Proyecto Inferencia Bayesiana

Vanessa Alcalde, Luciano Garrido, Pablo Martinez Angerosa

27/11/2020

Introducción

Para el trabajo vamos a intentar obtener un modelo lineal bayesiano que explique la relación de la tasa de mortalidad en las 60 ciudades estadounidenses del año 1963. Lo compararemos con los resultados obtenidos por modelos lineales múltiples en el trabajo. Esta base de datos fue la que utilizamos en los trabajos de Análisis Multivariado I.

Nuestro primer enfoque fue realizar un ajuste de modelo de regresión lineal múltiple y realizando los test necesarios para la obtención de un modelo que consideramos adecuado que cumple con todos los supuestos teóricos necesarios y logra un R^2 de 0.8076 y reduciendo el modelo a 6 variables significativas.

El modelo resultante por modelos lineales es:

$$MORT_i = \beta_0 + \beta_1 PREC_i + \beta_2 JANT_i + \beta_3 JULT_i + \beta_4 EDUC_i + \beta_5 NONW_i + \beta_6 SO_i + \varepsilon_i$$

Donde $\varepsilon_i \sim N(0, \sigma^2)$.

Utilizamos el mismo modelo desde un enfoque bayesiano. Como información a priori de cada uno de los β_i y σ^2 utilizamos las estimaciones puntuales obtenidas desde modelos lineales en particular para los β_i tomamos distribuciones normales cuya media son las estimaciones puntuales y su correspondiente error estándar como desviación típica.

Datos

La base de datos consta de 16 variables de polución del aire, socioeconómicas y meteorológicas de 60 ciudades de Estados Unidos en 1963. El artículo original de donde se extrae la base de datos es G.C. McDonald and R.C. Schwing, "Instabilities of Regression Estimates Relating Air Pollution to Mortality," Technometrics, vol. 15, pp. 463-482, 1973.

En la Tabla 1 hacemos una descripción de las variables en la base de datos.

Resultados

Los resultados obtenidos

```
datos <- utils::read.table('polucion.txt', sep='\t',
                           header=TRUE,
                           row.names=NULL, stringsAsFactors=FALSE,
                           dec = ",")
```

Tabla 1: Descripción de variables en la base de datos.

Nombre de variables	Descripción
PREC	Promedio anual de precipitación (en pulgadas).
JANT	Promedio de temperatura del mes de Enero (en Farenheit).
JULT	Promedio de temperatura del mes de Julio (en Farenheit).
OVR65	Porcentaje de población mayor de 65 años en áreas metropolitanas.
POPN	Promedio del tamaño del hogar.
EDUC	Mediana de años de escolarización completos para mayores de 22 años.
HOUS	% de viviendas en buenas condiciones con todos los servicios.
DENS	Población por milla cuadrada en áreas urbanas en 1960.
NONW	% de población no blanca en áreas urbanas en 1960.
WWDRK	% de trabajadores en ocupaciones "no manuales".
POOR	% de familias con ingresos anuales menores $3000.
HC	Polución potencial relativa de hidrocarbono.
NOX	Polución potencial relativa de óxido nítrico.
SO	Polución potencial relativa de dióxido de azufre.
HUMID	Promedio anual del porcentaje de humedad relativa a las 13 horas.
MORT	Tasa de mortalidad cada 100.000 habitantes.

```r
datosML <- datos[-c(6, 28, 32, 37, 2),]

modelo1 <- rstanarm::stan_glm(MORT ~ PREC + JANT +
                    JULT + EDUC + NONW + SO,
                    data = datosML,
                    family = gaussian(link = "identity"),
                    prior_intercept = rstanarm::normal(1122.79104, 101.12442),
                    prior = rstanarm::normal(base::c(2.23936, -1.03784, -2.10246,
                                                     -12.41246, 3.89109, 0.30278),
                                             base::c(0.52209, 0.48114, 0.98114,
                                                     5.36332, 0.63640, 0.06265)),
                    prior_aux = rstanarm::exponential(2),
                    seed = 12345)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.258 seconds (Warm-up)
## Chain 1:                0.073 seconds (Sampling)
## Chain 1:                0.331 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.204 seconds (Warm-up)
## Chain 2:                0.106 seconds (Sampling)
## Chain 2:                0.31 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.399 seconds (Warm-up)
## Chain 3:                0.091 seconds (Sampling)
## Chain 3:                0.49 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.364 seconds (Warm-up)
## Chain 4:                0.093 seconds (Sampling)
## Chain 4:                0.457 seconds (Total)
## Chain 4:
```

```
base::summary(modelo1)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      MORT ~ PREC + JANT + JULT + EDUC + NONW + SO
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 55
##  predictors:   7
##
## Estimates:
##               mean   sd     10%    50%    90%
## (Intercept) 1121.4   56.2 1051.6 1120.7 1195.0
## PREC           2.2    0.3    1.9    2.2    2.6
## JANT          -1.0    0.3   -1.4   -1.0   -0.7
## JULT          -2.1    0.6   -2.8   -2.1   -1.4
## EDUC         -12.3    3.1  -16.2  -12.4   -8.4
## NONW           3.9    0.3    3.5    3.9    4.3
## SO             0.3    0.0    0.3    0.3    0.4
## sigma         19.5    1.4   17.8   19.4   21.2
##
## Fit Diagnostics:
```

```
##            mean   sd    10%   50%   90%
## mean_PPD 937.8    3.8 932.9 937.8 942.7
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   1.0  1.0  3339
## PREC          0.0  1.0  3375
## JANT          0.0  1.0  4042
## JULT          0.0  1.0  3729
## EDUC          0.1  1.0  3537
## NONW          0.0  1.0  3211
## SO            0.0  1.0  3640
## sigma         0.0  1.0  4216
## mean_PPD      0.1  1.0  4083
## log-posterior 0.0  1.0  1932
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
rstanarm::prior_summary(modelo1)
```

```
## Priors for model 'modelo1'
## ------
## Intercept (after predictors centered)
##  ~ normal(location = 1123, scale = 101)
##
## Coefficients
##  ~ normal(location = [ 2.2,-1.0,-2.1,...], scale = [0.52,0.48,0.98,...])
##
## Auxiliary (sigma)
##  ~ exponential(rate = 2)
## ------
## See help('prior_summary.stanreg') for more details
```

```
modelo1
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      MORT ~ PREC + JANT + JULT + EDUC + NONW + SO
##  observations: 55
##  predictors:   7
## ------
##             Median MAD_SD
## (Intercept) 1120.7   56.9
## PREC           2.2    0.3
## JANT          -1.0    0.3
## JULT          -2.1    0.6
## EDUC         -12.4    3.1
## NONW           3.9    0.3
## SO             0.3    0.0
##
## Auxiliary parameter(s):
```

```
##       Median MAD_SD
## sigma 19.4    1.3  
## 
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

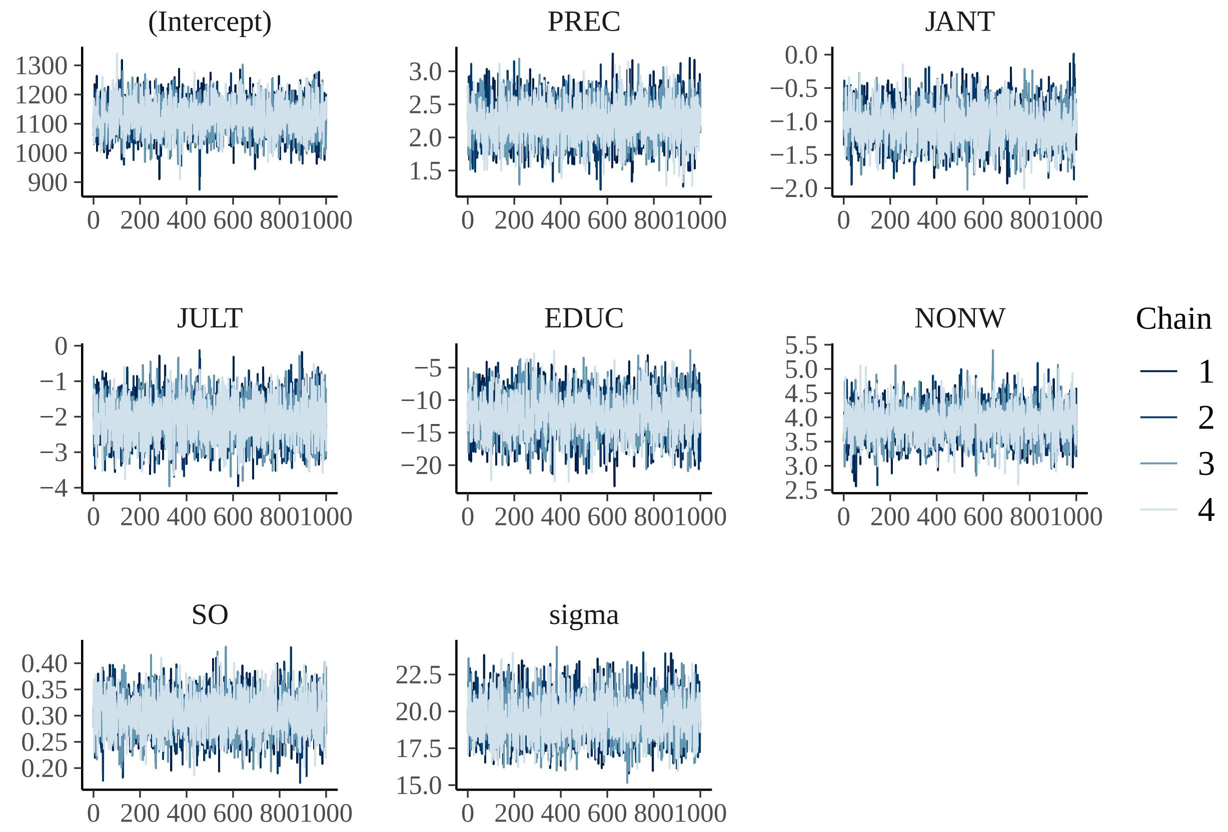
```
modelo1$coefficients
```

