aneur: comparando stan y cgam

1. Aortic Aneurysm Progression Data

This dataset contains longitudinal measurements of grades of aortic aneurysms, measured by ultrasound examination of the diameter of the aorta.

A data frame containing 4337 rows, with each row corresponding to an ultrasound scan from one of 838 men over 65 years of age.

- ptnum (numeric) Patient identification number
- age (numeric) Recipient age at examination (years)
- diam (numeric) Aortic diameter
- state (numeric) State of aneurysm.

The states represent successive degrees of aneurysm severity, as indicated by the aortic diameter.

- State 1 Aneurysm-free < 30 cm
- State 2 Mild aneurysm 30-44 cm
- State 3 Moderate aneurysm 45-54 cm
- State 4 Severe aneurysm > 55 cm

683 of these men were aneurysm-free at age 65 and were re-screened every two years. The remaining men were aneurysmal at entry and had successive screens with frequency depending on the state of the aneurysm. Severe aneurysms are repaired by surgery.

```
data(aneur)
attach(aneur)
head(aneur)
```

```
##   ptnum      age diam state
## 1     1 60.00000   29     1
## 2     1 65.47671   29     1
## 3     1 67.50411   29     1
## 4     1 70.04384   29     1
## 5     1 72.07671   29     1
## 6     1 74.08767   29     1
```

```
tail(aneur)
```

```
##      ptnum      age diam state
## 4332   838 73.40822   43     2
## 4333   838 73.61644   43     2
## 4334   838 73.87671   42     2
## 4335   838 74.05753   43     2
## 4336   838 74.31507   41     2
## 4337   838 74.56712   40     2
```

```r
#help(aneur)
dim(aneur)
```

```
## [1] 4337    4
```

```r
(N = n_distinct(aneur$ptnum))   # subjects
```

```
## [1] 838
```

```r
(K = max(table(aneur$ptnum)))   # times
```

```
## [1] 21
```

```r
table(table(aneur$ptnum))
```

```
##
##   2   3   4   5   6   7   8   9  10  11  12  14  15  16  17  18  19  21
## 121 107  99  96 260  97  12  12   9   5   2   5   5   3   1   2   1   1
```

```r
J = 4   # categories
Y_diam = array(NA,dim=c(N,K))
Y_state = array(NA,dim=c(N,K))
X_age = array(NA,dim=c(N,K))
Ki = table(aneur$ptnum)
Ni = c(0,cumsum(Ki))+1
for(i in 1:N){
    aneur_i = aneur[aneur$ptnum==i,]
    for(k in 1:Ki[i]){
        Y_diam[i,k] = aneur_i$diam[k]
        Y_state[i,k] = aneur_i$state[k]
        X_age[i,k] = aneur_i$age[k]
    }
}
```

```r
(Y_diam[11:18,1:8])
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]   29   29   29   29   29   29   29   NA
## [2,]   29   29   29   29   29   29   29   NA
## [3,]   29   29   29   29   29   29   29   NA
## [4,]   29   29   NA   NA   NA   NA   NA   NA
## [5,]   29   29   29   29   29   29   29   NA
## [6,]   29   29   29   29   29   29   29   NA
## [7,]   29   29   29   29   NA   NA   NA   NA
## [8,]   29   29   34   NA   NA   NA   NA   NA
```

```r
(Y_state[11:18,1:8])
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    1    1    1    1    1    1    1   NA
## [2,]    1    1    1    1    1    1    1   NA
## [3,]    1    1    1    1    1    1    1   NA
## [4,]    1    1   NA   NA   NA   NA   NA   NA
## [5,]    1    1    1    1    1    1    1   NA
## [6,]    1    1    1    1    1    1    1   NA
## [7,]    1    1    1    1   NA   NA   NA   NA
## [8,]    1    1    2   NA   NA   NA   NA   NA
```

```r
(X_age[11:18,1:8])
```

```
##      [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7] [,8]
## [1,]   60 65.45205 67.45205 69.92877 72.01096 74.01096 76.00000   NA
## [2,]   60 65.44932 67.46301 69.92603 71.96986 73.96986 75.92055   NA
## [3,]   60 65.45753 67.44658 69.92329 71.96712 73.96712 75.91781   NA
## [4,]   60 65.44384       NA       NA       NA       NA       NA   NA
## [5,]   60 65.43836 67.42192 69.93699 71.94247 73.94247 75.89315   NA
## [6,]   60 65.40822 67.40822 70.04932 72.07123 74.06575 76.09041   NA
## [7,]   60 65.38082 67.38082 70.02192       NA       NA       NA   NA
## [8,]   60 65.47123 67.47123       NA       NA       NA       NA   NA
```

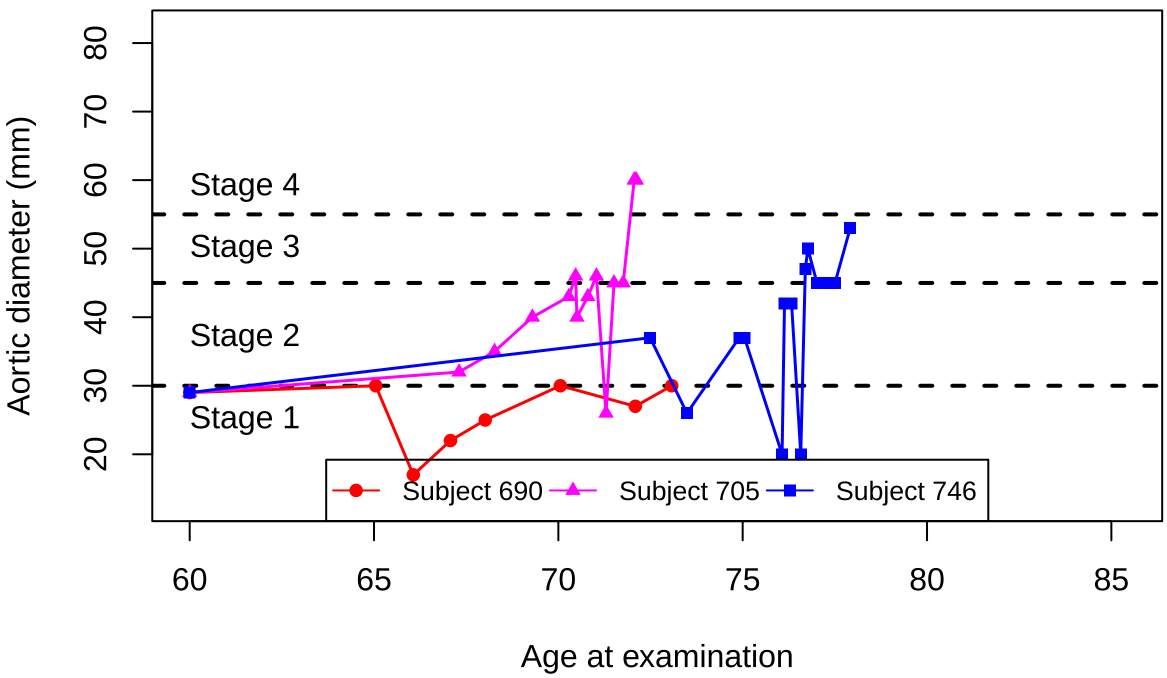
```r
(Ki[11:18])
```

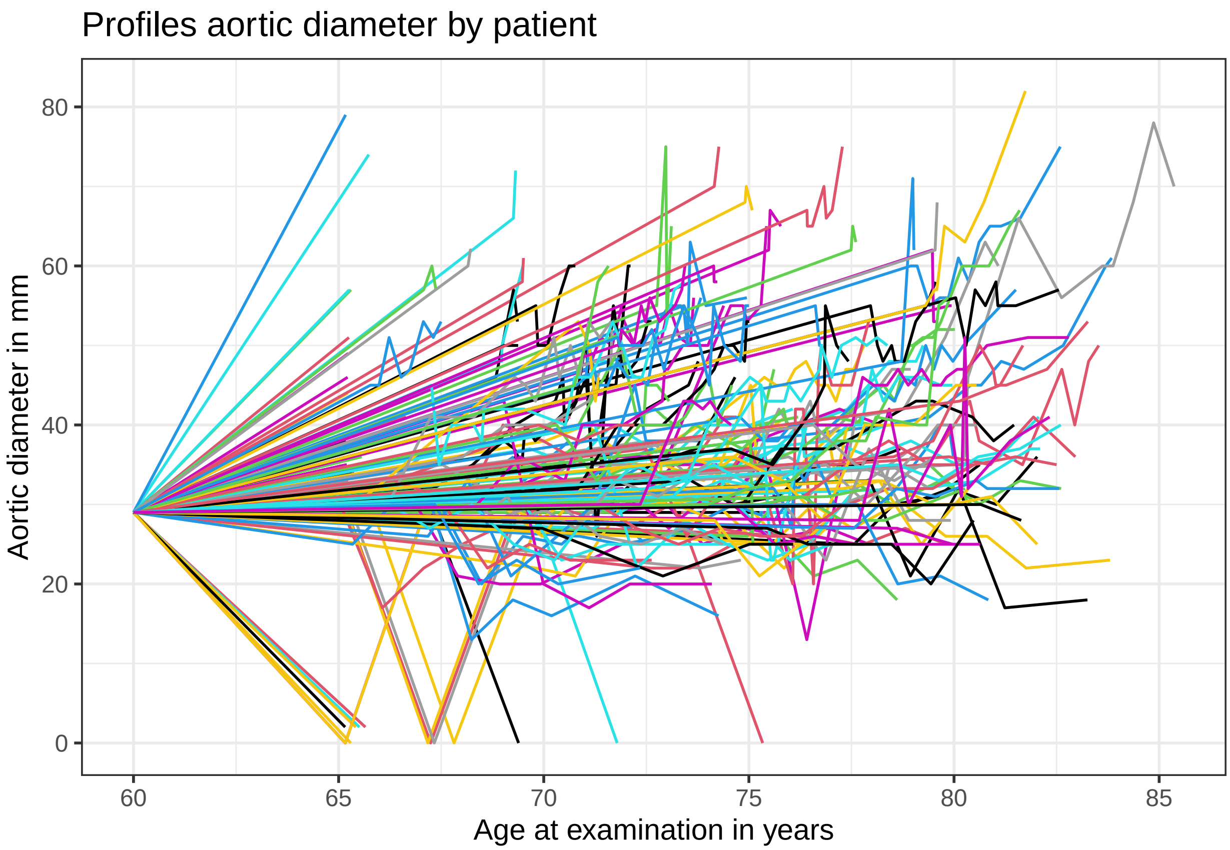
```
##
## 11 12 13 14 15 16 17 18
##  7  7  7  2  7  7  4  3
```

```r
### Considering only data having more than one screen (state>1)
idx2 = c()
for(i in 1:N){
  if( sum(Y_state[i,1:Ki[i]])>Ki[i]){
    idx2 = c(idx2,i)
  }
}
Y2_diam = Y_diam[idx2,]
Y2_state = Y_state[idx2,]
X2_age = X_age[idx2,]
N2 = length(idx2)
Ki2 = Ki[idx2]

### Considering only data having more than one screen (diam!=29, or diam<29 & dim>29)
idx3 = c()
for(i in 1:N){
  if( min(Y_diam[i,1:Ki[i]])!=max(Y_diam[i,1:Ki[i]])){
    idx3 = c(idx3,i)
  }
}
Y3_diam = Y_diam[idx3,]
```

