BDA - Assignment 8

Anonymous

Contents


```r
library (rstan)
library(loo)
rstan_options (auto_write = TRUE)
options(mc.cores = parallel::detectCores())
```

Exercise 1

Separate model

Although the assignment asked us to use uniform priors (i.e. the default in Stan), there was a separate message on behalf of the course staff that recommended using weakly informative priors instead. Therefore, I use $N \sim (100, 20^2)$ as the weakly informative prior for the means and $\tau \sim Cauchy(0, 10^2)$ as the weakly informative prior for the standard deviations.

```r
library('aaltobda')
data(factory)

factory_stan = "
data {
  int<lower=0> N;                    // number of data points
  int<lower=0> K;                    // number of groups
  int<lower=1,upper=K> x[N];         // group indicator
  vector[N] y;
}
parameters {
  vector[K] mu;                      // group means
  vector<lower=0>[K] sigma;          // group standard deviations
}
model {
  mu ~ normal(100, 20);              // weakly informative prior for mean
  sigma ~ cauchy(0, 10);             // weakly informative prior for st.deviation
  y ~ normal(mu[x], sigma[x]);       // ypred for each model with their own means and st.deviations
```

```
}
generated quantities {
  real ypred;
  vector[N] log_lik;
  ypred = normal_rng(mu[6], sigma[6]); // ypred for sixth model with its own mean and st.deviation
  for(i in 1:N) {
    log_lik[i] = normal_lpdf(y[i] | mu[x[i]], sigma[x[i]]); //log-likelihood
  }
}
"

factory_pooled = list(N = nrow(factory)*ncol(factory),
                      K = ncol(factory),
                      x = rep(1:6, nrow(factory)),      # [1,6] as a bracket
                      y = c(t(factory[,1:6])))

fit_sep = stan(model_code=factory_stan, data=factory_pooled, refresh=0)
monitor(fit_sep)
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##                 Q5   Q50   Q95  Mean   SD  Rhat Bulk_ESS Tail_ESS
## mu[1]         67.1  80.3  97.3  81.0  9.0  1.00     3106     2156
## mu[2]         95.4 105.7 114.9 105.6  6.1  1.00     3026     1900
## mu[3]         79.6  88.8 100.2  89.2  6.4  1.00     2620     1522
## mu[4]        103.4 111.1 117.6 110.8  4.7  1.00     2288     1451
## mu[5]         81.8  90.7 100.4  90.8  5.8  1.00     2831     2033
## mu[6]         75.7  88.2 103.0  88.6  8.7  1.00     3011     1726
## sigma[1]      12.4  19.8  36.9  21.8  8.8  1.00     3270     2593
## sigma[2]       7.6  12.2  23.6  13.5  5.5  1.00     3061     2296
## sigma[3]       8.3  13.1  24.6  14.5  6.0  1.00     2964     1961
## sigma[4]       5.1   8.5  17.1   9.4  4.2  1.00     2170     1741
## sigma[5]       6.9  11.4  22.0  12.7  5.3  1.00     3234     2236
## sigma[6]      12.2  19.1  36.0  21.0  8.3  1.00     2876     1936
## ypred         51.5  87.8 127.8  88.5 24.0  1.00     3383     2896
## log_lik[1]    -4.6  -4.0  -3.5  -4.0  0.3  1.00     2355     2396
## log_lik[2]    -4.9  -4.0  -3.4  -4.0  0.5  1.00     4119     2946
## log_lik[3]    -5.0  -4.1  -3.5  -4.1  0.5  1.00     4171     3699
## log_lik[4]    -4.3  -3.5  -2.9  -3.5  0.4  1.00     2925     2338
## log_lik[5]    -5.0  -4.0  -3.4  -4.1  0.5  1.00     4282     3303
## log_lik[6]    -7.1  -5.3  -4.4  -5.5  0.9  1.00     4847     3356
## log_lik[7]    -4.9  -4.2  -3.8  -4.3  0.4  1.00     3847     2945
## log_lik[8]    -4.3  -3.5  -3.0  -3.6  0.4  1.00     2324     1881
## log_lik[9]    -4.3  -3.6  -3.2  -3.7  0.3  1.00     2197     1848
## log_lik[10]   -4.6  -3.6  -3.0  -3.7  0.5  1.00     3377     1920
## log_lik[11]   -4.4  -3.6  -3.1  -3.7  0.4  1.00     2772     2258
## log_lik[12]   -4.6  -4.0  -3.5  -4.0  0.3  1.00     2032     1443
## log_lik[13]   -4.9  -4.2  -3.8  -4.3  0.4  1.00     3847     2945
## log_lik[14]   -4.5  -3.8  -3.2  -3.8  0.4  1.00     3147     2437
## log_lik[15]   -4.3  -3.6  -3.1  -3.7  0.4  1.00     2134     1624
## log_lik[16]   -4.1  -3.3  -2.8  -3.4  0.4  1.00     2055     1313
## log_lik[17]   -5.3  -4.1  -3.5  -4.2  0.6  1.00     4942     3611
## log_lik[18]   -5.1  -4.3  -3.8  -4.4  0.4  1.00     4425     3221
```

```
## log_lik[19]  -7.4  -5.5  -4.6  -5.7  0.9  1.00     6184     3641
## log_lik[20]  -4.2  -3.5  -3.0  -3.6  0.4  1.01     2232     1884
## log_lik[21]  -4.3  -3.6  -3.1  -3.6  0.4  1.00     2084     1533
## log_lik[22]  -5.0  -3.8  -3.2  -3.9  0.6  1.00     4286     3128
## log_lik[23]  -5.0  -4.0  -3.4  -4.1  0.5  1.00     4282     3303
## log_lik[24]  -4.8  -4.1  -3.6  -4.2  0.4  1.00     2507     1512
## log_lik[25]  -5.0  -4.2  -3.7  -4.2  0.4  1.00     3016     2578
## log_lik[26]  -6.8  -4.7  -3.9  -4.9  0.9  1.00     3963     2807
## log_lik[27]  -7.1  -5.0  -4.1  -5.2  1.0  1.00     4615     3165
## log_lik[28]  -4.1  -3.3  -2.8  -3.4  0.4  1.00     2055     1313
## log_lik[29]  -4.1  -3.5  -2.9  -3.5  0.4  1.00     2168     1928
## log_lik[30]  -4.9  -4.2  -3.7  -4.2  0.4  1.00     2940     2303
## lp__        -92.7 -86.8 -83.5 -87.3  2.8  1.00     1360     1715
##
## For each parameter, Bulk_ESS and Tail_ESS are crude measures of
## effective sample size for bulk and tail quantities respectively (an ESS > 100
## per chain is considered good), and Rhat is the potential scale reduction
## factor on rank normalized split chains (at convergence, Rhat <= 1.05).
```

