HW3

Isaac Kleisle-Murphy

2/7/2022

8.8

a.)

Since there are $K = 5$ response categories, we include $K - 1 = 4$ intercepts – one for categories 1, through 4, with the fifth omitted and thus serving as a baseline (lest the model be overparameterized). This is akin to the baseline category in multinomial logit regression (which can be derived off of this setup). Hence, looking at the table, we see that the model is, for cumulative probabilities and $j = 1, \dots, 5$

$$g(P(Response \leq j)) = \left(\sum_{i=1}^{4} \alpha_i \cdot \delta(j = i)\right) + \beta_g \cdot \delta(female) + \beta_\ell \cdot \delta(rural) + \beta_s \delta(no\ seatbelt) + \beta_{\ell,s} \delta(rural) \cdot \delta(no\ seatbelt)$$

for logit g. Note that the remaining features are indicators, with males, urban, and yes seatbelt all baselines (i.e. indicator switched on for not those categories); α_j's correspond to intercepts, and β's correspond to feature/indicator coefficients. Now consider what this looks like for males ($\delta(female) = 0$)) in urban locations ($\delta(rural) = 0$) in seat belts ($\delta(no\ seatbelt) = 0$) – i.e. when all indicators have been switched off. Probabilities are then (σ is the sigmoid)

- $j = 1 : \sigma(\alpha_1) = 1/(1 + exp(-3.3074)) \approx .965$
- $j = 2 : \sigma(\alpha_2) = 1/(1 + exp(-3.4818)) \approx .970$
- $j = 3 : \sigma(\alpha_3) = 1/(1 + exp(-5.3494)) \approx .995$
- $j = 4 : \sigma(\alpha_4) = 1/(1 + exp(-7.2563)) \approx .9993$
- $j = 5 : 1$

We can then subtract sequentially to get probabilities (j=1 to 5, in that order) of $(.965 - 0, .970 - .965, .995 - .970, .9993 - .995, 1 - .9993) = (.965, .005, .025, .0043, .007))$.

In other words, the intercepts dictate the baseline probabilities, where the baseline is constructed to be male/urban/seatbelted.

b.)

An estimate/SD in the table assume everything else held constant, so we can extracting the female row of the table allows us to answer the question at hand. We have estimate -.5463 and SD .0272, and hence a log-space CI of $-.5463 \pm 1.96 \cdot .0272 = [-0.599612, -0.492988]$, which exponentiates to 0.54902460.6107986. That is, given urban vs. rural location and seat belt vs. not choice, we are 95% confident that the odds of injury below any of the five specified levels for a female passenger sit between .549 and .622 times that for a male.

c.)

First, observe that for rural locations without a seatbelt, the interaction effect is -.1244 in the log space. First, plugging in indicators $\delta(rural = 1)$ for the rural setting, we have an odds ratio of (the gender terms, as they are given and un-interacted, simply cancel out):

$$OR = \frac{P(R \leq j|\delta(no\ seatbelt) = 1, \delta(rural) = 1)P(R > j|\delta(no\ seatbelt) = 0, \delta(rural) = 1)}{P(R \leq j|\delta(no\ seatbelt) = 0, \delta(rural) = 1)P(R > |\delta(no\ seatbelt) = 1, \delta(rural) = 1)} = \frac{\sigma(\alpha_j + \beta_s \cdot 1 + \beta_\ell \cdot 1 + \beta_{s,\ell} \cdot 1}{\sigma(\alpha_j + \beta_s \cdot 0 + \beta_\ell \cdot 1 + \beta_{s,\ell} \cdot 0 \cdot}$$

For the urban odds ratio, the above holds, however the $\beta_{s,\ell}$ is always switched off, as $\delta(rural) = 0$ and the interaction is always zeroed. Hence, the algebra from above holds, but the $\beta_{s,\ell}$ is always zero'd out, and the odds ratio is thus

$$\exp(-.7602 - 0) \approx 0.4675729.$$

In other words, the interaction term has a multiplicative effect of $\exp(-.1244)$ on the odds ratios, in moving from the urban case to the rural case (given gender).

8.10

```
require(dplyr)
require(VGAM)
```

```
## Loading required package: VGAM
```

```
## Loading required package: stats4
```

```
## Loading required package: splines
```

```
#' using Prof. Taylors vglm package

data =c(
  1, 1, 28, 45, 29, 26,
  1, 0, 4, 12, 5, 2,
  0, 1, 41, 44, 20, 20,
  0, 0, 12, 7, 3, 1
) %>%
  matrix(., byrow=T, ncol=6) %>%
  data.frame() %>%
  `colnames<-`(c("therapy", "gender", "prog", "no_ch", "partial", "complete"))

# from slides
vfit = VGAM::vglm(
  cbind(prog, no_ch, partial, complete) ~ therapy + gender,
  data=data,
  family=cumulative(parallel=T)
)
```

a.)

```
summary(vfit)
```

```
##
## Call:
## VGAM::vglm(formula = cbind(prog, no_ch, partial, complete) ~
##     therapy + gender, family = cumulative(parallel = T), data = data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  -0.1960     0.2947  -0.665  0.50605
## (Intercept):2   1.3713     0.3059   4.482 7.38e-06 ***
## (Intercept):3   2.4221     0.3276   7.393 1.43e-13 ***
## therapy        -0.5807     0.2119  -2.741  0.00613 **
## gender         -0.5414     0.2953  -1.834  0.06671 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]), logitlink(P[Y<=2]),
## logitlink(P[Y<=3])
##
## Residual deviance: 5.5677 on 7 degrees of freedom
##
## Log-likelihood: -25.5417 on 7 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
##   therapy    gender
## 0.5595152 0.5819358
```

As detailed in the summary, the baseline is the final category (see docs), i.e. complete or complete remission. The fit then spits out exponentiated coefficients for therapy (indicator switched on for sequential, switched off for alternating) and gender (indicator on for male, off for female). The interpretation here is as follows: given all else, the odds of any sort of response (partial remission through progressive disease) for sequential (`therapy = 1`) therapy are .5595 times those for alternating (`therapy = 0`); likewise, the odds of any sort of response for males are .5819 times those of females, all else given. This casts well on the sequential therapy, as the estimated odds of any sort of response decrease with it – identically, the odds of a complete remission increase with it.

b.)

Suppose we collapse the remission categories and disease (same or worsening), i.e.

```
data_ =c(
  1, 1, 28 + 45, 29 +  26,
  1, 0, 4 +  12, 5 + 2,
  0, 1, 41 + 44, 20 +  20,
  0, 0, 12 + 7, 3 + 1
```

```
) %>%
  matrix(., byrow=T, ncol=4) %>%
  data.frame() %>%
  `colnames<-`(c("therapy", "gender", "same_or_worse", "any_remission"))

# from slides
vfit_ = VGAM::vglm(
  cbind(same_or_worse, any_remission) ~ therapy + gender,
  data=data_,
  family=cumulative(parallel=T)
)

summary(vfit_)
```

```
##
## Call:
## VGAM::vglm(formula = cbind(same_or_worse, any_remission) ~ therapy +
##     gender, family = cumulative(parallel = T), data = data_)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.4250     0.3744   3.806 0.000141 ***
## therapy      -0.5022     0.2457  -2.044 0.040949 *
## gender       -0.6543     0.3715  -1.761 0.078171 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Name of linear predictor: logitlink(P[Y<=1])
##
## Residual deviance: 0.1192 on 1 degrees of freedom
##
## Log-likelihood: -8.5384 on 1 degrees of freedom
##
## Number of Fisher scoring iterations: 3
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
##   therapy    gender
## 0.6051969 0.5197993
```

Now, the odds of same-or-worsening under sequential are – given all else – about .61 times those under alternating. The odds of same-or-worsening as a male – given all else – are about .52 those for a female.

Broadly, the therapy coefficients across both models are similar. As set forth by Agresti 8.2.2, this reflects the collapsibility property of such cumulative logit models.

c.)

Unde collapsing, we have (lookup in tables above) $\hat{\beta}_1/SE_1 = \frac{-.5022}{2.457} = -2.044.$, whereas uncollapsed we have $\hat{\beta}_1/SE_1 = \frac{-.5807}{.2119} = -2.740.$. Indeed, this implies the significance of the effect has decreased (against a null

hypothesis of $beta_1 = 0$; we're just z-scoring). Hence, a Wald test might be less likely to reject and find significance of the effect/treatment.

d.)

