STATISTICAL RETHINKING 2022
WEEK 3 SOLUTIONS

1. Because there are no back-door paths from `area` to `avgfood`, we only need to include `area` to the causal effect of area. No other variables are needed. In fact, adding other variables could cause bias. Here is a model using standardized versions of the variables and those standardized priors from the book:

```
library(rethinking)
data(foxes)
d <- foxes
d$W <- standardize(d$weight)
d$A <- standardize(d$area)
d$F <- standardize(d$avgfood)
d$G <- standardize(d$groupsize)

m1 <- quap(
    alist(
        F ~ dnorm( mu , sigma ),
        mu <- a + bA*A,
        a ~ dnorm(0,0.2),
        bA ~ dnorm(0,0.5),
        sigma ~ dexp(1)
    ), data=d )

precis(m1)
```

```
      mean   sd  5.5% 94.5%
a     0.00 0.04 -0.07  0.07
bA    0.88 0.04  0.81  0.95
sigma 0.47 0.03  0.42  0.52
```

Territory size seems to have a substantial effect on food availability. These are standardized variables, so `bA` above means that each standard deviation change in area results on average in about 0.9 standard deviations of change in food availability.

2. To infer the causal influence of `avgfood` on `weight`, we need to close any back-door paths. There are no back-door paths in the DAG. So again, just use a model with a single predictor.

```
m2 <- quap(
    alist(
        W ~ dnorm( mu , sigma ),
        mu <- a + bF*F,
        a ~ dnorm(0,0.2),
        bF ~ dnorm(0,0.5),
        sigma ~ dexp(1)
    ), data=d )
precis(m2)
```

```
       mean   sd  5.5% 94.5%
a      0.00 0.08 -0.13  0.13
bF    -0.02 0.09 -0.17  0.12
sigma  0.99 0.06  0.89  1.09
```

There seems to be only a small total effect of food on weight, if there is any effect at all. It's about equally plausible that it's negative as positive, and it's small either way.

Now for the direct effect. We need to block the mediated path through group size *G*. That means stratify by group size.

```
m2b <- quap(
    alist(
        W ~ dnorm( mu , sigma ),
        mu <- a + bF*F + bG*G,
        a ~ dnorm(0,0.2),
        c(bF,bG) ~ dnorm(0,0.5),
        sigma ~ dexp(1)
    ), data=d )
precis(m2b)
```

```
       mean   sd  5.5% 94.5%
a      0.00 0.08 -0.13  0.13
bF     0.48 0.18  0.19  0.76
bG    -0.57 0.18 -0.86 -0.29
sigma  0.94 0.06  0.84  1.04
```

The direct effect of food on weight is positive (0.19–0.76), it seems. That makes sense. This model also gives us the direct effect (also the total effect) of group size on weight. And it is the opposite and of the same magnitude as the direct effect of food. These two effects seem to cancel one another. That may be why the total effect of food is about zero: the direct effect is positive but the mediated effect through groups size is negative.

What is going on here? Larger territories increase available food (problem 1). But increases in food (and territory) do not influence fox weight. The reason seems to be because adding more food directly increases weight, but the path through group size cancels that increase. To check this idea, we can estimate the causal effect of food on groups size:

```
m2c <- quap(
    alist(
        G ~ dnorm( mu , sigma ),
        mu <- a + bF*F,
        a ~ dnorm(0,0.2),
        bF ~ dnorm(0,0.5),
        sigma ~ dexp(1)
    ), data=d )
precis(m2c)
```

```
      mean   sd  5.5% 94.5%
a     0.00 0.04 -0.06  0.06
bF    0.90 0.04  0.83  0.96
sigma 0.43 0.03  0.39  0.48
```

Food appears to have a large and reliably (0.83–0.96) effect on group size. That is, more food means more foxes. This is consistent with the idea that the mediating influence of group size cancels the direct influence of more food on individual fox body weight. In simple terms, the benefits of more food are canceled by more foxes being attracted to the food, so each fox gets the same amount.

The ecologists will recognize this situation as an *ideal free distribution*.

3. As in the example from the lecture, we need to use the backdoor criterion to find the adjustment set to estimate $X \rightarrow Y$. It turns out to be the same as in the lecture: We need to stratify again A an S. If this isn't obvious, that's okay. Notice that adding U has not introduced any new arrow into X. And while stratifying by S does now open a collider path $A \rightarrow S \leftarrow U$, that doesn't influence our estimate of $X \rightarrow Y$. So all good.

