Hierarchical models

Stat 340, Fall 2021

Example: ELS math scores

Today, we'll consider data from the 2002 Educational Longitudinal Study (ELS), a survey of schools across the United States. Data were collected by sampling schools and then sampling students within each selected school. We'll focus on 10th grade math scores from a sample of 10 schools. The tests contained items in arithmetic, algebra, geometry, data/probability, and advanced topics were divided into process categories of skill/knowledge, understanding/ comprehension, and problem solving.

There are a few types of questions that we might ask of these data:

- What's the typical math score?
- To what extent do scores vary from school to school?
- For any single school, how much might scores vary from student to student?

Competing models

There are three ways we might think to model the data:

1. **Complete pooling (combined estimates)** Ignore schools and lump all students together
2. **No pooling (separate groups)** Separately analyze each school and assume that one school's data doesn't contain valuable information about another school
3. **Partial pooling (compromise estimates)** Acknowledge the grouping structure, so that even though schools differ in performance, they might share valuable information about each other and about the broader population of schools

Complete pooling model

Your turn

Suppose we decide to pool all of the scores together, disregarding what school each comes from.

Let Y_i denote the exam score for student i.

Write down a model (likelihood and priors) that could be used for the exams scores in this setting. (Just list distributions, don't do any algebra.)

Fitting the complete pooling model in JAGS

First, write the model in a way JAGS understands:

```
complete_pooling <-"
model {
## sampling
for (i in 1:N){
   y[i] ~ dnorm(mu, phi)
```

```
}

## priors
mu ~ dnorm(50, 1/25)  # remember to use precision here
phi ~ dgamma(1, 1/100)
sigma <- sqrt(pow(phi, -1))
}
"
```

Now, we can use `runjags` to fit the model via MCMC:

```
pooled_model <- run.jags(
  complete_pooling,
  data = list(y = sub_school$mathscore, N = length(sub_school$mathscore)),
  inits = list(list(mu = rnorm(1, 50, sd = 5), phi = runif(1, .1, .3)),
               list(mu = rnorm(1, 50, sd = 5), phi = runif(1, .1, .3)),
               list(mu = rnorm(1, 50, sd = 5), phi = runif(1, .1, .3))),
  monitor = c("mu", "sigma"),
  n.chains = 3,
  sample = 5000,
  silent.jags = TRUE
)
```

Then, you can diagnose convergence:

```
mcmc_trace(pooled_model$mcmc)
mcmc_dens_overlay(pooled_model$mcmc)
mcmc_acf(pooled_model$mcmc)
```

And finally summarize the posterior:

```
print(pooled_model, digits = 3)
```

No pooling model

Your turn

Suppose we decide to model each school's scores separately. In this case, we would need 10 models, one for each school. Let Y_{ij} denote the exam score for student i in school j. (This is the book's notation). Write down a model (likelihood and priors) that could be used for the exams scores in school j.

Fitting the no pooling model in JAGS

First, write the model in a way JAGS understands:

```
no_pooling <- "
  model {
  ## Sampling
```

```
    for (i in 1:N){
      y[i] ~ dnorm(mu[school[i]], phi)
    }

    ## Priors
    for (j in 1:J){
      mu[j]  ~ dnorm(50, 1/100)
    }

    phi ~ dgamma(1, 1/100)
    sigma <- 1 / sqrt(phi)
  }
"
```

Now, we can use `runjags` to fit the model via MCMC:

```
no_pooled_model <- run.jags(
  no_pooling,
  data = list(y = sub_school$mathscore, school = sub_school$school,
              N = nrow(sub_school), J = n_distinct(sub_school$school)),
  monitor = c("mu", "sigma"),
  n.chains = 3,
  sample = 5000,
  silent.jags = TRUE
)
```

Then, you can diagnose convergence.

```
mcmc_trace(no_pooled_model$mcmc)
mcmc_dens_overlay(no_pooled_model$mcmc)
mcmc_acf(no_pooled_model$mcmc)
```

And finally summarize the posterior:

```
print(no_pooled_model, digits = 3)
```

Hierarchical model (the compromise)

By using a *two-stage prior* specification, we can reach a compromise between the two models:

Fitting the hierarchical model in JAGS

First, write the model in a way JAGS understands:

```
modelString <-"
model {

## sampling
for (i in 1:N){
   y[i] ~ dnorm(mu_j[school[i]], invsigma2)
}

## priors
for (j in 1:J){
   mu_j[j] ~ dnorm(mu, invtau2)
}

invsigma2 ~ dgamma(a_s, b_s)
sigma <- sqrt(pow(invsigma2, -1))

## hyperpriors
mu ~ dnorm(mu0, g0)
invtau2 ~ dgamma(a_t, b_t)
tau <- sqrt(pow(invtau2, -1))
}
"
```

]

Next, define the data and any initial values for your parameters:

```
y <- sub_school$mathscore
school <- sub_school$school
N <- length(y)
J <- length(unique(school))
the_data <- list(y = y, school = school,
                 N = N, J = J,
                 mu0 = 50, g0 = .04,  # prior parameters
                 a_t = 1, b_t = .01,  # hyperparameters
                 a_s = 1, b_s = .01)  # hyperparameters
```

Now, we can use `runjags` to fit the model via MCMC:

```
posterior <- run.jags(
  modelString,
  n.chains = 1,
  data = the_data,
  monitor = c("mu", "tau", "mu_j", "sigma"),
  adapt = 1000,
  burnin = 5000,
  sample = 5000,
  silent.jags = TRUE
)
```

Diagnosing convergence is the same idea, so let's focus on summarizing the posterior. You can print a summary:

```
print(posterior, digits = 3)
```

Or create plots of each parameter:

```
mcmc_intervals(posterior$mcmc, regex_pars = "mu")
mcmc_intervals(posterior$mcmc, pars = c("sigma", "tau"))
```