We fit the interacted model

```
vfit_interact = VGAM::vglm(
  cbind(prog, no_ch, partial, complete) ~ therapy * gender,
  data=data,
  family=cumulative(parallel=T)
)
summary(vfit_interact)
```

```
##
## Call:
## VGAM::vglm(formula = cbind(prog, no_ch, partial, complete) ~
##     therapy * gender, family = cumulative(parallel = T), data = data)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept):1   0.07702    0.39861   0.193   0.8468
## (Intercept):2   1.64839    0.41023   4.018 5.86e-05 ***
## (Intercept):3   2.69784    0.42603   6.332 2.41e-10 ***
## therapy        -1.07848    0.54980  -1.962   0.0498 *
## gender         -0.86461    0.43087  -2.007   0.0448 *
## therapy:gender  0.59041    0.59353   0.995   0.3199
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]), logitlink(P[Y<=2]),
## logitlink(P[Y<=3])
##
## Residual deviance: 4.5209 on 6 degrees of freedom
##
## Log-likelihood: -25.0183 on 6 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
##        therapy         gender therapy:gender
##      0.3401121      0.4212172      1.8047266
```

Functionally, the interacted model does not improve the fit much: whereas the uninteracted model had a residual deviance of 5.5677 on 7 degrees of freedom, this new fit has a residual deviance of 4.5209 on 6 degrees of freedom. This gives us a drop-in-deviance test of (5.5677 - 4.5209) = 1.0468 on a single degree of freedom, the p-value for such a test is:

```
pchisq(5.5677 - 4.5209, 1, lower.tail = F)
```

```
## [1] 0.3062452
```

so we would fail to reject a null hypothesis that any interaction is zero; hence we would likely continue to assume the interaction is zero, and stick with the uninteracted model.

Consider what happens in the four variable setup. If we arbitrarily choose a baseline of alternating with gender female, and then define levels of (Alt. x Female), (Seq. x Female), (Seq. x Male), our regression will effectively boil down to: we will have

$$\beta_1 \cdot \delta(\text{Alt. x Male}) + \beta_2 \cdot \delta(\text{Seq. x Female}) + \beta_3 \cdot \delta(\text{Seq. x Male})$$

Observe that this is functionally equivalent – i.e. it corresponds 1:1 – to

$$\beta_1 \cdot \delta(Male) + \beta_2 \delta(Seq) + \beta_4 \cdot \delta(Male) \cdot \delta(Seq)$$

where $\beta_3 = \beta_1 + \beta_2 + \beta_4$.

8.17

Just as in the previous problem, the data sets up nicely for a cumulative logit model, as the cholesterol buckets impose a nice ordinal structure. We're instructed to use beginning state and treatement/not as covariates; I'll go additive for simplicity, but an interaction (as in the previous problem) would be fair game as well.

```
# groups
# 1 : 3.4;
# 2 : 3.4-4.1;
# 3 : 4.1-4.9;
# 4: d > 4.9

data = c(
    0, 1, 18, 8, 0, 0,
    0, 2, 16, 30, 13, 2,
    0, 3, 0, 14, 28, 7,
    0, 4, 0, 2, 15, 22,
    1, 1, 21, 4, 2, 0,
    1, 2, 17, 25, 6, 0,
    1, 3, 11, 35, 36, 6,
    1, 4, 1, 5, 14, 12
  ) %>%
  matrix(byrow=T, ncol=6) %>%
  data.frame() %>%
  `colnames<-`(
    c("is_treatment", "begin_group", "group_1",
      "group_2", "group_3", "group_4")
    )

vfit_ldl_1 = VGAM::vglm(
  cbind(group_1, group_2, group_3, group_4) ~ is_treatment + begin_group,
  data=data,
  family=cumulative(parallel=T)
```

```
)

vfit_ldl_2 = VGAM::vglm(
  cbind(group_1, group_2, group_3, group_4) ~ begin_group + is_treatment,
  data=data %>% mutate(begin_group = as.character(begin_group)),
  family=cumulative(parallel=T)
)
summary(vfit_ldl_1)
```

```
##
## Call:
## VGAM::vglm(formula = cbind(group_1, group_2, group_3, group_4) ~
##     is_treatment + begin_group, family = cumulative(parallel = T),
##     data = data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1   2.5507     0.3435   7.427 1.11e-13 ***
## (Intercept):2   4.8517     0.4113  11.795  < 2e-16 ***
## (Intercept):3   7.3108     0.5046  14.489  < 2e-16 ***
## is_treatment    0.8017     0.2066   3.880 0.000105 ***
## begin_group    -1.8722     0.1441 -12.992  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]), logitlink(P[Y<=2]),
## logitlink(P[Y<=3])
##
## Residual deviance: 14.543 on 19 degrees of freedom
##
## Log-likelihood: -39.066 on 19 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
## is_treatment  begin_group
##    2.2292601    0.1537787
```

Looking at the exponentiated coefficients, we have the following interpretation: given all else and for any level of the four ordered cholesterol levels, the odds of having an ending cholesterol less than or equal to that level among treated patients are about 2.23 times those among control patients.

b.)

```
summary(vfit_ldl_2)
```

```
##
```

```
## Call:
## VGAM::vglm(formula = cbind(group_1, group_2, group_3, group_4) ~
##     begin_group + is_treatment, family = cumulative(parallel = T),
##     data = data %>% mutate(begin_group = as.character(begin_group)))
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1   0.6738     0.3271   2.060 0.039412 *
## (Intercept):2   2.9648     0.3694   8.026 1.01e-15 ***
## (Intercept):3   5.4379     0.4223  12.878  < 2e-16 ***
## begin_group2   -1.8749     0.3701  -5.066 4.06e-07 ***
## begin_group3   -3.6972     0.3958  -9.340  < 2e-16 ***
## begin_group4   -5.6441     0.4661 -12.110  < 2e-16 ***
## is_treatment    0.7924     0.2097   3.778 0.000158 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]), logitlink(P[Y<=2]),
## logitlink(P[Y<=3])
##
## Residual deviance: 14.4679 on 17 degrees of freedom
##
## Log-likelihood: -39.0285 on 17 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
## begin_group2 begin_group3 begin_group4 is_treatment
##  0.153370198  0.024793796  0.003538291  2.208756477
```

Looking at the exponentiated coefficients, we have the following interpretation: given all else and for any level of the four ordered cholesterol levels, the odds of having an ending cholesterol less than or equal to that level among treated patients are about 2.21 times those among control patients. This is a remarkably close estimate to that obtained in part a, suggesting there may be little difference between treating begin group ordinally or as a factor. We largely make the same inference about the effect of treatment.

8.23

First, we fit the uninformative prior, i.e. $\alpha, \beta \sim N(0, \sigma = 100)$.

