Bortkewitsch's Horsekicks

20173204 곽명빈

2020-05-22

프러시아에서 20년간 14개 연대에서 발생한 말발굽에 채여 사망한 사고 기록 [^1] [^1]: 원시자료는 `pscl` 패키지의 `prussian` 이다.

```
# install.packages("pscl", repos = "https://cran.rstudio.com")
library(pscl)
library(extrafont)
data(prussian)
str(prussian)
```

```
## 'data.frame':    280 obs. of  3 variables:
##  $ y   : int  0 2 2 1 0 0 1 1 0 3 ...
##  $ year: int  75 76 77 78 79 80 81 82 83 84 ...
##  $ corp: Factor w/ 14 levels "G","I","II","III",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
head(prussian)
```

```
##   y year corp
## 1 0   75    G
## 2 2   76    G
## 3 2   77    G
## 4 1   78    G
## 5 0   79    G
## 6 0   80    G
```

```
table(prussian$y)
```

```
##
##   0   1   2   3   4
## 144  91  32  11   2
```

```
options(width = 180)
```

```
n_deaths <- 0:4
n_corps <- c(144, 91, 32, 11, 2)
horsekick <- data.frame(Deaths = n_deaths, Corps = n_corps)
horsekick
```

```
##   Deaths Corps
## 1      0   144
## 2      1    91
## 3      2    32
## 4      3    11
## 5      4     2
```

이 자료를 하나의 긴 벡터로 나타내는 것은 간단히

```
horsekick_long <- rep(n_deaths, n_corps)
str(horsekick_long)
```

```
##  int [1:280] 0 0 0 0 0 0 0 0 0 0 ...
```

```
summary(horsekick_long)
```

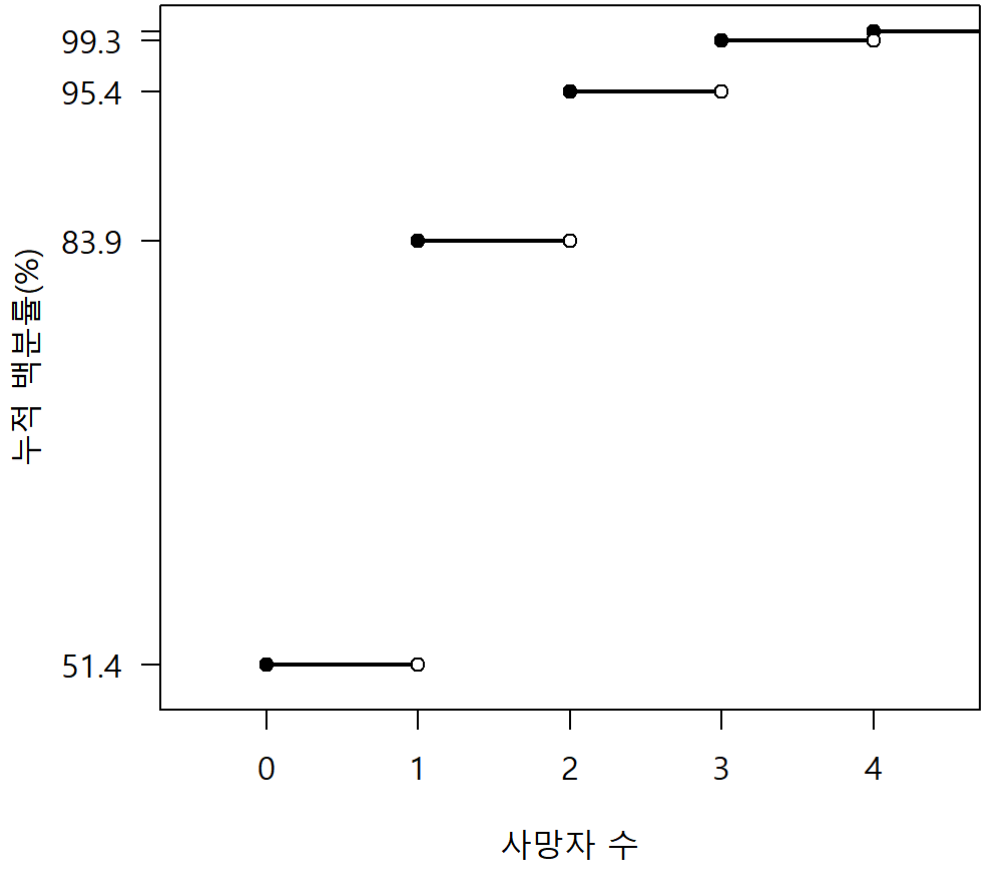
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0     0.0     0.0     0.7     1.0     4.0
```

위 자료를 누적분포로 도식화하기 위한 첫 작업

