seq_along(time_points)) {
    t <- time_points[i]
    d[i] <- sum(df$T_obs == t & df$status == 1)
    n_risk[i] <- sum(df$T_obs >= t)
  }
```

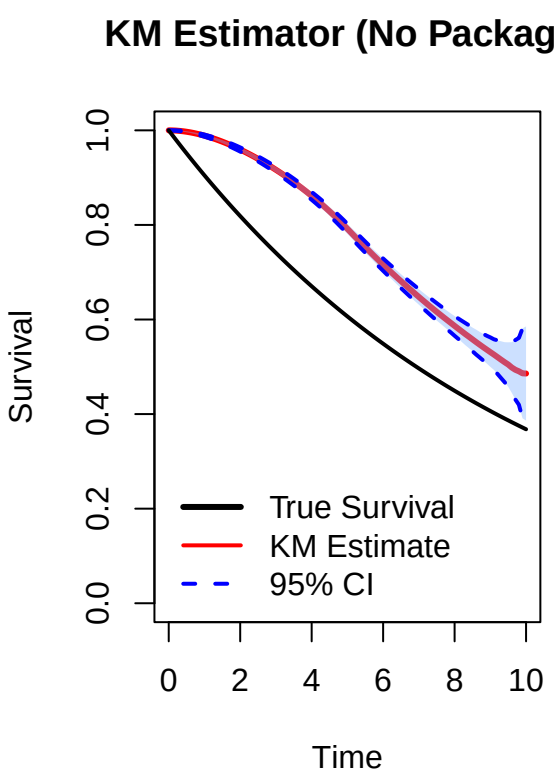
```r
  surv_est <- cumprod(1 - d / n_risk)


  t_grid <- seq(0, beta1 + beta2, by = 0.1)

  breaks_manual <- c(0, time_points, Inf)
  surv_est_full <- c(1, surv_est, tail(surv_est, 1))  # S(0)=1, extend last value
  group_manual <- cut(t_grid, breaks = breaks_manual, labels = FALSE, right = TRUE)
  sr_no_pkg <- surv_est_full[group_manual]

  # ------------KM package------------------
  fit <- survfit(Surv(T_obs, status) ~ 1)
  breaks_pkg <- c(0, fit$time, Inf)
  surv_pkg_full <- c(1, fit$surv, tail(fit$surv, 1))
  group_pkg <- cut(t_grid, breaks = breaks_pkg, labels = FALSE, right = TRUE)
  sr_pkg <- surv_pkg_full[group_pkg]

  #  S(0) = 1
  sr_no_pkg[1] <- 1
  sr_pkg[1] <- 1

  return(data.frame(
    time = t_grid,
    surv_no_pkg = sr_no_pkg,
    surv_pkg = sr_pkg
  ))
}

# ---- Simulation ----
m <- 1000
n <- 10000
theta <- 0.1
beta1 <- 5
beta2 <- 5
set.seed(12345)

#pb <- progress_bar$new(
#  format = " Simulating [:bar] :percent eta: :eta",
#  total = m, clear = FALSE, width = 60
#)

#KMb1_sim <- vector("list", m)
#for (i in 1:m) {
#  KMb1_sim[[i]] <- sim_censor(n, beta1, beta2, theta)
#  pb$tick()
#}

