Data 405 Assignment 2 2025W

2025-10-21

Data 405 Assignment 2

Q1a code:

```
set.seed(12345)
n <- 10000
lambdas <- c(5, 25, 125, 625)

sim_results <- data.frame(lambda = lambdas, E_X = NA_real_, E_sqrtX = NA_real_, Var_sqrtX = NA_real_, Var_X = NA_
real_)

for (i in seq_along(lambdas)) {
    lam <- lambdas[i]
    P <- rpois(n, lambda = lam)
    sim_results$E_X[i]       <- mean(P)
    sim_results$E_sqrtX[i]   <- mean(sqrt(P))
    sim_results$Var_sqrtX[i] <- var(sqrt(P))
    sim_results$Var_X[i]     <- var(P)
}

print(sim_results)
```

```
##   lambda      E_X   E_sqrtX Var_sqrtX      Var_X
## 1      5   4.9990  2.172351 0.2799170   4.923491
## 2     25  24.9790  4.972203 0.2562207  25.217481
## 3    125 125.3005 11.182422 0.2539532 126.881888
## 4    625 625.3433 25.001937 0.2464837 616.709916
```

Q1b discussion:

Taking the sqrt compresses larger counts and reduces the dependence of variance on the mean. This is called ‘variance stabilization’ for count data — after transformation the variance becomes roughly constant across levels of the mean, - which makes many statistical methods (ANOVA, ANCoVA, & lin reg) more appropriate

Q2a code:

since u = 1-(e$^{(-x}$2)), we can use algebra to find x=sqrt{-ln(1-u)}

thus:

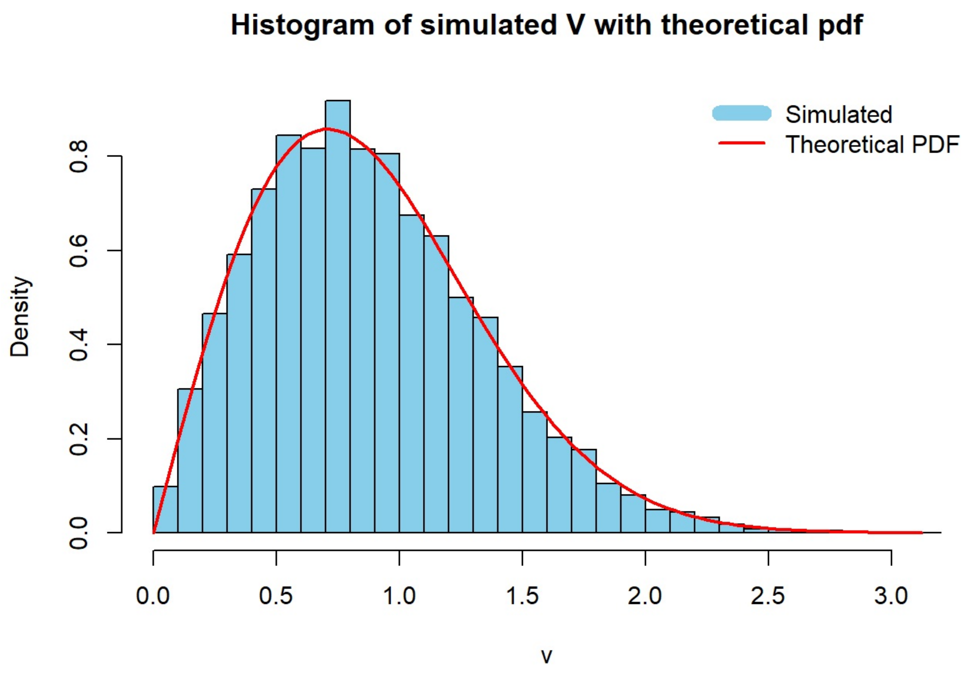
```
rmyV <- function(n) {
  u <- runif(n)  # uniform rands on (0,1)
  v <- sqrt(-log(1 - u))
  return(v)
}
```

Q2b code:

```
set.seed(123)
v <- rmyV(10000)

# histogram
hist(v, breaks = 40, freq = FALSE, col = "skyblue",
     main = "Histogram of simulated V with theoretical pdf",
     xlab = "v")

# expected pdf
curve(2 * x * exp(-x^2), from = 0, to = max(v), add = TRUE, col = "red", lwd = 2)
legend("topright", legend = c("Simulated", "Theoretical PDF"),
       col = c("skyblue", "red"), lwd = c(10, 2), bty = "n")
```




```
# this should be = d/dx f(x)
```

Q3a discussion:

