Labbrapport i Statistik

Laboration 3

732g43

Jakob Lindén

Avdelningen för Statistik och maskininlärning
Institutionen för datavetenskap
Linköpings universitet
10-12-2019

Innehåll

1. Uppgift 1

Kommaden uppgifter behandlar data om lägenhetspriset i miljontalskronor vilket är responsvariabeln y. Följande förklaringsvariabler finns med i datamaterialet:

- x_1 = area i kvadratmeter på lägenheten
- x_2 = antal rum i lägenheten
- x_3 = bostadsavgift per månad i tusentals kronor för lägenheten
- x_4 = antal trappor till lägenheten i bostadshuset
- x_5 = dummyvariabel som är lika med 1 om lägenheten såldes i region City
- x_6 = dummyvariabel som är lika med 1 om lägenheten såldes i Syd.

1.1 a)

```
#Null modell
flist <- alist(
  y ~ dnorm(mu, exp(logsigma)) ,
  mu ~ dnorm( 3 , 10 ) ,
  logsigma ~ dnorm ( 0 , 1 )
)

#En variabel
flist <- alist(
  y ~ dnorm(mu, exp(logsigma)) ,
  mu <- b0 + b1*area ,
  b0 ~ dnorm( 3 , 10 ) ,
  b1 ~ dnorm( 0 , 10 ) ,
  logsigma ~ dnorm ( 0, 1 )
)

#Full modell
flist <- alist(
  y ~ dnorm(mu, exp(logsigma)) ,
  mu <- b0 + b1*area + b2*antal_rum + b3*avgift + b4*trappor + b5*cityyes + b6*sydyes,
  b0 ~ dnorm( 3 , 10 ) ,
  c(b1,b2,b3,b4) ~ dnorm( 0 , 10 ) ,
  c(b5,b6) ~ dnorm( 0 , 10 ) ,
  logsigma ~ dnorm ( 0, 1 )
)
```

I denna uppgift ska tre stycken Bayesianska linjära regerssionsmodeller anpassas som ska användas i kommande uppgifter. Följande modeller ska anpassas:

i. Bayesiansk linjär regressionsmodell utan förklaringsvariabler.

ii. Bayesiansk linjär regressionsmodell med den kontinuerliga förklaringsvariabeln $area$.

iii. Bayesiansk linjär regressionsmodell med alla förklaringsvariabler.

1.2 b)

```
# Deviance Null model
theta_Null <- coef(resNormal_Null)
dev_Null <- (-2)*sum( dnorm(
  data$y ,
  mean=theta_Null[1] ,
  sd=exp(theta_Null[2]) ,
  log=TRUE ) )
dev_Null

# Deviance EnXvar
theta_EnXvar <- coef(resNormal_EnXvar)
dev_EnXvar <- (-2)*sum( dnorm(
  data$y ,
  mean=theta_EnXvar[1] + theta_EnXvar[2]*data$area,
  sd=exp(theta_EnXvar[3]) ,
  log=TRUE ) )
dev_EnXvar

# Deviance Full
theta_Full <- coef(resNormal_Full)
dev_Full <- (-2)*sum( dnorm(
  data$y ,
  mean=theta_Full[1] + theta_Full[2]*data$area + theta_Full[3]*data$antal_rum +
    theta_Full[4]*data$avgift + theta_Full[5]*data$trappor +
    theta_Full[6]*data$cityyes + theta_Full[7]*data$sydyes,
  sd=exp(theta_Full[8]) ,
  log=TRUE ) )
dev_Full
```

Deviance $D(q)$ ska beräknas och redovisas för respektive modell på hela datamaterialet och jämföras modellerna sinsemellan. Deviance är ett mått som baseras på modellens sannolikhetsfördelning för data och är ett mått på den relativa osäkerheten i modellen jämfört med perfekt anpassning. Således beräknar vi $D(q)$ och vill få så litet värde som möjligt på detta värde för att undersöka den bäst anpassade modellen till data. Deviancen för modellen utan förklaringsgrader beräknas till 189.83, för modellen med *area* som enda förklaringsvariabel bärknas deviance till 157.83, tillsist beräknas deviance för den fulla modellen till 112.32. Den fulla modellen hade lägst Deviance vilket är att förvänta eftersom *in-sample deviance* alltid kommer öka då modellen blir mer komplex. Detta kommer tas hänsyn till i kommande uppgift då data delas upp i träning och valideringsmängder för att beräkna *out-of-sample deviance*.