```
##  (Intercept)         PREC         JANT         JULT         EDUC         NONW 
## 1120.7221909    2.2394305   -1.0322829   -2.0960594  -12.3737964    3.8933694 
##           SO 
##    0.3035902
```

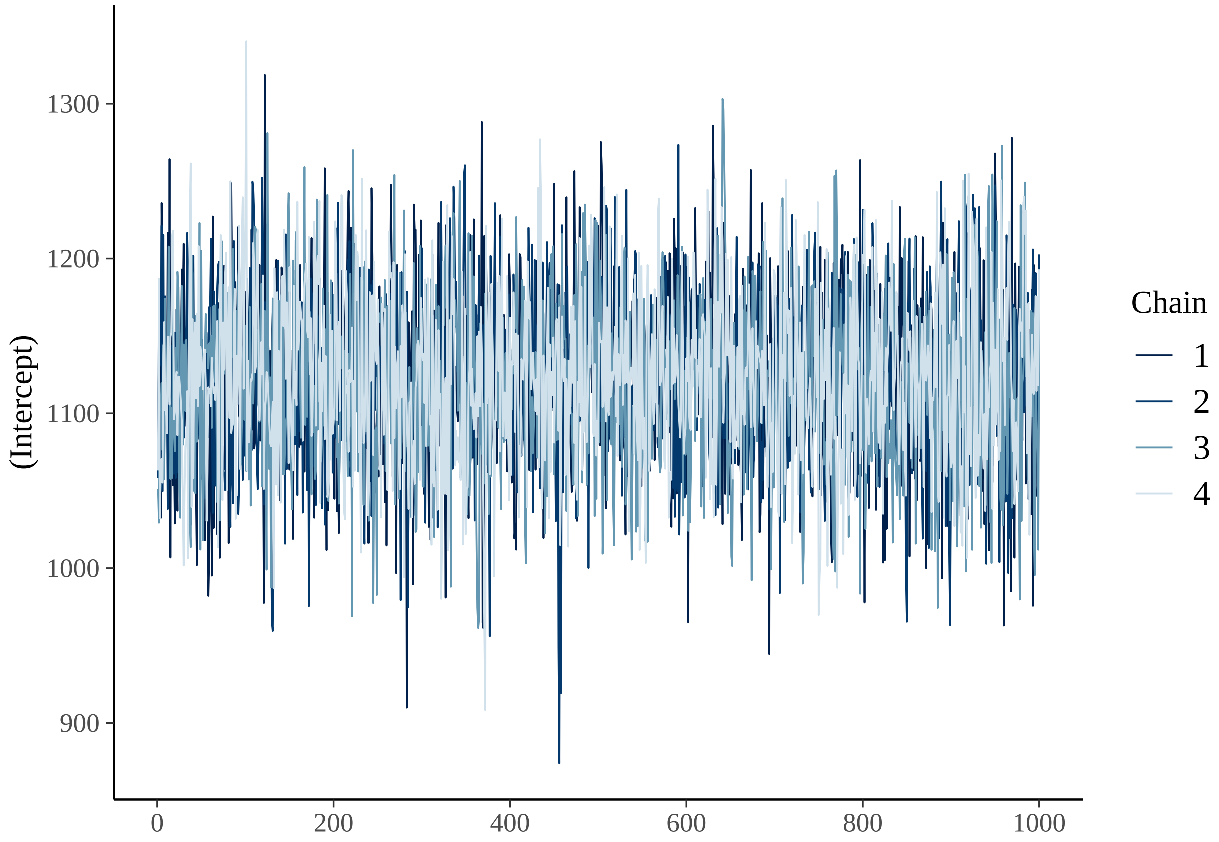
```
modelo1$stanfit
```

