Homework 3

Wendy Liang

Disclaimer

A few things to keep in mind : 1) Use set.seed() to make sure that the document produces the same random simulation as when you ran the code. 2) Use refresh=0 for any stan_glm() or stan-based model. lm() or non-stan models don't need this! 3) You can type outside of the r chunks and make new r chunks where it's convenient. Make sure it's clear which questions you're answering. 4) Even if you're not too confident, please try giving an answer to the text responses! 5) Please don't print data in the document unless the question asks. It's good for you to do it to look at the data, but not as good for someone trying to read the document later on. 6) Check your document before submitting! Please put your name where "name" is by the author!

4.1 Comparison of proportions

A randomized experiment is performed within a survey. 1000 people are contacted. Half the people contacted are promised a $5 incentive to participate, and half are not promised an incentive. The result is a 50% response rate among the treated group and 40% response rate among the control group. Give an estimate and standard error of the average treatment effect.

```
se_treat = sqrt(0.5*0.5/500)
se_control = sqrt(0.4*0.6/500)
se=sqrt(se_treat^2+se_control^2)
se
```

```
## [1] 0.03130495
```

The estimate of the average treatment effect is the gap between the two groups.

The estimate is $se_{treat} - se_{control} = 0.1$

The standard error is $\sqrt{se_{treat}^2 + se_{control}^2} = 0.0313$

4.2 Choosing sample size

You are designing a survey to estimate the gender gap: the difference in support for a candidate among men and women. Assuming the respondents are a simple random sample of the voting population, how many people do you need to poll so that the standard error is less than 5 percentage points?

assume the poll is N

$se_{woman} = se_{man} = \sqrt{0.5 \times 0.5 \times 2 \div N}$

$se = \sqrt{se_{man}^2 + se_{woman}^2} = 0.05$

So $N = \frac{1}{se^2} = 400$

4.4 Designing an experiment

You want to gather data to determine which of two students is a better basketball shooter. You plan to have each student take N shots and then compare their shooting percentages. Roughly how large does N have to be for you to have a good chance of distinguishing a 30% shooter from a 40% shooter?

Assuming the goal is 80% to distinguish the p1=30% and p2=40% properties each with size N

$se_1 = \sqrt{0.3 \times 0.7 \div N}$

$se_2 = \sqrt{0.4 \times 0.6 \div N}$

$se = \sqrt{se_1^2 + se_2^2} = 0.8$

So $N = \frac{2.8}{0.4-0.3}^2 = 784$

More precisely $N = \frac{2.8}{0.4-0.3}^2 \times 2(0.4 \times 0.6 + 0.3 \times 0.7) = 706$

4.6 Hypothesis testing

The following are the proportions of girl births in Vienna for each month in Girl births 1908 and 1909 (out of an average of 3900 births per month):

```
birthdata = c(.4777,.4875,.4859,.4754,.4874,.4864,.4813,.4787,.4895,.4797,.4876,.4859,
              .4857,.4907,.5010,.4903,.4860,.4911,.4871,.4725,.4822,.4870,.4823,.4973)
```

The data are in the folder Girls. These proportions were used by von Mises (1957) to support a claim that that the sex ratios were less variable than would be expected under the binomial distribution. We think von Mises was mistaken in that he did not account for the possibility that this discrepancy could arise just by chance.

(a) Compute the standard deviation of these proportions and compare to the standard deviation that would be expected if the sexes of babies were independently decided with a constant probability over the 24-month period.

```
sd_obs=sd(birthdata)
p_obs=mean(birthdata)
#print(p_obs)
sd_exp=sqrt(p_obs*(1-p_obs)/3900)
diffrence = sd_exp-sd_obs
print(sd_obs)
```

```
## [1] 0.006409724
```

```
print(sd_exp)
```

```
## [1] 0.008003121
```

```
print(diffrence)
```

```
## [1] 0.001593397
```

(b) The observed standard deviation of the 24 proportions will not be identical to its theoretical expectation. In this case, is this difference small enough to be explained by random variation? Under the randomness model, the actual variance should have a distribution with expected value equal to the theoretical variance, and proportional to a chi-square random variable with 23 degrees of freedom; see page 53.

confidence interval for the population variance equals to $\frac{(n-1)s^2}{\chi^2_{\alpha/2}} \leq \sigma^2 \leq \frac{(n-1)s^2}{\chi^2_{1-\alpha/2}}$.

```
#alpha=0.05
df = 23
upper = qchisq(0.025, 23)
lower = qchisq(0.975, 23)

sd_upper = sqrt(df*sd_obs^2/upper)
sd_lower = sqrt(df*sd_obs^2/lower)

print(sd_lower)
```

```
## [1] 0.004981725
```

```
print(sd_upper)
```

```
## [1] 0.008991309
```

