

## *Exercises week 4*

*prof. Richard Torkar*

*Sept. 20, 2021*

It's like with the helmets. Once they added those in there were more head injuries. But less deaths.

Unknown

The weekly exercises are mainly from McElreath's notes which can be found on GitHub: [https://github.com/rmcelreath/stat\\_rethinking\\_2020/tree/main/homework](https://github.com/rmcelreath/stat_rethinking_2020/tree/main/homework)

EXERCISE 1. Consider the `data(Wines2012)` data table. These data are expert ratings of 20 different French and American wines by 9 different French and American judges. Your goal is to model score, the subjective rating assigned by each judge to each wine. I recommend standardizing it.

In this first problem, consider only variation among judges and wines. Construct index variables of judge and wine and then use these index variables to construct a linear regression model. Justify your priors. You should end up with 9 judge parameters and 20 wine parameters. Use `ulam()` instead of `quap()` to build this model, and be sure to check the chains for convergence. If you'd rather build the model directly in Stan or PyMC3 or Julia (Turing is a good choice!), go ahead. I just want you to use MCMC instead of quadratic approximation.

How do you interpret the variation among individual judges and individual wines? Do you notice any patterns, just by plotting the differences? Which judges gave the highest/lowest ratings? Which wines were rated worst/best on average?

EXERCISE 2. Now consider three features of the wines and judges:

`flight`: Whether the wine is red or white.

`wine.amer`: Indicator variable for American wines.

`judge.amer`: Indicator variable for American judges.

Use indicator or index variables to model the influence of these features on the scores. Omit the individual judge and wine index variables from Problem 1. Do not include interaction effects yet. Again use `ulam`, justify your priors, and be sure to check the chains. What do you conclude about the differences among the wines and judges? Try to relate the results to the inferences in Problem 1.

EXERCISE 3. Now consider two-way interactions among the three features. You should end up with three different interaction terms

in your model. These will be easier to build, if you use indicator variables. Again use `ulam`, justify your priors, and be sure to check the chains. Explain what each interaction means. Be sure to interpret the model's predictions on the outcome scale ( $\mu$ , the expected score), not on the scale of individual parameters. You can use `link` to help with this, or just use your knowledge of the linear model instead.

What do you conclude about the features and the scores? Can you relate the results of your model(s) to the individual judge and wine inferences from Exercise 1?

SOLUTION 1. Load the data and construct the variables we'll need:

```
library(rethinking)
data(Wines2012)
d <- Wines2012
dat_list <- list(
  S = standardize(d$score),
  jid = as.integer(d$judge),
  wid = as.integer(d$wine)
)
```

The model is straightforward. The only issue is the priors. Since I've standardized the outcome, we can use the ordinary  $\text{Normal}(0, 0.5)$  prior from the examples in the text with standardized outcomes. Then the prior outcomes will stay largely within the possible outcome space. A bit more regularization than that wouldn't be a bad idea either.

```
m1 <- ulam(
  alist(
    S ~ dnorm( mu , sigma ),
    mu <- a[jid] + w[wid],
    a[jid] ~ dnorm(0,0.5),
    w[wid] ~ dnorm(0,0.5),
    sigma ~ dexp(1)
  ), data=dat_list , chains=4 , cores=4 )
```

Since this is your first MCMC homework, we'll spend some time inspecting the chains to ensure they worked. First, the diagnostics that precis provides:

```
precis( m1 , 2 )
```