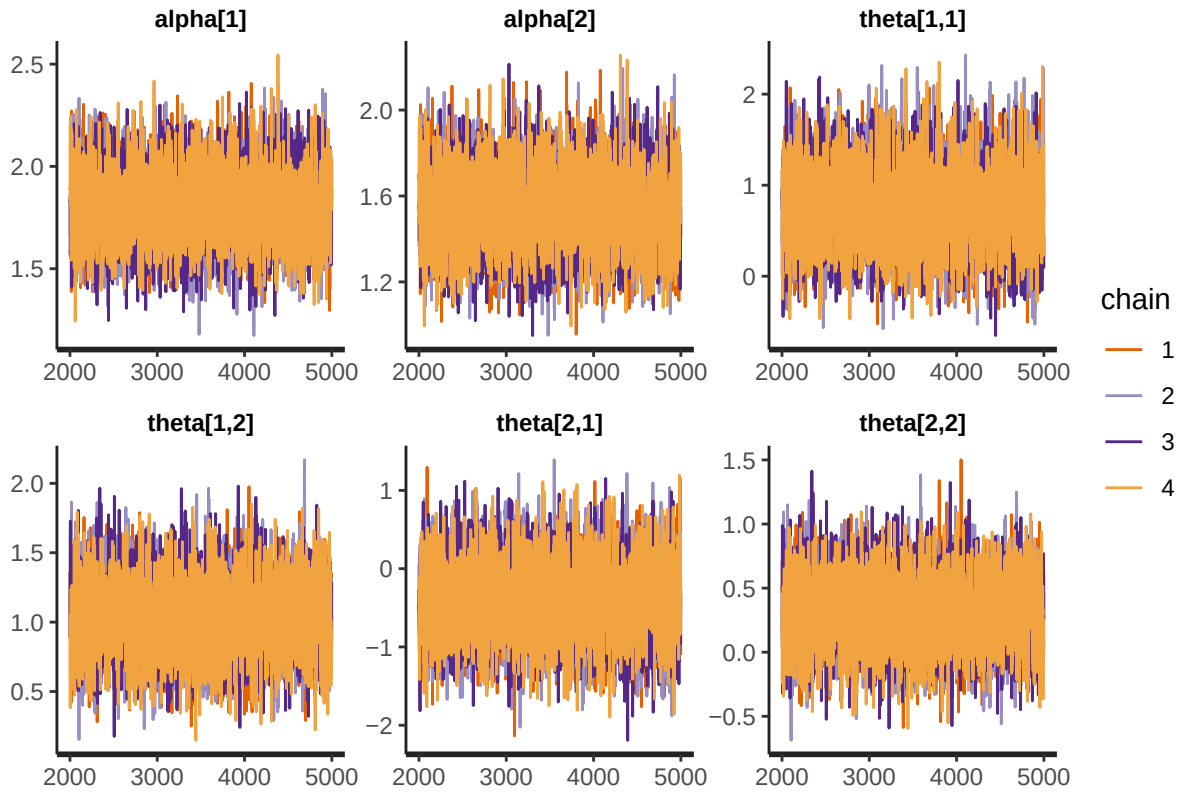
```
suppressMessages(library(rstan))
suppressMessages((library(bayesplot)))

# thank you Saskia for copying over the data
data = list(
  # race
  X1 = c(
    rep(1, 88+16+2), rep(1, 54+7+5), rep(0, 397+141+24), rep(0, 235+189+39)
  ),
  # gender
```

```
  X2 = c(
    rep(1, 88+16+2), rep(0,54+7+5), rep(1,397+141+24), rep(0, 235+189+39)
  ),
  # belief
  Y = c(
    rep(1, 88), rep(2, 16), rep(3, 2), #1=yes, 2=unsure, 3=no
    rep(1, 54), rep(2, 7), rep(3, 5),
    rep(1, 397), rep(2, 141), rep(3, 24),
    rep(1, 235), rep(2, 189), rep(3,39)
  )
)
data$P = 3
data$N = length(data$Y)
data$K = 3 - 1


stan_model = "
data { // data block: observed variables names, types, sizes
  int<lower=1> N;
  int<lower=1> K;
  int<lower=1> P;
  real<lower=0> sigma_alpha;
  real<lower=0> sigma_theta;
  vector[N] X1; // race
  vector[N] X2; // gender
  int<lower=1, upper=3> Y[N];
}
parameters { // unobserved variables (name, type, size)
  vector[K] alpha; // intercept
  matrix[K, 2] theta; // coefs
  //vector[K] theta1;
  //vector[K] theta2;
}
model {
  vector[K+1] eta; // TIL you can define stuff like this iter-wise
  alpha ~ normal(0, sigma_alpha);
  theta[1:K, 1] ~ normal(0, sigma_theta);
  theta[1:K, 2] ~ normal(0, sigma_theta);
  //theta1 ~ normal(0, sigma_theta);
  //theta2 ~ normal(0, sigma_theta);

  for (i in 1:N){ // N outcome values, w/ N corresponding parameter values
    for (j in 1:K){
      eta[j] = alpha[j] + theta[j, 1]*X1[i] + theta[j, 2]*X2[i];
     //eta[j] = alpha[j] + theta1[j]*X1[i] + theta2[j]*X2[i];
    }
    eta[K+1] = 0; // baseline
    Y[i] ~ categorical_logit(eta); // from slides
  }
}"

# sample: NUTS
stan_fit_1 = stan(
```

```
  model_code=stan_model,
  data=append(
    data,
    list(sigma_alpha=100., sigma_theta=100.)
  ),
  chains=4,
  iter=5000,
  warmup=2000,
  thin=1,
  seed=100
)
```

Trace:

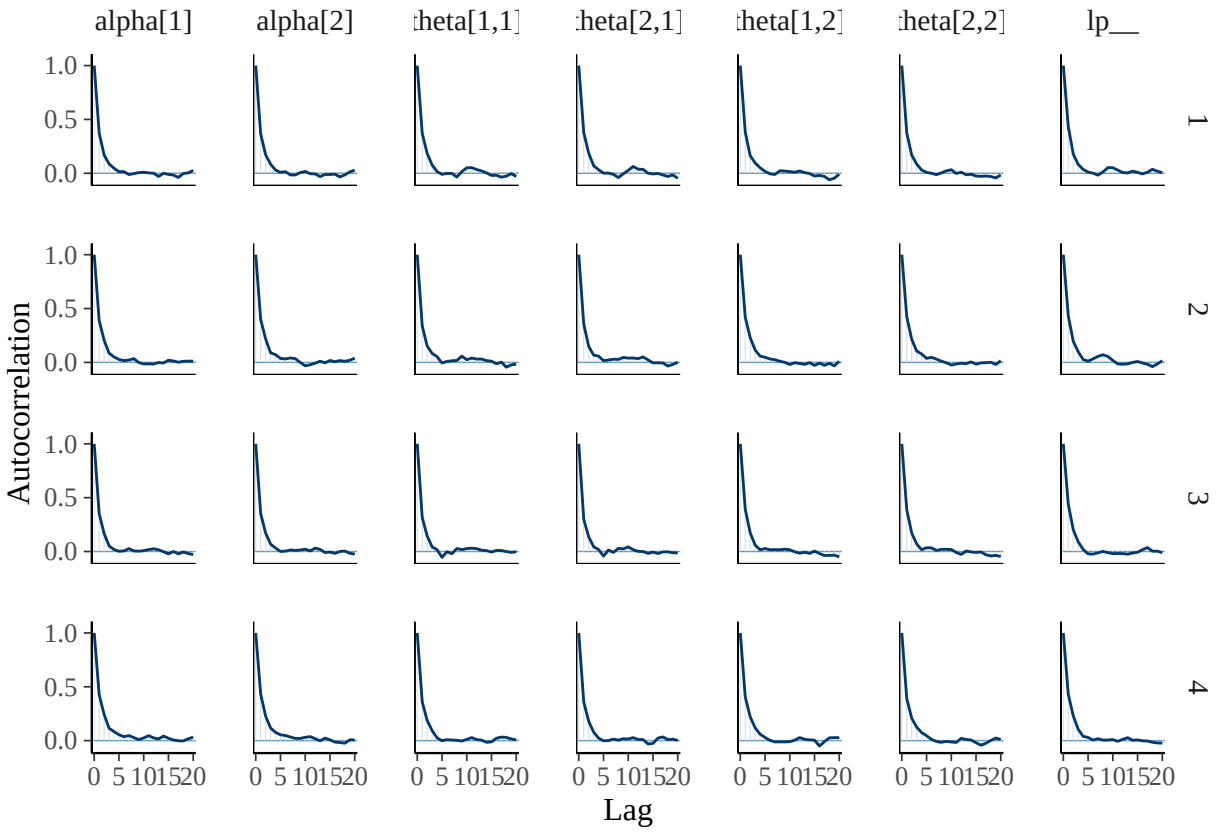
```
stan_trace(stan_fit_1)
```



Autocorrelation:

```
mcmc_acf(stan_fit_1, eval=FALSE)
```

```
## Warning: The following arguments were unrecognized and ignored: eval
```



Looking at traceplots and autocorrelations, convergence passes muster. Posterior parameters are summarized as follows:

```
summary(stan_fit_1)$summary
```

```
##                   mean     se_mean        sd         2.5%          25%
## alpha[1]      1.8036385 0.002549095 0.1704029    1.4804563    1.6877515
## alpha[2]      1.5400749 0.002601180 0.1738637    1.2103200    1.4205425
## theta[1,1]    0.7346790 0.005748433 0.4209829   -0.0344862    0.4422243
## theta[1,2]    1.0382527 0.003858210 0.2603955    0.5378338    0.8637058
## theta[2,1]   -0.4315445 0.006486733 0.4630389   -1.2982529   -0.7492013
## theta[2,2]    0.3112586 0.004003428 0.2707089   -0.2037328    0.1267926
## lp__       -933.9119786 0.025997989 1.7557219 -938.1301027 -934.8746747
##                    50%          75%        97.5%    n_eff     Rhat
## alpha[1]      1.7984994    1.9195051    2.1524122 4468.705 1.000754
## alpha[2]      1.5371634    1.6550034    1.8859748 4467.629 1.000638
## theta[1,1]    0.7159359    1.0047589    1.6200980 5363.272 1.000174
## theta[1,2]    1.0358074    1.2125758    1.5623270 4555.074 1.000370
## theta[2,1]   -0.4479603   -0.1272903    0.5221396 5095.454 1.000323
## theta[2,2]    0.3070692    0.4961187    0.8423948 4572.368 1.000281
## lp__       -933.5816987 -932.6166319 -931.5167904 4560.705 1.000002
```

Here, `beta[1, 2]` is our parameter of interest, as it corresponds to `Gender = Yes = Female = j = 1`. We we have posterior [q2.5, mean, q97.5]

```
summary(stan_fit_1)$summary[4, c(4, 1, 8)]
```

