# PS C236A / Stat C239A
# Problem Set 2 - Solutions

1: a) Under the null hypothesis, there are $2^4 = 16$ possible ways that the Lady can label cups as tea-first or milk-first, where each possible assignment is equally likely. Five of these possibilities have 1 or fewer mistake. Thus, the $p$-value for the hypothesis test is 5/16.

b) Under the null hypothesis, there are $\binom{8}{4} = 70$ possible assignments of tea-first and milk-first cups, with each assignment equally likely. Of these assignments, 1 of these assignments has no mistakes, and $\binom{4}{3}\binom{4}{1} = 16$ of these assignments have one mistake. Thus, the $p$-value is 17/70. This $p$-value is smaller than the one in part a).

The probability in part a) (a "two-margins-fixed model") assumes a different model of randomness than the probability in part b) (a "one-margin-fixed model"). If you could control the random process in which someone without the ability to discern between milk-first and tea-first cups labels these cups, you would prefer the model in part b); you have more power to detect if the Lady actually has knowledge of which cups were tea-first and which cups were milk-first.

However, if you assume the model in part b) when the model in part a) is true, the $p$-values you compute may be much smaller than what should be observed; the probability of performing a Type I Error may be larger the stated significance level $\alpha$. The moral of the story: when performing a statistical hypothesis test, it is important to know exactly how randomness is generated. Assuming the wrong model may lead to inaccurate results.

c) You would prefer to pour tea first when the coin is heads. If you have a large number of milk-first cups in your trials, it will be hard to discern whether the Lady actually can tell the difference between milk-first and tea-first cups, or whether the Lady is guessing milk-first many times.

For example, suppose that the Lady guesses all cups correctly. If there was, by chance, two milk-first cups and 6 tea-first cups, the $p$-value under this null (assuming that the test statistic is the number of correct cups) would be

$$(2/3)^2(1/3)^6 \approx .0006.$$

If there were six milk-first cups and two tea-first cups, the $p$-value would be

$$(1/3)^2(2/3)^6 \approx .01.$$

That is, you have a much greater power for detecting knowledge of tea-first and milk-first cups against this null if there were more tea-first cups than milk-first cups.

d) WLOG: suppose the true assignment of cups is

T M M M M M M M

where T denotes a tea-first cup and M denotes a milk-first cup. There are

$$\sum_{k=2}^{6} \binom{8}{k} = 238$$

possible treatment assignments; under the null hypothesis, all of these assignments are equally likely.

Now we count the number of possible permutations such that there are two or fewer mistakes. Since the Lady labels at least two cups as "tea-first," it follows that the Lady must correctly identify the sole tea-first cup in the population. There are 7 permutations that correctly identify the tea-first cup and incorrectly label one of the milk-first cups as tea-first. There are $\binom{7}{2} = 21$ permutations that correctly identify the tea-first cup and incorrectly label two of the milk-first cups as tea-first. Thus, the $p$-value is $28/238 = .118$.

Note: This problem is related to re-randomization, where a randomized treatment assignment is only applied if it leads to sufficiently balanced pre-treatment covariates. By chance, a random treatment assignment might have bad covariate balance; if treatment assignment is re-randomized until covariate balance is acceptable, permutation inference should only consider those permuted treatment assignments that have the required level of covariate balance.

2: a) Recall the average treatment effect can be decomposed as follows:

$$E[\delta] = \{\pi E[Y_1|D=1] + (1-\pi)E[Y_1|D=0]\} - \{\pi E[Y_0|D=1] + (1-\pi)E[Y_0|D=0]\}$$

The largest possible ATE estimate is:

$$E[\delta] = \{0.5 \times 0.7 + 0.5 \times 1\} - \{0.5 \times 0 + 0.5 \times 0.5\} = 0.6$$

And the smallest possible ATE estimate is:

$$E[\delta] = \{0.5 \times 0.7 + 0.5 \times 0\} - \{0.5 \times 1 + 0.5 \times 0.5\} = -0.4$$

Thus, the ATE will be *bounded* between 0.6 and -0.4. And the difference is 1. Without further assumptions, this difference will always be 1, regardless of the observed values of the outcomes for the treated and control units.

b) Making no additional assumptions except that Catholic does not prevent any students from graduating, the new lower bound will be:

$$E[\delta] = \{0.5 \times 0.7 + 0.5 \times 0.5\} - \{0.5 \times 0.7 + 0.5 \times 0.5\} = 0$$

The upper bound will remain the same at 0.6. The difference between the bounds then will be 0.6.

