6. sklop: Primer hierarhicnega modela z Gibbsovim vzorcevalnikom

Gradivo tega sklopa temelji na knjigi P. D. Hoff, *A First Course in Bayesian Statistical Methods*, 2009, in sicer na 8. poglavju *Group comparisons and hierarchical modeling*, str. 125.

1 Podatki

V raziskavi *Educational Longitudinal Study (ELS)* iz leta 2002 so preucevali rezultate testov ucencev ameriskih srednjih sol. Podane imamo rezultate matematicnih testov ucencev 10. razreda iz 100 javnih srednjih sol (velike sole z vsaj 400 ucenci 10. razreda, urbano okolje).

Za vsakega ucenca imamo podano solo in rezultat matematicnega testa – nasi podatki so torej vecnivojski/hierarhicni.

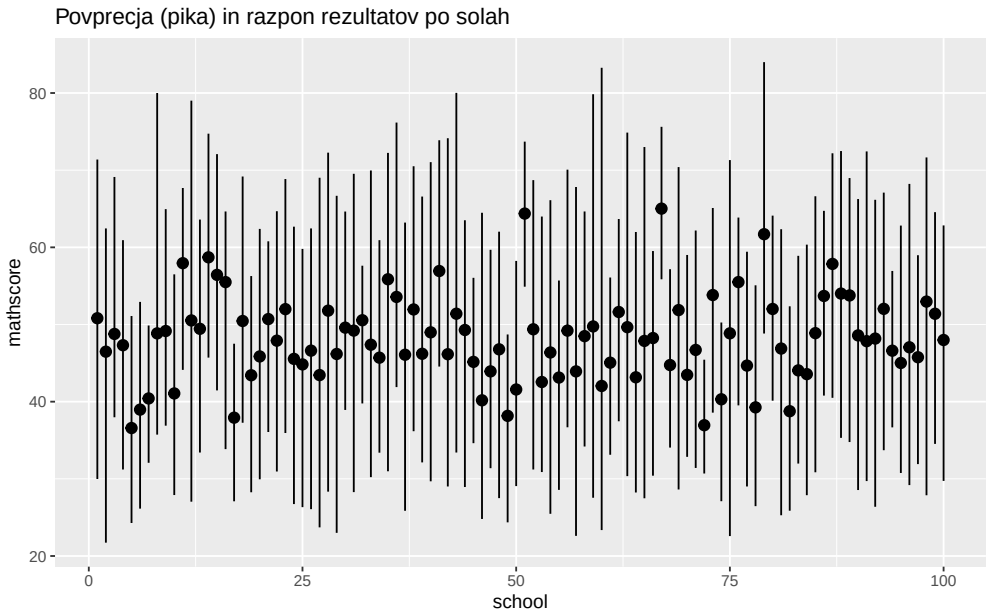
Rezultati matematicnega testa so del nacionalnega preverjanja znanja, ki naj bi bil konstruiran tako, da je pricakovana vrednost enaka 50 in standardni odklon 10.

Oglejmo si podatke.

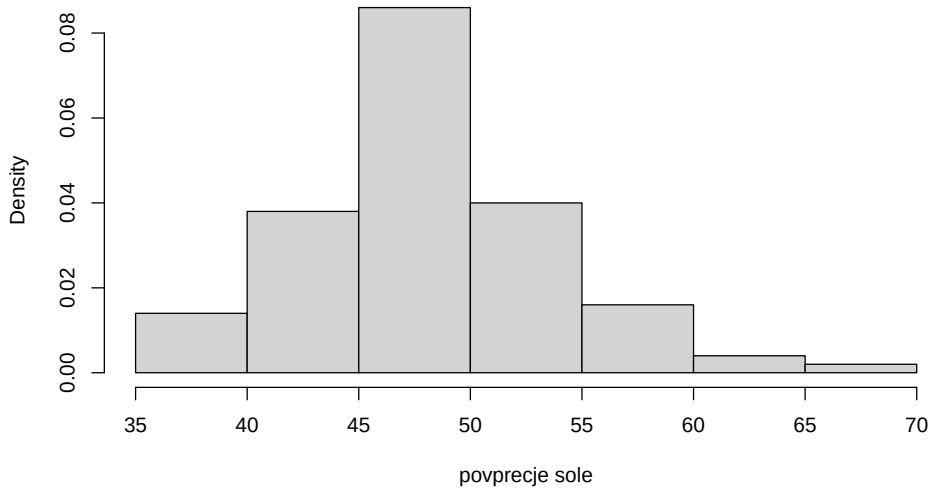
```
source("podatki_sole.R")
str(pod)
```

```
## 'data.frame':    1993 obs. of  2 variables:
##  $ school   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ mathscore: num  52.1 57.6 66.4 44.7 40.6 ...
```

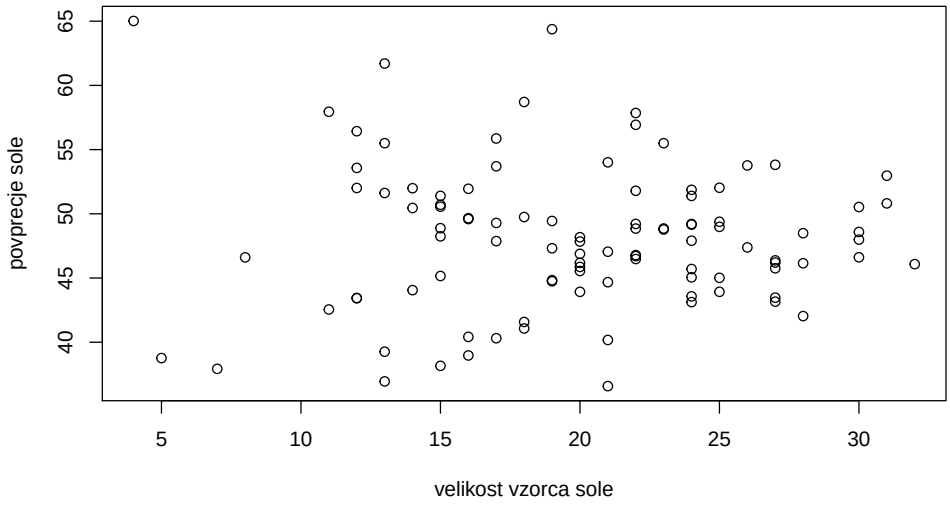
```
library(ggplot2)
ggplot(pod, aes(x = school, y = mathscore, group = school)) +
  stat_summary(fun.ymin = min, fun.ymax = max, fun.y = mean) +
  labs(title = "Povprecja (pika) in razpon rezultatov po solah")
```




```
library(dplyr)
# Preureditev podatkov:
pod.sole = pod %>%
  group_by(school) %>%
  summarise(povprecje = mean(mathscore),
            n=length(mathscore),
            varianca = var(mathscore))

# Ocenjena gostota:
hist(pod.sole$povprecje, prob=T,
     xlab = "povprecje sole", main = "")
```



```r
# Povprecna ocena glede na velikost sole:
plot(pod.sole$n, pod.sole$povprecje,
     xlab = "velikost vzorca sole", ylab = "povprecje sole")
```



