Sesión 8: Modelo jerárquico con STAN en R

Aplicaciones en Computación Estadística

Natalie Julian - www.nataliejulian.com

Estadística UC y Data Scientist en Zippedi Inc.

Experimentos de pruebas educativas en escuelas

Ejemplo de *Bayesian Data Analysis Gelman A, Carlin J, Stern H, Rubin D (2003)*

Se realizó un estudio para analizar los efectos de los programas especiales de entrenamiento en los puntajes de las pruebas SAT-V (Prueba de Aptitud Escolar-Verbal). La variable de resultado en cada estudio fue el puntaje en una administración especial del SAT-V, una prueba estandarizada de opción múltiple administrada por el Servicio de Pruebas Educativas y utilizada para ayudar a las universidades a tomar decisiones de admisión.

Experimentos de pruebas educativas en escuelas

Considere el siguiente modelo jerárquico:

$$y_j \sim N(\theta_j, \sigma_j)$$

$$\theta_j \sim N(\mu, \tau)$$

$$\mu \sim N(0, 20)$$

$$\tau \sim \chi^2(8)$$

Según el modelo, los puntajes en las escuelas siguen una distribución normal cuya media es θ_j, y desviación estándar es σ_j (conocida a partir de los datos). Mientras que θ_j, sigue una distribución normal con parámetros μ (media global de los puntajes) y τ variabilidad entre las escuelas.

Implementación con STAN en R usando la librería `rstan`

Instalación

Primero es necesario instalar `rstan`:

```
install.packages("rstan", repos = "https://cloud.r-project.org/",
dependencies = TRUE)

library(rstan)
```

Modelo

```
scode<-"data {
int<lower=0> J; //  Numero de escuelas
real y[J]; // Puntaje SAT estimado
real<lower=0> sigma[J]; // error estandar por escuela
}
parameters {
real theta[J]; // Puntaje medio por escuela
real mu; // Puntaje medio global entre todas las escuelas
real<lower=0> tau; // Variabilidad de puntajes SAT entre escuelas
}
model {
mu~normal(0, 20);  //Priori para mu: normal(0,20)
tau~chi_square(8); //Priori para tau: chi square (8df)
theta ~ normal(mu, tau); //Modelo jerárquico
y ~ normal(theta, sigma);

}
generated quantities { //Aquí extraemos la logverosimilitud
  vector[J] log_lik;
  for (j in 1:J) {
    log_lik[j] = normal_lpdf(y[j] | theta[j], sigma[j]);
  }
}
"
```

