BDA Project

Dylan, Francesco, Pragati

Contents

1) Introduction

QSAR toxicity

```
QSAR <- read.csv(file = 'qsar_aquatic_toxicity.csv')
colnames(QSAR)
```

```
## [1] "TSPA"                  "Saacc"                 "H050"
## [4] "MLOGP"                 "RDCHI"                 "GATS1p"
## [7] "nN"                    "C040"                  "quantitative_response"
```

Motivation

The problem

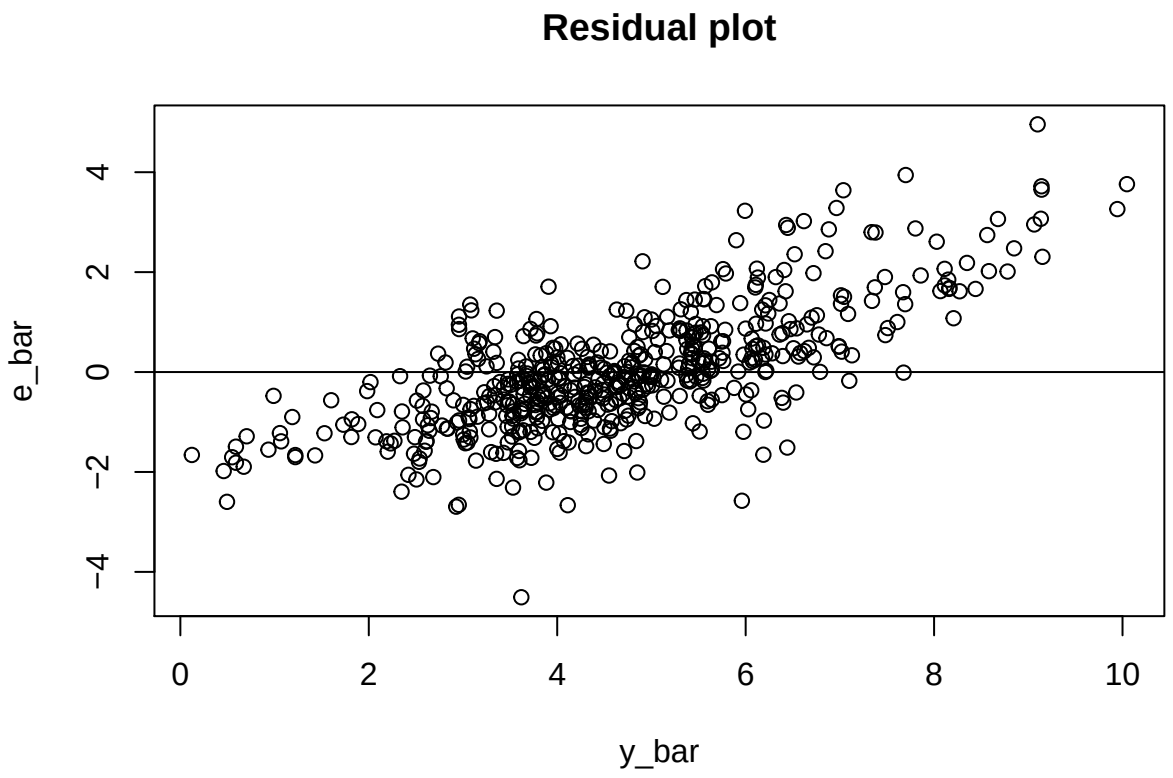
Modeling idea

We want our predictions of the linear model to be close to the linear model

```
#creating linear model
fullmodel=lm(quantitative_response~TSPA+Saacc+H050+MLOGP+RDCHI+GATS1p+nN+C040, data =QSAR)
summary(fullmodel)
```

```
##
## Call:
## lm(formula = quantitative_response ~ TSPA + Saacc + H050 + MLOGP +
##     RDCHI + GATS1p + nN + C040, data = QSAR)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5076 -0.7615 -0.1023  0.6109  4.9580
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.705317   0.244832  11.050  < 2e-16 ***
## TSPA         0.027341   0.002670  10.240  < 2e-16 ***
## Saacc       -0.015030   0.002094  -7.179 2.36e-12 ***
## H050         0.038974   0.059859   0.651 0.515266
## MLOGP        0.446237   0.063325   7.047 5.66e-12 ***
## RDCHI        0.514491   0.135630   3.793 0.000166 ***
## GATS1p      -0.570089   0.153962  -3.703 0.000235 ***
## nN          -0.232559   0.049547  -4.694 3.41e-06 ***
## C040        -0.030181   0.090962  -0.332 0.740175
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.203 on 536 degrees of freedom
## Multiple R-squared:  0.4866, Adjusted R-squared:  0.479
## F-statistic: 63.51 on 8 and 536 DF,  p-value: < 2.2e-16
```

```
plot(QSAR$quantitative_response,fullmodel$res, ylab="e_bar", xlab="y_bar", main="Residual plot")
abline(0,0)
```



