10.11

Bayesian Logistic Model Example

This re-analysis of data in Section 10.1.3 was done by Nathan James, Dinesh Karki, and Elizabeth McNeer. Ryan Jarrett added the section on bivariate confidence regions. This Bayesian analysis uses the R `brms` package [33] front-end to the Stan Bayesian modeling system [208, 36].

```
require(brms)
require(cluster)
require(MASS)
set.seed(42)
```

Fit the frequentist model using `lrm` and the Bayesian model using `brm` in the `brms` package. For the Bayesian model, the intercept prior was a Student's t-distribution with 3 degrees of freedom and the `age` and `sex` parameters were given mean zero priors with standard deviations computed to achieve specified tail prior probabilities. Four MCMC chains with 5000 iterations were used with a warm-up of 2500 iterations each resulting in 10000 draws from the posterior distribution.

```
dd ← datadist(sex.age.response)
options(datadist = 'dd')

# Frequentist model
fit_lrm ← lrm(response ~ sex + age, data=sex.age.response)

# Bayesian model
# Distribute chains across cpu cores:
options(mc.cores=parallel::detectCores())

# Set priors
# Solve for SD such that sex effect has only a 0.025 chance of
# being above 5 (or being below -5)

s1 ← 5 / qnorm(0.975)

# Solve for SD such that 10-year age effect has only 0.025 chance
```

```
# of being above 20

s2 ← (20 / qnorm(0.975)) / 10   # divide by 10 since ratio on 10b scale

stanvars ← stanvar(s1, name='s1') + stanvar(s2, name='s2')

prs ← c(prior(student_t(3,0,10), class='Intercept'),
        prior(normal(0, s1), class='b', coef='sexmale'),
        prior(normal(0, s2), class='b', coef='age'))

# Full model
fit_brms ← brm(response ~ sex + age, data=sex.age.response,
               family=bernoulli("logit"), prior=prs, iter=5000,
               stanvars=stanvars)
```

The model summaries for the frequentist and Bayesian models are shown below, with posterior means computed as Bayesian "point estimates." The parameter estimates are similar for the two approaches. The frequentist 0.95 confidence interval for the age parameter is 0.037 - 0.279 while the Bayesian 0.95 credible interval is 0.051 - 0.270. Similarly, the 0.95 confidence interval for sex is 1.139 - 5.840 and the corresponding Bayesian 0.95 credible interval is 1.377 - 5.336.

```
# Frequentist model output
fit_lrm
```

Logistic Regression Model

```
lrm(formula = response ~ sex + age, data = sex.age.response)
```

		Model Likelihood Ratio Test		Discrimination Indexes		Rank Discrim. Indexes	
Obs	40	LR χ^2	16.54	R^2	0.451	C	0.849
0	20	d.f.	2	g	2.104	D_{xy}	0.698
1	20	Pr($>\chi^2$)	0.0003	g_r	8.199	γ	0.703
$\max \lvert\frac{\partial \log L}{\partial \beta}\rvert$	7×10^{-8}			g_p	0.350	τ_a	0.358
				Brier	0.162		

	$\hat{\beta}$	S.E.	Wald Z	Pr($>\lvert Z\rvert$)
Intercept	-9.8429	3.6758	-2.68	0.0074
sex=male	3.4898	1.1992	2.91	0.0036
age	0.1581	0.0616	2.56	0.0103

```
summary(fit_lrm, age=20:21)
```