```
options(digits = 3)
#> 한글 폰트 설정
par(family = "Malgun Gothic")
#> 백분율로 표시
horsekick_p <- n_corps / sum(n_corps)
#> 누적백분율
horsekick_cum <- cumsum(horsekick_p) * 100
#> 백분율과 누적백분율 비교
cbind(horsekick_p, horsekick_cum)
```

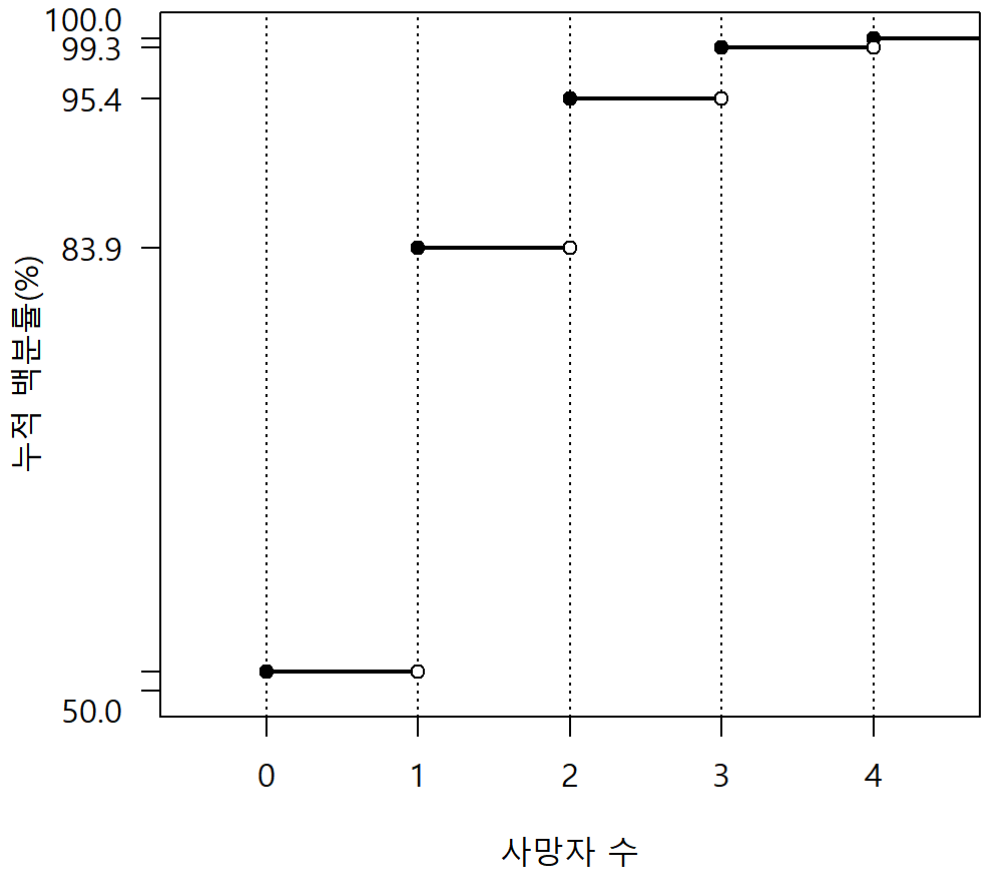
```
##      horsekick_p horsekick_cum
## [1,]     0.51429          51.4
## [2,]     0.32500          83.9
## [3,]     0.11429          95.4
## [4,]     0.03929          99.3
## [5,]     0.00714         100.0
```

```
#> 누적백분율을 점으로 표시. `yaxt = "n"` 의 효과 확인
plot(x = n_deaths, y = horsekick_cum,
     xlim = c(-0.5, 4.5), ylim = c(50, 100),
     xlab = "사망자 수", ylab = "누적 백분률(%)",
     yaxt = "n")
#> `axis()` 함수를 이용하여 `y`축 설정. `las = 2`의 역할에 유의
axis(side = 2,
     at = horsekick_cum,
     labels = format(horsekick_cum, nsmall = 1),
     las = 2)
lines(x = c(0, 1), y = rep(horsekick_cum[1], 2), lty = 1, lwd = 2)
lines(x = c(1, 2), y = rep(horsekick_cum[2], 2), lty = 1, lwd = 2)
lines(x = c(2, 3), y = rep(horsekick_cum[3], 2), lty = 1, lwd = 2)
lines(x = c(3, 4), y = rep(horsekick_cum[4], 2), lty = 1, lwd = 2)
lines(x = c(4, 5), y = rep(horsekick_cum[5], 2), lty = 1, lwd = 2)
points(x = n_deaths, y = horsekick_cum,
       pch = 21, bg = "black", col = "black")
points(x = n_deaths[2:5], y = horsekick_cum[1:4],
       pch = 21, bg = "white", col = "black")
```



누적 분포를 알기 쉽도록 격자 설정. 구분선을 아래서부터 차례로 그려갈 때

