# Class 7: Multi-layer hierarchical models

Andrew Parnell
andrew.parnell@mu.ie

Maynooth University
National University of Ireland Maynooth

## Learning outcomes:

- Understand how to add in multiple layers to a hierarchical model
- Follow a detailed example of building a model
- Be able to work with missing data in JAGS

## Some new terminology

- Most of the models we have covered so far contain only one hidden or *latent* set of parameters
- For example, the data $y$ may depend on a parameter $\beta$, which itself depends on a parameter $\theta$. $\theta$ is given a prior distribution
- We say that the data are at the 'top level', the parameter $\beta$ is a *latent parameter* at the second level, and the hyper-parameter $\theta$ is also a latent parameter at the third level
- We say that the prior distribution on $\beta$ is *conditional* on $\theta$, whilst the prior distribution (if it just involves numbers) is a *marginal prior distribution*

## What is a multi-layer model?

- A multi-layer model is one where have many (usually more than 2 or 3) layers of parameters conditional on each other
- It's very straightforward to add in these extra layers in JAGS/Stan
- The question is whether they are necessary or not, and how much the data can tell us about them

## A simple example

- We will work through the `earnings` example, extending it to produce a much more complex model
- The model we last met had the following JAGS code:

```
'
...
y[i] ~ dnorm(alpha[eth[i]] +
                beta[eth[i]]*(x[i] - mean(x)),
                  sigma^-2)
}
...
# Priors
for(j in 1:N_eth) {
  alpha[j] ~ dnorm(mu_alpha, sigma_alpha^-2)
  beta[j] ~ dnorm(mu_beta, sigma_beta^-2)
}'
```

## Adding extra layers

- The priors for the hyper-parameters at the bottom of the model were

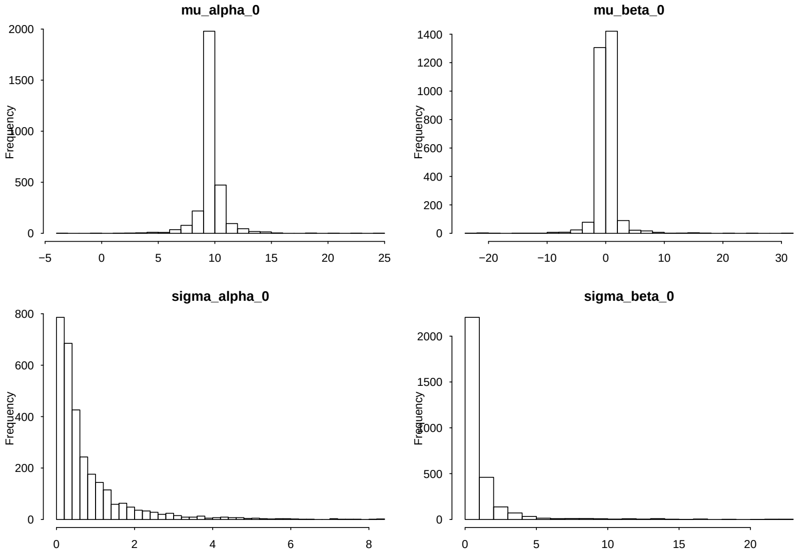
```
'
  ...
  mu_alpha ~ dnorm(11, 2^-2)
  mu_beta ~ dnorm(0, 0.1^-2)
  sigma ~ dunif(0, 5)
  sigma_alpha ~ dunif(0, 2)
  sigma_beta ~ dunif(0, 2)
}
'
```