Pooled model

For the pooled model, each machine j will have a common mean μ and standard deviation σ meaning that the parameter μ and σ are now single real valued parameters instead of vectors. Once again I apply $N \sim (100, 20^2)$ as the weakly informative prior for the means and $\tau \sim Cauchy(0, 10^2)$ as the weakly informative prior for the standard deviations.

```
library('aaltobda')
data(factory)

factory_stan = "
data {
  int<lower=0> N;                   // numebr of data points
  vector[N] y;
}
parameters {
  real mu;                          // Common mean
  real<lower=0> sigma;              // Common standard deviation
}
model {
  mu ~ normal(100, 20);             // weakly informative prior for mean
  sigma ~ cauchy(0, 10);            // weakly informative prior for st.deviation
  y ~ normal(mu, sigma);            // ypred for each model with their common means and st.deviations
}
generated quantities {
  real ypred;
  vector[N] log_lik;
  ypred = normal_rng(mu, sigma);    // ypred for sixth model with common mean and st.deviation
  for (i in 1:N) {
    log_lik[i] = normal_lpdf(y[i] | mu, sigma);
  }
}
"
```

```r
length = length(c(factory$V1, factory$V2, factory$V3, factory$V4, factory$V5, factory$V6))
factory_pooled = list(N=length,
                      y = c(factory$V1, factory$V2, factory$V3, factory$V4, factory$V5, factory$V6))

fit_pool = stan(model_code=factory_stan, data=factory_pooled, refresh=0)
monitor(fit_pool)
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##                 Q5    Q50   Q95   Mean   SD  Rhat Bulk_ESS Tail_ESS
## mu            87.7   93.2  98.6   93.2  3.3     1     2461     2069
## sigma         14.7   18.0  22.6   18.2  2.4     1     2170     2001
## ypred         62.2   92.5 123.6   92.7 18.8     1     3738     3704
## log_lik[1]    -4.2   -4.0  -3.8   -4.0  0.1     1     2345     1893
## log_lik[2]    -4.1   -3.8  -3.6   -3.8  0.1     1     1970     2015
## log_lik[3]    -4.1   -3.8  -3.6   -3.8  0.1     1     1970     2015
## log_lik[4]    -9.0   -7.3  -6.1   -7.3  0.9     1     2239     2203
## log_lik[5]    -5.5   -4.9  -4.5   -4.9  0.3     1     2519     2365
## log_lik[6]    -5.2   -4.7  -4.3   -4.7  0.3     1     2538     2141
## log_lik[7]    -4.5   -4.2  -4.0   -4.2  0.2     1     2703     2316
## log_lik[8]    -4.9   -4.5  -4.2   -4.5  0.2     1     2702     2226
## log_lik[9]    -4.3   -4.0  -3.8   -4.0  0.1     1     2367     2030
## log_lik[10]   -4.1   -3.9  -3.7   -3.9  0.1     1     2119     1825
## log_lik[11]   -4.1   -3.9  -3.7   -3.9  0.1     1     2134     2064
## log_lik[12]   -4.1   -3.8  -3.6   -3.8  0.1     1     1960     2018
## log_lik[13]   -4.1   -3.8  -3.6   -3.8  0.1     1     1970     2015
## log_lik[14]   -4.1   -3.9  -3.7   -3.9  0.1     1     2156     1802
## log_lik[15]   -5.5   -4.9  -4.5   -4.9  0.3     1     2519     2365
## log_lik[16]   -4.3   -4.0  -3.8   -4.1  0.1     1     2456     2048
## log_lik[17]   -5.4   -4.9  -4.5   -4.9  0.3     1     2424     2092
## log_lik[18]   -5.1   -4.6  -4.3   -4.7  0.3     1     2615     2087
## log_lik[19]   -4.2   -3.9  -3.7   -4.0  0.1     1     2201     2050
## log_lik[20]   -5.1   -4.6  -4.3   -4.7  0.3     1     2615     2087
## log_lik[21]   -4.4   -4.1  -3.9   -4.1  0.2     1     2661     2077
## log_lik[22]   -4.1   -3.8  -3.6   -3.9  0.1     1     1967     1998
## log_lik[23]   -4.2   -4.0  -3.8   -4.0  0.1     1     2283     1944
## log_lik[24]   -4.4   -4.1  -3.9   -4.1  0.2     1     2661     2077
## log_lik[25]   -4.1   -3.8  -3.6   -3.8  0.1     1     1970     2015
## log_lik[26]   -6.9   -5.8  -5.1   -5.9  0.5     1     2282     1902
## log_lik[27]   -4.1   -3.8  -3.6   -3.8  0.1     1     1970     2015
## log_lik[28]   -4.3   -4.0  -3.8   -4.0  0.1     1     2367     2030
## log_lik[29]   -4.5   -4.2  -4.0   -4.2  0.2     1     2691     2254
## log_lik[30]   -4.1   -3.9  -3.7   -3.9  0.1     1     2067     2142
## lp__        -102.9 -100.5 -99.9 -100.8  1.0     1     1612     1868
##
## For each parameter, Bulk_ESS and Tail_ESS are crude measures of
## effective sample size for bulk and tail quantities respectively (an ESS > 100
## per chain is considered good), and Rhat is the potential scale reduction
## factor on rank normalized split chains (at convergence, Rhat <= 1.05).
```