	Low	High	Δ	Effect	S.E.	Lower 0.95	Upper 0.95
age	20	21	1	0.15806	0.061638	0.037249	0.27887
Odds Ratio	20	21	1	1.17120		1.038000	1.32160
sex — male:female	1	2		3.48980	1.199200	1.139500	5.84020
Odds Ratio	1	2		32.78000		3.125200	343.83000

```
# Bayesian model output
post_samps ← posterior_samples(fit_brms, c("age","sex"))

fit_brms
```

```
 Family: bernoulli
  Links: mu = logit
Formula: response ~ sex + age
   Data: sex.age.response (Number of observations: 40)
Samples: 4 chains, each with iter = 5000; warmup = 2500; thin = 1;
         total post-warmup samples = 10000

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    -9.32      3.19   -15.99    -3.66 1.00     5447     5435
sexmale       3.18      1.00     1.39     5.27 1.00     5787     6194
age           0.15      0.05     0.05     0.27 1.00     5805     5805

Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
posterior_interval(fit_brms, c("age","sex"))
```

```
               2.5%     97.5%
b_age     0.05288013 0.2663081
b_sexmale 1.39201656 5.2727436
```

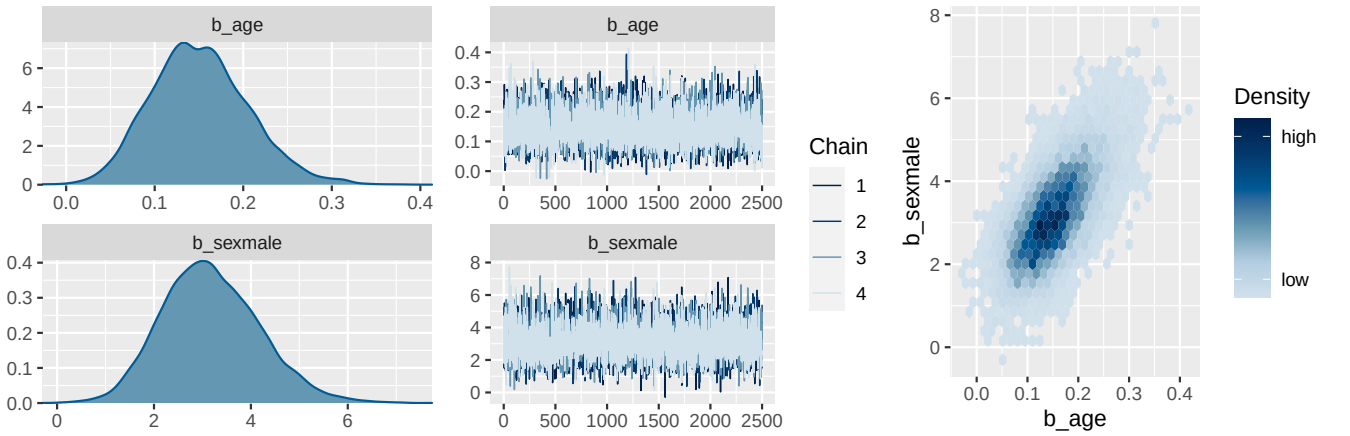
```
prior_summary(fit_brms)
```

```
                prior     class    coef group resp dpar nlpar bound
1                             b
2       normal(0, s2)         b     age
3       normal(0, s1)         b sexmale
4 student_t(3, 0, 10) Intercept
```

The figure shows the posterior draws for the age and sex parameters as well as the trace of the 4 MCMC chains for each parameter and the bivariate posterior distribution. The posterior distributions of each parameter are roughly mound shaped