3: Recall the data given in the table:

Table 1: Catholic School Graduation Data

| Unit | $Y$ | $T$ | $X$ | $U_{true}$ | $U_{obs}$ | $\pi_i(\gamma, \beta, X_i, U_i)$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1.37 | 1 | 1 | 0.9347 |
| 2 | 0 | 1 | 0.16 | 1 | - | 0.7023 |
| 3 | 0 | 0 | 0.51 | 0 | 0 | 0.6813 |
| 4 | 1 | 1 | 0.99 | 1 | - | 0.8904 |
| 5 | 1 | 0 | 1.53 | 1 | 1 | 0.9478 |
| 6 | 1 | 0 | -0.46 | 0 | - | 0.3351 |

a) Matching without replacemant results in the matches in Table 2. The $t_{McNemar} = 2$ (or 1 by excluding concordant pairs), and the $ATT = 0$.

Table 2: Matching without replacement

| Treated | Control | $d^2$ | $Y_1 - Y_0$ | $U_1 - U_0$ | $\frac{\pi_i/(1-\pi_i)}{\pi_j/(1-\pi_j)}$ |
|---|---|---|---|---|---|
| 1 | 5 | 0.0256 | 1 - 1 | 1 - 1 | 0.7883 |
| 2 | 6 | 0.3844 | 0 - 1 | 1 - 0 | 4.6809 |
| 4 | 3 | 0.2304 | 1 - 0 | 1 - 0 | 3.8003 |
| – | – | $\Sigma = 0.6404$ | $ATT = 0$ | | |

Table 3: Matching with replacement

| Treated | Control | $d^2$ | $Y_1 - Y_0$ | $U_1 - U_0$ | $\frac{\pi_i/(1-\pi_i)}{\pi_j/(1-\pi_j)}$ |
|---|---|---|---|---|---|
| 1 | 5 | 0.0256 | 1 - 1 | 1 - 1 | 0.7883 |
| 2 | 3 | 0.1225 | 0 - 0 | ? - 0 | 1.1035 |
| 4 | 3 | 0.2304 | 1 - 0 | ? - 0 | 3.8003 |
| – | – | $\Sigma = 0.3785$ | $ATT = 1/3$ | | |

b) Since we assume after matching $\pi_i = \pi_j$ for each $\{i, j\}$ in $s$, this means that each permutation of treatment *within* matched pairs is equally likely, with $m_s$ fixed to be 1. So we count the number of permutations where $t(T, r) \geq 2$ (utilizing all pairs), and divide this nuber by the total number of permutations, $|\Omega| = 2^3$. There are 6 permutations with $t(T, r) \geq 2$, so $p = 3/4$. [Note, you should recover the same probability if you exclude concordant pairs.]

c) The probability of treatment is not equal across units in any of the matched pairs. The strata with the largest difference is strata 2, with a ratio of treatment odds being 4.6809. To calculate the $p$-value under the null hypothesis of no effect we must take into account the probabilty of each permutation, which is no longer $1/|\Omega|$. We can show that each permutation probability is:

$$Pr(T = t|m) = \prod_{s=1}^{S} \pi_{si}^{T_{si}}(1 - \pi_{si})^{(1-T_{si})} \times \pi_{sj}^{T_{sj}}(1 - \pi_{sj})^{(1-T_{sj})}$$

where $T_{si} + T_{sj} = 1$. Now we have to calculate each the probability of each permutation $T \in \Omega$.

