From fixed to random effects

Bayesian statistics 4 – random and mixed effects models

Frédéric Barraquand (CNRS, IMB)

09/11/2021

Some things that we learned the last time(s)

Session 3

- MCMC = Monte Carlo + Markov Chain
- Requires two types of convergence to compute an posterior means or posterior distribution
- JAGS uses the Gibbs sampler, a multicomponent variant of the Metropolis algorithm
- The Gibbs sampler allows to sample parameter-rich models

Session 2

- T-tests, ANOVA and the likes can be framed as the General Linear Model
- The Linear Model $Y = X\beta + E$ is easily fitted with JAGS
- Uncertainties in effects $\rightarrow$ posteriors

Back to Snout-Vent Length (SVL) Snake data

From Kéry (2010) & TD 2

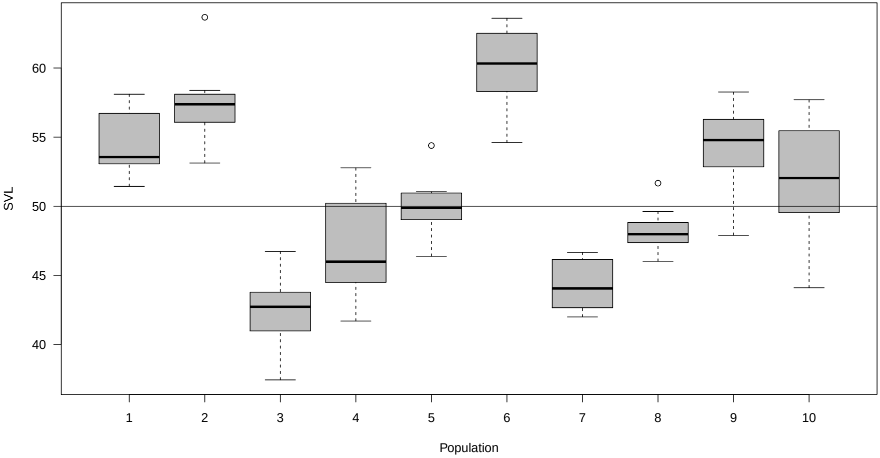
```
### Data generation
# same as TD2 but number of groups x 2
npop <- 10                # Number of populations: now choose 10 rath
nsample <- 12                 # Number of snakes in each
n <- npop * nsample           # Total number of data points

pop.grand.mean <- 50              # Grand mean SVL
pop.sd <- 5               # sd of population effects about mean
pop.means <- rnorm(n = npop, mean = pop.grand.mean, sd = pop.sd)
sigma <- 3                # Residual sd
eps <- rnorm(n, 0, sigma)         # Draw residuals

x <- rep(1:npop, rep(nsample, npop))
X <- as.matrix(model.matrix(~ as.factor(x)-1))
y <- as.numeric(X %*% as.matrix(pop.means) + eps) # as.numeric is E
```

The data: Snout-vent length in snakes

```
boxplot(y ~ x, col = "grey", xlab = "Population", ylab = "SVL", mai
abline(h = pop.grand.mean)
```



Questions that we could ask

- Effect of being in population i
- Is there more variation between populations or more residual variation?

The one-way ANOVA: from code to maths

$J = 10$ Groups. Notations

$$Y_{ij} = \alpha_j + \epsilon_{ij}, \epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$$

Practical if $i = 1, ..., I$ is the same number of individuals per group. $n = I \times J$.

The one-way ANOVA: from code to maths

$J = 10$ Groups. Notations

$$Y_{ij} = \alpha_j + \epsilon_{ij}, \epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$$

Practical if $i = 1, ..., I$ is the same number of individuals per group. $n = I \times J$.

or again

$$Y_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma^2), \mu_{ij} = \alpha_j$$

By that we mean that $\mathbb{E}(Y_{ij}) = \alpha_j$.

The one-way ANOVA: from code to maths

$J = 10$ Groups. Notations

$$Y_{ij} = \alpha_j + \epsilon_{ij}, \epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$$

Practical if $i = 1, ..., I$ is the same number of individuals per group. $n = I \times J$.

or again

$$Y_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma^2), \mu_{ij} = \alpha_j$$

By that we mean that $\mathbb{E}(Y_{ij}) = \alpha_j$.

Note: if you have an overall mean μ you need to remove a group

The one-way ANOVA: from code to maths

$J = 10$ Groups. Notations

$$Y_{ij} = \alpha_j + \epsilon_{ij}, \epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$$

Practical if $i = 1, ..., I$ is the same number of individuals per group. $n = I \times J$.

or again

$$Y_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma^2), \mu_{ij} = \alpha_j$$

By that we mean that $\mathbb{E}(Y_{ij}) = \alpha_j$.