#saveRDS(KMb1_sim, file = "KMb1_sim_beta1_5.rds")
```

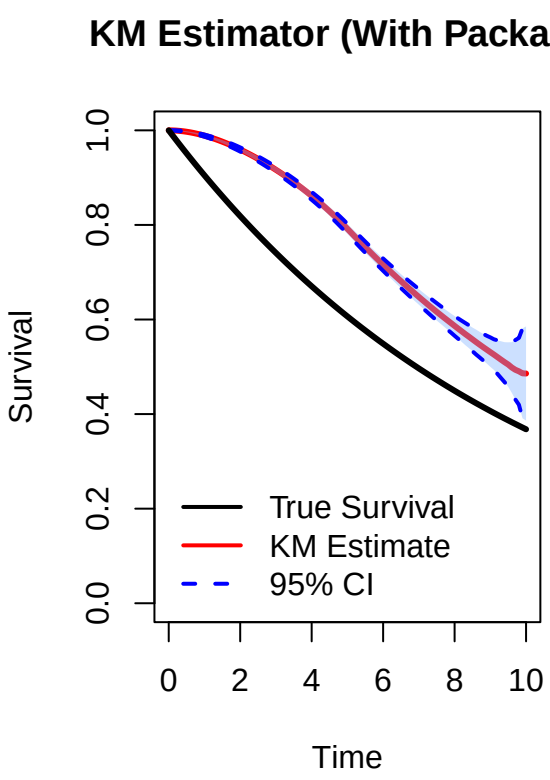
```r
KMb1_5 <- readRDS("KMb1_sim_beta1_5.rds")
m <- length(KMb1_5)
time_points <- KMb1_5[[1]]$time
n_time <- length(time_points)
```

```r
# -------- no_pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)
for (i in 1:m) {
  surv_mat[, i] <- KMb1_5[[i]]$surv_no_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)

z <- qnorm(0.975)
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb1_5_summary_no_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)

# -------- pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)

for (i in 1:m) {
  surv_mat[, i] <- KMb1_5[[i]]$surv_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb1_5_summary_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)
```

```r
par(mfrow = c(1, 2))

theta <- 0.1

# === no pkg ===
true_curve <- exp(-theta * KMb1_5_summary_no_pkg$Time)

plot(KMb1_5_summary_no_pkg$Time, KMb1_5_summary_no_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (No Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1), lwd = 3)

polygon(c(KMb1_5_summary_no_pkg$Time, rev(KMb1_5_summary_no_pkg$Time)),
        c(KMb1_5_summary_no_pkg$UpperCI, rev(KMb1_5_summary_no_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)
```

```r
lines(KMb1_5_summary_no_pkg$Time, KMb1_5_summary_no_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb1_5_summary_no_pkg$Time, KMb1_5_summary_no_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb1_5_summary_no_pkg$Time, true_curve, col = "black", lwd = 2, lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = c(3, 2, 2), bty = "n")

# === pkg ===
true_curve <- exp(-theta * KMb1_5_summary_pkg$Time)

plot(KMb1_5_summary_pkg$Time, KMb1_5_summary_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (With Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1),lwd = 3)

polygon(c(KMb1_5_summary_pkg$Time, rev(KMb1_5_summary_pkg$Time)),
        c(KMb1_5_summary_pkg$UpperCI, rev(KMb1_5_summary_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)