1.3 c)

```
train_data <- data[1:29,]
vali_data <- data[30:57,]

# Train deviance
#################

theta_Null <- coef(resNormal_Null)
dev_Null_Train <- (-2)*sum( dnorm(
  train_data$y ,
  mean=theta_Null[1] ,
  sd=exp(theta_Null[2]) ,
  log=TRUE ) )
dev_Null_Train
```

```
# Deviance EnXvar

theta_EnXvar <- coef(resNormal_EnXvar)
dev_EnXvar_Train <- (-2)*sum( dnorm(
  train_data$y ,
  mean=theta_EnXvar[1] + theta_EnXvar[2]*train_data$area,
  sd=exp(theta_EnXvar[3]) ,
  log=TRUE ) )
dev_EnXvar_Train

# Deviance Full
theta_Full <- coef(resNormal_Full)
dev_Full_Train <- (-2)*sum( dnorm(
  train_data$y ,
  mean=theta_Full[1] + theta_Full[2]*train_data$area + theta_Full[3]*train_data$antal_rum +
    theta_Full[4]*train_data$avgift + theta_Full[5]*train_data$trappor +
    theta_Full[6]*train_data$cityyes + theta_Full[7]*train_data$sydyes,
  sd=exp(theta_Full[8]) ,
  log=TRUE ) )
dev_Full_Train

# Validation deviance
#################

theta_Null <- coef(resNormal_Null)
dev_Null_Vali <- (-2)*sum( dnorm(
  vali_data$y ,
  mean=theta_Null[1] ,
  sd=exp(theta_Null[2]) ,
  log=TRUE ) )
dev_Null_Vali

# Deviance EnXvar

theta_EnXvar <- coef(resNormal_EnXvar)
dev_EnXvar_Vali <- (-2)*sum( dnorm(
  vali_data$y ,
  mean=theta_EnXvar[1] + theta_EnXvar[2]*vali_data$area,
  sd=exp(theta_EnXvar[3]) ,
  log=TRUE ) )
dev_EnXvar_Vali

# Deviance Full

theta_Full <- coef(resNormal_Full)
dev_Full_Vali <- (-2)*sum( dnorm(
  vali_data$y ,
  mean=theta_Full[1] + theta_Full[2]*vali_data$area + theta_Full[3]*vali_data$antal_rum +
    theta_Full[4]*vali_data$avgift + theta_Full[5]*vali_data$trappor +
    theta_Full[6]*vali_data$cityyes + theta_Full[7]*vali_data$sydyes,
  sd=exp(theta_Full_vali[8]) ,
  log=TRUE ) )
dev_Full_Vali
```

Data ska nu delas upp i träning och validerings mängd där de första 29 observationerna i data används till träningoch resterande observationer används till validering. Deviance ska beräknas för respektive sample och jämföras mellan modellerna. För modellen utan förklaringsvariabler beräknas deviance för träning till 106.93 och för validering till 82.90. Modellen med förklaringsvariabeln *area* har deviance för träning på 87.90 och validering på 69.92. Slutligen den fulla moedllen har deviance för träning på 61.58 och validering

på 50.55. Här kan vi se att den fulla modellen presterar bäst i både *out-of-sample* och *in-sample deviance*. Ett förväntat resultat är att deviancen för träningsmängden alltid kommer sjunka då flera parametrar läggs till i modellen. För valideringsmängden förväntas dock deviance öka i genomsnitt då fler variabler läggs till. Vi kan i detta fall se att den fulla modellen inte överanpassar data eftersom den presterar bäst i *out-of-sample deviance*, det går dock att se tendensen av ökning i träningsmängdens deviance då minskningen av deviance från träning till validering minskar vid mer komplex modell.

