Solution to Series 6

1. Assume that the distribution of $\hat{\theta}_n^* - \hat{\theta}_n$ is symmetric around zero, then $-q_{\hat{\theta}_n^* - \hat{\theta}_n}(1-\alpha/2) = q_{\hat{\theta}_n^* - \hat{\theta}_n}(\alpha/2)$.

Hence, the left end of the reversed quantile bootstrap confidence interval can be reformulated as follows:

$$\begin{aligned}\hat{\theta}_n - q_{\hat{\theta}_n^* - \hat{\theta}_n}(1-\alpha/2) &= \hat{\theta}_n + q_{\hat{\theta}_n^* - \hat{\theta}_n}(\alpha/2) \\ &= \hat{\theta}_n + q_{\hat{\theta}_n^*}(\alpha/2) - \hat{\theta}_n \\ &= q_{\hat{\theta}_n^*}(\alpha/2).\end{aligned}$$

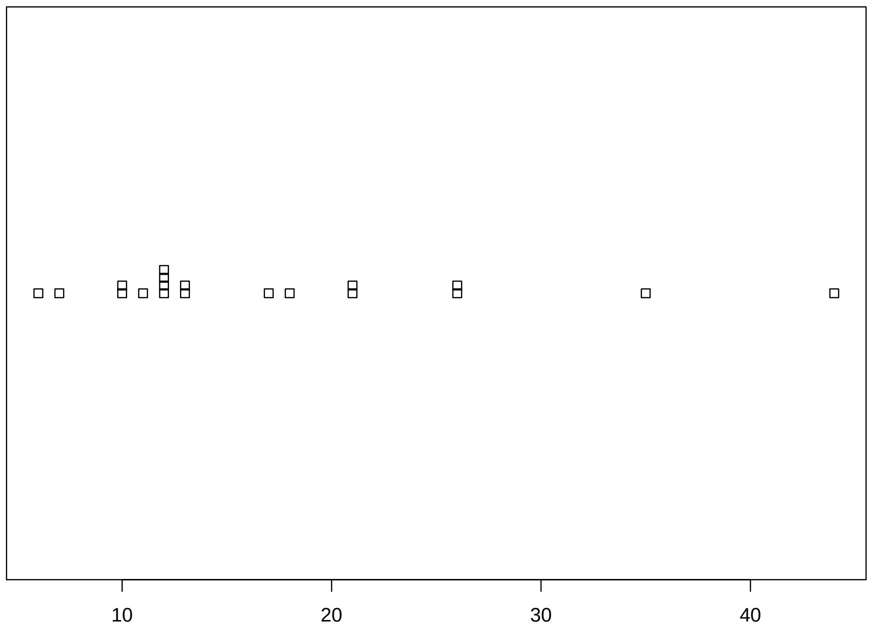
A similar derivation for the right end of the reversed quantile bootstrap confidence interval yields

$$\hat{\theta}_n - q_{\hat{\theta}_n^* - \hat{\theta}_n}(\alpha/2) = q_{\hat{\theta}_n^*}(1-\alpha/2).$$

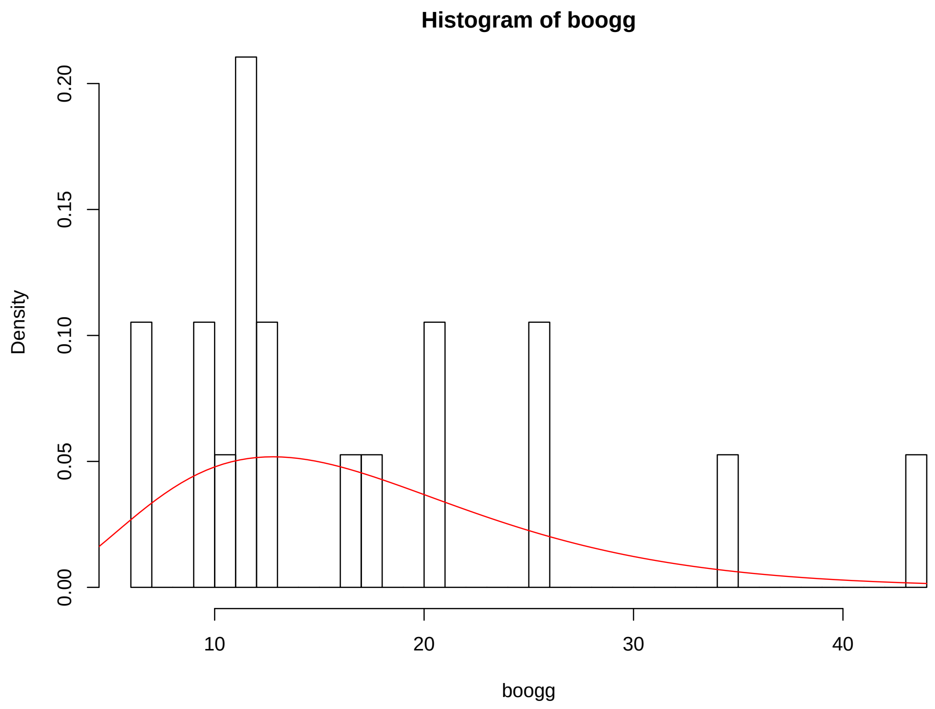
This shows that the quantile and reversed quantile bootstrap confidence intervals are identical.

