

# Sesión 7: Modelo jerárquico con Openbugs y JAGS en R

## Aplicaciones en Computación Estadística

Natalie Julian - [www.nataliejulian.com](http://www.nataliejulian.com)

Estadística UC y Data Scientist en Zippedi Inc.

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

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

Los datos `school` se encuentran en la librería `R2openBUGS`.

Se busca generar simulaciones de la distribución posterior utilizando MCMC del siguiente modelo jerárquico:

$$Y_j \sim N(\theta_j, \tau_j)$$

$$\theta_j \sim N(\mu_\theta, \tau_\theta)$$

$$\tau_j \sim \frac{1}{\sigma_j^2}$$

Con distribuciones a priori:

$$\mu_\theta \sim N(0, 1) \quad \tau_\theta \sim \frac{1}{\sigma_\theta^2} \quad \sigma_\theta^2 \sim U(0, 1000)$$

# **Implementación con OpenBugs en R usando la librería R2OpenBUGS**

# Implementación usando OpenBugs

*Notar que es necesario tener instalado OpenBUGS previamente.*



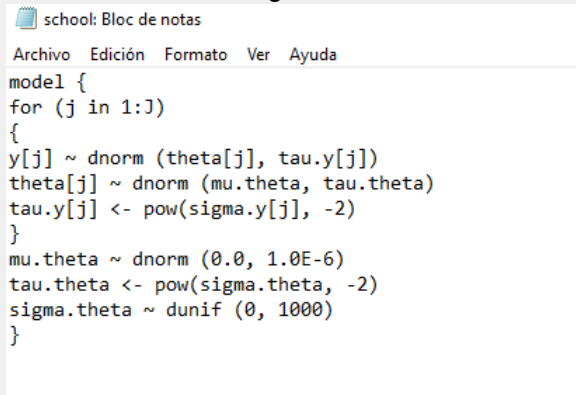
OpenBUGS

Aplicación

Link: <http://www.openbugs.net/w/Downloads>

```
model {  
  for (j in 1:J)  
  {  
    y[j] ~ dnorm (theta[j], tau.y[j])  
    theta[j] ~ dnorm (mu.theta, tau.theta)  
    tau.y[j] <- pow(sigma.y[j], -2)  
  }  
  mu.theta ~ dnorm (0.0, 1.0E-6)  
  tau.theta <- pow(sigma.theta, -2)  
  sigma.theta ~ dunif (0, 1000)  
}
```

Nuestro modelo será guardado en un archivo de tipo txt:



```
model {  
  for (j in 1:J)  
  {  
    y[j] ~ dnorm (theta[j], tau.y[j])  
    theta[j] ~ dnorm (mu.theta, tau.theta)  
    tau.y[j] <- pow(sigma.y[j], -2)  
  }  
  mu.theta ~ dnorm (0.0, 1.0E-6)  
  tau.theta <- pow(sigma.theta, -2)  
  sigma.theta ~ dunif (0, 1000)  
}
```

# Simulaciones MCMC

```
library(R2OpenBUGS)
data("schools")

J <- nrow(schools) #Se deben recorrer todas las filas
y <- schools$estimate #Corresponden a  $Y_j$ 
sigma.y <- schools$sd #Desviaciones estándar son

data <- list(J=J, y=y, sigma.y=sigma.y) #Se guardan los datos

#Los parámetros que tenemos son:
# -  $\theta_j$  (para cada  $y_j$ )  $\sim N(\mu.\theta, \tau.\theta)$ 
# -  $\mu.\theta$  (para  $\theta_j$ )  $\sim N(0, 1)$ 
# -  $\sigma.\theta$  (para  $\tau.\theta$ )  $\sim U(0, 1000)$ 

inits <- function()
  list(theta=rnorm(J,0,100), mu.theta=rnorm(1,0,100),
        sigma.theta=runif(1,0,100))

#Model.file es un archivo de tipo txt con el modelo a utilizar!

#Tendremos una simulación MCMC para  $\theta$ ,  $\mu.\theta$  y  $\sigma.\theta$ 

schools.sim <- bugs(data, inits,
  model.file = "C:/Users/HP/Desktop/Clases UC/2020-2/Introducción a la Computación/Ayudantía 7/school.txt",
  parameters = c("theta", "mu.theta", "sigma.theta"),
  n.chains = 3, n.iter = 1000)
```