$$Pr\begin{bmatrix}1&0\\1&0\\1&0\end{bmatrix} = 0.00647 \qquad Pr\begin{bmatrix}0&1\\1&0\\1&0\end{bmatrix} = 0.00820 \qquad Pr\begin{bmatrix}1&0\\0&1\\1&0\end{bmatrix} = 0.00138 \qquad Pr\begin{bmatrix}1&0\\1&0\\0&1\end{bmatrix} = 0.00170$$

$$Pr\begin{bmatrix}0&1\\0&1\\1&0\end{bmatrix} = 0.00175 \qquad Pr\begin{bmatrix}1&0\\0&1\\0&1\end{bmatrix} = 0.00036 \qquad Pr\begin{bmatrix}0&1\\1&0\\0&1\end{bmatrix} = 0.00216 \qquad Pr\begin{bmatrix}0&1\\0&1\\0&1\end{bmatrix} = 0.00046$$

$$t\begin{bmatrix}1&0\\1&0\\1&0\end{bmatrix} = 2 \qquad t\begin{bmatrix}0&1\\1&0\\1&0\end{bmatrix} = 2 \qquad t\begin{bmatrix}1&0\\0&1\\1&0\end{bmatrix} = 3 \qquad t\begin{bmatrix}1&0\\1&0\\0&1\end{bmatrix} = 1$$

$$t\begin{bmatrix}0&1\\0&1\\1&0\end{bmatrix} = 3 \qquad t\begin{bmatrix}1&0\\0&1\\0&1\end{bmatrix} = 2 \qquad t\begin{bmatrix}0&1\\1&0\\0&1\end{bmatrix} = 1 \qquad t\begin{bmatrix}0&1\\0&1\\0&1\end{bmatrix} = 2$$

So:

$$Pr\{t(T, r) \geq \tilde{t}\} = \frac{(0.00647 + 0.00820 + 0.00138 + 0.00175 + 0.00036 + 0.00046)}{(0.00647 + 0.00820 + 0.00138 + 0.00170 + 0.00175 + 0.00036 + 0.00216 + 0.00046)}$$
$$= 0.828$$

This $p$-value is different than that calculated above since the probability of some permutations are more likely than others, in this case, those that are more likely to have treatment probability correspond to a "success" on the graduation outcome.

d) Matching with replacemant results in the matches in Table 3. From here the distribution of ATT statistics is:

$$t\begin{bmatrix}1&0\\1&0\\1&0\end{bmatrix} = 1/3 \qquad t\begin{bmatrix}0&1\\1&0\\1&0\end{bmatrix} = 1/3 \qquad t\begin{bmatrix}1&0\\0&1\\1&0\end{bmatrix} = 1/3 \qquad t\begin{bmatrix}1&0\\1&0\\0&1\end{bmatrix} = -1/3$$

$$t\begin{bmatrix}0&1\\0&1\\1&0\end{bmatrix} = 1/3 \qquad t\begin{bmatrix}1&0\\0&1\\0&1\end{bmatrix} = -1/3 \qquad t\begin{bmatrix}0&1\\1&0\\0&1\end{bmatrix} = -1/3 \qquad t\begin{bmatrix}0&1\\0&1\\0&1\end{bmatrix} = -1/3$$

After doing so, we see that only two unobserved values of $U$ are needed to make the correct inference about the hypothesis test regarding the ATT. We have four possible imputations, and so a general approach would be to repeat c) for all four:

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 1, U_4 = 1, U_{obs}, X\} = 0.792$$

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 1, U_4 = 0, U_{obs}, X\} = 0.672$$

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 0, U_4 = 1, U_{obs}, X\} = 0.792$$

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 0, U_4 = 0, U_{obs}, X\} = 0.672$$

A "sensitivity test" based on this unobserved variable would suggest that the correct $p$-value lies between 0.672 and 0.792.

[Note that if we repeat the analysis for the McNemar test statistic we get the following result:

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 1, U_4 = 1, U_{obs}, X\} = 0.891$$

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 1, U_4 = 0, U_{obs}, X\} = 0.828$$

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 0, U_4 = 1, U_{obs}, X\} = 0.922$$

$$Pr\{t(T,r) \geq \tilde{t}|U_2 = 0, U_4 = 0, U_{obs}, X\} = 0.878$$

And a correct $p$-value that lies between 0.828 and 0.922.]

