Lab: Easy mode

Frank Edwards

4/24/2020

GLMs

Refresher: lm()

- We can fit a frequentist regression model with a normal likelihood (using OLS) with the lm() function.

Refresher: lm()

- We can fit a frequentist regression model with a normal likelihood (using OLS) with the lm() function.
- R formulas take on an outcome ~ predictor1 + predictor2, data structure

```r
library(broom)
data(WaffleDivorce)
m0<-lm(Divorce ~ Marriage, data = WaffleDivorce)
tidy(m0)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)    6.08     1.31        4.63 0.0000278
## 2 Marriage       0.179    0.0642      2.79 0.00751
```

Refresher: transformations and interactions

- We can use math functions like log(), exp() and sqrt() on outcomes or predictors
- To use other math, wrap a statement in I()

```
m1<-lm(log(Divorce) ~ sqrt(Marriage) + I(WaffleHouses / Population), data = WaffleDivorce)
tidy(m1)
```

```
## # A tibble: 3 x 5
##   term                       estimate std.error statistic    p.value
##   <chr>                         <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)                  1.46      0.263       5.55 0.00000128
## 2 sqrt(Marriage)               0.170     0.0588      2.89 0.00586
## 3 I(WaffleHouses/Population)   0.00699   0.00272     2.57 0.0134
```

```
m2<-lm(Divorce ~ Marriage * South, data = WaffleDivorce)
tidy(m2)
```

```
## # A tibble: 4 x 5
##   term           estimate std.error statistic    p.value
##   <chr>             <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)       6.73     1.28        5.27 0.00000351
## 2 Marriage          0.129    0.0627      2.06 0.0448
## 3 South            -6.64     4.15       -1.60 0.117
## 4 Marriage:South    0.385    0.201       1.92 0.0613
```

The GLM function and model families

- We can estimate generalized linear models using glm()
- The family argument specifies the likelihood: family = “binomial”, “gaussian”, “Gamma”, “poisson” are most common
- The canonical link function for each family is specified by default

```
admissions <- read_csv(”https://stats.idre.ucla.edu/stat/data/binary.csv”)
head(admissions)
```

```
## # A tibble: 6 x 4
##   admit   gre   gpa  rank
##   <dbl> <dbl> <dbl> <dbl>
## 1     0   380  3.61     3
## 2     1   660  3.67     3
## 3     1   800  4        1
## 4     1   640  3.19     4
## 5     0   520  2.93     4
## 6     1   760  3        2
```

Logistic regression

```
m3<-glm(admit ~ gre + gpa + factor(rank),
        data = admissions,
        family = ”binomial”)
tidy(m3)
```

```
## # A tibble: 6 x 5
##   term          estimate std.error statistic  p.value
##   <chr>            <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   -3.99     1.14         -3.50 0.000465
## 2 gre            0.00226  0.00109       2.07 0.0385
## 3 gpa            0.804    0.332         2.42 0.0154
## 4 factor(rank)2 -0.675    0.316        -2.13 0.0328
## 5 factor(rank)3 -1.34     0.345        -3.88 0.000104
## 6 factor(rank)4 -1.55     0.418        -3.71 0.000205
```

Binomial count models

```r
data("UCBadmit")
head(UCBadmit)
```

```
##   dept applicant.gender admit reject applications
## 1    A             male   512    313          825
## 2    A           female    89     19          108
## 3    B             male   353    207          560
## 4    B           female    17      8           25
## 5    C             male   120    205          325
## 6    C           female   202    391          593
```