Obtenemos simulaciones de nuestros parámetros:

```
print(schools.sim)
Inference for Bugs model at "C:/Users/HP/Desktop/Clases UC/2020-2/Introducción a la Computación/Ayudantía 7/school.txt",
Current: 3 chains, each with 1000 iterations (first 500 discarded)
Cumulative: n.sims = 1500 iterations saved
      mean sd 2.5% 25% 50% 75% 97.5% Rhat n.eff
theta[1] 11.7 7.7 -1.4  7.3 11.4 15.0  30.0  1.1  250
theta[2]  8.9 6.1 -4.2  5.1  9.9 12.4  19.2  1.1   20
theta[3]  7.6 7.2 -9.4  3.2  9.0 12.4  18.9  1.2   17
theta[4]  8.8 6.3 -4.8  4.6 10.0 12.4  20.1  1.1   20
theta[5]  6.7 6.6 -8.8  2.5  7.9 11.9  16.7  1.3   10
theta[6]  7.6 6.6 -7.4  3.1  8.8 12.3  17.5  1.2   13
theta[7] 11.2 6.1 -0.8  7.7 11.5 14.5  24.3  1.1  130
theta[8]  9.7 7.2 -5.1  5.5 10.7 13.1  24.9  1.1   50
mu.theta  8.9 5.1 -1.9  5.7  9.8 12.4  17.2  1.2   14
sigma.theta 5.3 5.4  0.0  0.9  3.7  8.2  18.0  1.7    6
deviance 60.5 2.0 57.1 59.3 60.4 61.4  64.9  1.0  160
```

For each parameter, n.eff is a crude measure of effective sample size,  
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule,  $pD = \bar{D} - \hat{D}$ )

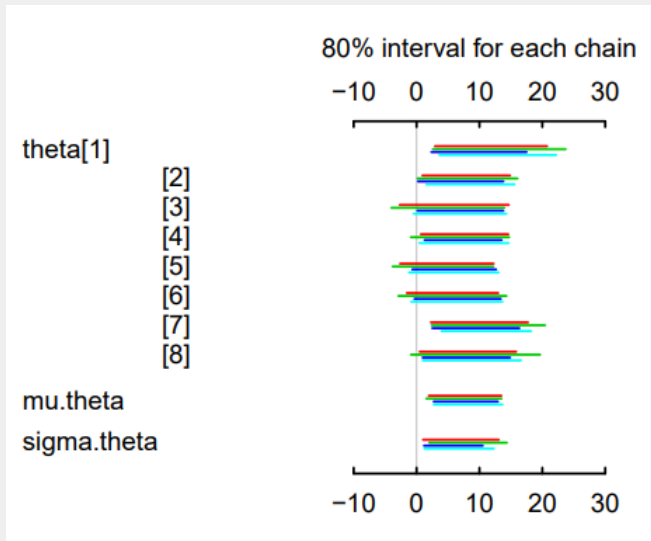
$pD = 2.5$  and  $DIC = 63.0$

DIC is an estimate of expected predictive error (lower deviance is better).

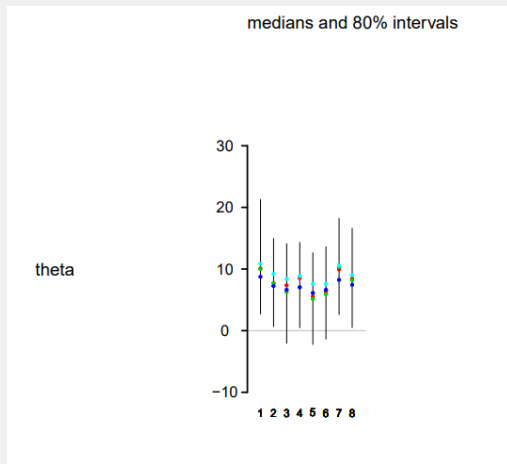
Con `plot(schools.sim)` podemos extraer resúmenes de las cadenas obtenidas.