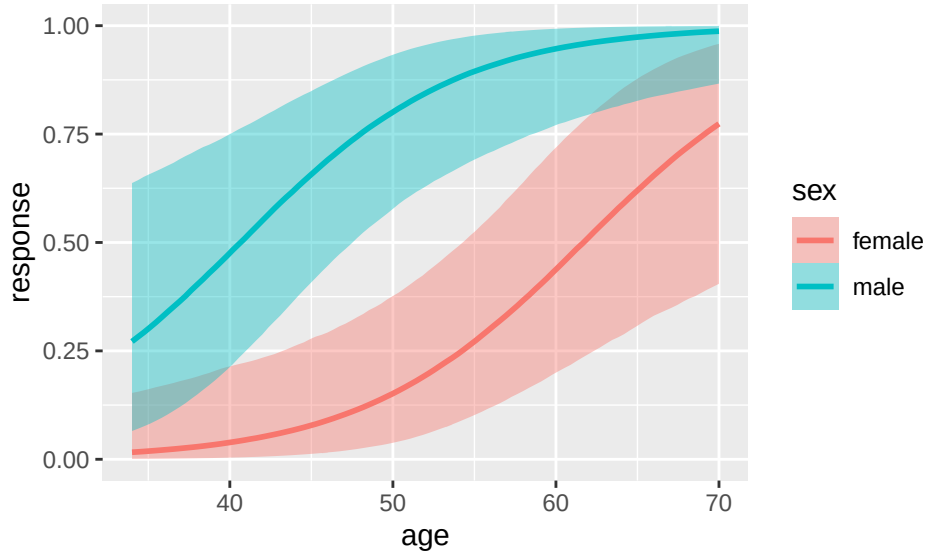
and the overlap between chains in the trace plots indicates good convergence. The bivariate density plot indicates moderate correlation between the age and sex parameters.

```
# display posterior densities for age and sex parameters
plot(fit_brms, c("age","sex"), combo=c("dens","trace","hex"))
```



A plot of the marginal effects from the Bayesian model reveals the same pattern as Figure 10.3.

```
# Marginal effects plot
plot(marginal_effects(fit_brms, "age:sex"))
```




```
# Frequentist
# variance-covariance for sex and age parameters
sex_age_vcov <- vcov(fit_lrm)[2:3,2:3]

# Sampling based parameter estimate correlation coefficient
f_cc <- sex_age_vcov[1,2] / (sqrt(sex_age_vcov[1,1]) * sqrt(sex_age_vcov[2,2]))
```

```
# Bayesian
# Linear correlation between params from posterior
b_cc ← cor(post_samps)[1,2]
```

Using the code in the block above, we calculate the frequentist sampling-based parameter estimate correlation coefficient is 0.75 while the linear correlation between the posterior draws for the age and sex parameters is 0.65. Both models indicate a comparable amount of correlation between the parameters, though in difference senses (sampling data vs. sampling posterior distribution of parameters).

```
# Define P() as mean() just to provide a nice notation for
# computing posterior probabilities
P ← function(x) mean(x)
b1 ← post_samps[, 'b_sexmale']
b2 ← post_samps[, 'b_age']
(p1 ← P(b1 > 0))   # post prob(sex has positive association with Y)
```

```
[1] 0.9999
```

```
(p2 ← P(b2 > 0))
```

```
[1] 0.9991
```

```
(p3 ← P(b1 > 0 & b2 > 0))
```

```
[1] 0.999
```

```
(p4 ← P(b1 > 0 | b2 > 0))
```

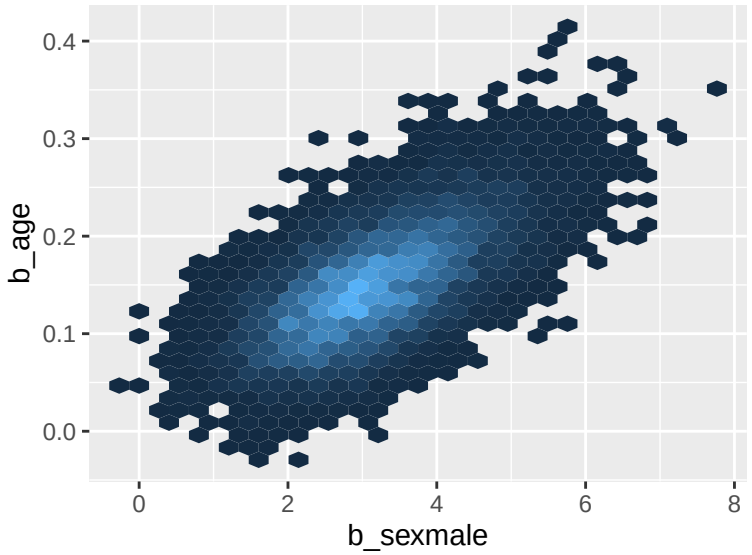
```
[1] 1
```

The posterior probability that sex has a postive relationship with hospital death is estimated as $\text{Prob}(\beta_{sex} > 0) = 0.9999$ while the posterior probability that age has a postive relationship with hospital death is $\text{Prob}(\beta_{age} > 0) = 0.9991$ and the probability of both events is $\text{Prob}(\beta_{sex} > 0 \cap \beta_{age} > 0) = 0.999$. Even using somewhat skeptical priors centered around 0, male gender

and increasing age are highly likely to be associated with the response.