Binomial count models

```
m4<-glm(cbind(admit, reject) ~ applicant.gender +
          factor(dept),
        data = UCBadmit, family = ”binomial”)
tidy(m4)
```

```
## # A tibble: 7 x 5
##   term                 estimate std.error statistic  p.value
##   <chr>                   <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)            0.682     0.0991     6.88  5.97e-12
## 2 applicant.gendermale  -0.0999    0.0808    -1.24  2.17e- 1
## 3 factor(dept)B         -0.0434    0.110     -0.395 6.93e- 1
## 4 factor(dept)C         -1.26      0.107    -11.8   2.41e-32
## 5 factor(dept)D         -1.29      0.106    -12.2   2.05e-34
## 6 factor(dept)E         -1.74      0.126    -13.8   2.86e-43
## 7 factor(dept)F         -3.31      0.170    -19.5   2.80e-84
```

Poisson models

```
fe<-read_csv(”./fe_demo.csv”)
head(fe)
```

```
## # A tibble: 6 x 5
##    fips state   pop deaths race.ethn
##   <dbl> <chr> <dbl>  <dbl> <chr>
## 1  1001 AL      483      0 latino
## 2  1003 AL     3218      0 latino
## 3  1005 AL      473      0 latino
## 4  1007 AL      163      0 latino
## 5  1009 AL     1713      0 latino
## 6  1011 AL      359      0 latino
```

Poisson models

```
m5<-glm(deaths ~ race.ethn,
        data = fe, family = ”poisson”)
tidy(m5)
```

```
## # A tibble: 3 x 5
##   term              estimate std.error statistic   p.value
##   <chr>                <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)         -0.569    0.0237     -24.0 3.42e-127
## 2 race.ethnlatino     -0.441    0.0379     -11.6 2.88e- 31
## 3 race.ethnwhite       0.520    0.0299      17.4 1.23e- 67
```

Poisson models with offset

```
fe<-fe %>%
  filter(pop>0)
m6<-glm(deaths ~ race.ethn,
        offset = log(pop),
        data = fe, family = ”poisson”)
tidy(m6)
```

```
## # A tibble: 3 x 5
##   term            estimate std.error statistic  p.value
##   <chr>              <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)       -8.95     0.0237    -378.   0.
## 2 race.ethnlatino   -0.732    0.0379     -19.3 3.29e-83
## 3 race.ethnwhite    -1.21     0.0299     -40.4 0.
```

Other models

- MASS::glm.nb() for negative binomial regression
- pscl::zeroinfl() for zero-inflated poisson regression
- nnet::multinom() for multinomial (categorical) regression

Frequentist multilevel models

the lme4 package

```r
library(lme4)
```

- lme4 syntax follows basic R formula syntax, but now adds variable intercept and slope terms
- basic format: outcome ~ predictors + (1|varying slope) + (varying slope | varying intercept), data = data

Estimating a multilevel model: country intercepts

```r
library(gapminder)
m7<-lmer(lifeExp ~ year + (1|country),
         data = gapminder)
tidy(m7)
```

```
## # A tibble: 4 x 5
##   term                    estimate std.error statistic group
##   <chr>                      <dbl>     <dbl>     <dbl> <chr>
## 1 (Intercept)             -586.     10.0         -58.6 fixed
## 2 year                       0.326   0.00503      64.8 fixed
## 3 sd_(Intercept).country    11.1    NA            NA   country
## 4 sd_Observation.Residual    3.58   NA            NA   Residual
```

Estimating a multilevel model: country intercepts, nested within continent intercepts

```r
m8<-lmer(lifeExp ~ year + (1|continent/country),
         data = gapminder)
tidy(m8)
```

```
## # A tibble: 5 x 5
##   term                             estimate std.error statistic group
##   <chr>                               <dbl>     <dbl>     <dbl> <chr>
## 1 (Intercept)                      -582.     10.9          -53.2 fixed
## 2 year                                0.326   0.00503      64.8 fixed
## 3 sd_(Intercept).country:continent    6.48   NA            NA   country:contine~
## 4 sd_(Intercept).continent            9.86   NA            NA   continent
## 5 sd_Observation.Residual             3.58   NA            NA   Residual
```