- There is nothing to stop us adding in extra parameters beneath these layers

## New JAGS code

```
jags_code = '
model{
  # Likelihood
  for(i in 1:N) {
    y[i] ~ dnorm(alpha[eth[i]] +
                  beta[eth[i]]*(x[i] - mean(x)),
                    sigma^-2)
  }
  # Priors
  for(j in 1:N_eth) {
    alpha[j] ~ dnorm(mu_alpha, sigma_alpha^-2)
    beta[j] ~ dnorm(mu_beta, sigma_beta^-2)
  }
  mu_alpha ~ dnorm(mu_alpha_0, sigma_alpha_0^-2)
  mu_alpha_0 ~ dnorm(0, 20^-2)
  sigma_alpha_0 ~ dt(0,5,1)T(0,)
  mu_beta ~ dnorm(mu_beta_0, sigma_beta_0^-2)
  mu_beta_0 ~ dnorm(0, 20^-2)
  sigma_beta_0 ~ dt(0,5,1)T(0,)
  sigma ~ dunif(0, 5)
  sigma_alpha ~ dt(0,5,1)T(0,)
  sigma_beta ~ dt(0,5,1)T(0,)
}
'
```

## What information is there about these extra parameters?



## How many layers should I add?

- We could go on adding layers here if we wanted to, but it's not clear what benefit it would have
- For example, do we really need to know the standard deviation of the mean of the intercept? The answer to this will depend on the questions of interest, and the amount and type of prior information available
- A general rule is, in the absence of any strong prior information, to add one extra layer of parameters beyond the latent parameters of key interest

## Writing the same model in different ways

- The model on the previous slides has a different intercept and slope for each ethnicity group, with the information about them tied together through the prior distributions on them
- The likelihood was written as:

```
y[i] ~ dnorm(alpha[eth[i]] +
               beta[eth[i]]*(x[i] - mean(x)),
                 sigma^-2)
```

which in maths can be written as:

$$y_i \sim N(\alpha_{\text{eth}_i} + \beta_{\text{eth}_i} x_i, \sigma^2)$$

where $\text{eth}_i$ takes the values 1, 2, 3, or 4

- Remember, $y_i$ is the log-earnings of individual $i$ where $i = 1, \ldots, N$

## Re-writing the model

- Commonly you'll see $y$ here re-defined as $y_{ij}$ where $j = 1, .., 4$ represents ethnicity, and $i = 1, \ldots, N_j$ is the number of individuals with ethnicity $j$
- The likelihood can then be written as:

$$y_{ij} \sim N(\alpha_j + \beta_j x_i, \sigma^2)$$

- Note that this is exactly the same model, just re-written slightly differently. In fact, this latter model is much harder to write out in JAGS/Stan code

## Fixed vs random effect models

- Thinking about this model in more detail

$$y_{ij} \sim N(\alpha_j + \beta_j x_i, \sigma^2)$$

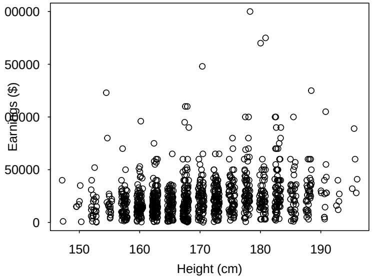
- If the $\alpha_j$ and $\beta_j$ parameters are all given independent prior distributions, e.g. $\alpha_j \sim N(0, 100)$ or similar, then this is considered a *fixed effects* model
- If the $\alpha_j$ and $\beta_j$ are given prior distributions that tie the values together, e.g. $\alpha_j \sim N(\mu_\alpha, \sigma^2_\alpha)$, then this is often called a *random effects* model
- (In fact, nobody can agree on what a fixed or random effects model actually is)
- If you have multiple different types of priors then this is called a *mixed effects* model

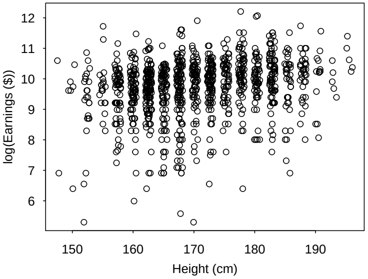
## Mixed effects vs hierarchical models

- The hierarchical models we have been studying all use the *random effects* approach wherever possible
- The big advantage of using this approach is that we get to *borrow* strength between the different groups (here `eth`, but it could be anything)
- Whenever we have a categorical covariate we should always be putting a constraining/tying prior distribution on them, and looking at how the effects vary between the groups
- This is strongly linked to the idea of *partial pooling* which we will meet later in the course