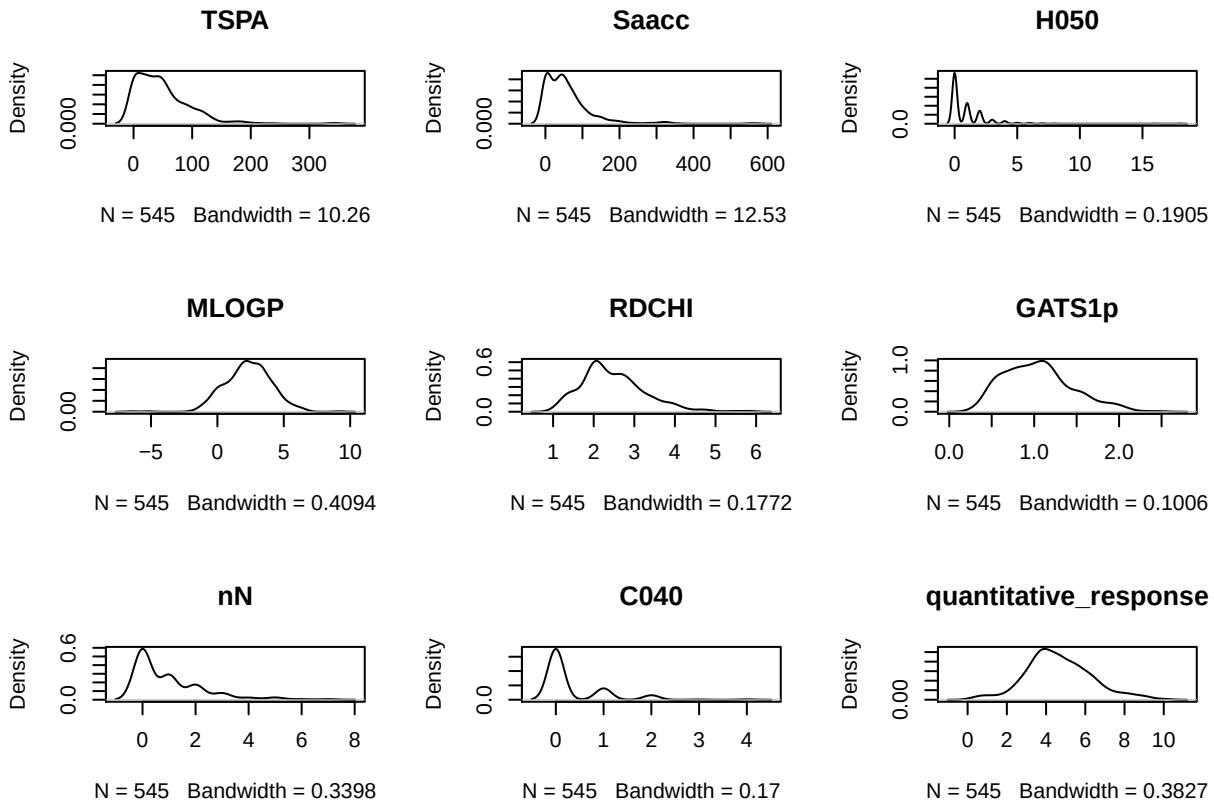
Illustrative figure

Visualizing the density plot of each variable

```
par(mfrow=c(3,3))
cols = colnames(QSAR)

for(col in cols)
{
plot(density(QSAR[col][,1]), main=col)

}
```



2) Description of the data and the analysis problem.

Provide information where the data was obtained, and if it has been previously used in some online case study and how your analysis differs from the existing analyses.

feature number	Feature name	Feature Description
1	TSPA	31.51109
2	SAACC	13.74853
3	H050	25.01827
4	MLOGP	11.12849
5	RDCHI	14.39153
6	GATS1p	14.39153
7	nN	14.39153
8	C040	14.39153
9	Quantitative Response	14.39153

3) Description of at least two models, for example:

a) non hierarchical(linear)

b) hierarchical,

4) Informative or weakly informative priors

and justification of their choices.

