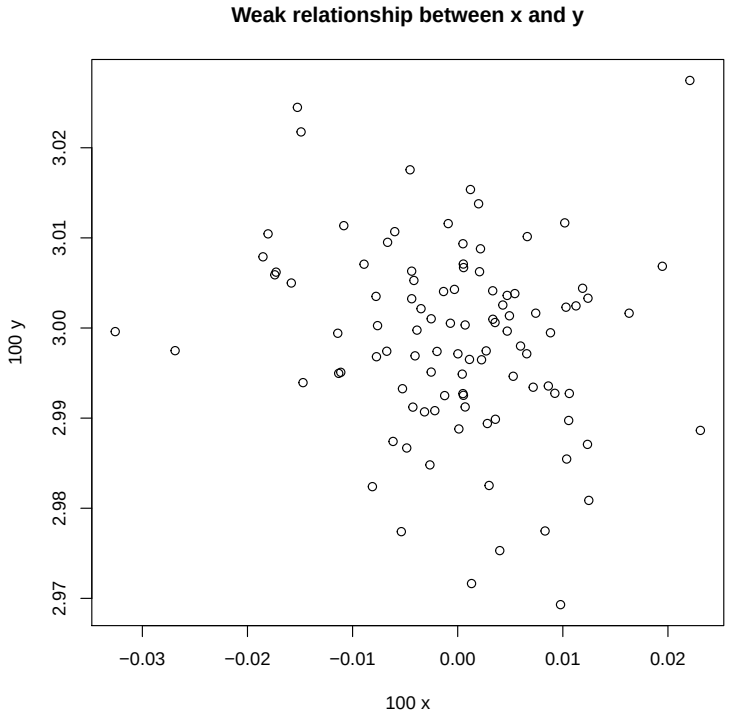
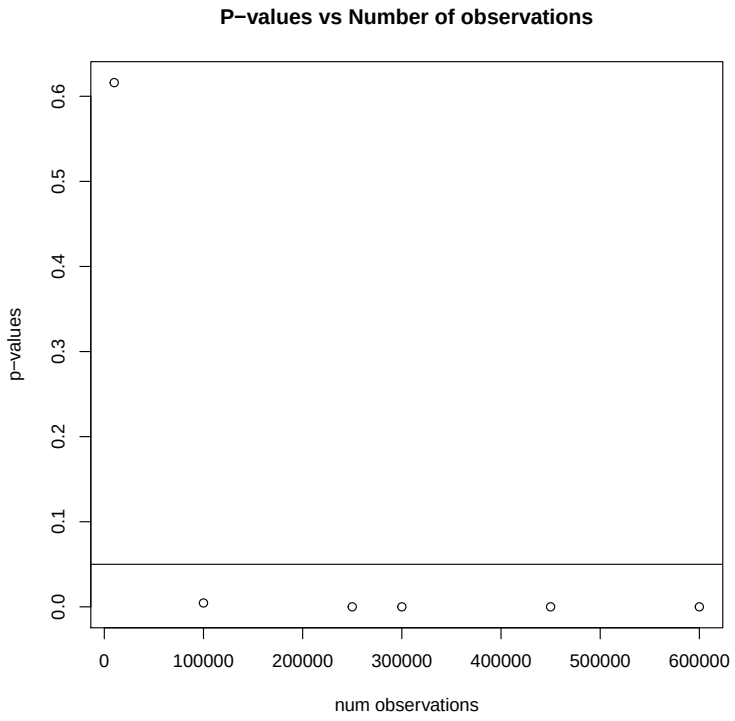
Problem 1

```r
set.seed(08544)
# N <- c(10000, 100000, 250000, 300000, 450000, 600000, 1000000, 1500000, 2000000, 2500000)
N <- c(10000, 100000, 250000, 300000, 450000, 600000)

ps <- c()
x <- c()
meanx <- c()
meany <- c()
y <- c()
for (i in 1:length(N)){
  e <- rnorm(N[i], 0, sd = 0.01)
  x <- rnorm(N[i], 0, sd = 0.01)
  y <- 3 + 0.01*x + e
  meanx <- c(meanx, mean(x))
  meany <- c(meany, mean(y))
  summary(lm(y ~ x))
  ps[i] <- summary(lm(y ~ x))$coefficients[2,4]
}

cor(x,y)
plot(x[1:100], y[1:100],
     main = "Weak relationship between x and y",
     xlab = "100 x",
     ylab = '100 y')
```



```r
#hist(y)
plot(N, ps,
     main = "P-values vs Number of observations",
     ylab = "p-values",
     xlab = "num observations"
)
abline(h = .05)
```




```r
#cor(x, y)
#N[ps < 0.05]
```

We set the parameters to $y = 3 + 0.01 * x + e$, for which y has a weak relationship with x. The correlation is 0.009. We then ran regressions on samples of increasing size, from 10000 to NA and plotted the p-values for each regression. As seen in the figure, the relationship is consistently statistically significant after 1000000 samples, which shows that as sample size increases, even weak relationships will be shown as significant.