2 Uporaba modela na podatkih

Dolociti moramo parametre (hiper)apriornih porazdelitev $\mu_0, \tau_0^2, \sigma_0^2, \nu_0, \eta_0^2, \kappa_0$:

$$\begin{aligned}
\sigma^2 &\sim \text{Inv-Gama}(\nu_0/2, \sigma_0^2\nu_0/2),\\
\mu &\sim \mathcal{N}(\mu_0, \tau_0^2),\\
\eta^2 &\sim \text{Inv-Gama}(\kappa_0/2, \eta_0^2\kappa_0/2),\\
\mu_j &\sim N(\mu, \eta^2).
\end{aligned}$$

Spomnimo se, da je matematicni test konstruiran tako, da je pricakovana vrednost enaka 50 in standardni odklon 10.

Izberemo si:
$\nu_0 = 1$ (sibko informativna),
$\sigma_0^2 = 100$ (ker naj bi bil standardni odklon 10),
$\mu_0 = 50$ (ker naj bi bil standardni odklon 10),
$\tau_0^2 = 25$ (ker zelimo sibko informativno in s tem dopuscamo s 95% μ med 40 in 60),
$\kappa_0 = 1$ (sibko informativna),
$\eta_0^2 = 100$.

2.1 Gibbsov vzorcevalnik za nas primer

```
### Parametri (hiper)apriornih porazdelitev
sigma20 = 100
nu0 = 1
eta20 = 100
kappa0 = 1
mu0 = 50
tau20 = 25

### Pripravimo si kolicine, ki jih bomo potrebovali iz podatkov
x = pod
m = length(pod.sole$school)
n = pod.sole$n
x.povpr = pod.sole$povprecje

### Dolocimo si zacetne vrednosti
muGroups = x.povpr
sigma2 = mean(pod.sole$varianca)
mu = mean(muGroups)
eta2 = var(muGroups)

### Pripravimo si prostor za shranjevanje
n.iter = 5000

muGroups.all = matrix(nrow = n.iter, ncol = m)
sigma2.all = rep(NA, n.iter)
mu.all = rep(NA, n.iter)
eta2.all = rep(NA, n.iter)
```

```
### Na prvo mesto si shranimo zacetne vrednosti (nepotrebno)
muGroups.all[1, ] = muGroups
sigma2.all[1] = sigma2
mu.all[1] = mu
eta2.all[1] = eta2

### Pozenemo Gibbsov vzorcevalnik
set.seed(1)
for (s in 1:n.iter) {
  ### Vzorcimo muGroups
  for (j in 1:m) {
    muGroups[j] = rnorm(1,
                        mean = (x.povpr[j] * n[j] / sigma2 + mu / eta2) /
                          (n[j] / sigma2 + 1 / eta2),
                        sd = sqrt(1 / (n[j] / sigma2 + 1 / eta2)))
  }

  ### Vzorcimo sigma2
  ss = nu0 * sigma20
  for (j in 1:m) {
    ss = ss + sum((x[x[, 1] == j, 2] - muGroups[j])^2)
  }
  sigma2 = 1 / rgamma(1, (nu0 + sum(n)) / 2, ss / 2)

  ### Vzorcimo mu
  mu = rnorm(1,
             mean = (mean(muGroups) * m / eta2 + mu0 / tau20) /
               (m / eta2 + 1 /tau20),
             sd = sqrt(1 / (m / eta2 + 1 /tau20)))

  ### Vzorcimo eta2
  ss = kappa0 * eta20 + sum((muGroups - mu)^2)
  eta2 = 1 / rgamma(1, (kappa0 + m) / 2, ss / 2)

  ### Shranimo nove parametre
  muGroups.all[s, ] = muGroups
  sigma2.all[s] = sigma2
  mu.all[s] = mu
  eta2.all[s] = eta2
}
```

2.2 Preucevanje konvergence

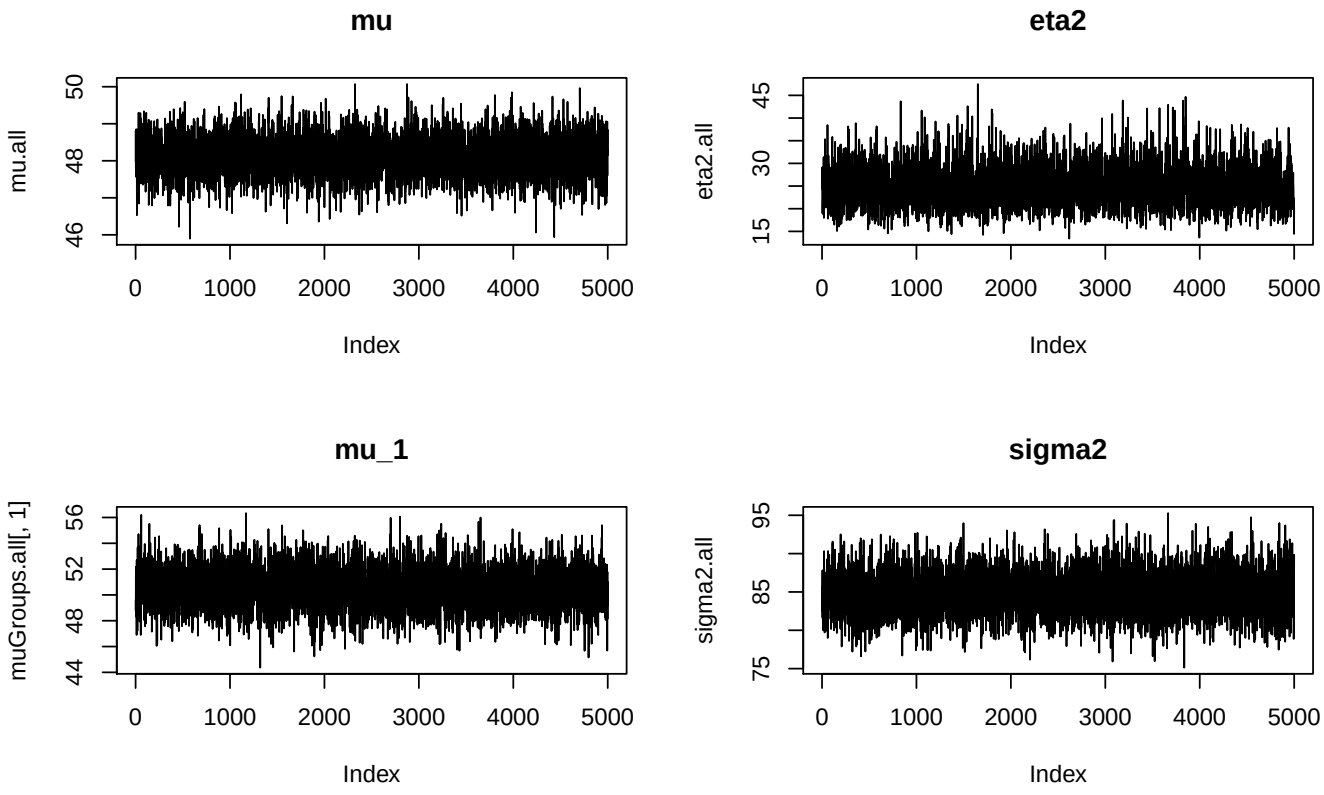
V podrazdelkih so navedeni nacini za preucevanje konvergence.