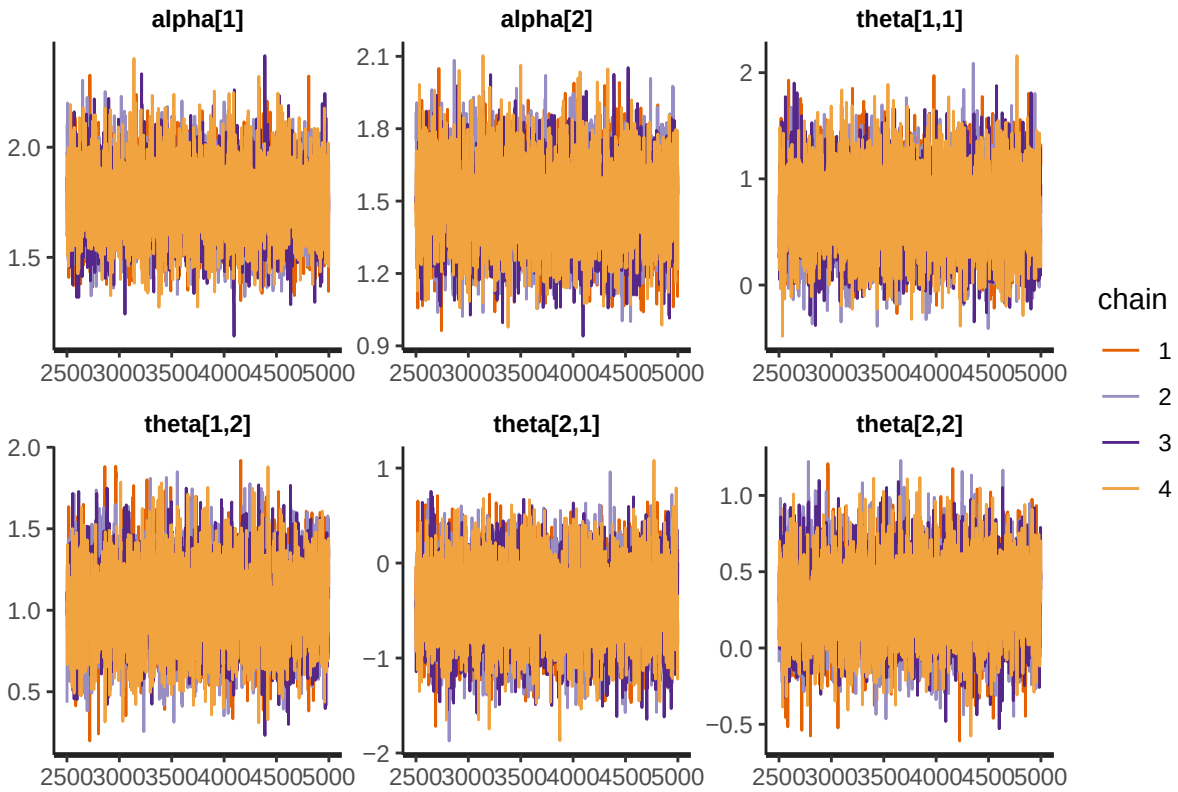
```
##      2.5%      mean     97.5%
## 0.5378338 1.0382527 1.5623270
```

Next, we attempt a fit, albeit with standard Gaussian priors, i.e. $\alpha, \beta \sim N(0, \sigma = 1)$.

```
# sample: NUTS
stan_fit_2 = stan(
  model_code=stan_model,
  data=append(
    data,
    list(sigma_alpha=1., sigma_theta=1.)
  ),
  chains=4,
  iter=5000,
  warmup=2500,
  thin=1,
  seed=123
)
```

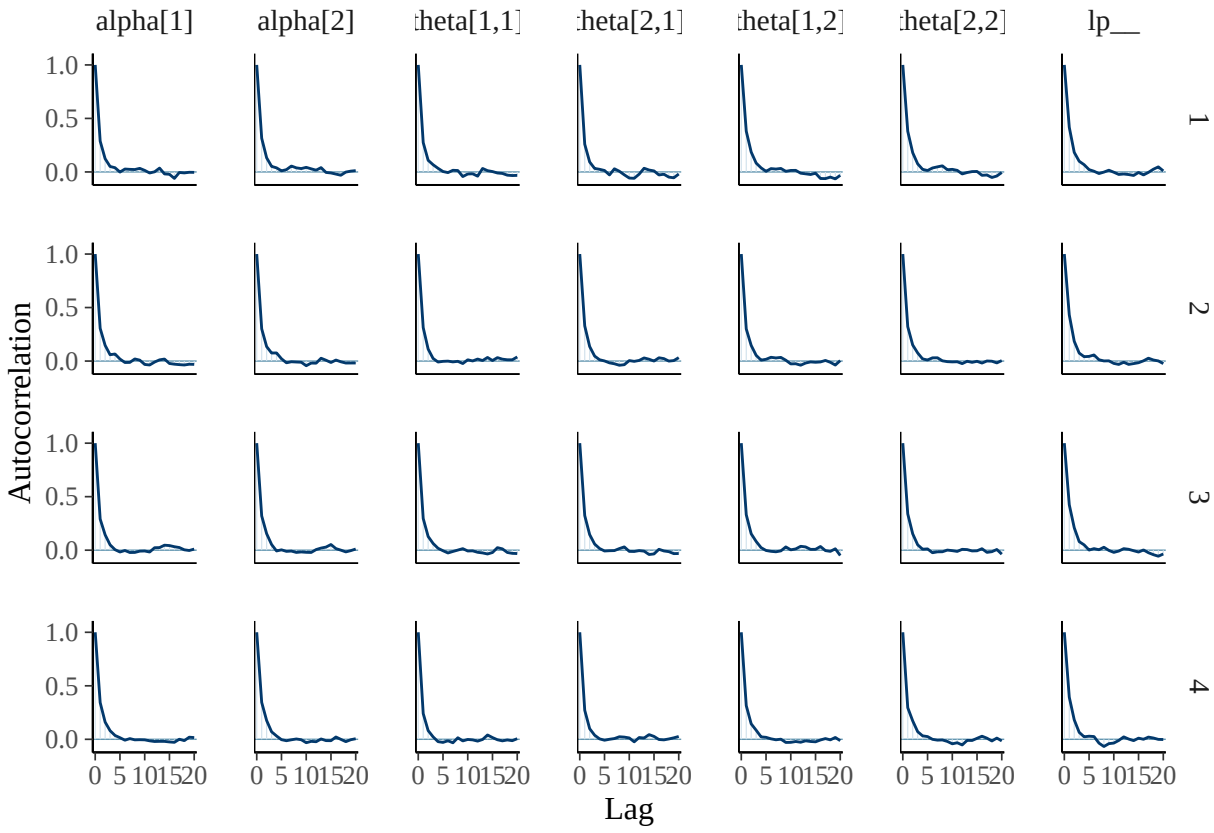
Again, all good on convergence check. Trace:

```
stan_trace(stan_fit_2)
```



Autocorrelation:

```
mcmc_acf(stan_fit_2)
```



Now, our posterior q2.5/mean/q97.5/SD is:

```
summary(stan_fit_2)$summary[4, c(4, 1, 8, 3)]
```

```
##      2.5%      mean     97.5%        sd
## 0.5827874 1.0376281 1.5070288 0.2368063
```

To recap, we have the following q2.5/mean/q97.5 tuples across all three of our models:

i.) Uninformative ($\sim N(0, 100)$): q2.5/mean/q97.5/SD:

```
summary(stan_fit_1)$summary[4, c(4, 1, 8, 3)]
```

```
##      2.5%      mean     97.5%        sd
## 0.5378338 1.0382527 1.5623270 0.2603955
```

```
summary(stan_fit_2)$summary[4, c(4, 1, 8, 3)]
```

ii.) Standard Gaussian Prior ($\sim N(0, 1)$):

```
##      2.5%      mean     97.5%        sd
## 0.5827874 1.0376281 1.5070288 0.2368063
```

Posterior SD: 0.9242415, 0.4694007, 0

```
c(1.044 - 1.96 * .259, 1.044, 1.044 + 1.96 * .259, .259)
```

iii.) MLE ($\propto 1$) – Table

```
## [1] 0.53636 1.04400 1.55164 0.25900
```

In general, the results are fairly close (largely because with 1197 data points, the prior gets swamped). However, we do see that the uninformative prior is pretty close to the MLE fit – this makes sense, because as $\sigma \to \infty$, a normal tends towards an improper flat prior, which is the MLE fit. So there is a natural connection there – σ's getting big and starting to look flat, hence the MLE similarity (roughly .26 SD). In contrast, the standard Gaussian prior imposes a stricter prior belief (i.e. stronger L2 regularization), so coefficients should be shrunk a tad bit closer to zero, and more conspicuously, the posterior interval should be narrower. Indeed, the standard prior has a narrower posterior credible interview (.23 SD), in support of the previous point.