5) Stan code

a) non hierarchical(linear)

Stan Code

```
code <- file("QSARproject.stan")
writeLines(readLines(code))
```

```
## data {
## int < lower =1> N; // number of data points
## vector [N] qr;// observation year
## vector [N] TSPA;
## vector [N] Saacc;
## vector [N] H050;
## vector [N] MLOGP;
## vector [N] RDCHI;
## vector [N] GATS1p;
## vector [N] nN;
## vector [N] C040;
##
## }
## parameters {
##   real a;
##   real b;
##   real c;
##   real d;
##   real e;
##   real f;
##   real g;
##   real h;
##   real i;
##   real < lower =0> sigma ;
## }
## transformed parameters {
##   vector [N] mu = a + b * TSPA + c * Saacc + d *H050 + e *MLOGP+ f*RDCHI +g*GATS1p
##   + h*nN + i*C040;
## }
## model {
##
## //beta ~ normal(0,26.74);
## //alpha ~ normal(789,2523.813);
## //a~ gamma(1,1);
## //b~ gamma(1,1);
## //c~ gamma(1,1);
```

```
## //d~ gamma(1,1);
## //e~ gamma(1,1);
## //f~ gamma(1,1);
## //g~ gamma(1,1);
## //h~ gamma(1,1);
## //i~ gamma(1,1);
## sigma ~ normal(0,100);
## qr ~ normal (mu , sigma );
##
## }
## generated quantities {
## //real ypred ;
## vector[N] log_lik;
## vector[N] gen_lik;
## //Compute predictive distribution for the first machine
## //ypred = normal_rng ( mu , sigma);
## for (ind in 1:N)
## {
## log_lik[ind]= normal_lpdf(qr[ind] | mu[ind] ,sigma);
## gen_lik[ind]= normal_rng (mu[ind] ,sigma);
## };
##
## }
```

b) hierarchical

```
code_hierarchial <- file("hierarchial.stan")
writeLines(readLines(code_hierarchial))
```

```
## data {
## int < lower =1> N; // number of data points
## vector [N] qr;// observation year
## vector [N] TSPA;
## vector [N] Saacc;
## vector [N] HO50;
## vector [N] MLOGP;
## vector [N] RDCHI;
## vector [N] GATS1p;
## vector [N] nN;
## int CO40[N];
## int <lower = 0> nc; //number of carbon atoms //5 here
## //number of hierarchies
##
## }
## parameters {
##   vector[nc] a; // 5 values
##   vector[nc] b;
##   vector[nc] c;
##   vector[nc] d;
##   vector[nc] e;
##   vector[nc] f;
##   vector[nc] g;
##   vector[nc] h;
##   //vector[nc] i;
```

```
##   real < lower =0> sigma ;
##   //vector[nc] i;
##   //hyper parameters
##   real mu_a;real mu_b;real mu_c; real mu_d; real mu_e;real mu_f;
##   real mu_g;real mu_h; real mu_i;
##   real <lower= 0> tau_a;real <lower= 0> tau_b;real <lower= 0> tau_c;real <lower= 0> tau_d;
##   real<lower= 0> tau_e;real <lower= 0> tau_f;
##   real <lower= 0> tau_g;real <lower= 0> tau_h;real <lower= 0> tau_i;
##
## }
## transformed parameters {
##   vector [N] mu;
##
##   for(ind in 1:N)
##   {
##   mu[ind]= (a[C040[ind]+1]) + (b[C040[ind]+1]*TSPA[ind]) +
##   (c[C040[ind]+1] * Saacc[ind]) + (d[C040[ind]+1]*H050[ind]) + (e[C040[ind]+1] *MLOGP[ind])+
##   (f[C040[ind]+1]*RDCHI[ind]) + (g[C040[ind]+1]*GATS1p[ind]) + (h[C040[ind]+1]*nN[ind]) ;
##
##   };
##
## //+ i*C040;
## //+ i*C040 + d*H050
## }
## model {
##
## //beta ~ normal(0,26.74);
## //alpha ~ normal(789,2523.813);
## //a~ gamma(1,1);
## //b~ gamma(1,1);
## //c~ gamma(1,1);
## //d~ gamma(1,1);
## //e~ gamma(1,1);
## //f~ gamma(1,1);
## //g~ gamma(1,1);
## //h~ gamma(1,1);
## //i~ gamma(1,1);
##
## //hyperpriors
## mu_a~normal(0,0.1);
## tau_a~ normal(0,1);
##
## mu_b~normal(0,0.1);
## tau_b~ normal(0,1);
##
## mu_c~normal(0,0.1);
## tau_c~ normal(0,1);
##
## mu_d~normal(0,0.1);
## tau_d~ normal(0,1);
##
## mu_e~normal(0,0.1);
## tau_e~ normal(0,1);
##
```

```
## mu_f~normal(0,0.1);
## tau_f~ normal(0,1);
##
## mu_g~normal(0,0.1);
## tau_g~ normal(0,1);
##
## mu_h~normal(0,0.1);
## tau_h~ normal(0,1);
##
## mu_i~normal(0,0.1);
## tau_i~ normal(0,1);
##
##
## //priors
## a~ normal(mu_a,tau_a);
## b~ normal(mu_b,tau_b);
## c~ normal(mu_c,tau_c);
## d~ normal(mu_d,tau_d);
## e~ normal(mu_e,tau_e);
## f~ normal(mu_f,tau_f);
## g~ normal(mu_g,tau_g);
## h~ normal(mu_h,tau_h);
##
## sigma ~ normal(0,100);
## qr ~ normal (mu , sigma );
##
## }
## generated quantities {
## //real ypred ;
## vector[N] log_lik;
## vector[N] gen_lik;
## //Compute predictive distribution for the first machine
## //ypred = normal_rng ( mu , sigma);
## for (ind in 1:N)
## {
## log_lik[ind]= normal_lpdf(qr[ind] | mu[ind] ,sigma);
## gen_lik[ind]= normal_rng (mu[ind] ,sigma);
## };
##
## }
```

