Statistical Rethinking Winter 2020 – Homework Week 6

Alessandro Gentilini .@gmail (just auditing)

January 24, 2021

1. The data in data(NWOGrants) are outcomes for scientific funding applications for the Netherlands Organization for Scientific Research (NWO) from 2010–2012 (see van der Lee and Ellemers doi:10.1073/pnas.1510159112). These data have a very similar structure to the UCBAdmit data discussed in Chapter 11. I want you to consider a similar question: What are the total and indirect causal effects of gender on grant awards? Consider a mediation path (a pipe) through discipline. Draw the corresponding DAG and then use one or more binomial GLMs to answer the question. What is your causal interpretation? If NWO's goal is to equalize rates of funding between the genders, what type of intervention would be most effective?

```
library(rethinking)
```

```
## Loading required package: rstan
## Loading required package: StanHeaders
## Loading required package: ggplot2
## rstan (Version 2.21.2, GitRev: 2e1f913d3ca3)
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## Loading required package: parallel
## rethinking (Version 2.13)
##
## Attaching package: 'rethinking'
## The following object is masked from 'package:stats':
##
##     rstudent
```

```
data(NWOGrants)
d<-NWOGrants
d
```

```
##             discipline gender applications awards
## 1    Chemical sciences      m           83     22
## 2    Chemical sciences      f           39     10
## 3    Physical sciences      m          135     26
## 4    Physical sciences      f           39      9
## 5              Physics      m           67     18
## 6              Physics      f            9      2
## 7           Humanities      m          230     33
```

```
## 8           Humanities      f          166     32
## 9   Technical sciences      m          189     30
## 10  Technical sciences      f           62     13
## 11   Interdisciplinary      m          105     12
## 12   Interdisciplinary      f           78     17
## 13 Earth/life sciences      m          156     38
## 14 Earth/life sciences      f          126     18
## 15     Social sciences      m          425     65
## 16     Social sciences      f          409     47
## 17    Medical sciences      m          245     46
## 18    Medical sciences      f          260     29
```

```
library(dagitty)
dag <- dagitty( "dag {
    G -> D
    D -> A
}")
coordinates(dag) <- list(
    x=c(G=-1, D=0, A=1),
    y=c(G=0, D=0, A=0)
)
drawdag(dag)
```


```
adjustmentSets( dag, exposure="G" , outcome="A" )
```

```
##  {}
```

For the above DAG, as `dagitty.net/dags.html` says *No adjustment is necessary to estimate the total effect of G on A*. So for the total effect the following model is fine:

```
## R code 11.29
dat_list <- list(
    admit = d$awards,
    applications = d$applications,
    gid = ifelse( d$gender=="m" , 1 , 2 )
)
m11.7 <- ulam(
    alist(
        admit ~ dbinom( applications , p ) ,
        logit(p) <- a[gid] ,
        a[gid] ~ dnorm( 0 , 1.5 )
    ) , data=dat_list , chains=4 )
```

```
##
## SAMPLING FOR MODEL '1c8fe12824d87b1e9ee4f9e6659968f2' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.11 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 1: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 1: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 1: Iteration: 300 / 1000 [ 30%]  (Warmup)
```

```
## Chain 1: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 1: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 1: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 1: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 1: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 1: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 1: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 1: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.011277 seconds (Warm-up)
## Chain 1:                0.010711 seconds (Sampling)
## Chain 1:                0.021988 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL '1c8fe12824d87b1e9ee4f9e6659968f2' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 5e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 2: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 2: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 2: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 2: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 2: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 2: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 2: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 2: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 2: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 2: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 2: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.011319 seconds (Warm-up)
## Chain 2:                0.010854 seconds (Sampling)
## Chain 2:                0.022173 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL '1c8fe12824d87b1e9ee4f9e6659968f2' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 5e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 3: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 3: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 3: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 3: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 3: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 3: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 3: Iteration: 600 / 1000 [ 60%]  (Sampling)
```

```
## Chain 3: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 3: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 3: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 3: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.011713 seconds (Warm-up)
## Chain 3:                0.010605 seconds (Sampling)
## Chain 3:                0.022318 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL '1c8fe12824d87b1e9ee4f9e6659968f2' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 5e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 4: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 4: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 4: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 4: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 4: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 4: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 4: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 4: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 4: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 4: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 4: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.011362 seconds (Warm-up)
## Chain 4:                0.010703 seconds (Sampling)
## Chain 4:                0.022065 seconds (Total)
## Chain 4:
```

```
precis( m11.7 , depth=2 )
```