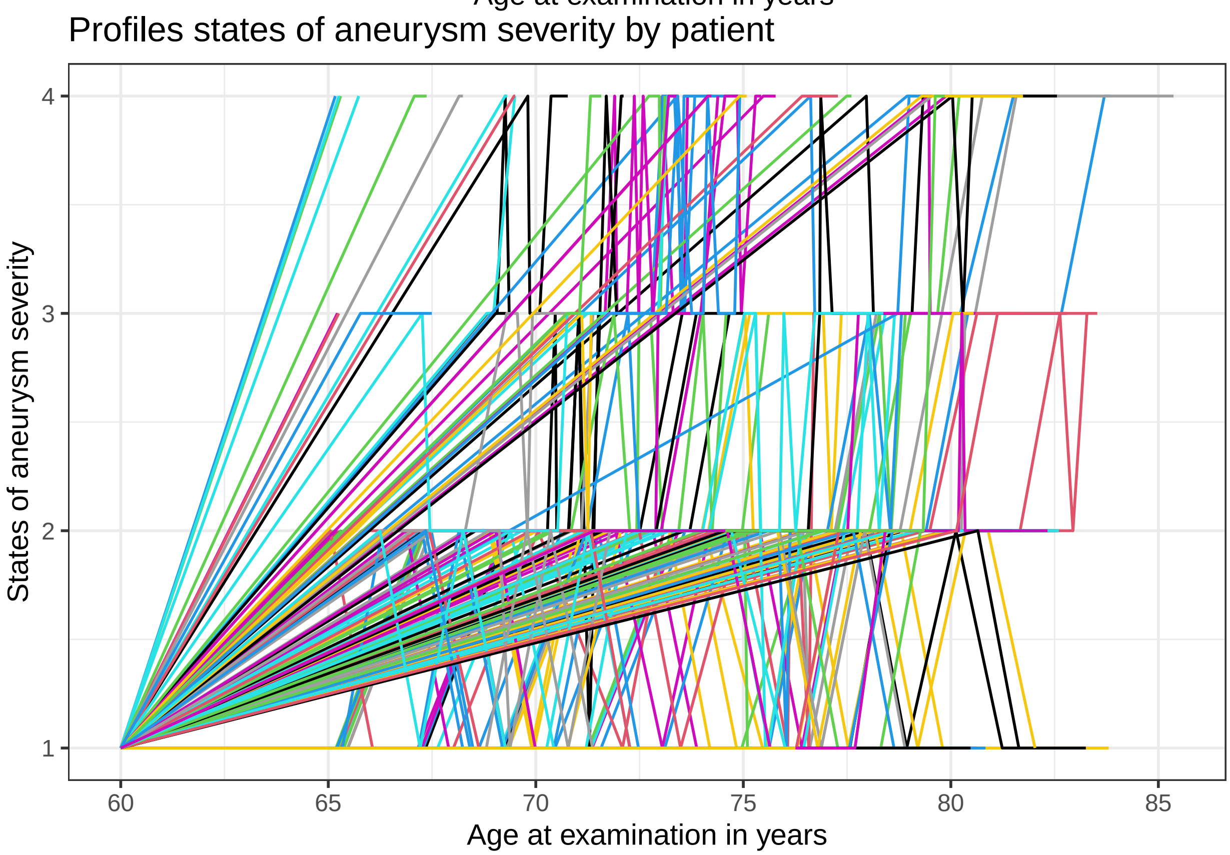
```r
Y3_state = Y_state[idx3,]
X3_age = X_age[idx3,]
N3 = length(idx3)
Ki3 = Ki[idx3]

aneur2 = aneur%>%filter(aneur$ptnum%in%idx2)
aneur3 = aneur%>%filter(aneur$ptnum%in%idx3)
### Creo que es mejor trabajar con aneur3
```

```
##
##  67  80 119
##   6   6   6
```

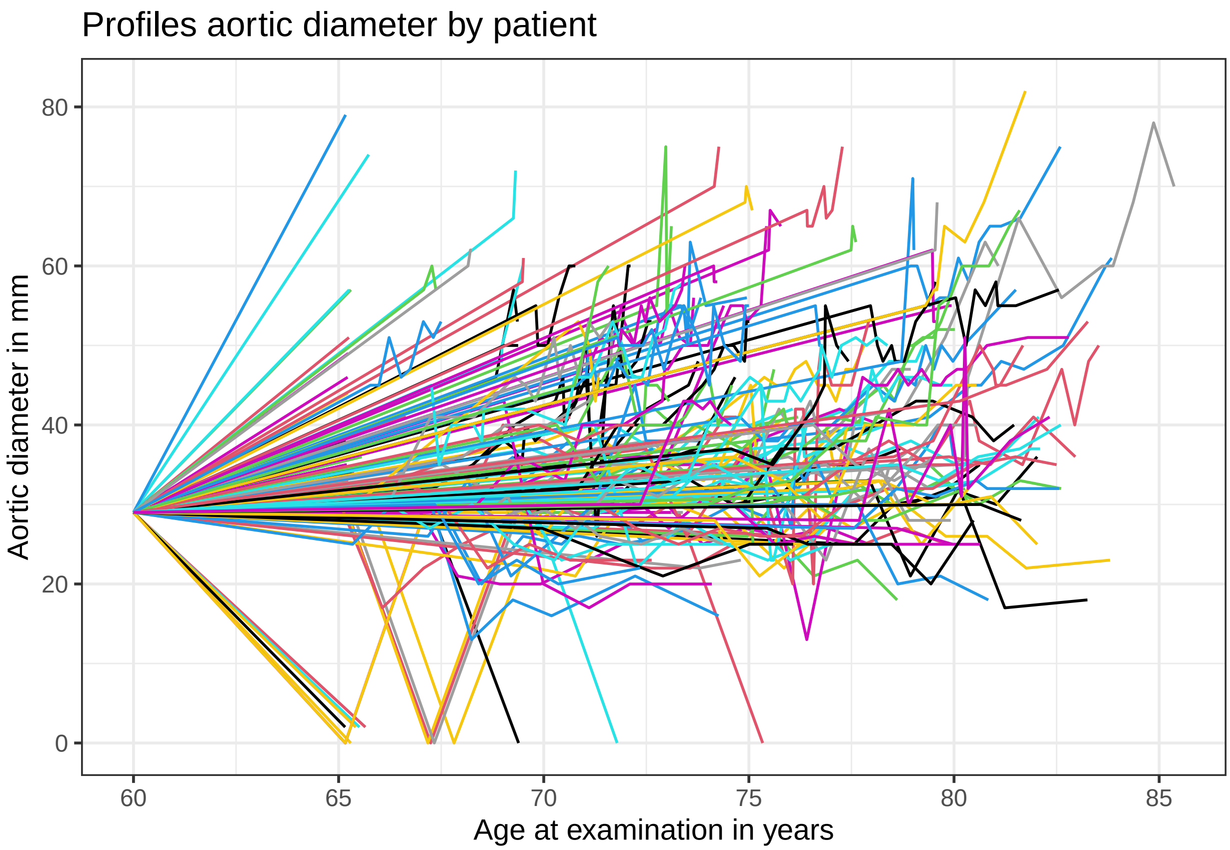


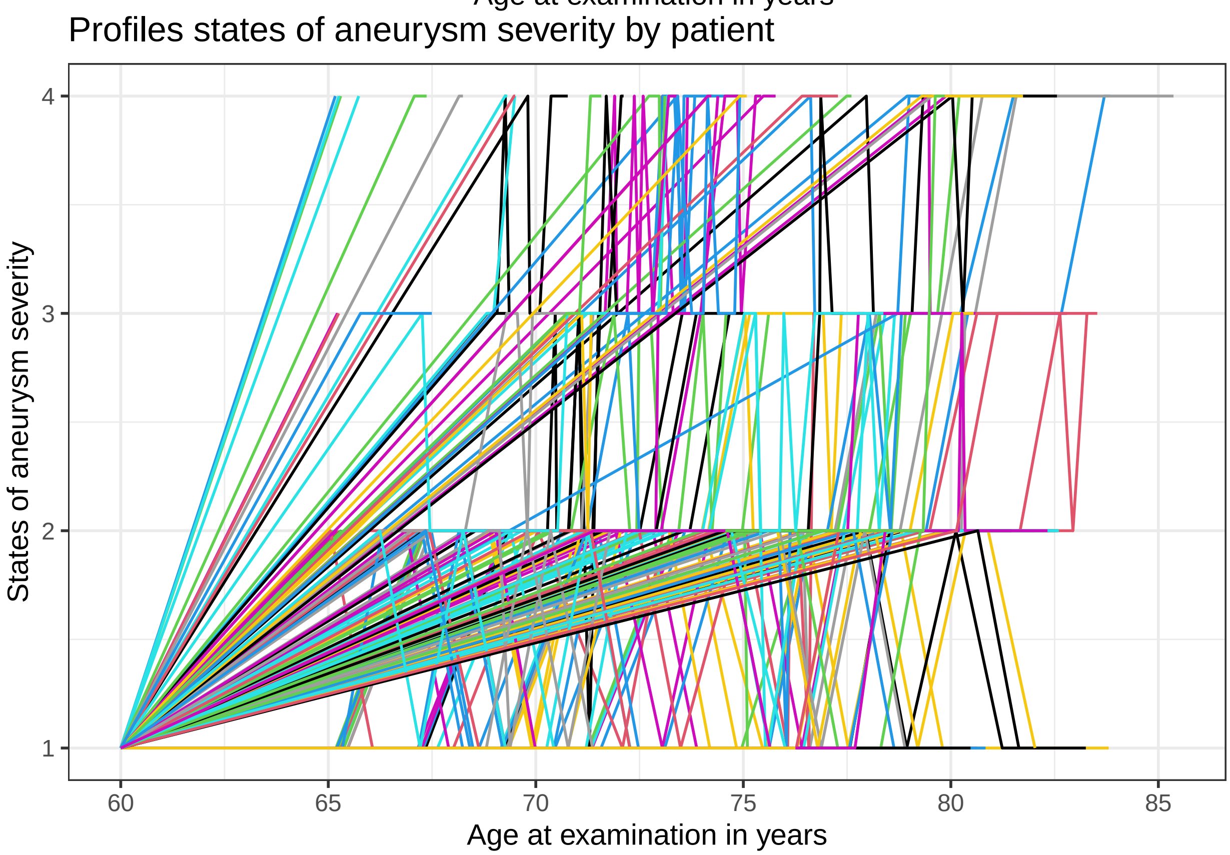


Profiles aortic diameter by patient
Aortic diameter in mm
0
20
40
60
80
60
65
70
75
80
85
Age at examination in years



Profiles states of aneurysm severity by patient
States of aneurysm severity
1
2
3
4
60
65
70
75
80
85
Age at examination in years





La variable respuesta puede ser continua (''diam'') u ordinal (''state''), y la unica covariable es la edad ("age") \

$$diam_{it} \quad = \quad \beta_0 + f_1(age_{it}) + b_{0i} + age_{it} \times b_{1i} + \varepsilon_{it}, \qquad b_i \sim \mathrm{N}(0,\psi), \quad \varepsilon_i \sim \mathrm{N}(0, \Lambda\sigma^2),$$

where f_1 is a non-decreasing smoothing function and $b_{1i} > 0$.

