Seminarska naloga pri predmetu Bayesova statistika

Jan Črne

9 2 2022

Primerjava linearne in Bayesove regresije

Uvod

Za primerjanje je regresij sem si izbral model $Y = 12.2 + 6X_1 + 3.7X_2 + 0X_3 - 1.5X_4 + 0.1X_5 + napaka$, kjer so $X_1 \sim N(3, 0.5)$, $X_2 \sim Unif(0, 7)$, $X_3 \sim Exp(\frac{1}{3})$, $X_4 \sim Bin(10, 0.2)$, $X_5 \sim Pois(2.5)$. Napaka je v prvem primeru (v nadeljevanju povsod indeks 1) porazdeljena $N(0, 0.01)$, v drugem primeru (indeks 2) je $N(0, 10)$, v tretjem primeru, kjer je asimetrična (indeks 3) pa χ^2_6 (prostostnih stopenj je 6).

```
library(dplyr)
library(R2BayesX)
library(data.table)
library(coda)
```

```
n <- 100
k <- 1000

# # prednastavljanje prostora za realizacije
# matrika <- matrix(0, k, 6,dimnames=list(c(), c("Intercept", "x1", "x2", "x3", "x4", "x5")))
#
# klr1.koef <- data.frame(matrika)
# klr1.odklon <- data.frame(matrika)
# b1.povp <- data.frame(matrika)
# b1.odklon <- data.frame(matrika)
#
# klr2.koef <- data.frame(matrika)
# klr2.odklon <- data.frame(matrika)
# b2.povp <- data.frame(matrika)
# b2.odklon <- data.frame(matrika)
#
# klr3.koef <- data.frame(matrika)
# klr3.odklon <- data.frame(matrika)
# b3.povp <- data.frame(matrika)
# b3.odklon <- data.frame(matrika)
#
#
# for (i in 1:k) {
#   x1 <- rnorm(n, mean = 3, sd = 0.5)
#   x2 <- runif(n, 0, 7)
#   x3 <- rexp(n, 1/3)
#   x4 <- rbinom(n, 10, 0.2)
```