2.2.1 *Trace plots*

Na podlagi spodnjih slik verig dolocimo potreben *burn-in* in graficno presodimo, ali je konvergenca dosezena (tako kot v 4. sklopu).

Vse iteracije za $\mu_1, \mu, \eta^2, \sigma^2$ (izgleda OK):

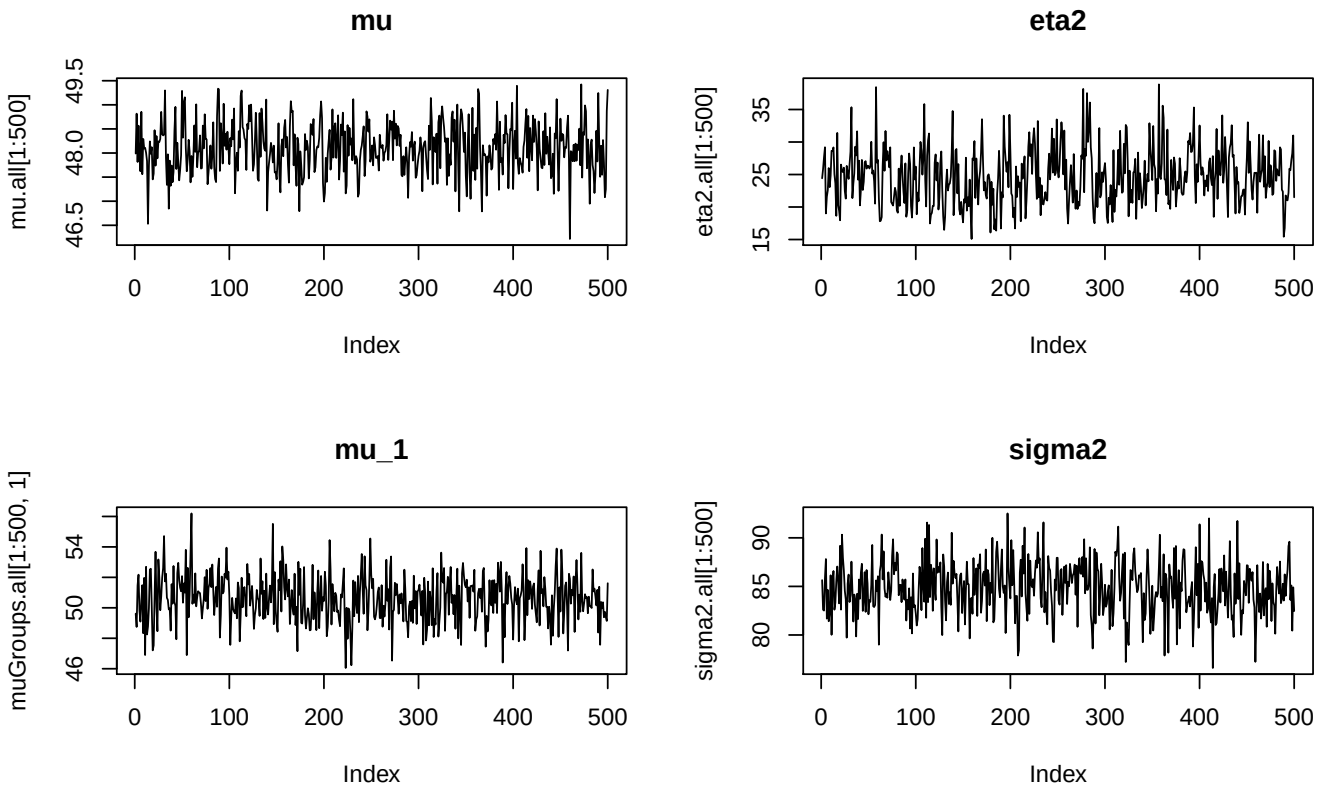
```
par(mfrow=c(2,2))
plot(mu.all, type="l", main="mu")
plot(eta2.all, type="l", main="eta2")
plot(muGroups.all[,1], type="l", main="mu_1")
plot(sigma2.all, type="l", main="sigma2")
```



(lahko bi tudi pogledali $\mu_2, ..., \mu_m$)

Prvih 500 iteracij (izgleda v redu):

```
par(mfrow=c(2,2))
plot(mu.all[1:500], type="l", main="mu")
plot(eta2.all[1:500], type="l", main="eta2")
plot(muGroups.all[1:500,1], type="l", main="mu_1")
plot(sigma2.all[1:500], type="l", main="sigma2")
```