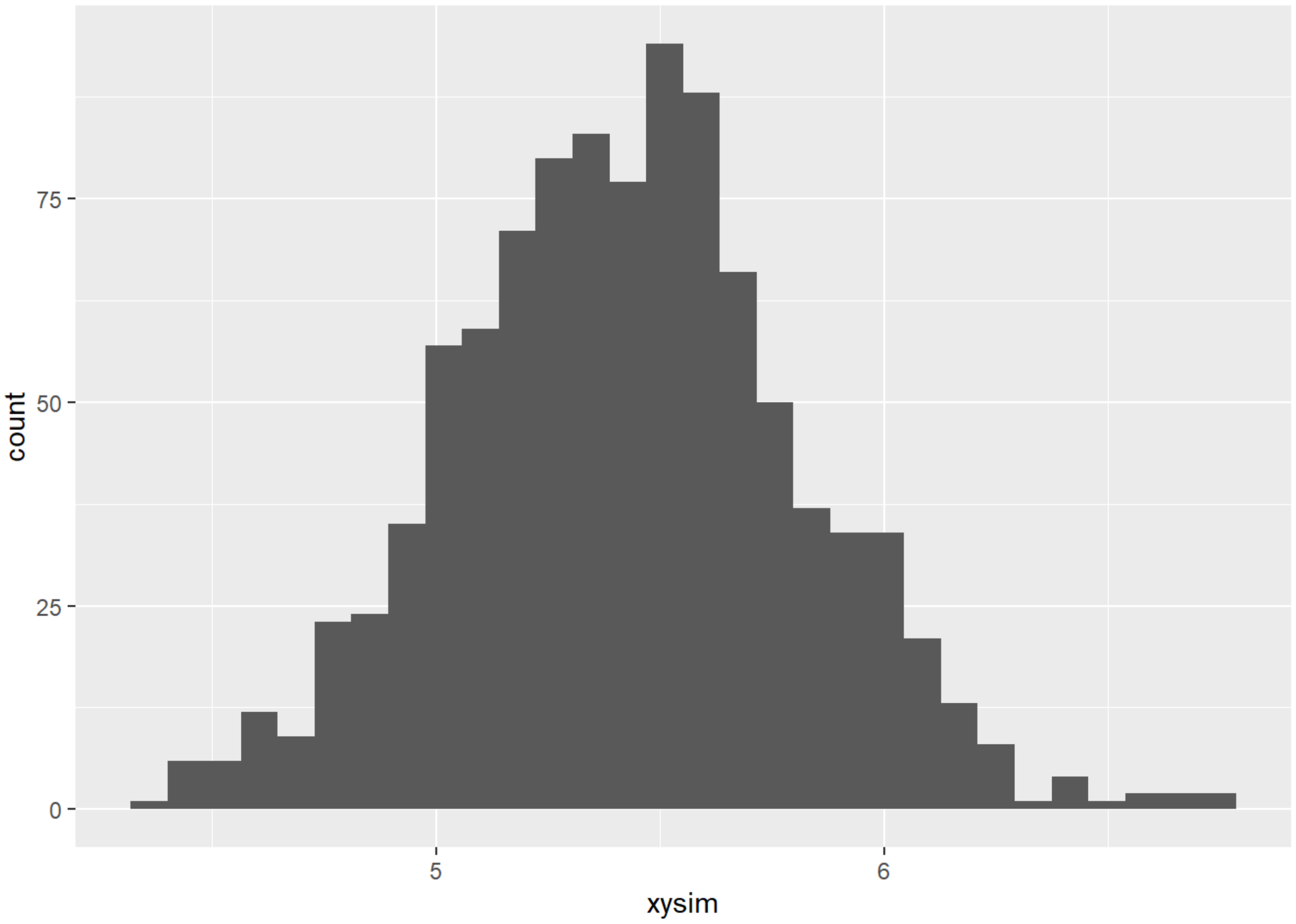
As a result, the 95% confidence interval for the population variance is [0.0050 , 0.0090]. Since the observed standard deviation of 0.0064 is in this range, we can conclude that the observed difference is not signficant at $\alpha = 0.05$.