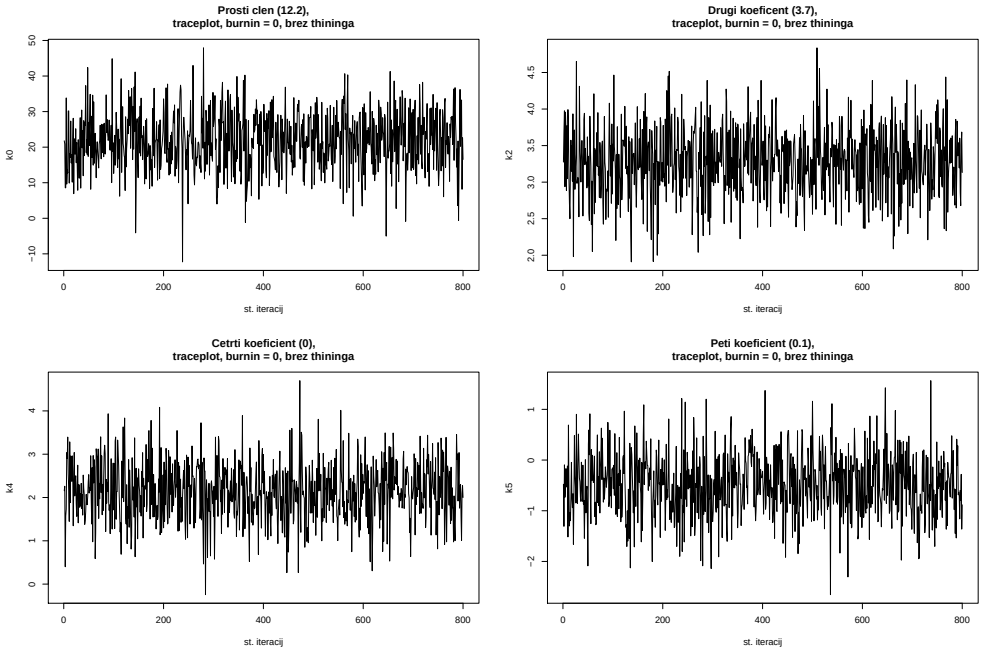
```
#   x5 <- rpois(n, 2.5)
#
#   napaka1 <- rnorm(n, mean = 0, sd = 0.01)
#   napaka2<- rnorm(n, mean = 0, sd = 10)
#   napaka3 <- rchisq(n, df = 6)
#
#   y1 <- 12.2 + 6*x1 + 3.7*x2 - 0*x3 + 1.5*x4 + 0.1*x5 + napaka1
#   y2 <- 12.2 + 6*x1 + 3.7*x2 - 0*x3 + 1.5*x4 + 0.1*x5 + napaka2
#   y3 <- 12.2 + 6*x1 + 3.7*x2 - 0*x3 + 1.5*x4 + 0.1*x5 + napaka3
#
#   realizacije <- data.frame(x1, x2, x3, x4, x5, y1, y2, y3)
#
#   klr1 <- lm(y1 ~ x1 + x2 + x3 + x4 + x5, data=realizacije)
#   klr2 <- lm(y2 ~ x1 + x2 + x3 + x4 + x5, data=realizacije)
#   klr3 <- lm(y3 ~ x1 + x2 + x3 + x4 + x5, data=realizacije)
#
#
#   bayes1 <- bayesx(y1 ~ x1 + x2 + x3 + x4 + x5, data = realizacije,
#                    family = "gaussian", method = "MCMC")
#   bayes2 <- bayesx(y2 ~ x1 + x2 + x3 + x4 + x5, data = realizacije,
#                    family = "gaussian", method = "MCMC")
#   bayes3 <- bayesx(y3 ~ x1 + x2 + x3 + x4 + x5, data = realizacije,
#                    family = "gaussian", method = "MCMC")
#
#   klr1.koef[i,] <- klr1$coefficients
#   klr1.odklon[i,] <- summary(klr1)$coef[,2]
#   b1.povp[i,] <- bayes1$fixed.effects[,1]
#   b1.odklon[i,] <- bayes1$fixed.effects[,2]
#
#   klr2.koef[i,] <- klr2$coefficients
#   klr2.odklon[i,] <- summary(klr2)$coef[,2]
#   b2.povp[i,] <- bayes2$fixed.effects[,1]
#   b2.odklon[i,] <- bayes2$fixed.effects[,2]
#
#   klr3.koef[i,] <- klr3$coefficients
#   klr3.odklon[i,] <- summary(klr3)$coef[,2]
#   b3.povp[i,] <- bayes3$fixed.effects[,1]
#   b3.odklon[i,] <- bayes3$fixed.effects[,2]
# }
#
# dump(c("klr1.koef", "klr1.odklon", "b1.povp", "b1.odklon",
#        "klr2.koef", "klr2.odklon", "b2.povp", "b2.odklon",
#        "klr3.koef", "klr3.odklon", "b3.povp", "b3.odklon"),
#      file = "simulacija.R")
```

```
source("simulacija.R")
```