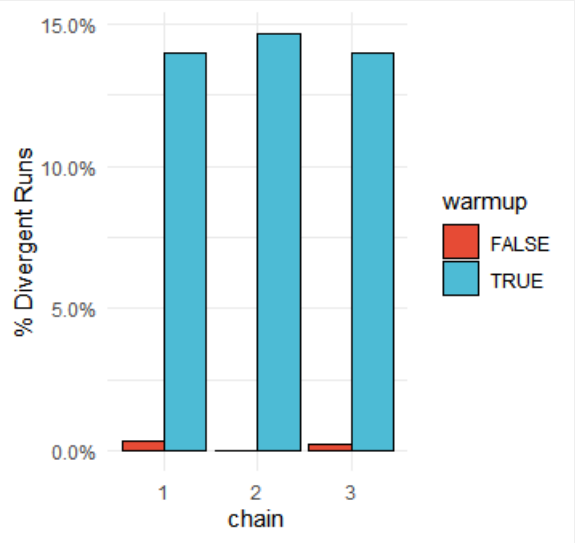
stan()

```
fit1 <- stan(model_code=scode,
             data=schools_data,
             warmup=150, #quema
             iter=1000, #largo de la cadena
             chains=3) #cadenas

Inference for Stan model: 4748570b0b5d01cf10671ff4249d188d.
3 chains, each with iter=1000; warmup=150; thin=1;
post-warmup draws per chain=850, total post-warmup draws=2550.

              mean se_mean   sd   2.5%    25%    50%    75% 97.5% n_eff Rhat
theta[1]     10.78    0.26 7.73  -2.70   5.58  10.33  15.24 27.32   888 1.00
theta[2]      7.76    0.21 6.31  -4.56   3.56   7.82  11.87 20.24   884 1.00
theta[3]      6.26    0.23 7.56  -8.96   1.86   6.53  11.29 20.18  1099 1.00
theta[4]      7.58    0.21 6.61  -5.30   3.30   7.67  11.94 20.63   996 1.00
theta[5]      5.28    0.21 6.20  -7.90   1.45   5.48   9.34 16.96   876 1.00
theta[6]      6.11    0.23 6.67  -7.76   1.84   6.31  10.59 18.64   834 1.00
theta[7]     10.31    0.20 6.84  -2.67   5.78  10.16  14.53 24.69  1197 1.00
theta[8]      8.22    0.23 7.61  -6.61   3.51   8.30  12.84 24.14  1081 1.00
mu            7.72    0.20 4.91  -2.11   4.41   7.72  10.99 17.30   591 1.00
tau           6.14    0.18 2.98   1.77   3.96   5.63   7.87 13.21   290 1.01
log_lik[1]   -4.45    0.02 0.60  -5.80  -4.79  -4.36  -4.01 -3.64   931 1.00
log_lik[2]   -3.43    0.01 0.28  -4.29  -3.49  -3.33  -3.26 -3.24   898 1.00
log_lik[3]   -3.97    0.01 0.29  -4.71  -4.09  -3.88  -3.76 -3.71  1059 1.00
log_lik[4]   -3.50    0.01 0.27  -4.26  -3.55  -3.40  -3.33 -3.32  1094 1.00
log_lik[5]   -3.58    0.02 0.50  -4.91  -3.75  -3.40  -3.22 -3.16   997 1.00
log_lik[6]   -3.64    0.01 0.36  -4.60  -3.75  -3.50  -3.39 -3.35   805 1.00
log_lik[7]   -3.75    0.02 0.55  -5.25  -3.96  -3.57  -3.35 -3.26  1088 1.00
log_lik[8]   -3.91    0.01 0.18  -4.41  -3.94  -3.84  -3.80 -3.79   764 1.00
```

Convergencia

```
check_divergences(fit1) #Indica si hubo divergencia en algún caso
24 of 2550 iterations ended with a divergence (0.941176470588235%).
Try increasing 'adapt_delta' to remove the divergences.

library(dplyr)
library(purrr)
library(ggsci)

mack_diagnostics <- rstan::get_sampler_params(fit1) %>%
  set_names(1:3) %>%
  map_df(as_tibble,.id = 'chain') %>%
  group_by(chain) %>%
  mutate(iteration = 1:length(chain)) %>%
  mutate(warmup = iteration <= 150) #Se indica cuantas se quemaron

mack_diagnostics %>%
  group_by(warmup, chain) %>%
  summarise(percent_divergent = mean(divergent__ >0)) %>%
  ggplot() +
  geom_col(aes(chain, percent_divergent, fill = warmup), position = 'dodge', color = 'black') +
  scale_y_continuous(labels = scales::percent, name = "% Divergent Runs")  +
  scale_fill_npg()+theme_minimal()
```