lines(KMb1_5_summary_pkg$Time, KMb1_5_summary_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb1_5_summary_pkg$Time, KMb1_5_summary_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb1_5_summary_pkg$Time, true_curve, col = "black", lwd = c(3, 2, 2), lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = 2, bty = "n")
```



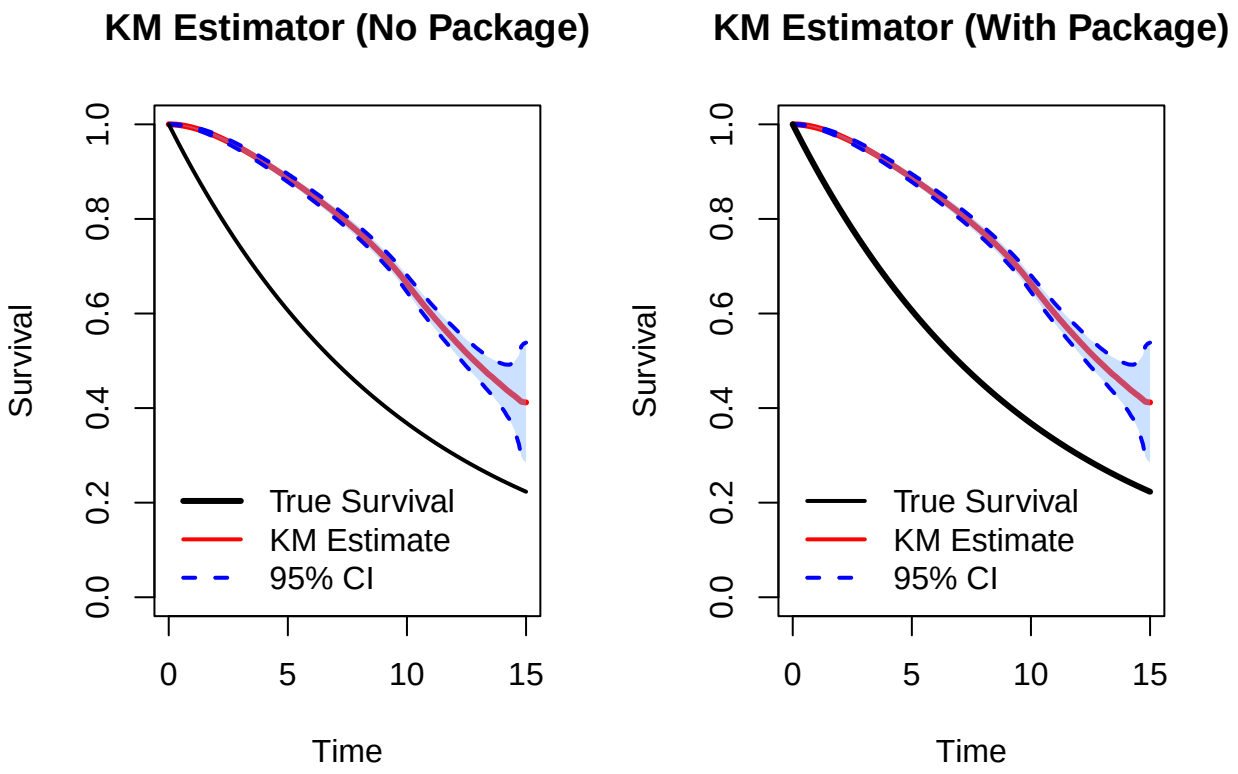

```r
m <- 1000
n <- 10000
theta <- 0.1
beta1 <- 10
beta2 <- 5
set.seed(12345)

#pb <- progress_bar$new(
#  format = " Simulating [:bar] :percent eta: :eta",
#  total = m, clear = FALSE, width = 60
#)

#KMb1_sim <- vector("list", m)
#for (i in 1:m) {
#  KMb1_sim[[i]] <- sim_censor(n, beta1, beta2, theta)
#  pb$tick()
#}

#saveRDS(KMb1_sim, file = "KMb1_sim_beta1_10.rds")
```

```r
KMb1_10 <- readRDS("KMb1_sim_beta1_10.rds")

m <- length(KMb1_10)
time_points <- KMb1_10[[1]]$time
n_time <- length(time_points)
```

```r
# -------- no_pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)
for (i in 1:m) {
  surv_mat[, i] <- KMb1_10[[i]]$surv_no_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)

z <- qnorm(0.975)
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb1_10_summary_no_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)

# -------- pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)

for (i in 1:m) {
  surv_mat[, i] <- KMb1_10[[i]]$surv_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb1_10_summary_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)
```

```r
par(mfrow = c(1, 2))

theta <- 0.1

# === no pkg ===
true_curve <- exp(-theta * KMb1_10_summary_no_pkg$Time)

plot(KMb1_10_summary_no_pkg$Time, KMb1_10_summary_no_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (No Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1), lwd = 3)

polygon(c(KMb1_10_summary_no_pkg$Time, rev(KMb1_10_summary_no_pkg$Time)),
        c(KMb1_10_summary_no_pkg$UpperCI, rev(KMb1_10_summary_no_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)
```

```r
lines(KMb1_10_summary_no_pkg$Time, KMb1_10_summary_no_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb1_10_summary_no_pkg$Time, KMb1_10_summary_no_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb1_10_summary_no_pkg$Time, true_curve, col = "black", lwd = 2, lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = c(3, 2, 2), bty = "n")