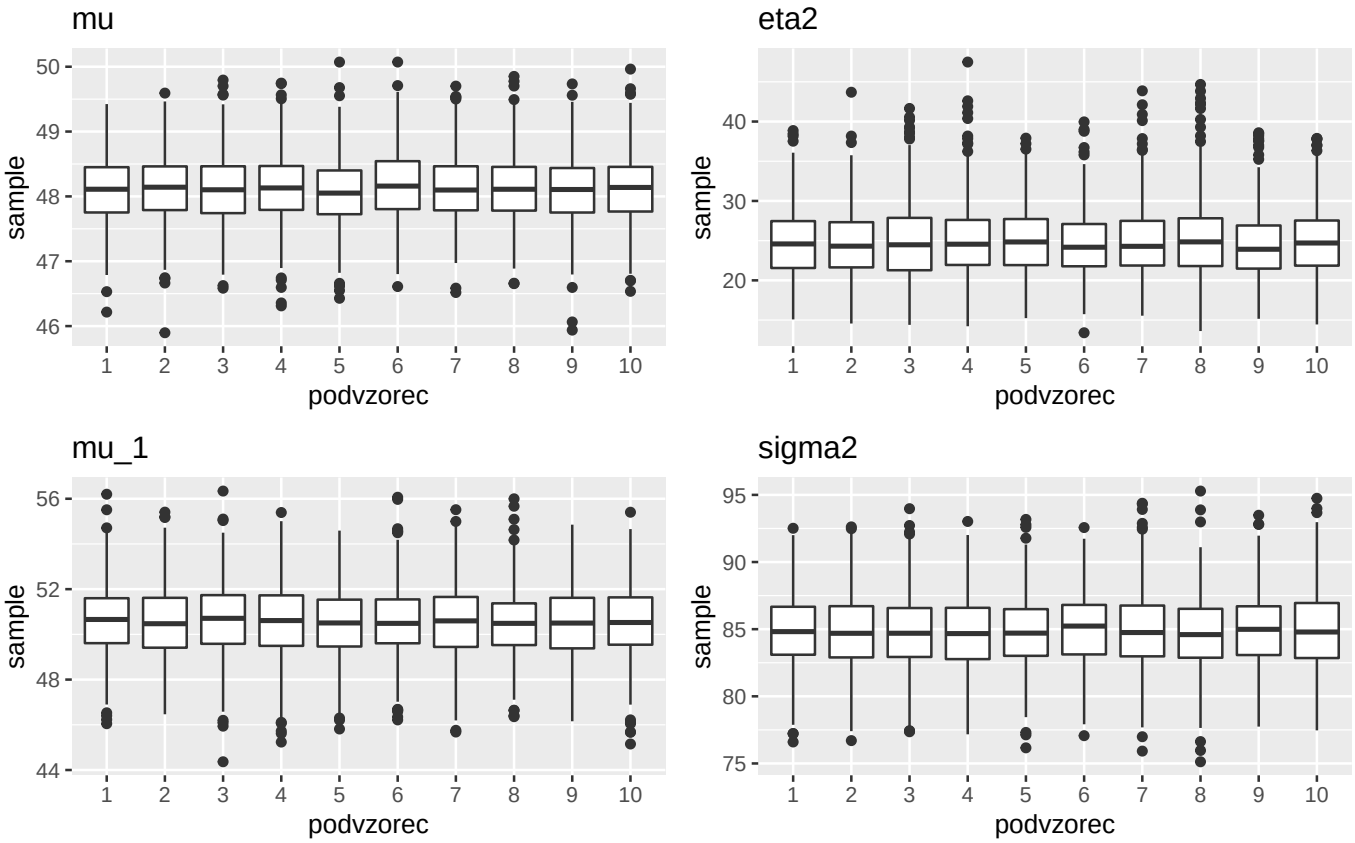
2.2.2 Vec verig in *mixing*

Preizkusite razlicne zacetne vrednosti (kljucno je, da vkljucite tudi manj smiselne) in pogledate, ali dobite po nekem zacetnem *burn-in* podobno verigo. Verige lahko narisete na eno sliko in pogledate, ali je dober *mixing*. V koncni vzorec potem lahko zdruzite vse verige.

2.2.3 Porazdelitve podvzorcev

Pogledamo, ali se približno ujemajo porazdelitve za prvih 10 šol, tj. ali je veriga "stabilna" (izgleda OK).

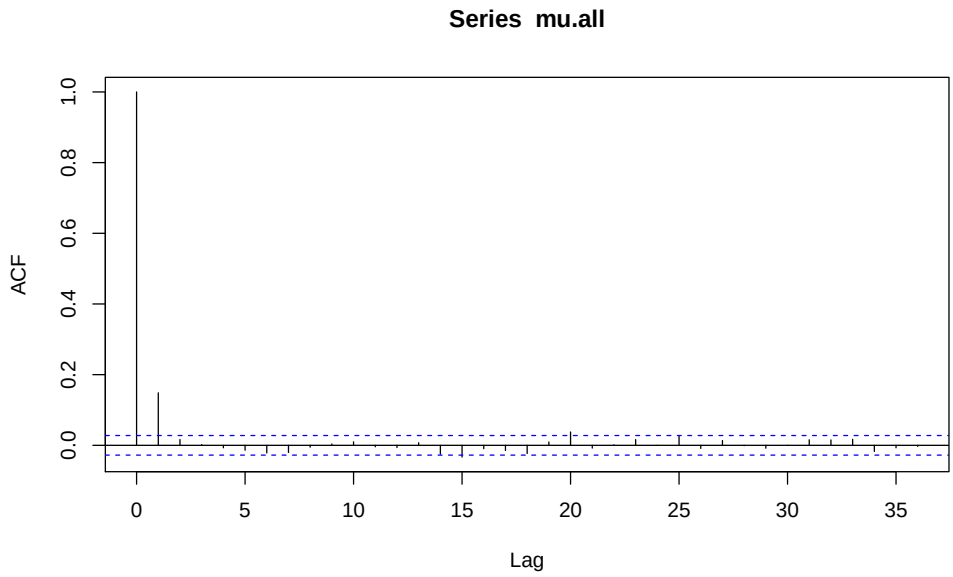
```
library(gridExtra)
mu.all2 = data.frame(sample = mu.all, podvzorec = factor(sort(rep(1:10,500))))
p1 = ggplot(mu.all2, aes(x = podvzorec, y = sample)) +
  geom_boxplot() + labs(title = "mu")

eta2.all2 = data.frame(sample = eta2.all, podvzorec = factor(sort(rep(1:10,500))))
p2 = ggplot(eta2.all2, aes(x = podvzorec, y = sample)) +
  geom_boxplot() + labs(title = "eta2")

mu1 = data.frame(sample = muGroups.all[,1], podvzorec = factor(sort(rep(1:10,500))))
p3 = ggplot(mu1, aes(x = podvzorec, y = sample)) +
  geom_boxplot() + labs(title = "mu_1")

sigma2.all2 = data.frame(sample = sigma2.all, podvzorec = factor(sort(rep(1:10,500))))
p4 = ggplot(sigma2.all2, aes(x = podvzorec, y = sample)) +
  geom_boxplot() + labs(title = "sigma2")
grid.arrange(p1, p2, p3, p4, ncol = 2)
```