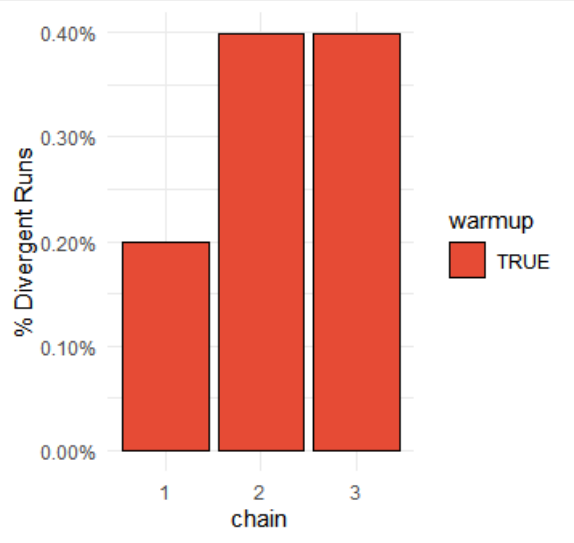
Convergencia



Aumentamos la cadena

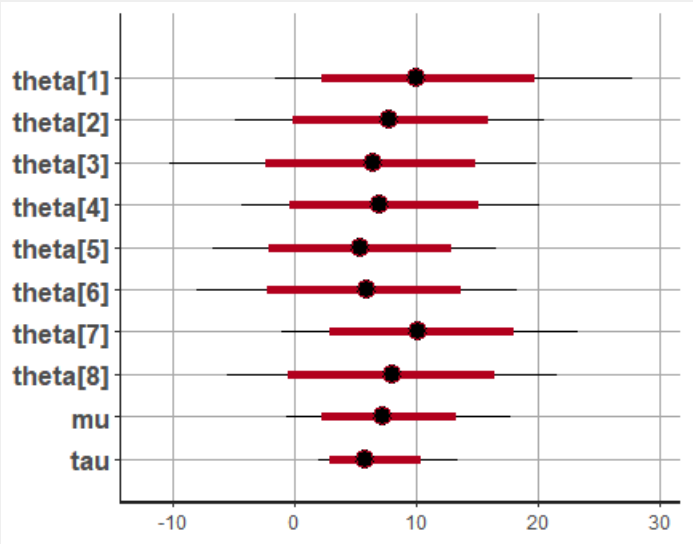
```
fit2 <- stan(model_code=scode,
             data=schools_data,
             warmup=8000, #quema
             iter=15000, #largo de la cadena
             thin=30, #saltos
             chains=3, #cadenas
             control = list(adapt_delta = 0.99)) #Tasa aceptación

Inference for Stan model: 4748570b0b5d01cf10671ff4249d188d.
3 chains, each with iter=15000; warmup=8000; thin=30;
post-warmup draws per chain=234, total post-warmup draws=702.

              mean se_mean   sd   2.5%    25%    50%    75% 97.5% n_eff Rhat
theta[1]     10.63    0.30 7.17  -1.55   5.66  10.03  15.13 27.77   561 1.01
theta[2]      7.90    0.23 6.29  -4.82   3.93   7.74  12.09 20.45   750 1.00
theta[3]      6.18    0.27 7.06 -10.28   1.88   6.32  11.05 19.22   709 1.00
theta[4]      7.39    0.25 6.27  -4.29   3.30   7.01  11.61 20.07   650 1.00
theta[5]      5.28    0.23 5.98  -6.77   1.72   5.43   9.09 16.53   668 1.00
theta[6]      5.84    0.27 6.59  -8.04   1.86   5.86  10.38 18.28   587 1.00
theta[7]     10.37    0.23 6.30  -1.01   6.05  10.11  14.49 23.29   739 1.00
theta[8]      8.00    0.28 6.94  -5.44   3.66   8.06  12.07 21.54   614 1.00
mu            7.64    0.18 4.63  -0.59   4.78   7.21  10.63 17.48   645 1.00
tau           6.30    0.12 3.05   2.02   4.15   5.76   7.93 13.48   651 1.00
log_lik[1]   -4.45    0.02 0.54  -5.64  -4.78  -4.38  -4.02 -3.64   641 1.00
log_lik[2]   -3.43    0.01 0.29  -4.22  -3.48  -3.32  -3.26 -3.24   596 1.00
log_lik[3]   -3.95    0.01 0.27  -4.62  -4.07  -3.87  -3.76 -3.71   687 1.00
log_lik[4]   -3.48    0.01 0.28  -4.14  -3.53  -3.39  -3.33 -3.32   577 1.00
log_lik[5]   -3.56    0.02 0.48  -4.84  -3.71  -3.38  -3.23 -3.16   716 1.00
log_lik[6]   -3.62    0.01 0.34  -4.55  -3.75  -3.50  -3.39 -3.35   659 1.00
log_lik[7]   -3.71    0.02 0.47  -4.93  -3.93  -3.57  -3.35 -3.26   736 1.00
log_lik[8]   -3.89    0.01 0.15  -4.31  -3.92  -3.84  -3.80 -3.79   628 1.00
```