1.4 d)

Måtten AIC, DIC och WAIC ska beräknas för respektive modell och jämföras mellan modellerna. Alla dessa mått är mått på modellens prediktiva förmåga alltså *out-of-sample deviance*, således kan dessa avgöra vilken modell som göra bäst prediktioner.

1.4.1 AIC

```
AIC_Null <- dev_Null + 2 * length(coef(resNormal_Null))
AIC_EnXvar <- dev_EnXvar + 2 * length(coef(resNormal_EnXvar))
AIC_Full <- dev_Full + 2 * length(coef(resNormal_Full))
```

AIC för modellen utan förklaringsvariabler beräknas till 193.83. För modellen med *area* som förklaringsvariabel beräknas AIC till 163.83. Den fulla modellen har ett AIC värde på 128.32. Således ses den fulla modellen ha bäst prediktiv förmåga. AIC är endast en bra approximation av modellens prediktiva förmåga om modellen har flacka priors, posteriorn är approxmativt multivariat normalfördelad och antalet observationer N är mycket större än antalet parametrar p.

AIC är straffande för fler antal parametrar, trots detta preseterar den fulla modellen bäst med avseende på detta mått.

1.4.2 DIC

```
#Null modell

samples_Null <- extract.samples(resNormal_Null, n = 1e4)

deviance_posterior_NULL <- c(rep(NA,nrow(samples_Null)))
for (i in 1:nrow(samples_Null)) {
  deviance_posterior_NULL[i] <- (-2)*sum( dnorm(
    data$y ,
    mean=samples_Null[i,1] ,
    sd=exp(samples_Null[i,2]) ,
    log=TRUE ) )
}


D_bar_null <- mean(deviance_posterior_NULL)
D_hat_null <- (-2)*sum( dnorm(
  data$y ,
  mean=mean(samples_Null[,1]) ,
  sd=exp(mean(samples_Null[,2])) ,
  log=TRUE ) )
DIC_null <- D_bar_null + (D_bar_null - D_hat_null)

#EnXVar Modell

samples_EnXvar <- extract.samples(resNormal_EnXvar, n = 1e4)
```

```
deviance_posterior_EnXvar <- c(rep(NA,nrow(samples_EnXvar)))
for (i in 1:nrow(samples_EnXvar)) {
  deviance_posterior_EnXvar[i] <- (-2)*sum( dnorm(
    data$y ,
    mean=samples_EnXvar[i,1] + samples_EnXvar[i,2]*data$area ,
    sd=exp(samples_EnXvar[i,3]) ,
    log=TRUE ) )
}

D_bar_EnXvar <- mean(deviance_posterior_EnXvar)
D_hat_EnXvar <- (-2)*sum( dnorm(
  data$y ,
  mean=mean(samples_EnXvar[,1]) + mean(samples_EnXvar[,2]) * data$area ,
  sd=exp(mean(samples_EnXvar[,3])) ,
  log=TRUE ) )
DIC_EnXvar <- D_bar_EnXvar + (D_bar_EnXvar - D_hat_EnXvar)

#Full Modell
samples_Full <- extract.samples(resNormal_Full, n = 1e4)

deviance_posterior_Full <- c(rep(NA,nrow(samples_Full)))
for (i in 1:nrow(samples_Full)) {
  deviance_posterior_Full[i] <- (-2)*sum( dnorm(
    data$y ,
    mean=samples_Full[i,1] + samples_Full[i,2]*data$area +
      samples_Full[i,3]*data$antal_rum + samples_Full[i,4]*data$avgift +
      samples_Full[i,5]*data$trappor + samples_Full[i,6]*data$cityyes +
      samples_Full[i,7]*data$sydyes,
    sd=exp(samples_Full[i,8]) ,
    log=TRUE ) )
}

D_bar_Full <- mean(deviance_posterior_Full)
D_hat_Full <- (-2)*sum( dnorm(
  data$y ,
  mean=mean(samples_Full[,1]) + mean(samples_Full[,2])*data$area +
    mean(samples_Full[,3])*data$antal_rum + mean(samples_Full[,4])*data$avgift +
    mean(samples_Full[,5])*data$trappor + mean(samples_Full[,6])*data$cityyes +
    mean(samples_Full[,7])*data$sydyes,
  sd=exp(mean(samples_Full[,8])) ,
  log=TRUE ) )
DIC_Full <- D_bar_Full + (D_bar_Full - D_hat_Full)
```