# === pkg ===
true_curve <- exp(-theta * KMb1_10_summary_pkg$Time)

plot(KMb1_10_summary_pkg$Time, KMb1_10_summary_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (With Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1),lwd = 3)

polygon(c(KMb1_10_summary_pkg$Time, rev(KMb1_10_summary_pkg$Time)),
        c(KMb1_10_summary_pkg$UpperCI, rev(KMb1_10_summary_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)

lines(KMb1_10_summary_pkg$Time, KMb1_10_summary_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb1_10_summary_pkg$Time, KMb1_10_summary_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb1_10_summary_pkg$Time, true_curve, col = "black", lwd = c(3, 2, 2), lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = 2, bty = "n")
```




**(c) Describe what you find in (b).**

We observe that the Kaplan-Meier (KM) estimators, both from the self-implemented version and the survfit() package, tend to overestimate the true survival function, particularly during the middle and later stages.

This overestimation is not only due to the increasing proportion of censored individuals, but also because the estimators fail to account for left-truncation in the data.

As time goes on, the proportion of censored observations increases, which introduces an upward bias in the estimated survival probabilities.

Additionally, the confidence intervals widen over time, reflecting the growing uncertainty due to fewer individuals remaining in the later stages.

**(d) Based on simulation data in (a), using Kaplan-Meier (left-truncated and right-censoring version), plot true and the average of estimated survival curve and 95% condence interval of estimated survival curve for both scenarios. (Please write down your own codes and compare to the result by using package)(Attached Code)**

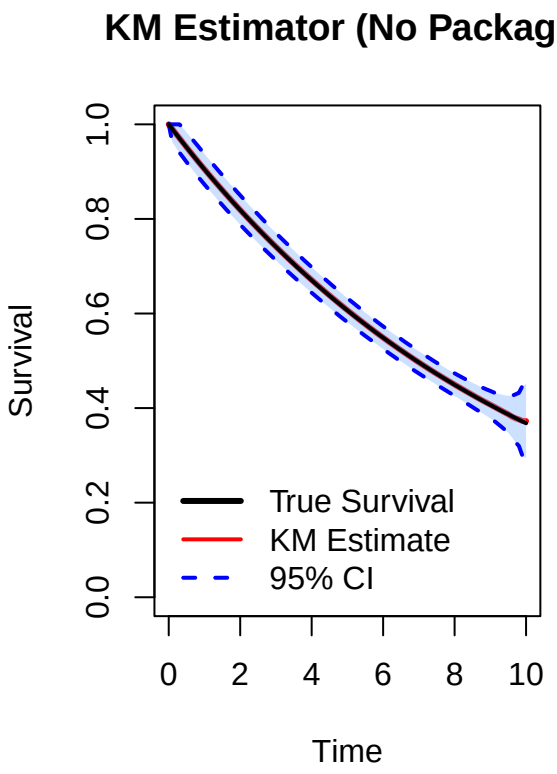
```
library(survival)
library(progress)

sim_truncate<- function(n, beta1, beta2, theta) {
  T <- rexp(n, theta)
  W <- runif(n, 0, beta1)
```

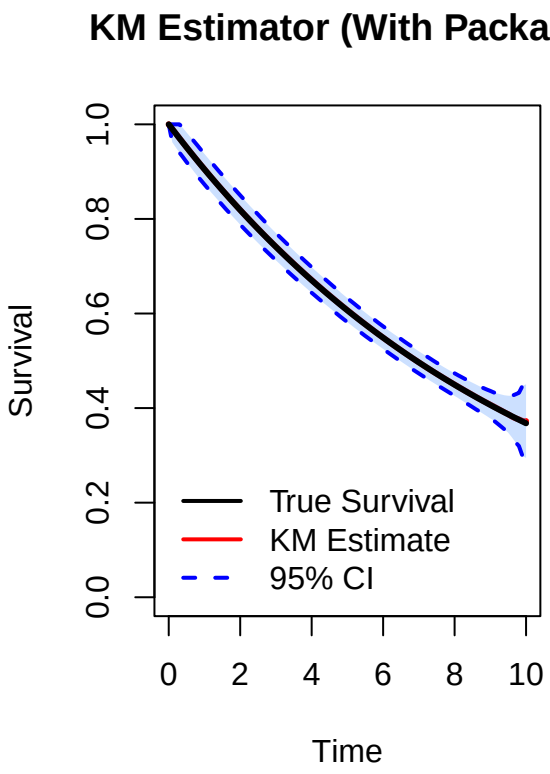
```r
  V <- runif(n, 0, beta2)
  C <- W + V