4: In this problem, you will analyze a famous experiment conducted by Leonard Wantchekon in Benin in 2001. Wantchekon wanted to examine the effectiveness of different types of campaign messages on voting behavior in a presidential election. For details, see:

`http://www.princeton.edu/~lwantche/Clientelism_and_Voting_Behavior_Wantchekon.pdf`

Wantchekon convinced the campaigns of the major presidential candidates to randomize the messages they employed in 24 villages. The two treatment conditions were as follows:

1. *Public Policy:* Wantchekon describes this treatment condition as: "It was decided that any public policy platform would raise issues pertaining to national unity and peace, eradicating corruption, alleviating poverty, developing agriculture and industry, protecting the rights of women and children, developing rural credit, providing access to the judicial system, protecting the environment, and/or fostering educational reforms."
2. *Clientelist*: Wantchekon describes this treatment as: "A clientelist message, by contrast, would take the form of a specific promise to the village, for example, for government patronage jobs or local public goods, such as establishing a new local university or providing financial support for local fishermen or cotton producers."

The data has been modified for the assignment, but the basic structure of the experiment was *block* randomization. For the purposes of the assignment, villages were divided into groups of 2 based on geography and treatment status was randomized within the 8 groups of 2. The outcome variable is the vote share of the candidate participating in the experiment. The only covariate is the number of registered voters. In the dataset, `block` indicates block group, `reg.voters` is the registered voters covariate,`vote.pop`is the outcome variable, `treat` is a variable indicating treatment status.The data can be found here:

`http://sekhon.berkeley.edu/causalinf/data/hw2data.RData`

In this problem, we are interested in the difference between the clientelist and public policy conditions.

a. Estimate the effect the clientelist message compared to the public policy message, using the ITT estimator and the regression estimator. For the regression estimate, include block level dummy variables in your regression equation.
b. Now test the sharp null of no treatment effect using randomization inference. Use two test statistics: Wilcoxon's signed rank test (Rosenbaum 2002, pg. 32) and the difference in means. What are the two sided $p$-values under these two tests?
c. Under the assumption of a constant, additive, treatment effect, use randomization inference to find a 95% confidence interval of the treatment effect. Use the signed rank as your test statistic. See pages 44-46 in Rosenbaum (2002).
d. What can you conclude about the effectiveness of clientelistic appeals in Benin?
e. Bonus: Perform randomization inference with covariance adjustment. How does this effect your results? For a very good article on covariance adjustment with randomization inference, see:

Rosenbaum, Paul. 2002. "Covariance Adjustment in Randomized Experiments and Observational Studies." *Statistical Science* 17(3): 286-327.