DIC beräknas till 193.85, 163.88 och 128.67 för modellen utan förklaringsvariabler, modellen med *area* som förklaringsvariabel och den fulla modellen. DIC är ett mer generellt mått än AIC eftersom det inte kräver flacka priors. Även DIC är straffande för fler antal parametrar, trots detta presterar den fulla modellen bäst med avseende detta mått.

1.4.3 WAIC

```
WAIC_Null <- WAIC(resNormal_Null)[1]
WAIC_EnXvar <- WAIC(resNormal_EnXvar)[1]
WAIC_Full <- WAIC(resNormal_Full)[1]
```

WAIC för modellen utan förklaringsvariabler beräknas till 193.94. För modellen med *area* som förklaringsvariabel beräknas WAIC till 163.62 och slutligen för den fulla modellen till 128.96. WAIC är ett mer generellt mått än både AIC och DIC för den inte kräver något av antagandena som gäller för AIC och

DIC. Så även vid detta mått presterar den fulla modellen bäst och bedöms ha bäst prediktiv förmåga av de tre modellerna.

Slutligen dras slutsatsen att den fulla modellen har bäst prediktiva förmåga med avseende alla mått. Det måttet som lämpar sig bäst är WAIC eftersom antalet observationer N i detta datamaterial inte är mycket större än antalet parametrar p.

1.5 e)

```
#i
flist <- alist(
  y ~ dnorm(mu, exp(logsigma)) ,
  mu <- b0 + b1*antal_rum ,
  b0 ~ dnorm( 3 , 10 ) ,
  b1 ~ dnorm( 0 , 10 ) ,
  logsigma ~ dnorm ( 0, 1 )
)

resNormal_EnXvar_2ndKorr <- map(flist, data=data)
#ii
resNormal_EnXvar
#iii
resNormal_Full
```

Följande tre Bayesianska linjära regressionmodeller kommer användas i kommande uppgifter:

i. Bayesiansk linjär regressionsmodell med förklaringsvaribeln $antal\_rum$

ii. Bayesiansk linjär regressionsmodell med förklaringsvariabeln $area$

iii. Bayesiansk linjär regressionsmodell med alla förklaringsvariabler.

1.6 f)

```
WAIC_EnXvar_2ndKorr <- WAIC(resNormal_EnXvar_2ndKorr)[1]

WAIC_df <- data.frame("Modell" = c(WAIC_EnXvar ,WAIC_EnXvar_2ndKorr, WAIC_Full), "Weight" =
    c(NA,NA,NA))
rownames(WAIC_df)<-c("EnXvar","EnXvar_2ndkorr","FULL")

for(i in 1:nrow(WAIC_df)) {
  WAIC_df[i,2] <- exp(-0.5 * (WAIC_df[i,1] - min(WAIC_df[,1]))) /
                  sum(exp(-0.5 * (WAIC_df[,1] - min(WAIC_df[,1]))))
}
```

Akaike-vikter ska beräknas för var och en av de tre modellerna och tolkas i ord. Vikterna beräknas där vikten för modellen med $area$ som förklaringsvaribel får en väldigt liten vikt på 0 avrundat med 7 decimaler. För modellen med $antal\_rum$ är också vikten 0 avrundat med 7 decimaler. Således kommer den fulla modellen ha en vikt på 1 avrundat med sju decimaler. Akaike vikterna är ett sätt att undersöka modellernas relativ prediktionsförmåga mellan varandra. Modellen med $area$ som förklaringsvariabel har således 0 procent sannolikhet att ge bäst prediktioner på ny data givet alla tre modellerna. Modellen med $antal\_rum$ som förklaringsvariabel har också 0 procent sannolikhet att ge bäst prediktioner på ny data givet alla tre modellerna. Således har den fulla modellen 100 procent sannolikhet att ge bäst prediktioner på ny data givet alla tre modellerna.