9.7

Make data:

```
data_9.7 <- data.frame(
  number=c(1105, 411111, 4624, 157342,
      14, 483, 497, 1008),
  seatbelt=rep(c(1, 1, 0, 0), 2),
  eject=rep(c(1, 0), 4),
  fatal=c(rep(0, 4), rep(1, 4))
)

data_9.7
```

```
##   number seatbelt eject fatal
## 1   1105        1     1     0
## 2 411111        1     0     0
## 3   4624        0     1     0
## 4 157342        0     0     0
## 5     14        1     1     1
## 6    483        1     0     1
## 7    497        0     1     1
## 8   1008        0     0     1
```

a.)

Using $S \in \{1, 0\}$ for seatbelt/not use, $E \in \{1, 0\}$ for ejection/not, $F \in \{1, 0\}$ for fatality/not, a natural choice of model is a first order interaction model, i.e. `~ S*E + S*F + E*F`, corresponding to (see Table 9.13) $\pi_{sef} = \psi_{se}\phi_{ef}\omega_{sf}$ setup. Using `glm()`, and specifically the `family=poisson()` plug-in/argument to do the necessary Newton-Raphson for us, we then have

```
glm(
  number ~ 1 + seatbelt*eject + seatbelt*fatal + eject*fatal,
  data=data_9.7,
  family="poisson"
) -> fit_ll

fit_ll
```

```
##
## Call:  glm(formula = number ~ 1 + seatbelt * eject + seatbelt * fatal +
##     eject * fatal, family = "poisson", data = data_9.7)
##
## Coefficients:
##    (Intercept)        seatbelt           eject           fatal  seatbelt:eject
##        11.9661          0.9605         -3.5256         -5.0436         -2.3996
## seatbelt:fatal     eject:fatal
##        -1.7173          2.7978
##
## Degrees of Freedom: 7 Total (i.e. Null);  1 Residual
## Null Deviance:       1625000
## Residual Deviance: 2.854     AIC: 93.85
```

b.)

Here, we replicate the rationale of 9.5.1. Much like Agresti's example, our log-linear model is based off of `(SE, SF, EF)`. Further, as demonstrated in 9.5.1, the logit for injury under this log-linear model is then

$$\begin{aligned}
\frac{P(F=1|S,E)}{P(F=0|S,E)} &= \log\left(\frac{\mu_{f=1,s,e}}{\mu_{f=0,s,e}}\right) \\
&= \log(\mu_{f=1,s,e}) - \log(\mu_{f=0,s,e}) \\
&= \Big(\lambda + \lambda^F \cdot 1 + \lambda^S S + \lambda^E E + \lambda^{SE} SE + \lambda^{EF} E \cdot 1 + \lambda^{SF} S \cdot 1\Big) - \\
&\qquad \Big(\lambda + \lambda^F \cdot 0 + \lambda^S S + \lambda^E E + \lambda^{SE} SE + \lambda^{EF} E \cdot 0 + \lambda^{SF} S \cdot 0\Big) \\
&= \Big(\lambda + \lambda^F + \lambda^S S + \lambda^E E + \lambda^{SE} SE + \lambda^{EF} E + \lambda^{SF} S\Big) - \\
&\qquad \Big(\lambda + \lambda^S S + \lambda^E E + \lambda^{SE} SE\Big) \\
&= \lambda^F + \lambda^{EF} E + \lambda^{SF} S.
\end{aligned}$$

This is a logistic regression (logit is modeled linearly), where λ^F is the intercept, λ^{EF} is the effect for E, and λ^{SE} is the effect for S. Of course, in this logistic regression, we'll need to weigh by number observed:

```
glm(
  fatal ~ 1 + eject + seatbelt,
  data=data_9.7,
  family="binomial",
  weights=number
) -> fit_logit

fit_logit
```

```
##
## Call:  glm(formula = fatal ~ 1 + eject + seatbelt, family = "binomial",
##     data = data_9.7, weights = number)
##
## Coefficients:
## (Intercept)        eject     seatbelt
##      -5.044        2.798       -1.717
##
## Degrees of Freedom: 7 Total (i.e. Null);  5 Residual
## Null Deviance:       26670
## Residual Deviance: 23100     AIC: 23110
```

Here, our interpretation goes as follows: all else held constant (i.e. given seatbelt status), being ejected (as opposed to not) increases one's odds of fatality by a factor of 16.4117903; similarly, all else held constant (i.e. given ejection status), not wearing a belt (as opposed to not) decreases one's odds of fatality by a factor of 5.5678. Crucially, we should recognize these coefficients (as interactions) from our log-linear fit above.

c.)

Notably, our deviance isn't two big, at -2.85 (see model output above). However, for dissimilarity, we have

```
sum(abs(data_9.7$number - predict(fit_ll, type="response"))) /
  (2 * sum(data_9.7$number))
```

```
## [1] 4.767967e-05
```

As this is pretty close to zero, the model looks like it's in decent shape.

9.8

a.)

Just as in the previous problem, we can reframe this problem as a logistic regression, i.e. `I ~ G + L + S`, where here the terms are reported (based on problem setup) as `1 = no injury` and `0 = injury`. Hence, the `IG=0.58` term implies that the odds of no injury for females, given all else (`L, S`) is .58 times that for males; thus, females are, given other features, less likely to be non-injured, i.e. more likely to be injured.

Similarly, we have that the odds of no-injury in an urban setting is `IL=2.13` times that of no-injury in a rural setting (perhaps due to higher traveling speeds in rural settings?), indicating that injury is more common in rural settings. So again, given other features (`G, S`), the rural crash is more likely to cause injury by this coefficient.

Lastly, given gender and location (`G, L`), we see that the odds of no-injury for seatbelt users are `IS=.44` times those of non-seatbelt users; in other words, given everything else, it's less injurious to wear a seatbelt, and hence more injurious to not wear a seatbelt.

As the model construction here features no interaction terms between `I` and two or more of the `S, L, G` features, the logistic regression has no interaction terms – recall it's LME form is `I ~ G + L + S`. Hence, the `G, L, S` components work additively, so female + rural + no-seatbelt sums up to the most at-risk group, at least according to this model.

b.)

```
# setup non-injury as Y=1 target
data_9.8 <- c(
  # G   L    S   N     I
  "F", "U", "N", 7287, 1,
  "F", "U", "N", 996, 0,

  "F", "U", "Y", 11587, 1,
  "F", "U", "Y", 759, 0,

  "F", "R", "N", 3246, 1,
  "F", "R", "N", 973, 0,

  "F", "R", "Y", 6134, 1,
  "F", "R", "Y", 757, 0,


  "M", "U", "N", 10381, 1,
  "M", "U", "N", 812, 0,

  "M", "U", "Y", 10969, 1,
  "M", "U", "Y", 380, 0,

  "M", "R", "N", 6123, 1,
  "M", "R", "N", 1084, 0,

  "M", "R", "Y", 6693, 1,
  "M", "R", "Y", 513, 0
) %>%
  matrix(., ncol=5, byrow=T) %>%
  data.frame() %>%
  `colnames<-`(c("Gen", "Loc", "StBlt", "N", "Inj")) %>%
  mutate_at(c("N", "Inj"), as.numeric)
# relevel to match problem interpretation
data_9.8$StBlt = relevel(as.factor(data_9.8$StBlt), "Y")
data_9.8$Gen = relevel(as.factor(data_9.8$Gen), "F")

glm(
  N ~ Gen * Loc * StBlt + Gen * Inj + Inj * Loc + Inj*StBlt,
  data=data_9.8,
  family="poisson"
) -> fit

fit%>%
  coef() %>%
  exp()
```

```
##      (Intercept)             GenM             LocU           StBltN
##      797.4978976        0.6374392        0.8945202        1.2092047
##              Inj       GenM:LocU      GenM:StBltN      LocU:StBltN
##        7.6407751        0.8571315        1.7192054        1.1706036
##         GenM:Inj         LocU:Inj       StBltN:Inj GenM:LocU:StBltN
```

```
##        1.7243138        2.1341283        0.4417119        0.8793430
```