Quiza solo debemos considerar intercepto fijo, pero NO intercepto aleatorio, y SI pendiente aleatorio

$$diam_{it} \quad = \quad \beta_0 + f_1(age_{it}) + age_{it} \times b_{1i} + \varepsilon_{it}, \qquad b_{1i} \sim \mathrm{N}(0,\psi), \quad \varepsilon_i \sim \mathrm{N}(0, \Lambda\sigma^2),$$

The ordinal response $state_{it}$ is modelled in terms of the cumulative probabilities $P(state_{it} \leq j|b_i)$ by using the proportional odds model,

$$P(state_{it} \leq j|b_i) \quad = \quad \eta_{it,j},$$

subject to

$$\eta_{it,j} \quad = \quad \kappa_j + \beta_0 + f_1(age_{it}) + age_{it} \times b_{1i}, \qquad b_{1i} \sim \mathrm{N}(0,\psi),$$

where the constraints are such that f_1 is a non-decreasing smoothing function and $b_{1i} > 0$, and for the breakpoints $\kappa_j < \kappa_{j+1}$ with $j = 1, 2$.

```
y = aneur3$diam -29
x1 = aneur3$age -60
x2 = aneur3$age -60
id = as.numeric(as.factor(aneur3$ptnum))

n = length(y)
N = n_distinct(id)
Ni = c(0,cumsum(table(id)))+1
k1 = 3 #
k2 = 3 #
knots1 = quantile(x1, c(0.5))
knots2 = quantile(x2, c(0.5))
```

2. Generar la matriz diseño X para los B-splines

Note que $f(x)$ se representa como:

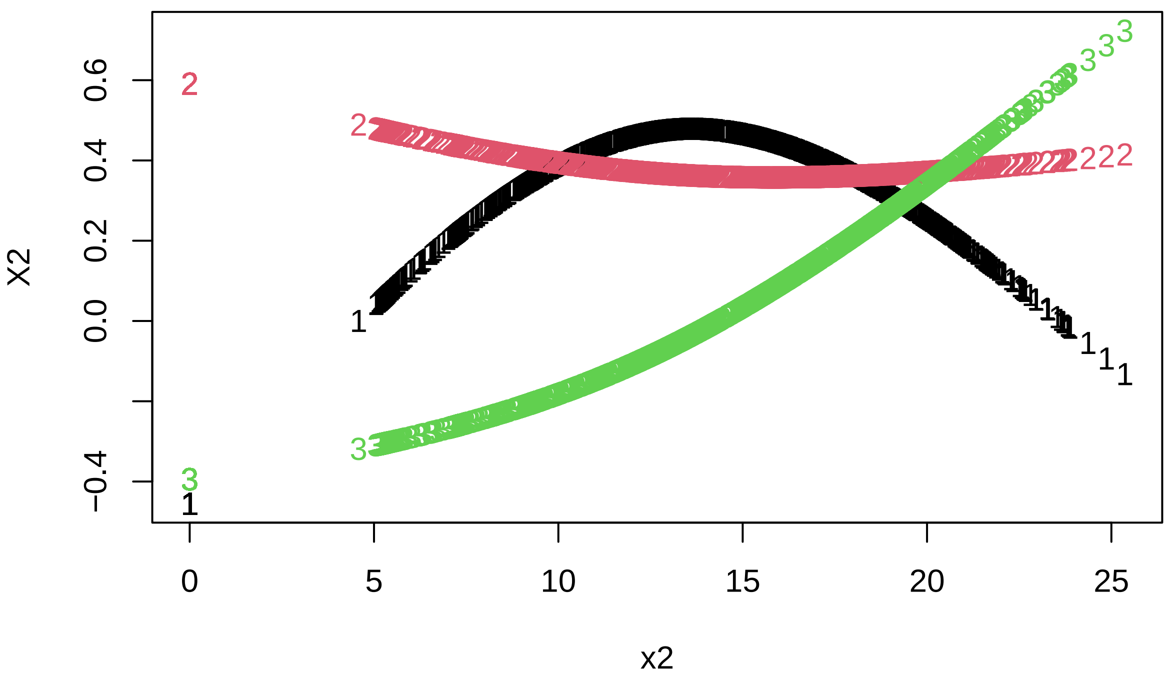
$$\begin{aligned} f(x) &= f_1(x_1) \\ &= \sum_{j=1}^{h_1} \beta_{1j} I_{1j}(x) \end{aligned}$$

para β_{1j} parámetros desconocidos, y para los $I_{1j}(x)$ se utilizar'an I-splines y B-splines.

El número de knots se elige lo suficientemente grande para evitar **over-smoothing**, pero lo suficientemente pequeño para evitar excesivo costo computacional.

El número de *knots* K es considerado a priori.

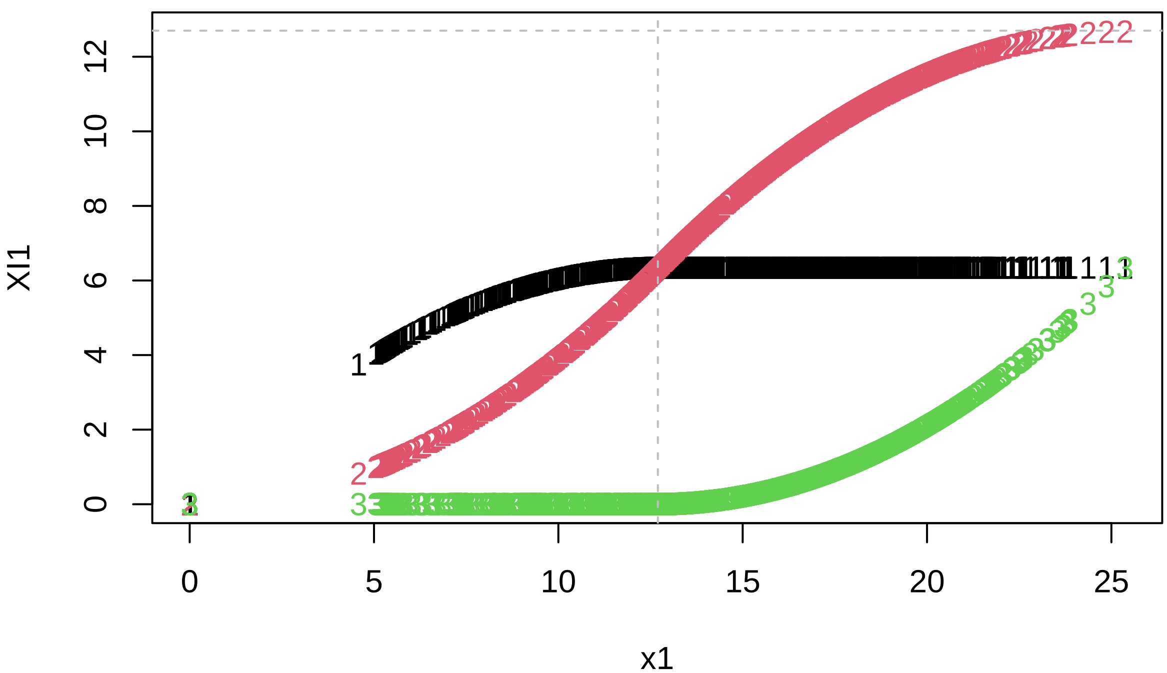
```
# Generate a basis matrix for Natural Cubic Splines
X2 <- ns(x = x2, knots = knots2, intercept = TRUE)
###X2 = (X2-mean(X2))/sd(X2)
matplot(x2, X2)
```



3. Generar la matriz diseño $XI1$ para los I-splines

$$
\begin{aligned}
f_1(x_1) & \quad \sum_{j=1}^{h_1} \beta_{1j} I_{1j}(x_1) \\
I_{1j}(x_1) & = \int_{x_0}^{x_1} B_{1j}(u) d_u
\end{aligned}
$$

```
### ibs: integrated basis splines
### degree = 3 cubic splines
XI1 <- ibs(x1, knots = knots1, degree = 1, intercept = TRUE)
###XI1 = (XI1-mean(XI1))/sd(XI1)
matplot(x1, XI1)
abline(v = knots1, h = knots1, lty = 2, col = "gray")
```



4. Definir la penalización $S1$ y $S2$

La flexibilidad ajustada de f es controlada por K, a través de una penalización cuadrática de la forma:

$$\sum_j \lambda_j \beta^T S_j \beta$$

donde los S_j son matrices de coeficientes conocidos, y los λ_j son parámetros de suavizamiento estimados.

```
#Este es el código que produce la matriz de diferenciación.
#No es el óptimo, pero funciona.
#"k" es el número de b-splines y
#"d" el orden de la diferenciación.
#Adjunto el artículo donde discutimos esto (página 7).

diffMatrix = function(k, d = 2){
  if( (d<1) || (d %% 1 != 0) )stop("d must be a positive integer value");
  if( (k<1) || (k %% 1 != 0) )stop("k must be a positive integer value");
  if(d >= k)stop("d must be lower than k");
  out = diag(k);
  for(i in 1:d){
    out = diff(out);
  }
  return(out)
}
(D1 = diffMatrix(k=k1, d=2))
```

```
##      [,1] [,2] [,3]
## [1,]    1   -2    1
```

```
(D2 = diffMatrix(k=k2, d=2))
```

```
##      [,1] [,2] [,3]
## [1,]    1   -2    1
```

```
(S1 = t(D1)%*%D1 + diag(1,k1)*10e-4)
```

```
##        [,1]   [,2]   [,3]
## [1,]  1.001 -2.000  1.000
## [2,] -2.000  4.001 -2.000
## [3,]  1.000 -2.000  1.001
```

```
(S2 = t(D2)%*%D2 + diag(1,k2)*10e-4)
```

```
##        [,1]   [,2]   [,3]
## [1,]  1.001 -2.000  1.000
## [2,] -2.000  4.001 -2.000
## [3,]  1.000 -2.000  1.001
```

8. Spline con restricciones creciente

8.2. LME: Spline con restricciones creciente

```
datos.lme.add.incr <- list( y = y ,
                           id = id ,
              n = length(y) ,
              N = N , Ni = Ni,
              k1=k1,
             XI1 = XI1,
             x1 = x1,
             zero = rep(0,1+k1),
             S1=S1  )

inits.lme.add.incr <- function(){   list(
  "b0" = rnorm(1,0,0.1) ,
  "b1" = abs(rnorm(k1,0,0.1)),
  "invtau2" = rgamma(1,1,1) ,
  "lambda" = rgamma(1,1,1) ,
  "invsig2" = rgamma(1,1,1)
)   }

param.add = c("b0","b1", "invtau2","tau2","tau", "lambda","rho")
param.lme.add = c("b0","b1", "invtau2","tau2","tau", "lambda","rho", "invsig2","sig2","sigma")
```