Izbira burnina, thininga, st. iteracij

Demonstracijo spreminjanja parametrov funkcije bayesx sem naredil le na 2. primeru, torej kjer je napaka porazdeljena $N(0, 10)$ (velika varianca).

```r
x1 <- rnorm(n, mean = 3, sd = 0.5)
x2 <- runif(n, 0, 7)
x3 <- rexp(n, 1/3)
x4 <- rbinom(n, 10, 0.2)
x5 <- rpois(n, 2.5)

napaka2<- rnorm(n, mean = 0, sd = 10)

y2 <- 12.2 + 6*x1 + 3.7*x2 - 0*x3 + 1.5*x4 + 0.1*x5 + napaka2

realizacije <- data.frame(x1, x2, x3, x4, x5, y2)


# PRVIČ
bayes2 <- bayesx(y2 ~ x1 + x2 + x3 + x4 + x5, data = realizacije,
                 family = "gaussian", method = "MCMC", burnin = 0, step = 1)



k0 <- attr(bayes2$fixed.effects, "sample")[1:800, 1]
k2 <- attr(bayes2$fixed.effects, "sample")[1:800, 3]
k4 <- attr(bayes2$fixed.effects, "sample")[1:800, 5]
k5 <- attr(bayes2$fixed.effects, "sample")[1:800, 6]

par(mfrow = c(2, 2))
plot(k0, type = "l", main = "Prosti člen (12.2),\ntraceplot, burnin = 0, brez thininga",
     xlab = "st. iteracij")
plot(k2, type = "l", main = "Drugi koeficent (3.7),\ntraceplot, burnin = 0, brez thininga",
     xlab = "st. iteracij")
plot(k4, type = "l", main = "Četrti koeficient (0),\ntraceplot, burnin = 0, brez thininga",
     xlab = "st. iteracij")
plot(k5, type = "l", main = "Peti koeficient (0.1),\ntraceplot, burnin = 0, brez thininga",
     xlab = "st. iteracij")
```