Problem 2

```r
set.seed(08544)
N.p2 <- 1000
#x1 is asian
x1 <- rbinom(N.p2, size = 1, prob = .056)
#hist(x1)
#personality "personality"
x2 <- rnorm(N.p2, (5 - x1*3), 1.5)
#hist(x2)
#not get into harvard?
```

```
y <- plogis(x2 - 6)
#hist(y)

summary(lm(y ~ x2 + x1))
cor(x1, y)
```

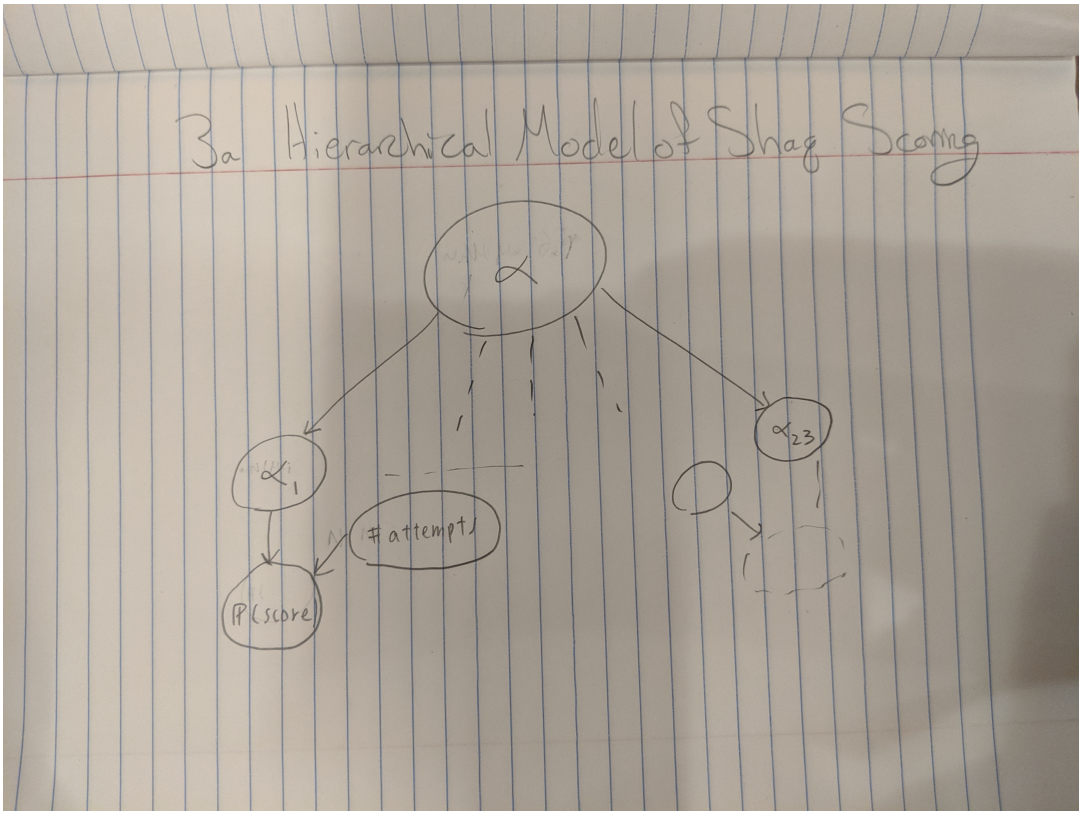
We completed Problem 2b prior to 2a, which is why the code is nearly identical. The correlation between x1 and y is -0.252, while the coefficient in the regression is 0.191, showing that the lurking variable x2 changes the sign.

```
set.seed(08544)
N.p2 <- 1000
#asian is asian
asian <- rbinom(N.p2, size = 1, prob = .056)
#hist(asian)
#personality "personality"
personality <- rnorm(N.p2, (5 - asian*3), 1.5)
#hist(personality)
#not get into harvard?
harvard <- plogis(personality - 6)
#hist(harvard)

summary(lm(harvard ~ personality + asian))
cor(asian, harvard)
```

We chose to comment on the Harvard Circut Court decision on admission of Asian American applicants. Here, x1 is binomially distributed to mark if the candidate is Asian (5.6% of the U.S. population is Asian). x2 is the "personality" score as described in the brief, which here is lowered if the applicant is Asian. y is the probability of acceptance to Harvard based on the personality score distribution, which is lower for Asian applicants.