2. a)

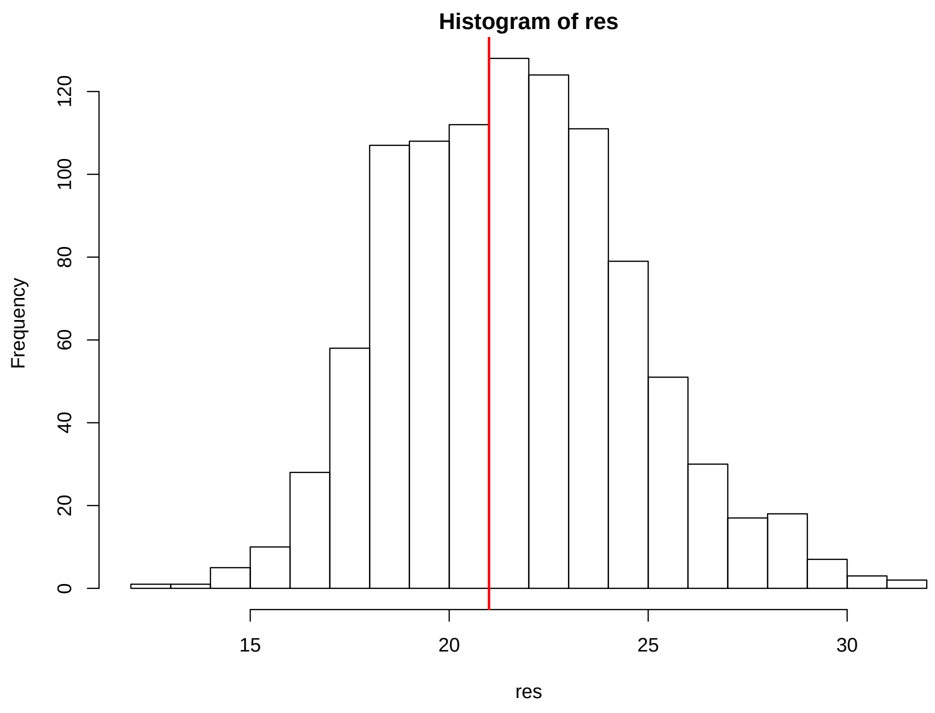
```
> # The values are rounded to minutes (from 2000 to 2018).
> boogg <- c(17, 26, 12, 6, 12, 18, 10, 12, 26, 13, 13, 11, 12, 35, 7, 21, 44,
            10, 21)
> stripchart(boogg, method = "stack")
> # Alternatively
> stem(boogg)
  The decimal point is 1 digit(s) to the right of the |

  0 | 67
  1 | 00122223378
  2 | 1166
  3 | 5
  4 | 4
> require("MASS")
> (fit.gamma <- fitdistr(boogg, "gamma"))
     shape         rate
  3.91681011   0.22828203
 (1.22046966) (0.07589377)
```



b)
```
> # Plot the density on top of the histogram
> hist(boogg, freq = FALSE, breaks = 50)
> lines(x = seq(from = 0, to = max(boogg), by = 0.4),
       y = dgamma(x = seq(from = 0, to = max(boogg), by = 0.4),
                  shape = fit.gamma$estimate["shape"],
                  rate = fit.gamma$estimate["rate"]),
       col = 2)
```