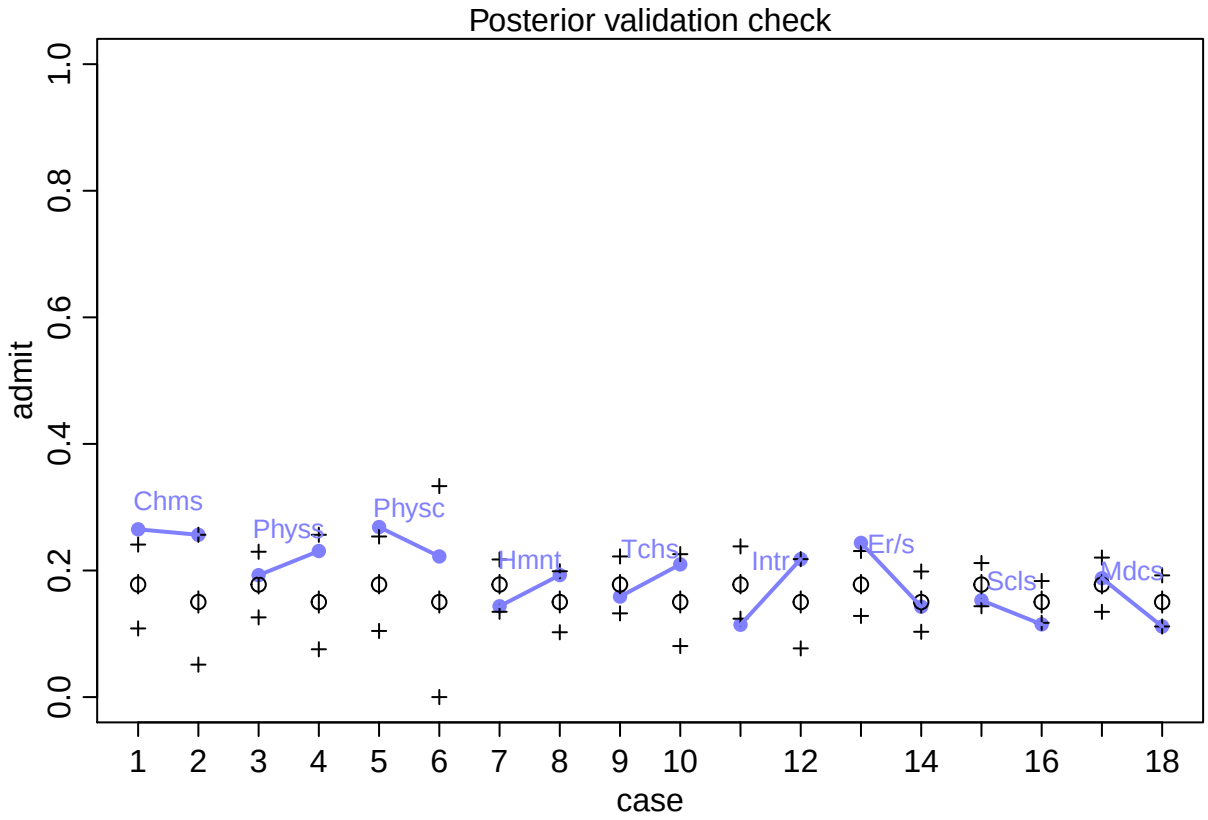
```
##           mean         sd      5.5%     94.5%    n_eff    Rhat4
## a[1] -1.532595 0.06339437 -1.631797 -1.430435 1400.117 1.003180
## a[2] -1.735931 0.08293421 -1.871127 -1.606557 1161.991 1.002776
```

```
## R code 11.30
post <- extract.samples(m11.7)
diff_a <- post$a[,1] - post$a[,2]
diff_p <- inv_logit(post$a[,1]) - inv_logit(post$a[,2])
precis(list( diff_a=diff_a , diff_p=diff_p ), hist=FALSE)
```

```
##              mean         sd        5.5%      94.5%
## diff_a 0.20333519 0.10030135 0.043029503 0.36780188
## diff_p 0.02766706 0.01345783 0.005852621 0.04908667
```

```
## R code 11.31
postcheck( m11.7 )
# draw lines connecting points from same dept
labels <- abbreviate(d$discipline)
for ( i in 1:9 ) {
```

```
    x <- 1 + 2*(i-1)
    y1 <- d$awards[x]/d$applications[x]
    y2 <- d$awards[x+1]/d$applications[x+1]
    lines( c(x,x+1) , c(y1,y2) , col=rangi2 , lwd=2 )
    text( x+0.5 , (y1+y2)/2 + 0.05 , labels[x] , cex=0.8 , col=rangi2 )
}
```