Hierarchical model

As informed by the course staff, applying hyperprios is preferred over uniform priors. Therefore, I will one again use $N \sim (100, 20^2)$ as the weakly informative prior for the means and $\tau \sim Cauchy(0, 10^2)$ as the weakly informative prior for the standard deviations.

```
library('aaltobda')
data(factory)

factory_stan = "
data {
  int<lower=0> N;                    // number of data points
  int<lower=0> K;                    // number of groups
  int<lower=1,upper=K> x[N];         // group indicator
  vector[N] y;
}
parameters {
  real mu0;                          // prior mean
  real<lower=0> sigma0;              // prior std
  vector[K] mu;                      // group means
  real<lower=0> sigma;               // common st.deviation
}
model {
  mu0 ~ normal(100,20);              // weakly informative prior for hierarchical mean (hyperprior)
  sigma0 ~ cauchy(0, 10);            // weakly informative prior for hierarchical sigma (hyperprior)
  mu ~ normal(mu0, sigma0);          // weakly informative prior for mean
  sigma ~ cauchy(0,10);              // weakly informative prior for st.deviation
  y ~ normal(mu[x], sigma);
}
generated quantities {
  real ypred;
  real mu7;
  vector[N] log_lik;
  // ypred for sixth model with individual means, common st.deviation and hyperprios
  ypred = normal_rng(mu[6], sigma);
  // posterior od the seventh machine with individual means, common st.deviation and hyperprios
  mu7 = normal_rng(mu0, sigma0);
  for (i in 1:N)
    log_lik[i] = normal_lpdf(y[i] | mu[x[i]], sigma); //log-likelihood
}
"

factory_pooled = list(N = nrow(factory)*ncol(factory),
                      K = ncol(factory),
                      x = rep(1:6, nrow(factory)),     # [1,6] as a bracket
                      y = c(t(factory[,1:6])))

fit_hier = stan(model_code=factory_stan, data=factory_pooled, refresh=0)
```

```
## Warning: There were 6 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help.
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

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