```
## Inference for Stan model: continuous.
## 4 chains, each with iter=2000; warmup=1000; thin=1; 
## post-warmup draws per chain=1000, total post-warmup draws=4000.
## 
##                  mean se_mean    sd    2.5%     25%     50%     75%   97.5%
## (Intercept)   1121.43    0.97 56.20 1013.04 1082.94 1120.72 1159.89 1231.92
## PREC             2.24    0.01  0.29    1.67    2.05    2.24    2.44    2.83
## JANT            -1.04    0.00  0.28   -1.58   -1.22   -1.03   -0.86   -0.50
## JULT            -2.10    0.01  0.57   -3.19   -2.49   -2.10   -1.70   -0.98
## EDUC           -12.33    0.05  3.10  -18.34  -14.42  -12.37  -10.21   -6.20
## NONW             3.89    0.01  0.35    3.21    3.66    3.89    4.13    4.57
## SO               0.30    0.00  0.04    0.23    0.28    0.30    0.33    0.37
## sigma           19.48    0.02  1.36   16.98   18.55   19.43   20.36   22.37
## mean_PPD       937.80    0.06  3.85  930.54  935.21  937.79  940.38  945.35
## log-posterior -309.33    0.05  2.07 -314.38 -310.40 -308.99 -307.82 -306.34
##               n_eff Rhat
## (Intercept)    3339    1
## PREC           3375    1
## JANT           4042    1
## JULT           3729    1
## EDUC           3537    1
## NONW           3211    1
## SO             3640    1
## sigma          4216    1
## mean_PPD       4083    1
## log-posterior  1932    1
## 
## Samples were drawn using NUTS(diag_e) at Tue Nov 24 21:39:05 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at 
## convergence, Rhat=1).
```

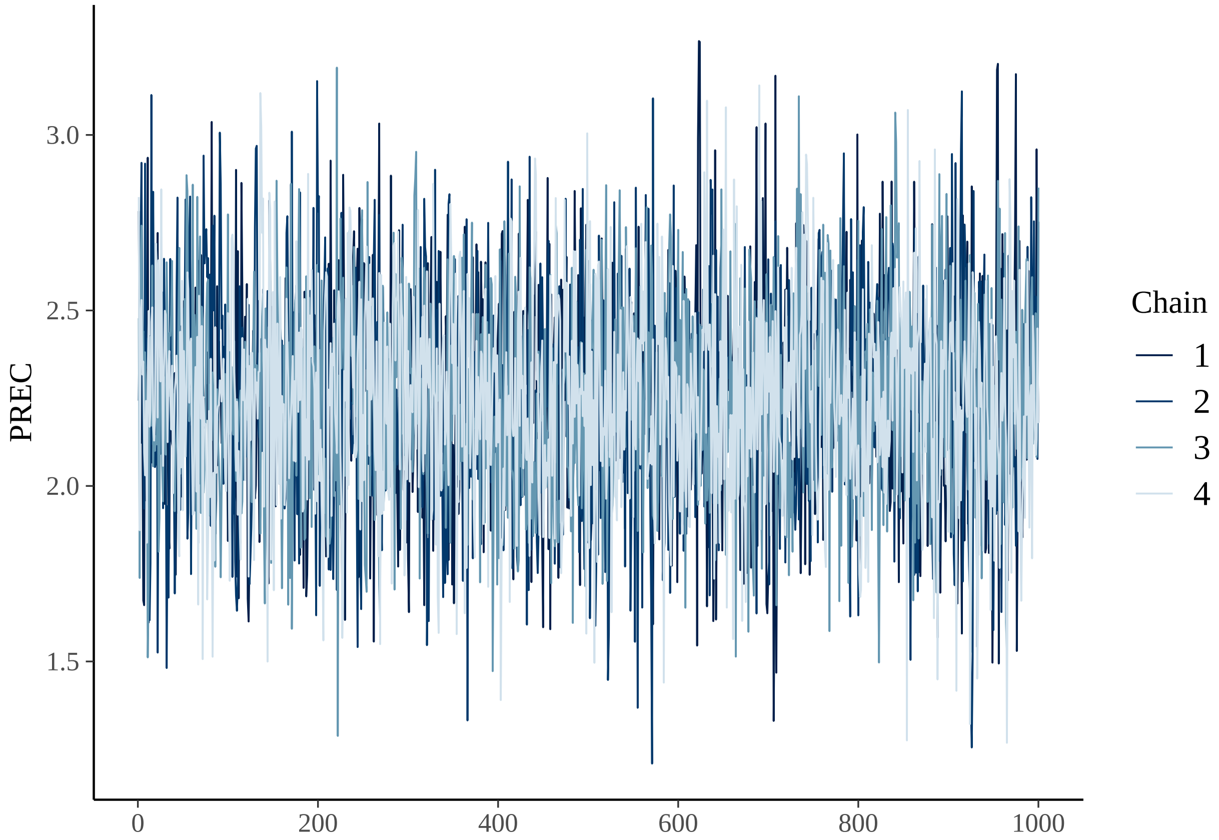
```
###Diagnostico de las Cadenas
bayesplot::mcmc_trace(modelo1)
```



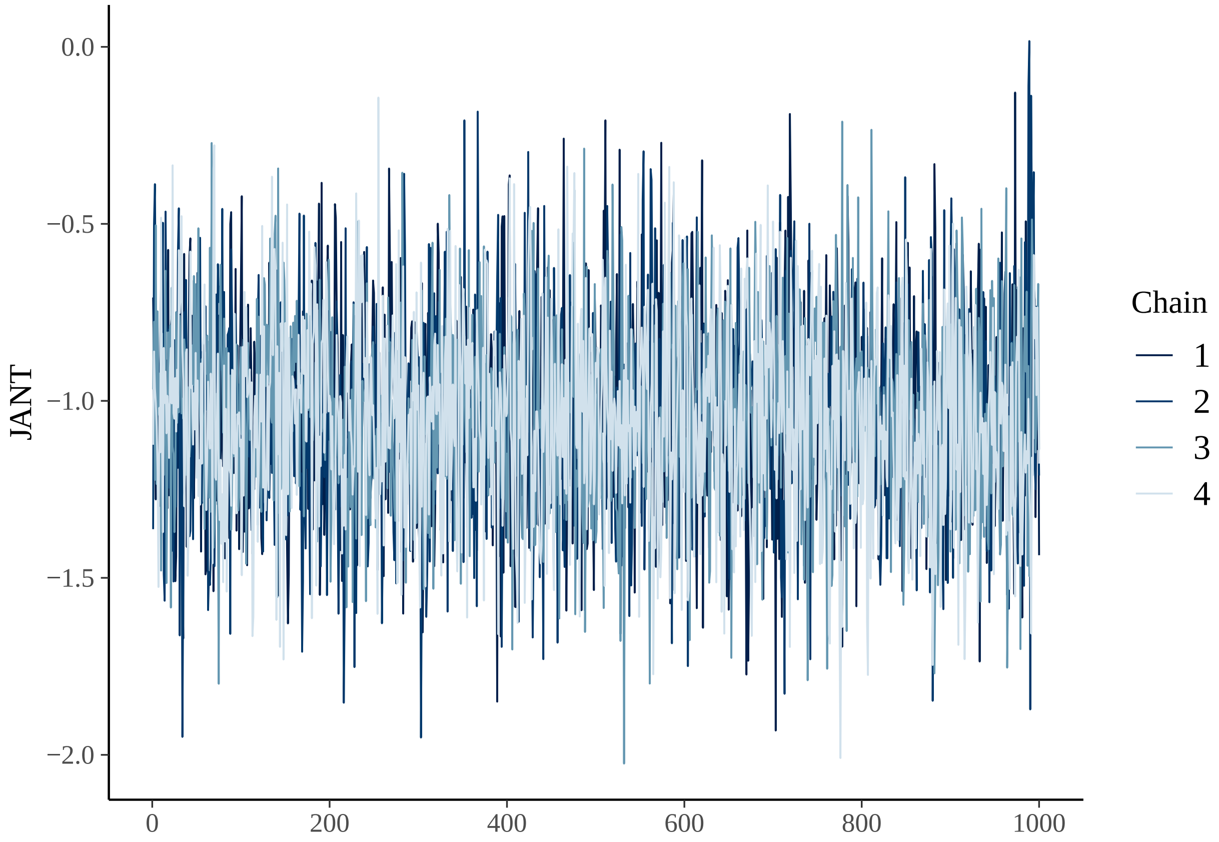
```
bayesplot::mcmc_trace(modelo1,pars = "(Intercept)")
```



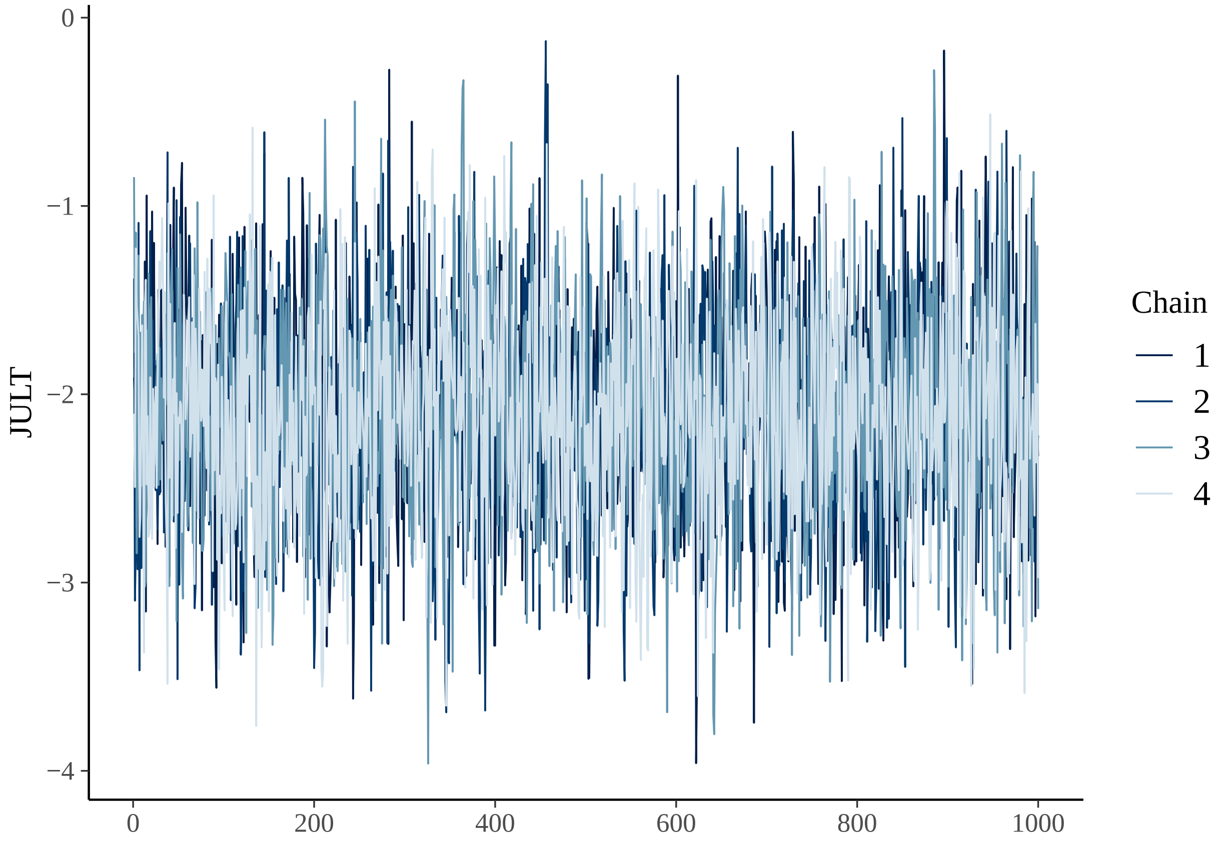
```r
bayesplot::mcmc_trace(modelo1,pars = "PREC")
```



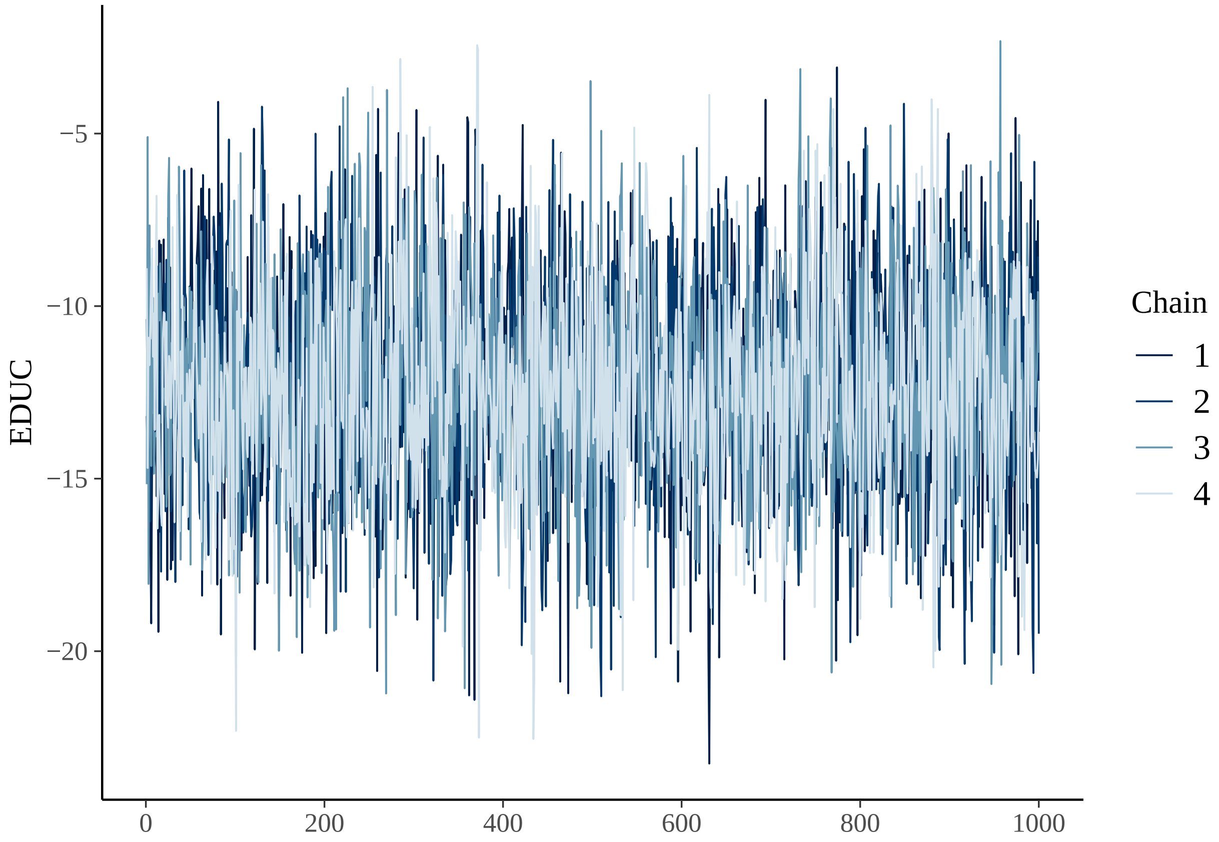
```r
bayesplot::mcmc_trace(modelo1,pars = "JANT")
```



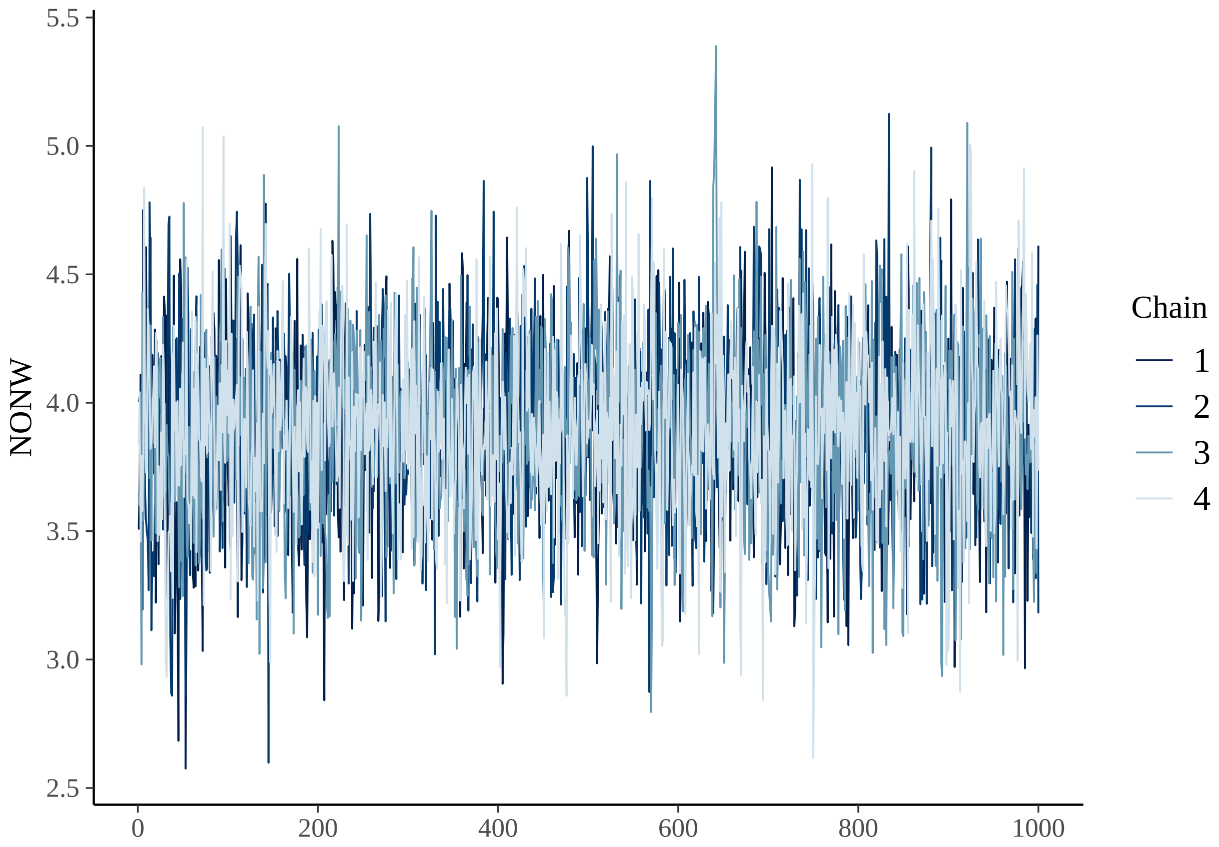
```
bayesplot::mcmc_trace(modelo1,pars = "JULT")
```



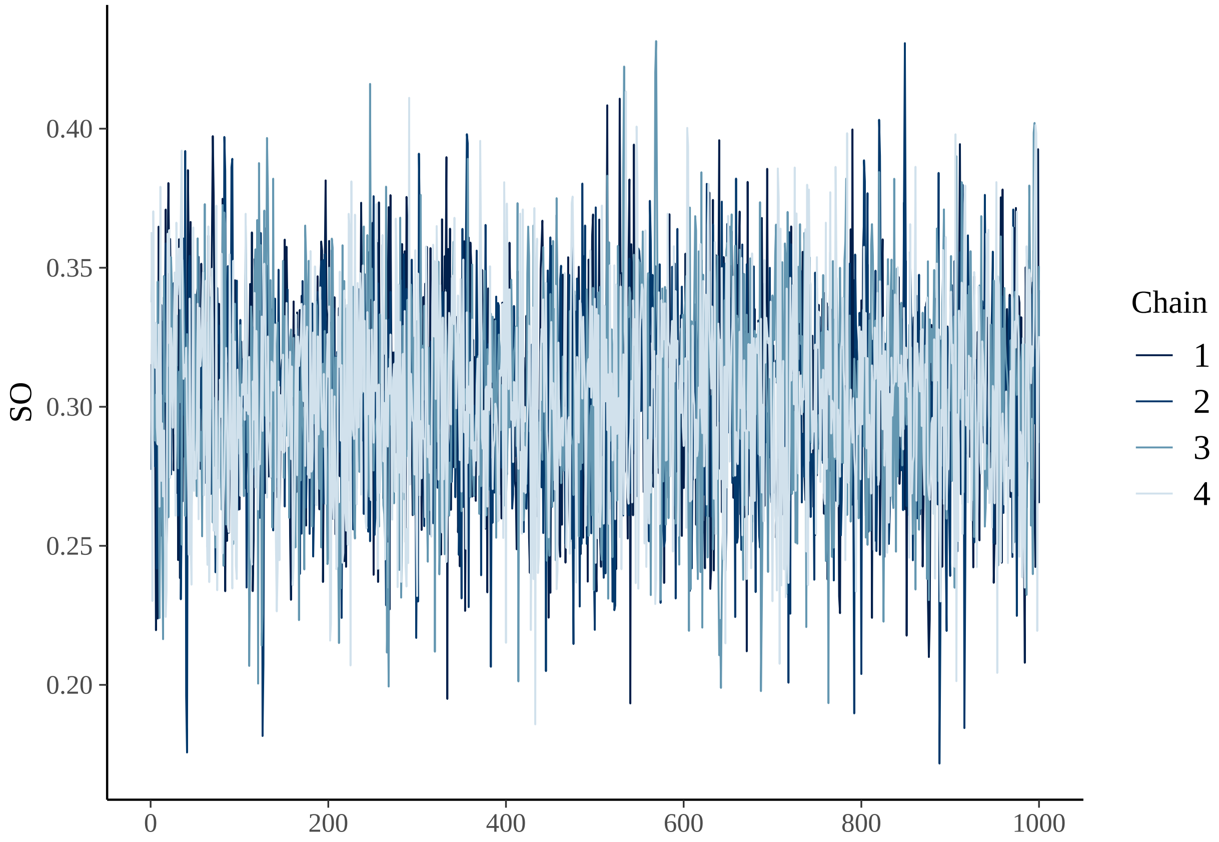
```
bayesplot::mcmc_trace(modelo1,pars = "EDUC")
```



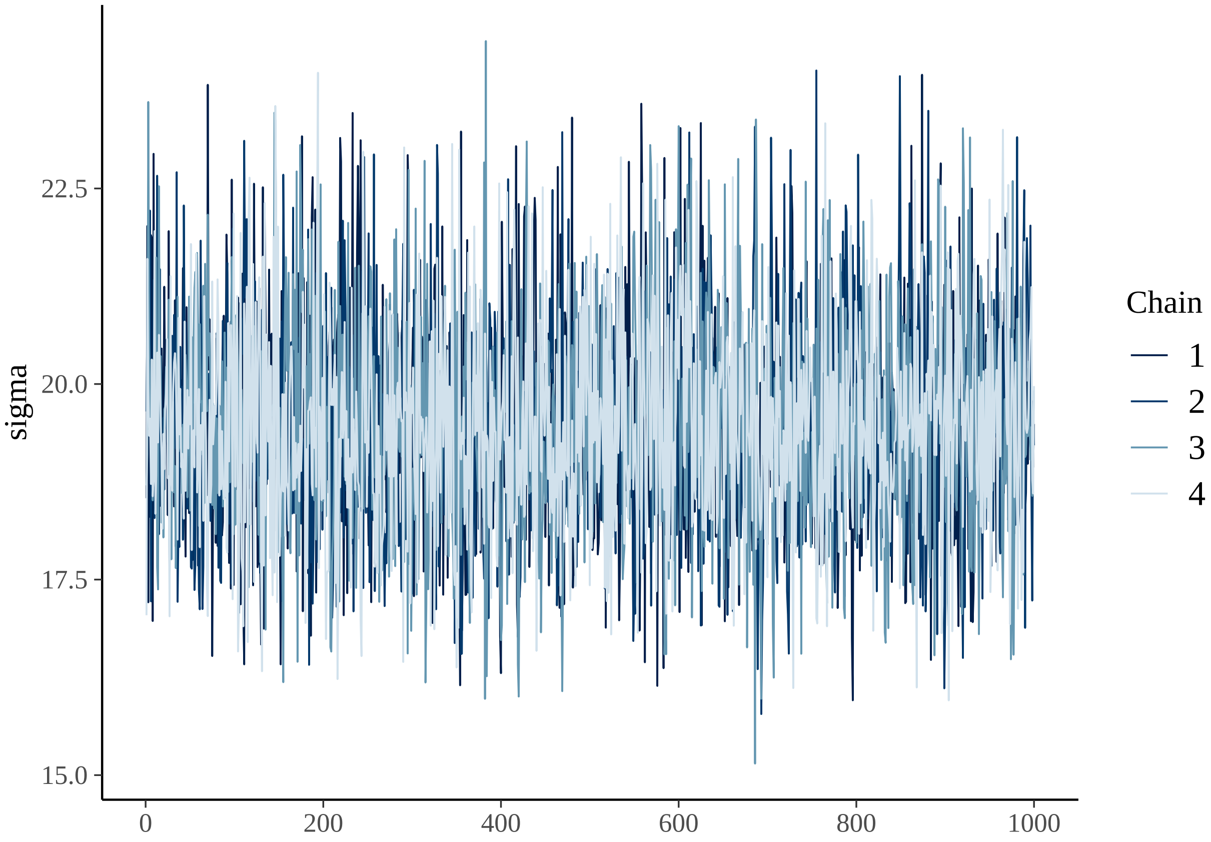
```r
bayesplot::mcmc_trace(modelo1,pars = "NONW")
```