which provides this output,

	mean	sd	5.5%	94.5%	n_eff	Rhat4
a[1]	-0.27	0.20	-0.59	0.05	2496	1
a[2]	0.21	0.20	-0.11	0.55	2258	1
a[3]	0.21	0.20	-0.11	0.52	2005	1
a[4]	-0.54	0.19	-0.84	-0.23	2107	1
a[5]	0.80	0.19	0.48	1.10	1958	1
a[6]	0.48	0.19	0.17	0.79	2122	1
a[7]	0.14	0.20	-0.18	0.44	2137	1
a[8]	-0.65	0.20	-0.97	-0.33	2103	1
a[9]	-0.34	0.20	-0.66	-0.02	2280	1
w[1]	0.12	0.27	-0.32	0.52	2977	1
w[2]	0.08	0.25	-0.33	0.49	2760	1
w[3]	0.22	0.26	-0.21	0.63	2636	1
w[4]	0.46	0.25	0.04	0.86	2655	1
w[5]	-0.11	0.25	-0.48	0.29	2802	1
w[6]	-0.31	0.25	-0.71	0.10	2724	1
w[7]	0.24	0.25	-0.16	0.64	2551	1
w[8]	0.23	0.26	-0.18	0.66	2661	1
w[9]	0.07	0.27	-0.36	0.52	2892	1
w[10]	0.09	0.26	-0.32	0.52	2946	1
w[11]	-0.01	0.26	-0.42	0.41	2957	1
w[12]	-0.03	0.26	-0.43	0.38	2776	1
w[13]	-0.10	0.26	-0.52	0.32	3201	1
w[14]	0.00	0.26	-0.41	0.43	3115	1
w[15]	-0.19	0.26	-0.62	0.22	2235	1

```

w[16] -0.17 0.26 -0.58 0.24 2542 1
w[17] -0.12 0.26 -0.53 0.29 2773 1
w[18] -0.72 0.26 -1.14 -0.29 2640 1
w[19] -0.13 0.26 -0.54 0.27 2566 1
w[20] 0.33 0.26 -0.11 0.74 2465 1
sigma 0.85 0.05 0.77 0.93 2725 1

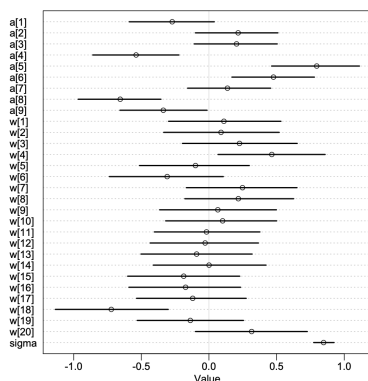
```

The `n_eff` values are all actually higher than the number of samples (2000), and all the `Rhat` values at exactly 1. Looks good so far. These diagnostics can mislead, however, so let's look at the trace plots too using `traceplot(m1)` (an example of well-mixed chains can be seen to the right).

You'll see that the traceplots pass the hairy-caterpillar-ocular-inspection-test: all the chains mix in the same region, and they move quickly through it, not getting stuck anywhere.

Now let's plot these parameters so they are easier to interpret:

```
plot( precis( m1 , 2 ) )
```



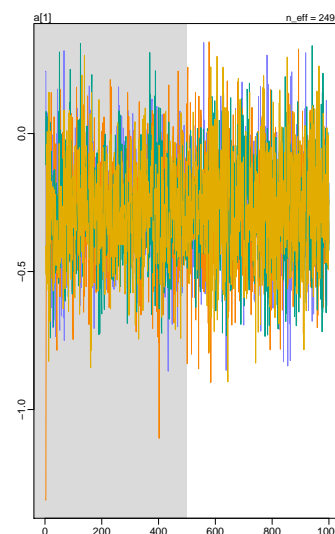
The `a` parameters are the judges. Each represents an average deviation of the scores. So judges with lower values are harsher on average. Judges with higher values liked the wines more on average. There is some noticeable variation here. It is fairly easy to tell the judges apart. The `w` parameters are the wines. Each represents an average score across all judges. Except for wine 18, there isn't that much variation. These are good wines, after all. Overall, there is more variation from judge than from wine!

**SOLUTION 2.** The easiest way to code the data is to use indicator variables. Let's look at that approach first. I'll do an index variable version next. I'll use the three indicator variables `W` (NJ wine), `J` (American NJ), and `R` (red wine).

```

dat_list2 <- list(
  S = standardize(d$score),
  W = d$wine.amer,