```
options(digits = 3)
par(family = "Malgun Gothic")
plot(x = n_deaths, y = horsekick_cum,
     xlim = c(-0.5, 4.5), ylim = c(50, 100),
     xlab = "사망자 수", ylab = "누적 백분률(%)",
     yaxt = "n")
axis(side = 2,
     at = c(50, horsekick_cum),
     labels = c("", format(horsekick_cum[1:4], digits = 2, nsmall = 1), ""),
     las = 2)
#> y축 50.0, 100.0의 눈금으로부터 라벨값의 위치를 조정하여 겹치지 않도록 조치
axis(side = 2,
     at = c(48.5, horsekick_cum[5] + 1.5),
     tick = FALSE,
     labels = format(c(50, horsekick_cum[5]), digits = 2, nsmall = 1),
     las = 2)
abline(v = 0:4, lty = 3)
lines(c(0, 1), rep(horsekick_cum[1], 2), lty = 1, lwd = 2)
lines(c(1, 2), rep(horsekick_cum[2], 2), lty = 1, lwd = 2)
lines(c(2, 3), rep(horsekick_cum[3], 2), lty = 1, lwd = 2)
lines(c(3, 4), rep(horsekick_cum[4], 2), lty = 1, lwd = 2)
lines(c(4, 5), rep(horsekick_cum[5], 2), lty = 1, lwd = 2)
points(x = n_deaths, y = horsekick_cum,
       pch = 21, bg = "black", col = "black")
points(x = n_deaths[2:5], y = horsekick_cum[1:4],
       pch = 21, bg = "white", col = "black")
```



반복되는 `lines()` 는 아래와 같이 `apply()` 를 이용하여 작업을 다소 줄일 수도 있다. 우선, 좌표를 정리한다.

```
h_x <- cbind(rep(0, 5), 1:5)
h_x
```

```
##      [,1] [,2]
## [1,]    0    1
## [2,]    0    2
## [3,]    0    3
## [4,]    0    4
## [5,]    0    5
```

```
h_y <- sapply(horsekick_cum, FUN = rep, times = 2)
h_y
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,] 51.4 83.9 95.4 99.3  100
## [2,] 51.4 83.9 95.4 99.3  100
```

```
h_xy <- cbind(h_x, t(h_y))
h_xy
```

```
##      [,1] [,2]  [,3]  [,4]
## [1,]    0    1  51.4  51.4
## [2,]    0    2  83.9  83.9
## [3,]    0    3  95.4  95.4
## [4,]    0    4  99.3  99.3
## [5,]    0    5 100.0 100.0
```

`plot()` 에서 `abline()` 까지가 반복해서 나올 것이므로, `source()` 를 이용하기 위한 코드 작성.

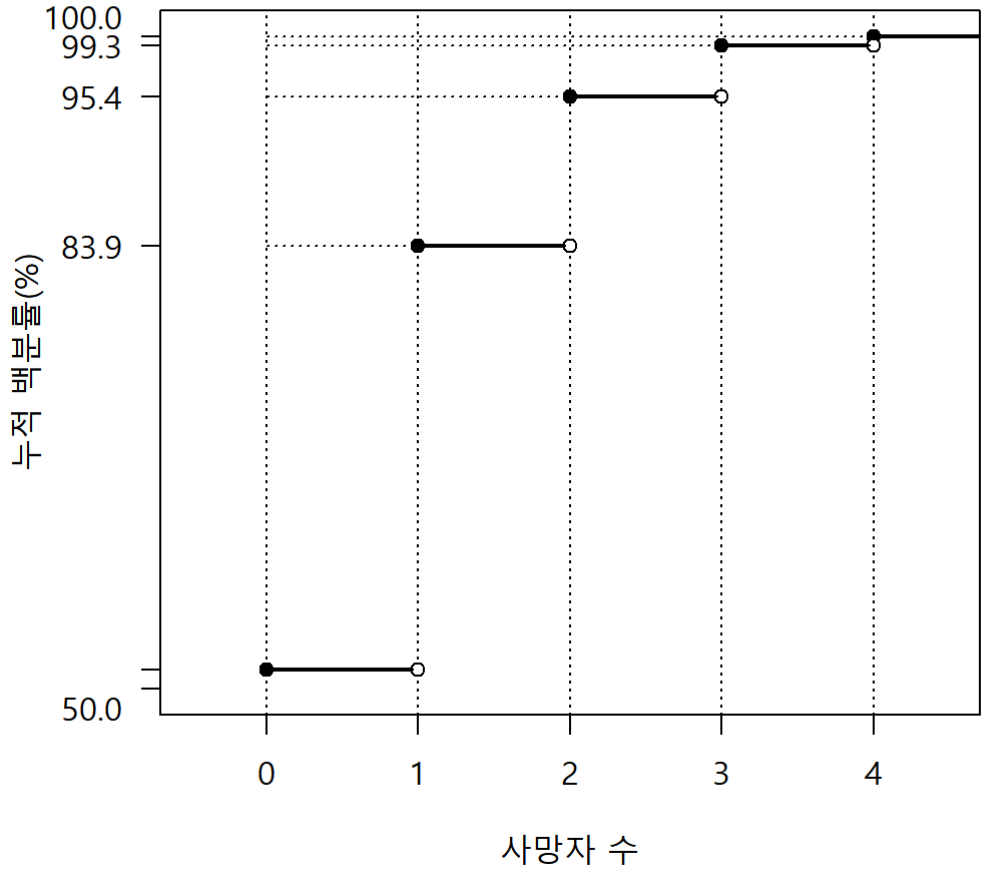
격자선을 놓는 과정까지를 두 줄로 줄일 수 있다.