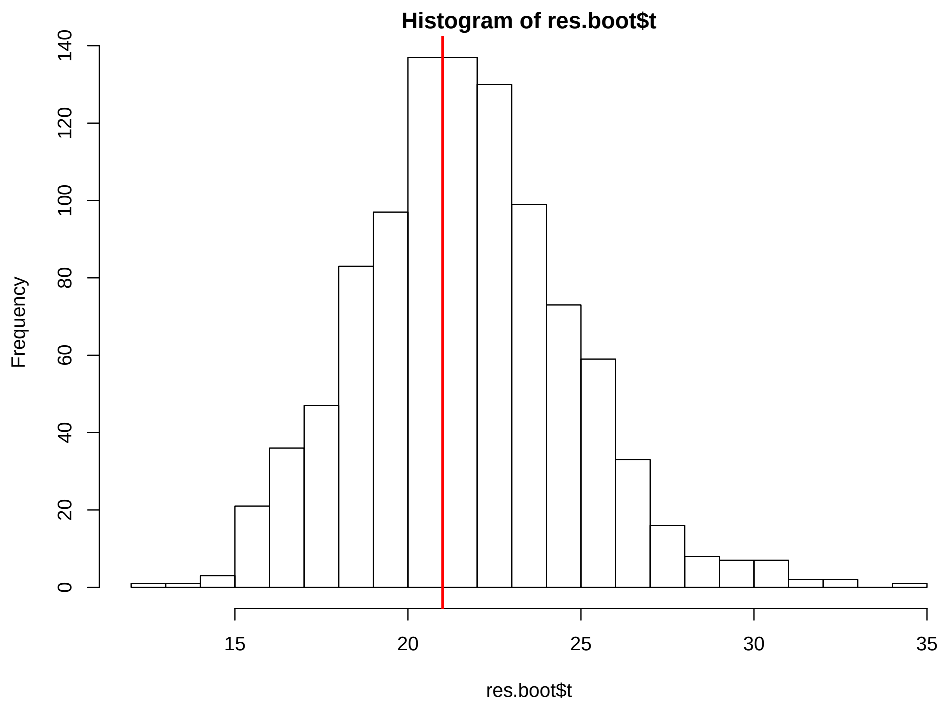
c)
```
> R <- 1000
> len.b <- length(boogg)
> set.seed(987)
> res <- rep(NA, R)
> for (i in 1:R) {
    x <- rgamma(n = len.b, shape = fit.gamma$estimate["shape"],
                rate = fit.gamma$estimate["rate"])
    res[i] <- quantile(x, probs = 0.75)
  }
> # Plot theta*
> hist(res, breaks = 20)
> abline(v = quantile(boogg, probs = 0.75), col = 2, lwd = 2)
```



d)
```
> # theta hat
> boogg.75quant <- quantile(boogg, probs = 0.75)
> names(boogg.75quant) <- NULL
> # quantile CI
> (CI.q1 <- quantile(res, probs = c(0.025, 0.975)))

    2.5%    97.5%
16.53821 28.14101

> # reversed CI
> CI.r1 <- boogg.75quant - quantile(res - boogg.75quant, probs = c(0.975, 0.025))
> names(CI.r1) <- NULL
> CI.r1

[1] 13.85899 25.46179

> # normal approx CI
> (CI.n1 <- c(boogg.75quant - qnorm(0.975) * sd(res), boogg.75quant + qnorm(0.975) * sd(res)))

[1] 15.12878 26.87122
```