In looking at the exponentiated coefficients above – that is, the conditional fitted odds ratios – we indeed see `SI=.4417`, as given by `StBltN:Inj`. Further, when it is a female, each of the `GenM` indicators zero out/are switched off, so the conditional odds ratio derived from `GLS` is just `LS=1.1706` (as the `G=Female` specification switches off all other indicators and reduces conditional odds ratio to a single $\hat{\lambda}$, i.e. $exp(-0.8170974 + 0 - 0 - 0))$. However, when `G=Male`, those indicators switch back on, and we look at the un-exponentiated coefficients to derive conditional odds ratios from the familiar $\exp(\hat{\lambda}_{11} + \ldots - \hat{\lambda}_{21})$ formula. Note that in the fit, we have:

```
coef(fit)
```

```
##      (Intercept)             GenM             LocU           StBltN
##        6.6814792       -0.4502964       -0.1114678        0.1899628
##              Inj        GenM:LocU      GenM:StBltN      LocU:StBltN
##        2.0334991       -0.1541640        0.5418622        0.1575195
##         GenM:Inj         LocU:Inj       StBltN:Inj GenM:LocU:StBltN
##        0.5448292        0.7580583       -0.8170974       -0.1285802
```

Hence, by including the `GLS` term switched on for `G=Male` into the log odds ratio, we recover the desired ratio via `exp(LocU:StBltN - GenM:LocU:StBltN) = exp(0.1575195 - 0.1285802)=1.029`.

c.)

```
# a bit of data munging
# Seatbelt already factored to first level == seatbelt
data_9.8_ = data_9.8 %>%
  mutate(S=ifelse(StBlt=="Y", 1, 0))

fit_a = glm(S ~ Gen + Loc, data=data_9.8_, weights=N, family="binomial")
data_9.8_$S_hat = predict(fit_a, type="response")
fit_b = glm(Inj ~ S + Gen + Loc, data=data_9.8_, weights=N, family="binomial")
```

The composite model is sensible here, as it allows us to play "choose your own adventure" to understand and make inferences about the decision/action sequence of a car crash. Specifically, we can use gender and location to predict whether or not someone was likely wearing a seatbelt. This is the first step in the sequence. Then, conditioned on this first step information, we can leap to the second step, to predict injury. In short, the composite approach makes understanding the chain of events much more interpretable: we can first ask, "was this person wearing a seatbelt when they crashed?", followed by "if so, how did that change their chances of injury?" In this way, we can chain our inference/predictions.

Coefficients for the first model (the intermediate/first leap) are as follows:

```
summary(fit_a)
```

```
##
## Call:
## glm(formula = S ~ Gen + Loc, family = "binomial", data = data_9.8_,
##     weights = N)
##
## Deviance Residuals:
```

```
##      Min        1Q    Median        3Q       Max
## -119.781   -51.767    -5.257    40.201   123.500
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.45195    0.01550   29.15   <2e-16 ***
## GenM        -0.42387    0.01551  -27.32   <2e-16 ***
## LocU        -0.03227    0.01598   -2.02   0.0434 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 94538  on 15  degrees of freedom
## Residual deviance: 93785  on 13  degrees of freedom
## AIC: 93791
##
## Number of Fisher Scoring iterations: 4
```

- `genM`: all else constant, male occupants had odds of wearing a seatbelt that were 0.654509 times those of female occupants. That is, they were less likely.
- `LocU`: all else constant, urban crashes had odds of wearing a seatbelt that were 0.9682451 times those of rural crashes.

Coefficients for the second model (the second leap, as conditioned on the first leap) are as follows:

```
summary(fit_b)
```

```
##
## Call:
## glm(formula = Inj ~ S + Gen + Loc, family = "binomial", data = data_9.8_,
##     weights = N)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -65.79  -58.98  -11.46   39.16   44.07
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.21640    0.02649   45.92   <2e-16 ***
## S            0.81710    0.02765   29.55   <2e-16 ***
## GenM         0.54483    0.02727   19.98   <2e-16 ***
## LocU         0.75806    0.02697   28.11   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 41987  on 15  degrees of freedom
## Residual deviance: 40082  on 12  degrees of freedom
## AIC: 40090
##
## Number of Fisher Scoring iterations: 6
```

- `genM`: all else constant, male occupants had odds of injury that were 1.7243152 times those of female occupants.
- `LocU`: all else constant, urban crashes had odds of injury that were 2.134132 times those of rural crashes.
- `S`: all else constant, crashes in which the occupant was unbelted (S=1) had odds of injury that were 2.2639249 times as those wearing seat belts.

9.20

We have $P(X|Y,Z) \overset{(1)}{=} P(X|Z) \overset{(2)}{=} P(X)$, by the conditional (first equality) and marginal independence (second equality) suppositions. Accordingly, for any $X = x, Y = y, Z = z$, we are guaranteed:

$$\begin{aligned} p(x|y) &= \int_{dZ} p(x,z|y)d\mu(z) \\ &= \int_{dZ} p(x|y,z)p(z)d\mu(z) \\ &= \int_{dZ} p(x|z)p(z)d\mu(z) \\ &= \int_{dZ} p(x)p(z)d\mu(z) \\ &= p(x)\int_{dZ} p(z)d\mu(z) \\ &= p(x)\cdot 1 \\ &= p(x), \end{aligned}$$

as desired. The steps are justified by (i) introducing a nuisance parameter z to marginalize over; (ii) conditional probability property/algebra; (iii) conditional independence; (iv) marginal independence; (v) and the remainder by integration of densities over their support. Hence, we see that $P(X|Y) = P(X)$, satisfying the desired marginal independence.

9.25

a.)

As set forth in Table 9.1, and more generally in Section 9.2.1, conditional independence holds between X and Y if there exists no λ^{XY} interaction term (intuitively this makes sense; we effectively showed this in the homogenous association model in HW1). Hence, for our model to maintain this desired conditional independence, it cannot have such an interaction. And `(WXZ, WYZ)`, which amounts to a LME formula of the form `~ 1 + W + X + Z + Y + WX + WZ + XZ + WY + YZ + WXZ + WYZ` contains all possible terms but for those that have an `X, Y` interaction – hence it is the most general model here.

b.)

As the above is most general, we need only to remove the three factor interactions, to wit `WXZ` and `WYZ`. This gives `~ 1 + W + X + Z + Y + WX + WZ + XZ + WY + YZ`, or `(WX, WZ, XZ, WY, YZ)` in textbook notation.

10.19

The joint density for the `(WX, XY, YZ)` loglinear model has joint density:

$$p(w,x,y,z) \propto \exp\left(\lambda + \lambda_w w + \lambda_x x + \lambda_y y + \lambda_z z + \lambda_{wx} wx + \lambda_{xy} xy + \lambda_{yz} yz\right)$$

First, for $W|X, Y$, we have (just dropping constants here)

$$p(w|x, y) \propto \exp\left(\lambda_w w + \lambda_{wx} wx\right)$$

and then identically for $Z|X, Y$.

$$p(z|x, y) \propto \exp\left(\lambda_z z + \lambda_{yz} yz\right)$$

Then, we have (again dropping constants)

$$p(w, z|x, y) \propto \exp\left(\lambda_w w + \lambda_z z + \lambda_{yz} yz + \lambda_{wx} wx\right) = \exp\left(\lambda_w w + \lambda_{wx} wx\right) \cdot \exp\left(\lambda_z z + \lambda_{yz} yz\right) = p(w|x, y)p(z|x, y)$$

For the $|X$ case (and identically the $|Y$ case too), a similar proof holds: First, for $W|X, Y$, we have (just dropping constants here)

$$p(w|x) \propto \exp\left(\lambda_w w + \lambda_{wx} wx\right)$$

and then

$$p(z|x, y) \propto \exp\left(\lambda_z z + \lambda'_{yz} yz\right)$$

where the $\lambda'_{yz} z$ falls out during the process of integrating out y, i.e. $\exp(\lambda'_{yz} yz) \propto \int_Y \exp(\lambda_{yz} yz) dy$.

Then, we have (again dropping constants)

$$\begin{aligned} p(w, z|x) &\propto \int_Y \exp\left(\lambda_w w + \lambda_z z + \lambda_{yz} yz + \lambda_{wx} wx\right) dy \\ &\propto \exp\left(\lambda_w w + \lambda_z z + \lambda_{wx} wx\right) \int_Y \exp(\lambda_{yz}) dy \\ &\propto \exp\left(\lambda_w w + \lambda_z z + \lambda_{wx} wx + \lambda'_{yz} yz\right) \\ &= \exp\left(\lambda_w w + \lambda_{wx} wx\right) \exp\left(\lambda_z z + \lambda'_{yz} yz\right). \end{aligned}$$

As desired. the $|Y$ case holds identically. More generally, we see that because the graph is

$$W <--> X <--> Y <--> Z$$

In this graph, X and Y are in the middle, effectively splitting W and Z. Thus, if we condition on either (or both), we effectively "break" the graph in half and separate W and Z, giving us the conditional independence.