1.7 g)

```
sample_EnXvar <- extract.samples(resNormal_EnXvar,n = 1e4)
sample_EnXvar2ndKorr <- extract.samples(resNormal_EnXvar_2ndKorr,n = 1e4)
sample_Full <- extract.samples(resNormal_Full,n=1e4)

y <- matrix(NA,ncol=3,nrow=1e4)
colnames(y) <- c("EnXvar","EnXvar2ndKorr","Full")
for (i in 1:1e4) {
  y[i,1] <- sample_EnXvar[i,1] + sample_EnXvar[i,2] * mean(data$area)
  y[i,2] <- sample_EnXvar2ndKorr[i,1] + sample_EnXvar2ndKorr[i,2] * mean(data$antal_rum)
  y[i,3] <- sample_Full[i,1] + sample_Full[i,2]*mean(data$area) +
            sample_Full[i,3]*mean(data$antal_rum) +
            sample_Full[i,4]*mean(data$avgift) +
            sample_Full[i,5]*mean(data$trappor) +
            sample_Full[i,6]*0 +
            sample_Full[i,7]*0
}

y_intervall <- WAIC_df[1,2]*y[,1] + WAIC_df[2,2]*y[,2] + WAIC_df[3,2]*y[,3]
PI(y_intervall,prob = 0.952)
```

Ett 95.2 procent kredibilitetsintervall för y ska beräknas för de genomsnittliga värdena på förklaringsvariablerna för de tre modellerna i uppgift (e). De genomsnittliga värdena på de binära variablerna *cityyes* och *sydyes* kommer inte kunna användas eftersom dessa är binära. Istället undersöks den vanligaste kombinationen av de två binära variablerna vilket är då $cityyes = 0$ och $sydyes = 0$.

Intervallet beräknas och ses ligga mellan 2.26 och 3.33. Således ligger y mellan 2.26 och 3.33 med 95.2 procent sannolikhet.

1.8 h)

```
#Detta upprepas 20 gånger.
grid_sd <- seq(from = 0.01,to = 2, by=0.1)
h_df <- matrix(NA, ncol=2,nrow=20)
i <- 1
flist <- alist(
    y ~ dnorm(mu, exp(logsigma)) ,
    mu <- b0 + b1*area + b2*antal_rum + b3*avgift + b4*trappor + b5*cityyes + b6*sydyes,
    b0 ~ dnorm( 3 , 10 ) ,
    c(b1,b2,b3,b4,b5,b6) ~ dnorm( 0 , 0.01 ) ,
    logsigma ~ dnorm ( 0, 1 )
  )

resNormal_grid_sd <- map(flist, data=data)
DICen <- DIC(resNormal_grid_sd)

h_df[i,1] <- 0.01
h_df[i,2] <- DICen[1]
```

Samma modell som *modell iii.* i uppgift (e) ska nu anpassas med samma normalfördelade regulariserande prior för varje lutningsparameter. Med hjälp av DIC ska ett beslut tas om vilken standardavvikelse för den regulariserande priorn som ger den bästa modellen. 20 stycken olika värden på standardavvikelsen kommer undersökas. Det intervall som kommer undersökas är en standardavvikelse mellan 0.01 och 2 med 0.1 hopp.

I tabell 1.1 ses utvecklingen av DIC då standardavvikelsen i den regulariserande priorn ökar. Med hjälp

av dessa 20 värden går det att se att någonstans efter 0.61 i standardavvikelse så ligger värdet på DIC kontinuerligt runt 128. Således kan slutsatsen dras att med den observerade griden av 20 värden kan ett värde på ungefär 0.61 väljas eftersom vi vill ha så låg standardavvikelse som möjligt för att bromsa in lärandet från data men samtidigt inte underanpassa modellen genom att ta för tight prior.