# Gráficos por default

```
plot(school.sim)
```

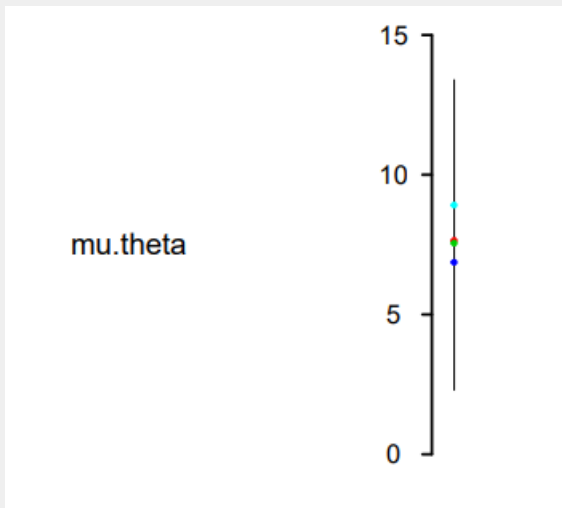


# Gráficos por default

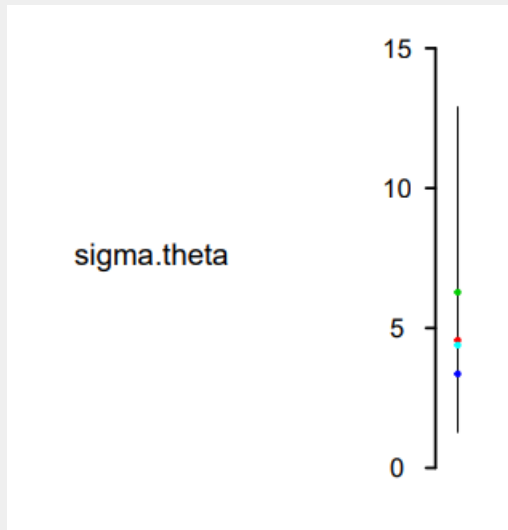


Se grafican los intervalos de credibilidad al 80% de la distribución a posteriori. Además las medianas para cada cadena.

# Gráficos por default

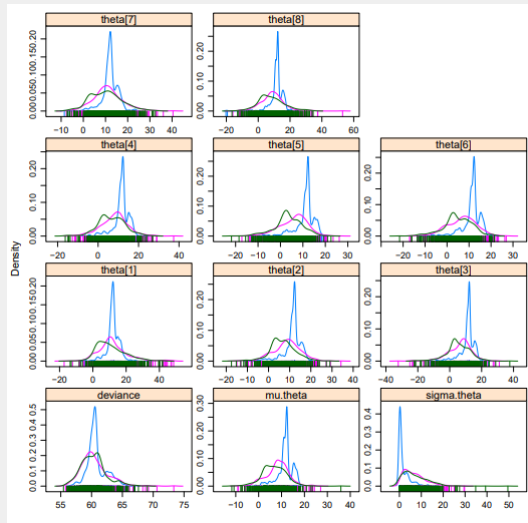


## Gráficos por default



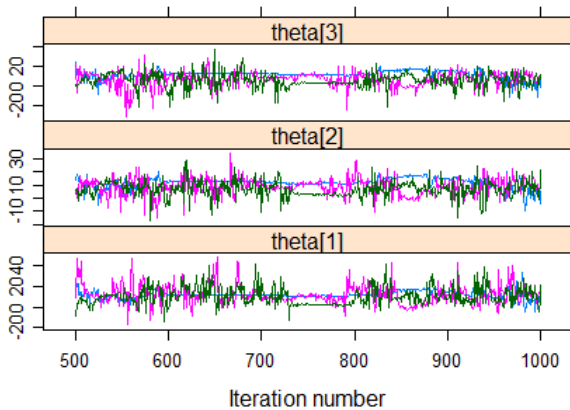
# Librería coda y lattice

```
library(lattice)  
library(coda)  
openbcoda<-as.mcmc.list(schools.sim)  
densityplot(openbcoda)
```