6) How to the Stan model was run

that is, what options were used. This is also more clear as combination of textual explanation and the actual code line.

a) non hierarchical(linear)

```
TSPA=QSAR$TSPA
Saacc=QSAR$Saacc
H050=QSAR$H050
MLOGP=QSAR$MLOGP
RDCHI=QSAR$RDCHI
```

```
GATS1p=QSAR$GATS1p
nN=QSAR$nN
C040=QSAR$C040
qr=QSAR$quantitative_response
n=dim(QSAR)[1]

qsar_data <-list(N=n,qr=qr,TSPA=TSPA,Saacc=Saacc,H050=H050,MLOGP=MLOGP,RDCHI=RDCHI,GATS1p=GATS1p,nN=nN,

model_simple <-stan(file = 'QSARproject.stan' , data = qsar_data, chains=4, iter=1000)
#print(model_simple)
params=extract(model_simple, permuted=FALSE, inc_warmup=TRUE)
```

b) hierarchical

```
TSPA=QSAR$TSPA
Saacc=QSAR$Saacc
H050=QSAR$H050
MLOGP=QSAR$MLOGP
RDCHI=QSAR$RDCHI
GATS1p=QSAR$GATS1p
nN=QSAR$nN
C040=as.integer(QSAR$C040)
qr=QSAR$quantitative_response
n=dim(QSAR)[1]
nc = length(unique(QSAR$C040))

qsar_data <-list(N=n,qr=qr,TSPA=TSPA,Saacc=Saacc,H050=H050,MLOGP=MLOGP,RDCHI=RDCHI,GATS1p=GATS1p,nN=nN,

hierarchial <-stan(file = 'hierarchial.stan' , data = qsar_data, chains=4, iter=1000)
```

```
## Warning: There were 257 divergent transitions after warmup. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
## Warning: The largest R-hat is 1.17, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#r-hat
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess
```

```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quanti
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