You can check your logic using the dagitty package:

```
library(dagitty)
t2f_dag <- dagitty( "dag {
    X -> Y
    S -> X
    S -> Y
    A -> Y
    A -> X
    A -> S
    S <- U -> Y
}")
adjustmentSets( t2f_dag , exposure="X" , outcome="Y" )
```

```
{ A, S }
```

Now the implications for each coefficient. The coefficient for X should still be the estimate of the causal effect of X on Y, $P(Y|\text{do}(X))$. But the other coefficients are now biased by U. When we stratify by S, we open the collider path that S is on: $A \rightarrow S \leftarrow U$. Now the coefficients for A and S are not even partial causal effects, because both are biased by the collider through U. In effect the unobserved confound makes the control coefficients uninterpretable even as partial causal effects.

The irony here is that is still possible to estimate the casual effect of age A on Y. But in the model that stratifies by S, the coefficient for age becomes confounded. It really is not safe to interpret control coefficients, unless there is an explicit causal model.

4-OPTIONAL CHALLENGE. To translate the DAG into a generative simulation, we first simulate the variables that have no parents (causes in the graph). Here, these are U and A. Then we simulate the variables that depend only on U or A, which is only S. Then we can simulate X, since it depends on U and A and S. Finally we can simulate Y, which depends upon all of the other variables. I'll put all this into a function, so we can run it with variable inputs.

```
f <- function(N=100,bX=0) {
    U <- rnorm(N)
    A <- rnorm(N)
    S <- rnorm(N,A+U)
    X <- rnorm(N,A+S)
    Y <- rnorm(N,A+S+bX*X+U)
    return(list(
        A=standardize(A),
        S=standardize(S),
        X=standardize(X),
        Y=standardize(Y)))
}
```

And we can try it out, fitting the model to estimate $P(Y|\text{do}(X))$ by stratifying by A and S:

```
flist <- alist(
    Y ~ dnorm(mu,exp(log_sigma)),
    mu <- a + bX*X + bS*S + bA*A,
    a ~ dnorm(0,0.2),
    c(bX,bS,bA) ~ dnorm(0,0.5),
    log_sigma ~ dnorm(0,1)
)
sim_dat <- f(N=10,bX=0)
m4 <- quap( flist , data=sim_dat )
precis(m4)
```

```
           mean   sd  5.5% 94.5%
a          0.00 0.10 -0.16  0.16
bX        -0.23 0.24 -0.61  0.16
bS         0.64 0.21  0.31  0.98
bA         0.52 0.19  0.22  0.82
log_sigma -1.04 0.25 -1.43 -0.64
```

That's just one example. To get a sense of how sample size influences the posterior distribution of bX, we need to repeat it. Let's write another function. This one will repeat the analysis above for any specific values of N and bX.

```
# function to repeat analysis at specific N
g <- function(N_reps=1e3,N=100,bX=0) {
    r <- mcreplicate( N_reps , mc.cores=8 , {
        sim_dat <- f(N=N,bX=bX)
        m <- quap( flist , data=sim_dat )
        # return width of 89% interval and mean distance from true value
        iw <- as.numeric( precis(m)[2,4] - precis(m)[2,3] )
        md <- as.numeric( abs(precis(m)[2,1] - bX) )
        return(c(md,iw))
    } )
    return(r)
}
```

I've written this to return two measures of accuracy. The first is the mean distance from the true value (which is zero in my simulations, but you can change it—beware the standardization step if you do!). People sometimes call this "bias". The second is the width of the 89% interval. This is a measure of precision. There's nothing special about 89%. But it's wide, so covers the vast majority of the highly-plausible values.

As an example, let's run this function for $N = 10$ and visualize the output.