# Example: multi-layer earnings data

- We will now go through and build a much more complicated model for the earnings data, taken from the Gelman and Hill book, using only weak priors
- We can generate data from these models (either using the prior or the posterior), and we can draw a DAG
- Our goal is to explore the factors which explain earnings. We have variables on height, age, and ethnicity.
- If we first plot the data





## Transformations

- From the left-hand plot there seem to be quite a few extreme observations, and there's a possibility that the relationship between height and earnings is non-linear
- The right-hand plot seems to have stabilised most of the extreme observations, and perhaps linearity is more appropriate
- Notice that a linear model implies:

$$y_i = \alpha + \beta x_i + \epsilon_i$$

whilst the log-linear model implies:

$$y_i = \exp(\alpha + \beta x_i + \epsilon_i) = e^{\alpha} \times e^{\beta x_i} \times e_i^{\epsilon}$$

so the coefficients, once exponentiated, have multiplicative effects that are relatively easy to interpret

## Fitting the first model

- If we fit a model with just height (mean centered) we get the following JAGS output

```
## Inference for Bugs model at "5", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded)
##  n.sims = 3000 iterations saved
##             mu.vect sd.vect     2.5%      25%      50%      75%    97.5%
## alpha         9.738   0.028    9.684    9.718    9.738    9.757    9.791
## beta_height   0.023   0.003    0.017    0.021    0.022    0.024    0.028
## sigma         0.906   0.020    0.869    0.893    0.906    0.919    0.948
## deviance   2798.511   2.430 2795.735 2796.728 2797.876 2799.646 2804.863
##             Rhat n.eff
## alpha      1.001  3000
## beta_height 1.001  3000
## sigma      1.002  1200
## deviance   1.000  3000
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 3.0 and DIC = 2801.5
## DIC is an estimate of expected predictive error (lower deviance is better).
```

## Interpreting the parameters

- These parameters are directly interpretable:
  - The mean of the log earnings at the mean height is about 9.737, which is about 17k on the original scale
  - We can also get e.g. a 95% confidence interval using the JAGS output. From 16000 to 18000
  - For every extra cm so you gain 0.0225 on the log scale, i.e. an 2.28% gain in income
  - From the posterior of $\sigma$, we can guess that about 68% of predictions will be within 0.906 on the log scale or within a factor of about 2.48 of the prediction
- Interpretation for the intercept would have been harder had we not mean-centered the height variable
- The DIC is 2801.47 with 2.95 effective parameters

## Improving the model

- Now suppose we fit a model with a random intercept for ethnicity

```
jags_code = '
model{
  # Likelihood
  for(i in 1:N) {
    log_earn[i] ~ dnorm(alpha_eth[eth[i]] +
                    beta_height*(height[i] - mean(height)),
                    sigma^-2)
  }
  # Priors
  for(j in 1:N_eth) {
    alpha_eth[j] ~ dnorm(mu_eth, sigma_eth^-2)
  }
  beta_height ~ dnorm(0, 20^-2)
  mu_eth ~ dnorm(0, 20^-2)
  sigma_eth ~ dt(0,10,1)T(0,)
  sigma ~ dt(0,10,1)T(0,)
}
'
```