```r
monitor(fit_hier)
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##                   Q5    Q50    Q95   Mean   SD  Rhat Bulk_ESS Tail_ESS
## mu0             84.4   93.4  102.6   93.4  5.6     1     2488     2055
## sigma0           4.8   11.0   21.7   11.9  5.5     1     1052      800
## mu[1]           70.2   81.4   92.2   81.3  6.8     1     2014     1247
## mu[2]           92.8  102.5  112.4  102.4  6.0     1     2537     2751
## mu[3]           80.2   89.5   98.9   89.5  5.6     1     3125     2875
## mu[4]           95.2  106.2  117.3  106.2  6.8     1     2357     1919
## mu[5]           81.9   91.2  100.1   91.1  5.6     1     3051     2718
## mu[6]           78.5   88.3   98.1   88.3  6.0     1     2953     2359
## sigma           11.8   14.6   18.9   14.9  2.2     1     2417     2440
## ypred           62.0   88.3  115.1   88.4 16.3     1     3950     3862
## mu7             71.2   93.3  115.1   93.3 13.7     1     3193     3309
## log_lik[1]      -4.1   -3.7   -3.4   -3.7  0.2     1     2450     2371
## log_lik[2]      -4.9   -4.1   -3.6   -4.2  0.4     1     2820     3243
## log_lik[3]      -4.7   -3.9   -3.6   -4.0  0.3     1     3684     3301
## log_lik[4]      -4.1   -3.7   -3.4   -3.7  0.2     1     2156     2712
## log_lik[5]      -4.6   -4.0   -3.6   -4.0  0.3     1     3119     3039
## log_lik[6]      -7.8   -5.9   -4.7   -6.0  1.0     1     4158     2621
## log_lik[7]      -4.8   -3.9   -3.6   -4.0  0.4     1     3675     3245
## log_lik[8]      -4.2   -3.7   -3.5   -3.8  0.2     1     2205     2792
## log_lik[9]      -4.0   -3.7   -3.4   -3.7  0.2     1     2513     2814
## log_lik[10]     -4.9   -4.0   -3.5   -4.1  0.4     1     2210     3211
## log_lik[11]     -4.2   -3.7   -3.5   -3.8  0.2     1     3074     2762
## log_lik[12]     -4.1   -3.7   -3.4   -3.7  0.2     1     2762     3170
## log_lik[13]     -4.8   -3.9   -3.6   -4.0  0.4     1     3675     3245
## log_lik[14]     -4.6   -3.9   -3.5   -4.0  0.3     1     2591     3076
## log_lik[15]     -4.0   -3.7   -3.4   -3.7  0.2     1     2312     2813
## log_lik[16]     -4.6   -3.9   -3.5   -3.9  0.3     1     1952     2800
## log_lik[17]     -4.7   -4.0   -3.6   -4.0  0.3     1     3506     3008
## log_lik[18]     -5.2   -4.2   -3.7   -4.3  0.5     1     3593     3118
## log_lik[19]     -8.8   -6.5   -4.9   -6.6  1.2     1     3441     1581
## log_lik[20]     -4.0   -3.7   -3.4   -3.7  0.2     1     2004     2550
## log_lik[21]     -4.1   -3.7   -3.4   -3.7  0.2     1     2092     2234
## log_lik[22]     -4.2   -3.7   -3.5   -3.8  0.2     1     3089     2881
## log_lik[23]     -4.6   -4.0   -3.6   -4.0  0.3     1     3119     3039
## log_lik[24]     -4.6   -3.9   -3.5   -4.0  0.3     1     2967     2631
## log_lik[25]     -5.0   -4.1   -3.6   -4.2  0.5     1     2209     1478
## log_lik[26]     -5.2   -4.2   -3.7   -4.3  0.5     1     2966     3639
## log_lik[27]     -6.0   -4.8   -4.0   -4.9  0.6     1     3840     3192
## log_lik[28]     -4.6   -3.9   -3.5   -3.9  0.3     1     1952     2800
## log_lik[29]     -4.0   -3.7   -3.4   -3.7  0.2     1     2320     2611
## log_lik[30]     -4.7   -4.0   -3.6   -4.0  0.4     1     3911     3637
## lp__          -114.6 -109.9 -107.1 -110.3  2.3     1     1480     2595
##
## For each parameter, Bulk_ESS and Tail_ESS are crude measures of
## effective sample size for bulk and tail quantities respectively (an ESS > 100
## per chain is considered good), and Rhat is the potential scale reduction
## factor on rank normalized split chains (at convergence, Rhat <= 1.05).
```

Exercise 2

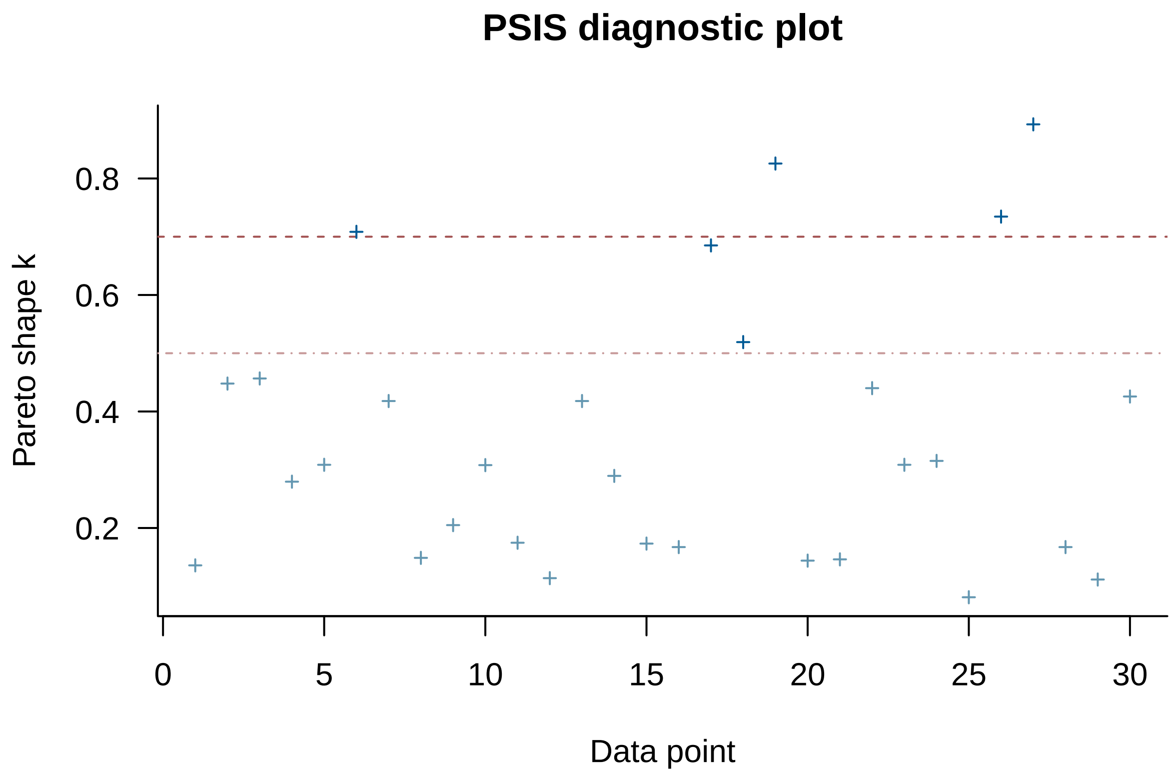
Note 1: As I use weakly informative prios, the resulting values for $\hat{k}$ and $\widehat{epld}_{loo}$ can vary from the example solutions

Note 2: To highlight the answers, I report some of the highlighted values as approximations because the knit to pdf conversion always changes the results a bit as the code chunks get re-run. The strictly correct values are, however, displayed in the print tables.