F(x) = int(0, x)(3t^2)dt = t^3 (0, x) = x^3

Q3b code:

inverse func: instead of u = x^(3) we use x = u^(1/3)

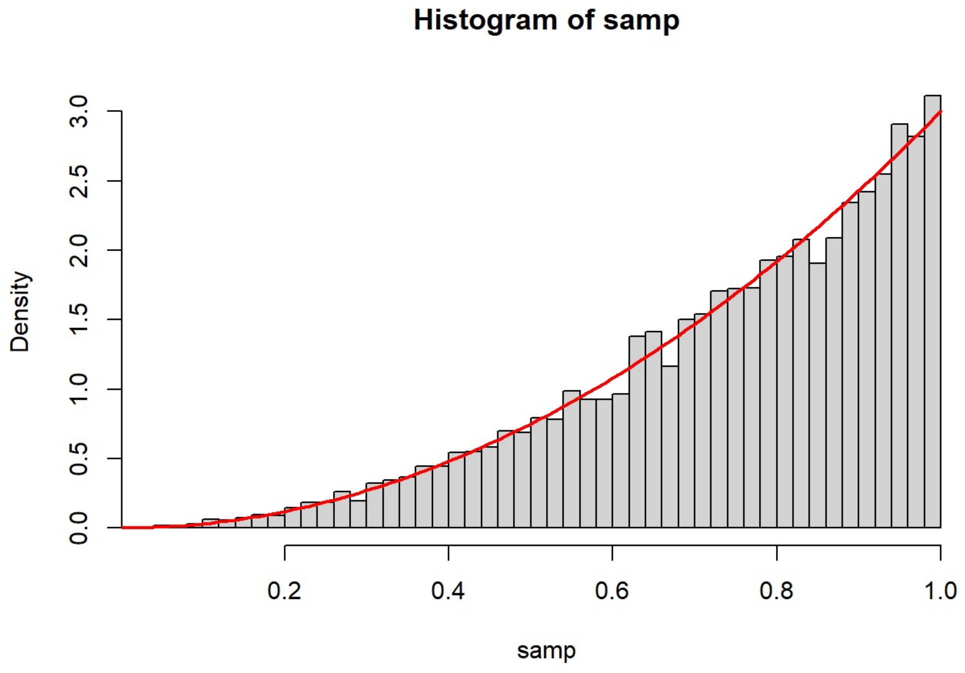
```
rmyX <- function(n) {
  u <- runif(n)
  x <- u^(1/3)
  return(x)
}
```

E[x] = int(0, 1)(x * 3x^2)dx = 3 * int(0, 1)(x^3)dx = 3 * x^4 /4 (0, 1) = 3* 1/4 - 0 = 3/4 = 0.75

```
set.seed(1)
samp <- rmyX(10000)
mean(samp)   # expect 0.75 +/-
```

```
## [1] 0.7492406
```

```
hist(samp, prob=TRUE, breaks=40)
curve(3*x^2, from=0, to=1, add=TRUE, col="red", lwd=2)
```



0.7492406 is indeed within acceptable tolerance of expected 0.75

Q4a discussion:

FY(y) = pFV(y) + (1−p) FX(y)

recall:

FV(y)=0 & FV(y) = 1-e$^{(-y}$2) FX(y)=0 & FX(y) = y^3

and thus:

FY(y) = p(1-e(-y^2)) + (1-p)y^3 when 0<y<1 & FY(y) = p(1-e(-y^2)) + (1-p) = 1 - p * e$^{(-y}$2)

Q4b code:

```
rmyY <- function(n, p) {
  # n: number of draws
  # p: probability of selecting V
  chooseV <- rbinom(n, size = 1, prob = p)  # 1 => use V, 0 => use X
  result <- numeric(n)

  nV <- sum(chooseV == 1)
  nX <- n - nV

  if (nV > 0) result[which(chooseV == 1)] <- rmyV(nV)
  if (nX > 0) result[which(chooseV == 0)] <- rmyX(nX)

  return(result)
}
```

Q4c code:

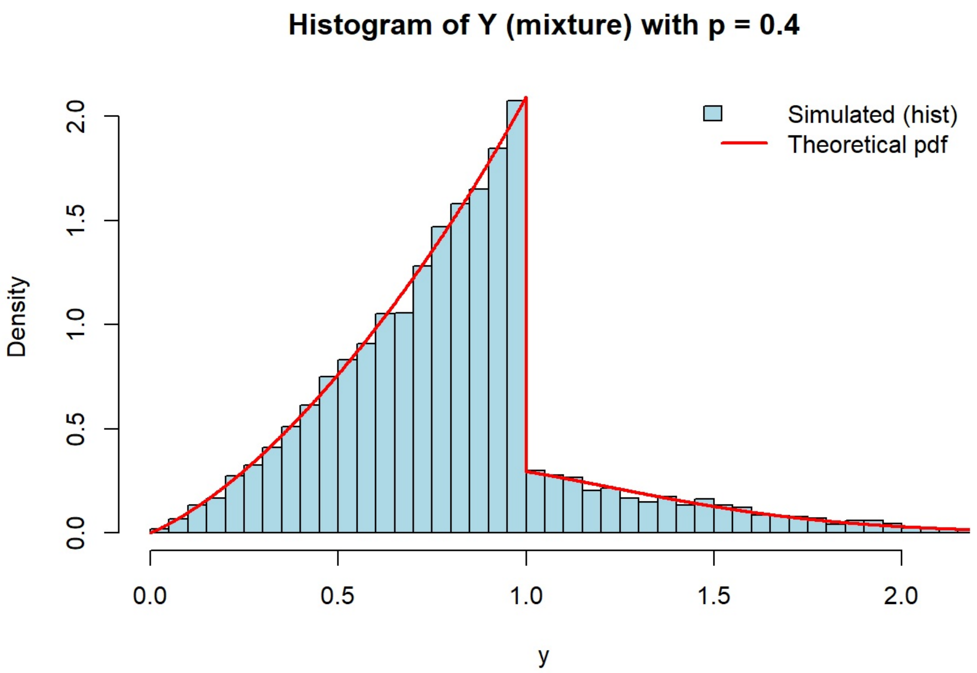
```
set.seed(2025)
p <- 0.4
n <- 10000
y <- rmyY(n, p)

# hist gram (relative frequency hist)
hist(y, breaks = 60, freq = FALSE, col = "lightblue",
     main = paste("Histogram of Y (mixture) with p =", p),
     xlab = "y", xlim = c(0, quantile(y, 0.995)))  # limit x for nicer view

# define pdf funct
g_pdf <- function(x, p) {
  out <- numeric(length(x))
  # for 0 <= x <= 1
  idx1 <- which(x >= 0 & x <= 1)
  if (length(idx1) > 0) {
    out[idx1] <- p * (2 * x[idx1] * exp(-x[idx1]^2)) + (1 - p) * (3 * x[idx1]^2)
  }
  # for x > 1
  idx2 <- which(x > 1)
  if (length(idx2) > 0) {
    out[idx2] <- p * (2 * x[idx2] * exp(-x[idx2]^2))
  }
  return(out)
}

# overlay pdf curve
xs <- seq(0, max(y), length.out = 2000)
lines(xs, g_pdf(xs, p), col = "red", lwd = 2)

legend("topright", legend = c("Simulated (hist)", "Theoretical pdf"),
       fill = c("lightblue", NA), border = c("black", NA),
       lty = c(NA, 1), col = c(NA, "red"), lwd = c(NA, 2), bty = "n")
```



The graph is shaped like " _/-. " because the function is defined differently above y = 1 & y between 0&1 – hence the piecewise

Q5 code:

reverse func is X = sqrt{ -ln(1-2U) }

```
rmyH <- function(n) {
  u <- runif(n)
  x <- sqrt(-log(1 - 2 * u))
  sign <- sample(c(-1, 1), n, replace = TRUE) #random even odd since h(x) is symetric
  return(sign * x)
}
```

```r
rmyW <- function(n, a, b, p) {
  # choose which component
  component <- rbinom(n, 1, p)

  # sample from h(x)
  w <- rmyH(n)

  # shift depending on mixture
  w[component == 1] <- w[component == 1] + a
  w[component == 0] <- w[component == 0] + b

  return(w)
}
```