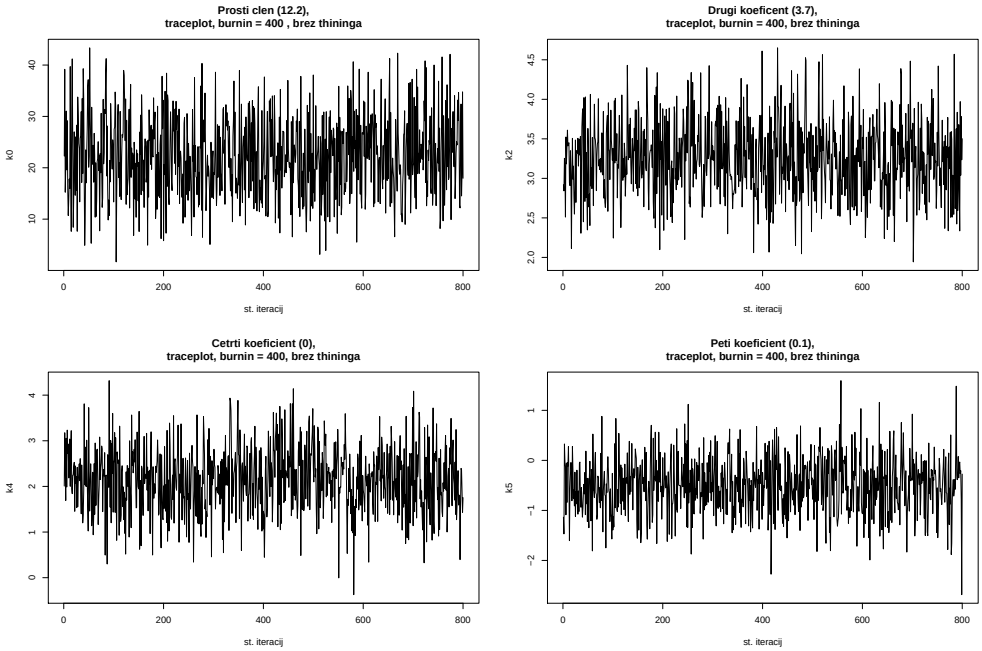


Za začetek sem za nisem zahteval nobenega burnin vzorca, prav tako je thining za boljso sliko in predstavo nastavljen na 1, torej ga ni, vzamemo vsako iteracijo. Pričakovano se vidi da vsaj v prvih 400 iteracijah skoraj pri vseh koeficientih pristranskost bila velika, kar seveda ni dobro. Pričakovano takšna izbira burnina in thininga ne pride v postev.

```
# DRUGIČ
bayes2 <- bayesx(y2 ~ x1 + x2 + x3 + x4 + x5, data = realizacije,
                 family = "gaussian", method = "MCMC", burnin = 400, step = 1)
```

```
## Note: created new output directory 'C:/Users/HP/AppData/Local/Temp/RtmpkHmX8h/bayesx1'!
```

```
k0 <- attr(bayes2$fixed.effects, "sample")[1:800, 1]
k2 <- attr(bayes2$fixed.effects, "sample")[1:800, 3]
k4 <- attr(bayes2$fixed.effects, "sample")[1:800, 5]
k5 <- attr(bayes2$fixed.effects, "sample")[1:800, 6]

par(mfrow = c(2, 2))
plot(k0, type = "l", main = "Prosti člen (12.2),\ntraceplot, burnin = 400 , brez thininga",
     xlab = "st. iteracij")
plot(k2, type = "l", main = "Drugi koeficent (3.7),\ntraceplot, burnin = 400, brez thininga",
     xlab = "st. iteracij")
plot(k4, type = "l", main = "Četrti koeficient (0),\ntraceplot, burnin = 400, brez thininga",
     xlab = "st. iteracij")
plot(k5, type = "l", main = "Peti koeficient (0.1),\ntraceplot, burnin = 400, brez thininga",
     xlab = "st. iteracij")
```

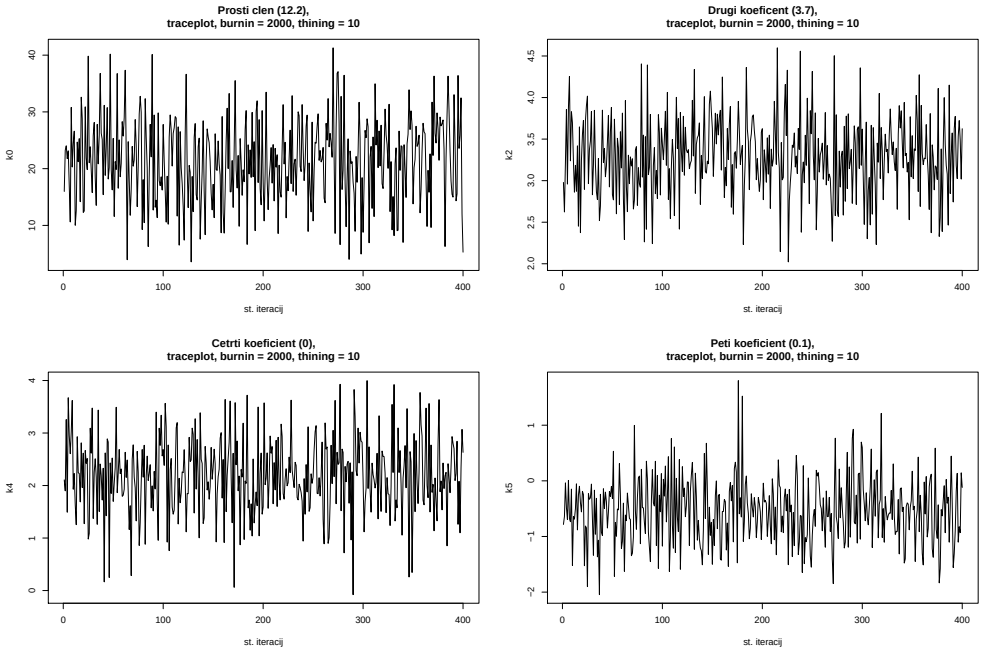


Za drugo demonstracijo sem za burnin vzorec vzel 400 iteracij, ponovno brez thiniga. Tezave nastopajo predvsem pri prostem členu in četrtem koeficientu, se pa tekom vzorca v povprečju približujemo željenim vrednostim.

```
# TRETJIČ
bayes2 <- bayesx(y2 ~ x1 + x2 + x3 + x4 + x5, data = realizacije,
                 family = "gaussian", method = "MCMC")
```

```
## Note: created new output directory 'C:/Users/HP/AppData/Local/Temp/RtmpkHmX8h/bayesx2'!
```

```
k0 <- attr(bayes2$fixed.effects, "sample")[1:400,1]
k2 <- attr(bayes2$fixed.effects, "sample")[1:400,3]
k4 <- attr(bayes2$fixed.effects, "sample")[1:400,5]
k5 <- attr(bayes2$fixed.effects, "sample")[1:400,6]

par(mfrow = c(2, 2))
plot(k0, type = "l", main = "Prosti člen (12.2),\ntraceplot, burnin = 2000, thining = 10",
     xlab = "st. iteracij")
plot(k2, type = "l", main = "Drugi koeficent (3.7),\ntraceplot, burnin = 2000, thining = 10",
     xlab = "st. iteracij")
plot(k4, type = "l", main = "Četrti koeficient (0),\ntraceplot, burnin = 2000, thining = 10",
     xlab = "st. iteracij")
plot(k5, type = "l", main = "Peti koeficient (0.1),\ntraceplot, burnin = 2000, thining = 10",
     xlab = "st. iteracij")
```