```
#print(hierarchial)
```

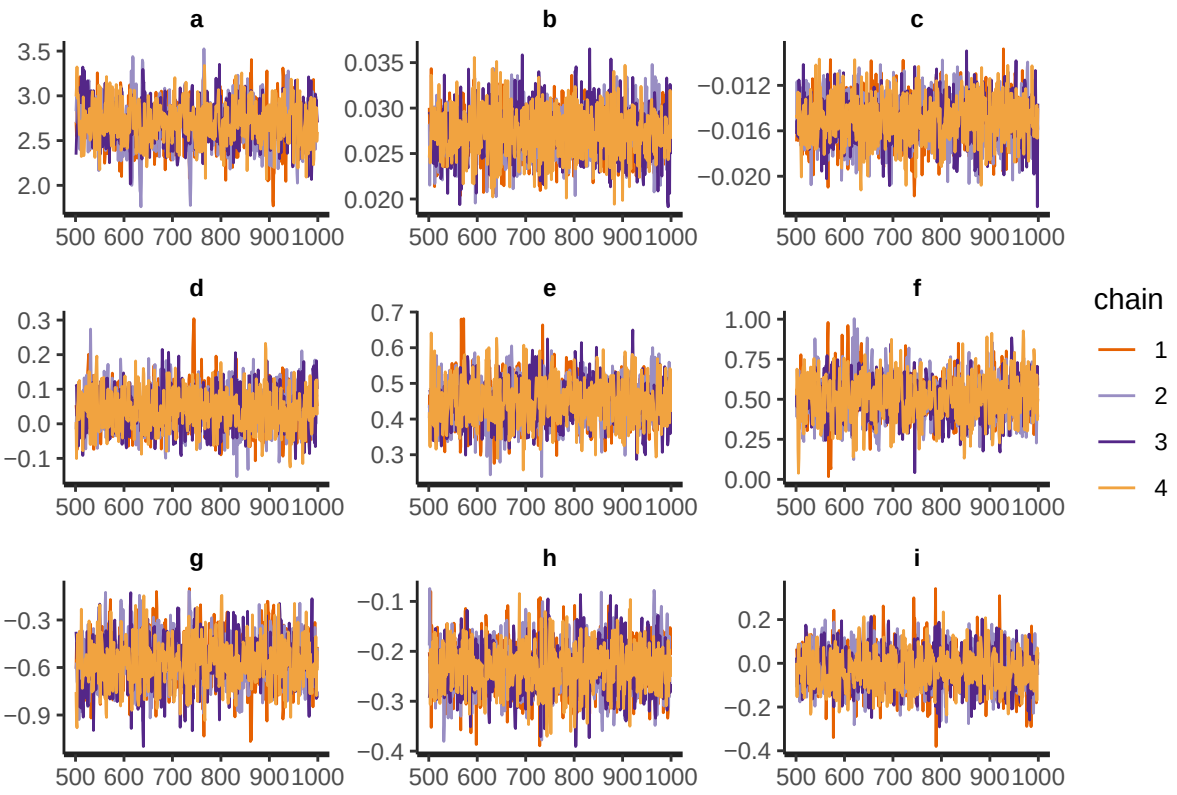
7) Convergence diagnostics

Here we will discuss 5 types of convergence tests - Traceplots - $\hat{R}$ - n_{eff} - Bulk ESS and Tail ESS - Divergences

Heirarchial

1) Traceplots

```
#mcmc_trace(as.array(model_simple), pars = c("a","b","c","d","e","f","g","h","i"), facet_args = list(nr
traceplot(model_simple, pars=c("a","b","c","d","e","f","g","h","i"))
```



2) $\hat{R}$

From printed output we can see that $\hat{R} < 1.01$ for all the parameters

3) n_{eff}

```
neff=summary(model_simple)$summary[,'n_eff']
val=neff/1000
#print(val)
which(val < 0.01)
```