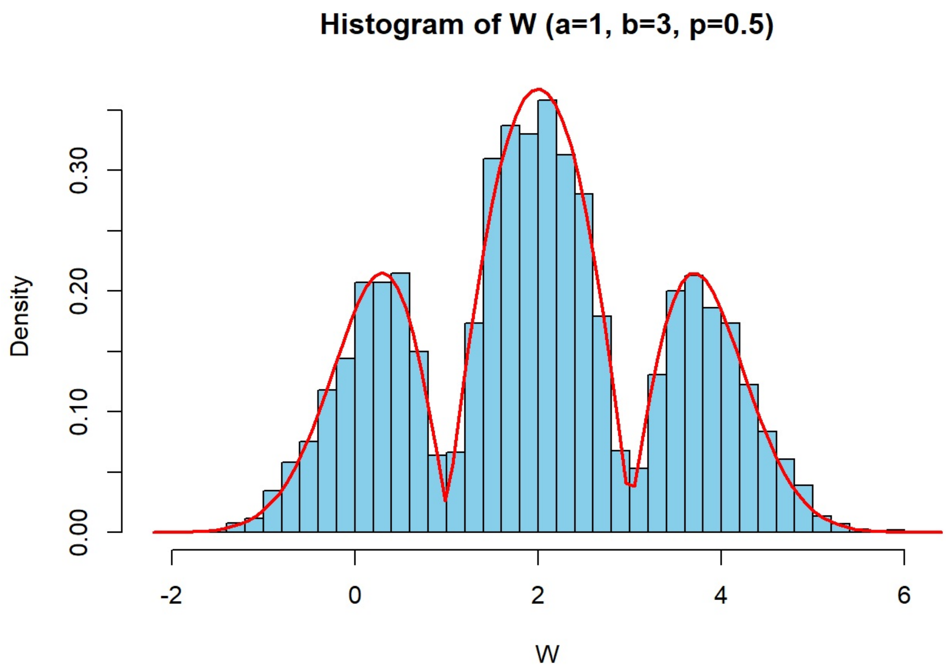
Q5 plot:

```r
set.seed(123)
W1 <- rmyW(10000, a = 1, b = 3, p = 0.5)
```

```
## Warning in log(1 - 2 * u): NaNs produced
```

```r
hist(W1, breaks = 50, freq = FALSE, col = "skyblue",
     main = "Histogram of W (a=1, b=3, p=0.5)",
     xlab = "W")

# overlay pdf curve
curve(0.5 * abs(x - 1) * exp(-(x - 1)^2) +
      0.5 * abs(x - 3) * exp(-(x - 3)^2),
      add = TRUE, col = "red", lwd = 2)
```

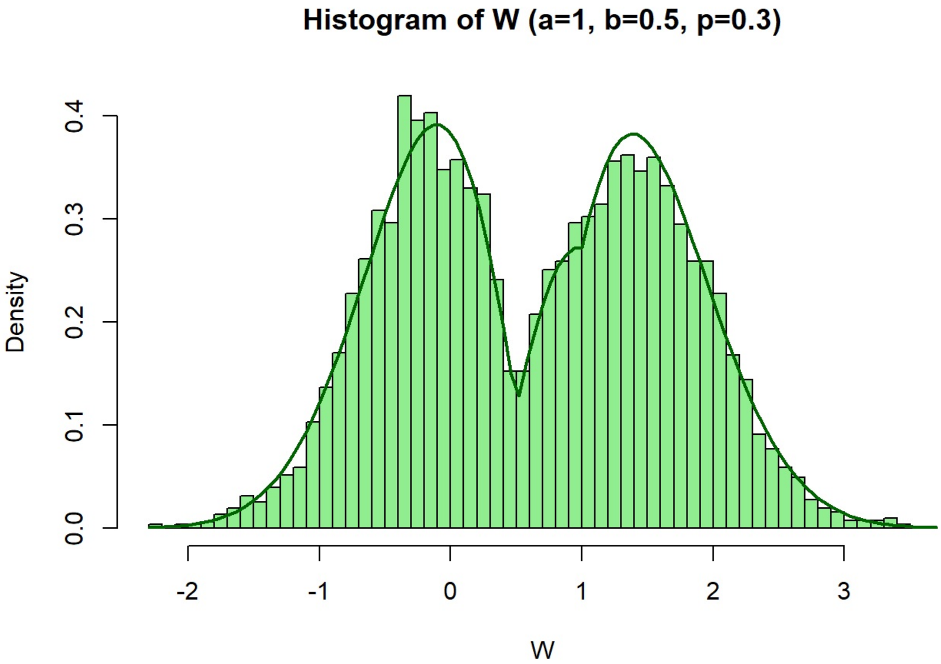



```r
set.seed(456)
W2 <- rmyW(10000, a = 1, b = 0.5, p = 0.3)
```

```
## Warning in log(1 - 2 * u): NaNs produced
```

```r
hist(W2, breaks = 50, freq = FALSE, col = "lightgreen",
     main = "Histogram of W (a=1, b=0.5, p=0.3)",
     xlab = "W")

curve(0.3 * abs(x - 1) * exp(-(x - 1)^2) +
      0.7 * abs(x - 0.5) * exp(-(x - 0.5)^2),
      add = TRUE, col = "darkgreen", lwd = 2)
```



trimodal & bimodal spectrums respectively

for (a=1, b=3, p=0.5) (blue), we see two clear dips near 1 and 3 & for (a=1, b=0.5, p=0.3), we only see 2 modes with a dip slightly below 0.5