  idx <- T >= W
  T <- T[idx]
  W <- W[idx]
  C <- C[idx]

  status <- as.numeric(T <= C)
  T_obs <- pmin(T, C)

  df <- data.frame(T = T, W = W, C = C, T_obs = T_obs, status = status)
  df <- df[order(df$T_obs), ]

  # === no pkg ===
  time_points <- unique(df$T_obs[df$status == 1])
  d <- numeric(length(time_points))
  n_risk <- numeric(length(time_points))

  for (i in seq_along(time_points)) {
    t <- time_points[i]
    d[i] <- sum(df$T_obs == t & df$status == 1)
    n_risk[i] <- sum(df$T_obs >= t & df$W <= t)
  }
  surv_est <- cumprod(1 - d / n_risk)

  t_grid <- seq(0, beta1 + beta2, by = 0.1)

  breaks_manual <- c(0, time_points, Inf)
  surv_est_full <- c(1, surv_est, tail(surv_est, 1))  # S(0)=1, extend last value
  group_manual <- cut(t_grid, breaks = breaks_manual, labels = FALSE, right = TRUE)
  sr_no_pkg <- surv_est_full[group_manual]

  # ======= pkg =======
  fit <- survfit(Surv(time = df$W, time2 = df$T_obs, event = df$status) ~ 1)
  breaks_pkg <- c(0, fit$time, Inf)
  surv_pkg_full <- c(1, fit$surv, tail(fit$surv, 1))
  group_pkg <- cut(t_grid, breaks = breaks_pkg, labels = FALSE, right = TRUE)
  sr_pkg <- surv_pkg_full[group_pkg]

  sr_no_pkg[1] <- 1
  sr_pkg[1] <- 1

  return(data.frame(
    time = t_grid,
    surv_no_pkg = sr_no_pkg,
    surv_pkg = sr_pkg
  ))
}

# ---- Simulation ----
m <- 1000
n <- 10000
```

```r
theta <- 0.1
beta1 <- 5
beta2 <- 5
set.seed(12345)

#pb <- progress_bar$new(
#  format = " Simulating [:bar] :percent eta: :eta",
#  total = m, clear = FALSE, width = 60
#)

#KMb2_sim <- vector("list", m)
#for (i in 1:m) {
#  KMb2_sim[[i]] <- sim_truncate(n, beta1, beta2, theta)
#  pb$tick()
#}

#saveRDS(KMb2_sim, file = "KMb2_sim_beta1_5.rds")
```

```r
KMb2_5 <- readRDS("KMb2_sim_beta1_5.rds")

m <- length(KMb2_5)
time_points <- KMb2_5[[1]]$time
n_time <- length(time_points)

# -------- no_pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)
for (i in 1:m) {
  surv_mat[, i] <- KMb2_5[[i]]$surv_no_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)

z <- qnorm(0.975)
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb2_5_summary_no_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)

# -------- pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)

for (i in 1:m) {
  surv_mat[, i] <- KMb2_5[[i]]$surv_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)
```

```r
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb2_5_summary_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)

# -------- Draw CI --------
par(mfrow = c(1, 2))
theta <- 0.1

# === no pkg ===
true_curve <- exp(-theta * KMb2_5_summary_no_pkg$Time)

plot(KMb2_5_summary_no_pkg$Time, KMb2_5_summary_no_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (No Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1), lwd = 3)

polygon(c(KMb2_5_summary_no_pkg$Time, rev(KMb2_5_summary_no_pkg$Time)),
        c(KMb2_5_summary_no_pkg$UpperCI, rev(KMb2_5_summary_no_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)

lines(KMb2_5_summary_no_pkg$Time, KMb2_5_summary_no_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb2_5_summary_no_pkg$Time, KMb2_5_summary_no_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb2_5_summary_no_pkg$Time, true_curve, col = "black", lwd = 2, lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = c(3, 2, 2), bty = "n")