So the total effect is that: probability difference between male (1) and female (2) is between 0.6% and 5%, so positive, so male advantage.

Instead, the *Minimal sufficient adjustment sets for estimating the direct effect of G on A:* is the set with just one element: D; so the following model adjusts for D:

```
## R code 11.32
dat_list$dept_id <- rep(1:9,each=2)
m11.8 <- ulam(
    alist(
        admit ~ dbinom( applications , p ) ,
        logit(p) <- a[gid] + delta[dept_id] ,
        a[gid] ~ dnorm( 0 , 1.5 ) ,
        delta[dept_id] ~ dnorm( 0 , 1.5 )
    ) , data=dat_list , chains=4 , iter=4000 )
```

```
##
## SAMPLING FOR MODEL 'a7a7a4ab228c37095f791eebea8cb75b' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.4e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.14 seconds.
## Chain 1: Adjust your expectations accordingly!
```

```
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 1: Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1: Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 1: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 1: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.217035 seconds (Warm-up)
## Chain 1:                0.207608 seconds (Sampling)
## Chain 1:                0.424643 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'a7a7a4ab228c37095f791eebea8cb75b' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2: Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 2: Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 2: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 2: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.220109 seconds (Warm-up)
## Chain 2:                0.20813 seconds (Sampling)
## Chain 2:                0.428239 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'a7a7a4ab228c37095f791eebea8cb75b' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 7e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3: Iteration:  400 / 4000 [ 10%]  (Warmup)
```

```
## Chain 3: Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 3: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 3: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.208164 seconds (Warm-up)
## Chain 3:                0.198221 seconds (Sampling)
## Chain 3:                0.406385 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'a7a7a4ab228c37095f791eebea8cb75b' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4: Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4: Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 4: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 4: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 4: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.225939 seconds (Warm-up)
## Chain 4:                0.208479 seconds (Sampling)
## Chain 4:                0.434418 seconds (Total)
## Chain 4:
```

```
precis( m11.8 , depth=2, hist=FALSE )
```

```
##                mean        sd       5.5%       94.5%    n_eff    Rhat4
## a[1]     -1.1520957 0.4330668 -1.8653579 -0.47552670 675.8125 1.002696
## a[2]     -1.2901530 0.4379805 -2.0082673 -0.60245713 690.2583 1.002543
## delta[1]  0.1480699 0.4723550 -0.5971921  0.92606177 790.9869 1.002317
## delta[2] -0.2079857 0.4654809 -0.9468920  0.55095978 783.9943 1.002298
## delta[3]  0.1229196 0.4896582 -0.6542438  0.92418965 843.8522 1.002410
## delta[4] -0.4241209 0.4495409 -1.1437122  0.32149448 733.7159 1.002336
## delta[5] -0.3967260 0.4578030 -1.1251849  0.34623643 752.2993 1.002168
## delta[6] -0.4689525 0.4678046 -1.2027930  0.28748691 805.4458 1.002584
## delta[7] -0.1892292 0.4529644 -0.9029191  0.54376392 742.6094 1.002233
## delta[8] -0.6460323 0.4394002 -1.3401361  0.07773726 698.0800 1.002695
## delta[9] -0.5273413 0.4465558 -1.2231240  0.21287282 718.0016 1.002282
```

```
## R code 11.33
post <- extract.samples(m11.8)
diff_a <- post$a[,1] - post$a[,2]
diff_p <- inv_logit(post$a[,1]) - inv_logit(post$a[,2])
precis( list( diff_a=diff_a , diff_p=diff_p ), hist=FALSE )
```

```
##              mean         sd        5.5%      94.5%
## diff_a 0.13805729 0.10511377 -0.03141501 0.30727038
## diff_p 0.02393829 0.01930684 -0.00510158 0.05652522
```

```
## R code 11.34
pg <- with( dat_list , sapply( 1:9 , function(k)
    applications[dept_id==k]/sum(applications[dept_id==k]) ) )
rownames(pg) <- c("male","female")
colnames(pg) <- abbreviate(unique(d$discipline))
round( pg , 2 )
```

```
##        Chms Physs Physc Hmnt Tchs Intr Er/s Scls Mdcs
## male   0.68  0.78  0.88 0.58 0.75 0.57 0.55 0.51 0.49
## female 0.32  0.22  0.12 0.42 0.25 0.43 0.45 0.49 0.51
```