2.2.4 Avtokorelacije in *effective sample size* VS *thinning*

Pri zamiku (*lag*) 1 dobimo po definiciji avtokorelacijo 1, za kasnejse pa si zelimo, da so cim blizje 0 (kar pricakujemo pri n.e.p. vzorcu). Pri vzorcu dobljenim z MCMC metodami bo prisotna avtokorelacija, ki pa se z zamikom (*lag*) zmanjsuje (odvisnost neke vrednosti od vrednosti iz enega koraka nazaj je najvecja, iz dveh korakov nazaj malo manjsa, itd.).

```
acf(mu.all) #dobimo graf, cel vzorec
```




```
ac.mu = acf(mu.all, plot = FALSE)
ac.mu #dobimo izpis
```

```
##
## Autocorrelations of series 'mu.all', by lag
##
##      0      1      2      3      4      5      6      7      8      9     10
##  1.000  0.149  0.016  0.002 -0.007 -0.014 -0.022 -0.020 -0.005  0.004  0.010
##     11     12     13     14     15     16     17     18     19     20     21
## -0.004 -0.006  0.007 -0.025 -0.033 -0.010 -0.015 -0.023  0.009  0.038 -0.008
##     22     23     24     25     26     27     28     29     30     31     32
##  0.001  0.016  0.000  0.026 -0.009  0.014 -0.001 -0.008 -0.001  0.016  0.015
##     33     34     35     36
##  0.017 -0.017 -0.007 -0.003
```

```
head(ac.mu$acf) #dobimo natancnejsi izpis
```

```
## , , 1
##
##               [,1]
## [1,]  1.000000000
```

```
## [2,]  0.148547488
## [3,]  0.016374954
## [4,]  0.001949737
## [5,] -0.007071617
## [6,] -0.013949257
```

Kako lahko priblizno izracunamo avtokorelacijo z zamikom 1?

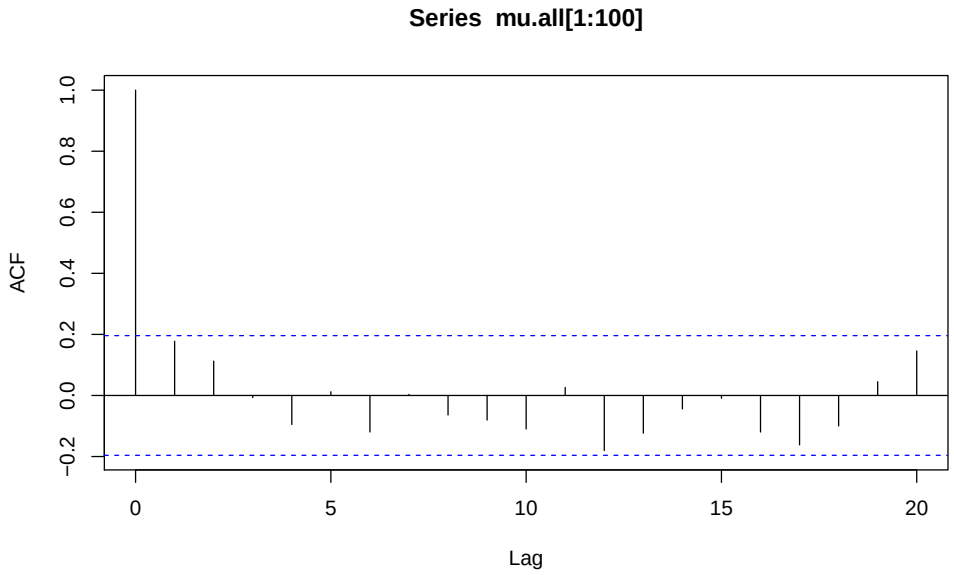
```
cor(mu.all[-length(mu.all)], mu.all[-1])
```

```
## [1] 0.1485619
```

```
ac.mu$acf[2]
```

```
## [1] 0.1485475
```

```
acf(mu.all[1:100]) #za prvih 100 iteracij
```

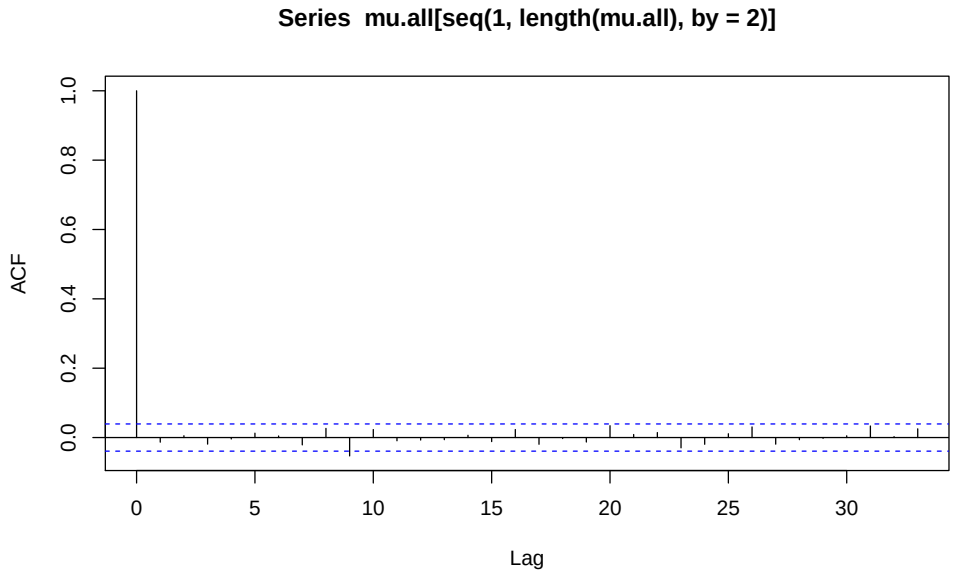


Kaj predstavljata crti?

$\pm 1.96/\sqrt{N}$, N stevilo iteracij – avtokorelacije izven obmocja so statisticno znacilno razlicne od 0 (oz. le 5% jih lahko pricakujemo *nekoliko* izven pri n.e.p. vzorcu).

Kaj se v nasem primeru zgodi z avtokorelacijami, ce v zaporedju izbrisemo vsakega drugega (uporabimo *thinning*)?

```
acf(mu.all[seq(1, length(mu.all), by=2)])
```



Pricakovano se zmanjsajo, vendar smo pri tem zmanjsali tudi velikost vzorca, za katerega smo ze porabili cas za izracun.