```
bayesplot::mcmc_trace(modelo1,pars = "SO")
```

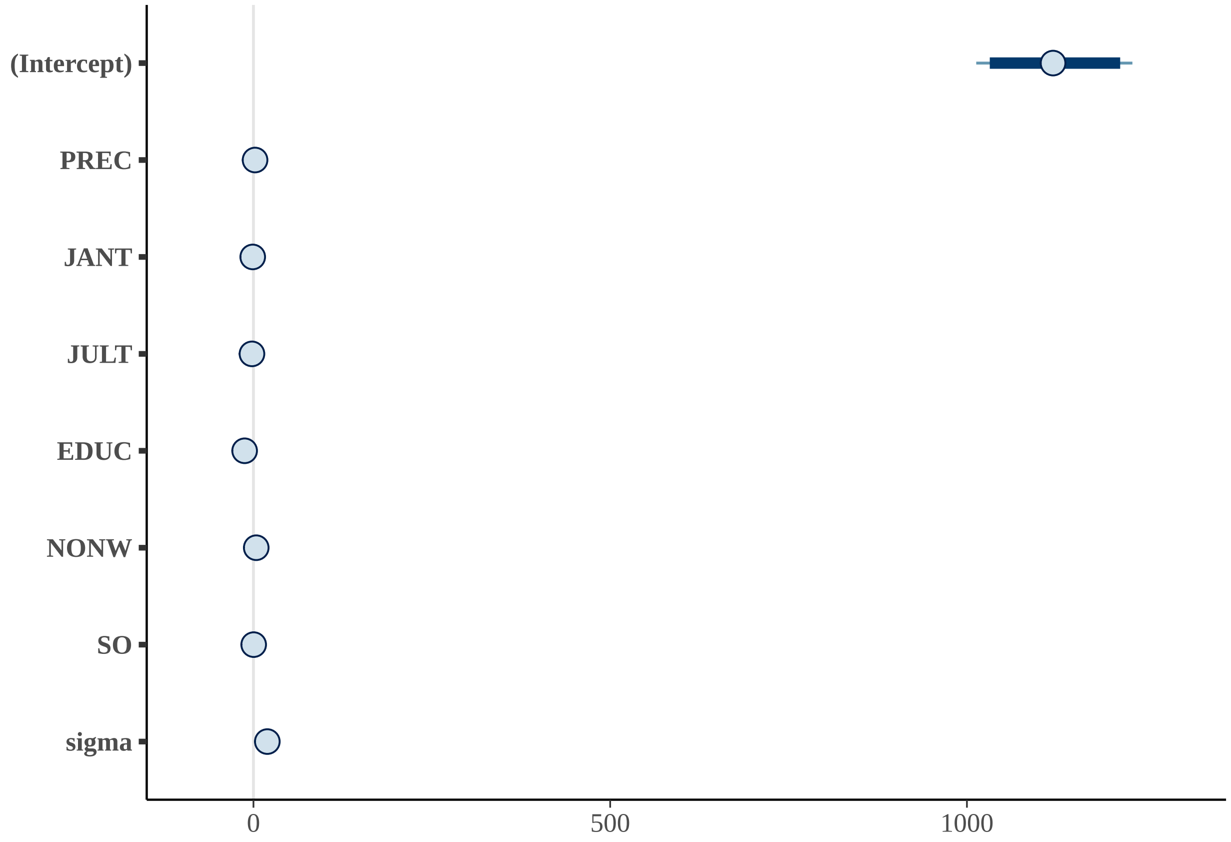