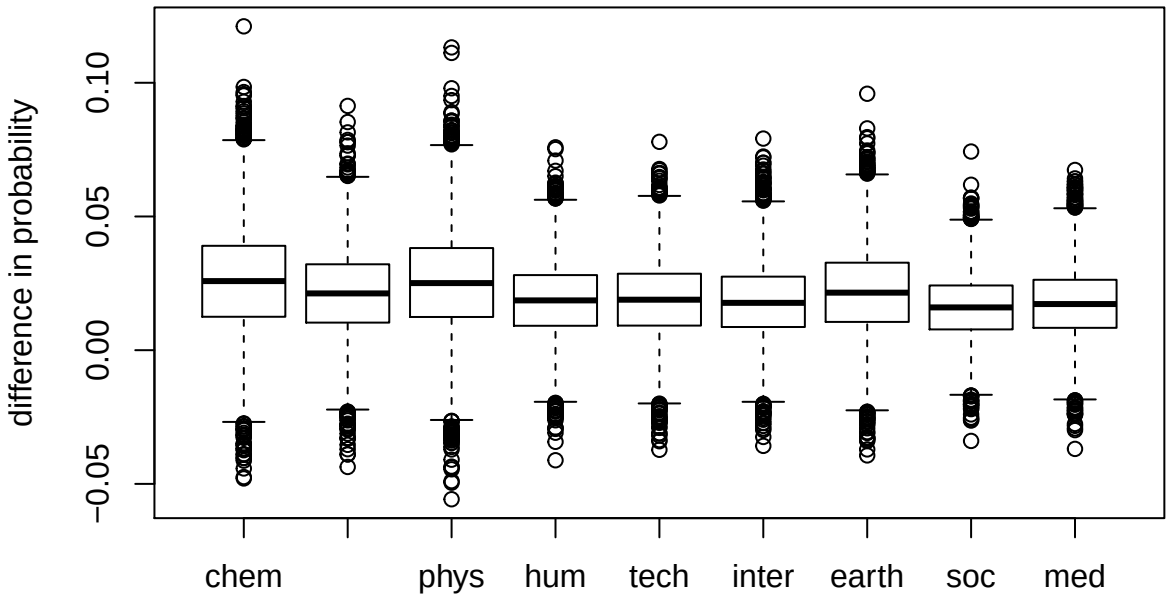
```
chem<-inv_logit(post$a[,1]+post$delta[,1])-inv_logit(post$a[,2]+post$delta[,1])
physical<-inv_logit(post$a[,1]+post$delta[,2])-inv_logit(post$a[,2]+post$delta[,2])
phys<-inv_logit(post$a[,1]+post$delta[,3])-inv_logit(post$a[,2]+post$delta[,3])
hum<-inv_logit(post$a[,1]+post$delta[,4])-inv_logit(post$a[,2]+post$delta[,4])
tech<-inv_logit(post$a[,1]+post$delta[,5])-inv_logit(post$a[,2]+post$delta[,5])
inter<-inv_logit(post$a[,1]+post$delta[,6])-inv_logit(post$a[,2]+post$delta[,6])
earth<-inv_logit(post$a[,1]+post$delta[,7])-inv_logit(post$a[,2]+post$delta[,7])
soc<-inv_logit(post$a[,1]+post$delta[,8])-inv_logit(post$a[,2]+post$delta[,8])
med<-inv_logit(post$a[,1]+post$delta[,9])-inv_logit(post$a[,2]+post$delta[,9])

diff_in_prob<-data.frame(
    chem=chem,physical=physical,phys=phys,hum=hum,
    tech=tech,inter=inter,earth=earth,soc=soc,med=med
)

precis(diff_in_prob,digits=2,hist=FALSE)
```

```
##                mean         sd         5.5%      94.5%
## chem     0.02595783 0.01996730 -0.005971757 0.05786889
## physical 0.02123070 0.01629810 -0.005020212 0.04739417
## phys     0.02535719 0.01946690 -0.005745347 0.05666063
## hum      0.01865716 0.01427393 -0.004311214 0.04162472
## tech     0.01887235 0.01443463 -0.004389686 0.04225132
## inter    0.01816500 0.01417233 -0.004076405 0.04120737
## earth    0.02171954 0.01668648 -0.004834403 0.04876268
## soc      0.01602025 0.01226160 -0.003678893 0.03597327
## med      0.01745778 0.01340609 -0.003971226 0.03908535
```

```
boxplot(diff_in_prob,main='Difference in prob. of award between men and women, within disciplines',
ylab='difference in probability')
```