```



```
J = d$judge.amer,
R = ifelse(d$flight=="red", 1L, 0L)
)
```

The model structure is just a linear model with an ordinary intercept. I'll put a relatively tight prior on the intercept, since it must be near zero (centered outcome). What about the coefficients for the indicator variables? Let's pretend we haven't already seen the results from Problem 1—there aren't any big wine differences to find there. Without that cheating foresight, we should consider what the most extreme effect could be. How big could the difference between NJ and French wines be? Could it be a full standard deviation? If so, then maybe a  $\text{Normal}(0, 0.5)$  prior makes sense, since they place a full standard deviation difference out in the tails of the prior. I'd personally be inclined to something even tighter, so that it regularizes more. But let's go with these wide priors, which nevertheless stay within the outcome space. It would make even more sense to put a tighter prior on the difference between red and white wines—on average they should be the no different, because judges only compare within flights. Here's the model:

```
m2a <- ulam(
  alist(
    S ~ dnorm( mu , sigma ),
    mu <- a + bW*W + bJ*J + bR*R,
    a ~ dnorm( 0 , 0.2 ),
    c(bW,bJ,bR) ~ dnorm( 0 , 0.5 ),
    sigma ~ dexp(1)
  ), data=dat_list2 , chains=4 , cores=4 )
precis( m2a )
```

which provides the following output,

	mean	sd	5.5%	94.5%	n_eff	Rhat4
a	-0.02	0.12	-0.21	0.18	1523	1
bR	0.00	0.14	-0.22	0.22	1752	1
bJ	0.22	0.13	0.00	0.44	1685	1
bW	-0.17	0.14	-0.40	0.05	1755	1
sigma	1.00	0.05	0.92	1.09	1728	1

As expected, red and wines are on average the same—bR is right on top of zero. American judges seem to be, on average, slightly more generous with ratings—bJ is slightly but reliably above zero. American wines have slightly lower average ratings than French wines—bW is mostly below zero, but not very large in absolute size.

Okay, now for an index variable version. The thing about index variables is that you can easily end up with more parameters than in an equivalent indicator variable model. But it's still the same posterior distribution. You can convert from one to the other (if the priors are also equivalent). We'll need three index variables:

```
dat_list2b <- list(
  S = standardize(d$score),
```

```

wid = d$wine.amer + 1L,
jid = d$judge.amer + 1L,
fid = ifelse(d$flight=="red",1L,2L)
)

```

Now `wid` is 1 for a French wine and 2 for a NJ wine, `jid` is 1 for a French judge and 2 for an American judge, and `fid` is 1 for red and 2 for white. Those 1L numbers are just the R way to type the number as an integer—"1L" is the integer 1, while "1" is the real number 1. We want integers for an index variable.

Now let's think about priors for the parameters that correspond to each index value. Now the question isn't how big the difference could be, but rather how far from the mean an indexed category could be. If we use `Normal(0,0.5)` priors, that would make a full standard deviation difference from the global mean rare. It will also match what we had above, in a crude sense. Again, I'd be tempted to something narrow, for the sake of regularization. But certainly something like `Normal(0,10)` is flat out silly, because it makes impossible values routine. Let's see what we get:

```

m2b <- ulam(
  alist(
    S ~ dnorm( mu , sigma ),
    mu <- w[wid] + j[jid] + f[fid],
    w[wid] ~ dnorm( 0 , 0.5 ),
    j[wid] ~ dnorm( 0 , 0.5 ),
    f[wid] ~ dnorm( 0 , 0.5 ),
    sigma ~ dexp(1)
  ), data=dat_list2b , chains=4 , cores=4 )
precis( m2b , depth = 2)

```

and the output is,

	mean	sd	5.5%	94.5%	n_eff	Rhat4
w[1]	0.12	0.30	-0.38	0.59	783	1
w[2]	-0.07	0.30	-0.55	0.42	686	1
j[1]	-0.13	0.31	-0.63	0.35	769	1
j[2]	0.11	0.31	-0.39	0.60	733	1
f[1]	-0.01	0.29	-0.46	0.46	873	1
f[2]	0.00	0.29	-0.46	0.47	845	1
sigma	1.00	0.05	0.92	1.09	1133	1

To see that this model is the same as the previous, let's compute contrasts. The contrast between American and French wines is:

```

post <- extract.samples(m2b)
diff_w <- post$w[,2] - post$w[,1]
precis( diff_w )

```

which gives the output to the right.


