STATISTICAL RETHINKING 2024
WEEK 7 SOLUTIONS

1. Here's the DAG from lecture, for reference:

$$\begin{tikzcd}
A \arrow[r] \arrow[d] & C & D \arrow[l] \arrow[d] \\
K \arrow[ur] & & U \arrow[ll] \arrow[ul]
\end{tikzcd}$$

The total effect only of U needs only D, the district tendencies. Now the model for the direct effect must include all of the variables: K, A and D. Why? Because we need to condition on K to block the indirect effect through K. But K is a collider between A and U. So we also need to condition on A to block the collider confounding path that opens when we condition on K. Oh bother.

Now how we include A, K, D is up to us. I start with the correlated varying effects model from the lecture, which includes U and D. I will use an ordered monotonic effect for K. There are only the values 1 to 4 observed. So we only need 4 parameters. It it tempting to include A in the same way, but we only have it as a centered value, so we don't know the actual ages (although it isn't too hard to reconstruct them from the relative values). Here's my model:

```
library(rethinking)
data(bangladesh)
d <- bangladesh
dat <- list(
    C = d$use.contraception,
    D = as.integer(d$district),
    U = d$urban,
    A = standardize(d$age.centered),
    K = d$living.children )
dat$Kprior <- rep(2,3)

m1 <- ulam(
    alist(
        C ~ bernoulli(p),
        logit(p) <- a[D] + b[D]*U + bA*A +
                    bK*sum( delta_j[1:K] ),

        # ordered monotonic kids
        vector[4]: delta_j <<- append_row( 0 , delta ),
        simplex[3]: delta ~ dirichlet( Kprior ),
        c(bK,bA) ~ normal(0,0.5),
        # non-centered varying effects for D and U
        transpars> vector[61]:a <<- abar[1] + v[,1],
        transpars> vector[61]:b <<- abar[2] + v[,2],
        transpars> matrix[61,2]:v <-
            compose_noncentered( sigma , L_Rho , Z ),

        # non-centered priors
        matrix[2,61]:Z ~ normal( 0 , 1 ),
        vector[2]:abar ~ normal(0,1),
        cholesky_factor_corr[2]:L_Rho ~ lkj_corr_cholesky( 4 ),
        vector[2]:sigma ~ exponential(1),

        # convert Cholesky to Corr matrix
        gq> matrix[2,2]:Rho <<- Chol_to_Corr(L_Rho)
    ) , data=dat , chains=4 , cores=4 )

precis(m1,depth=2)
```

```
          mean   sd  5.5% 94.5% rhat ess_bulk
delta[1]  0.74 0.09  0.60  0.87 1.00  3116.20
delta[2]  0.16 0.08  0.05  0.30 1.00  3868.06
delta[3]  0.10 0.06  0.02  0.20 1.01  3714.54
bA       -0.22 0.07 -0.33 -0.12 1.00  3427.85
bK        1.26 0.16  1.01  1.51 1.00  2953.40
abar[1]  -1.57 0.16 -1.83 -1.33 1.00  2184.70
abar[2]   0.73 0.17  0.47  0.99 1.00  1986.45
sigma[1]  0.57 0.10  0.43  0.73 1.01   704.97
sigma[2]  0.71 0.22  0.39  1.05 1.01   458.05
b[1]      1.05 0.38  0.46  1.67 1.00  2584.04
b[2]      0.76 0.70 -0.37  1.86 1.01  4201.14
...
a[59]    -2.10 0.50 -2.93 -1.35 1.00  3405.96
a[60]    -2.02 0.43 -2.73 -1.36 1.00  4838.58
a[61]    -2.12 0.36 -2.74 -1.57 1.00  3227.99
```

I abbreviated the vector output for `a` and `b`, but you get the idea.

The average direct effect of *U* (`abar[2]`), according to this model, is basically unchanged. Compare to the coefficients in `precis(mCDUcov_nc,2)` from the lecture. This implies that there is no strong effect through *K* (assuming the DAG is correct). Most of the effect of urbanization is not mediated through impacts on children.

2. For the causal effect of children *K*, we can actually use exactly the same model as above. The adjustment set of *K* is *A* and *U*, and we will want to include *D* as well as a competing cause.

The K estimates, the `delta` parameters in the previous problem, are interesting. According to the DAG, the estimates for K should estimate its direct effect on C. The `delta` parameters are the incremental effects of each child (after the first). So the largest effect is the second child, which accounts for 74% (59%-87%) of the effect, which is about $1.26 \times 0.74 = 0.93$ on the log-odds scale, which corresponds to a more than doubling of the odds of contraception. The third and fourth have much smaller effects. Two is often a magic number in fertility studies.