# === pkg ===
true_curve <- exp(-theta * KMb2_5_summary_pkg$Time)

plot(KMb2_5_summary_pkg$Time, KMb2_5_summary_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (With Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1),lwd = 3)

polygon(c(KMb2_5_summary_pkg$Time, rev(KMb2_5_summary_pkg$Time)),
        c(KMb2_5_summary_pkg$UpperCI, rev(KMb2_5_summary_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)

lines(KMb2_5_summary_pkg$Time, KMb2_5_summary_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb2_5_summary_pkg$Time, KMb2_5_summary_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb2_5_summary_pkg$Time, true_curve, col = "black", lwd = c(3, 2, 2), lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = 2, bty = "n")
```



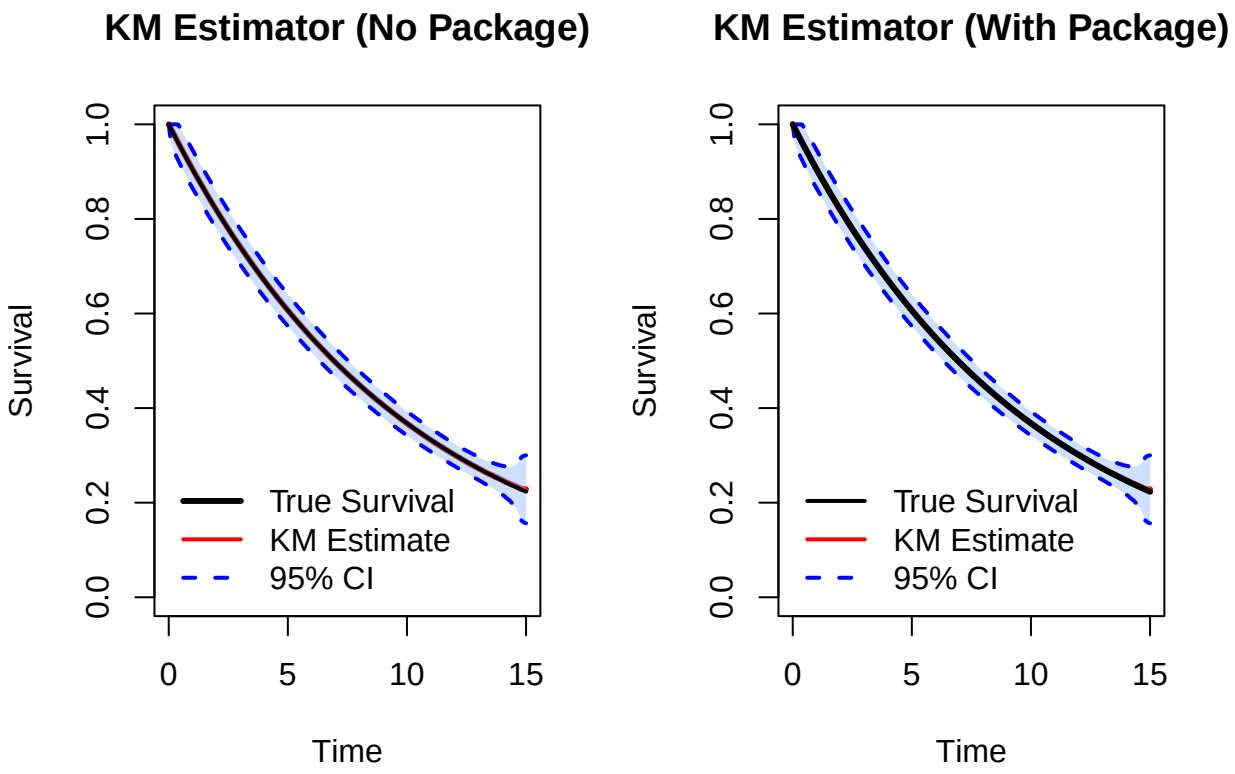