```
## named integer(0)
```

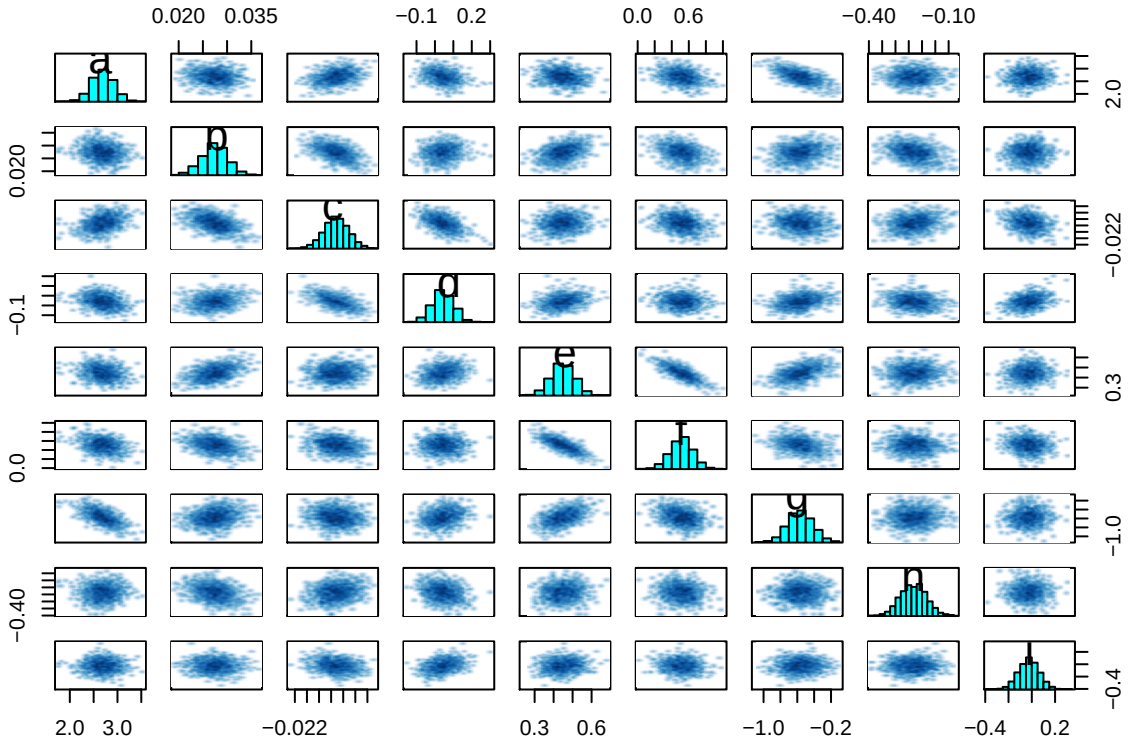
samples/ total iterations > 0.01, this means samples are not biased and true effect of sample size is not overestimated.

4) Bulk ESS and Tail ESS

BUlk ESS and tail ess over 100 for all the parameters

5) Divergences

```
pairs(model_simple,pars=c("a","b","c","d","e","f","g","h","i"))
```




```
get_num_divergent(model_simple)
```

```
## [1] 0
```

No divergences in the pairplot

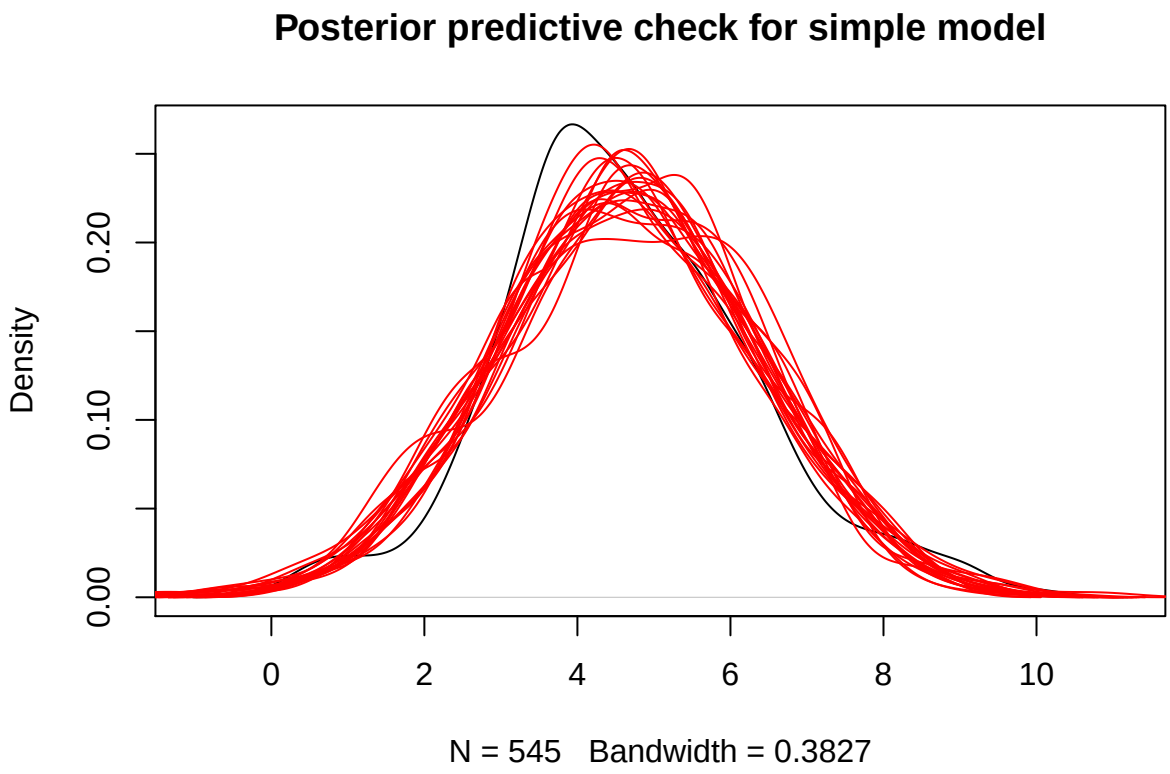
Heirarchial

8) Posterior predictive checks

Non-Heirarchial

and what was done to improve the model.