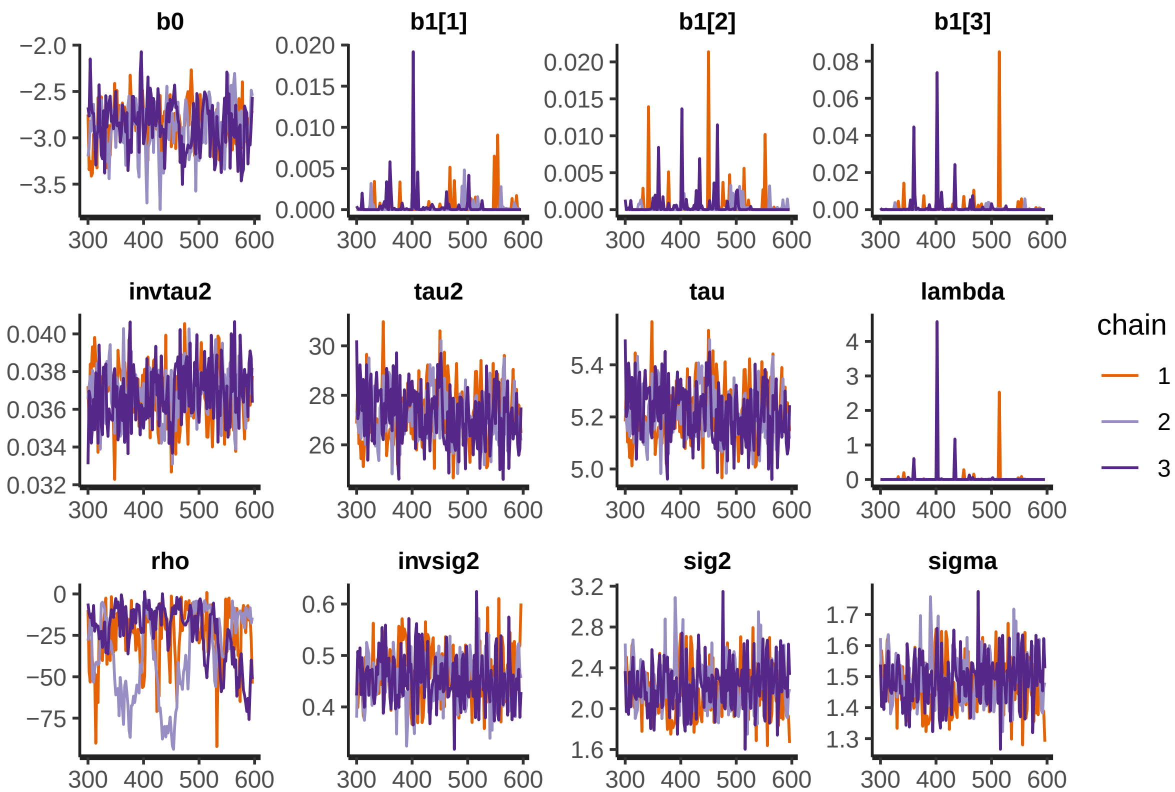
```
fit.lme.add.incr.reslope <- stan("jagam_9_aneur_lme_add_incr_reslope.stan",
            data=datos.lme.add.incr,
            chains=3,warmup=300,iter=600,thin=2,cores=4,
            init= inits.lme.add.incr)
```

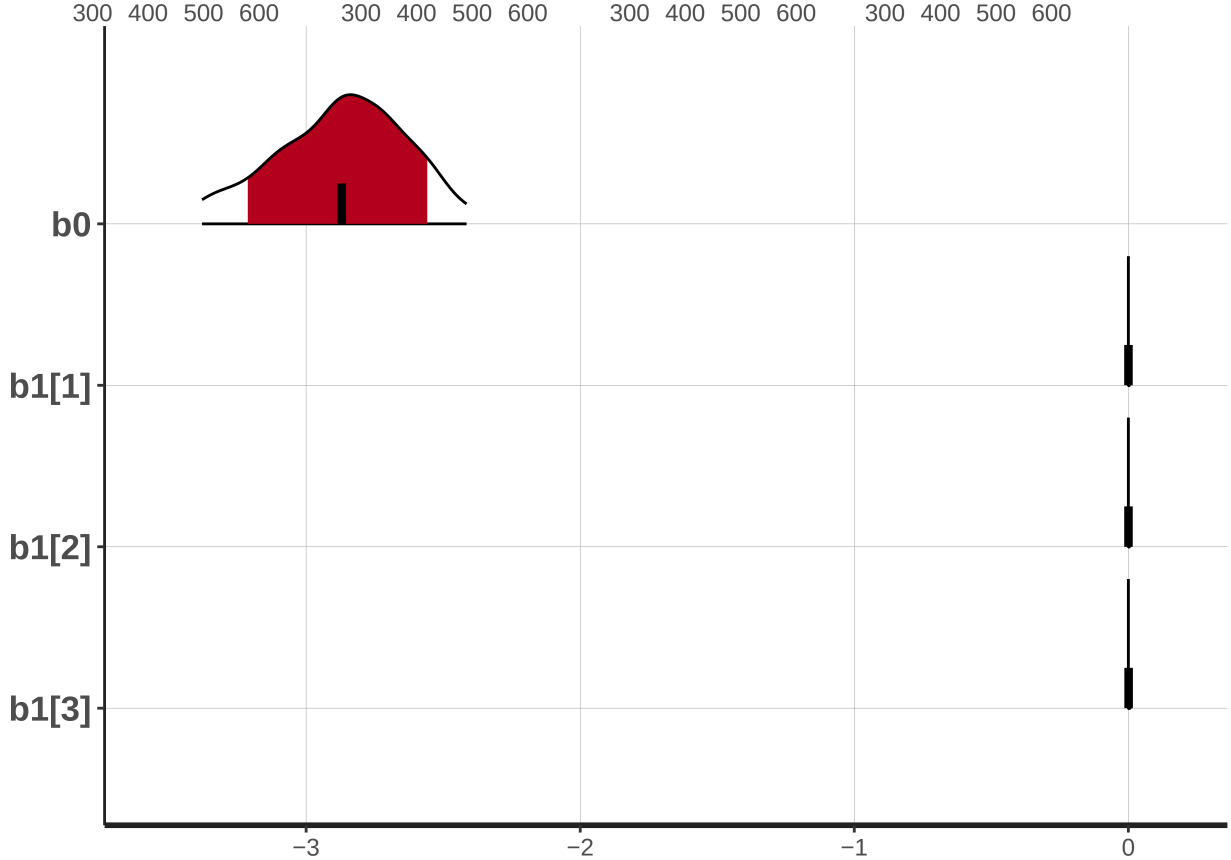
```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -mmacosx-version-min=10.13 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
##                ^
##                ;
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/Core:96:10: fa
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```

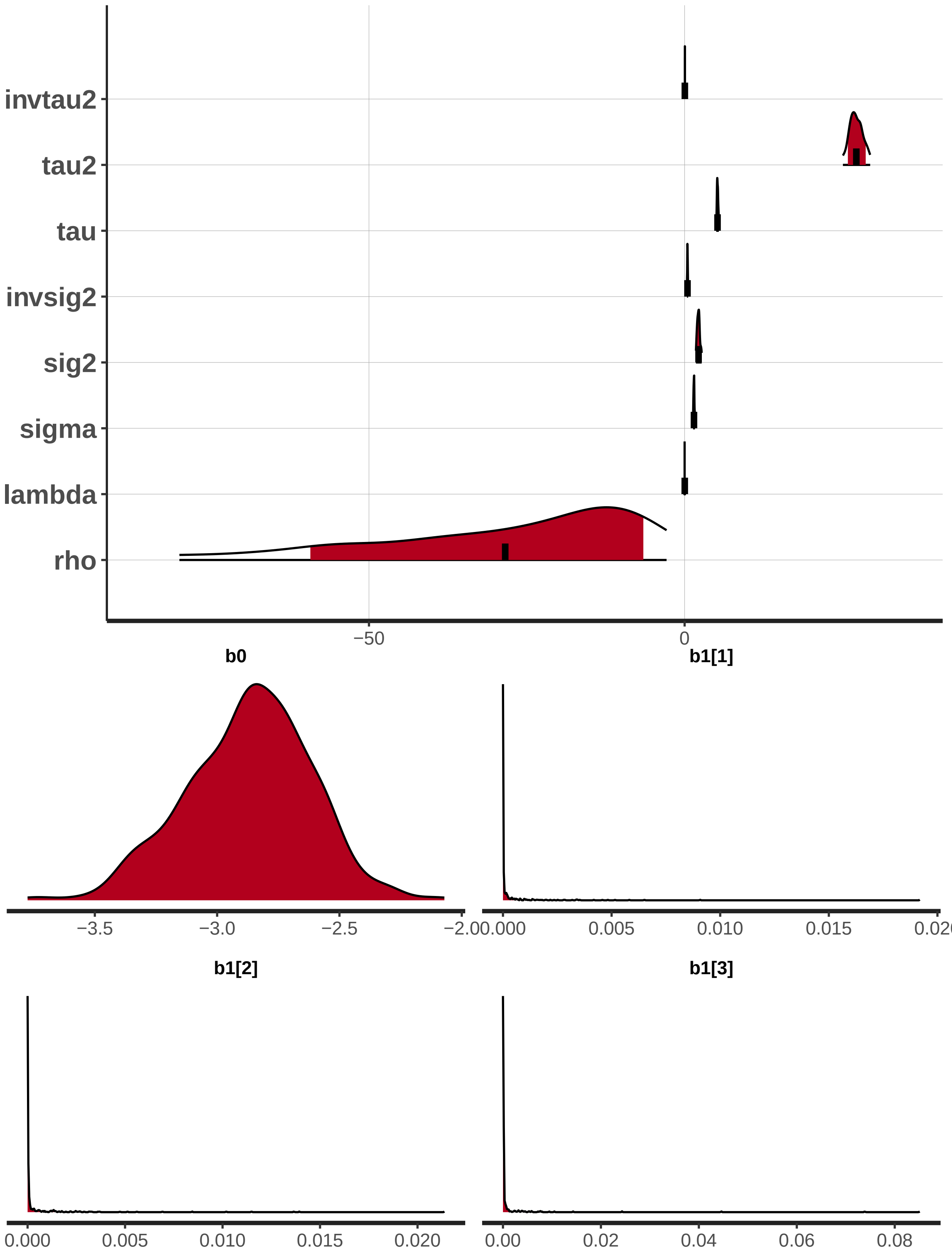
```r
print(fit.lme.add.incr.reslope, pars=param.lme.add)
```

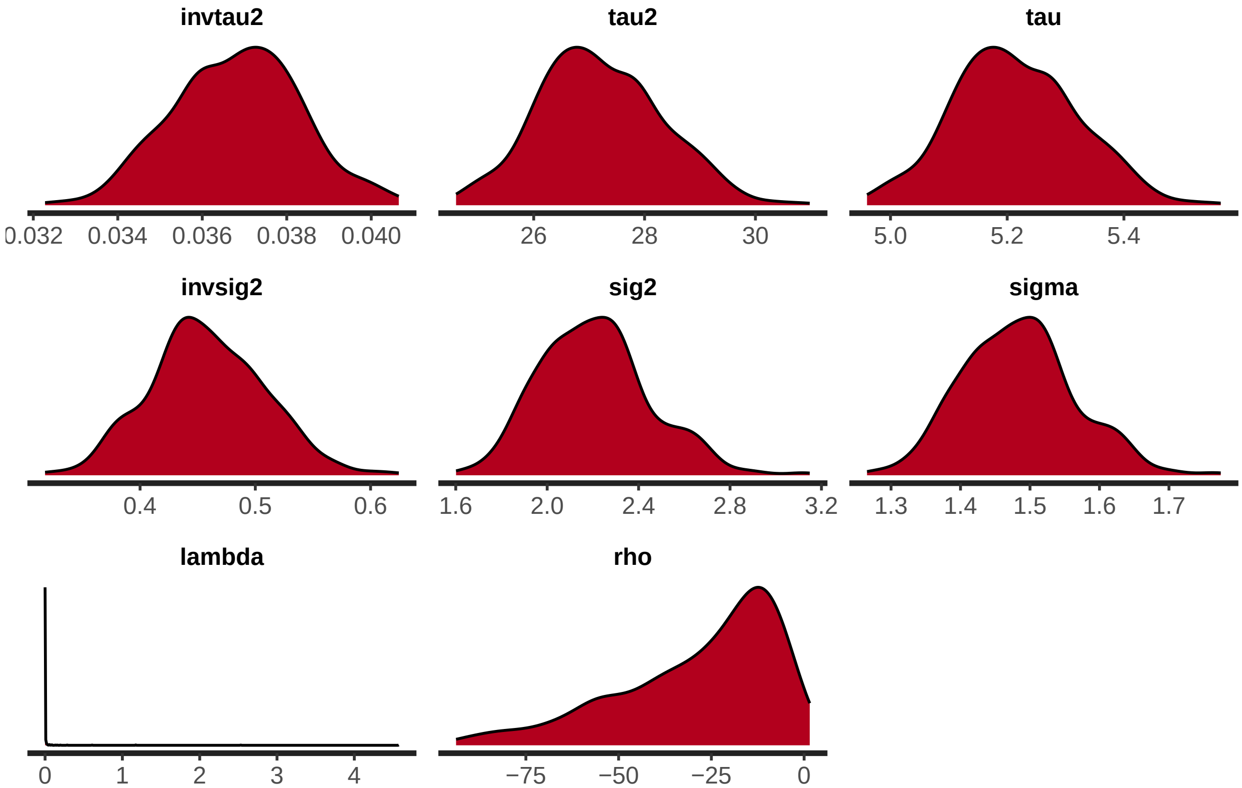
```
## Inference for Stan model: jagam_9_aneur_lme_add_incr_reslope.
## 3 chains, each with iter=600; warmup=300; thin=2;
## post-warmup draws per chain=150, total post-warmup draws=450.
##
##            mean se_mean    sd   2.5%    25%    50%    75% 97.5% n_eff Rhat
## b0        -2.87    0.02  0.25  -3.38  -3.04  -2.86  -2.70 -2.42   222 1.01
## b1[1]      0.00    0.00  0.00   0.00   0.00   0.00   0.00  0.00   339 1.01
## b1[2]      0.00    0.00  0.00   0.00   0.00   0.00   0.00  0.00   436 1.00
## b1[3]      0.00    0.00  0.01   0.00   0.00   0.00   0.00  0.01   448 1.01
## invtau2    0.04    0.00  0.00   0.03   0.04   0.04   0.04  0.04   345 1.01
## tau2      27.19    0.06  1.11  25.08  26.44  27.08  27.91 29.29   341 1.01
## tau        5.21    0.01  0.11   5.01   5.14   5.20   5.28  5.41   342 1.01
## lambda     0.02    0.01  0.25   0.00   0.00   0.00   0.00  0.06   461 1.00
## rho      -28.61    4.05 21.32 -80.03 -41.43 -22.41 -11.85 -2.76    28 1.13
## invsig2    0.46    0.00  0.05   0.37   0.43   0.46   0.49  0.56   376 1.01
## sig2       2.21    0.01  0.25   1.78   2.03   2.19   2.33  2.71   375 1.01
## sigma      1.48    0.00  0.08   1.33   1.42   1.48   1.53  1.65   375 1.01
##
## Samples were drawn using NUTS(diag_e) at Wed Aug 16 11:48:16 2023.
## 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).
```