Varying slopes, country and continent intercepts

```r
m9<-lmer(lifeExp ~ year + (year|continent/country),
         data = gapminder)
tidy(m9)
```

```
## # A tibble: 9 x 5
##   term                               estimate std.error statistic group
##   <chr>                                 <dbl>     <dbl>     <dbl> <chr>
## 1 (Intercept)                      -587.          10.7    -55.0   fixed
## 2 year                                0.329        1.65     0.200 fixed
## 3 sd_(Intercept).country:continent    3.12        NA       NA     country:conti~
## 4 sd_year.country:continent           0.00488     NA       NA     country:conti~
## 5 cor_(Intercept).year.country:co~   -0.999       NA       NA     country:conti~
## 6 sd_(Intercept).continent            7.98        NA       NA     continent
## 7 sd_year.continent                   3.68        NA       NA     continent
## 8 cor_(Intercept).year.continent      0.00249     NA       NA     continent
## 9 sd_Observation.Residual             3.55        NA       NA     Residual
```

GLMs and lme4, Poisson model with state intercepts

```r
m10<-glmer(deaths ~ race.ethn + (1|state),
           data = fe, family = ”poisson”)
tidy(m10)
```

```
## # A tibble: 4 x 6
##   term                 estimate std.error statistic   p.value group
##   <chr>                   <dbl>     <dbl>     <dbl>     <dbl> <chr>
## 1 (Intercept)            -0.692    0.142      -4.86  1.17e- 6 fixed
## 2 race.ethnlatino        -0.465    0.0378    -12.3   8.44e-35 fixed
## 3 race.ethnwhite          0.491    0.0299     16.4   1.49e-60 fixed
## 4 sd_(Intercept).state    0.995   NA          NA    NA        state
```

Add an offset

```
m11<-glmer(deaths ~ race.ethn + offset(log(pop)) +
             (1|state),
           data = fe, family = ”poisson”)
tidy(m11)
```

```
## # A tibble: 4 x 6
##   term                  estimate std.error statistic    p.value group
##   <chr>                    <dbl>     <dbl>     <dbl>      <dbl> <chr>
## 1 (Intercept)             -8.86     0.0708    -125.   0.        fixed
## 2 race.ethnlatino         -0.997    0.0408     -24.5  3.39e-132 fixed
## 3 race.ethnwhite          -1.30     0.0314     -41.4  0.        fixed
## 4 sd_(Intercept).state     0.456   NA           NA   NA         state
```

Add overdispersion

We can model overdispersion in counts with a variable intercept for each observation. This effectively creates an error term for a model that typically lacks one.

```
fe<-fe %>%
  mutate(obs_n = 1:nrow(fe))
```

Add overdispersion

```
m12<-glmer(deaths ~ race.ethn + offset(log(pop)) +
             (1|state) + (1|obs_n),
           data = fe, family = ”poisson”)
tidy(m12)
```

```
## # A tibble: 5 x 6
##   term                 estimate std.error statistic    p.value group
##   <chr>                   <dbl>     <dbl>     <dbl>      <dbl> <chr>
## 1 (Intercept)            -9.02     0.0747    -121.   0.        fixed
## 2 race.ethnlatino        -0.976    0.0572     -17.1  3.25e- 65 fixed
## 3 race.ethnwhite         -1.19     0.0417     -28.4  5.08e-178 fixed
## 4 sd_(Intercept).obs_n    0.418   NA           NA   NA         obs_n
## 5 sd_(Intercept).state    0.446   NA           NA   NA         state
```

Bayesian models with lme4 and glm syntax

The brms package

- brms uses the core R and lme4 formula syntax, but uses rstan and HMC to fit models
- we can specify priors relatively easily
- For a full translation of Statistical Rethinking into brms, see this very detailed guide: `https://bookdown.org/ajkurz/Statistical_Rethinking_recoded/`

brms basics

```r
library(brms)
brm0<-brm(Divorce ~ Marriage,
        data = WaffleDivorce,
        prior = c(prior(normal(0,2), class = Intercept),
                  prior(normal(0,2), class = b, coef = Marriage),
                  prior(exponential(1), class = sigma))
        )
```