Convergencia



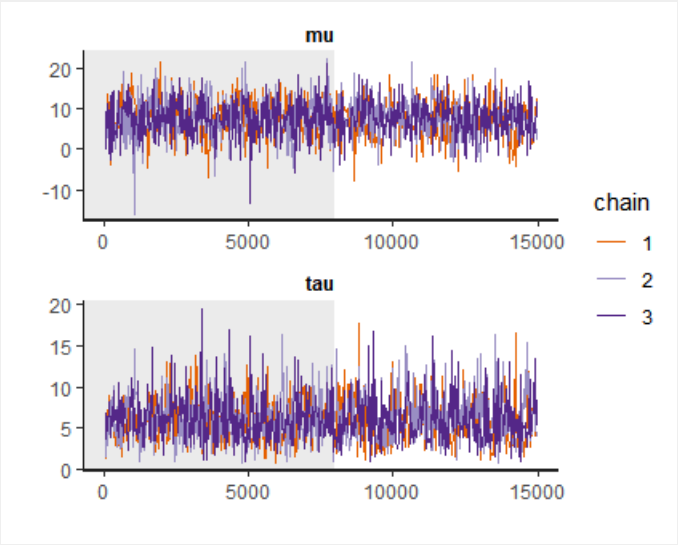
Gráficos por default

```
plot(fit2)
```



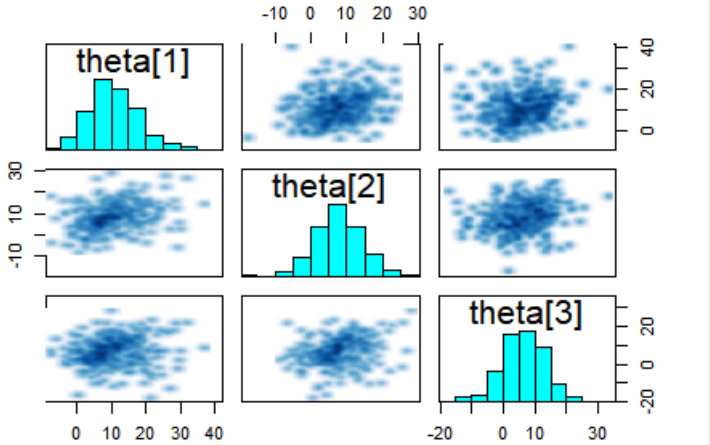
Gráficos por default

```
traceplot(fit2,pars=c("mu","tau"), inc_warmup=TRUE, nrow=3)
```



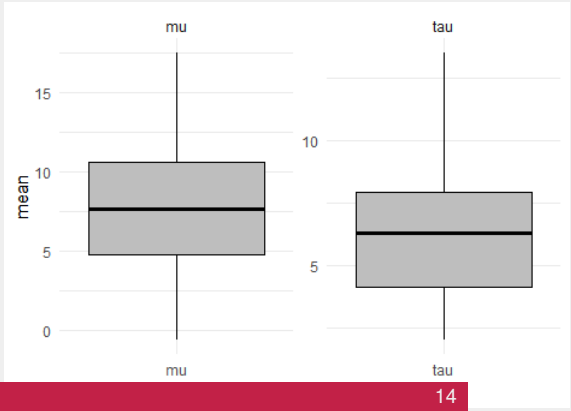
Gráficos por default

```
pairs(fit2,pars=c("theta[1]","theta[2]","theta[3]"))
```