Note: if you have an overall mean μ you need to remove a group

or again with $i = 1, ..., n$

$$Y_i \sim \mathcal{N}(\mu_i, \sigma^2), \mu_i = \alpha_{k[i]}$$

where $k[i]$ returns in which group is i. How we coded this JAGS.

Running again the ANOVA

```
## List of 2
##  $ y: num [1:120] 53.2 58.1 56.8 53 53.5 ...
##  $ x: int [1:120] 1 1 1 1 1 1 1 1 1 1 ...
```

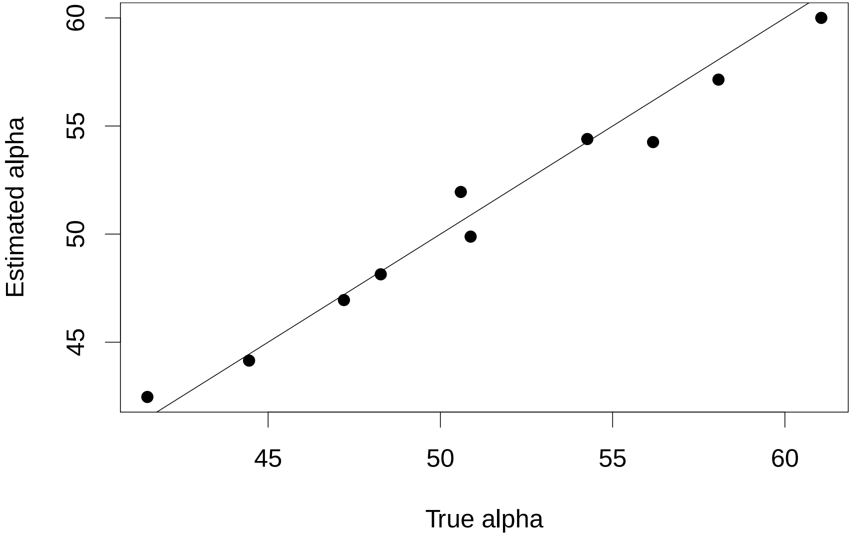
```
# Specify model in BUGS language
cat(file = "anova.txt", "
    model {

    # Priors
    for (i in 1:10){            # Implicitly define alpha as a vector
    alpha[i] ~ dnorm(0, 0.001) # Beware that a mean at 0 only works because
    }
    sigma ~ dunif(0, 100)

    # Likelihood
    for (i in 1:120) {
    y[i] ~ dnorm(mean[i], tau)
    mean[i] <- alpha[x[i]]
    }

    # Derived quantities
    tau <- 1 / ( sigma * sigma)
    }
```

Estimated effects vs theoretical effects



Classical random effect modelling I

```
### Restricted maximum likelihood (REML) analysis using R
library('lme4')                 # Load lme4

pop <- as.factor(x)             # Define x as a factor and call it pop

lme.fit <- lmer(y ~ 1 + 1 | pop, REML = TRUE)
lme.fit                         # Inspect results
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ 1 + 1 | pop
## REML criterion at convergence: 614.727
## Random effects:
##  Groups   Name        Std.Dev.
##  pop      (Intercept) 5.603
##  Residual             2.702
## Number of obs: 120, groups:  pop, 10
## Fixed Effects:
## (Intercept)
```

Classical random effect modelling II

```
##       50.98
```

```
ranef(lme.fit)                    # Print random effects
```

```
## $pop
##    (Intercept)
## 1    3.3707132
## 2    6.1042208
## 3   -8.3558761
## 4   -3.9048059
## 5   -1.0347003
## 6    8.8867973
## 7   -6.6287828
## 8   -2.7391127
## 9    3.3112092
## 10   0.9903372
##
## with conditional variances for "pop"
```

Classical random effect model - maths

$$Y_i \sim \mathcal{N}(\mu_i, \sigma^2), \mu_i = \alpha_{k[i]}$$

where $k[i]$ returns in which group is i. What's missing?

Classical random effect model - maths

$$Y_i \sim \mathcal{N}(\mu_i, \sigma^2), \mu_i = \alpha_{k[i]}$$

where $k[i]$ returns in which group is i. What's missing?

i.i.d. observations. And then?