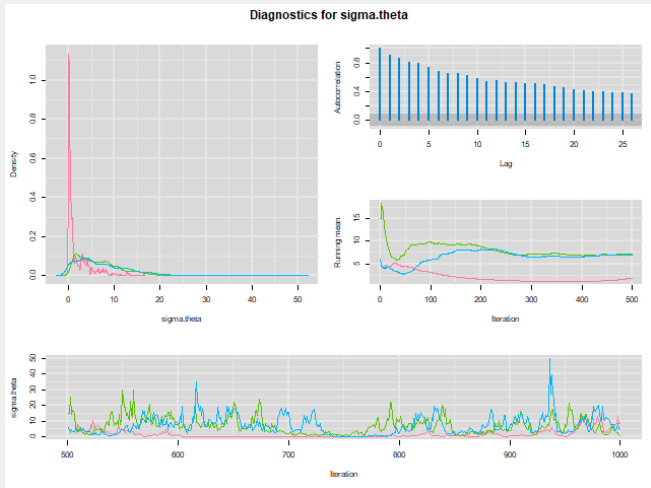
# Librería coda y lattice

```
xyplot(openbcoda[1:3][, 4:6]) #primeros 3 thetas
```



# Librería library(mcmcplots)

Esta es una librería muy interesante! Corriendo `mcmcplot(openbcoda)` te abre una nueva pestaña con **absolutamente todos** los gráficos para cada uno de los parámetros!

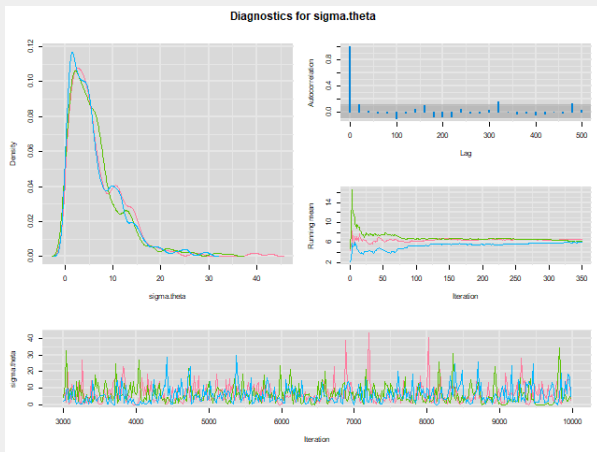




# Aumentar la cantidad de iteraciones y dar saltos

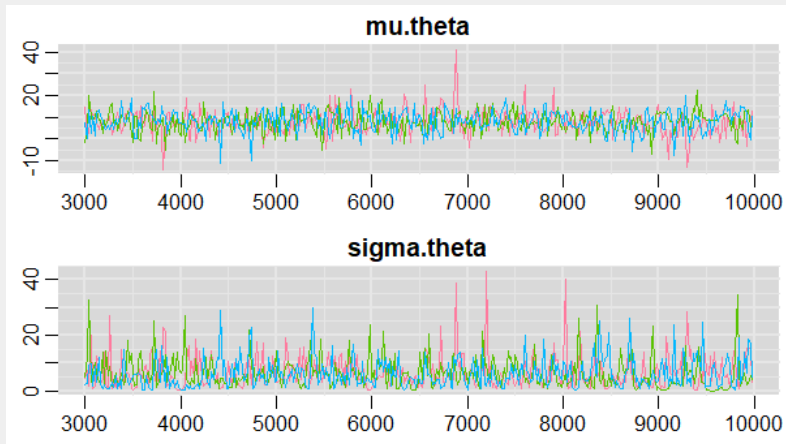
```
schools.simbigsaltos <- bugs(data, inits, model.file = "C:/Users/HP/Desktop/Clases UC/2020-2/Introducción a la Computación/Ayudantía 7/schools.model",  
  parameters = c("theta", "mu.theta", "sigma.theta"),  
  n.chains = 3, n.iter = 10000, n.burnin = 3000, n.thin = 20)
```

```
openbcodabigsaltos <- as.mcmc.list(schools.simbigsaltos)  
mcmcplot(openbcodabigsaltos)
```