Effective sample size lahko interpretiramo kakor stevilo slucajnih vzorcev (n.e.p.), ki nam dajo enako natancno informacijo kakor nas dobljeni MCMC vzorec.

```
library(coda)
effectiveSize(mu.all)
```

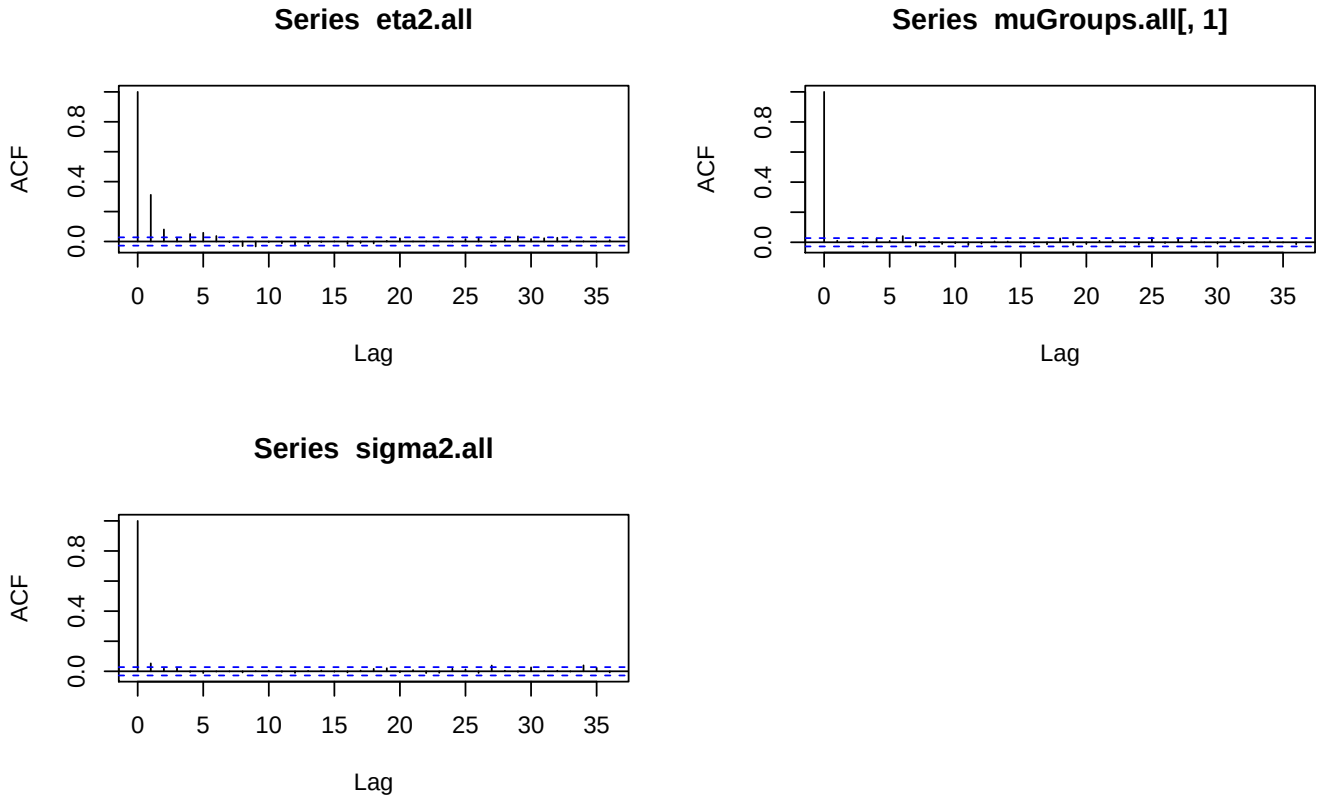
```
##     var1
## 3705.908
```

Torej:

- Celoten vzorec za μ dolzine 5000 je avtokoreliran – ne smemo ga imeti za slucajni vzorec (n.e.p.) iz aposteriorne porazdelitve.
- Ce v verigo vzamemo vsakega drugega (*thinning* = 2, kar je najmanjse mozno), potem vzorec ni avtokoreliran, njegova velikost pa je 2500.
- *Effective sample size*, ki pove za koliko n.e.p. realizacij je nas vzorec "vreden", je enak 3705, kar je mnogo vec od 2500. Uporabimo zato raje celoten vzorec in upostevamo *effective sample size* kot njegovo "pravo velikost".

Avtokorelacije preostalih (nekaj) parametrov in njihovi *effective sample size*:

```
par(mfrow=c(2,2))
acf(eta2.all)
acf(muGroups.all[,1])
acf(sigma2.all)
```




```
effectiveSize(eta2.all)
```

```
##     var1
## 2503.468
```

```
effectiveSize(muGroups.all[,1])
```

```
##     var1
## 4421.197
```

```
effectiveSize(sigma2.all)
```

```
##     var1
## 4498.545
```

Preko *effective sample size* izracunamo **"standardne napake" nasih ocen**, kjer je ocena povprecje marginalne aposteriorne porazdelitve:

```
sd(mu.all) / sqrt( effectiveSize(mu.all) )
```

```
##        var1
## 0.008795844
```

```
sd(eta2.all) / sqrt( effectiveSize(eta2.all) )
```

```
##       var1
## 0.08842911
```

```
sd(muGroups.all[,1]) / sqrt( effectiveSize(muGroups.all[,1]) )
```

```
##       var1
## 0.02359033
```