Digit Logistic Regression

```
suppressMessages(library(dplyr))
suppressMessages(library(stringr))
suppressMessages(library(glmnet))
sigmoid <- function(z){exp(z) / (1 + exp(z))}

########################################
# Part A
########################################
read_digits <- function(zip_obj){
  lapply(1:nrow(zip_obj),
         function(i){
           zip_rows = zip_obj[i, ] %>%
             str_split(., " ") %>%
             unlist()
           # sometimes it catches an extra ""
           if (zip_rows[length(zip_rows)] == ""){
             zip_rows = zip_rows[1:(length(zip_rows) - 1)]
```

```
          }
          as.numeric(zip_rows)
        }
  ) %>%
    do.call("rbind", .) %>%
    data.frame() %>%
    `colnames<-`(c("digit", paste0("X", 1:256))) %>%
    filter(digit == 6. | digit == 8.) %>%
    mutate(y=ifelse(digit == 6., 1, 0))
}

zip_train = read_digits(read.delim2("/Users/IKleisle/Downloads/zip_train.txt"))
zip_test = read_digits(read.delim2("/Users/IKleisle/Downloads/zip_test.txt"))


########################################
# Part B
########################################
# letting glmnet decide the lambda sequence, as instructed
fit_lasso = cv.glmnet(
  x=zip_train %>% dplyr::select(X1:X256) %>% as.matrix() %>% unname(),
  y=zip_train %>% pull(y),
  family="binomial",
  alpha=1
)

fit_ridge = cv.glmnet(
  x=zip_train %>% dplyr::select(X1:X256) %>% as.matrix() %>% unname(),
  y=zip_train %>% pull(y),
  family="binomial",
  alpha=0
)

fit_enet = cv.glmnet(
  x=zip_train %>% dplyr::select(X1:X256) %>% as.matrix() %>% unname(),
  y=zip_train %>% pull(y),
  family="binomial",
  alpha=0.5
)

### extract lambda corresponding to 1SE
# lam_idx_lasso = which(fit_lasso$lambda == fit_lasso$lambda.1se)
# lam_idx_ridge = which(fit_ridge$lambda == fit_ridge$lambda.1se)
# lam_idx_enet = which(fit_enet$lambda == fit_enet$lambda.1se)

########################################
# Part C
########################################
yhat_lasso = predict(
  fit_lasso,
  s=fit_lasso$lambda.1se,
  newx=zip_test %>% dplyr::select(X1:X256) %>% as.matrix() %>% unname()
) %>%
```

```
  as.numeric() %>%
  sigmoid() %>%
  round()

yhat_ridge = predict(
  fit_ridge,
  s=fit_ridge$lambda.1se,
  newx=zip_test %>% dplyr::select(X1:X256) %>% as.matrix() %>% unname()
) %>%
  as.numeric() %>%
  sigmoid() %>%
  round()

yhat_enet = predict(
  fit_enet,
  s=fit_enet$lambda.1se,
  newx=zip_test %>% dplyr::select(X1:X256) %>% as.matrix() %>% unname()
) %>%
  as.numeric() %>%
  sigmoid() %>%
  round()


cat("Lasso Accuracy (Test):", mean(yhat_lasso == zip_test$y), "\n")
```

```
## Lasso Accuracy (Test): 0.9732143
```

```
cat("Ridge Accuracy (Test):", mean(yhat_ridge== zip_test$y), "\n")
```

```
## Ridge Accuracy (Test): 0.985119
```

```
cat("E-Net Accuracy (Test):", mean(yhat_enet == zip_test$y), "\n")
```

```
## E-Net Accuracy (Test): 0.9821429
```

Here, we see that the Ridge model performs best on the test set. This may indicate that we want to shrink coefficients for each of the 256 pixels, instead of eliminating some pixels entirely (as ridge would, and potentially E-Net.). Importantly, note that `standardize=T` inside of GLMNET (i.e. it scales down and then back up at prediction time), so mean/centering is taken care of under the hood.

Deeper Lasso Dive

1.)

As discussed in class, the E block contains all of the features that have *not* been eliminated/zeroed out by the lasso – that is, all the features that are still alive (hence, active) in the model. All features in this block should be zero with probability zero. In contrast, the $-E$ contains all the coefficients that *have* been eliminated/zero'd out by the lasso. Here, all features in the inactive block are zero (wp 1.)

2.)

```
suppressMessages(library(glmnet))
# FEATURES = c("lcavol", "lweight", "age", "lbph","svi","lcp","gleason","pgg45")
# ### ESL prostate data
prostate = read.csv("~/Stanford/STATS305A/lprostate.csv")
# X = scale(prostate[FEATURES] %>% as.matrix() %>% unname())
# Y = prostate %>% pull(lpsa); Y = Y - mean(Y)
# fit = glmnet(X, Y, alpha=1, intercept=F)

### given by HW ###
X = model.matrix(lm(lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason + pgg45, data=prostate))
X = scale(X, TRUE, TRUE)[, 2:ncol(X)]
Y = as.numeric(prostate$lpsa - mean(prostate$lpsa))
G = glmnet(X, Y, intercept=FALSE, standardize=FALSE)
beta.hat =  coef(G, s=0.17, exact=TRUE, x=X, y=Y)

### print bhat at this sparsity setting
beta.hat
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) .
## lcavol      0.5452621
## lweight     0.1498760
## age         .
## lbph        .
## svi         0.1647422
## lcp         .
## gleason     .
## pgg45       .
```

As we see, the active block contains `svi`, `lcavol`, and `lweight`.

Now recall the KKT conditions, and in particular our construction for the desired u.

$$\partial \underbrace{\lambda||\hat{\beta}||_1}_{penalty} = \begin{cases} \lambda sign(\hat{\beta}_j) & \hat{\beta} \neq 0 \\ [-\lambda, \lambda] & else \end{cases},$$

i.e.

$$u \in \partial P(\beta) \iff \beta \in N_u(K),$$

where $N_u(K)$ is the set of normal directions of $\mu \in K$. We can also more succinctly say that $\hat{u} = \{u : u^T\hat{\beta} = P(\hat{\beta})\}$.

Further, we similarly have as part of KKT

$$s_j \in \begin{cases} sign(\hat{\beta}_j) & \hat{\beta} \neq 0 \\ [-1, 1] & else \end{cases}$$

and this is subject to

$$X^T(y - X\hat{\beta}) = \lambda\vec{s}$$

However here, we make one modification – since GLMNET devides the squared error term by N, i.e.

$$L(\beta; Y, X) + P(\beta) = \frac{1}{n}||y - X\beta||_2^2 + \lambda||\beta||_1,$$

the modified KKT is in fact

$$X^T(Y - X\hat{\beta}) = n\lambda\vec{s}$$

So let's compute the RHS and see what falls out. Note that numeric errors preclude recovery of the precise result:

```
N = dim(X)[1]
LAMBDA = 0.17
beta_hat = as.matrix(beta.hat)
# intercept is zero'd, so drop it
beta_hat = beta_hat[2:length(beta_hat)]
# LHS
(t(X) %*% Y - t(X) %*% X %*% beta_hat)
```

```
##               [,1]
## lcavol  16.4881825
## lweight 16.4896252
## age      0.1492448
## lbph    13.4867701
## svi     16.4900000
## lcp     12.4563166
## gleason 12.3711366
## pgg45   15.3188336
```

Ostensibly, this equals $n\lambda\vec{s}$, so if we divide the above by $n\lambda$ we should recover $\hat{s}$. Indeed:

```
s_fit = (t(X) %*% Y - t(X) %*% X %*% beta_hat) / (LAMBDA * N)
s_true = beta_hat %>%
  sapply(., function(x){ifelse(x != 0, sign(x), "[-1, 1]")})

data.frame(s_fit, s_true)
```

```
##               s_fit  s_true
## lcavol  0.999889784       1
## lweight 0.999977270       1
## age     0.009050622 [-1, 1]
## lbph    0.817875687 [-1, 1]
## svi     1.000000000       1
## lcp     0.755386088 [-1, 1]
## gleason 0.750220533 [-1, 1]
## pgg45   0.928977174 [-1, 1]
```

which is quite close to (up to numeric round offs) $\vec{s}$. So indeed, we satisfy KKT. Note that along the way, we found

$$\hat{u} = \lambda\hat{s},$$

assuming the normalizing N is assumed to have been moved to the LHS.

3.)

Critically, note that

- (i) our dataset, and thus N, have not changed
- (ii) the indices of the non-zero coefficients have not changed; only the coefficent values at those nonzero positions have
- (iii) lambda has not changed.