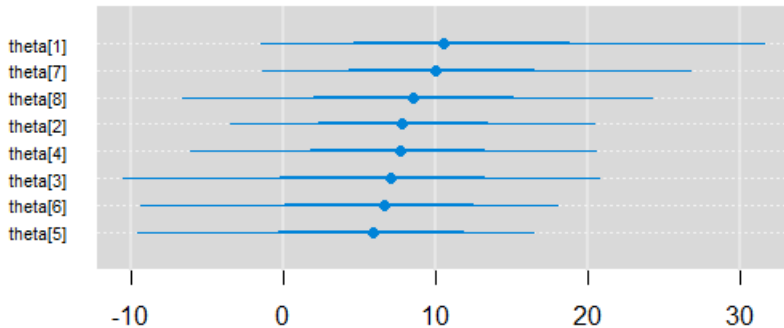
# Más gráficos

```
traplot(openbcodabigsaltos, parms = c("mu.theta", "sigma.theta"))
```



# Más gráficos

```
caterplot(openbcodabigsaltos, parms = paste("theta[", 1:8, "]", sep=""))#Intervalos
```



# Implementación con JAGS en R usando la librería rjags

# Implementación usando JAGS

*Notar que es necesario tener instalado JAGS previamente.*



JAGS 4.3.0 (32-bit)

Aplicación

Link: [sourceforge.net/projects/mcmc-jags/](http://sourceforge.net/projects/mcmc-jags/)

Nuestro modelo se guardará mediante una conexión, es el mismo utilizando anteriormente:

```
model <- textConnection("  
model {  
  for (j in 1:J)  
  {  
    y[j] ~ dnorm (theta[j], tau.y[j])  
    theta[j] ~ dnorm (mu.theta, tau.theta)  
    tau.y[j] <- pow(sigma.y[j], -2)  
  }  
  mu.theta ~ dnorm (0.0, 1.0E-6)  
  tau.theta <- pow(sigma.theta, -2)  
  sigma.theta ~ dunif (0, 1000)  
}  
")
```

```
library(rjags)

schools.sim <- jags.model(file=model, data, inits, n.chains=3)

# Quema
update(schools.sim, 2000)

post.samples <- coda.samples(schools.sim,
variable.names= c("theta", "mu.theta", "sigma.theta"), n.iter=10000, thin=20)

summary(post.samples)
```

# coda.samples

```
Iterations = 11020:21000
Thinning interval = 20
Number of chains = 3
Sample size per chain = 500
```

1. Empirical mean and standard deviation for each variable,  
plus standard error of the mean:

	Mean	SD	Naive SE	Time-series SE
mu.theta	8.153	5.323	0.1374	0.1584
sigma.theta	6.386	5.328	0.1376	0.1912
theta[1]	11.462	8.180	0.2112	0.2457
theta[2]	7.952	6.223	0.1607	0.1640
theta[3]	6.384	7.835	0.2023	0.2199
theta[4]	7.979	6.354	0.1641	0.1641
theta[5]	5.729	6.265	0.1618	0.2025
theta[6]	6.128	7.062	0.1823	0.1870
theta[7]	10.527	7.010	0.1810	0.2092
theta[8]	8.911	8.078	0.2086	0.2267

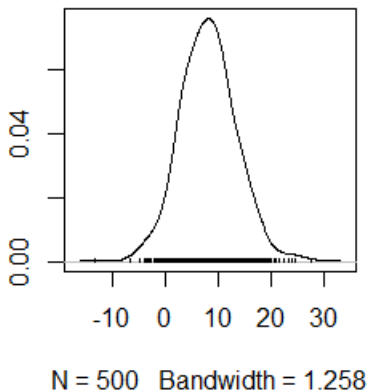
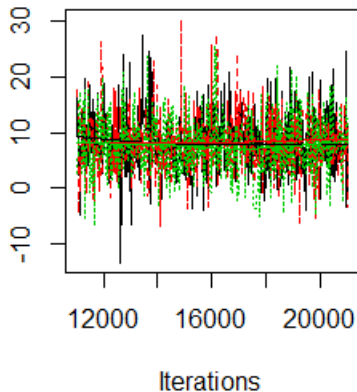
2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
mu.theta	-2.0190	4.576	8.080	11.443	18.63
sigma.theta	0.2447	2.230	5.017	8.955	19.79
theta[1]	-1.8038	5.914	10.493	15.330	31.64
theta[2]	-4.1424	4.172	7.912	11.781	20.12
theta[3]	-10.8130	1.871	6.848	11.258	20.91
theta[4]	-4.1467	3.930	7.966	12.006	21.10
theta[5]	-7.6960	2.082	6.099	9.905	16.59
theta[6]	-10.0584	1.967	6.423	10.924	18.83
theta[7]	-2.6147	5.896	10.132	14.346	26.24
theta[8]	-6.3004	4.029	8.479	13.152	27.39