```
par(family = "Malgun Gothic")
source("horsekick_plot.R", echo = TRUE)
```

```
##
## > plot(x = n_deaths, y = horsekick_cum, xlim = c(-0.5,
## +     4.5), ylim = c(50, 100), xlab = "사망자 수", ylab = "누적 백분률(%)",
## +     yaxt = "n")
```

```
##
## > axis(side = 2, at = c(50, horsekick_cum), labels = c("",
## +     format(horsekick_cum[1:4], nsmall = 1), ""), las = 2)
##
## > axis(side = 2, at = c(48.5, horsekick_cum[5] + 1.5),
## +     tick = FALSE, labels = format(c(50, horsekick_cum[5]), nsmall = 1),
## +     las = 2)
##
## > abline(v = 0:4, lty = 3)
##
## > lines(c(0, 1), rep(horsekick_cum[1], 2), lty = 1,
## +     lwd = 2)
##
## > lines(c(1, 2), rep(horsekick_cum[2], 2), lty = 1,
## +     lwd = 2)
##
## > lines(c(2, 3), rep(horsekick_cum[3], 2), lty = 1,
## +     lwd = 2)
##
## > lines(c(3, 4), rep(horsekick_cum[4], 2), lty = 1,
## +     lwd = 2)
##
## > lines(c(4, 5), rep(horsekick_cum[5], 2), lty = 1,
## +     lwd = 2)
##
## > points(x = n_deaths, y = horsekick_cum, pch = 21,
## +     bg = "black", col = "black")
##
## > points(x = tail(n_deaths, -1), y = head(horsekick_cum,
## +     -1), pch = 21, bg = "white", col = "black")
```

```
apply(h_xy, MARGIN = 1,
      FUN = function(h) lines(x = h[1:2], y = h[3:4], lty = 3))
```




```
## NULL
```

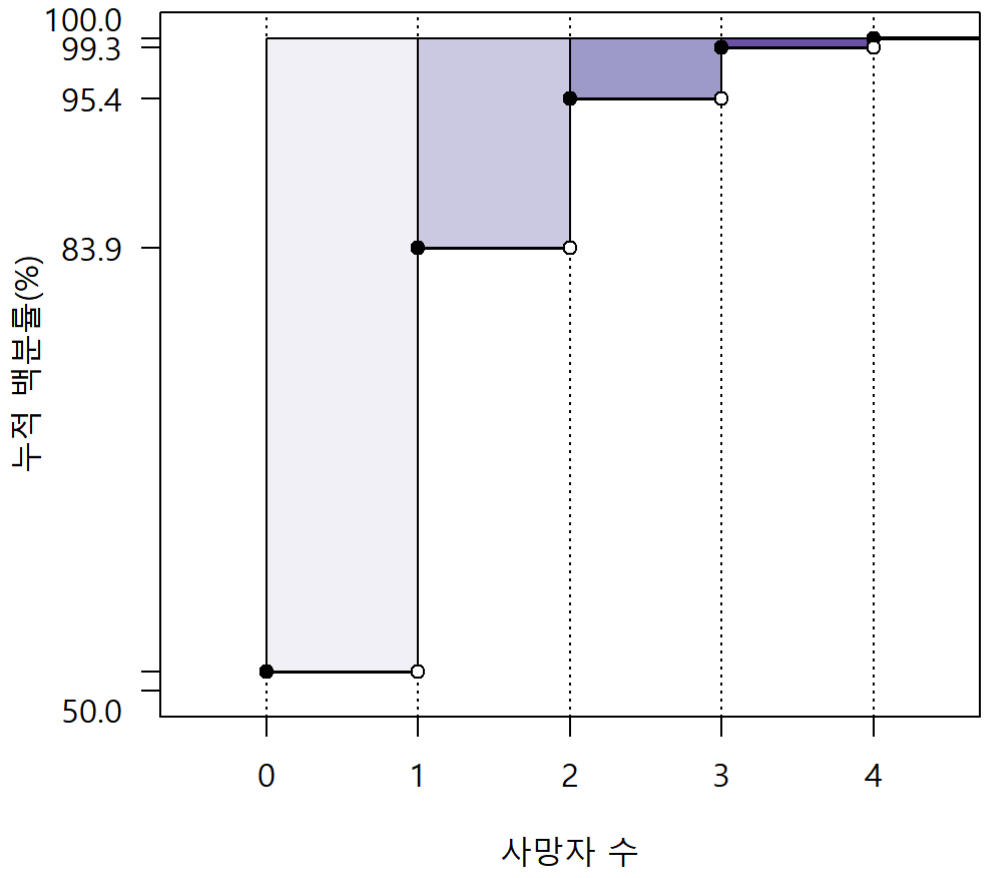
```
l_xy <- function(h) lines(x = h[1:2], y = h[3:4], lty = 3)
```

누적분포 윗 면적을 명확히 표시하기 위하여 빗금