3-OPTIONAL CHALLENGE. Again for reference, here's the DAG: Here's the DAG from lecture, for reference:

$$\begin{tikzcd}
A \arrow[r] \arrow[d] & C & D \arrow[l] \arrow[d] \\
K \arrow[ur, leftrightarrow] & & U \arrow[ul] \arrow[ll]
\end{tikzcd}$$

When we think about simulating from this causal model, we should obey the directionality of the arrows, but we can make it an agent-based model so that the effects of age make sense. That will also allow us to have reciprocal causation between contraception and children, as it must be. That's why I've made that arrow above have two heads. If C and K influence one another over a woman's life, then in any cross-sectional sample, the association will reflect a history of feedback.

The basic idea is to simulate a population of women (we can mostly ignore men, as is typical in biological simulations). Women are born and age, and there are probabilities at each age that they have a child and begin using contraception, conditional on things like where they live and how many children they already have. Since it's a simulation, we get to specify those probabilities.

In simulations like this, there are lots of complexities around the demographic dynamics. In real human populations, constant fertility rates and mortality rates imply some quasi-stable population size and distribution of ages. But real human populations don't have constant fertility rates and mortality rates. Everything is always in flux. I am not going to make a simulation that complicated. Instead I'll make one with constant fertility and mortality schedules, let it run long enough to reach its steady state distribution, and then we can harvest a cross-section from it.

The tricky bit in this type of simulation is how to keep track of the population. I will simulate in discrete year time steps, instead of on continuous time. But still people are being born and dying each year. So there is a lot of index management in these simulations to handle this population turnover. A lot of my code is just for that purpose.

Whatever you do, it is always better to pre-allocate some maximize population size, larger than you think you'll need, and then reuse that pre-allocation during the simulation. Do not grow or shrink the lengths of vectors during a simulation, because that is very slow. It is slow, because for the programming language it requires allocating a new vector with a different length and then copying from the previous one. Memory allocation (making new variables) is slow. So try to avoid it, so your code is faster.

Okay, with all that, here is my simulation code. Note that I specify a lot of the parameters as function arguments, so they are easy to change when you run the function.

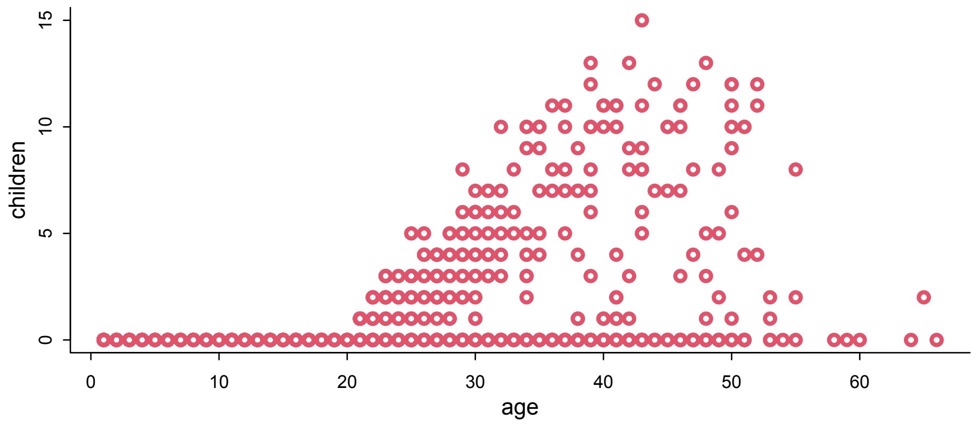
```
sim_fertility <- function(
    n_districts = 60, # number of districts
    p_urban = runif(n_districts,0.1,0.2), # init prob woman moves to city
    n_init = rep(100,n_districts), # init number of women in each district
    f = c(rep(0,20),rep(0.5,20),rep(0,100)), # fertility schedule by age
    m = c(0.2,rep(0.01,19),seq(from=0.01,to=0.5,len=100)), # mortality schedule
    # causal effects
    bKC = rep(0.01,20), # prob adopt C at each parity starting at zero
    bDC = runif(n_districts,0,0.1), # district offsets for prob adopt C
    bUC = 0.1, # effect of urban on prob use C at all parities
    # sim controls
    t_max = 1e2, # number of years to simulate population
    n_max = 1e4 # maximum population size across all districts
) {
    # init population
    # all women start at age 1
    # so need to run sim until reach stable age distribution
    n <- sum(n_init)
    age <- rep(NA,n_max)
    age[1:n] <- 1 # newborns
    D <- rep(NA,n_max) # districts
    D[1:n] <- rep(1:n_districts,times=n_init)
    K <- rep(NA,n_max)
    K[1:n] <- 0 # no kids yet
    U <- rep(NA,n_max)
    U[1:n] <- rbern(n,p_urban[D[1:n]])
    C <- rep(NA,n_max)
    C[1:n] <- rep(0,n) # no one starts using contraception

    # sim loop
    for ( i in 1:t_max ) {
        # loop over living women
        n_births <- 0
        for ( j in 1:n_max ) {
            if ( !is.na(age[j]) ) {
                # she's alive!
                # survive to next year?
                if ( runif(1) > m[age[j]] ) {
                    age[j] <- age[j] + 1 # get older
                    # adopt C?
                    if ( C[j]==0 ) {
                        pC <- bKC[K[j]+1] + bDC[D[j]] + bUC*U[j]
                        if ( pC > 1 ) pC <- 1
                        C[j] <- rbern(1,pC)
                    }
                    if ( C[j] ==0 ) {
                        # birth?
                        if ( runif(1) < f[age[j]] ) {
                            K[j] <- K[j] + 1
                            # is female?
                            if ( runif(1) < 0.5 ) {
                                # add to tally to update at end of this loop
                                n_births <- n_births + 1
                            }
                        }
                    }
                } else {
                    # death - remove from population
                    age[j] <- NA
                }
            }
        }#j

        # now add new women to population
        open_spots <- which(is.na(age))
        n_births <- min( n_births , length(open_spots) ) #bound population
        if ( n_births > 0 )
            for ( j in 1:n_births ) {
                k <- open_spots[j]
                age[k] <- 1
                C[k] <- 0
                D[k] <- sample(60,size=1)
                U[k] <- rbern(1, p_urban[D[k]] )
                K[k] <- 0
            }#j
    }#i

    out <- data.frame(
        A = age, U = U, C = C, K = K, D = D
    )

    # remove empty slots in population
    x <- which(!is.na(out$A))
    out <- out[x,]

    return(out)

}
```