```r
stan_trace(fit.lme.add.incr.reslope, pars=param.lme.add)
stan_plot(fit.lme.add.incr.reslope, pars=c("b0","b1"), point_est = "mean", show_density = TRUE)
stan_plot(fit.lme.add.incr.reslope, pars=c("invtau2","tau2","tau", "invsig2","sig2","sigma",  "lambda",
stan_dens(fit.lme.add.incr.reslope, pars=c("b0","b1"))
stan_dens(fit.lme.add.incr.reslope, pars=c("invtau2","tau2","tau", "invsig2","sig2","sigma", "lambda","
```

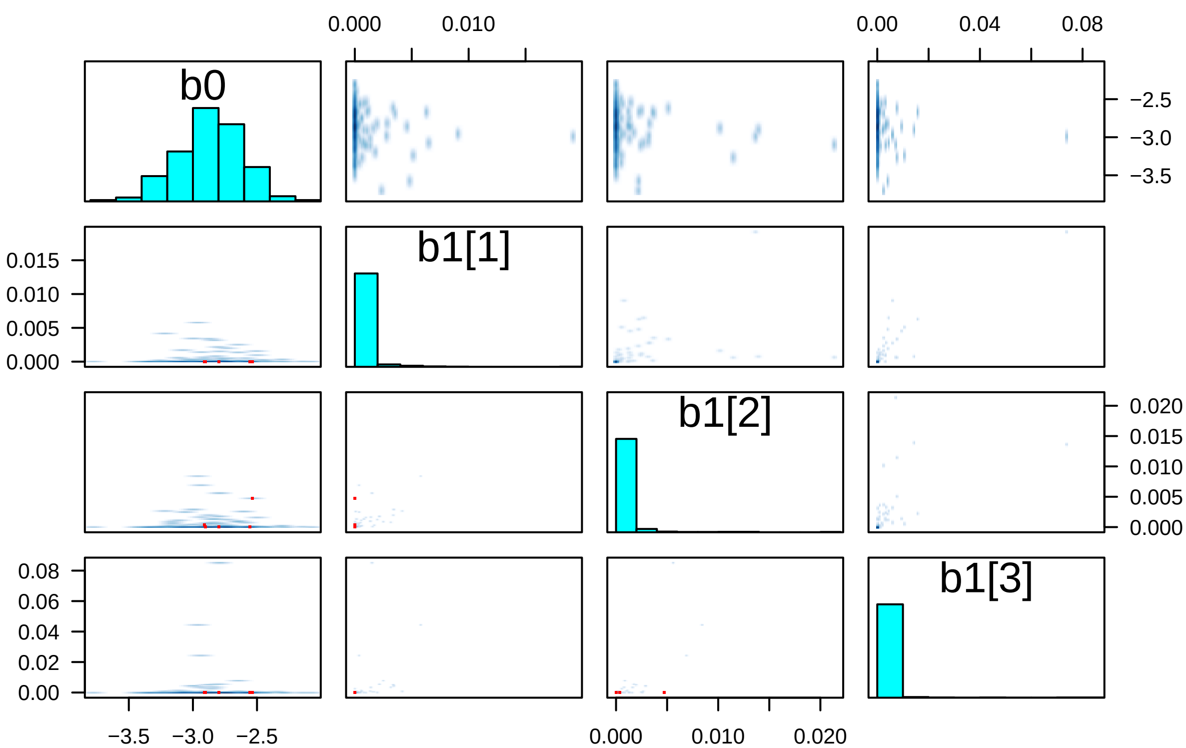

b0
b1[1]
b1[2]
b1[3]
invtau2
tau2
tau
lambda
rho
invsig2
sig2
sigma
chain
1
2
3


b0
b1[1]
b1[2]
b1[3]
−3
−2
−1
0

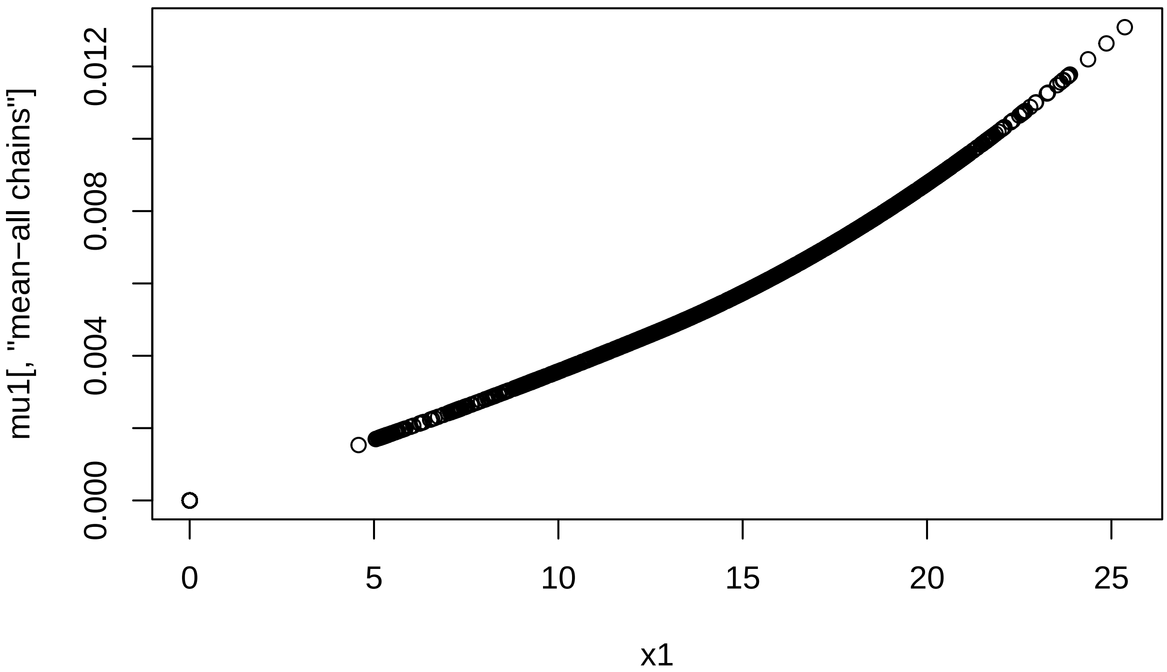

invtau2
tau2
tau
invsig2
sig2
sigma
lambda
rho
−50
0
b0
b1[1]
−3.5
−3.0
−2.5
−2.0
0.000
0.005
0.010
0.015
b1[2]
b1[3]
0.000
0.005
0.010
0.015
0.020
0.00
0.02
0.04
0.06
0.08



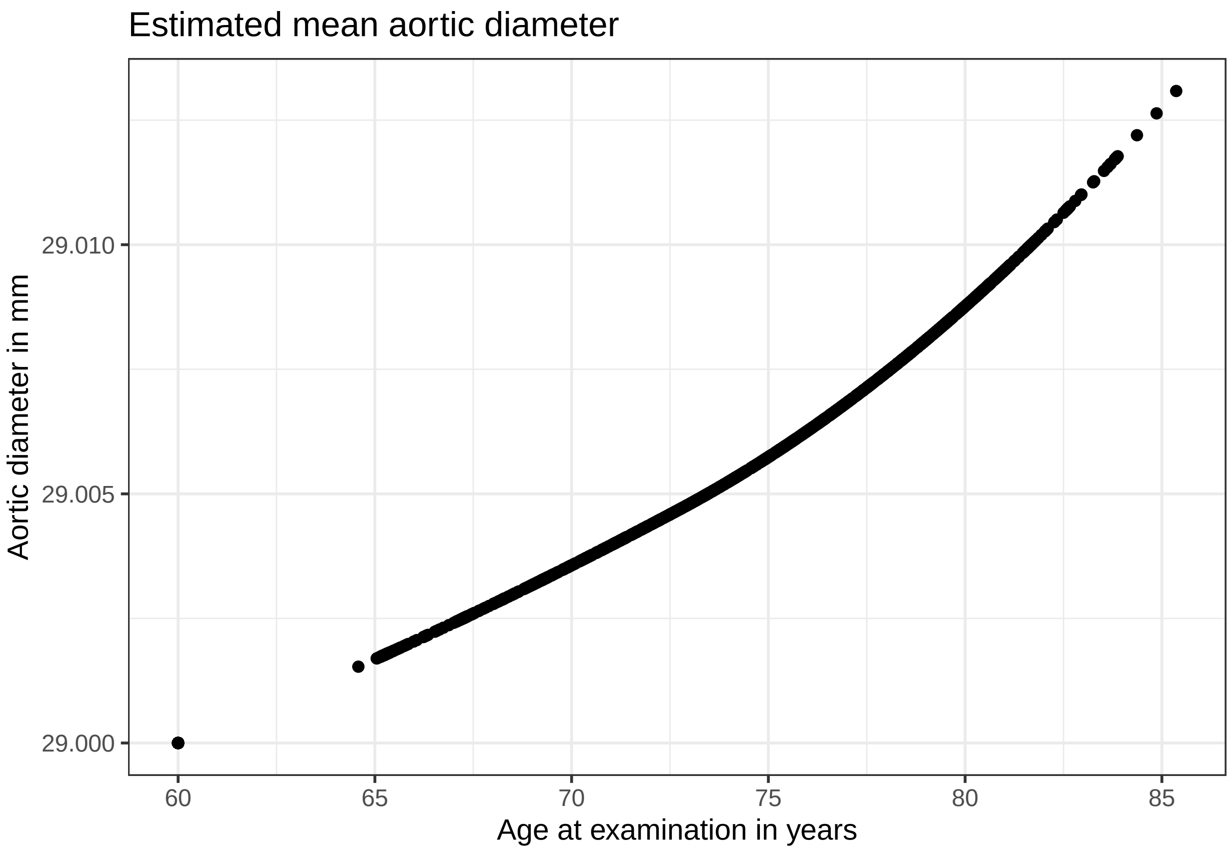