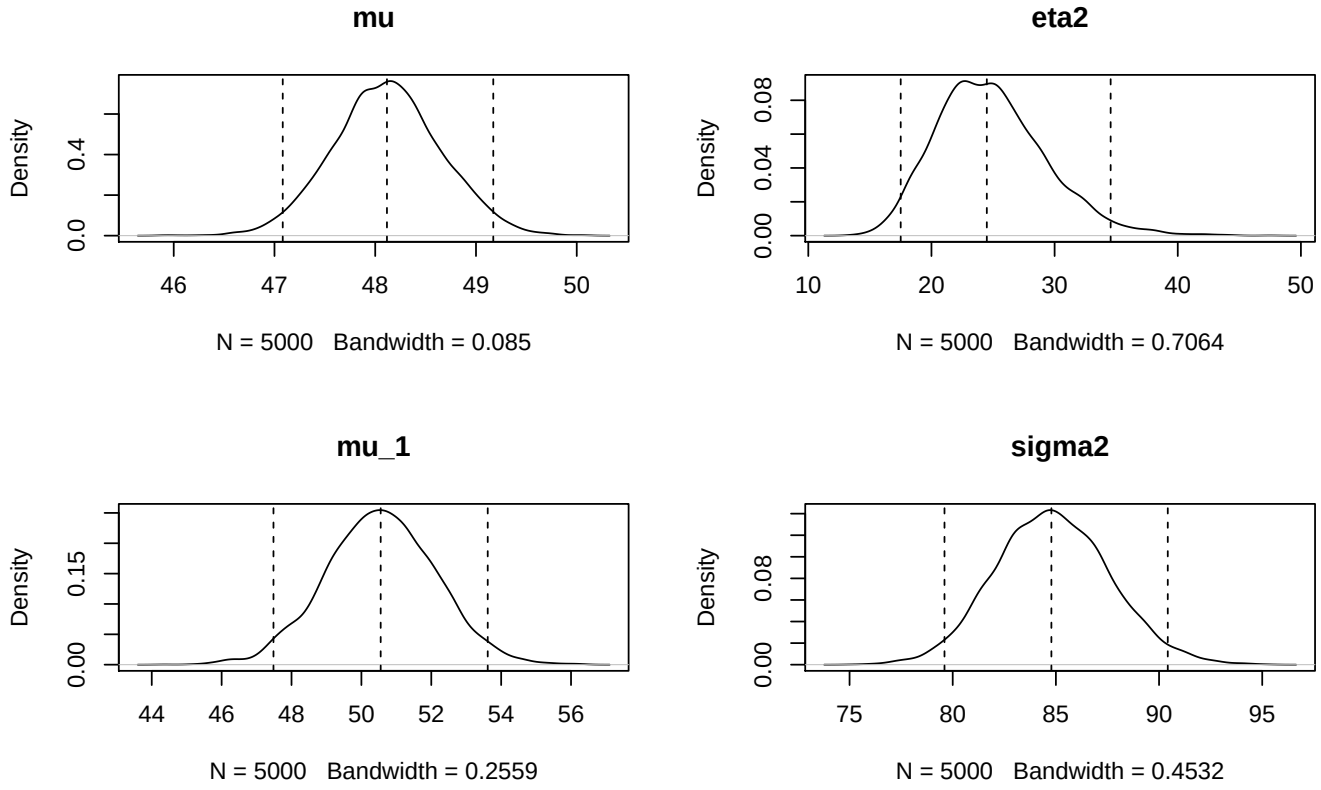
```
sd(sigma2.all) / sqrt( effectiveSize(sigma2.all) )
```

```
##       var1
## 0.04123463
```

Ali so velike ali majhne? Odvisno glede na skalo parametrov oz. enote spremenljivk.

2.3 Robne aposteriorne porazdelitve

```
par(mfrow=c(2,2))
plot(density(mu.all), type="l", main="mu")
abline(v = quantile(mu.all, prob=c(0.025, 0.5, 0.975)), lty = 2)
plot(density(eta2.all), type="l", main="eta2")
abline(v = quantile(eta2.all, prob=c(0.025, 0.5, 0.975)), lty = 2)
plot(density(muGroups.all[,1]), type="l", main="mu_1")
abline(v = quantile(muGroups.all[,1], prob=c(0.025, 0.5, 0.975)), lty = 2)
plot(density(sigma2.all), type="l", main="sigma2")
abline(v = quantile(sigma2.all, prob=c(0.025, 0.5, 0.975)), lty = 2)
```

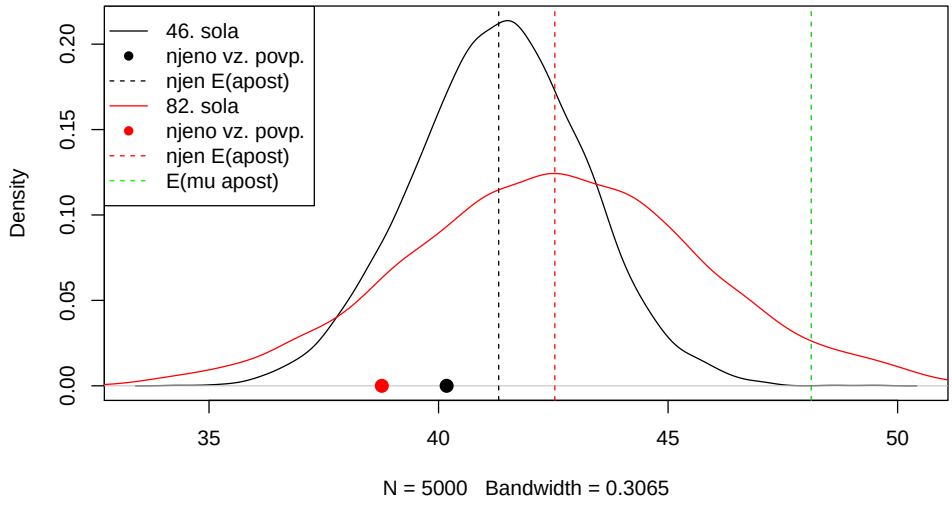


Interpretirati rezultate (srednja vrednost in razprsenost porazdelitve sta pomembni) tako za hiperparametre, kakor tudi za posamezne sole (katera ima najboljse, katera najslabse rezultate, ali se parametri "statisticno znacilno" razlikujejo, ipd.).

2.3.1 *Shrinkage*

Oglejmo si aposteriorni porazdelitvi dveh sol:

```
plot(density(muGroups.all[,46]), type="l", main="")
points(pod.sole[46,]$povprecje, 0, pch=16, cex=1.5)
abline(v = mean(muGroups.all[,46]), lty=2)
lines(density(muGroups.all[,82]), type="l", col="red")
points(pod.sole[82,]$povprecje, 0, pch=16, cex=1.5, col="red")
abline(v = mean(muGroups.all[,82]), lty=2, col="red")
abline(v = mean(mu.all), lty=2, col="green3")
legend("topleft", c("46. sola", "njeno vz. povp.", "njen E(apost)",
                    "82. sola", "njeno vz. povp.", "njen E(apost)",
                    "E(mu apost)"),
       col=c("black","black","black","red","red","red","green"),
       lty=c(1,NA,2,1,NA,2,2),
       pch=c(NA,16,NA,NA,16,NA))
```



Navidez paradoksalno, saj je vzorcno povprecje 82. sole manjse od 46., medtem ko je pricakovana vrednosti aposteriorne porazdelitve velja ravno obratno.

Iz pogojne porazdelitve μ_j glede na vse ostalo vemo, da je:

$$\mathrm{E}(\mu_j \mid \text{vse ostalo}) = \frac{n_j/\sigma^2}{n_j/\sigma^2 + 1/\eta^2}\bar{x}_{\cdot j} + \frac{1/\eta^2}{n_j/\sigma^2 + 1/\eta^2}\mu$$

Pricakovana vrednost μ_j se torej pomakne proti skupnemu populacijskemu povprecju μ.