Classical random effect model - maths

$$Y_i \sim \mathcal{N}(\mu_i, \sigma^2), \mu_i = \alpha_{k[i]}$$

where $k[i]$ returns in which group is i. What's missing?

i.i.d. observations. And then?

We estimate the variance of the random effects

$$\alpha_j \sim \mathcal{N}(\mu_\alpha, \sigma_\alpha^2)$$

for $j = 1, ..., J$ (we have to specify a mean μ_α too, we can set it to zero if there is an overall mean μ though)

Random effect model in a Bayesian framework I

```r
# Bundle and summarize the data set passed to JAGS
str(bdata <- list(y = y, x = x, npop = npop, n = n))
```

```
## List of 4
##  $ y   : num [1:120] 53.2 58.1 56.8 53 53.5 ...
##  $ x   : int [1:120] 1 1 1 1 1 1 1 1 1 1 ...
##  $ npop: num 10
##  $ n   : num 120
```

Random effect model in a Bayesian framework II

```
# Specify model in BUGS language
cat(file = "re.anova.txt", "
model {

# Priors and some derived things
for (i in 1:npop){
    alpha[i] ~ dnorm(mu, tau.alpha)     # Prior for population means
    effect[i] <- alpha[i] - mu  # Population effects as derived quan
 }
 mu ~ dnorm(0,0.001)            # Hyperprior for grand mean svl
 sigma.alpha ~ dunif(0, 10)     # Hyperprior for sd of population ef
 sigma.res ~ dunif(0, 10)       # Prior for residual sd

# Likelihood
 for (i in 1:n) {
    y[i] ~ dnorm(mean[i], tau.res)
    mean[i] <- alpha[x[i]]
 }

# Derived quantities
```

Fitting the model I

```
# Inits function
inits <- function(){ list(mu = runif(1, 0, 100), sigma.alpha = rlnor

# Params to estimate
params <- c("mu", "alpha", "effect", "sigma.alpha", "sigma.res")

# MCMC settings
nb <- 1000  ;  nc <- 3  ;  ni <- 2000  ;  nt <- 2

# Call JAGS, check convergence and summarize posteriors
out2 <- jags(bdata, inits, params, "re.anova.txt", n.thin = nt, n.cl
            n.burnin = nb, n.iter = ni)
```

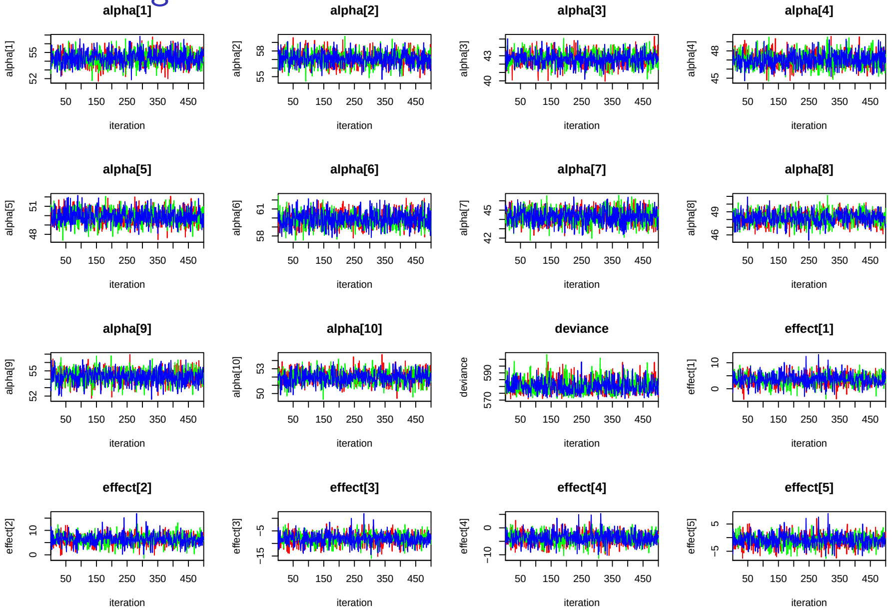
Fitting the model II

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 120
##    Unobserved stochastic nodes: 13
##    Total graph size: 273
##
## Initializing model
```

Model diagnostics I