```
pairs(fit.lme.add.incr.reslope, pars = c("b0","b1"), las = 1)
```



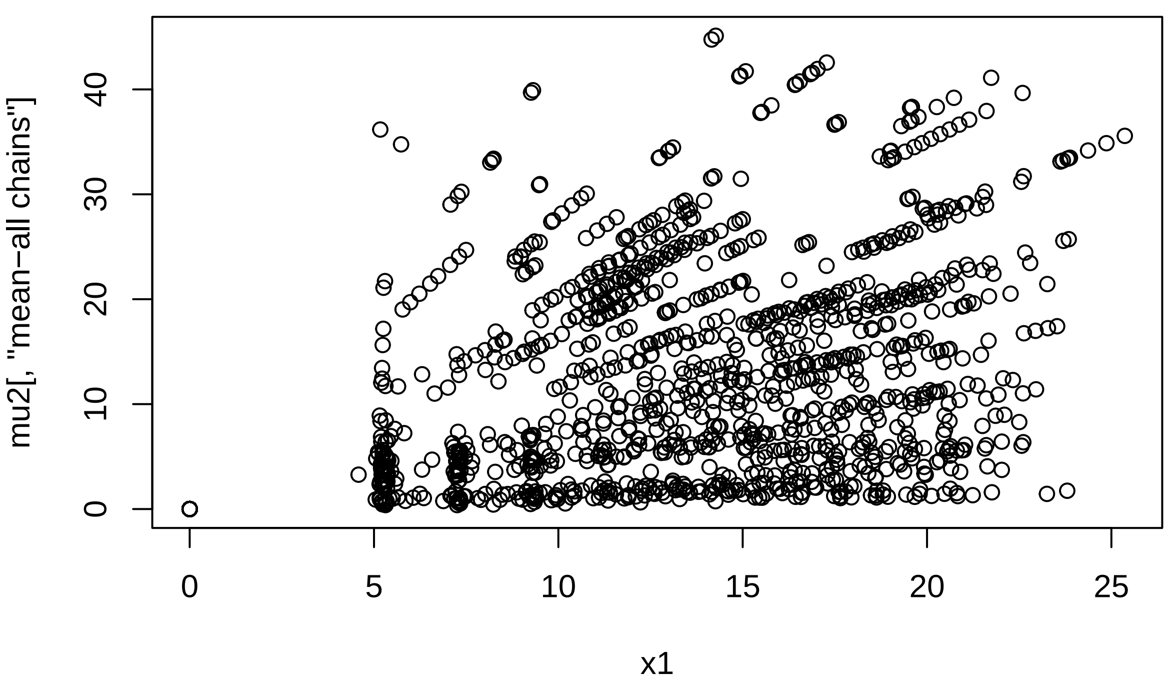

```
mu1 = get_posterior_mean(fit.lme.add.incr.reslope,"mu1")
plot(x1,mu1[,"mean-all chains"])
```



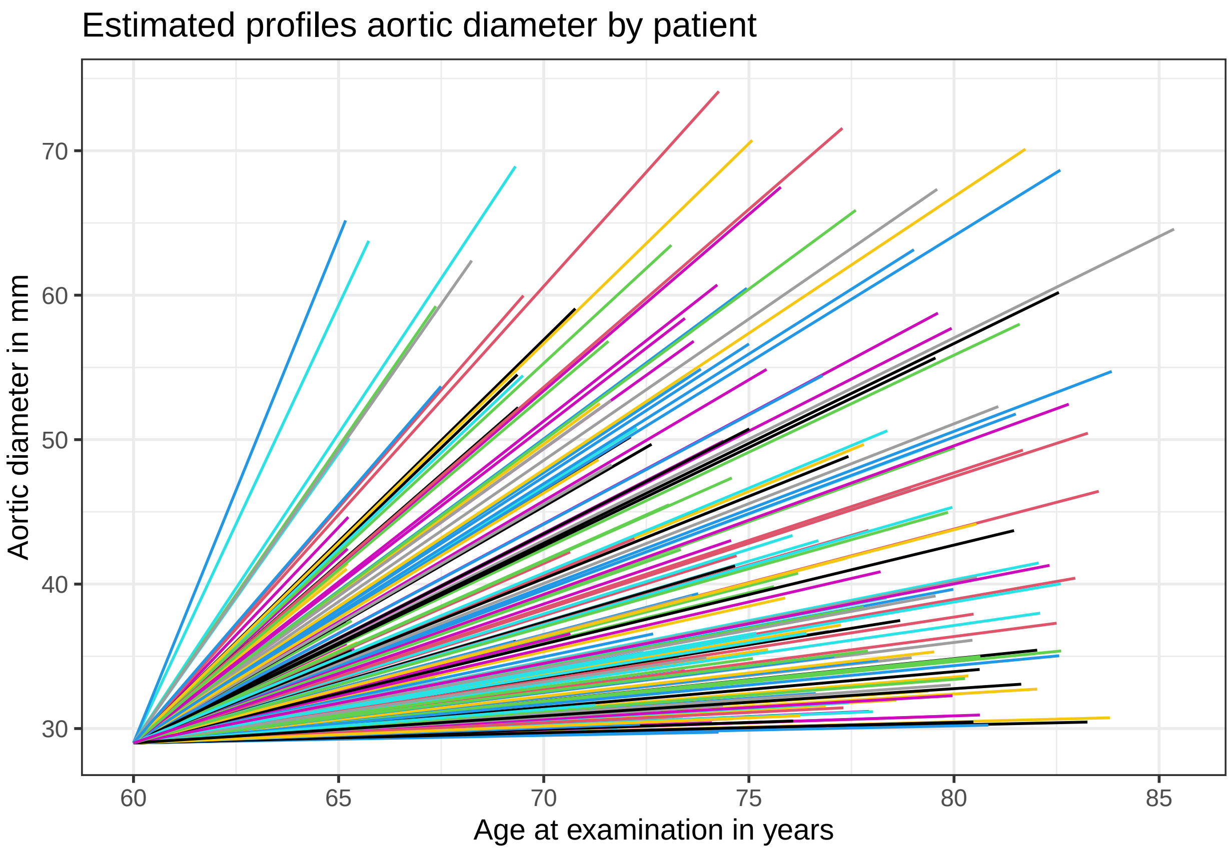

```
mu1 = get_posterior_mean(fit.lme.add.incr.reslope,"mu1")
aneur3$mu1_all = 29 + mu1[,"mean-all chains"]


### paper size A9: 3.7 x 5.2 cm

ggplot(data=aneur3,
       mapping=aes(x=age,y=mu1_all,group=ptnum)) +
  geom_point() +
  theme_bw() +
  xlab("Age at examination in years") + ylab("Aortic diameter in mm") +
  ggtitle("Estimated mean aortic diameter")
```

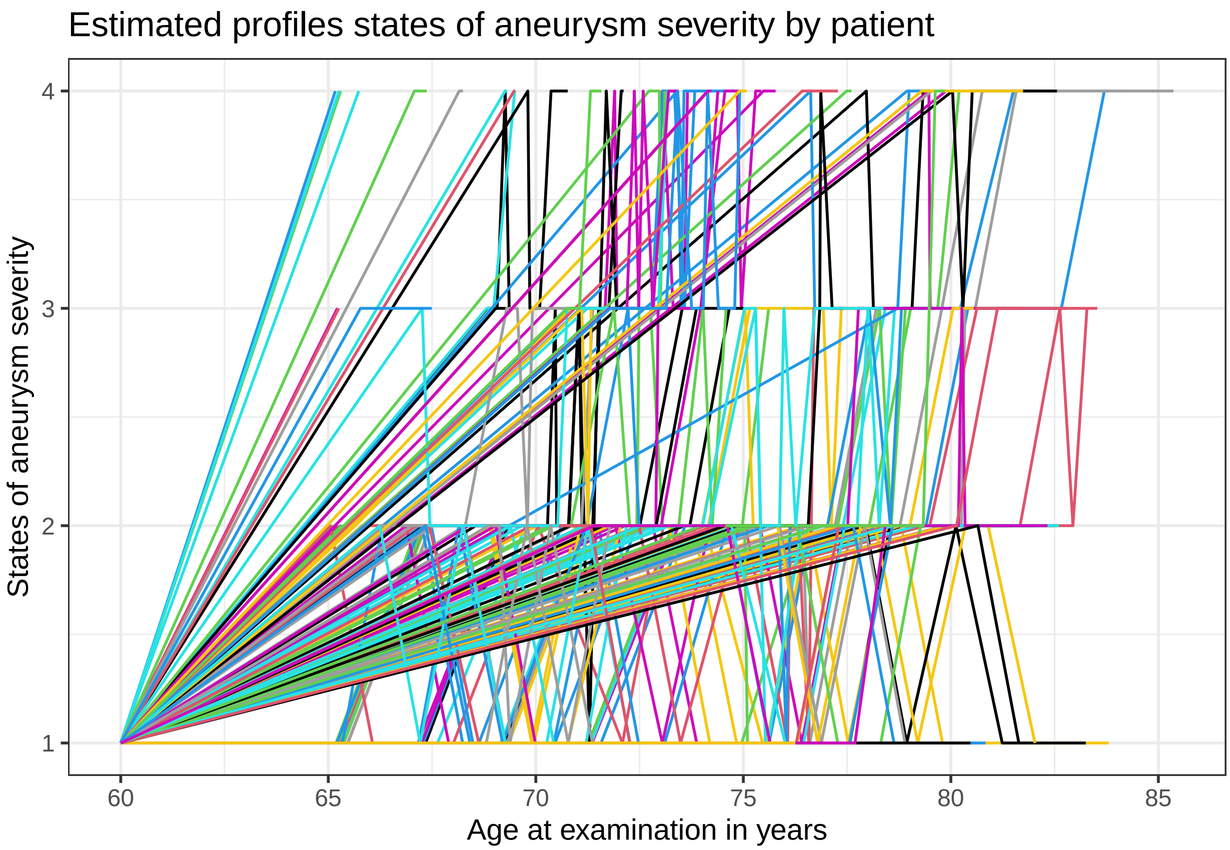



```r
mu2 = get_posterior_mean(fit.lme.add.incr.reslope,"mu2")
plot(x1,mu2[,"mean-all chains"])
```



```r
mu2 = get_posterior_mean(fit.lme.add.incr.reslope,"mu2")
aneur3$mu2_all = 29 + mu2[,"mean-all chains"]


### paper size A9: 3.7 x 5.2 cm

ggplot(data=aneur3,
       mapping=aes(x=age,y=mu2_all,group=ptnum)) +
  geom_line(color=aneur3$ptnum) +
  theme_bw() +
  xlab("Age at examination in years") + ylab("Aortic diameter in mm") +
  ggtitle("Estimated profiles aortic diameter by patient")

ggplot(data=aneur3,
       mapping=aes(x=age,y=state,group=ptnum)) +
  geom_line(color=aneur3$ptnum) +
    theme_bw() +
  xlab("Age at examination in years") + ylab("States of aneurysm severity") + ggtitle("Estimated profile
```




Estimated profiles states of aneurysm severity by patient
States of aneurysm severity
4
3
2
1
60
65
70
75
80
85
Age at examination in years

CASES 9.2. LME: Spline con restricciones creciente