There is a slightly advantage for men: on average it is about 2.5%, so the most effective intervention is the following: when you have to choose between a man grant and a woman grant that have the same value then pick a random number n between 0 and 1 and give the award to the woman if $n < 0.525$.

---

2. Suppose that the NWO Grants sample has an unobserved confound that influ- ences both choice of discipline and the probability of an award. One example of such a confound could be the career stage of each applicant. Suppose that in some disciplines, junior scholars apply for most of the grants. In other disciplines, schol- ars from all career stages compete. As a result, career stage influences discipline as well as the probability of being awarded a grant. Add these influences to your DAG from Problem 1. What happens now when you condition on discipline? Does it provide an un-confounded estimate of the direct path from gender to an award? Why or why not? Justify your answer with the back-door criterion. Hint: This is structurally a lot like the grandparents-parents- children-neighborhoods example from a previous week. If you have trouble thinking this though, try simulating fake data, assuming your DAG is true. Then analyze it using the model from Problem 1. What do you con- clude? Is it possible for gender to have a real direct causal influence but for a regres- sion conditioning on both gender and discipline to suggest zero influence?

Using `dagitty.net/dags.html`, if Career Stage can be observed then:

1. No adjustment is necessary to estimate the **total** effect of G on A.
2. The Minimal sufficient adjustment sets for estimating the **direct** effect of G on A is the set {Career Stage, D}; adjusting just for the Career Stage is fine (white node is adjusted) while adjusting just for the discipline introduces a biasing path. Adjusting for both D and Career Stage is fine.

---

3. The data contained in library(MASS);data(eagles) are records of salmon pirating attempts by Bald Eagles in Washington State. See ?eagles for details. While one eagle feeds, sometimes another will swoop in and try to steal the salmon from it. Call the feeding eagle the "victim" and the thief the "pirate." Use the avail- able data to build one or more binomial GLMs of successful pirating attempts, using size and age as predictors. Consider any relevant interactions.