```r
bayesplot::mcmc_trace(modelo1,pars = "sigma")
```



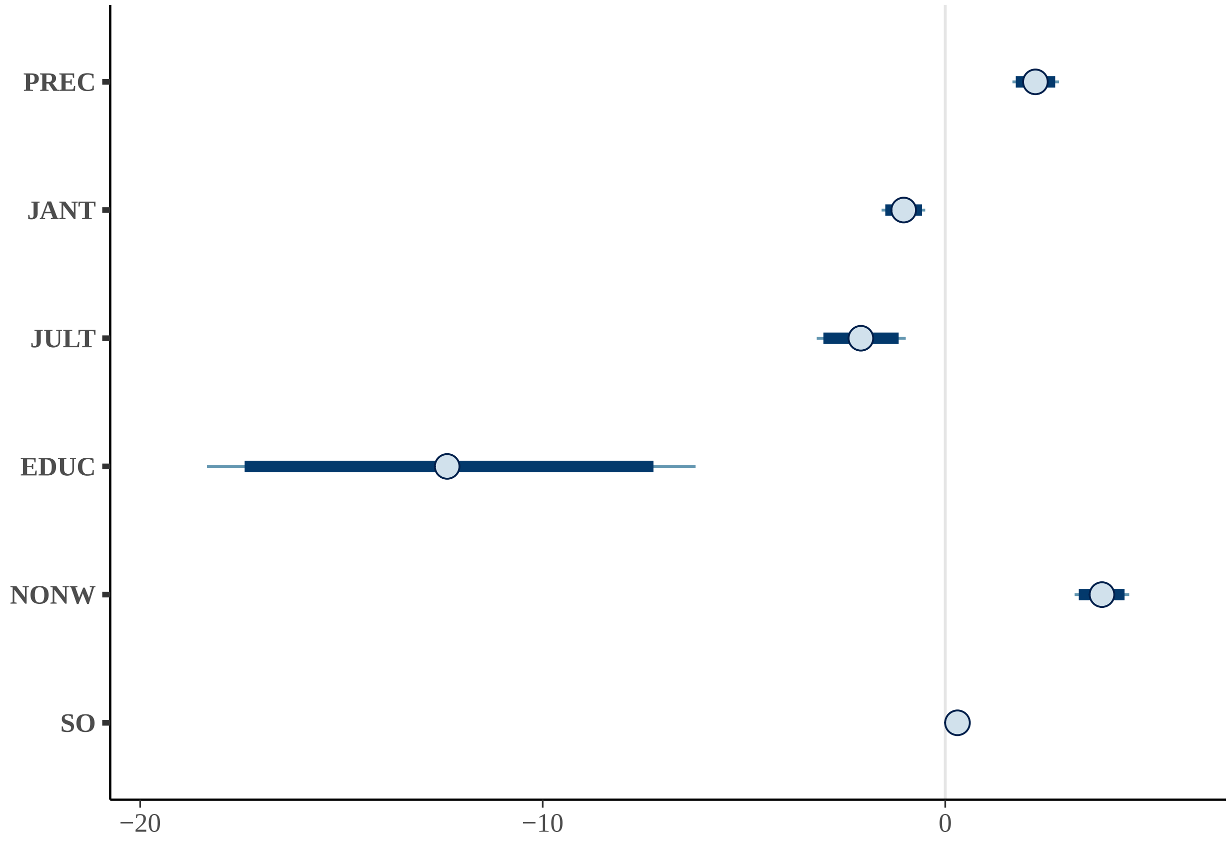
```
##Intervalos de credibilidad del 95%
bayesplot::mcmc_intervals(modelo1, prob = 0.95)
```

```
## Warning: ‘prob_outer‘ (0.9) is less than ‘prob‘ (0.95)
## ... Swapping the values of ‘prob_outer‘ and ‘prob‘
```



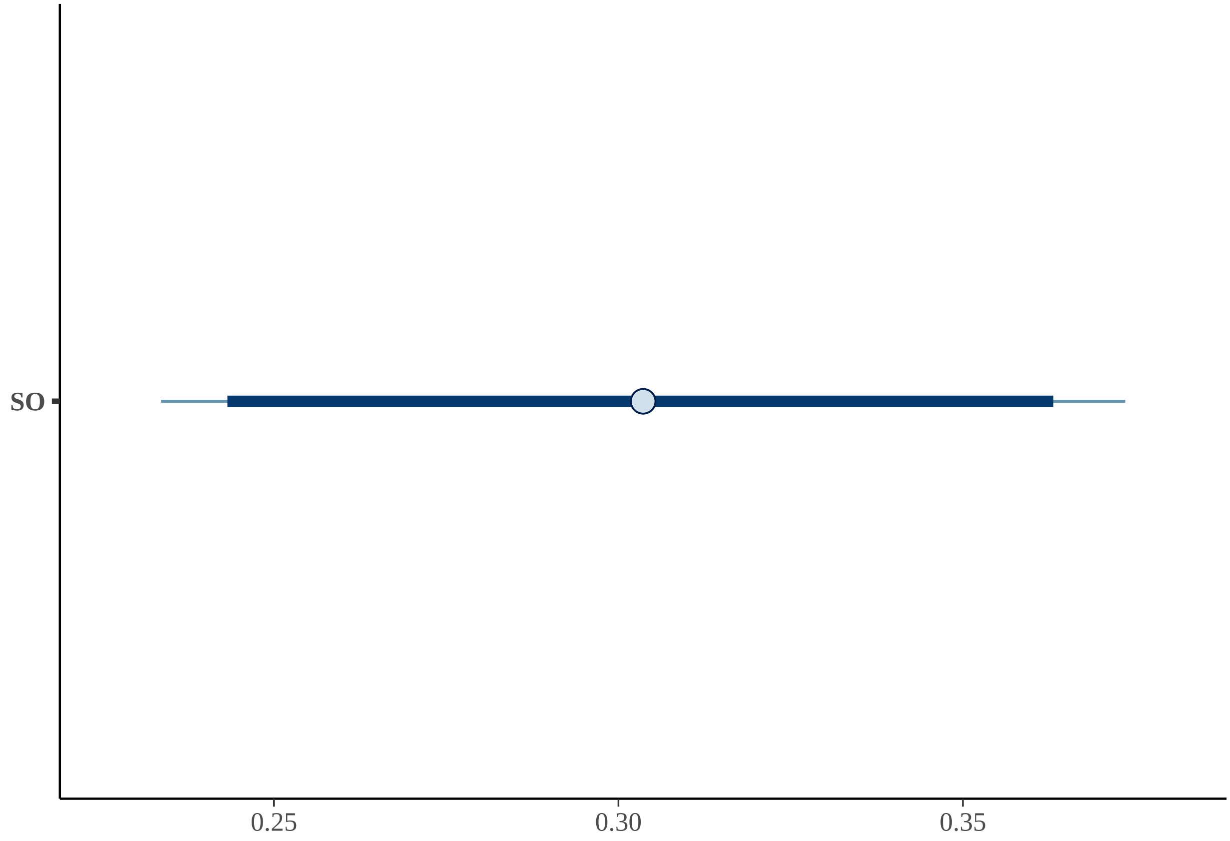
```
bayesplot::mcmc_intervals(modelo1 ,pars=c("PREC",
                               "JANT",
                               "JULT",
                               "EDUC",
                               "NONW",
                               "SO"), prob = 0.95)
```

```
## Warning: ‘prob_outer‘ (0.9) is less than ‘prob‘ (0.95)
## ... Swapping the values of ‘prob_outer‘ and ‘prob‘
```



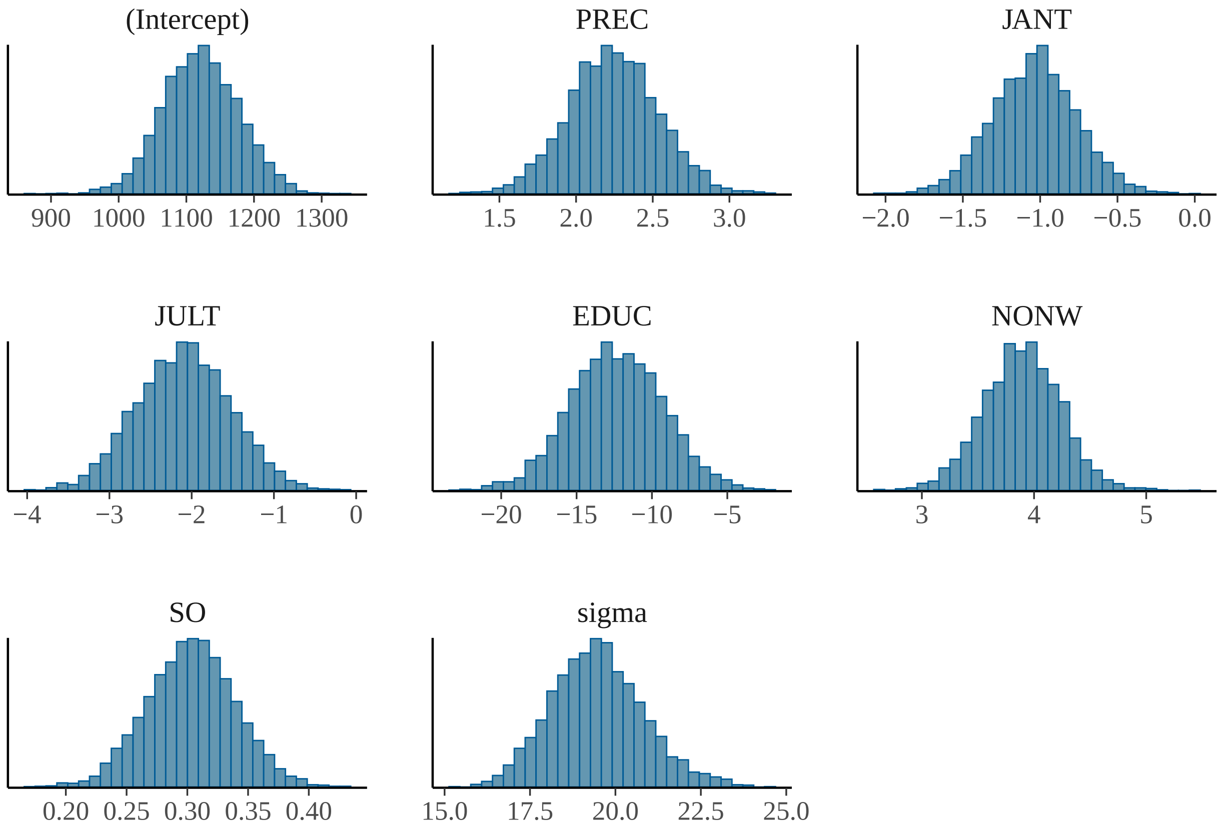
```
bayesplot::mcmc_intervals(modelo1 ,pars=c("SO"), prob = 0.95)
```