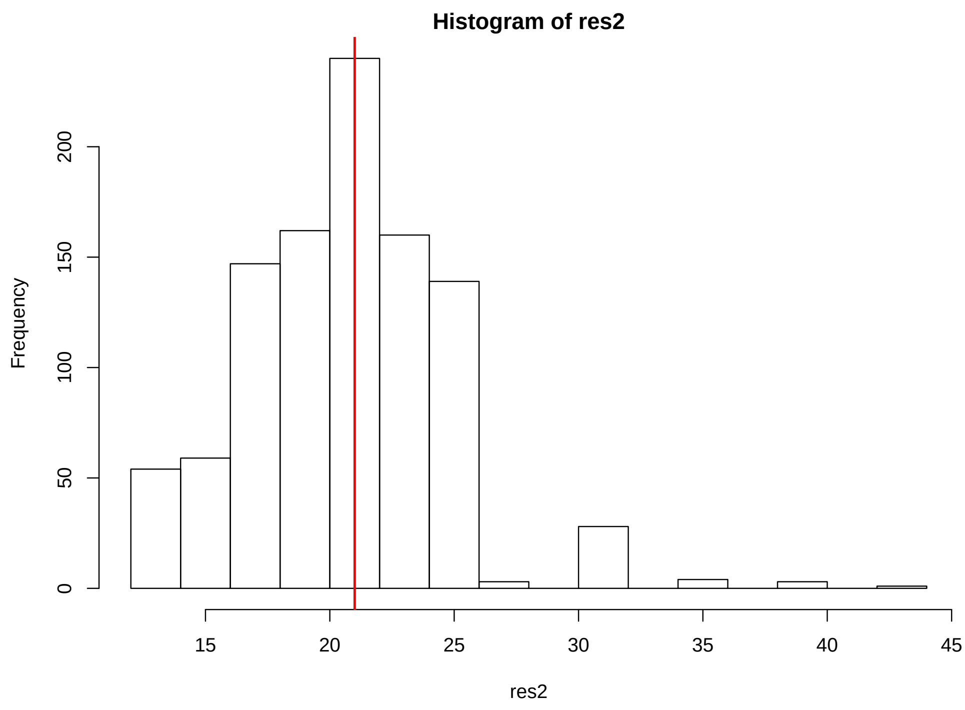
e)
```
> require("boot")
> fun.theta <- function(x, ind) {quantile(x[ind], probs = 0.75)}
> fun.gen <- function(x, mle) {
   rgamma(length(x), shape = mle[1], rate = mle[2])
 }
> res.boot <- boot(boogg, fun.theta, R = 1000, sim = "parametric",
                  ran.gen = fun.gen, mle = fit.gamma$estimate)
> res.boot

PARAMETRIC BOOTSTRAP


Call:
boot(data = boogg, statistic = fun.theta, R = 1000, sim = "parametric",
    ran.gen = fun.gen, mle = fit.gamma$estimate)


Bootstrap Statistics :
    original   bias    std. error
t1*       21 0.663449    3.056728
```

```
> # Plot theta*
> hist(res.boot$t, breaks = 20)
> abline(v = boogg.75quant, col = 2, lwd = 2)
> # Calculate CIs
> (res.boot.ci <- boot.ci(res.boot, type = c("norm", "basic", "perc")))
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = res.boot, type = c("norm", "basic", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   (14.35, 26.33 )   (13.84, 26.07 )   (15.93, 28.16 )
Calculations and Intervals on Original Scale
```



The CIs are very similar, obviously they do not have to be exactly the same since they are based on different bootstrap samples.

f)
```
> R <- 1000
> res2 <- rep(NA, R)
> for (i in 1:R) {
    ind <- sample(1:len.b, len.b, replace = TRUE)
    res2[i] <- quantile(boogg[ind], probs = 0.75)
  }
> hist(res2, breaks = 20)
> abline(v = boogg.75quant, col = 2, lwd = 2)
```



```
> # quantile CI
> (CI.q2 <- quantile(res2, probs = c(0.025, 0.975)))
```

```
 2.5% 97.5%
 13.0  30.5
```

```
> # reversed CI
> CI.r2 <- boogg.75quant - quantile(res2 - boogg.75quant, probs = c(0.975, 0.025))
> names(CI.r2) <- NULL
> CI.r2
```

```
[1] 11.5 29.0
```