5.5 Distribution of averages and differences

The heights of men in the United States are approximately normally distributed with mean 69.1 inches and standard deviation 2.9 inches. The heights of women are approximately normally distributed with mean 63.7 inches and standard deviation 2.7 inches. Let x be the average height of 100 randomly sampled men, and y be the average height of 100 randomly sampled women. In R, create 1000 simulations of x - y and plot their histogram. Using the simulations, compute the mean and standard deviation of the distribution of x - y and compare to their exact values.

```
set.seed(12)
xy=data.frame(xysim=replicate(1000,mean(rnorm(100,69.1,2.9))-mean(rnorm(100,63.7,2.7))))
ggplot(xy,aes(x=xysim))+geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
mean=mean(xy$xysim)
mean_real=69.1-63.7
sd=sd(xy$xysim)
sd_real=2.9-2.7

print(c(mean,mean_real))
```

```
## [1] 5.422998 5.400000
```

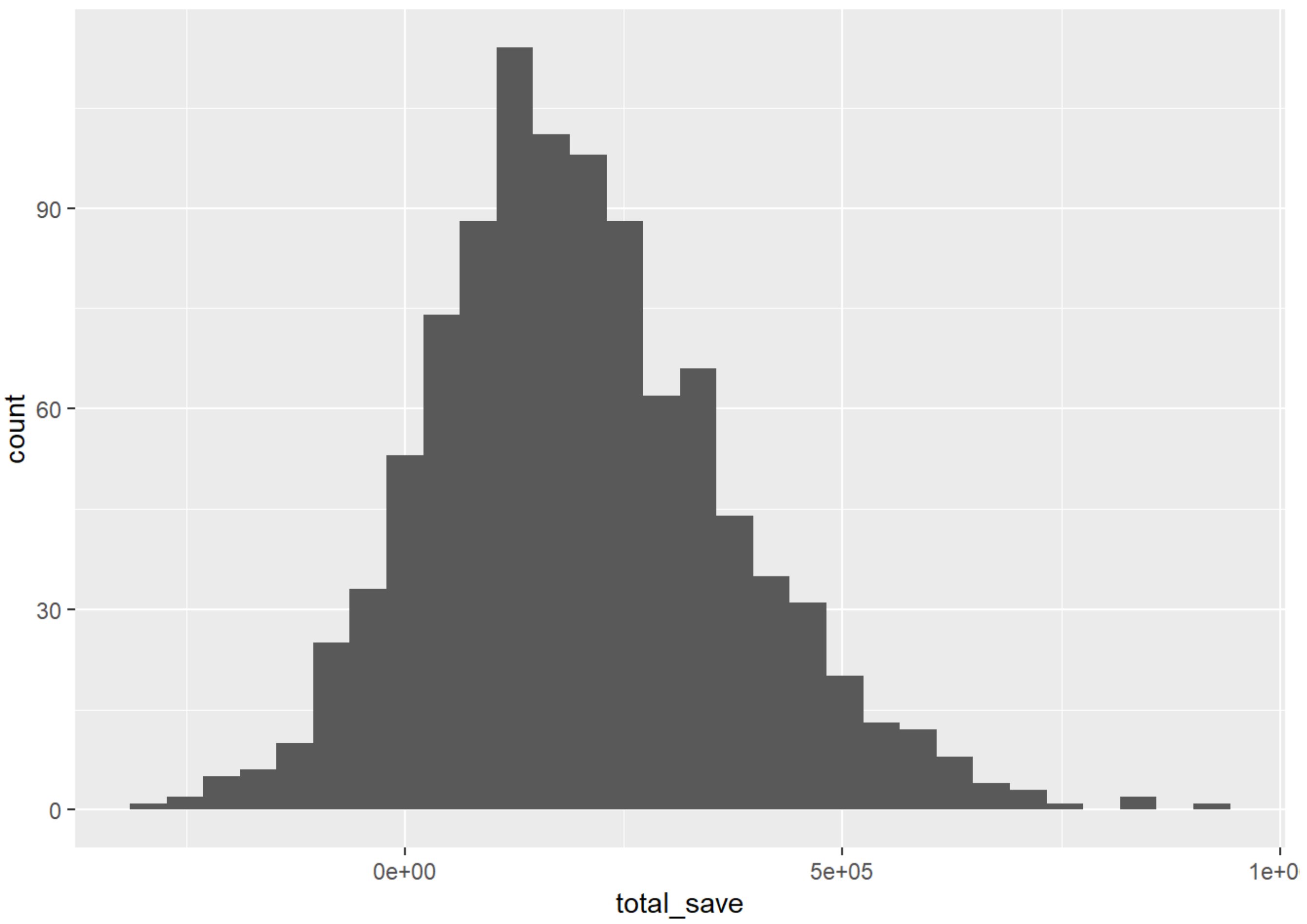
```
print(c(sd,sd_real))
```

```
## [1] 0.3852651 0.2000000
```