See Below for R code:

```
rm(list=ls())
load(file=url("http://sekhon.berkeley.edu/causalinf/data/hw2data.RData"))

#Problem 4
```

```
##Part a
reg.model = lm(vote.pop~treat+as.factor(block), data = data)
reg.estimate = coef(reg.model)[2]

diff.means = function(y,treat,blocks=NA){
  if(length(unique(blocks)==1)){
    ave.treated = mean(y[treat=="client"])
    ave.control = mean(y[treat=="pub.pol"])
    test.stat = ave.treated-ave.control
  }
else{
  #use tapply to calculate the within block averages
  ave.treated = tapply(y[treat=="client"],blocks[treat=="client"],mean)
  ave.control = tapply(y[treat=="pub.pol"],blocks[treat=="pub.pol"],mean)
  wt = tapply(y,blocks,length)/length(y)
  test.stat = sum(wt*(ave.treated-ave.control))
}
  return(test.stat)
}

itt.estimate = diff.means(data$vote.pop,data$treat,data$block)

#Problem 4
##Part b
#Let's create a treatment assignment function
treat.assign = function(treat,blocks=NA){
  if(length(unique(blocks))==1){
    treat.vector = sample(treat)
  }
  else{
    #randomize within blocks using tapply
    treat.vector = tapply(treat,blocks,sample)
  #tapply returns a list, to turn into a vector, use "unlist"
  treat.vector = unlist(treat.vector)
  }
  return(treat.vector)
}

omega = replicate(5000,treat.assign(data$treat,data$block))
#We only want unique treatment assignments, so let's get rid
 #of duplicates. Specifying "margin=2" keeps unique columns
omega = unique(omega,MARGIN=2)


#what's the true test stat?
true.test.stat = diff.means(data$vote.pop,data$treat,data$block)

#Now let's compute the entire distribution of test-statistics
 #under the null hypothesis
#create a matrix to hold the test statistics
test.stat.dist = matrix(nrow=ncol(omega))
```

```
#loop over omega, caculate the test stat every iteration
for (i in 1:ncol(omega)){
  treat.fake = omega[,i]
  fake.test.stat = diff.means(data$vote.pop,treat.fake,data$block)
  test.stat.dist[i] = fake.test.stat
}

#Let's plot the randomization distribution
plot(density(test.stat.dist),col="blue", main="Randomization Distribution")
#where does the true test statistic fall?
abline(v=true.test.stat,col="red",lwd=2)


#what's the p-value?
 2* min(sum(test.stat.dist>=true.test.stat)/ncol(omega),
  1-(sum(test.stat.dist>=true.test.stat)/ncol(omega)))

#create a rank sum function
strat.ranksum = function(y,treat,blocks){
  #rank within strata
  ranks = unlist(tapply(y,blocks,rank))
  #sum ranks of treated units
  ranksum = sum(ranks[treat=="client"])
  return(ranksum)
}

#observed test statistic?
true.test.stat = strat.ranksum(data$vote.pop, data$treat, data$block)

#Now let's compute the entire distribution of test-statistics
 #under the null hypothesis
#create a matrix to hold the test statistics
test.stat.dist = matrix(ncol=ncol(omega))
#loop over omega, caculate the test stat every iteration
for (i in 1:ncol(omega)){
  treat.fake = omega[,i]
  fake.test.stat = strat.ranksum(data$vote.pop, treat.fake, data$block)
  test.stat.dist[i] = fake.test.stat
}

#Let's plot the randomization distribution
plot(density(test.stat.dist),col="blue", main="Randomization Distribution")
#where does the true test statistic fall?
abline(v=true.test.stat,col="red",lwd=2)


#what's the p-value?
 2* min(sum(test.stat.dist>=true.test.stat)/ncol(omega),
  1-(sum(test.stat.dist>=true.test.stat)/ncol(omega)))
```

```
#Problem 4
#Part c

omega = replicate(5000,treat.assign(data$treat,data$block))
#We only want unique treatment assignments, so let's get rid
 #of duplicates. Specifying "margin=2" keeps unique columns
omega = unique(omega,MARGIN=2)


val.noreject = c()
for(j in seq(-1,1,by=.01)){
  #create new outcome vector with hypothesis subtracted out from treated units
  vote.pop.new = ifelse(data$treat=="client", data$vote.pop-j, data$vote.pop)
  #observed test statistic?
  true.test.stat = strat.ranksum(vote.pop.new, data$treat, data$block)
  #Now let's compute the entire distribution of test-statistics
   #under the null hypothesis
  #create a matrix to hold the test statistics
  test.stat.dist = matrix(ncol=ncol(omega))
  #loop over omega, caculate the test stat every iteration

  for (i in 1:ncol(omega)){
    treat.fake = omega[,i]
    fake.test.stat = strat.ranksum(vote.pop.new, treat.fake, data$block)
    test.stat.dist[i] = fake.test.stat
  }

                                        #what's the p-value?
  p.val = 2* min(sum(test.stat.dist>=true.test.stat)/ncol(omega),
   1-(sum(test.stat.dist>=true.test.stat)/ncol(omega)))
  #if the null hypothesis is not rejected, keep value
  if(p.val>.05) val.noreject = c(val.noreject, j)
  print(j)
}

#The 95% confidence interval is
range(val.noreject)


# Problem 4
# Part d

reg.model = lm(vote.pop~treat+as.factor(block), data = data)
y = lm(vote.pop~reg.voters+as.factor(block),data=data)$residuals

#whats the true or observed test statistic?
true.test.stat = strat.ranksum(y,data$treat,data$block)

#Now let us compute the entire distribution of test-statistics
 #under the null hypothesis
#create a matrix to hold the test statistics
```

```
test.stat.dist = matrix()
#loop over omega, caculate the test stat every iteration
# reuse omega from above
for (i in 1:ncol(omega)){
  treat.fake = omega[,i]
  fake.test.stat = strat.ranksum(y,treat.fake,data$block)
  test.stat.dist[i] = fake.test.stat
}

#Lets plot the randomization distribution
plot(density(test.stat.dist),col="blue", main="Randomization Distribution")
#where does the true test statistic fall?
abline(v=true.test.stat,col="red",lwd=2)

#whats the p-value?
1-sum(test.stat.dist>true.test.stat)/ncol(omega)
```