Tabell 1.1

i	sd	DIC
1	0.01	193.44
2	0.11	158.78
3	0.21	140.24
4	0.31	133.26
5	0.41	130.96
6	0.51	129.30
7	0.61	128.44
8	0.71	128.31
9	0.81	128.01
10	0.91	127.95
11	1.01	128.14
12	1.11	128.07
13	1.21	128.53
14	1.31	128.58
15	1.41	127.99
16	1.51	128.28
17	1.61	128.33
18	1.71	127.97
19	1.81	128.58
20	1.91	128.81

2. Uppgift 2

I följande uppgift ska en Bayesiansk linjär regressionsmodell skapas med alla förklaringsvariabler med hjälp av funktionen map2stan i R. Antalet posterior dragningar som väljs är 3000 per MCMC kedja efter uppvärmingsfas och antalet dragning i uppvärmingsfasen är 1000 per MCMC kedja där 4 MCMC kedjor kommer avnändas. Denna modell anpassad med MCMC kommer jämföras med modellen skapad med kvadrisk approximation i uppgift 1(a) iii.

2.1 a)

```
# Skatta modell med map2stan
stan.modell <- alist( data_mcmc$y ~ dnorm( mu , sigma ) ,
                        mu <- b0 + b1*area + b2*antal_rum + b3*avgift + b4*trappor + b5*cityyes
                            + b6*sydyes ,
                        b0 ~ dnorm(3,10),
                        b1 ~ dnorm(0,10),
                        b2 ~ dnorm(0,10),
                        b3 ~ dnorm(0,10),
                        b4 ~ dnorm(0,10),
                        b5 ~ dnorm(0,10),
                        b6 ~ dnorm(0,10),
                        sigma ~ dcauchy(0,2)
)

stan.man <- map2stan(stan.modell,data=data_mcmc,iter = 4000,warmup = 1000,chains = 4)
precis(stan.man,prob = 0.952)
precis(resNormal_Full,prob= 0.952)

#Logsigma -> Sigma
test_samples <- extract.samples(resNormal_Full)
sigma<-exp(test_samples$logsigma)
precis(sigma,prob=0.952)
```

Posteriorresultaten från de två modellerna skattade med de två olika metoderna ska jämföras. Då man undersöker medelvärdes och standardavvikelse skattningarna för parametrarna med respektive metod så ses det vara väldigt liten skillnad där det på vissa parametrar kan skilja sig på tredje decimalen. Lite större skillnader kan observeras på kredibilitetsintervallen där skillnaderna ibland kan ligga på andra decimalen istället för tredje, där man även kan observera att modellen skattad med kvadratisk approximation verkar skatta lite snävare intervall generellt. Dessa små skillnader kan bero på lite olika faktorer. En av dom skulle kunna vara att kvadratisk approximation antar att posteriorn är normalfördelad, vilket den är, men kanske inte perfekt normalfördelad för att få en perfekt anpassning. Men som tidigare påvisat är skillnaderna väldigt små i detta fall, detta eftersom data är tillräckligt normalfördelad för att den kvadratiska approximationen ska göra en bra anpassning.

Tabell 2.1: Modell skattad med MCMC

	Mean	StdDev	Lower 95.52%	Upper 95.52%	n_eff	Rhat
b0	2.80	0.30	2.19	3.40	5894	1
b1	1.32	0.31	0.72	1.94	7942	1
b2	-0.07	0.25	-0.57	0.42	8739	1
b3	-0.12	0.24	-0.60	0.36	8925	1
b4	0.24	0.11	0.04	0.46	9322	1
b5	1.47	0.36	0.77	2.21	5969	1
b6	0.23	0.34	-0.45	0.89	6297	1
sigma	0.71	0.07	0.57	0.86	8692	1

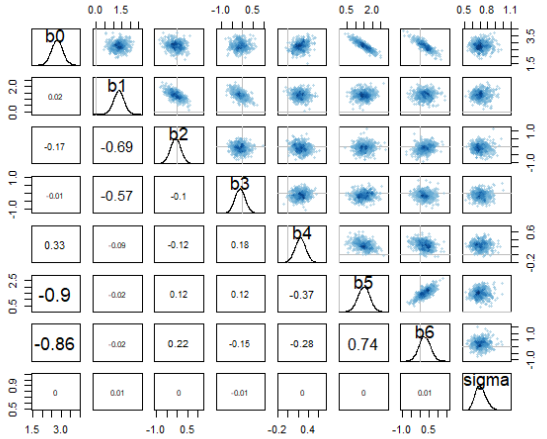
Tabell 2.2: Modell skattad med kvadratisk approximation