5.6 Propagation of uncertainty:

We use a highly idealized setting to illustrate the use of simulations in combining uncertainties. Suppose a company changes its technology for widget production, and a study estimates the cost savings at 5 dollars per unit, but with a standard error of 4 dollars. Furthermore, a forecast estimates the size of the market (that is, the number of widgets that will be sold) at 40 000, with a standard error of 10 000. Assuming these two sources of uncertainty are independent, use simulation to estimate the total amount of money saved by the new product (that is, savings per unit, multiplied by size of the market).

```
set.seed(12)
total=data.frame(total_save=replicate(1000,rnorm(1,5,4)*rnorm(1,40000,10000)))
ggplot(total,aes(x=total_save))+geom_histogram()
```



```
mean(total$total_save)
```

```
## [1] 200878.1
```

The estimate of the total amount of money saving is 208453.3

5.8 Coverage of confidence intervals:

On page 15 there is a discussion of an experimental study of an education-related intervention in Jamaica, in which the point estimate of the treatment effect, on the log scale, was 0.35 with a standard error of 0.17. Suppose the true effect is 0.10—this seems more realistic than the point estimate of 0.35—so that the treatment on average would increase earnings by 0.10 on the log scale. Use simulation to study the statistical properties of this experiment, assuming the standard error is 0.17.

(a) Simulate 1000 independent replications of the experiment assuming that the point estimate is normally distributed with mean 0.10 and standard deviation 0.17.

```
set.seed(12)
x_hat=rnorm(1000,0.10,0.17)
```

(b) For each replication, compute the 95% confidence interval. Check how many of these intervals include the true parameter value.

```
sd=0.17
alpha=0.05
n=1000
lower=rep(NA,1000)
upper=rep(NA,1000)
for(i in 1:n){
  lower[i]=x_hat[i]-sd*qnorm(1-alpha/2)/sqrt(1)
  upper[i]=x_hat[i]+sd*qnorm(1-alpha/2)/sqrt(1)
}
ci95=data.frame(lower,upper)
indicate=ifelse(ci95$lower<0.1&ci95$upper>0.1,1,0)
number=sum(indicate)
number
```

```
## [1] 954
```

(c) Compute the average and standard deviation of the 1000 point estimates; these represent the mean and standard deviation of the sampling distribution of the estimated treatment effect.

```
print(c(mean=mean(x_hat),sd=sd(x_hat)))
```

```
##       mean         sd
## 0.09550579 0.16308967
```

5.9 Coverage of confidence intervals after selection on statistical significance:

Take your 1000 simulations from Exercise 5.8, and select just the ones where the estimate is statistically significantly different from zero. Compute the average and standard deviation of the selected point estimates. Compare these to the result from Exercise 5.8.

```
for(i in 1:1000){
if(ci95$lower[i]<0.1&ci95$upper[i]>0.1){
  x_hat[i]=x_hat[i]
}
  else{
    x_hat[i]=NA
  }
}
#remove the NA, leave the significant ci
print(c(mean=mean(x_hat,na.rm=T),sd=sd(x_hat,na.rm=T)))
```

```
##       mean         sd
## 0.09688432 0.14274923
```

9.8 Simulation for decision analysis:

An experiment is performed to measure the efficacy of a television advertising program. The result is an estimate that each minute spent on a national advertising program will increase sales by 500,000 dollars, and this estimate has a standard error of 200000 dollars. Assume the uncertainty in the treatment effect can be approximated by a normal distribution. Suppose ads cost 300000 dollars per minute. What is the expected net gain for purchasing 20 minutes of ads? What is the probability that the net gain is negative?

```
set.seed(12)
n=1000
gain=rep(0,n)
for(i in 1:n){
gain[i]=sum(rnorm(20,500000,200000))-300000*20
}
gain_hat=mean(gain)
indicate=ifelse(gain<0,1,0)
prob=sum(indicate)/n
print(c(gain_hat=gain_hat,probability=prob))
```

```
##    gain_hat probability
##     4010453           0
```