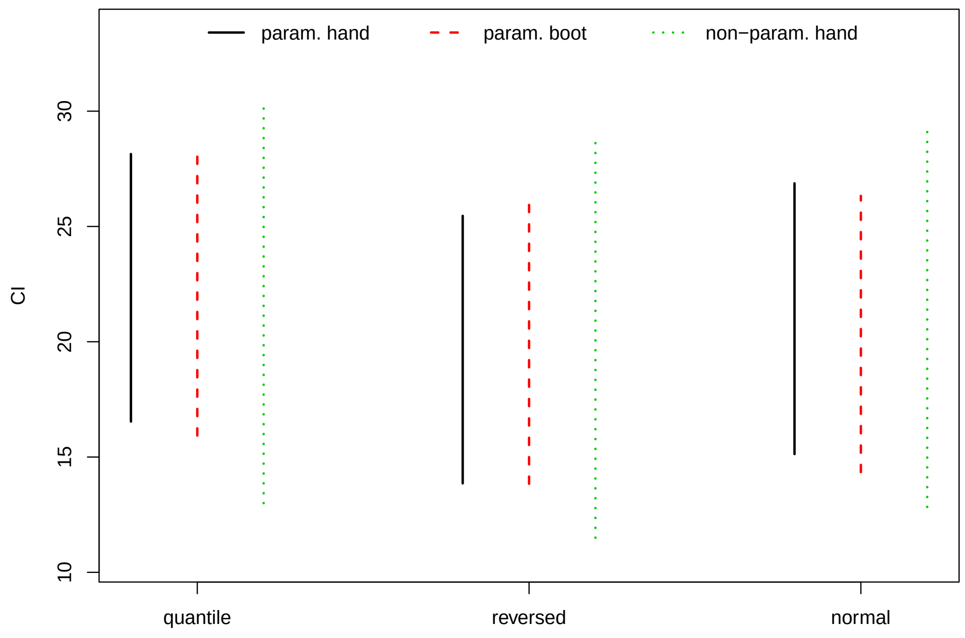
```
> # normal approx CI
> (CI.n2 <- c(boogg.75quant - qnorm(0.975) * sd(res2),
             boogg.75quant + qnorm(0.975) * sd(res2)))
```

```
[1] 12.83528 29.16472
```

As we would expect for such a small sample size, the confidence intervals for non-parametric bootstrap are wider.

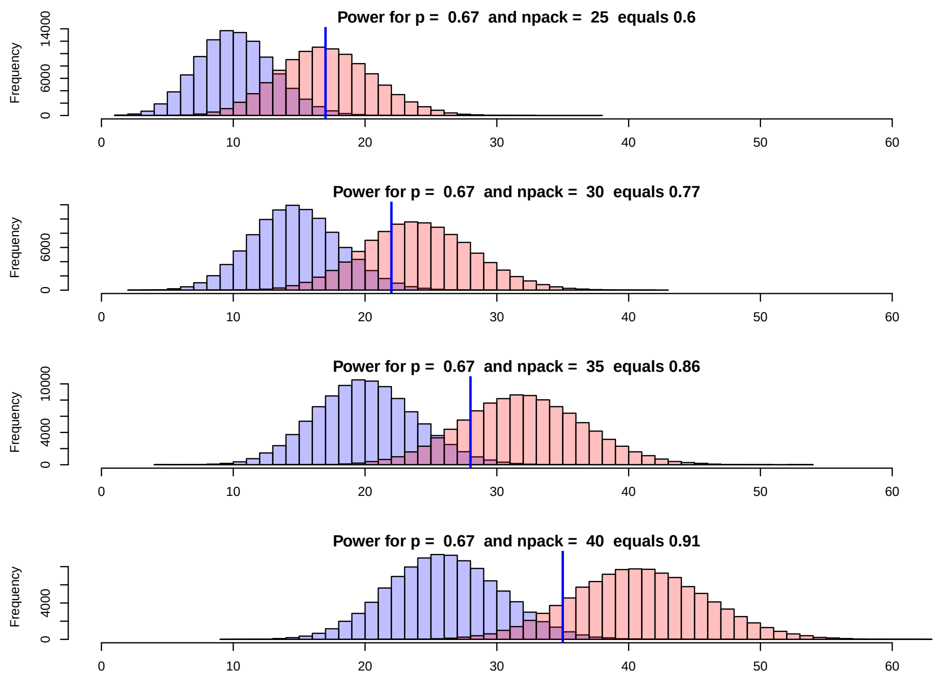
```
> # Plot of CIs as an overview
> CI.q1.boot <- res.boot.ci$percent[4:5]
> CI.r1.boot <- res.boot.ci$basic[4:5]
> CI.n1.boot <- res.boot.ci$normal[2:3]
> ylim.max <- max(c(CI.q1, CI.q1.boot, CI.q2, CI.r1, CI.r1.boot, CI.r2, CI.n1,
                    CI.n1.boot, CI.n2))
> ylim.min <- min(c(CI.q1, CI.q1.boot, CI.q2, CI.r1, CI.r1.boot, CI.r2, CI.n1,
                    CI.n1.boot, CI.n2))
> plot(x = NA, xlim =c(0, 0.6), ylim = c(ylim.min - 1, ylim.max + 3), ylab = "CI",
      xlab = "", xaxt='n')
> axis(side = 1, at = c(0.05, 0.3, 0.55),
      labels = c("quantile", "reversed", "normal"))
> legend("top", c("param. hand","param. boot","non-param. hand"), lty = 1:3,
        lwd = 2, col = 1:3, ncol = 3, bty ="n")
> # quantile
> lines(x = rep(0, times = 2), y = c(CI.q1[1], CI.q1[2]), lwd = 2)
> lines(x = rep(0.05, times = 2), y = c(CI.q1.boot[1], CI.q1.boot[2]), col = 2,
       lty = 2, lwd = 2)
> lines(x = rep(0.1, times = 2), y = c(CI.q2[1], CI.q2[2]), col = 3, lty = 3,
       lwd = 2)
```

```
> # reversed
> lines(x = rep(0.25, times = 2), y = c(CI.r1[1], CI.r1[2]), lwd = 2)
> lines(x = rep(0.3, times = 2), y = c(CI.r1.boot[1], CI.r1.boot[2]), col = 2,
       lty = 2, lwd = 2)
> lines(x = rep(0.35, times = 2), y = c(CI.r2[1], CI.r2[2]), col = 3, lty = 3,
       lwd = 2)
> # normal
> lines(x = rep(0.5, times = 2), y = c(CI.n1[1], CI.n1[2]), lwd = 2)
> lines(x = rep(0.55, times = 2), y = c(CI.n1.boot[1], CI.n1.boot[2]), col = 2,
       lty = 2, lwd = 2)
> lines(x = rep(0.6, times = 2), y = c(CI.n2[1], CI.n2[2]), col = 3, lty = 3,
       lwd = 2)
```