	Mean	StdDev	Lower 95.2%	Upper 95.2%
b0	2.80	0.27	2.26	3.34
b1	1.33	0.29	0.77	1.88
b2	-0.07	0.23	-0.55	0.39
b3	-0.12	0.22	-0.56	0.32
b4	0.24	0.10	0.05	0.43
b5	1.48	0.33	0.83	2.12
b6	0.24	0.30	-0.36	0.84
sigma	0.65	0.06	0.54	0.78

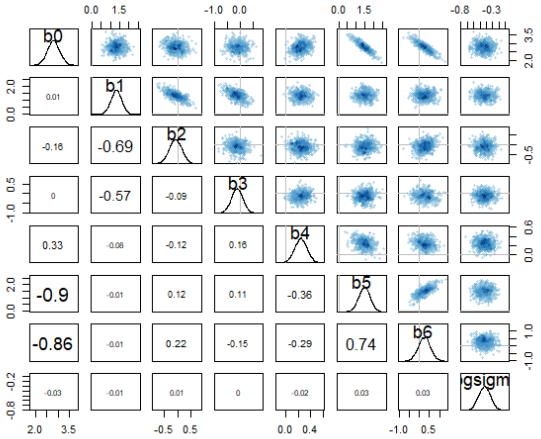
2.2 b)

```
pairs(stan.man)
pairs(resNormal_Full)
```

En figur för respektive metod av MCMC och kvadratisk approximation med alla separata posteriorfördelningar ska redovisas. Även alla parvisa posteriorfördelningar och alla parvisa korrelationskoefficienter för parametrarna. I figur 2.1 och 2.2 ses alla separata och bivariata posteriorfördelningar för parametrarna i respektive modell. Likt föregående uppgift ses anpassningen för respektive metod var väldigt lika, den största skillnaden ses vara i sigma för respektive metod eftersom annorlunda priorfördelningar valdes. Det är rimligt att de båda metoderna producerar liknande resultat när modellen och data för posteriorfördelningen ses vara nästan multivariat normalfördelad med viss skevhet. Kvadratisk approximation verkar fungera bra utifrån dom separata posteriorfördelningarna eftersom den har producerat nästan identiskt resultat som MCMC anpassningen, eftersom MCMC är en mer robust metod som inte kräver några antaganden.



Figur 2.1: MCMC



Figur 2.2: Kvadratisk approximation

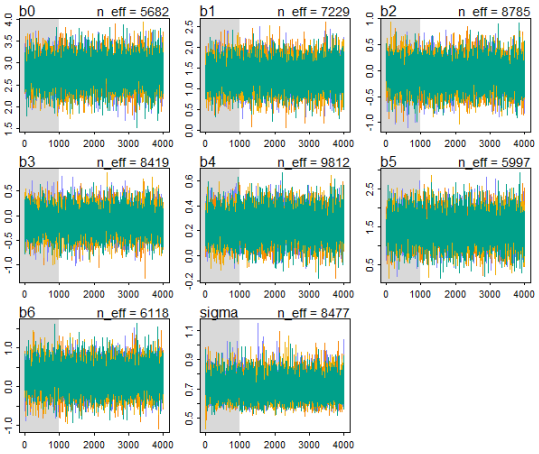
2.3 c)

```
Start <- coef(resNormal_Full)
stan.man.start <- map2stan(stan.modell,data=data_mcmc,
        start=list(b0=Start[1],b1=Start[2],b2=Start[3],b3=Start[4],
                   b4=Start[5],b5=Start[6],b6=Start[7],sigma=Start[8]),
        chains=4,iter=4000,warmup=1000)
tracerplot(stan.man.start)


plot(stan.man)
PostSamp <- extract.samples(stan.man)

AnvPar <- PostSamp$b1
NIter <- length(PostSamp$b1)
Means <- matrix(0,nrow=NIter,ncol=1,byrow=TRUE)
for (iter in 1:NIter){
  Means[iter] <- mean(AnvPar[1:iter])
}
plot(Means)

Quant <- matrix(0,nrow=NIter,ncol=4,byrow=TRUE)
for (iter in 1:NIter){
  Quant[iter,] <- quantile(AnvPar[1:iter],probs=c(0.25,0.50,0.75,0.99))
}
par(mfrow=c(2,2))
plot(Quant[,1])
plot(Quant[,2])
plot(Quant[,3])
plot(Quant[,4])
```