```
idx = names(table(aneur3$ptnum))

id690 = which(aneur3$ptnum==690)
n690 = length(id690)
i690 = which(idx==690)
XI690 = XI1[id690,]
x690 = x1[id690]

id703 = which(aneur3$ptnum==703)
n703 = length(id703)
i703 = which(idx==703)
XI703 = XI1[id703,]
x703 = x1[id703]

id705 = which(aneur3$ptnum==705)
n705 = length(id705)
i705 = which(idx==705)
XI705 = XI1[id705,]
x705 = x1[id705]

id745 = which(aneur3$ptnum==745)
n745 = length(id745)
i745 = which(idx==745)
XI745 = XI1[id745,]
x745 = x1[id745]

id746 = which(aneur3$ptnum==746)
n746 = length(id746)
i746 = which(idx==746)
XI746 = XI1[id746,]
x746 = x1[id746]

id837 = which(aneur3$ptnum==837)
n837 = length(id837)
i837 = which(idx==837)
XI837 = XI1[id837,]
x837 = x1[id837]

datos.lme.add.incr.cases <- list( y = y ,
                          id = id ,
               n = length(y) ,
               N = N , Ni = Ni,
               k1=k1,
              XI1 = XI1,
              x1 = x1,
              zero = rep(0,1+k1),
              S1=S1 ,
              n690=n690, i690=i690, XI690=XI690, x690=x690,
              n703=n703, i703=i703, XI703=XI703, x703=x703,
              n705=n705, i705=i705, XI705=XI705, x705=x705,
              n745=n745, i745=i745, XI745=XI745, x745=x745,
```

```r
              n746=n746, i746=i746, XI746=XI746, x746=x746,
              n837=n837, i837=i837, XI837=XI837, x837=x837 )
```

```r
fit.lme.add.incr.reslope.cases <- stan("jagam_9_aneur_lme_add_incr_reslope_cases.stan",
            data=datos.lme.add.incr.cases,
            chains=3,warmup=300,iter=600,thin=2,cores=4,
            init= inits.lme.add.incr)
```

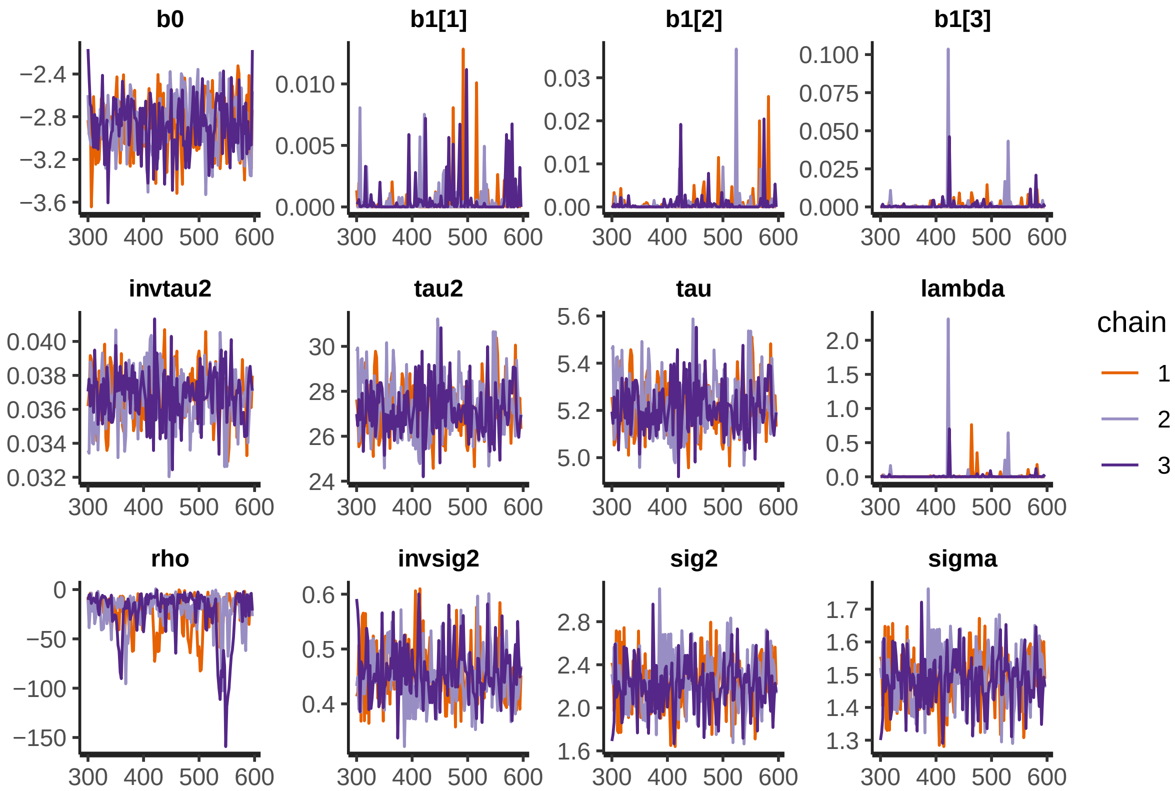
```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -mmacosx-version-min=10.13 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
##                ^
##                ;
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/Core:96:10: fa
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```

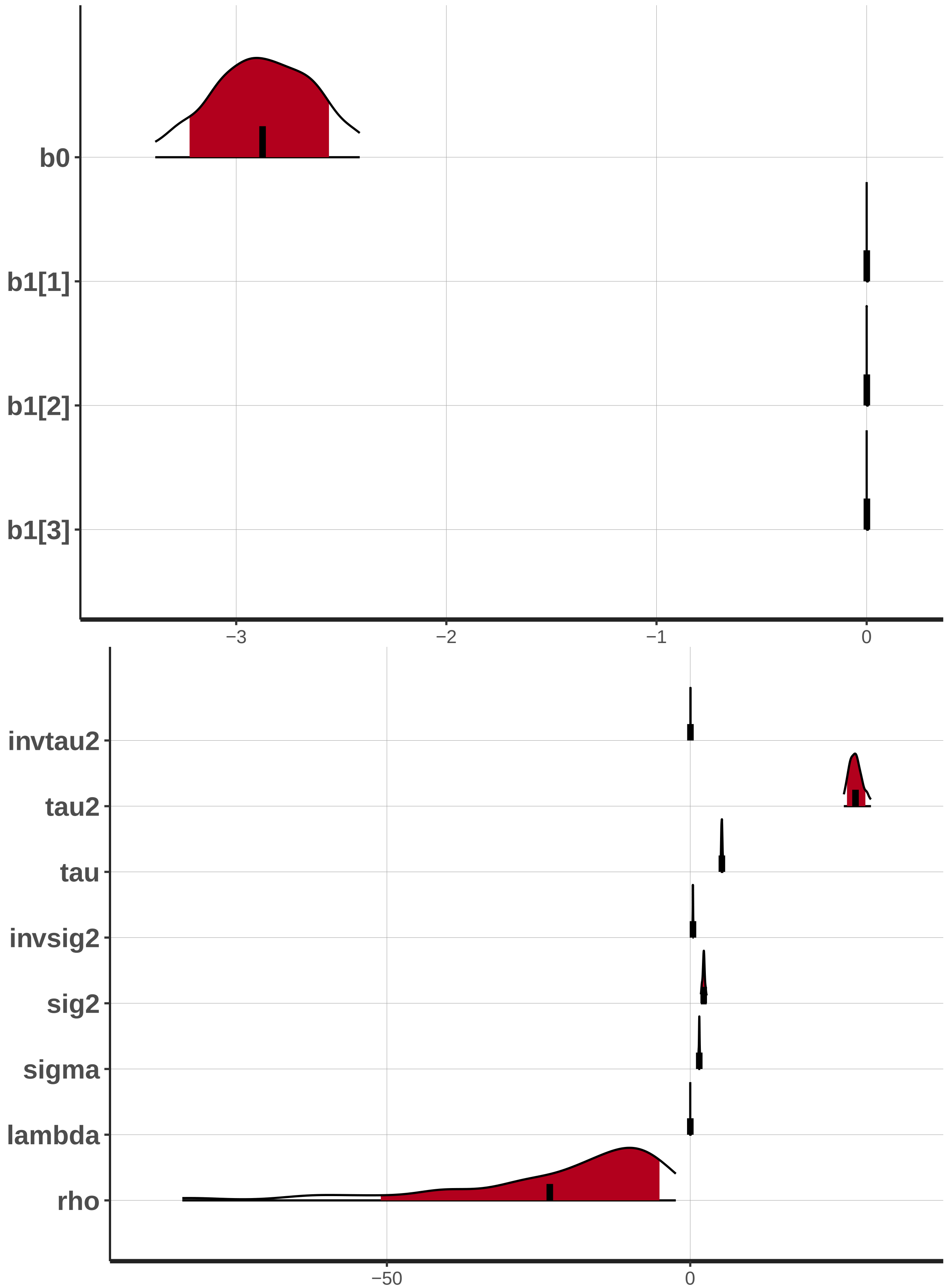
```r
print(fit.lme.add.incr.reslope.cases, pars=param.lme.add)
```