```
par(family = "Malgun Gothic")
library(RColorBrewer)
h_pal <- brewer.pal(4, name = "Purples")
source("horsekick_plot.R", echo = FALSE)
apply(h_xy, MARGIN = 1, FUN = l_xy)
```

```
## NULL
```

```
polygon(c(0, 1, 1, 0),
        c(rep(horsekick_cum[1], 2), rep(horsekick_cum[5], 2)),
        col = h_pal[1])
polygon(c(1, 2, 2, 1),
        c(rep(horsekick_cum[2], 2), rep(horsekick_cum[5], 2)),
        col = h_pal[2])
polygon(c(2, 3, 3, 2),
        c(rep(horsekick_cum[3], 2), rep(horsekick_cum[5], 2)),
        col = h_pal[3])
polygon(c(3, 4, 4, 3),
        c(rep(horsekick_cum[4], 2), rep(horsekick_cum[5], 2)),
        col = h_pal[4])
points(x = n_deaths, y = horsekick_cum,
       pch = 21, bg = "black", col = "black")
points(x = n_deaths[2:5], y = horsekick_cum[1:4],
       pch = 21, bg = "white", col = "black")
```



`polygon()` 들을 한 줄로 정리하기 위하여 좌표들을 모으고,

```
poly_x <- matrix(c(0:3, rep(1:4, 2), 0:3), ncol = 4, byrow = T)
poly_x
```

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

```
poly_y <- rbind(sapply(horsekick_cum[1:4], rep, 2),
                matrix(rep(horsekick_cum[5], 8), nrow = 2))
poly_y
```

```
##       [,1]  [,2]  [,3]  [,4]
## [1,]  51.4  83.9  95.4  99.3
## [2,]  51.4  83.9  95.4  99.3
## [3,] 100.0 100.0 100.0 100.0
## [4,] 100.0 100.0 100.0 100.0
```

```
poly_xy <- rbind(poly_x, poly_y)
poly_xy
```

```
##       [,1]  [,2]  [,3]  [,4]
## [1,]   0.0   1.0   2.0   3.0
## [2,]   1.0   2.0   3.0   4.0
## [3,]   1.0   2.0   3.0   4.0
## [4,]   0.0   1.0   2.0   3.0
## [5,]  51.4  83.9  95.4  99.3
## [6,]  51.4  83.9  95.4  99.3
## [7,] 100.0 100.0 100.0 100.0
## [8,] 100.0 100.0 100.0 100.0
```