That's almost exactly the same mean and standard deviation as `bw` in the first model. The other contrasts match as well, but please check yourself.

Something to notice about the two models is that the second one does sample less efficiently. The `n_eff` values are lower. This isn't a

```

'data.frame': 2000 obs. of 1 variables:
  mean  sd  5.5% 94.5% histogram
diff_w -0.19 0.15 -0.43 0.06

```



problem in this case, but it is a consequence of the higher correlations in the posterior, a result of the redundant parameterization. If you look at the `pairs(m2b)` plot, you'll see tight correlations for each pair of index parameters of the same type. This is because really it is a difference that matters, and many combinations of two numbers can produce the same difference. But the priors keep this from ruining our inference. If you tried the same thing without priors, it would likely fall apart and return very large standard errors.

**SOLUTION 3.** I'll use the indicator variable approach here, because it'll be much easier. Once you start using MCMC in the next chapter, it'll be possible to define very flexible parameter structures. Then the index approach will be easy again.

For the indicator approach, we can use the same predictor variables as before:

```
dat_list2 <- list(
  S = standardize(d$score),
  W = d$wine.amer,
  J = d$judge.amer,
  R = ifelse(d$flight=="red", 1L, 0L)
)
```

It's the model that is different.

```
m3 <- quap(
  alist(
    S ~ dnorm( mu , sigma ),
    mu <- a + bW*W + bJ*J + bR*R +
          bWJ*W*J + bWR*W*R + bJR*J*R,
    a ~ dnorm(0,0.2),
    c(bW,bJ,bR) ~ dnorm(0,0.5),
    c(bWJ,bWR,bJR) ~ dnorm(0,0.25),
    sigma ~ dexp(1)
  ), data=dat_list2 )
```

I used the same priors as before for the main effects. I used tighter priors for the interactions. Why? Because interactions represent sub-categories of data, and if we keep slicing up the sample, differences can't keep getting bigger. Again, the most important thing is not to use flat priors like  $\text{Normal}(0,10)$  that produce impossible outcomes.

Let's look to the right and see what `precis` gives us.

```
precis(m3)
```

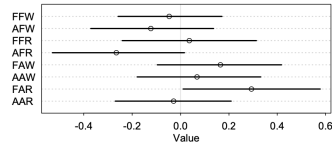
Reading the parameters this way is not easy. But right away you might notice that `bW` is now close to zero and overlaps it a lot on both sides. NJ wines are no longer on average worse. So the interactions did something. Glancing at the interaction parameters, you can see that only one of them has much mass away from zero, `bWR`, the interaction between NJ wines and red flight, so red NJ wines. To get the predicted scores for red and white wines from both NJ and France, for both types of judges, we can use `link`:

	mean	sd	5.5%	94.5%
a	-0.05	0.13	-0.25	0.16
bW	-0.07	0.17	-0.35	0.20
bJ	0.21	0.18	-0.08	0.49
bR	0.09	0.18	-0.21	0.38
bWJ	-0.02	0.18	-0.31	0.27
bWR	-0.23	0.18	-0.52	0.06
bJR	0.05	0.18	-0.24	0.34
sigma	0.98	0.05	0.89	1.06

```

pred_dat <- data.frame(
  W = rep( 0:1 , times=4 ),
  J = rep( 0:1 , each=4 ),
  R = rep( c(0,0,1,1) , times=2 )
)
mu <- link( m3 , data=pred_dat )
row_labels <- paste( ifelse(pred_dat$W==1,"A","F") ,
  ifelse(pred_dat$J==1,"A","F") ,
  ifelse(pred_dat$R==1,"R","W") , sep="" )
plot( precis( list(mu=mu) , 2 ) , labels=row_labels )

```



I've added informative labels. FFW means: French wine, French judge, White wine. So the first four rows are as judged by French judges. The last four are as judged by American judges. The two rows that jump out are the 4th and the 2nd-to-last, AFR and FAR. Those are NJ red wines as judged by French judges and French red wines as judged by American judges. French judges didn't like NJ reds so much (really only one NJ red, if you look back at Problem 1). And American judges liked French reds more. Besides these two interactions, notice that it is very hard to figure this out from the table of coefficients.