Okay that's a lot. But really all it does is initialize a population of women and then simulate, in this order, survival, choice of contraception, reproduction. New births are added to the population only when they are girls, because boys don't matter. But we do count all children. The thing missing in this simulation is updating K when children die. That would require a lot more bookkeeping. It would be a good goal for version 2 of this project.

To run the simulation, just do:

```
sim_dat <- sim_fertility()
```

The default parameters I've chosen here result in a rather small stable population. Play around with the mortality and fertility rates to see how it changes the size of the population. Here's the age-fertility distribution for one run:



This is a cross-section, with many (but not all) of the features of a real population. The next step is to feed these data into our statistical model, to see whether it can identify features of the generative model. For example, in the simulate urban residents adopt contraception at higher rates. Can we see this in the sample? We can reuse our model from before. We just need to adjust the number of districts so they match the simulation.

```
dat <- list(
    C = sim_dat$C,
    D = sim_dat$D,
    nD = max(sim_dat$D),
    U = sim_dat$U
)

m3 <- ulam(
    alist(
        C ~ bernoulli(p),
        logit(p) <- a[D] + b[D]*U,
        # define effects using other parameters
        # this is the non-centered Cholesky machine
        transpars> vector[nD]:a <<- abar[1] + v[,1],
        transpars> vector[nD]:b <<- abar[2] + v[,2],
        transpars> matrix[nD,2]:v <-
            compose_noncentered( sigma , L_Rho , Z ),
        # priors - note that none have parameters inside them
        # that is what makes them non-centered
        matrix[2,nD]:Z ~ normal( 0 , 1 ),
        vector[2]:abar ~ normal(0,1),
        cholesky_factor_corr[2]:L_Rho ~ lkj_corr_cholesky( 4 ),
        vector[2]:sigma ~ exponential(1),
        # convert Cholesky to Corr matrix
        gq> matrix[2,2]:Rho <<- Chol_to_Corr(L_Rho)
    ) , data=dat , chains=4 , cores=4 )

precis(m3,depth=2,pars="abar")
```

```
        mean   sd 5.5% 94.5% rhat ess_bulk
abar[1] 0.94 0.15 0.71  1.18    1  1682.50
abar[2] 1.89 0.45 1.21  2.65    1  3052.28
```

The distribution of `abar[2]` is the offset for urban residents. It is positive, measuring the cumulative effect of urban residency. Right away we see that there is no parameter in the simulation that we can compare this value to, because the simulation only had per-year probabilities of adoption. So probably we should modify the statistical model to interact age with urban, so that we can get an effect of urban residency that is adjusted for age, the exposure time.

This is just one example of how developing a simulation helps us develop better estimators. Before analyzing the real data, loops of simulation and statistical development clarify thinking and even lead to better research design down the road.