```
lapply(data.frame(poly_xy),
       FUN = function(v) cbind(v[1:4], v[5:8]))
```

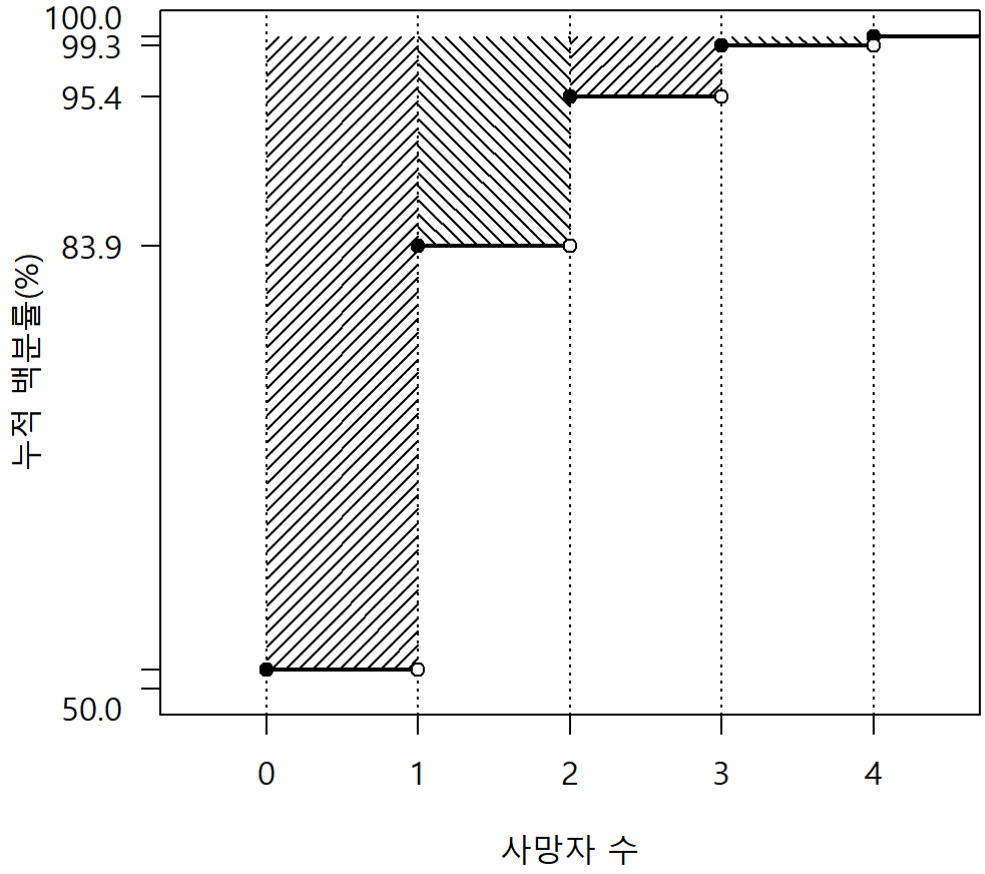
```
## $X1
##      [,1]  [,2]
## [1,]    0  51.4
## [2,]    1  51.4
## [3,]    1 100.0
## [4,]    0 100.0
##
## $X2
##      [,1]  [,2]
## [1,]    1  83.9
## [2,]    2  83.9
## [3,]    2 100.0
## [4,]    1 100.0
##
## $X3
##      [,1]  [,2]
## [1,]    2  95.4
## [2,]    3  95.4
## [3,]    3 100.0
## [4,]    2 100.0
##
## $X4
##      [,1]  [,2]
## [1,]    3  99.3
## [2,]    4  99.3
## [3,]    4 100.0
## [4,]    3 100.0
```

sapply() , lapply() , mapply() 를 적재적소에 활용하여 분량을 줄여보자.

```
par(family = "Malgun Gothic")
source("horsekick_plot.R")
# apply(h_xy, MARGIN = 1, FUN = l_xy)
mapply(polygon,
       lapply(data.frame(poly_xy),
              FUN = function(v) cbind(v[1:4], v[5:8])),
       density = 20, angle = c(45, 135, 45, 135), border = NA)
```

```
## $X1
## NULL
##
## $X2
## NULL
##
## $X3
## NULL
##
## $X4
## NULL
```

```
points(x = n_deaths, y = horsekick_cum,
       pch = 21, bg = "black", col = "black")
points(x = n_deaths[2:5], y = horsekick_cum[1:4],
       pch = 21, bg = "white", col = "black")
```



누적분포 윗 면적이 곧 평균임을 나타내기 위해 막대를 다른 방향으로 집적. 우선 좌표들을 다시 정리하자.

```
poly_x_2 <- matrix(c(rep(0, 4), rep(1:4, 2), rep(0, 4)),
                   ncol = 4, byrow = TRUE)
poly_y_2 <- rbind(sapply(horsekick_cum[1:4], rep, times = 2),
                  sapply(horsekick_cum[2:5], rep, times = 2))
poly_xy_2 <- rbind(poly_x_2, poly_y_2)
poly_xy_2
```