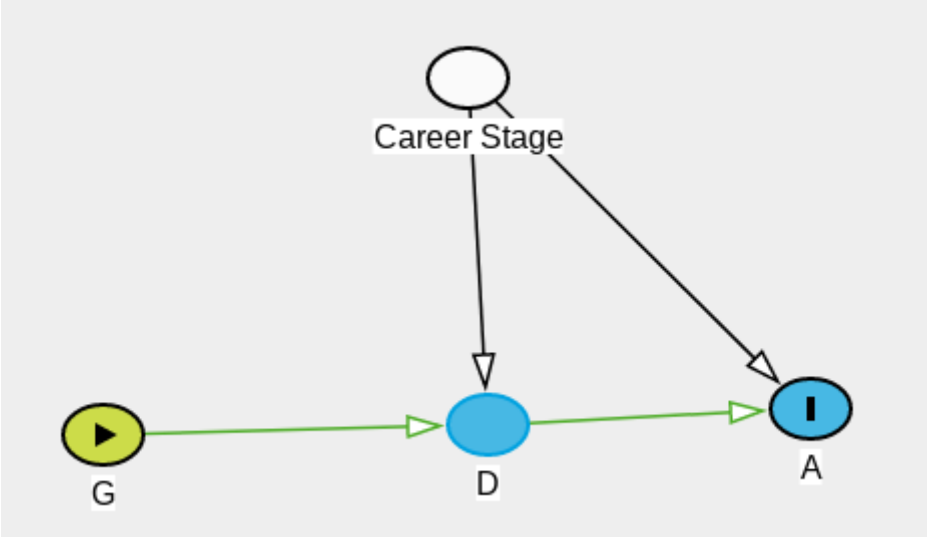


Figure 1: Adjusting just for career is fine.

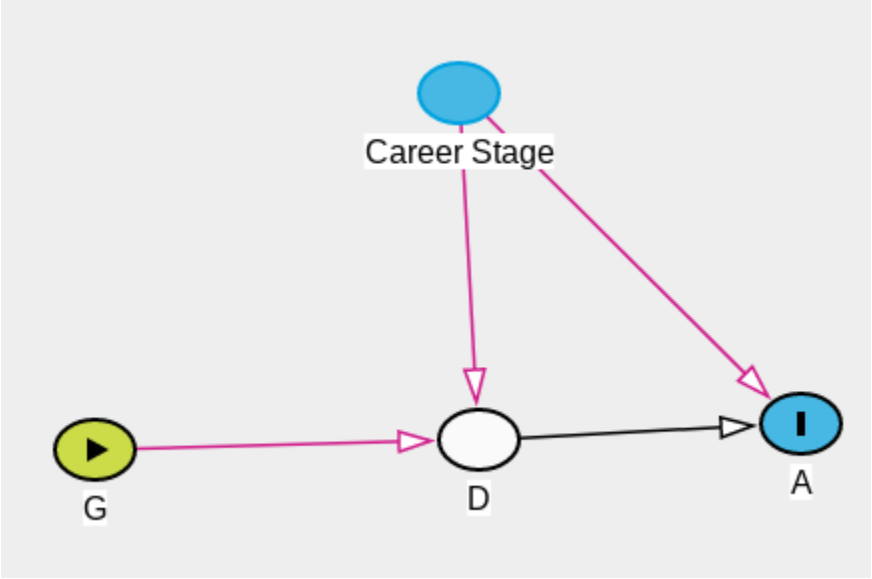


Figure 2: Biasing path (in magenta) appears when adjusting just for Discipline.