Na koncu sem se odločil kar za default izbiro burnina, torej 2000 in za thining 10, torej vzamemo vsako 10 iteracijo, tako bo naš sample dolžine 1000. Tudi po slikah se po 2000 iteracijha že vsi v povprečju gibljejo okoli željenih vrednosti. Tudi rezultati iz pristranskosti v nadaljevanju naloge ne kažejo, da bi bila takšna izbira thininga in burnina napačna.

Primerjava regresij

```
koef.teor <- c(12.2, 6, 3.7, 0, 1.5, 0.1) # koeficienti modela v teoriji
```

```
pr1l <- sapply(klr1.koef, mean) - koef.teor
pr2l <- sapply(klr2.koef, mean) - koef.teor
pr3l <- sapply(klr3.koef, mean) - koef.teor
pr1b <- sapply(b1.povp, mean) - koef.teor
pr2b <- sapply(b2.povp, mean) - koef.teor
pr3b <- sapply(b3.povp, mean) - koef.teor

pr <- data.frame(pr1l, pr1b, pr2l, pr2b, pr3l, pr3b) %>%
  rename("Lin. reg 1" = pr1l, "Lin. reg 2" = pr2l, "Lin. reg 3" = pr3l,
         "Bay. reg 1" = pr1b, "Bay. reg 2" = pr2b, "Bay. reg 3" = pr3b)

pr2 <- transpose(pr)
colnames(pr2) <- rownames(pr)
rownames(pr2) <- colnames(pr)

knitr::kable(pr2, caption = "Pristranskost")
```

Table 1: Pristranskost

	Intercept	x1	x2	x3	x4	x5
Lin. reg 1	-0.0000676	0.0000038	0.0000108	-0.0000056	-0.0000183	0.0000181
Bay. reg 1	-0.0001002	0.0000103	0.0000090	-0.0000025	-0.0000174	0.0000205
Lin. reg 2	0.0870873	-0.0267078	-0.0211149	-0.0067717	0.0349103	0.0040831
Bay. reg 2	0.0918759	-0.0283185	-0.0212697	-0.0069090	0.0349104	0.0041279
Lin. reg 3	5.9461679	0.0337633	0.0016611	0.0035963	-0.0157437	-0.0104445
Bay. reg 3	5.9501413	0.0321175	0.0018322	0.0036278	-0.0157409	-0.0103068

Pristranskost nam pove za koliko se v koeficienti pridobljeni iz regresij v povprecju razlikujejo od koeficientov nasega modela. Izracuna se po formuli $Bias_\theta(\hat{\theta}, \theta) = E_\theta(\hat{\theta}) - \theta$.

Po pregledu tabele je videti da so odstopanja najnizja v prvem modelu, torej normalnem z majhno varianco, kar je pricakovano saj je zaradi manjse razprsenosti simetrične napake verjetnost, da bi napaka "pobegnila" v previsoka odstopanja od ničle v eno ali drugo smer nizka.

V drugem primeru z simetrično napako z višjo varianco so odstopanja od koeficientov iz modela nekoliko višja, a vseeno zadosti dobra, da so se vedno uporabna.