```
##      [,1] [,2] [,3]  [,4]
## [1,]  0.0  0.0  0.0   0.0
## [2,]  1.0  2.0  3.0   4.0
## [3,]  1.0  2.0  3.0   4.0
## [4,]  0.0  0.0  0.0   0.0
## [5,] 51.4 83.9 95.4  99.3
## [6,] 51.4 83.9 95.4  99.3
## [7,] 83.9 95.4 99.3 100.0
## [8,] 83.9 95.4 99.3 100.0
```

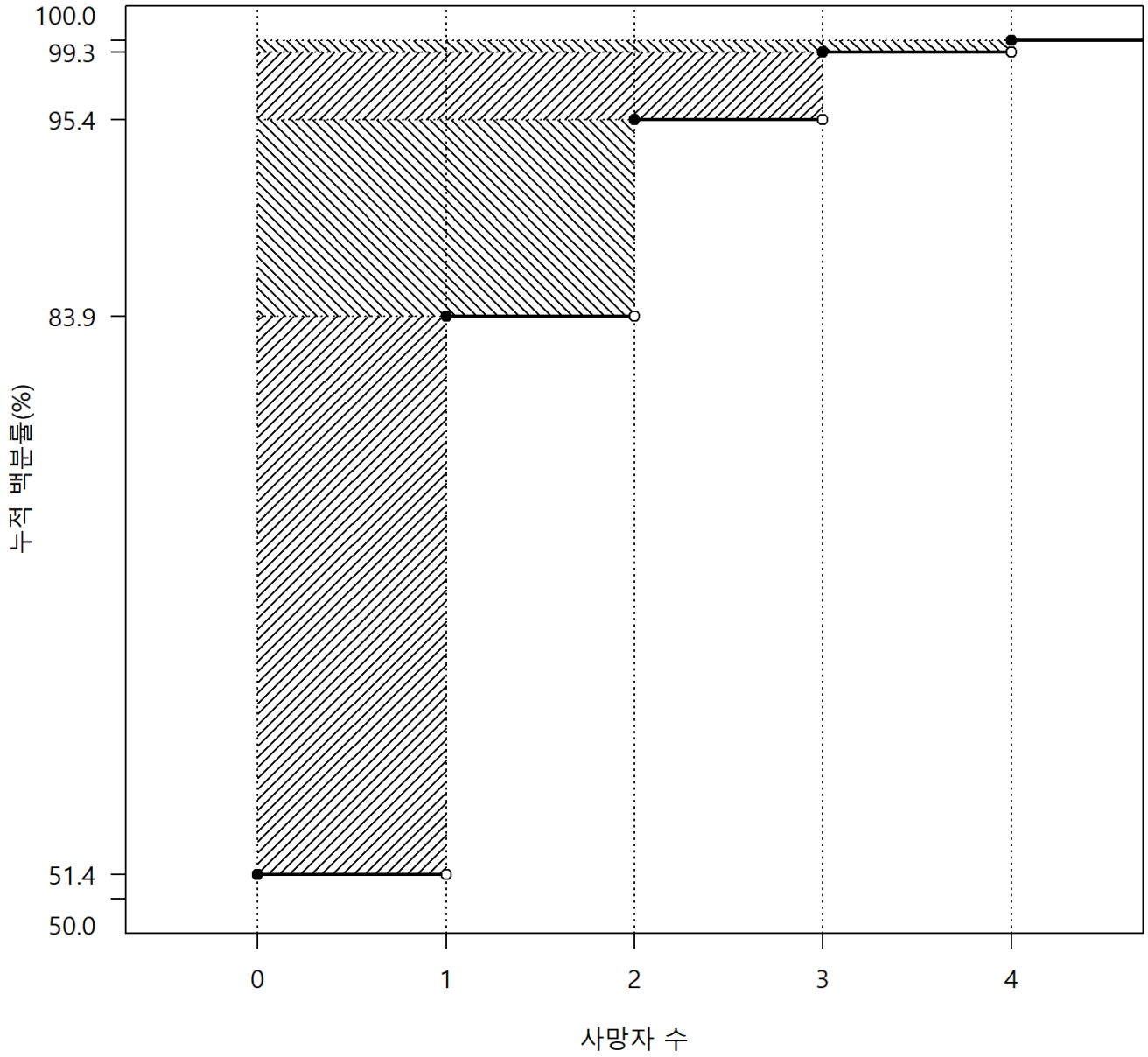
```
par(family = "Malgun Gothic")
source("horsekick_plot.R")
apply(h_xy, MARGIN = 1, FUN = l_xy)
```

```
## NULL
```

```
mapply(polygon,
       lapply(data.frame(poly_xy_2),
              function(v) cbind(v[1:4], v[5:8])),
       density = 20, angle = c(45, 135, 45, 135), border = NA)
```