The bootstrap CIs based on parametric bootstrap (by hand and using the package `boot`) are similar, but obviously they do not have to be exactly the same since they are based on different bootstrap samples. The normal approximation CI based on parametric bootstrap using the package `boot` is shifted compared to the normal approximation CI calculated by hand because the function `boot.ci` corrects for the bias.

3.
```
> nr.cards <- 682     # number of unique cards
> nsimul <- 100000    # number of simulations
> # Function that simulates packaging of the cards.
> # Returns the number of duplicates in npack packs.
> simulate.duplicate.k <- function(k, npack){
   probs <- c(rep(5, k), rep(1, nr.cards - k))
   probs <- probs/sum(probs)
   res <- sample(1:nr.cards, npack*5, prob=probs, replace = TRUE)
   return(5 * npack - length(unique(res)))
 }
> set.seed(456)
> n.npack <- c(25, 30, 35, 40)
> resNULL <- matrix(data = NA, nrow = length(n.npack),ncol = nsimul)
> resALT <- matrix(data = NA, nrow = length(n.npack),ncol = nsimul)
> for (i in 1:length(n.npack)) {
   npack <- n.npack[i]
   for (k in 1:nsimul) {
     # Note that p = 0 corresponds to the H0
     resNULL[i, k] <- simulate.duplicate.k(k = 0, npack = npack)
     resALT[i, k] <- simulate.duplicate.k(k = 100, npack = npack)
   }
 }
> # Alternatively, one could use mapply or some other function.

> # Creates three plots
> par(mfrow = c(4, 1))
> res.max <- max(c(resNULL[i, ], resALT[i, ]))
> for (i in 1:length(n.npack)) {
   npack <- n.npack[i]
   # boundary of rejection region at alpha = 0.05
   rej <- quantile(resNULL[i, ], 0.95)+1
   # power given the rejection boundary
   power <- sum(resALT[i, ] >= rej) / nsimul

   # plot
   res.range <- range(c(resNULL[i, ], resALT[i, ]))
   p1 <- hist(resNULL[i, ], plot = FALSE,
              breaks = seq(from = res.range[1], to = res.range[2],
                           by = 1))
   p2 <- hist(resALT[i, ], plot = FALSE,
              breaks = seq(from = res.range[1], to = res.range[2],
                           by = 1))
   plot(p1, col = rgb(0, 0, 1, 1/4), xlim = c(0, res.max),
        ylim = c(0, max(c(p1$counts, p2$counts))),  xlab = "",
        main = paste("Power for p = ", round(2/3, 2), " and npack = ", npack,
                     " equals", round(power, 2)))
   plot(p2, col = rgb(1, 0, 0, 1/4), xlab = "", add = TRUE)
   abline(v = rej, col = 4, lwd = 2)
 }
```



We need 35 packs to have a power greater or equal than 80%. (Techically the true answer is 31 packs; but out of the candidates 25, 30, 35, 40 the first value for which power exceeds 80% is 35)