As seen above, the MCMC algorithm used by `brms` provides us with samples from the joint posterior distribution of β_{age} and β_{sex}. Unlike frequentist intervals which require the log-likelihood to be approximately quadratic in form, there are no such restrictions placed on the posterior distribution, as it will always be proportional to the product of the likelihood density and the prior, regardless of the likelihood function that is used. In this specific example, we notice that the bivariate density is somewhat skewed – a characteristic that would likely lead to unequal tail converage probabilities if a symmetric confidence interval is used.

```
ggplot(post_samps, aes(x=b_sexmale, y = b_age)) +
  geom_hex() +
  theme(legend.position="none")
```

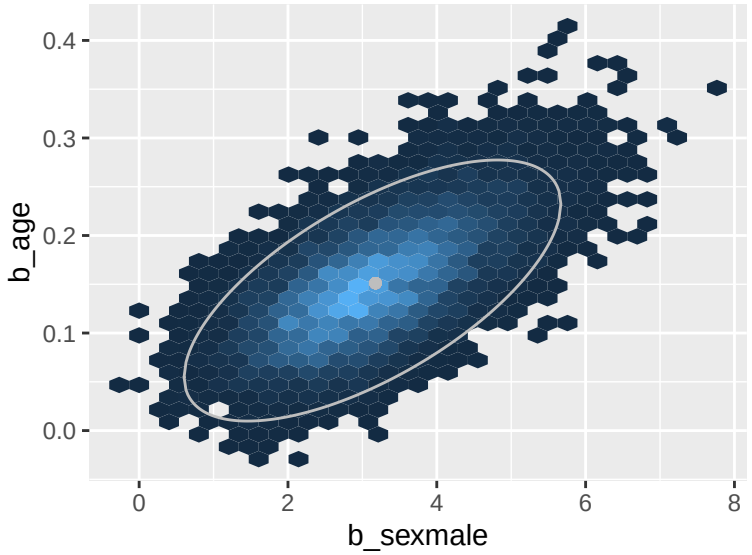


Create a 0.95 bivariate credible interval for the joint distribution of age and sex. Any number of intervals could be drawn, as any region that covers 0.95 of the posterior density could be

accurately be called a 0.95 credible interval. Commonly used: maximum a-posteriori probability (MAP) interval, which seeks to find the region that holds 0.95 of the density, while also having the smallest area. In a 1-dimensional setting, this would translate into having the shortest interval length, and therefore the most precise estimate. The figure below shows the point estimate calculated by `brms` as well as the corresponding MAP interval.

```
# Calculate MAP interval
# Code from http://www.sumsar.net/blog/2014/11/
      how-to-summarize-a-2d-posterior-using-a-highest-density-ellipse/
samples <- as.matrix(post_samps)
coverage = 0.95
fit <- cov.mve(samples, quantile.used = round(nrow(samples) * coverage))
points_in_ellipse <- samples[fit$best,]
ellipse_boundary <- predict(ellipsoidhull(points_in_ellipse))
map <- data.frame(ellipse_boundary)
names(map) <- c("y","x")

ggplot(post_samps, aes(x=b_sexmale, y = b_age)) +
  geom_hex() +
  geom_polygon(data = map, aes(x=x,y=y), color = "grey", alpha = 0) +
  geom_point(aes(x = fixef(fit_brms)[,1][2], y = fixef(fit_brms)[,1][3]), color
       = "grey") +
  theme(legend.position="none")
```