10.3 Checking statistical significance:

In this exercise and the next, you will simulate two variables that are statistically independent of each other to see what happens when we run a regression to predict one from the other. Generate 1000 data points from a normal distribution with mean 0 and standard deviation 1 by typing var1 <- rnorm(1000,0,1) in R. Generate another variable in the same way (call it var2). Run a regression of one variable on the other. Is the slope coefficient “statistically significant”? We do not recommend summarizing regressions in this way, but it can be useful to understand how this works, given that others will do so.

```
set.seed(12)
var1 = rnorm(1000,0,1)
var2 = rnorm(1000,0,1)
fit=lm(var1~var2)
summary(fit)
```

```
## 
## Call:
## lm(formula = var1 ~ var2)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -3.0083 -0.6108 -0.0147  0.5861  3.1326 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.026147   0.030369  -0.861    0.389
## var2        -0.008292   0.029613  -0.280    0.780
## 
## Residual standard error: 0.9598 on 998 degrees of freedom
## Multiple R-squared:  7.855e-05,  Adjusted R-squared:  -0.0009234 
## F-statistic: 0.0784 on 1 and 998 DF,  p-value: 0.7795
```

This slope is not significant at 0.05 level, since p value > 0.05.

10.4 Simulation study of statistical significance:

Continuing the previous exercise, run a simulation repeating this process 100 times. This can be done using a loop. From each simulation, save the z-score (the estimated coefficient of var1 divided by its standard error). If the absolute value of the z-score exceeds 2, the estimate is "statistically significant." To perform this computation, we start by creating an empty vector of z-scores filled with missing values (NAs). Another approach is to start with z_scores <- numeric(length=100), which would set up a vector of zeroes. In general, however, we prefer to initialize with NAs, because then when there is a bug in the code, it sometimes shows up as NAs in the final results, alerting us to the problem.

How many of these 100 z-scores exceed 2 in absolute value, thus achieving the conventional level of statistical significance?

Here is code to perform the simulation:

This chunk will have eval=FALSE. If you want it to run, please copy it to a new chunk, or remove eval=FALSE!

```
set.seed(12)
z_scores <- rep(NA,100)
for(k in 1:100) {
  var1 <- rnorm(1000,0,1)
  var2 <- rnorm(1000,0,1)
  fake <- data.frame(var1,var2)
  fit <- stan_glm(var2 ~ var1,data=fake,refresh=0)
  z_scores[k] <- coef(fit)[2] / se(fit)[2]
}
sum(z_scores>1.96)
```

```
## [1] 2
```

11.3 Coverage of confidence intervals:

Consider the following procedure:

- Set n = 100 and draw n continuous values xi uniformly distributed between 0 and 10. Then simulate data from the model yi = a + bxi + errori, for i = 1,…, n, with a = 2, b = 3, and independent errors from a normal distribution.
- Regress y on x. Look at the median and mad sd of b. Check to see if the interval formed by the median ± 2 mad sd includes the true value, b = 3.

```
set.seed(12)
x=runif(100,0,10)
y=2+3*x+rnorm(100,0,1)
fit=stan_glm(y~x,refresh=0)
lower=coef(fit)[2]-2*fit$ses[2]
upper=coef(fit)[2]+2*fit$ses[2]
interval=data.frame(lower,upper)
print(interval)
```

```
##      lower   upper
## x 2.985809 3.12544
```

- Repeat the above two steps 1000 times.

This process is too slow to run, my computer cannot run the result successfully.

```
set.seed(12)
interval=matrix(NA,1000,2)
for(i in 1:1000){
x=runif(100,0,10)
y=2+3*x+rnorm(100,0,1)
fit=stan_glm(y~x,refresh=0)
lower=coef(fit)[2]-2*fit$ses[2]
upper=coef(fit)[2]+2*fit$ses[2]
interval[i,1]=lower
interval[i,2]=upper
}
#data.frame(interval)
indicate=ifelse(interval$lower<3&interval$upper>3,1,0)
sum(indicate)
```

(a) True or false: the interval should contain the true value approximately 950 times. Explain your answer

TRUE.

(b) Same as above, except the error distribution is bimodal, not normal. True or false: the interval should contain the true value approximately 950 times. Explain your answer.

Repeat the following procedure 1000 times and we can get the result. The only difference with (a) is the error = rbinom(100,0,1).

```
set.seed(12)
x=runif(100,0,10)
y=2+3*x+rbinom(100,0,1)
fit_b=stan_glm(y~x,refresh=0)
lowerb=coef(fit_b)[2]-2*fit$ses[2]
upperb=coef(fit_b)[2]+2*fit$ses[2]
interval_b=data.frame(lowerb,upperb)
print(interval_b)
```