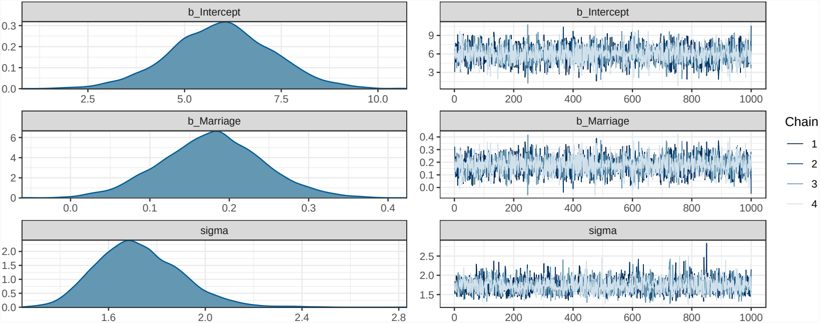
Results

```r
summary(brm0)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: Divorce ~ Marriage
##    Data: WaffleDivorce (Number of observations: 50)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     5.95      1.33     3.28     8.57 1.00     3499     2752
## Marriage      0.18      0.06     0.05     0.31 1.00     3438     2746
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     1.73      0.18     1.43     2.10 1.00     3957     2965
##
## 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).
```

Traceplots and density plots

```
## so easy!
plot(brm0)
```



Directly sampling the posterior

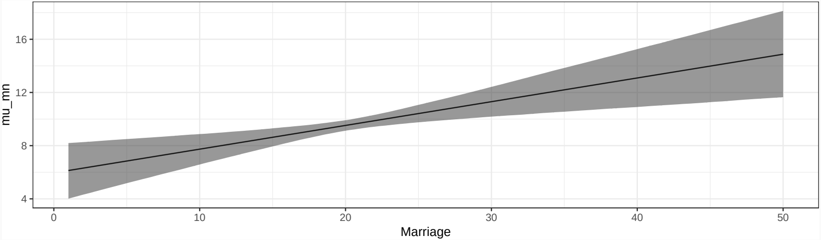
```r
post<-posterior_samples(brm0)
head(post)
```

```
##   b_Intercept b_Marriage    sigma      lp__
## 1    6.077008  0.1755933 1.803016 -112.9486
## 2    5.806530  0.1884207 1.524815 -113.0423
## 3    5.985369  0.1850693 1.736283 -112.9152
## 4    4.675092  0.2417333 1.630557 -113.1536
## 5    5.894895  0.1619366 1.567004 -114.6464
## 6    5.973628  0.1716880 1.990871 -114.0798
```

Using the linear link function and new data

```
sim_dat<-data.frame(Marriage = seq(1, 50, by = 0.1))
post_mu<-posterior_linpred(brm0, newdata = sim_dat)

sim_dat<-sim_dat %>%
  mutate(mu_lwr = apply(post_mu, 2, function(x) quantile(x, 0.05)),
         mu_upr = apply(post_mu, 2, function(x)quantile(x, 0.95)),
         mu_mn = apply(post_mu, 2, mean))
ggplot(sim_dat, aes(x = Marriage, y = mu_mn)) +
  geom_line() +
  geom_ribbon(aes(ymin = mu_lwr, ymax = mu_upr), alpha = 0.5)
```



GLMs: logistic regression

```
brm1<-brm(admit ~ gre + gpa,
          family = bernoulli,
          data = admissions,
          prior = c(prior(normal(0.5, 1), class = Intercept),
                    prior(normal(0, 1), class = b)))
```