Problem 3

Part a

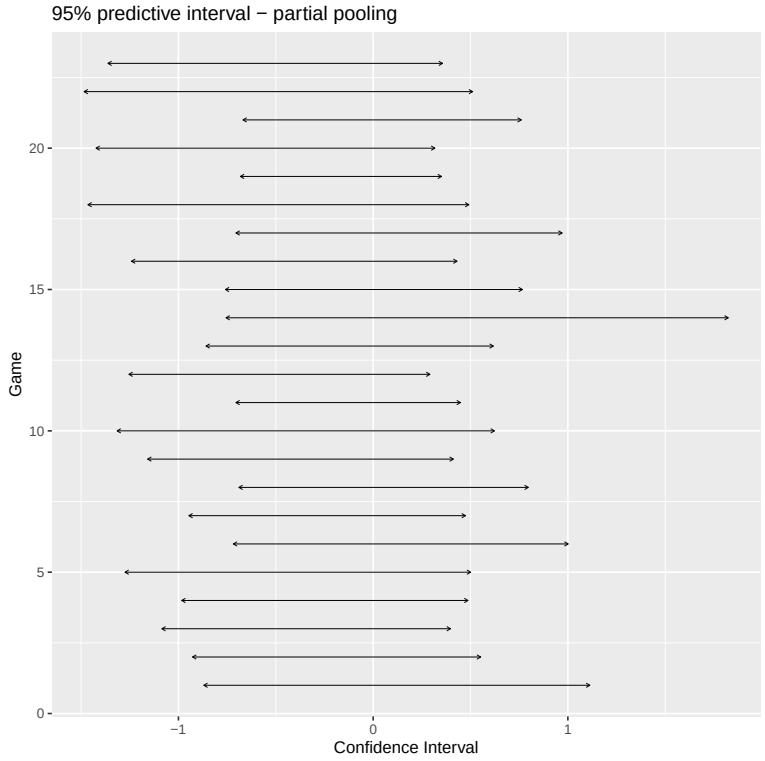


Part b

```
lines <-
"Game   Scored  N.Attempts
1   4   5
2   5   11
3   5   14
4   5   12
5   2   7
6   7   10
7   6   14
8   9   15
9   4   12
10  1   4
11  13  27
12  5   17
13  6   12
14  9   9
15  7   12
16  3   10
17  8   12
```