För att utvärdera och dra slutsats om MCMC dragningarna har lett till att MCMC algoritmen konvergerat till posteriorn ska diverse metoder användas. En av dessa är en traceplott över parameterdragningarna för varje MCMC kedja. Även en plott för det ackumulerade posterior medelvärdet över MCMC samples. En lutningsparameter ska användas för att utvärdera om kvartilerna i posteriorfördelningen för denna lutningsparameter verkar ha konvergerat till ett specifikt värde.

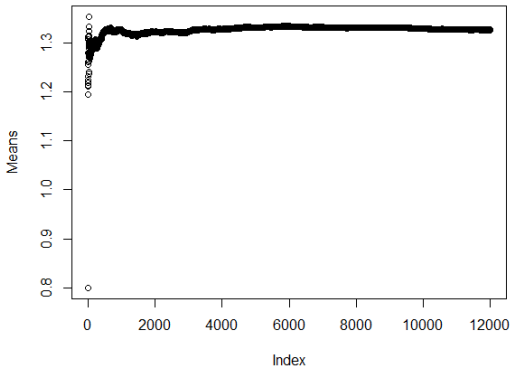
I figur 2.3 ses traceplotten för MCMC modellen, det observeras att det är en städade markov kedjor som producerats, både stationära och väl mixade med zikzak mönster. Den gråa zonen är uppvärmings

regionen där kedjan anpassade sig för att förbättra sampling effektiviteten. Den vita regioner visar de dragningar som använts för inferens.



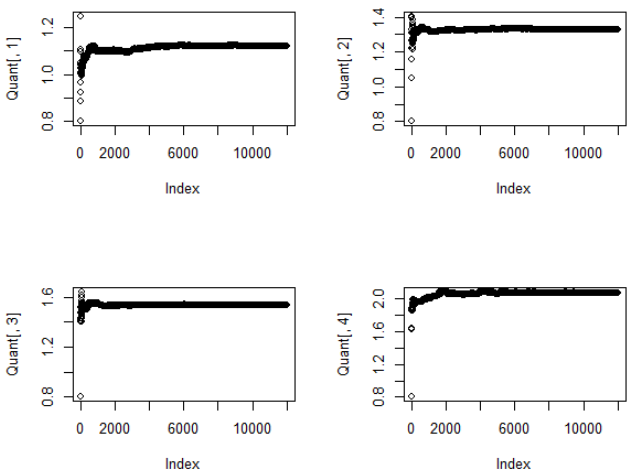
Figur 2.3: Traceplott

De ackumulerade posterior medelvärdet över MCMC samples observeras i figur 2.4. Den här visualiseringen används för att kontrollera så att de ackumulerade posterior medelvärdet konvergerar till ett visst värde över drangningarna. I detta fall finns ett tydligt konvergerat värde lite över 1.3 i posterior medelvärde.



Figur 2.4: Posterior mean över MCMC samples

Slutligen för att undersöka om kvartilerna i posteriorfördelningen för β_1 har konvergerat till ett specifikt värde visualiseras i figur 2.5 kvartilernas konvergens över MCMC samples. För respektive kvartil så ses konvergens vara bra till ett värde. Första kvartilen konvergerar till värdet 1.1 över samples, andra kvartilen till 1.3 över samples. Tredje och fjärde kvartilen konvergerar till 1.5 respektive 2 över samples. Det verkar generellt som ganska få dragning behövs i detta fall för att uppnå konvergens, för att vara säker ungefär 2000 dragningar eftersom fjärde kvartilen har lite osäkerhet fram tills ungefär den 2000e dragningen.



Figur 2.5: Kvartiler över MCMC samples