```
loo_sep = loo(fit_sep, cores=2)
print(loo_sep)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -128.4 4.4
## p_loo         9.4 1.9
## looic       256.8 8.8
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     24    80.0%   1417
##  (0.5, 0.7]   (ok)        2     6.7%   992
##    (0.7, 1]   (bad)       4    13.3%   57
##    (1, Inf)   (very bad)  0     0.0%   <NA>
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_sep)
```

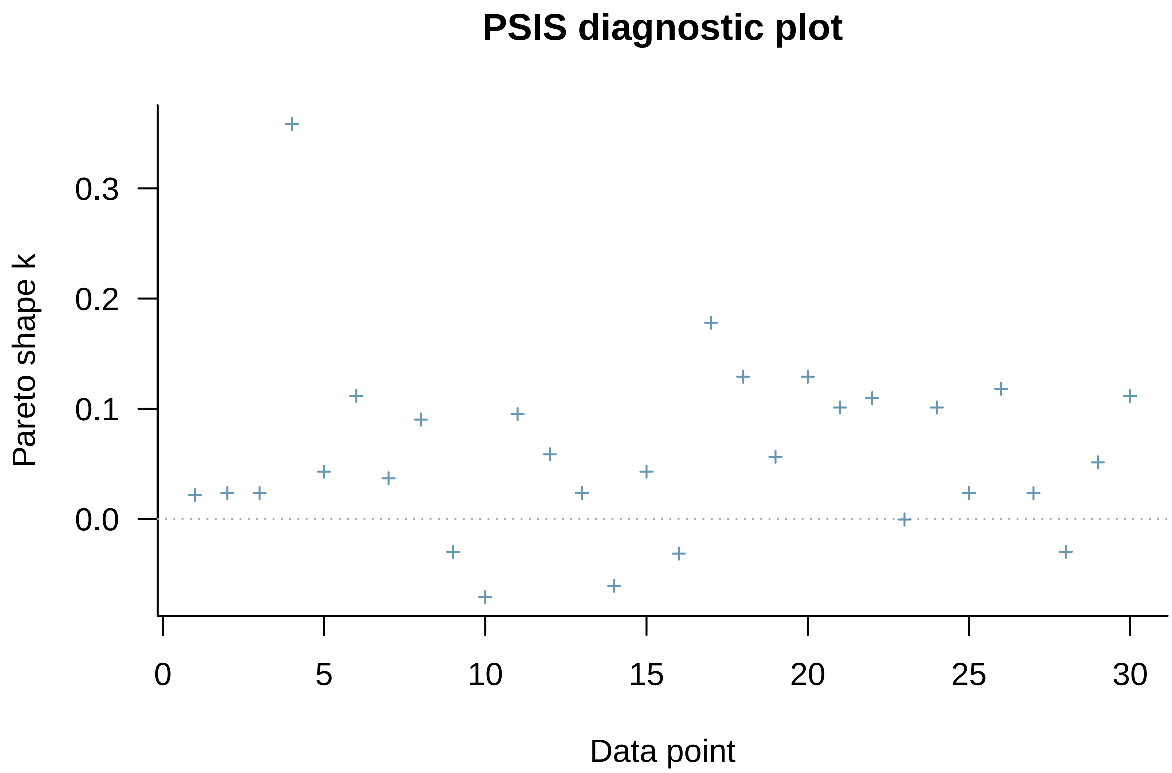


- $PSIS - LOO\ \widehat{elpd}_{loo}$ value for the separate model ≈ -128.5
- For the separate model $\hat{k} > 0.7$