In the above figure, the point estimate does not appear quite

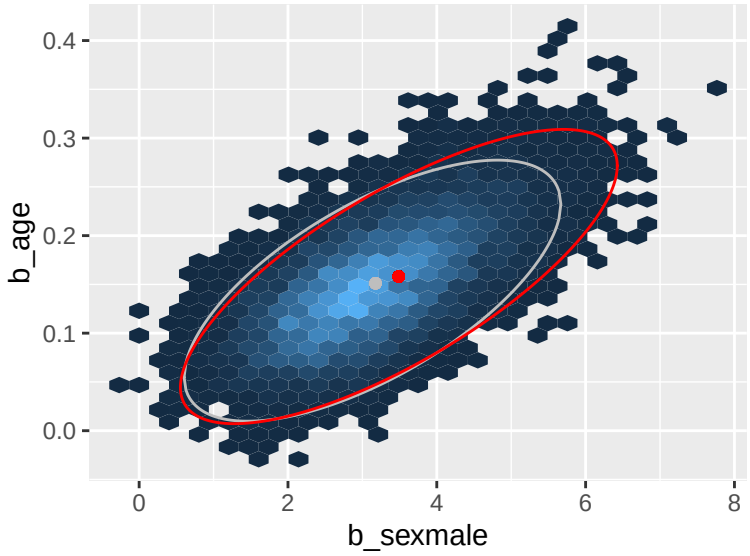
at the point of highest density. This is because `brms` estimates the posterior mean, rather than the posterior mode. You have the full posterior density, so you can calculate whatever you'd like if you don't want the mean.

See how the MAP interval compares to a confidence ellipse that we would calculate using a frequentist approach.

```
# Function takes in variance-covariance matrix (D), point estimates (d),
# and a level of significance (alpha)
EllipseDF ← function(D, d, alpha = 0.05) {
  delta = sqrt(eigen(D)$values)
  # Root eigenvalues correspond to the half-lengths of the ellipse
  V = eigen(D)$vectors
  # Eigenvectors give the axes of the confidence ellipse
  R = sqrt(qchisq(1-alpha, df = length(delta)))
  # Scaling factor to get to 0.95 confidence
  a = R*delta[1] # scale the ellipse axes
  b = R*delta[2]
  t ← seq(0, 2*pi, length.out=200)
  # Generate radian measures from 0 to 2pi
  points.proj = V %*% t(cbind(a * cos(t), b * sin(t)))
  # Transform circle into ellipse
  return(data.frame(x = (points.proj)[1, ] + d[1],
                    y = (points.proj)[2, ] + d[2]))
}

D ← vcov(fit_lrm)[-1,-1]
beta ← coef(fit_lrm)[-1]
ci_ellipse ← EllipseDF(D, beta, alpha = 0.05)

ggplot(post_samps, aes(x=b_sexmale, y = b_age)) +
  geom_hex() +
  geom_polygon(data = map, aes(x=x,y=y), color = "grey", alpha = 0) +
  geom_polygon(data = ci_ellipse, aes(x = x,y = y), color = "red", alpha = 0) +
  geom_point(aes(x = fixef(fit_brms)[,1][2], y = fixef(fit_brms)[,1][3]), color
       = "grey") +
  geom_point(aes(x = coef(fit_lrm)[2], y = coef(fit_lrm)[3]), color = "red") +
  theme(legend.position="none")
```



Much of the region covered by the confidence ellipse (in red) is shared by the MAP ellipse, but the point estimate appears to target the posterior mode and constructs a symmetric confidence region about it, while the MAP region is not symmetric about the posterior mean[e]. As a result, even in this simple 2-parameter logistic regression model, the confidence ellipse is likely to have problematic asymmetric coverage of the true parameter.

Note: Most of these results can be easily obtained using the `rms` package in conjunction with the `rstan` package, as shown in the Titanic case study.

[e]To be fair, there's no requirement saying that confidence intervals/regions must be symmetric, they just usually are computed that way.