Más gráficos

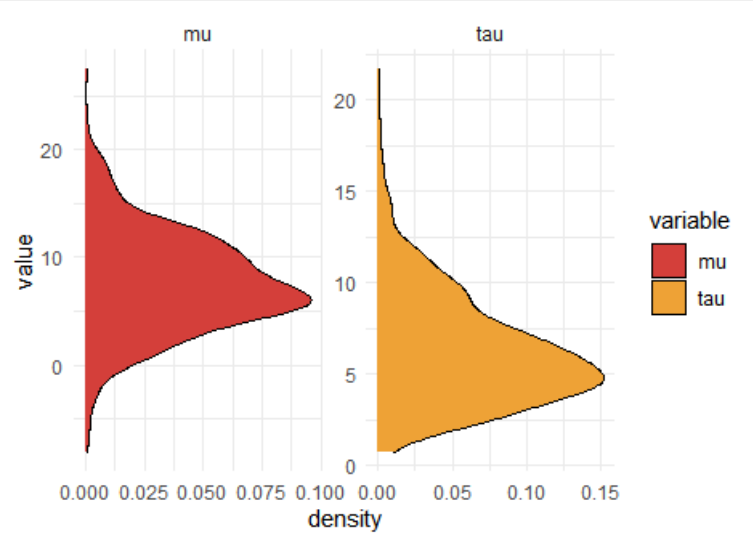
```
bh_summary <- summary(fit2)$summary %>%
  as.data.frame() %>%
  mutate(variable = rownames(.)) %>%
  select(variable, everything()) %>%
  as_data_frame()

bh_summary %>%  #Boxplot de los valores a posteriori al 95%
filter(variable %in% c('mu','tau')) %>%
ggplot() +
geom_linerange(aes(variable, ymin = '2.5%',ymax = '97.5%')) +
geom_crossbar(aes(variable, mean, ymin = '25%', ymax = '75%'), fill= 'grey') +
facet_wrap(~variable, scales = 'free')+theme_minimal()
```



Más gráficos

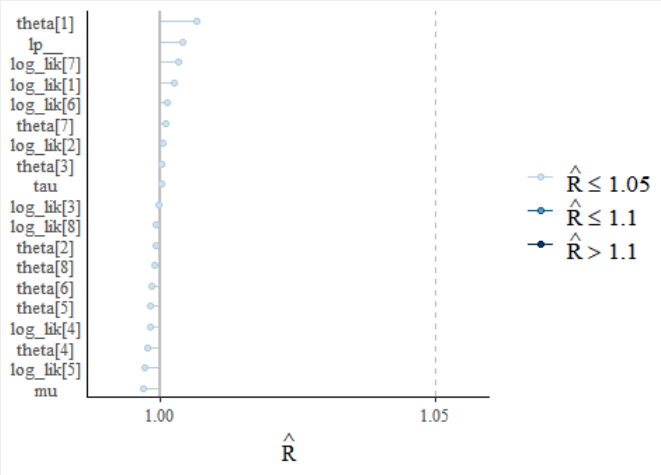
```
bh_mcmc <- fit2 %>%
  rstan::extract()

#Si se quiere incluir la etapa de quema en los graficos
#usar extract(inc_warmup = TRUE)

bh_pars <- bh_mcmc[ c('mu','tau')] %>%
  map_df(as_data_frame, .id = 'variable')

bh_pars %>%
  ggplot(aes(value, fill = variable)) +
  geom_density() +
  facet_wrap(~variable, scales = 'free') +
  coord_flip() +
  scale_fill_locuszoom()+theme_minimal()
```

Densidad


mu
tau
value
density
variable
mu
tau
20
10
0
20
15
10
5
0
0.000
0.025
0.050
0.075
0.100
0.00
0.05
0.10
0.15

Más gráficos

```
fit2%>% #Cuando todo converge, debe verse todo en el 1 aprox.
  rhat() %>%
  mcmc_rhat() +
  yaxis_text()
```



shinystan

```
library(shinystan)
fit1shiny <- launch_shinystan(fit2)

#Abre un shiny que viene con todo y más!
```