## Improving the model 2

```
## Inference for Bugs model at "6", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded)
##  n.sims = 3000 iterations saved
##               mu.vect sd.vect     2.5%      25%      50%      75%    97.5%
## alpha_eth[1]    9.692   0.072    9.536    9.647    9.702    9.744    9.812
## alpha_eth[2]    9.674   0.091    9.465    9.619    9.692    9.741    9.811
## alpha_eth[3]    9.749   0.029    9.690    9.730    9.749    9.769    9.806
## alpha_eth[4]    9.742   0.105    9.532    9.686    9.739    9.789    9.981
## beta_height     0.023   0.003    0.017    0.021    0.022    0.024    0.028
## mu_eth          9.709   0.090    9.504    9.671    9.721    9.757    9.861
## sigma           0.907   0.020    0.869    0.893    0.906    0.920    0.947
## deviance     2797.828   2.787 2793.869 2795.782 2797.318 2799.319 2804.569
##               Rhat n.eff
## alpha_eth[1] 1.004   850
## alpha_eth[2] 1.014   220
## alpha_eth[3] 1.007   340
## alpha_eth[4] 1.007  3000
## beta_height  1.001  3000
## mu_eth       1.031   830
## sigma        1.001  3000
## deviance     1.006   400
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 3.9 and DIC = 2801.7
## DIC is an estimate of expected predictive error (lower deviance is better).
```

## Interpreting the output

- The parameters $\alpha$ and $\beta_{\text{height}}$ haven't changed much in the mean
- The 95% confidence interval for $\alpha$ has increased: 13000 to 19000
- The DIC is 2801.7 with 3.87 effective parameters. Pretty much the same as above
- We also have estimates for each ethnicity, none of these have a strong effect away from zero

## Now an interaction model

```
jags_code = '
model{
  # Likelihood
  for(i in 1:N) {
    log_earn[i] ~ dnorm(alpha_eth[eth[i]] +
                    beta_height[eth[i]]*(height[i] - mean(height)),
                    sigma^-2)
  }
  # Priors
  for(j in 1:N_eth) {
    alpha_eth[j] ~ dnorm(mu_eth, sigma_eth^-2)
    beta_height[j] ~ dnorm(mu_beta_height, sigma_height^-2)
  }
  mu_beta_height ~ dnorm(0, 20^-2)
  mu_eth ~ dnorm(0, 20^-2)
  sigma_eth ~ dt(0,10,1)T(0,)
  sigma_height ~ dt(0,10,1)T(0,)
  sigma ~ dt(0,10,1)T(0,)
}
'
```

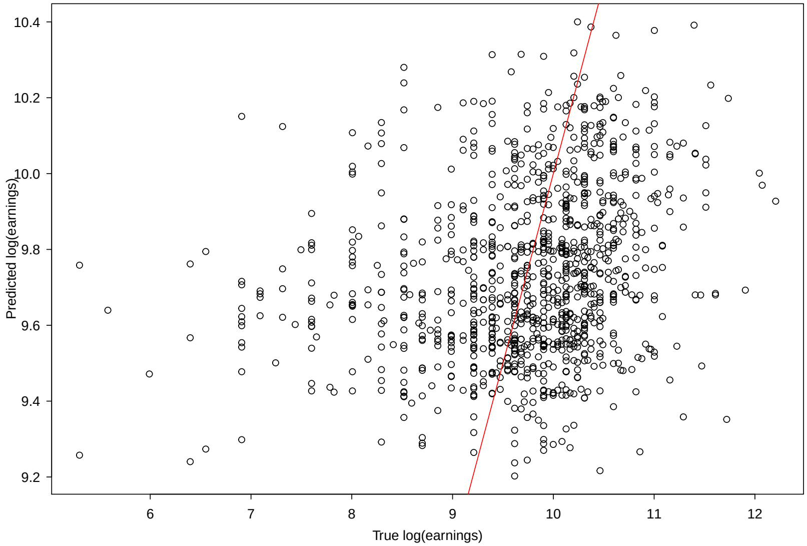
## Interaction model results

```
## Inference for Bugs model at "7", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded)
##  n.sims = 3000 iterations saved
##                 mu.vect sd.vect     2.5%      25%      50%      75%
## alpha_eth[1]      9.680   0.073    9.525    9.634    9.690    9.731
## alpha_eth[2]      9.658   0.092    9.443    9.602    9.673    9.723
## alpha_eth[3]      9.745   0.030    9.688    9.724    9.744    9.764
## alpha_eth[4]      9.719   0.109    9.478    9.666    9.723    9.769
## beta_height[1]    0.009   0.009   -0.008    0.004    0.009    0.015
## beta_height[2]    0.013   0.010   -0.008    0.006    0.013    0.019
## beta_height[3]    0.025   0.003    0.019    0.023    0.025    0.027
## beta_height[4]    0.008   0.016   -0.027   -0.002    0.010    0.019
## mu_beta_height    0.014   0.016   -0.020    0.008    0.015    0.021
## mu_eth            9.701   0.083    9.520    9.663    9.712    9.746
## sigma             0.905   0.019    0.868    0.892    0.905    0.918
## sigma_height      0.021   0.019    0.002    0.009    0.015    0.026
## deviance       2793.105   3.628 2787.550 2790.451 2792.622 2795.232
##                   97.5%  Rhat n.eff
## alpha_eth[1]      9.804 1.002  1600
## alpha_eth[2]      9.797 1.004   570
## alpha_eth[3]      9.806 1.007   310
## alpha_eth[4]      9.954 1.007   480
## beta_height[1]    0.025 1.003   810
## beta_height[2]    0.030 1.004   530
## beta_height[3]    0.031 1.003   710
## beta_height[4]    0.034 1.001  2500
## mu_beta_height    0.041 1.003  2000
## mu_eth            9.854 1.009  3000
## sigma             0.944 1.001  2400
## sigma_height      0.077 1.021   210
## deviance       2801.199 1.001  2500
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
```