```
# instead of log, use rng
plot(density(QSAR$quantitative_response),main="Posterior predictive check for simple model")
params<-extract(model_simple)
for (ind in 1980:2000)
{
  lines(density(params$gen_lik[ind,]), col='red');
}
```



9) Model comparison

(e.g. with LOO-CV).

Hierarchial

```
loo_model_hierarchial <- loo(extract_log_lik(hierarchial))
```

```
## Warning: Relative effective sample sizes ('r_eff' argument) not specified.
## For models fit with MCMC, the reported PSIS effective sample sizes and
## MCSE estimates will be over-optimistic.
```

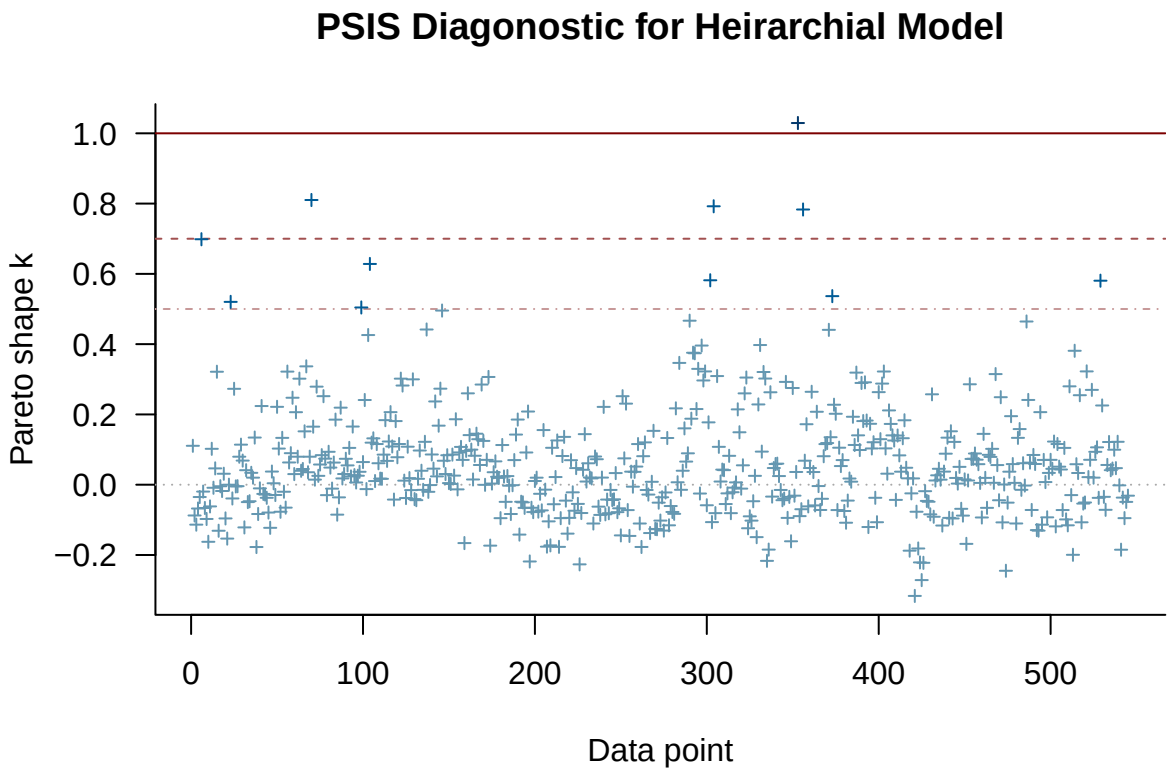
```
## Warning: Some Pareto k diagnostic values are too high. See help('pareto-k-diagnostic') for details.
```

```
loo_model_hierarchial
```

```
##
## Computed from 2000 by 545 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -870.1 21.8
## p_loo        27.3  3.0
## looic      1740.3 43.6
## ------
## Monte Carlo SE of elpd_loo is NA.
##
```