```
## Warning: ‘prob_outer‘ (0.9) is less than ‘prob‘ (0.95)
## ... Swapping the values of ‘prob_outer‘ and ‘prob‘
```



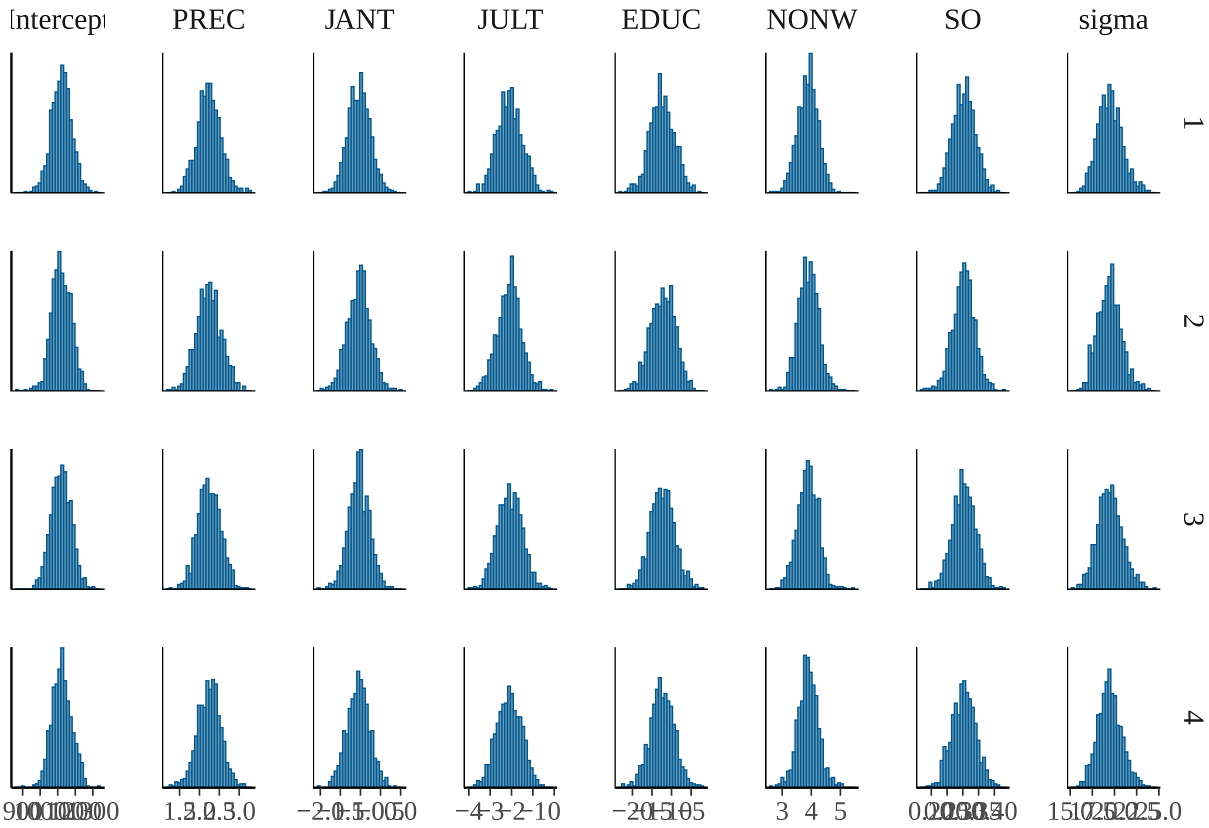
```
bayesplot::mcmc_hist(modelo1)#Histograma de los coeficientes
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```

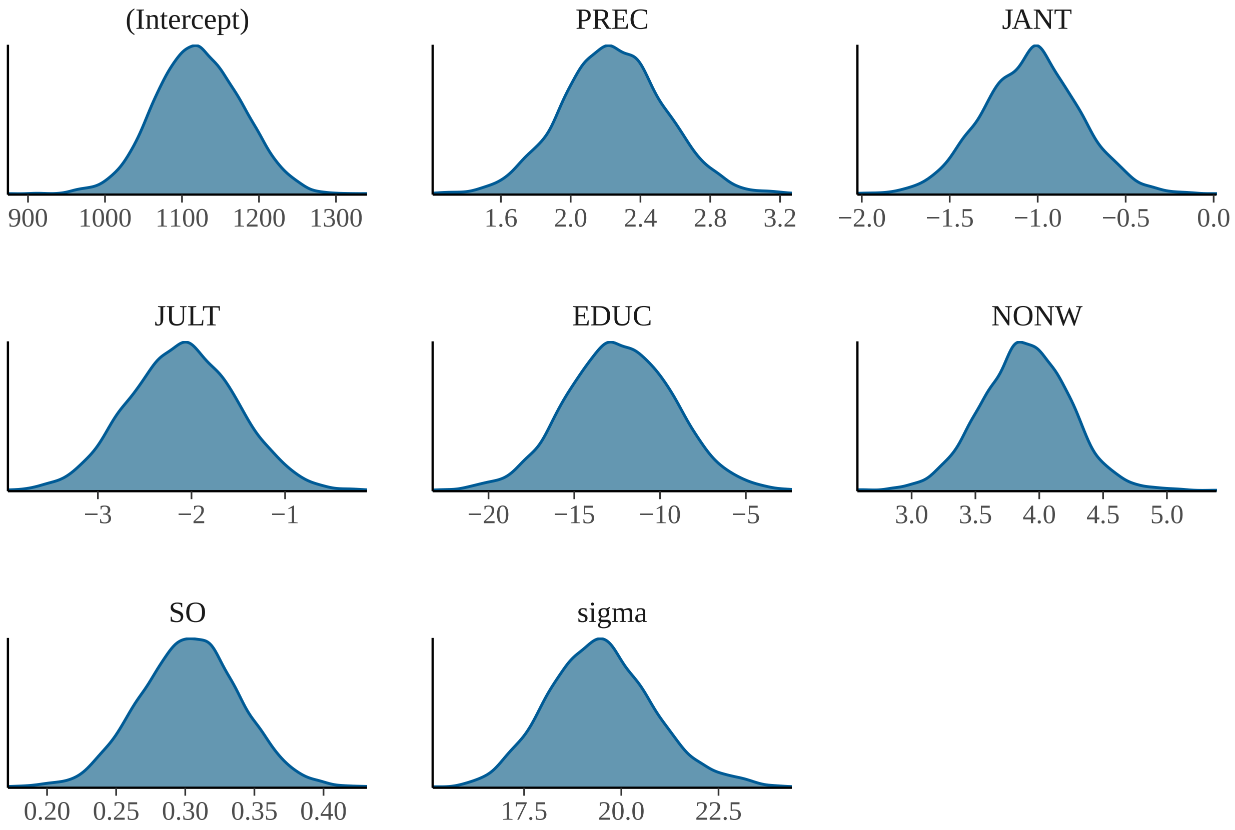


```
bayesplot::mcmc_hist_by_chain(modelo1)#Histograma de los coeficientes por cadena
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```

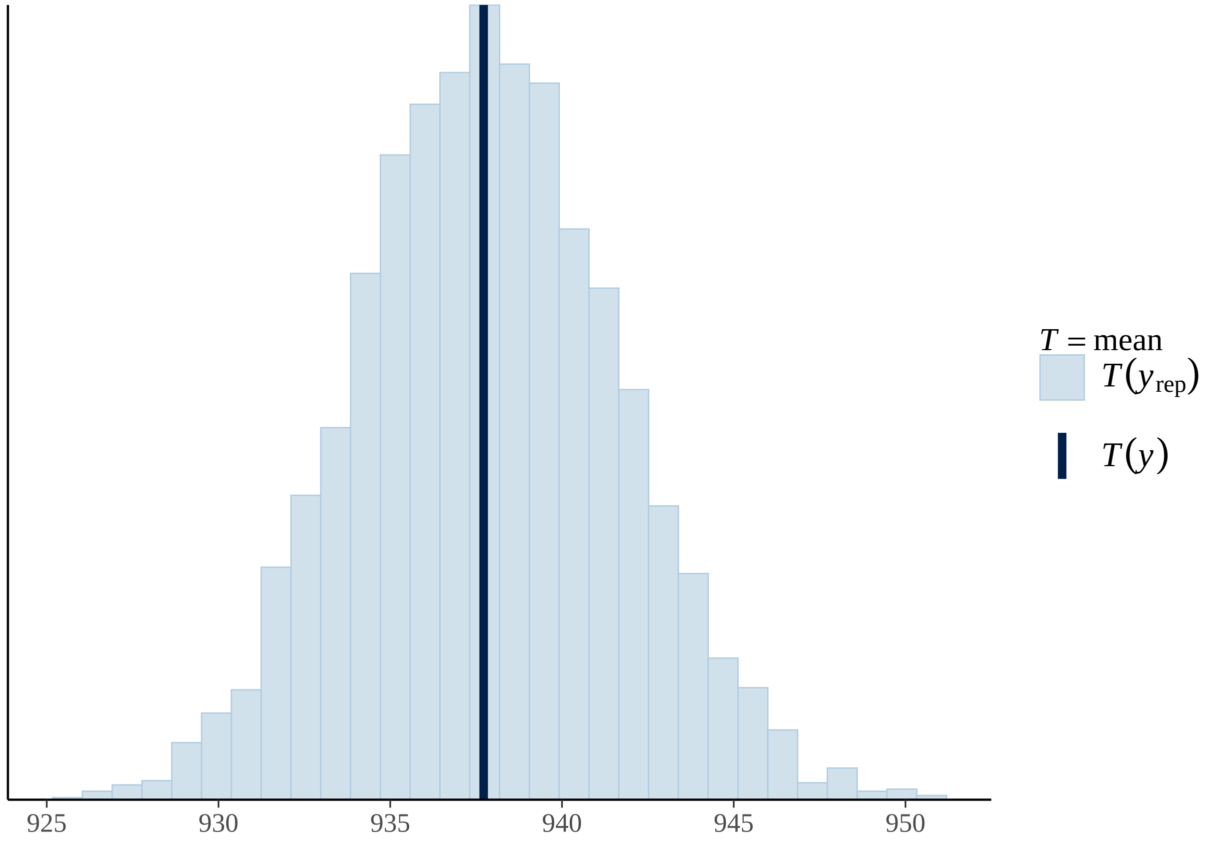


```
bayesplot::mcmc_dens(modelo1)#Estimacion de densidad de los coeficientes
```



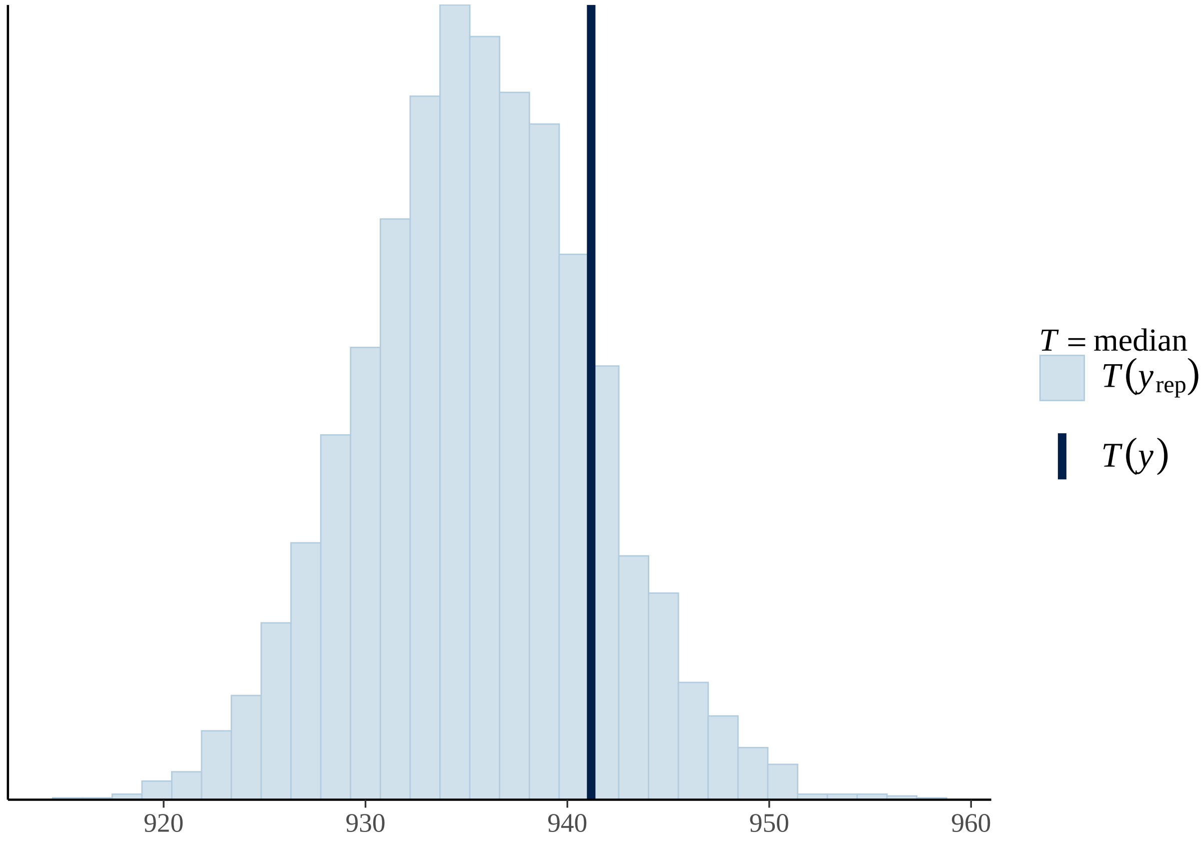
```
# Diagnostico de Modelo
##Predictivas posteriores
pred<-rstanarm::posterior_predict(modelo1,draws = 100)
bayesplot::pp_check(modelo1, plotfun = "stat", stat = "mean")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