Hence, whether under $\hat{\beta}$ or $\hat{\beta}'$, the new coefficients, $n\lambda\hat{s} = \hat{u}$ will be the same across the two of them; there will still be 1's corresponding to the non-zero coefficients, and $[-1, 1]$'s corresponding to the zeroed coefficients. We already know that the first fit satisfies KKT, i.e.

$$X^T(Y - X\hat{\beta}) = n\lambda\vec{s};$$

the same must be true of the second fit, i.e.

$$X^T(Y' - X\hat{\beta}') = n\lambda\vec{s}$$

In this second problem, Y' is the unknown variable for which we are trying to solve for. Looking at the above, we immediately have

$$n\lambda\vec{s} = X^T(Y - X\hat{\beta}) = X^T(Y - X\hat{\beta}) \implies (Y' - X\hat{\beta}') = (Y - X\hat{\beta}) \implies Y' = Y - X(\hat{\beta} - \hat{\beta}')$$

Still, in this second equality, Y' remains the only unknown. Note I played it a bit fast and loose here: we may not always be able to cancel off the X^T term, so this represents but one of the infinitely many solutions. In other words, it's guaranteed to work for at least this, and most likely more.

```
### make new B'
beta_hat_ = c(.6, .15, 0, 0, .2, 0, 0, 0)
### solve for the Y
Y_ = Y - X %*% (beta_hat - beta_hat_)

G_ = glmnet(X, Y_, intercept=FALSE, standardize=FALSE)
beta.hat =  coef(G_, s=0.17, exact=TRUE, x=X, y=Y_)
beta.hat
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) .
## lcavol      0.5999826
## lweight     0.1500009
## age         .
## lbph        .
## svi         0.2000092
## lcp         .
## gleason     .
## pgg45       .
```

Indeed, we retrieve the original coefficients under this reconstructed Y'.

Extra

1.)

Recall from our logistic regression module that (as before, σ = sigmoid) the score function gives:

$$-\nabla \log L(\beta|Y,X) = X^T(Y - E_\beta[Y|X]) = -\left[X^T\left(Y - \frac{\exp(X\beta)}{1+\exp(X\beta)}\right)\right]$$

Hence, over the active set, only the β_e such that $e \in E$ are the non-zero coefficients, i.e. $\forall e \in E, \beta_e \neq 0$. Hence, the score looks like

$$-\nabla \log L(\hat{\beta}|Y,X) = -\nabla \log L(\hat{\beta}_E|Y,X_E) = -\left[X_E^T\left(Y - \frac{\exp(X_E\hat{\beta}_E)}{1+\exp(X_E\hat{\beta}_E)}\right)\right] = -\left[X_E^T\left(Y - \frac{\exp(\sum_{e\in E} x_e\hat{\beta}_e)}{1+\exp(\sum_{e\in E} x_e\hat{\beta}_e)}\right)\right]$$

Just as before, it's X_E transposed and then dotted with Y minus the predicted mean/expectation, with respect to the E set/variables. Just by pattern matching, we can see the similarity.

2.)

Here, we're effectively (using "secret" knowledge) whittling our feature set down to E, and proceeding normally there to fit our model and obtain $\hat{\beta}_E$. Then, recall that Fisher scoring and Newton-Raphson are identical in the logistic regression setting (GLM Slides p. 13), and that a Newton-Raphson step really just reflects a rearranged Taylor expansion, and thus one Fisher scoring step (i.e. Taylor expansion) amounts to

$$\bar{\beta}_E = \hat{\beta}_E - (X_E^T W_{\hat{\beta}_E}(X_E)X_E)^{-1}X_E^T\ X_E^T\left(Y - \frac{\exp(X_E\hat{\beta}_E)}{1+\exp(X_E\hat{\beta}_E)}\right),$$

but our stationarity assumption (following from the convergence of $\hat{\beta}$ to the true MLE) satisfies,

$$X_E^T\left(Y - \frac{\exp(X_E\hat{\beta}_E)}{1+\exp(X_E\hat{\beta}_E)}\right) \approx 0 \implies E_{\hat{\beta}_E}\left[X_E^T\left(Y - \frac{\exp(X_E\hat{\beta}_E)}{1+\exp(X_E\hat{\beta}_E)}\right)\right] \approx 0$$

and rearrangement of the Taylor plus stationarity gives

$$X_E^T\left(Y - \frac{\exp(X_E\hat{\beta}_E)}{1+\exp(X_E\hat{\beta}_E)}\right) \approx X_E^T W_{\hat{\beta}_E}(X_E)X_E(\bar{\beta}_E - \hat{\beta}_E)$$

so thus

$$\bar{\beta}_E - \hat{\beta}_E = (X_E^T W_{\hat{\beta}_E}(X_E)X_E)^{-1}X_E^T\left(Y - \frac{\exp(X_E\hat{\beta}_E)}{1+\exp(X_E\hat{\beta}_E)}\right).$$

As set forth in Log. Regression Slides 33-34, this gives (with the help of Slutzky):

$$E[\bar{\beta}_E - \hat{\beta}_E] = 0,$$

$$VAR[\bar{\beta}_E - \hat{\beta}_E] = \left(X_E^T W_{\hat{\beta}_E}(X_E)X_E\right)^{-1},$$

and

$$\bar{\beta}_E - \hat{\beta}_E \sim N\left(0, \left(X_E^T W_{\hat{\beta}_E}(X_E)X_E\right)^{-1}\right) \implies \bar{\beta}_E \sim N\left(\hat{\beta}_E, \left(X_E^T W_{\hat{\beta}_E}(X_E)X_E\right)^{-1}\right).$$

All of this is recapitulated from the Logistic Regression and GLM slides, as presented in class.

3.)

Knowledge of the above normal then induces a familiar likelihood ratio test, where we test w.r.t. statistic

$$-2(\ell(\ell(\beta_0) - \hat{\beta}_{MLE}))$$

against a chi-squared, where $\beta_0 = 0$ reflects the given null. We'll use the variance from above as known.

4.)

First, we run the model as provided by the starter code:

```
SAheart = read.csv("~/Downloads/SAHeartDisease.csv")
### code provided by problem ###
X = model.matrix(glm(chd ~ ., data=SAheart, family=binomial))[,-1]
X = scale(X, TRUE, TRUE)
Y = SAheart$chd
cvG = cv.glmnet(x=X,
                y=Y,
                intercept=FALSE,
        standardize=FALSE,
        family="binomial")
G = glmnet(x=X, y=Y, family="binomial")
beta.hat =  coef(G,
                 s=0.043,
         exact=TRUE,
                 x=X, y=Y,
         intercept=FALSE,
         standardize=FALSE)
E = which(beta.hat != 0)

### IKM edits
### make E-specific variables/features
betaE = beta.hat[E]
XE = X[, c("tobacco", "ldl", "famhistPresent", "typea", "age")]
### compute terms relevant to fisher scoring
# prediction mean
yhat = 1 / (1 + exp(- as.numeric(XE %*% betaE)))
W  = diag(yhat)
```

Next, we perform the fisher update on just the E-set (as proffered on Logistic slides p.30):

```
beta_bar_E = betaE - solve(t(XE) %*% W %*% XE) %*% t(XE) %*% (Y - yhat)
beta_bar_E
```

```
##                       [,1]
## tobacco         0.17802298
## ldl             0.11564724
## famhistPresent  0.17599001
## typea          -0.03334598
## age             0.31864619
```

For context, $\hat{\beta}_E$ was

```
beta.hat
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)    .
## sbp            .
## tobacco        0.21769203
## ldl            0.16177055
## adiposity      .
## famhistPresent 0.23216966
## typea          0.05612436
## obesity        .
## alcohol        .
## age            0.38244052
```

Now, we compute the test statistic, i.e. the multivariate/shared-covariance normal $(\mu_1 - \mu_0)^T \Sigma (\mu_1 - \mu_0)$:

```
Sigma = solve(t(XE) %*% W %*% XE)
test_stat = -2 * (
  dmvnorm(as.numeric(beta_bar_E), rep(0, 5), unname(Sigma), log=T) -
    dmvnorm(as.numeric(beta_bar_E), as.numeric(beta_bar_E), unname(Sigma), log=T)
)
### alternatively equals
test_stat_backup = t(as.numeric(beta_bar_E) - rep(0, 5)) %*% solve(Sigma) %*% (as.numeric(beta_bar_E) -

### p-value: 5 moving variables:
1 - pchisq(test_stat, df=5)
```

```
## [1] 5.461187e-13
```

Which results in a clear rejection: even after one Fisher scoring, there is good evidence that $\bar{\beta}_E$ is not the zero vector.