```
loo_pool = loo(fit_pool, cores=2)
print(loo_pool)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -130.9 4.5
## p_loo         2.0 0.8
## looic       261.8 9.1
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_pool)
```

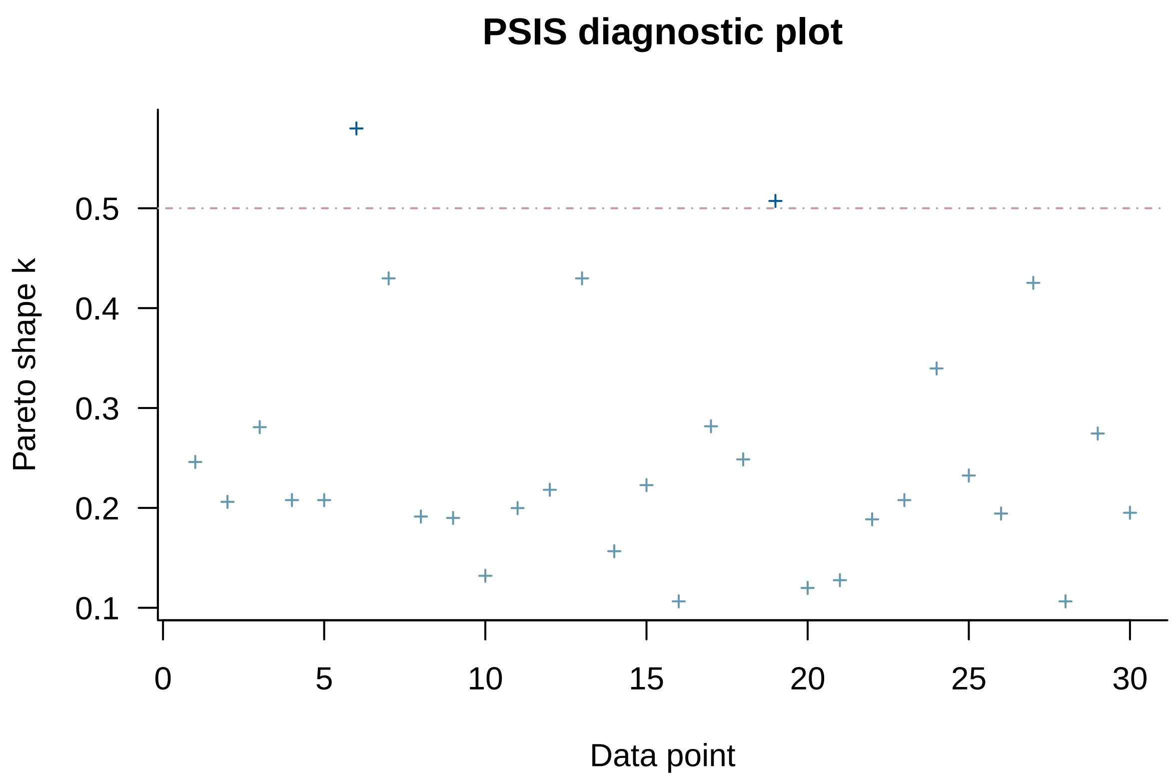


- $PSIS - LOO\ \widehat{elpd}_{loo}$ value for the pooled model ≈ -131.0
- For the pooled model $\hat{k} < 0.5$