```
traceplot(out2,mfrow=c(4,4))
```

Model diagnostics II



Model results I

```
print(out2,dig=3)
```

```
## Inference for Bugs model at "re.anova.txt", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded), n.thin = 2
##  n.sims = 1500 iterations saved
##            mu.vect sd.vect    2.5%    25%    50%    75%  97.5% Rhat n.eff
## alpha[1]    54.347   0.788  52.893 53.802 54.345 54.875 55.975 1.002  1300
## alpha[2]    57.122   0.775  55.620 56.602 57.124 57.623 58.730 1.000  1500
## alpha[3]    42.663   0.788  41.057 42.181 42.669 43.163 44.244 1.004   740
## alpha[4]    47.048   0.794  45.583 46.515 47.042 47.561 48.657 1.001  1500
## alpha[5]    49.906   0.774  48.404 49.359 49.907 50.447 51.407 1.000  1500
## alpha[6]    59.882   0.801  58.292 59.331 59.873 60.408 61.496 1.001  1500
## alpha[7]    44.332   0.790  42.773 43.795 44.335 44.863 45.837 1.000  1500
## alpha[8]    48.215   0.799  46.628 47.702 48.198 48.752 49.749 1.000  1500
## alpha[9]    54.319   0.799  52.674 53.815 54.337 54.836 55.815 1.004   570
## alpha[10]   51.983   0.764  50.477 51.464 51.983 52.499 53.467 1.003   760
## effect[1]    3.533   2.147  -0.812  2.167  3.488  4.874  7.823 1.004   670
## effect[2]    6.308   2.153   2.003  4.889  6.257  7.655 10.444 1.002   990
## effect[3]   -8.150   2.162 -12.740 -9.551 -8.138 -6.753 -3.957 1.004   790
## effect[4]   -3.765   2.173  -8.093 -5.177 -3.774 -2.370  0.475 1.002   910
## effect[5]   -0.907   2.169  -5.424 -2.307 -0.843  0.539  3.265 1.003   850
## effect[6]    9.068   2.159   4.679  7.666  9.022 10.412 13.311 1.002   820
## effect[7]   -6.481   2.166 -10.876 -7.841 -6.524 -5.021 -2.270 1.003   890
```

Model results II

```
## effect[8]    -2.599  2.194  -7.109  -3.998  -2.563  -1.221   1.806 1.003   710
## effect[9]     3.506  2.147  -1.055   2.157   3.490   4.871   7.682 1.002  1500
## effect[10]    1.169  2.168  -3.126  -0.204   1.190   2.600   5.414 1.002  1500
## mu           50.813  2.047  46.759  49.500  50.872  52.089  55.106 1.004   720
## sigma.alpha   6.232  1.415   3.910   5.154   6.074   7.165   9.273 1.002  1100
## sigma.res     2.740  0.183   2.409   2.620   2.726   2.857   3.129 1.001  1500
## deviance    580.326  5.033 572.558 576.680 579.668 583.337 591.732 1.001  1500
##
## 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 = 12.7 and DIC = 593.0
## DIC is an estimate of expected predictive error (lower deviance is better).
```

Comparison of variance estimates

```r
### Well, comparison of sigma's...
VarCorr(lme.fit)
```

```
##  Groups   Name        Std.Dev.
##  pop      (Intercept) 5.6025
##  Residual             2.7020
```

```r
out2$BUGSoutput$mean$sigma.res #true value is 3
```

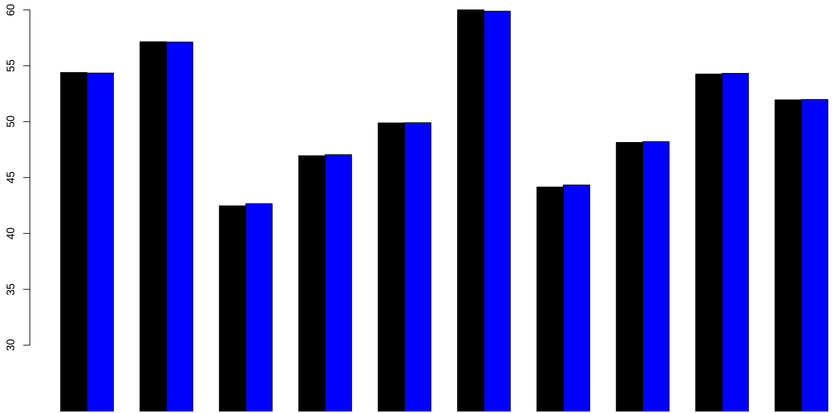
```
## [1] 2.74021
```