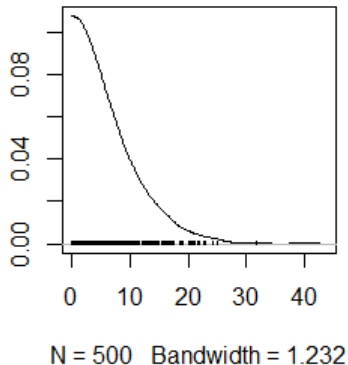
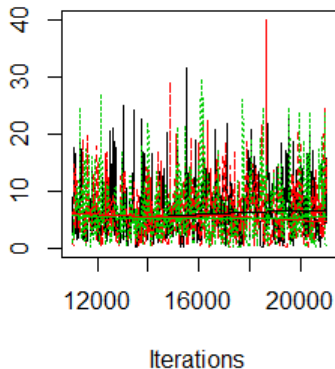
# Gráficos por default

```
plot(post.samples[1:3][,1]) #Cadena 1, 2 y 3 para mu.theta
```



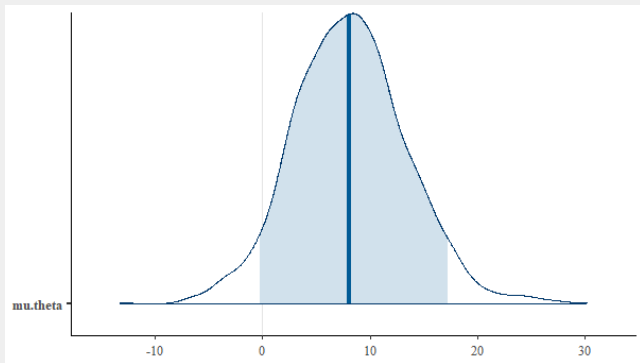
# Gráficos por default

```
plot(post.samples[1:3][,2]) #Cadena 1, 2 y 3 para sigma.theta
```



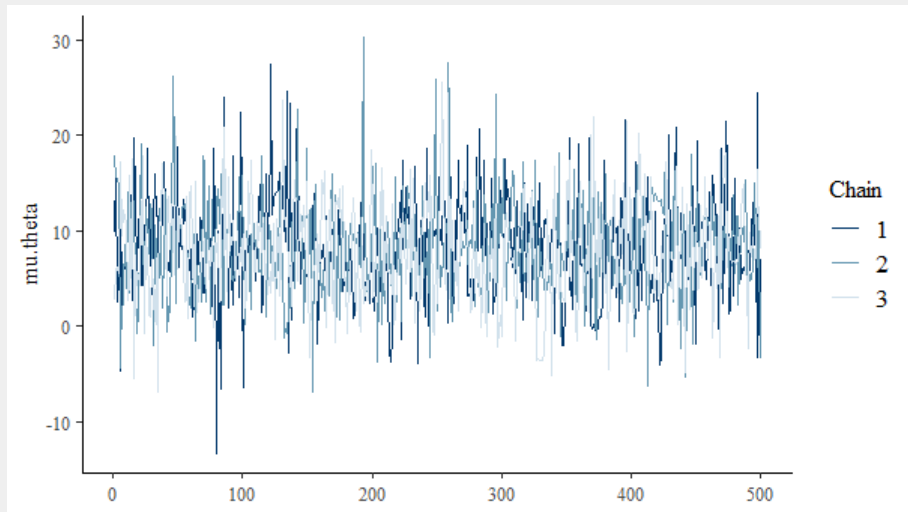
# Librería bayesplot

```
library(bayesplot)
mcmc_areas(
  post.samples,
  pars = c("mu.theta"),
  prob = 0.90)
```



# Librería bayesplot

```
library(bayesplot)  
mcmc_trace(post.samples, pars = "mu.theta")
```



# Customizando con otras librerías

```
library(dplyr)
library(ggplot2)
library(ggformula)

mcmc_trace(post.samples, pars = "mu.theta") %>%
  gf_facet_grid(chain ~ .) %>%
  gf_refine(scale_color_viridis_d())
```