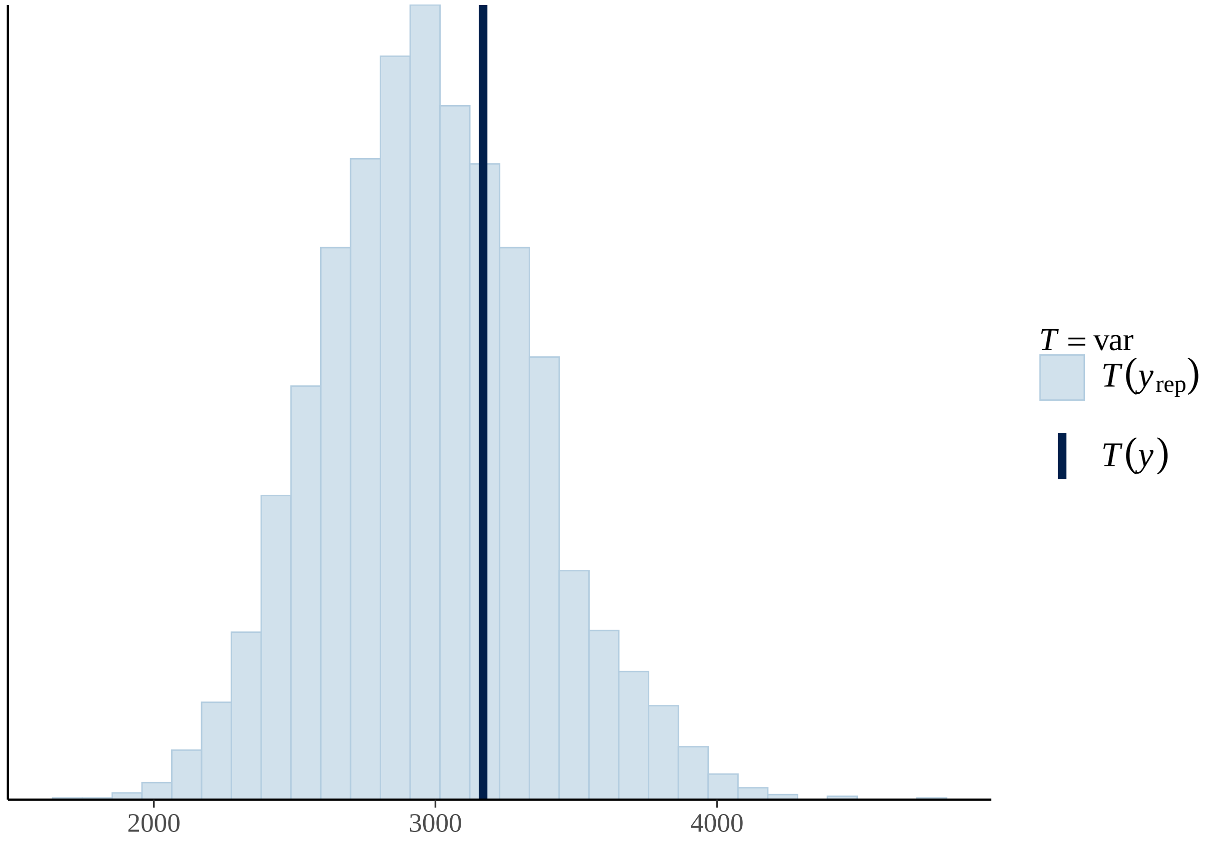
```r
bayesplot::pp_check(modelo1, plotfun = "stat", stat = "median")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



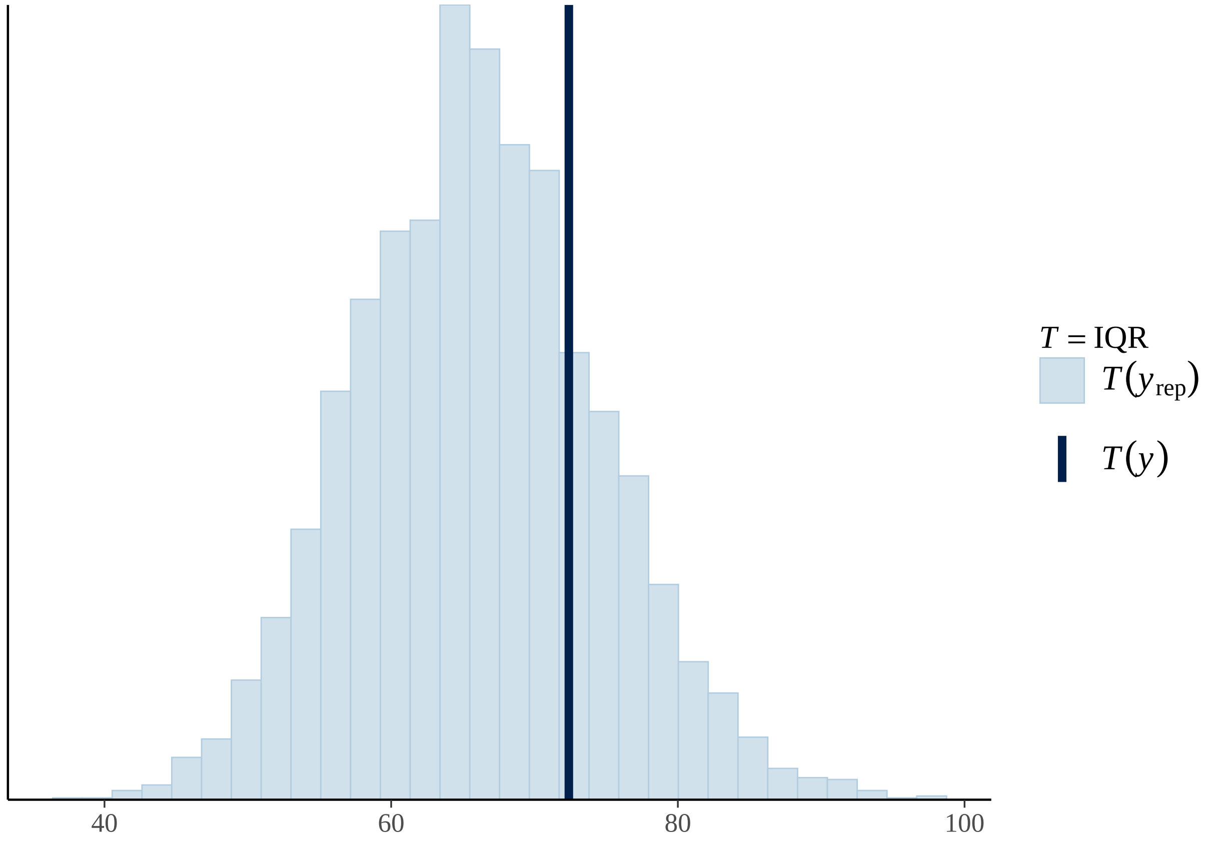
```r
bayesplot::pp_check(modelo1, plotfun = "stat", stat = "var")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