```r
out2$BUGSoutput$mean$sigma.alpha #true value is 5
```

```
## [1] 6.231839
```

Comparison of fixed and random effects

```
## Plotting shrinkage
alpha_mean2 = out2$BUGSoutput$mean$alpha
barplot(t(matrix(c(alpha_mean,alpha_mean2),ncol=2,nrow=10)),beside=T
```



Re-running the analysis with more shrinkage I

Now we assume a prior $\sigma_\alpha \sim \text{Gamma}(100, 50)$.

More details on the Gamma distribution

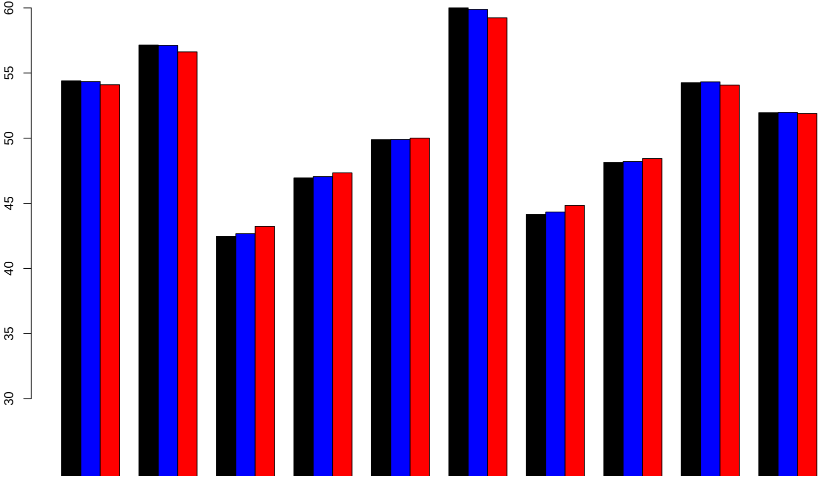
- $X \sim \text{Gamma}(a, b)$ with $a =$ shape, $b =$ rate $= \frac{1}{\theta}$ where θ is scale.
- Properties: $\mathbb{E}(X) = a\theta = 100/50 = 2$ and $\mathbb{V}(X) = a\theta^2 = \frac{100}{2500} = 0.04$ so that $\text{SD}(X) = 0.2$.

```
alpha_mean3 = out3$BUGSoutput$mean$alpha
out3$BUGSoutput$mean$sigma.alpha
```

```
## [1] 2.565298
```

```
barplot(t(matrix(c(alpha_mean,alpha_mean2,alpha_mean3),ncol=3,
```

Re-running the analysis with more shrinkage II



To wrap up – ANOVA vs REs in a Bayesian setup

- The difference between ANOVA and random effects models boils down to estimation of variance between group-level effects

To wrap up – ANOVA vs REs in a Bayesian setup

- The difference between ANOVA and random effects models boils down to estimation of variance between group-level effects
- (usually confusing because ANOVA *does* some variance partitioning)

To wrap up – ANOVA vs REs in a Bayesian setup

- The difference between ANOVA and random effects models boils down to estimation of variance between group-level effects
- (usually confusing because ANOVA *does* some variance partitioning)
- ANOVA e.g. $\alpha_j \sim \mathcal{N}(50, 100)$

To wrap up – ANOVA vs REs in a Bayesian setup

- The difference between ANOVA and random effects models boils down to estimation of variance between group-level effects
- (usually confusing because ANOVA *does* some variance partitioning)
- ANOVA e.g. $\alpha_j \sim \mathcal{N}(50, 100)$
- RE model $\alpha_j \sim \mathcal{N}(\mu_\alpha, \sigma_\alpha^2)$ and $\sigma_\alpha \sim \mathcal{U}(0, 10)$ or $\sigma_\alpha \sim \text{Exp}(0, 1/5)$

To wrap up – ANOVA vs REs in a Bayesian setup

- The difference between ANOVA and random effects models boils down to estimation of variance between group-level effects
- (usually confusing because ANOVA *does* some variance partitioning)
- ANOVA e.g. $\alpha_j \sim \mathcal{N}(50, 100)$
- RE model $\alpha_j \sim \mathcal{N}(\mu_\alpha, \sigma_\alpha^2)$ and $\sigma_\alpha \sim \mathcal{U}(0, 10)$ or $\sigma_\alpha \sim \text{Exp}(0, 1/5)$
- RE model with more shrinkage $\sigma_\alpha \sim \text{Gamma}(100, 50)$.

More material

- Shrinkage aka partial pooling is a property of mixed models, not Bayesian estimation (though you can top it up using informative priors)
- Kruschke's post on parameterizing the Gamma distribution

Bonus: fun and pretty snakes

Figure 1: *Vipera ursinii* Benny Trapp (CC BY)

Super épisode de La méthode scientifique sur France Culture, 08/11/2021