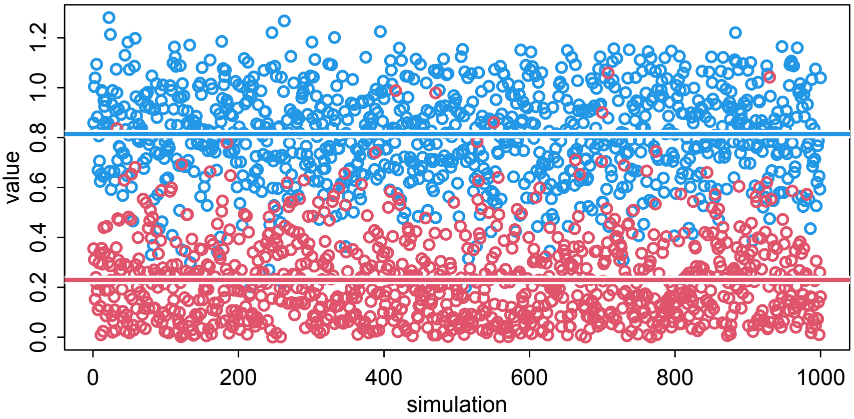
```
x <- g(N=10)

plot( x[2,] , lwd=2 , col=4 , ylim=c(0,max(x)) ,
  xlab="simulation" , ylab="value" )
points( 1:ncol(x) , x[1,] , lwd=2 , col=2 )

abline( h=mean(x[1,]) , lwd=6 , col="white" )
abline( h=mean(x[1,]) , lwd=3 , col=2 )

abline( h=mean(x[2,]) , lwd=6 , col="white" )
abline( h=mean(x[2,]) , lwd=3 , col=4 )
```

This is the result:



The blue points are the interval widths. And the red points are the bias measurements. The lines show the averages. With only 10 observations, the intervals can be very wide—remember the units here are standard deviations, because I standardized the data before running the regression. The average bias is 0.2 of a standard deviation. And the interval cover a huge range.

Maybe it's worth another sermon about zero. The fact that any interval includes zero does not mean that zero is the most likely value of coefficient. In a model like this one, in which we want to estimate something of epidemiological importance, a wide interval that spans zero is not evidence that there is no influence of X on Y. The posterior is consistent many large effects, both negative and positive. It would be irresponsible at best to report "no effect" just because some interval overlaps zero. You could kill people. Killing is bad, even if you don't intend it. This has been another sermon about zero.

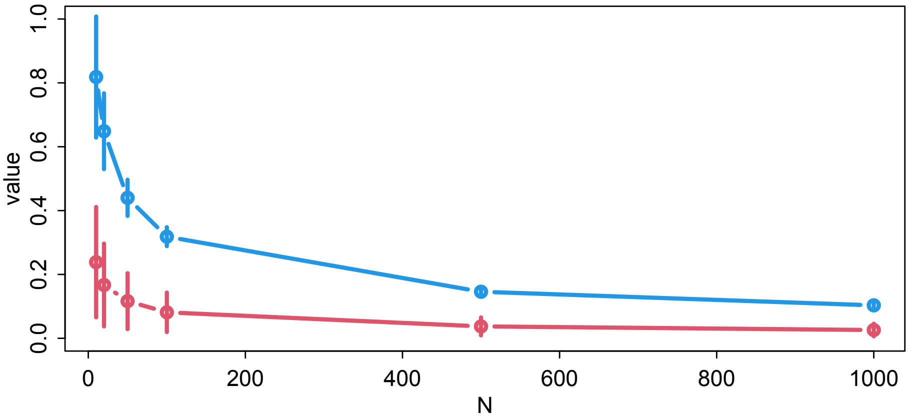
The last thing to do is repeat the simulations above for different values of N. I will write a loop to do it and then plot the results.

```
N_seq <- c(10,20,50,100,500,1000)

y <- sapply( N_seq , function(n) {
    x <- g(N=n)
    return( c( apply(x,1,mean) , apply(x,1,sd) ) )
} )

plot( N_seq , y[2,] , lwd=3 , col=4 , type="b" ,
    ylim=c(0,1) , xlab="N" , ylab="value" )
points( N_seq , y[1,] , lwd=3 , col=2 , type="b" )
for ( i in 1:length(N_seq) ) {
    lines( c(N_seq[i],N_seq[i]) , c(y[1,i]+y[3,i],y[1,i]-y[3,i]) ,
        lwd=3 , col=2 )
    lines( c(N_seq[i],N_seq[i]) , c(y[2,i]+y[4,i],y[2,i]-y[4,i]) ,
        lwd=3 , col=4 )
}
```

This is the result:



Again the blue points are interval widths, but now they are average widths at each sample size on the horizontal axis. The line segments are the standard deviations. The red points and segments are the means and standard deviations, respectively, for the bias. Picking an optimal sample size would require specifying some cost-benefit function. But even without that, we can see that the greatest inferential benefit comes from the first few hundred observations.

If there were substantial effect heterogeneity among clusters in the population, we might need much more data. But that's a different DAG and simulation...