Output

```r
summary(brm1)
```

```
##  Family: bernoulli 
##   Links: mu = logit 
## Formula: admit ~ gre + gpa 
##    Data: admissions (Number of observations: 400) 
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
## 
## Population-Level Effects: 
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    -4.75      1.02    -6.80    -2.78 1.00     2468     2461
## gre           0.00      0.00     0.00     0.00 1.00     4176     3002
## gpa           0.68      0.30     0.10     1.25 1.00     2121     1811
## 
## 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).
```

Multilevel models

- brms is built to accept the syntax of lme4 - like formulas

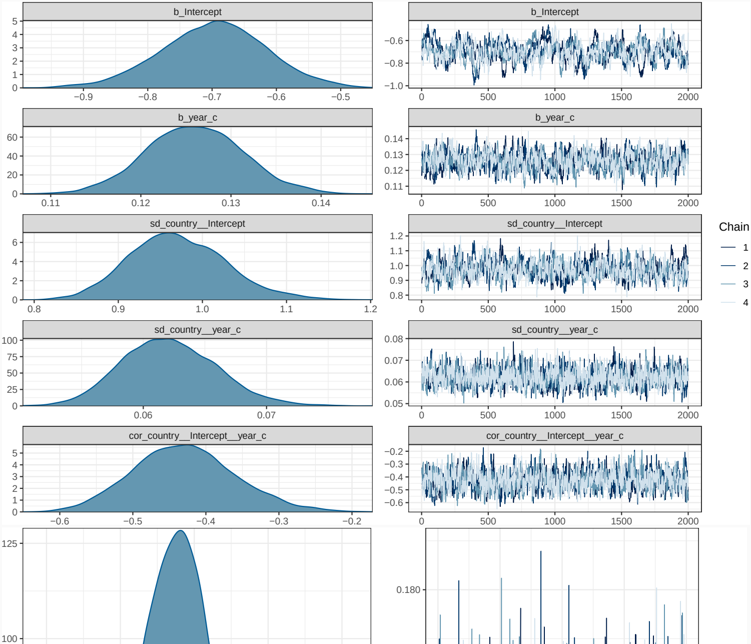
```r
gapminder<-gapminder %>%
  mutate(L_c = scale(lifeExp),
         year_c = ((year - min(year))/5))
brm7<-brm(L_c ~ year_c + (year_c|country),
         data = gapminder, family = gaussian,
         prior = c(prior(normal(0, 5), class = Intercept),
                   prior(exponential(1), class = sigma),
                   prior(normal(0,2), class = b),
                   prior(lkj(2), class = cor)),
         iter = 4000, cores = 4)
```

Output

```r
summary(brm7)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: L_c ~ year_c + (year_c | country)
##    Data: gapminder (Number of observations: 1704)
## Samples: 4 chains, each with iter = 4000; warmup = 2000; thin = 1;
##          total post-warmup samples = 8000
##
## Group-Level Effects:
## ~country (Number of levels: 142)
##                       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sd(Intercept)             0.97      0.06     0.87     1.09 1.01      399
## sd(year_c)                0.06      0.00     0.06     0.07 1.00      907
## cor(Intercept,year_c)    -0.43      0.07    -0.56    -0.28 1.01      681
##                       Tail_ESS
## sd(Intercept)             1034
## sd(year_c)                1937
## cor(Intercept,year_c)     1529
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    -0.70      0.08    -0.86    -0.54 1.01      224      421
## year_c        0.13      0.01     0.12     0.14 1.00      524     1158
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.17      0.00     0.16     0.18 1.00     8406     6097
##
```

Check convergence



Further reading

- An Introduction to Bayesian Multilevel Models Using brms: A Case Study of Gender Effects on Vowel Variability in Standard Indonesian `https://osf.io/dpzcb/`
- Fitting Linear Mixed-Effects Models Using lme4: `https://cran.r-project.org/web/packages/lme4/vignettes/lmer.pdf`
- brms: An R Package for Bayesian Multilevel Models using Stan: `https://cran.r-project.org/web/packages/brms/vignettes/brms_overview.pdf`