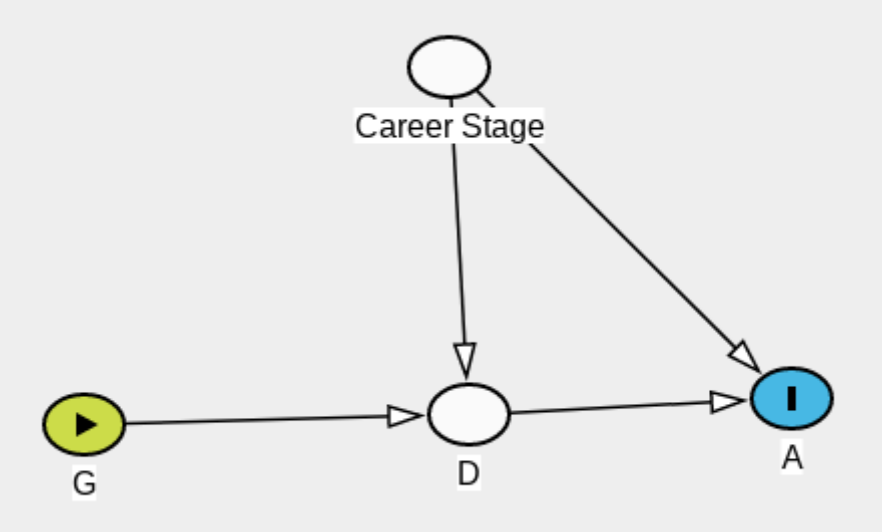
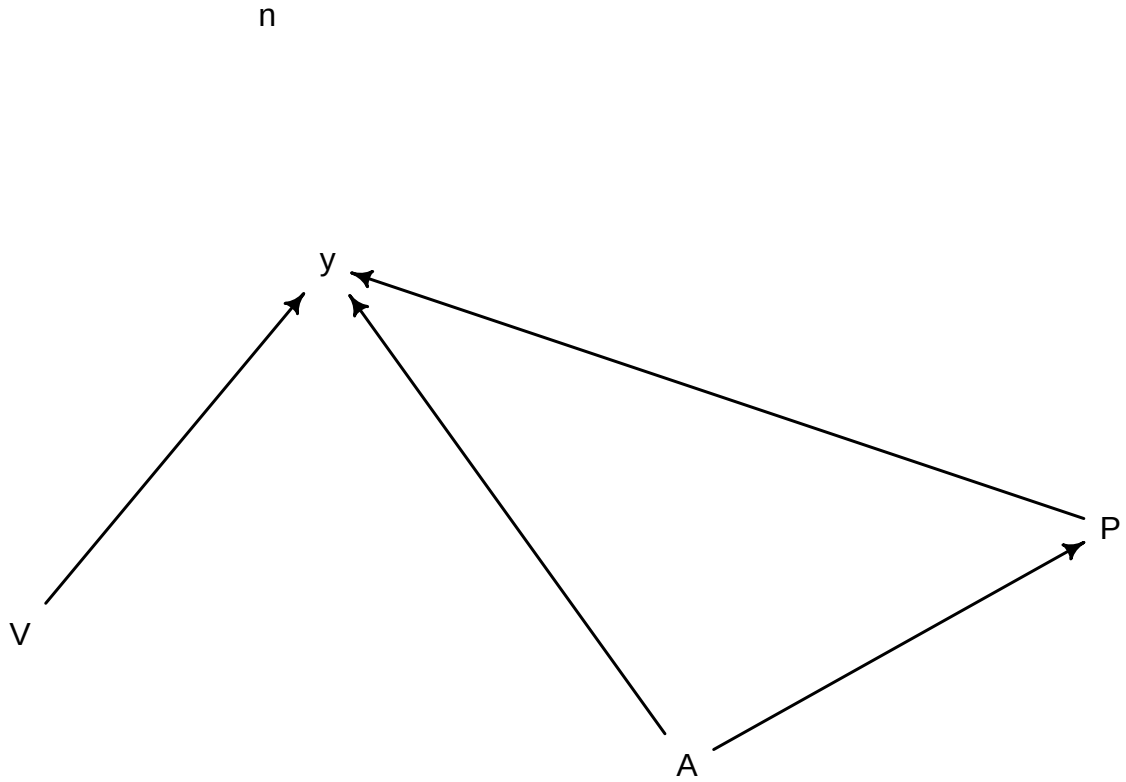


Figure 3: Adjusting for both does not introduce a biasing path.

```
edag1 <- dagitty('dag {
A [exposure,pos="-0.401,0.293"]
P [exposure,pos="0.147,-0.141"]
V [exposure,pos="-1.264,0.054"]
n [pos="-0.944,-1.075"]
y [outcome,pos="-0.866,-0.622"]
A -> P
A -> y
P -> y
V -> y
}
')
drawdag(edag1)
```




```
edag2 <- dagitty('dag {
A [exposure,pos="-0.401,0.293"]
P [exposure,pos="0.147,-0.141"]
V [exposure,pos="-1.264,0.054"]
n [pos="-0.944,-1.075"]
y [outcome,pos="-0.866,-0.622"]
A -> y
P -> y
V -> y
}
')
drawdag(edag2)
```

n