```r
# ---- Simulation ----
m <- 1000
n <- 10000
theta <- 0.1
beta1 <- 10
beta2 <- 5
set.seed(12345)

#pb <- progress_bar$new(
#  format = " Simulating [:bar] :percent eta: :eta",
#  total = m, clear = FALSE, width = 60
#)

#KMb2_sim <- vector("list", m)
#for (i in 1:m) {
#  KMb2_sim[[i]] <- sim_truncate(n, beta1, beta2, theta)
#  pb$tick()
#}

#saveRDS(KMb2_sim, file = "KMb2_sim_beta1_10.rds")
```

```r
KMb2_10 <- readRDS("KMb2_sim_beta1_10.rds")

m <- length(KMb2_10)
time_points <- KMb2_10[[1]]$time
n_time <- length(time_points)
```

```r
# -------- no_pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)
for (i in 1:m) {
  surv_mat[, i] <- KMb2_10[[i]]$surv_no_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)

z <- qnorm(0.975)
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb2_10_summary_no_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)

# -------- pkg --------
surv_mat <- matrix(NA, nrow = n_time, ncol = m)

for (i in 1:m) {
  surv_mat[, i] <- KMb2_10[[i]]$surv_pkg
}

avg_surv <- rowMeans(surv_mat, na.rm = TRUE)
sd_surv <- apply(surv_mat, 1, sd, na.rm = TRUE)
lower_ci <- pmax(0, avg_surv - z * sd_surv)
upper_ci <- pmin(1, avg_surv + z * sd_surv)

KMb2_10_summary_pkg <- data.frame(
  Time = time_points,
  Average_Survival = avg_surv,
  LowerCI = lower_ci,
  UpperCI = upper_ci
)

# -------- Draw CI --------
par(mfrow = c(1, 2))
theta <- 0.1

# === no pkg ===
true_curve <- exp(-theta * KMb2_10_summary_no_pkg$Time)

plot(KMb2_10_summary_no_pkg$Time, KMb2_10_summary_no_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (No Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1), lwd = 3)

polygon(c(KMb2_10_summary_no_pkg$Time, rev(KMb2_10_summary_no_pkg$Time)),
        c(KMb2_10_summary_no_pkg$UpperCI, rev(KMb2_10_summary_no_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)
```

```r
lines(KMb2_10_summary_no_pkg$Time, KMb2_10_summary_no_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb2_10_summary_no_pkg$Time, KMb2_10_summary_no_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb2_10_summary_no_pkg$Time, true_curve, col = "black", lwd = 2, lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = c(3, 2, 2), bty = "n")

# === pkg ===
true_curve <- exp(-theta * KMb2_10_summary_pkg$Time)

plot(KMb2_10_summary_pkg$Time, KMb2_10_summary_pkg$Average_Survival, type = "l", col = "red",
     main = "KM Estimator (With Package)", xlab = "Time", ylab = "Survival", ylim = c(0, 1),lwd = 3)

polygon(c(KMb2_10_summary_pkg$Time, rev(KMb2_10_summary_pkg$Time)),
        c(KMb2_10_summary_pkg$UpperCI, rev(KMb2_10_summary_pkg$LowerCI)),
        col = rgb(0.5, 0.7, 1, 0.4), border = NA)

lines(KMb2_10_summary_pkg$Time, KMb2_10_summary_pkg$LowerCI, col = "blue", lty = 2, lwd = 2)
lines(KMb2_10_summary_pkg$Time, KMb2_10_summary_pkg$UpperCI, col = "blue", lty = 2, lwd = 2)

lines(KMb2_10_summary_pkg$Time, true_curve, col = "black", lwd = c(3, 2, 2), lty = 1)

legend("bottomleft",
       legend = c("True Survival", "KM Estimate", "95% CI"),
       col = c("black", "red", "blue"), lty = c(1, 1, 2), lwd = 2, bty = "n")
```




**(e) Describe what you find in (d)**

By incorporating both left truncation and right censoring into the survival analysis, the Kaplan-Meier estimator closely match the true survival curve. Left truncation corrects for the selection bias introduced by delayed entry, where individuals who failed early are inherently excluded from observation.