V tretjem modelu, kjer je napaka asimetrična, pa je pristranskost pri prostem členu zelo blizu 6, kar je pričakovano, saj je povprečna vrednost χ^2_6 porazdelitve enaka kar številu prostostnih stopenj in tako koeficient prostega člena iz naših simulacij v povprečju zajame še povprečno vrednost asimetrične napake. Pri drugih koeficientih je pristranskost nizka, kar je zadavoljivo, pravilno in pričakovano.

```
st1l <- sapply(klr1.koef, sd) / sqrt(effectiveSize(klr1.koef))
st2l <- sapply(klr2.koef, sd) / sqrt(effectiveSize(klr2.koef))
st3l <- sapply(klr3.koef, sd) / sqrt(effectiveSize(klr3.koef))
st1b <- sapply(b1.povp, sd) / sqrt(effectiveSize(b1.povp))
st2b <- sapply(b2.povp, sd) / sqrt(effectiveSize(b2.povp))
st3b <- sapply(b3.povp, sd) / sqrt(effectiveSize(b3.povp))

stan.napaka <- data.frame(st1l, st1b, st2l, st2b, st3l, st3b) %>%
  rename("Lin. reg 1" = st1l, "Lin. reg 2" = st2l, "Lin. reg 3" = st3l,
         "Bay. reg 1" = st1b, "Bay. reg 2" = st2b, "Bay. reg 3" = st3b)

stan.napaka2 <- transpose(stan.napaka)
colnames(stan.napaka2) <- rownames(stan.napaka)
rownames(stan.napaka2) <- colnames(stan.napaka)

knitr::kable(stan.napaka2, caption = "Standardna napaka")
```

Table 2: Standardna napaka

	Intercept	x1	x2	x3	x4	x5
Lin. reg 1	0.0002241	0.0000676	0.0000156	0.0000114	0.0000270	0.0000214
Bay. reg 1	0.0002257	0.0000679	0.0000156	0.0000114	0.0000275	0.0000214
Lin. reg 2	0.2426664	0.0704064	0.0148428	0.0112388	0.0262539	0.0196532
Bay. reg 2	0.2429715	0.0673984	0.0148335	0.0112549	0.0263280	0.0196570
Lin. reg 3	0.0803888	0.0227117	0.0055511	0.0038875	0.0093974	0.0062945
Bay. reg 3	0.0799675	0.0227186	0.0055455	0.0038884	0.0094000	0.0062824

Standardna napaka nam pove kolikšen je odklon med koeficienti iz simulacij, višji kot je vzorec, nižja bo.

Po pregledu tabele ponovno opazimo, da je najmanj odstopanj v prvem modelu, ponovno pričakovano saj je varianca napake v njem daleč najnižja.

Standardna napaka tretjega modela je tudi pičakovano nižja od napake v drugem modelu, saj je varianca χ^2_6 porazdelitve enaka $\sqrt{2 * df}$, torej v našem primeru $\sqrt{12}$, kar je vseeno mnogo nižje od variance v drugem modelu, pri katerem je varianca napake enaka 100.

```
sk1l <- rep(0, 6)
sk2l <- rep(0, 6)
sk3l <- rep(0, 6)
sk1b <- rep(0, 6)
sk2b <- rep(0, 6)
sk3b <- rep(0, 6)

for (i in 1:6) {
  sk1l[i] <- sqrt(mean((as.matrix(klr1.koef[i])[,1] - koef.teor[i])^2))
  sk2l[i] <- sqrt(mean((as.matrix(klr2.koef[i])[,1] - koef.teor[i])^2))
  sk3l[i] <- sqrt(mean((as.matrix(klr3.koef[i])[,1] - koef.teor[i])^2))
  sk1b[i] <- sqrt(mean((as.matrix(b1.povp[i])[,1] - koef.teor[i])^2))
  sk2b[i] <- sqrt(mean((as.matrix(b2.povp[i])[,1] - koef.teor[i])^2))
  sk3b[i] <- sqrt(mean((as.matrix(b3.povp[i])[,1] - koef.teor[i])^2))
}

sr <- data.frame(sk1l, sk1b, sk2l, sk2b,
                           sk3l, sk3b,
                           row.names = c("Intercept", "x1", "x2",
                                         "x3", "x4", "x5")) %>%
  rename("Lin. reg 1" = sk1l, "Lin. reg 2" = sk2l,
         "Lin. reg 3" =sk3l, "Bay. reg 1" = sk1b,
         "Bay. reg 2" = sk2b, "Bay. reg 3" = sk3b)

sr2 <-transpose(sr)
colnames(sr2) <- rownames(sr)
rownames(sr2) <- colnames(sr)


  # mean(as.matrix(b1.povp[2])[,1] - 6)

knitr::kable(sr2, caption = "Srednja kvadratna napaka")
```