## Interpreting the output

- The model has improved a bit, DIC now 2799.68 with 6.58 effective parameters
- The confidence intervals for the different slopes are highly different, with the whites group (ethnicity $= 3$) having a much clearer relationship with height, possibly due to the large sample size
- Go back to the previous classes to see plots of these effects

# Checking the model - posterior predictive fit



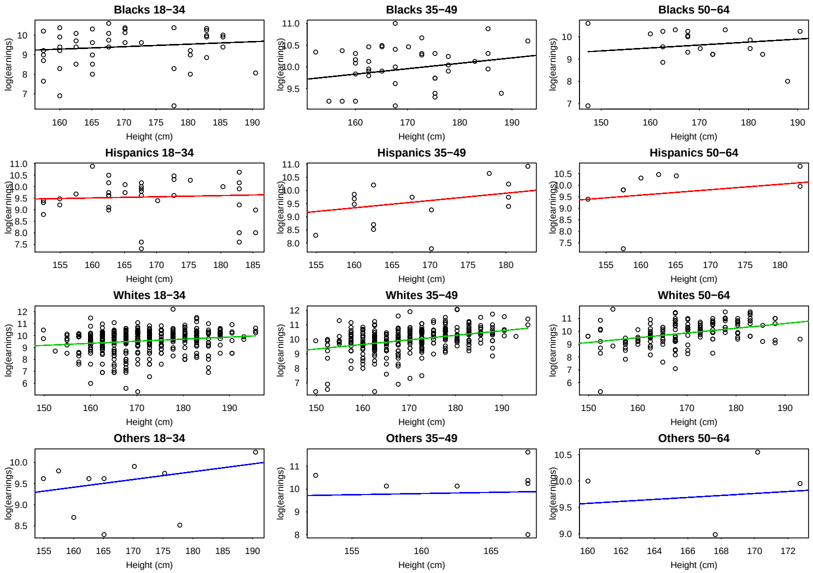
## Introducing age

- Let's fit an even more complicated model with intercepts and slopes varying by ethnicity and age group
- Age is divided up into three groups 1: 18-34, 2: 35-49, and 3: 50-64
- We want to know whether the degree to which height affects earnings for different ethnic/age group combinations