```
loo_hier = loo(fit_hier, cores=2)
print(loo_hier)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -126.9 4.6
## p_loo         5.7 1.7
## looic       253.8 9.2
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     28    93.3%   1600
##  (0.5, 0.7]   (ok)        2     6.7%   332
##    (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.
```

```r
plot(loo_hier)
```



- $PSIS - LOO\ \widehat{elpd}_{loo}$ value for the hierarchical model ≈ -127.0
- For the hierarchical model $\hat{k} < 0.7$

Exercise 3

Compute Effective number of parameters for the separate model

Note: As I use weakly informative prios, the resulting values for $\hat{p}_{loo}$ can vary from the example solutions

```r
loo_sep = loo(fit_sep, r_eff=TRUE, cores=2)
print(loo_sep)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -128.4 4.4
## p_loo         9.4 1.9
## looic       256.8 8.8
## ------
```

```
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     24    80.0%   1417
##  (0.5, 0.7]   (ok)        2     6.7%   992
##    (0.7, 1]   (bad)       4    13.3%   57
##    (1, Inf)   (very bad)  0     0.0%   <NA>
## See help('pareto-k-diagnostic') for details.
```

- As we see from results, the $\hat{p}_{loo}$ for the separated model is ≈ 9.6

```
loo_pool = loo(fit_pool, cores=2)
print(loo_pool)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -130.9 4.5
## p_loo         2.0 0.8
## looic       261.8 9.1
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

- As we see from results, the $\hat{p}_{loo}$ for the pooled model is ≈ 2.1

```
loo_hier = loo(fit_hier, cores=2)
print(loo_hier)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -126.9 4.6
## p_loo         5.7 1.7
## looic       253.8 9.2
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     28    93.3%   1600
##  (0.5, 0.7]   (ok)        2     6.7%   332
##    (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.
```

- As we see from results, the $\hat{p}_{loo}$ for the hierarchical model is ≈ 5.9

Exercise 4

According to Vehtari et al (2016) The estimated shape parameter $\hat{k}$ of the generalized Pareto distribution can be used to assess the reliability of the estimate:

1. If $\hat{k} < \frac{1}{2}$, the variance of the raw importance ratios is finite, and the estimate converges quickly.
2. If $\frac{1}{2} \leqslant \hat{k} < 1$, the variance of the raw importance ratios is infinite but the mean exists, and the convergence of the estimate is slower. If $\frac{1}{2} \leqslant \hat{k} < 0.7$ then we observe practically useful convergence rates and Monte Carlo error estimates, however, if $\hat{k} > 0.7$ we observe impractical convergence rates and unreliable Monte Carlo error estimates and the estimate may be biased (optimistic)
3. If $\hat{k} \geqslant 1$, the variance and the mean of the raw ratios distribution do not exist. The convergence rate is close to zero and the estimate is likely to be biased (optimistic)

As seen from the separate model results, the model is not reliable since $0.7 < \hat{k} \leqslant 1$ indicating that the convergence rate is close to zero and that he model is not reliable.

As seen from the pooled model results, all pareto $\hat{k}$ estimates are good ($\hat{k} < 0.5$), indicating it converges quickly and that the model is reliable.

As seen from the hierarchical model results, all pareto $\hat{k}$ estimates are good ($\hat{k} < 0.7$). In more detail, approximately 93% of the data points have $\hat{\hat{k}}$ values $\hat{k} < \frac{1}{2}$ and 7% have $\frac{1}{2} \leqslant \hat{k} < 0.7$. This indicates that the model converges fairly quickly and that the model is reliable.

(Visualizations for the $\hat{k}$ values in section "Exercise 2")

Exercise 5

We can observe from the results that the $PSIS - LOO\ \widehat{elpd}_{loo}$ values differ a bit between the three different models. According to these values, the hierachical model has the highest $PSIS - LOO\ \widehat{elpd}_{loo}$ value at $\approx$ -127.0, indicating that this model should be selected, as higher expected log pointwise predictive density corresponds to smaller predictive error

References

Vehtari, A., Gelman, A., and Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. Statistics and Computing. 27(5), 1413–1432. doi:10.1007/s11222-016-9696-4. arXiv preprint: http://arxiv.org/abs/1507.04544/