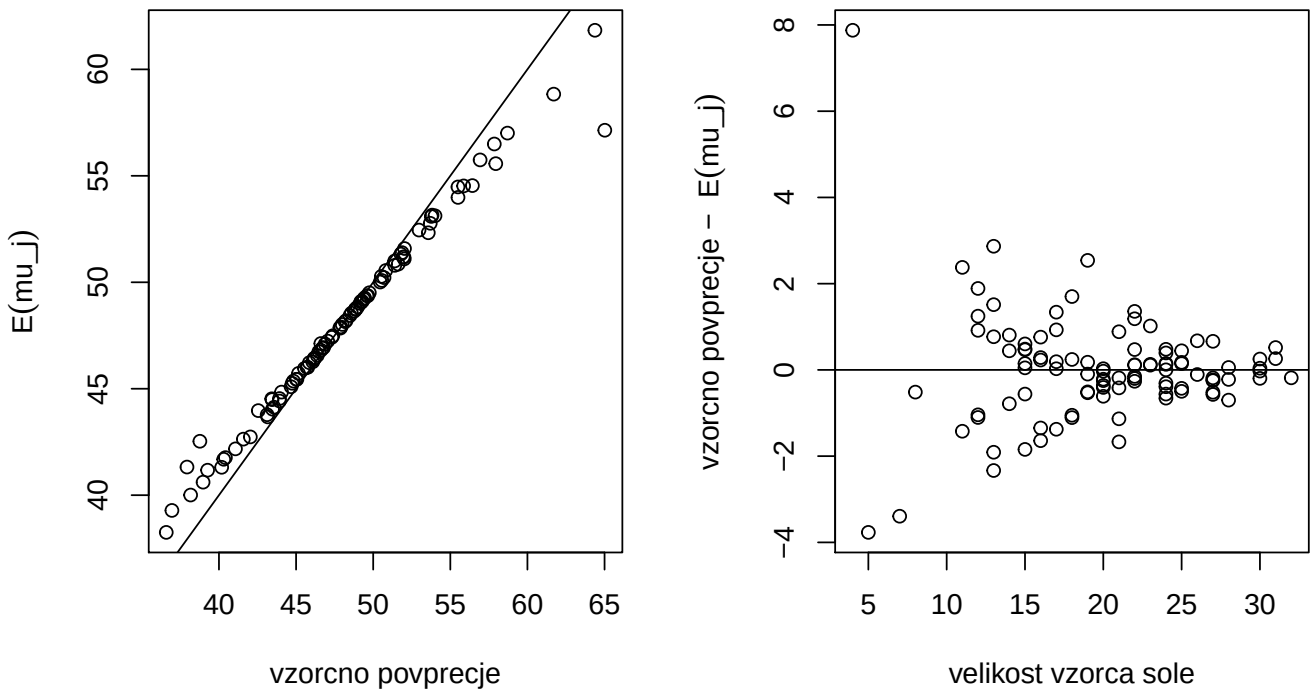
Ta efekt je pri velikem n_j majhen.

Poanta (zaradi katere zgornje ni paradoksalno): Pri soli z majhnim vzorcem smo lahko

bolj zgresili povprecje te sole in zato bolj verjamemo skupnemu povprecju μ kot njenemu vzorcnemu povprecju oz. se k μ premaknemo bolj kot pri veliki soli.

Ocene povprecij sol (pricakovane vrednosti aposteriornih porazdelitev μ_j) se torej skrcijo (*shrinkage*) proti skupnemu povprecju μ, v primerjavi z vzorcnimi povprecji sol $\bar{x}_{\cdot j}$, kjer je ta premik vecji pri manjsi velikosti vzorca sole n_j. Slednje je jasno prikazano spodaj.

```
pod.sole$EmuGroups = colMeans(muGroups.all)

par(mfrow=c(1,2))
plot(pod.sole$povprecje, pod.sole$EmuGroups,
     xlab = "vzorcno povprecje", ylab = expression(E(mu_j)))
abline(a = 0, b = 1)

plot(pod.sole$n, pod.sole$povprecje - pod.sole$EmuGroups,
     xlab = "velikost vzorca sole",
     ylab = expression(paste("vzorcno povprecje - "," ",E(mu_j), sep="")))
abline(h = 0)
```



Splosna opomba: preveriti odnos med aposterornimi in apriornimi porazdelitvami (hiper)parametrov

Mozne vrednosti aposteriorne morajo biti vsebovane v tistih od apriorne, ce:

- Zelimo imeti objektiven Bayesov pristop (tj. sibke apriorne, ki ne smejo informirati aposteriornih).
- Si slednje lahko privoscimo glede na stevilo enot oz. imamo dovolj podatkov, ki so zmozni informirati nase parametre (ce si ne moremo, je verjetno modeliranje parametra kljub odstopanjem aposteriorne od apriorne se zmeraj boljse kakor fiksiranje na neko vrednost, kar nekako ustreza apriorni z gostoto 1 v tej vrednosti).
- Ni namen modela drugacen, npr. skrciti koeficiente v regresiji proti nic.

3 O ideji hierarhicnih modelov na splosno...

... vendar ob razmisljanju na nasem primeru.

Zamenljivost (exchangeability)

Znotraj modela smo predpostavili, da so tako meritve znotraj sol $X_{1,j}, \ldots, X_{n_j,j}$ zamenljive (tj. slucajni vzorec, torej standardna predpostavka), kakor tudi populacijska povprecja sol $\mu_1, \ldots, \mu_m$. Ali je to smiselno?

Nismo pa predpostavili, da so vse meritve $X_{1,1}, \ldots, X_{n_1,1}, \ldots, X_{m,1}, \ldots, X_{n_m,m}$ med seboj zamenljive. Zakaj?

Kaj so prednosti/slabosti hierarhicnega modela v primerjavi z modelom z le enim skupnim povprecjem?

Primerjamo torej z modelom:

$$(X_{1,1}, \ldots, X_{n_1,1}, \ldots, X_{m,1}, \ldots, X_{n_m,m}) \mid \mu, \sigma^2 \sim \text{n.e.p. } N(\mu, \sigma^2),$$

tj. dvorazsezni normalni model, kjer podatke vseh sol zdruzimo v en vzorec.

Kaj so prednosti/slabosti hierarhicnega modela v primerjavi z modelom z vsemi razlicnimi nevezanimi povprecj?

Primerjamo torej z modelom:

$$(X_{1,j}, \ldots, X_{n_j,j}) \mid \mu_j, \sigma^2 \sim \text{n.e.p. } N(\mu_j, \sigma^2),$$

kjer nadalje velja

$$\begin{aligned} \pi(\mu_1, \ldots, \mu_m, \sigma^2) &= \pi(\mu_1, \ldots, \mu_m) \cdot \pi(\sigma^2) \\ &= \pi(\mu_1) \cdot \ldots \cdot \pi(\mu_m) \cdot \pi(\sigma^2). \end{aligned}$$