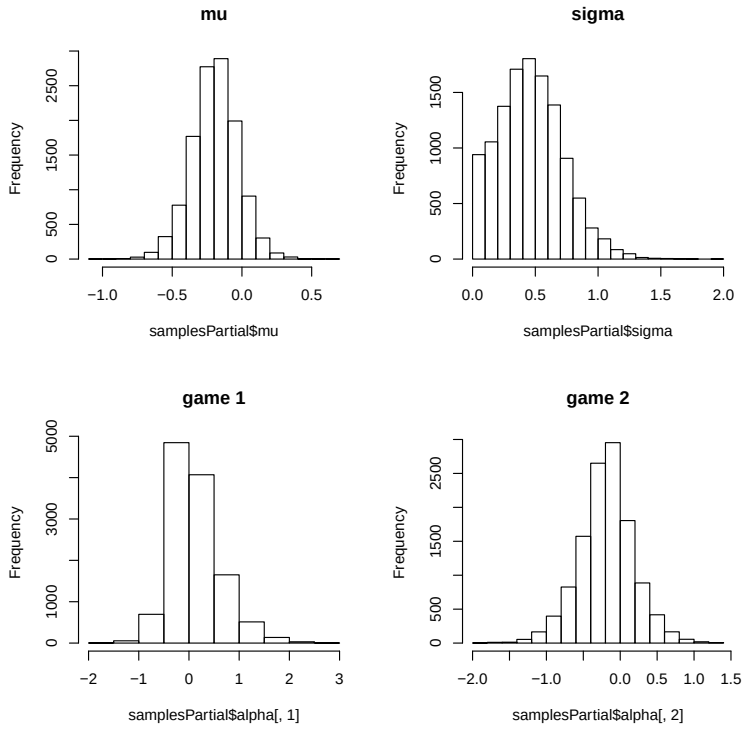
```
18  1   6
19  18  39
20  3   13
21  10  17
22  1   6
23  3   12"
con <- textConnection(lines)
shaq <- read.csv(con, sep="")
shaq

# partial pooling: each alpha affected by overall alpha
shaq_model_stan <- "
data{
  // data we supply
  int<lower=0> N; // game number, must be positive
  int scored[N]; // array of scores indexed by game
  int attempted[N]; // array of number of attempts indexed by games
}

parameters{
  real mu; // overall average skill level
  real<lower=0> sigma; // overall stdev of skill level, lower bounded by 0
  vector[N] alpha_norm; // array of 'zscores'
}

transformed parameters{
  real alpha[N]; // array of average skill level indexed by game
                 // produced using alpha_norms
  for(n in 1:N)
    alpha[n] = mu + sigma * alpha_norm[n];
}

model{
  alpha_norm ~ normal(0, 1);
  scored ~ binomial(attempted, inv_logit(alpha)); // scored modeled using num attempted & 0 - 1 of alpha
}"

adaptSteps = 1000             # Number of steps to "tune" the samplers.
burnInSteps = 5000            # Number of steps to "burn-in" the samplers.
nChains = 3                   # Number of chains to run.
numSavedSteps=12000           # Total number of steps in chains to save.
thinSteps=10                  # Number of steps to "thin" (1=keep every step).

shaq_model <- stan_model(model_code = shaq_model_stan, model_name = "shaq_model")
shaq_fit <- sampling(object=shaq_model,
                     data = list(N=nrow(shaq), scored=shaq$Scored, attempted = shaq$N.Attempts),
                     chains = nChains,
                     iter = (ceiling(numSavedSteps/nChains)*thinSteps
                             + burnInSteps),
                     warmup = burnInSteps,
                     thin = thinSteps,
                     init = "random")
```

```r
samplesPartial <- extract(shaq_fit)

# posterior distribution for each parameter
# hist(samplesPartial£mu) # prob distribution of mu
# hist(samplesPartial£sigma) # prob distribution of sigma
# hist(samplesPartial£alpha[, 1]) # prob distribution of alpha_1
# hist(samplesPartial£alpha[, 2]) # prob distribution of alpha_2
# 95% of data lies within 2 stdev of the mean 0 -->
good <- plogis(mean(samplesPartial$mu) + 2 * mean(samplesPartial$sigma))
bad <- plogis(mean(samplesPartial$mu) - 2 * mean(samplesPartial$sigma))
good
bad

N <- 23
conf_data <- data.frame(games = 1:N, mean = NA, upper = NA, lower = NA)

for (i in 1:N){
  conf_data[i, "mean"] = mean(samplesPartial$alpha[, i])
  conf_data[i, "upper"] = mean(samplesPartial$alpha[, i]) + 2 * sd(samplesPartial$alpha[, i])
  conf_data[i, "lower"] = mean(samplesPartial$alpha[, i]) - 2 * sd(samplesPartial$alpha[, i])
}


ggplot(conf_data) +
  geom_segment(aes(x=lower,y=games,xend=upper,yend=games),
               arrow=arrow(length=unit(0.1,"cm"),
                           ends='both'),size=.1) +
  labs(x = "Confidence Interval", y = "Game",
       title = "95% predictive interval - partial pooling")
```



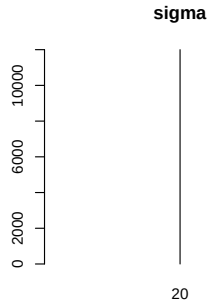
```
#ggplot(conf_data, aes(x = games, y = mean)) +
 # geom_errorbar(aes(ymin=lower, ymax=upper))
```

```
par(mfrow=c(2, 2))
hist(samplesPartial$mu, main = "mu")
hist(samplesPartial$sigma, main = "sigma")
hist(samplesPartial$alpha[, 1], main =  "game 1")
hist(samplesPartial$alpha[, 2], main = "game 2")
```




```
# plot an uniform dist of 0 for sigma
```

```
# plot 20 for sigma
sigma <- rep(0, 12000) # it looked weird

par(mfrow=c(2, 2))
hist(samplesNo$mu)

## Error in hist(samplesNo$mu):  object 'samplesNo' not found

barplot(length(sigma), sigma, main = "sigma", names.arg = "20")
hist(samplesNo$alpha[, 1])

## Error in hist(samplesNo$alpha[, 1]):  object 'samplesNo' not found

hist(samplesNo$alpha[, 2]) # they're all different

## Error in hist(samplesNo$alpha[, 2]):  object 'samplesNo' not found
```


sigma
0
2000
6000
10000
20