```
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     534   98.0%   387
##  (0.5, 0.7]   (ok)         7    1.3%   344
##    (0.7, 1]   (bad)        3    0.6%   176
##    (1, Inf)   (very bad)   1    0.2%   42
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_model_hierarchial, main = "PSIS Diagonostic for Heirarchial Model")
```



Non Heirarchial

```
loo_model_simple <- loo(extract_log_lik(model_simple))
```

```
## Warning: Relative effective sample sizes ('r_eff' argument) not specified.
## For models fit with MCMC, the reported PSIS effective sample sizes and
## MCSE estimates will be over-optimistic.
```

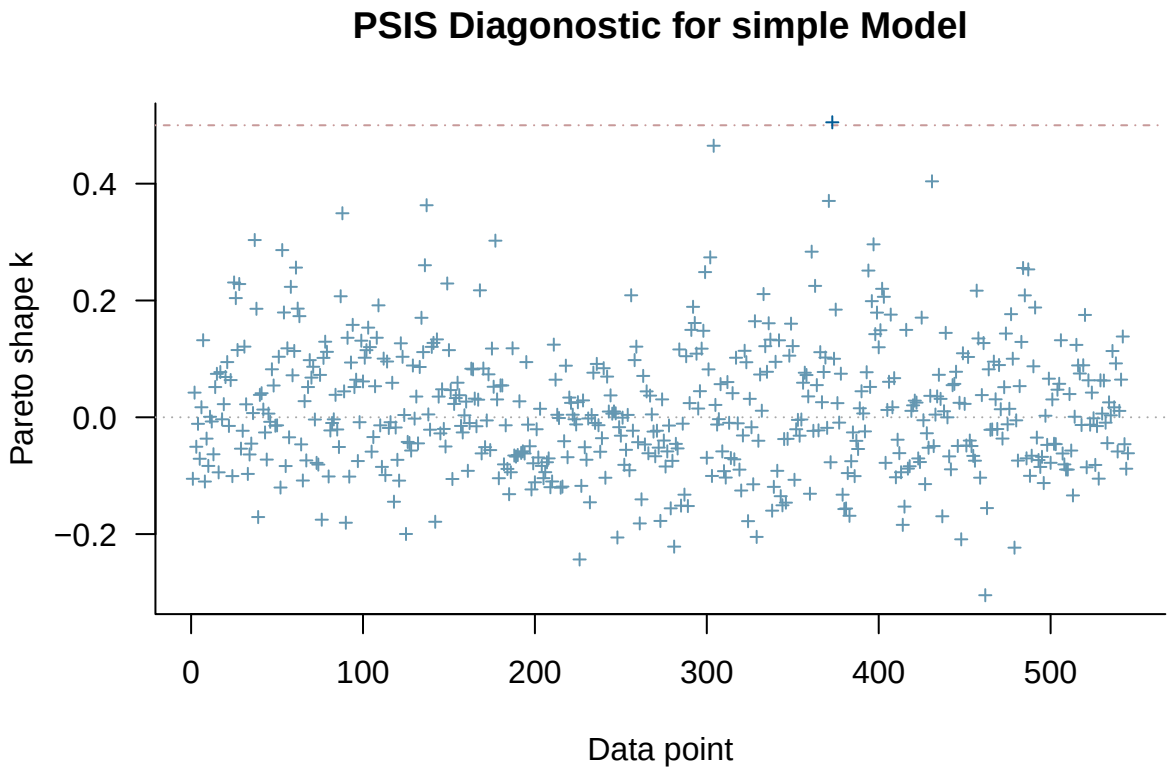
```
## Warning: Some Pareto k diagnostic values are slightly high. See help('pareto-k-diagnostic') for deta
```

```
print(loo_model_simple)
```

```
##
## Computed from 2000 by 545 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -881.0 21.3
## p_loo        12.5  1.5
```

```
## looic      1762.1 42.7
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     544   99.8%   693
##  (0.5, 0.7]   (ok)         1    0.2%   1076
##    (0.7, 1]   (bad)        0    0.0%   <NA>
##    (1, Inf)   (very bad)   0    0.0%   <NA>
##
## All Pareto k estimates are ok (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_model_simple, main = "PSIS Diagonostic for simple Model")
```



10) Predictive performance assessment

if applicable (e.g. classification accuracy) and evaluation of practical usefulness of the accuracy.

11) Sensitivity analysis

with respect to prior choices (i.e. checking whether the result changes a lot if prior is changed)

2 types of priors

12) Discussion of issues and potential improvements.

13) Conclusion

what was learned from the data analysis.

14) Self-reflection of what the group learned while making the project.