Modelo 2

¿Qué pasa si planteáramos un modelo asumiendo independencia entre las escuelas?

```
model2<-"data {
  int<lower=0> J;
  real y[J];
  real<lower=0> sigma[J];
}
parameters {
  real theta[J];
}
model {
  y ~ normal(theta, sigma);
}

generated quantities { //Aquí extraemos la logverosimilitud
  vector[J] log_lik;
  for (j in 1:J) {
    log_lik[j] = normal_lpdf(y[j] | theta[j], sigma[j]);
  }
}"
```

stan()

```
fit3 <-  stan(model_code=model2,
              data=schools_data,
              warmup=8000, #quema
              iter=15000, #largo de la cadena
              thin=30, #saltos
              chains=3, #cadenas
              control = list(adapt_delta = 0.99)) #Tasa aceptación
Inference for Stan model: f6edde0e7251aa1bf17c467ad529ab41.
3 chains, each with iter=15000; warmup=8000; thin=30;
post-warmup draws per chain=234, total post-warmup draws=702.

             mean se_mean    sd   2.5%    25%   50%   75% 97.5% n_eff Rhat
theta[1]   28.25    0.56 15.16   1.88  17.30 27.92 38.08 59.58   744    1
theta[2]    8.06    0.41 10.63 -11.53   0.28  8.80 15.39 28.12   676    1
theta[3]   -2.05    0.64 16.53 -36.96 -11.76 -2.40  9.13 28.92   662    1
theta[4]    7.27    0.38 11.10 -13.64   0.02  6.92 14.53 30.93   841    1
theta[5]   -0.15    0.38  9.44 -18.23  -6.61 -0.25  6.24 18.25   609    1
theta[6]    0.98    0.42 11.62 -21.32  -6.07  0.30  8.54 24.60   762    1
theta[7]   17.26    0.46 10.17  -2.20  10.78 17.50 23.68 37.49   491    1
theta[8]   11.47    0.71 17.93 -20.89  -0.74 11.43 22.92 45.10   630    1
log_lik[1] -4.14    0.03  0.76  -6.09  -4.28 -3.87 -3.68 -3.62   637    1
log_lik[2] -3.78    0.03  0.76  -6.24  -3.92 -3.51 -3.32 -3.24   608    1
log_lik[3] -4.22    0.03  0.71  -6.20  -4.41 -3.91 -3.76 -3.71   795    1
log_lik[4] -3.83    0.03  0.73  -6.00  -3.96 -3.54 -3.36 -3.32   706    1
log_lik[5] -3.66    0.03  0.68  -5.65  -3.85 -3.39 -3.21 -3.16   620    1
log_lik[6] -3.87    0.03  0.78  -6.11  -4.00 -3.56 -3.39 -3.35   680    1
log_lik[7] -3.74    0.03  0.71  -5.82  -3.85 -3.45 -3.31 -3.26   629    1
log_lik[8] -4.31    0.03  0.75  -6.22  -4.50 -4.00 -3.84 -3.79   636    1
```

Loo criteria

```
library(loo)
log_lik_1 <- extract_log_lik(fit2) #Modelo jerárquico combina escuelas
log_lik_2 <- extract_log_lik(fit3) #Modelo asumiendo independencia
(loofit1<-loo(log_lik_1))

        Estimate  SE
elpd_loo    -30.8 1.0
p_loo         1.2 0.3
looic        61.6 2.1
------
Monte Carlo SE of elpd_loo is 0.1.
Pareto k diagnostic values:
                         Count Pct.    Min. n_eff
(-Inf, 0.5]   (good)     6     75.0%   435
 (0.5, 0.7]   (ok)       2     25.0%   502
   (0.7, 1]   (bad)      0      0.0%   <NA>
   (1, Inf)   (very bad) 0      0.0%   <NA>

(loofit2<-loo(log_lik_2))
         Estimate  SE
elpd_loo    -35.5 0.7
p_loo         5.3 0.3
looic        71.1 1.4
------
Monte Carlo SE of elpd_loo is NA.
Pareto k diagnostic values:
                         Count Pct.    Min. n_eff
(-Inf, 0.5]   (good)     0      0.0%   <NA>
 (0.5, 0.7]   (ok)       2     25.0%   136
   (0.7, 1]   (bad)      5     62.5%   33
   (1, Inf)   (very bad) 1     12.5%   20
```