Table 3: Srednja kvadratna napaka

	Intercept	x1	x2	x3	x4	x5
Lin. reg 1	0.0070832	0.0021359	0.0004930	0.0003602	0.0008551	0.0006635
Bay. reg 1	0.0071337	0.0021474	0.0004933	0.0003616	0.0008700	0.0006680
Lin. reg 2	7.2628049	2.1270698	0.5090203	0.3552902	0.8305396	0.6463146
Bay. reg 2	7.2775804	2.1304460	0.5087486	0.3557999	0.8328798	0.6465234
Lin. reg 3	6.4227924	0.7186399	0.1754602	0.1229233	0.2974400	0.2230268
Bay. reg 3	6.4258330	0.7187842	0.1752851	0.1229532	0.2975211	0.2231640

Srednja kvadratna napaka nam pove za koliko se vrednosti koeficientov od vzorčenja razlikujejo od koeficientov v našem modelu, torej nam povejo nekaj o razponu vrednosti ocen iz naših simulacij. Formula zanjo je $\sqrt{E((\hat{\theta} - \theta)^2)}$.

Ponovno je napaka pričakovano najnižja pri modelu z nizko varianco, saj ocene koeficientov po posameznih simulacijah ne odstopajo veliko.

Pri drugem in tretjem modelu je razpon vrednosti simulacij zaradi večjih varianc napak pričakovano višji.

```
psn1l <- sapply(klr1.odklon, mean)
psn2l <- sapply(klr2.odklon, mean)
psn3l <- sapply(klr3.odklon, mean)
psn1b <- sapply(b1.odklon, mean) / sqrt(effectiveSize(b1.odklon))
psn2b <- sapply(b2.odklon, mean) / sqrt(effectiveSize(b2.odklon))
psn3b <- sapply(b3.odklon, mean) / sqrt(effectiveSize(b3.odklon))

psn <- data.frame(psn1l, psn1b, psn2l, psn2b, psn3l, psn3b,
                  row.names =
                    c("Intercept", "x1", "x2", "x3", "x4", "x5")) %>%
  rename("Lin. reg 1" = psn1l, "Lin. reg 2" = psn2l,
         "Lin. reg 3" =psn3l, "Bay. reg 1" = psn1b,
         "Bay. reg 2" = psn2b, "Bay. reg 3" = psn3b)

psn2 <-transpose(psn)
colnames(psn2) <- rownames(psn)
rownames(psn2) <- colnames(psn)

knitr::kable(psn2, caption = "Povprečje standardnih napak")
```

Table 4: Povprečje standardnih napak

	Intercept	x1	x2	x3	x4	x5
Lin. reg 1	0.0070192	0.0020720	0.0005103	0.0003504	0.0008169	0.0006535
Bay. reg 1	0.0008772	0.0003063	0.0000638	0.0000439	0.0001302	0.0000818
Lin. reg 2	7.0188084	2.0718678	0.5103478	0.3504604	0.8171494	0.6535529
Bay. reg 2	0.2243270	0.0662135	0.0162731	0.0112034	0.0261157	0.0208646
Lin. reg 3	2.4227543	0.7150617	0.1761310	0.1208372	0.2819721	0.2253936
Bay. reg 3	0.0774061	0.0228387	0.0056563	0.0038621	0.0098327	0.0075521

Povprečja standardnih napak je znova najnižje v prvem primeru in višje v drugih dveh, vseeno nam rezultati iz pristranosti, ki so ugodni povedo, da ni razloga za skrb.