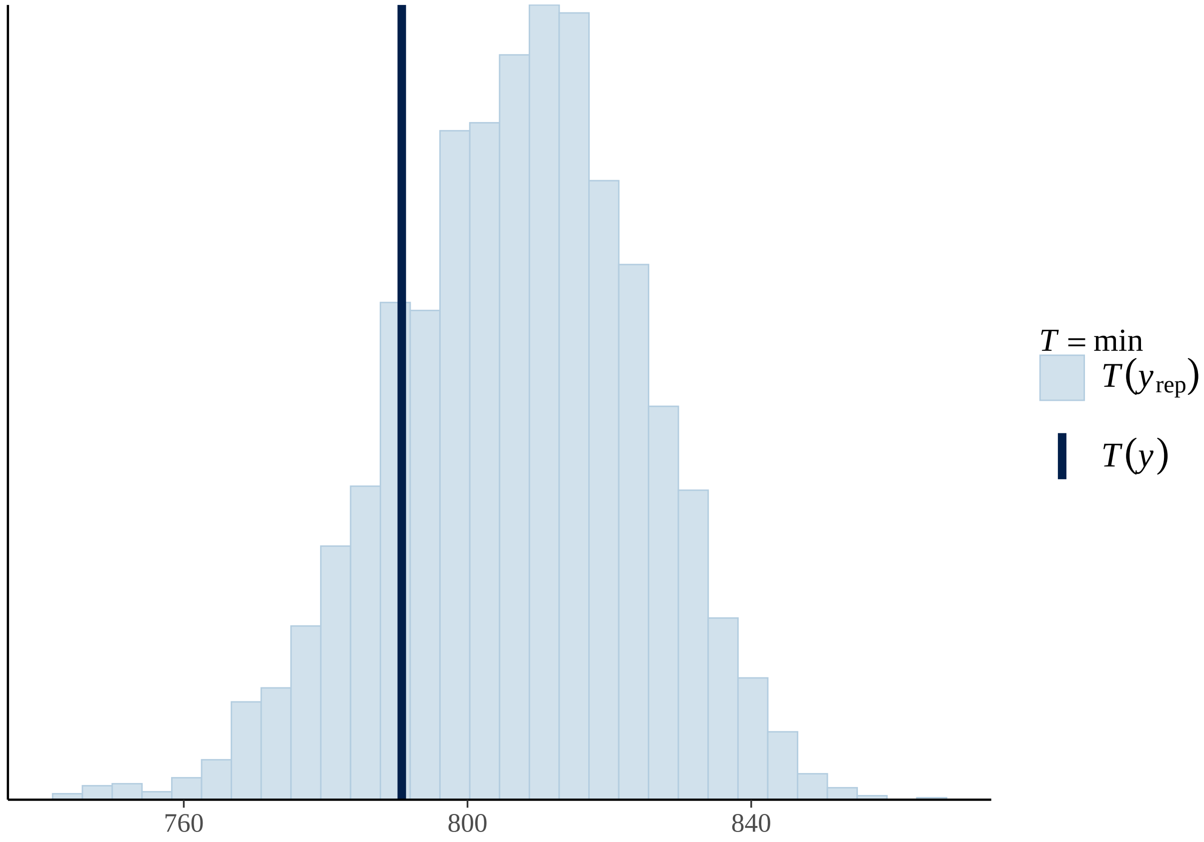
```r
bayesplot::pp_check(modelo1, plotfun = "stat", stat = "IQR")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



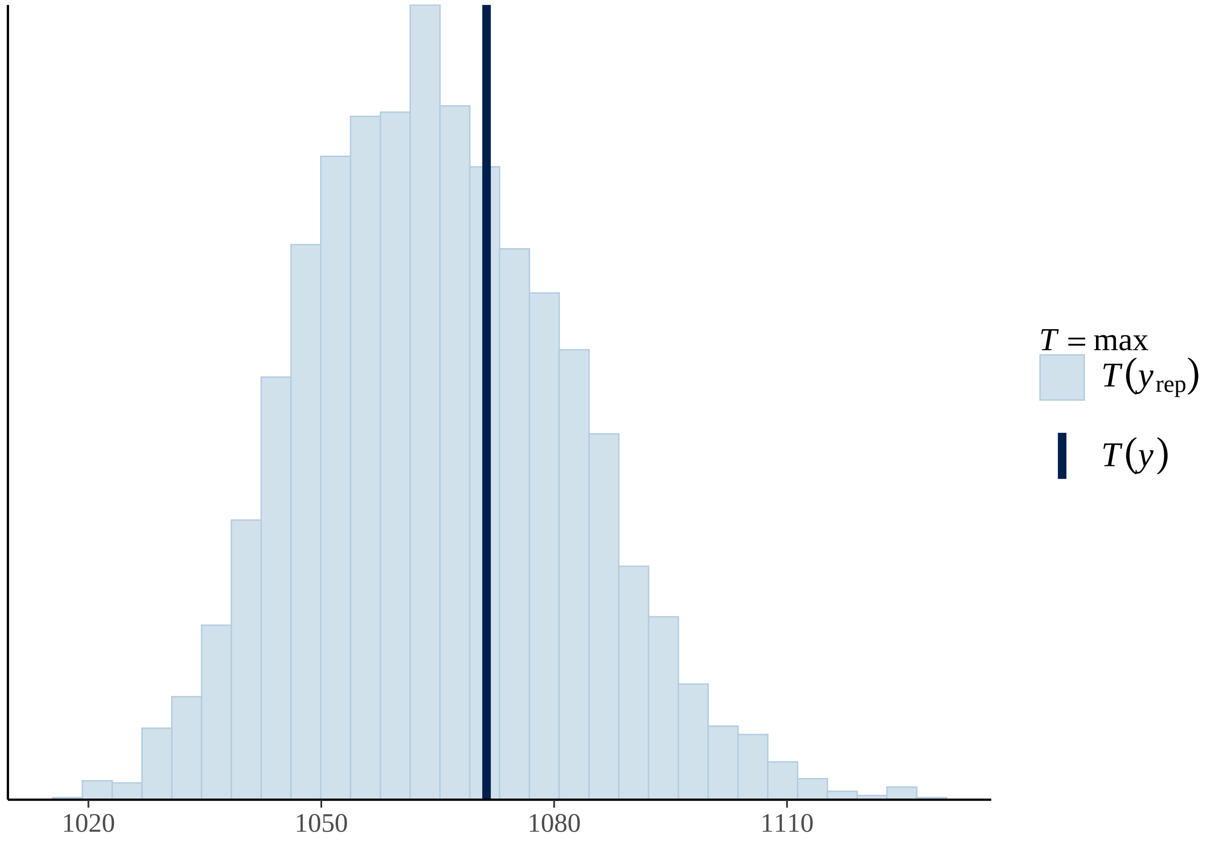
```
bayesplot::pp_check(modelo1, plotfun = "stat", stat = "min")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



```
bayesplot::pp_check(modelo1, plotfun = "stat", stat = "max")
```

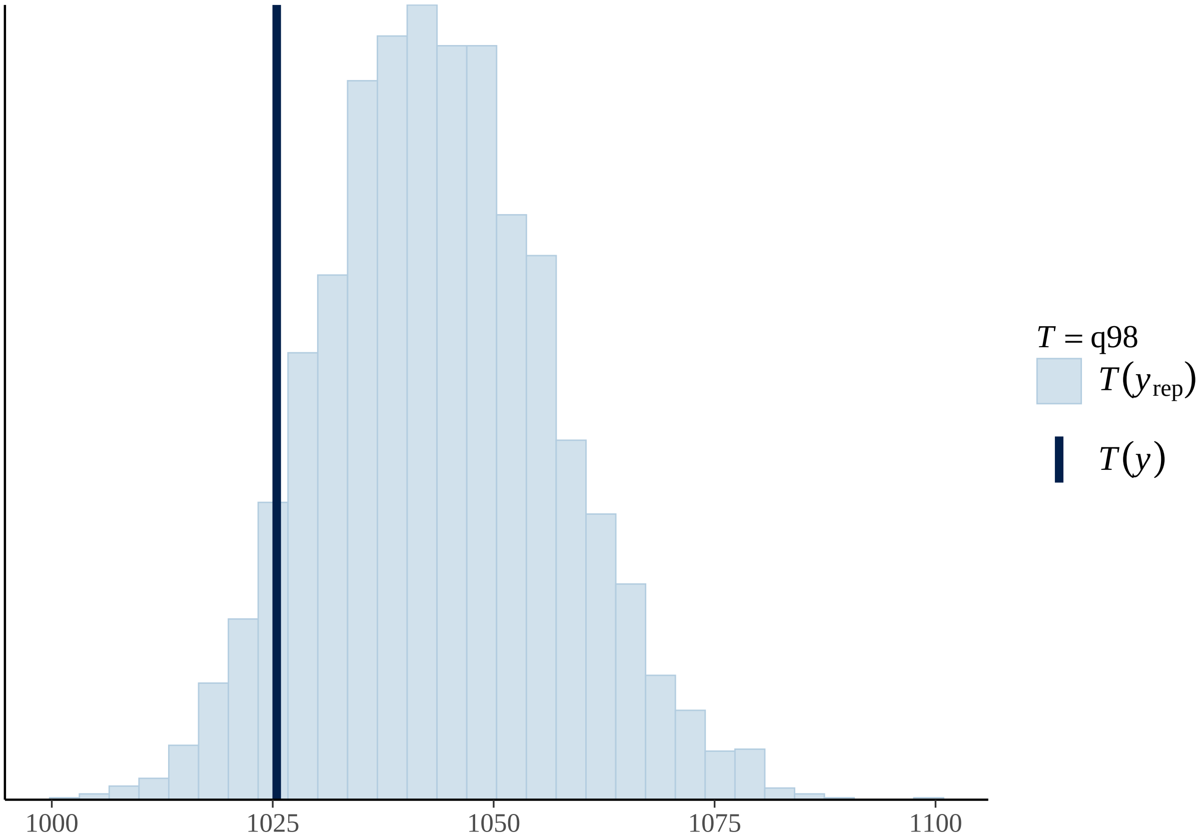
```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



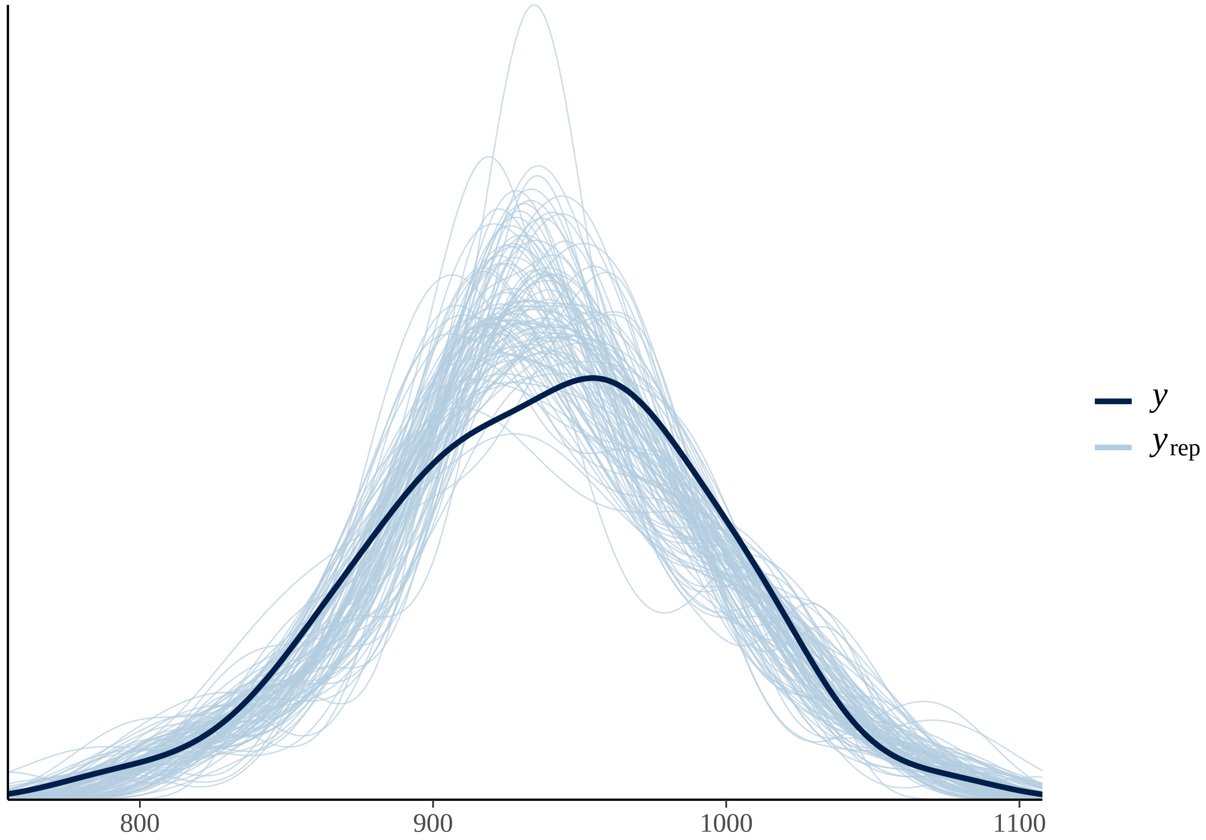
```
quant1 <- quantile(datosML$MORT, probs = seq(0,1, .01))
q98 <- function(y) quantile(y, 0.98)

bayesplot::pp_check(modelo1, plotfun = "stat", stat = "q98")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



```
bayesplot::ppc_dens_overlay(datosML$MORT,pred)
```


800
900
1000
1100
y
y_{rep}