## JAGS model

```
jags_code = '
model{
  # Likelihood
  for(i in 1:N) {
    log_earn[i] ~ dnorm(alpha[eth[i],age_grp[i]] +
                          beta[eth[i],age_grp[i]]*(height[i] - mean(height)),
                    sigma^-2)
  }
  # Priors
  for(j in 1:N_eth) {
    for(k in 1:N_age_grp) {
      alpha[j,k] ~ dnorm(mu_alpha, sigma_alpha^-2)
      beta[j,k] ~ dnorm(mu_beta, sigma_beta^-2)
    }
  }
  mu_alpha ~ dnorm(0, 20^-2)
  mu_beta ~ dnorm(0, 20^-2)
  sigma_alpha ~ dt(0,10,1)T(0,)
  sigma_beta ~ dt(0,10,1)T(0,)
  sigma ~ dt(0,10,1)T(0,)
}
'
```

## Model output



## More about this model

- So we now have varying effects - we should also plot the uncertainties in these lines (see practical)
- The DIC here is now DIC now 2737.85 with 19.71 effective parameters - a big drop!

## Missing and unbalanced data

- There are many definitions of what 'unbalanced' data means in statistics. Usually we mean that there are different numbers of observations in each group. Our format of writing e.g. $y_i \sim N(\alpha_{\mathrm{eth}_i} + \beta_{\mathrm{eth}_i} x_i, \sigma^2)$ allows us to deal with unbalanced data naturally
- Usually the smaller the sample size of the group the more uncertain the posterior distribution will be
- But what if we have some missing data? There are different types, and some need to be more carefully treated than others

## Different types of missing data

- There are many different types of missing data:
  - Missing response variables
  - Missing covariates
  - Missingness that occurs completely at random
  - Missingness that occurs as a consequence of the experiment or the data
- The first three are all very easy to deal with in JAGS (less so in Stan). The last one is much harder, and not something we will go into in any detail. It requires building a separate model for the missingness process

## The simple way of dealing with missing data in JAGS

- In JAGS it is absolutely trivial to to deal with missingness in the response variable. You simply fill in the missing values with `NA`
- JAGS then treats them as parameters to be estimated. You can 'watch' them in the normal way or just ignore them. You thus have the option of getting a posterior distribution of the missing data points
- Suppose we shoved in some NA values into our data

```
dat2 = dat
dat2$earn[c(177, 763, 771)] = NA
```

## Running the model with missingness

```r
print(jags_run)
```

```
## Inference for Bugs model at "5", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded)
##  n.sims = 3000 iterations saved
##                mu.vect sd.vect     2.5%      25%      50%      75%
## log_earn[177]   10.355   0.867    8.658    9.771   10.362   10.923
## log_earn[763]    9.506   0.868    7.832    8.901    9.520   10.111
## log_earn[771]    9.461   0.873    7.747    8.868    9.476   10.054
## deviance      2705.525   6.842 2694.115 2700.617 2704.980 2709.655
##                  97.5%  Rhat n.eff
## log_earn[177]   12.048 1.001  3000
## log_earn[763]   11.177 1.001  2700
## log_earn[771]   11.134 1.001  3000
## deviance      2720.454 1.024   130
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 23.0 and DIC = 2728.6
## DIC is an estimate of expected predictive error (lower deviance is better).
```

## More complex varieties of missing data

- If you have missing covariates or joint missing covariates/response variables, you can include these too
- The only extra issue is that you need to give JAGS a prior distribution for the missing covariate values which can make the code a bit fiddlier
- If the response variable (e.g. log earnings) exists but the covariate value is missing, then you are asking JAGS to perform an *inverse regression*
- In Stan missing data is fiddlier to incorporate as you have to separate out the parameters (i.e. missing data) from the observed data

## Summary

- We have seen how to create some rich multi-layers models
- We have gone through quite a detailed example
- We have learnt about the multivariate normal distribution
- We have discovered how to deal with missing and unbalanced data sets