```
## $X1
## NULL
##
## $X2
## NULL
##
## $X3
## NULL
##
## $X4
## NULL
```

```
points(x = n_deaths, y = horsekick_cum,
       pch = 21, bg = "black", col = "black")
points(x = n_deaths[2:5], y = horsekick_cum[1:4],
       pch = 21, bg = "white", col = "black")
```

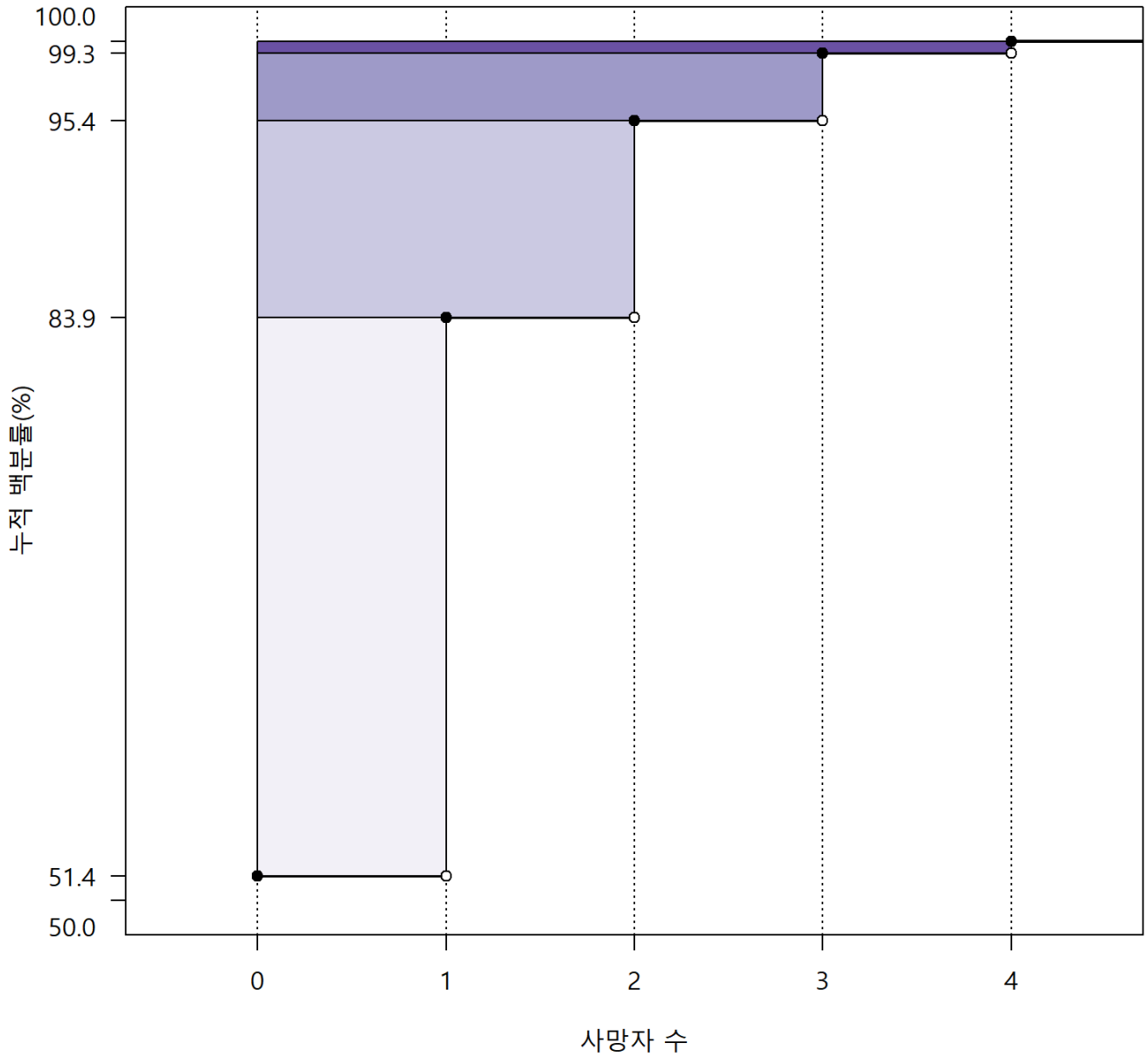


중간에 약간의 작업이 들어가긴 하지만 아래 소스코드와 비교하라.

```
options(digits = 3)
par(family = "Malgun Gothic")
source("horsekick_plot.R", echo = FALSE)
apply(h_xy, MARGIN = 1, FUN = l_xy)
```

```
## NULL
```

```
polygon(c(0, 1, 1, 0),
        c(rep(horsekick_cum[1], 2), rep(horsekick_cum[2], 2)),
        col = h_pal[1])
polygon(c(0, 2, 2, 0),
        c(rep(horsekick_cum[2], 2), rep(horsekick_cum[3], 2)),
        col = h_pal[2])
polygon(c(0, 3, 3, 0),
        c(rep(horsekick_cum[3], 2), rep(horsekick_cum[4], 2)),
        col = h_pal[3])
polygon(c(0, 4, 4, 0),
        c(rep(horsekick_cum[4], 2), rep(horsekick_cum[5], 2)),
        col = h_pal[4])
points(x = n_deaths, y = horsekick_cum,
       pch = 21, bg = "black", col = "black")
points(x = n_deaths[2:5], y = horsekick_cum[1:4],
       pch = 21, bg = "white", col = "black")
```



이번에 쌓아놓은 막대 면적의 합은 평균을 계산한 것임을 확인.

Comments

이번시간에는 자료를통해 누적분포를 도식화 하는방법에 대해 배울수 있었습니다. 데이터 프레임으로 자료를 확인하고 평균이 0.7이라는 것을 알게 되었는데 이를 도식화를 통해 평균을 구하는 방법을 알수있는 시간이 되었습니다. cumsum을통해 누적을 시키는법을 알수있었고,apply를통해 간단하게 만드는법을 알수있는 시간이였습니다.격자와 빗금을통해 자료의 평균을 알수있게 되었고, 한눈에 알아보기 쉽게 나타낼수 있어서 좋았습니다. 이번 과제를 통해 말발굽에 20년동안 14개 연대에서 말발굽에 치여 죽은 병사가 얼마나 있는지 데이터를 통해 알아볼수 있었습니다. 일부 사람이 말발굽에 의해 사고를 당했는데 생각보다 죽은 사람이 많아 안타깝게 느껴졌습니다… 또한 누적백분율을 통해 누적분포의 윗면적의 합이 평균이라는 사실을 그림을 통해 알 수 있는 계기가 되었습니다.