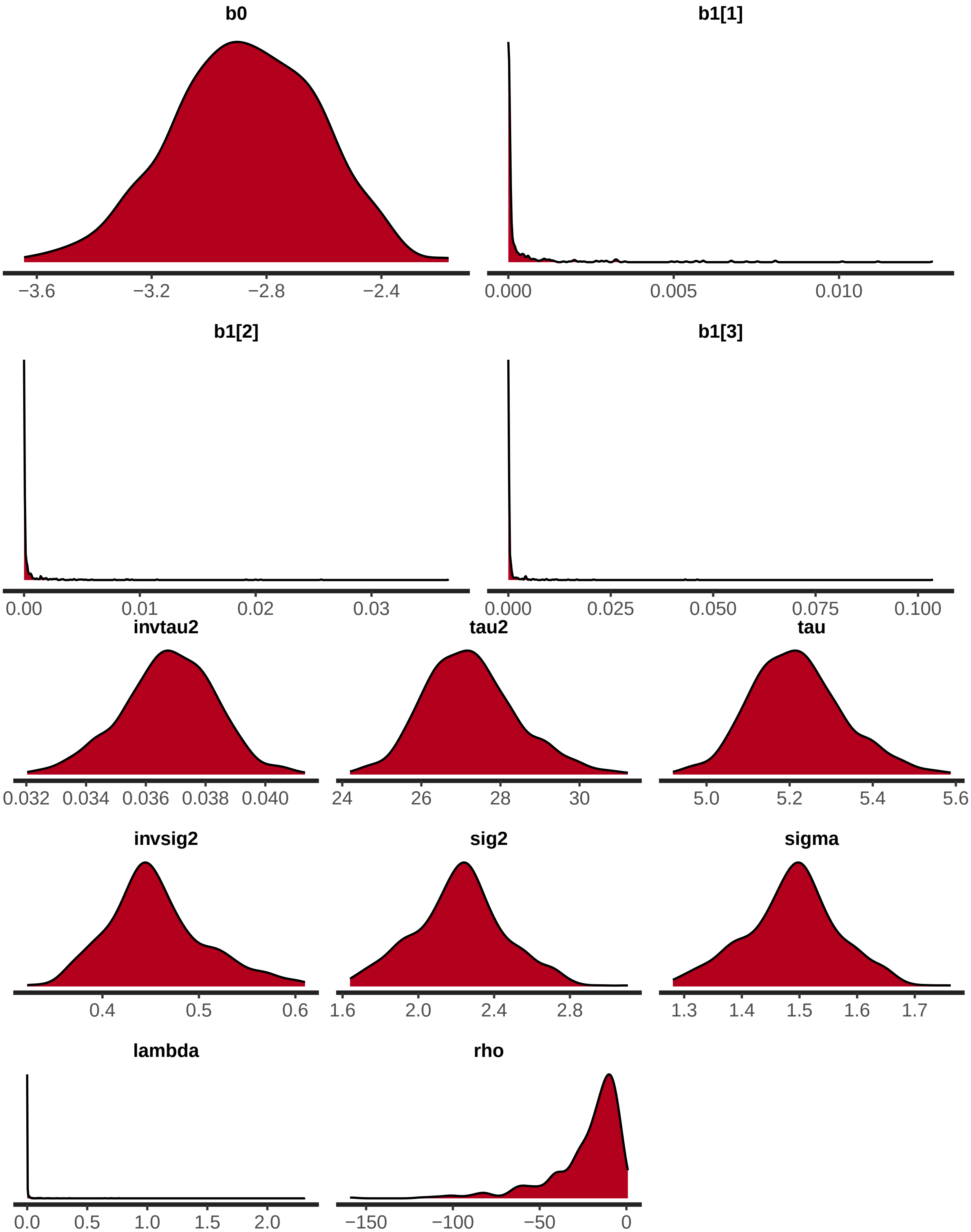
```
## Inference for Stan model: jagam_9_aneur_lme_add_incr_reslope_cases.
## 3 chains, each with iter=600; warmup=300; thin=2;
## post-warmup draws per chain=150, total post-warmup draws=450.
##
##           mean se_mean    sd   2.5%    25%    50%   75% 97.5% n_eff Rhat
## b0       -2.88    0.01  0.25  -3.38  -3.05  -2.87 -2.69 -2.41   288 1.01
## b1[1]     0.00    0.00  0.00   0.00   0.00   0.00  0.00  0.01   461 1.01
## b1[2]     0.00    0.00  0.00   0.00   0.00   0.00  0.00  0.01   474 1.00
## b1[3]     0.00    0.00  0.01   0.00   0.00   0.00  0.00  0.01   469 1.00
## invtau2   0.04    0.00  0.00   0.03   0.04   0.04  0.04  0.04   353 1.00
## tau2     27.24    0.06  1.17  25.32  26.44  27.16 27.93 29.79   352 1.00
## tau       5.22    0.01  0.11   5.03   5.14   5.21  5.28  5.46   352 1.00
## lambda    0.01    0.01  0.13   0.00   0.00   0.00  0.00  0.09   458 1.00
## rho     -23.21    2.62 21.59 -83.63 -29.16 -16.27 -8.93 -2.38    68 1.01
## invsig2   0.46    0.00  0.05   0.37   0.42   0.45  0.48  0.57   336 1.00
## sig2      2.22    0.01  0.24   1.75   2.07   2.23  2.36  2.71   292 1.00
## sigma     1.49    0.00  0.08   1.32   1.44   1.49  1.54  1.65   298 1.00
##
## Samples were drawn using NUTS(diag_e) at Wed Aug 16 11:50:45 2023.
## 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).
```

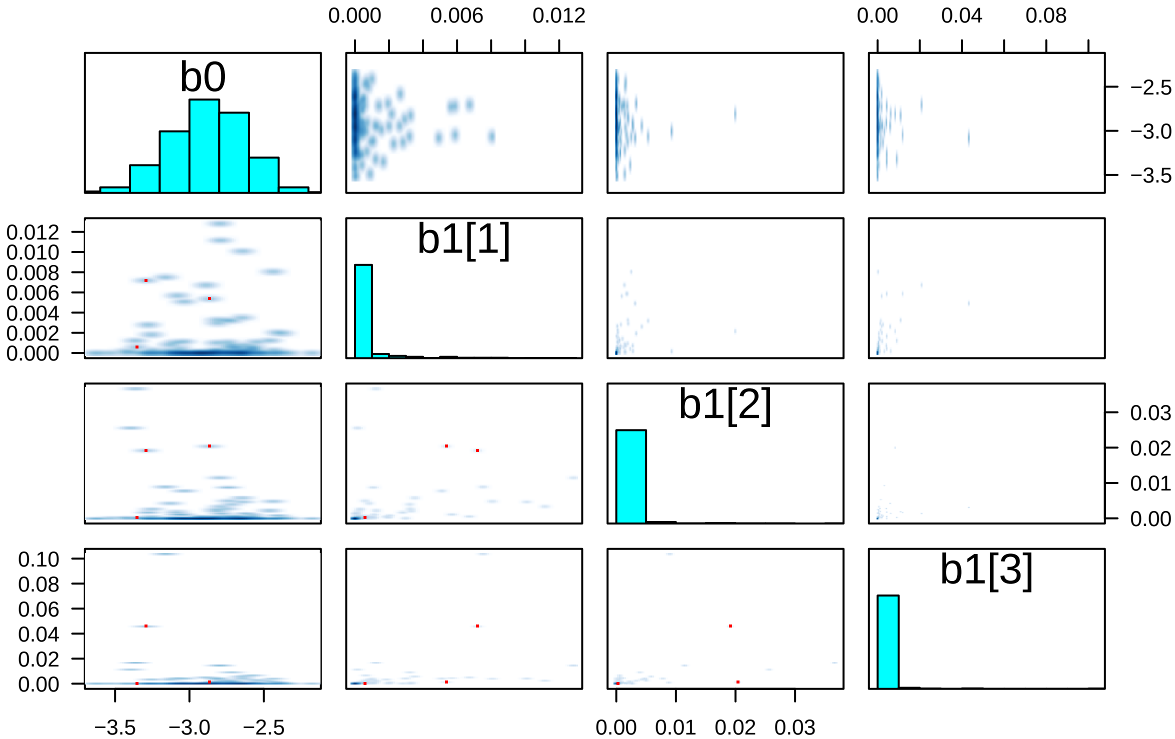
```
stan_trace(fit.lme.add.incr.reslope.cases, pars=param.lme.add)
stan_plot(fit.lme.add.incr.reslope.cases, pars=c("b0","b1"), point_est = "mean", show_density = TRUE)
stan_plot(fit.lme.add.incr.reslope.cases, pars=c("invtau2","tau2","tau", "invsig2","sig2","sigma",  "la
stan_dens(fit.lme.add.incr.reslope.cases, pars=c("b0","b1"))
stan_dens(fit.lme.add.incr.reslope.cases, pars=c("invtau2","tau2","tau", "invsig2","sig2","sigma", "lam
```



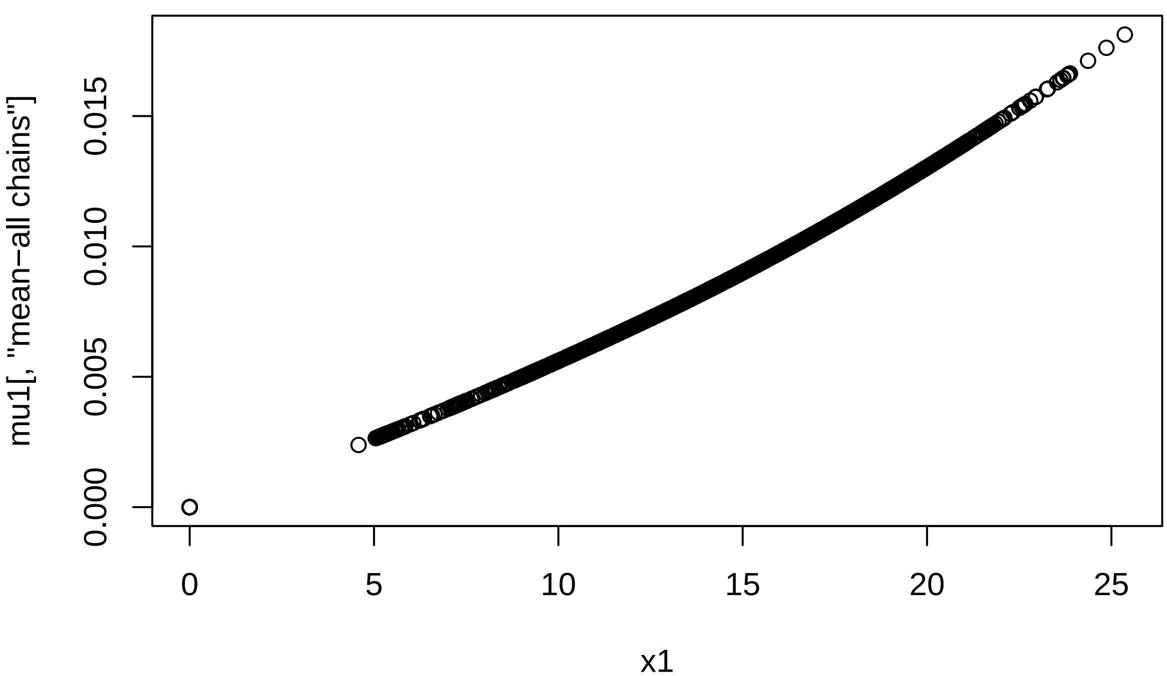

b0
b1[1]
b1[2]
b1[3]
−3
−2
−1
0
invtau2
tau2
tau
invsig2
sig2
sigma
lambda
rho
−50
0



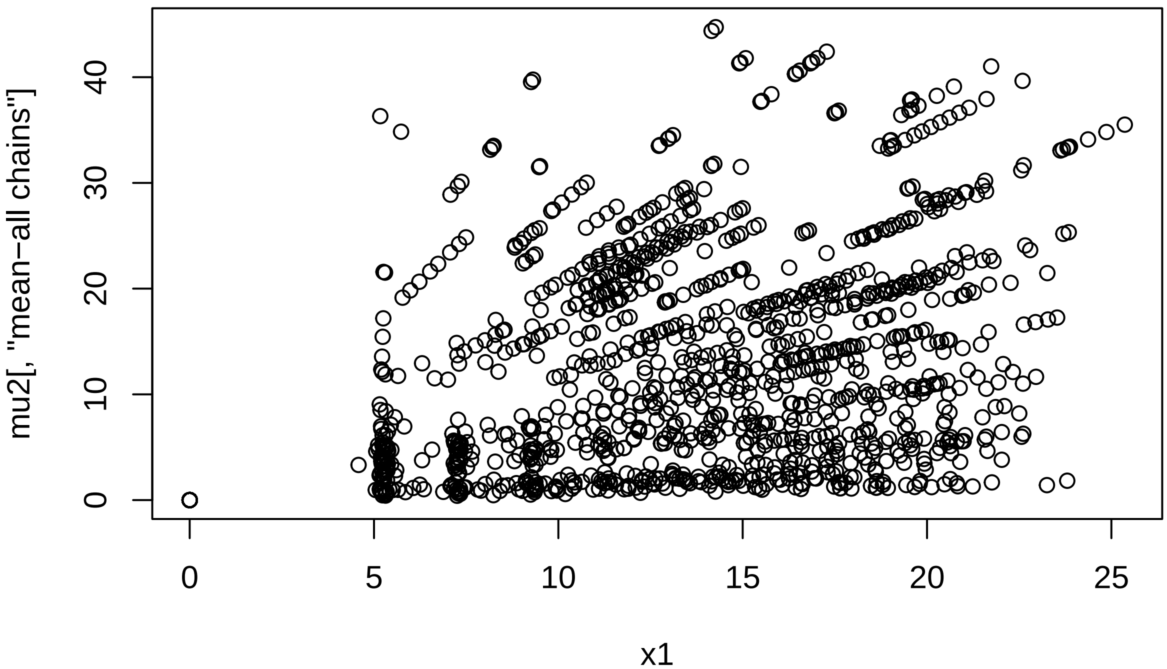

```
pairs(fit.lme.add.incr.reslope.cases, pars = c("b0","b1"), las = 1)
```



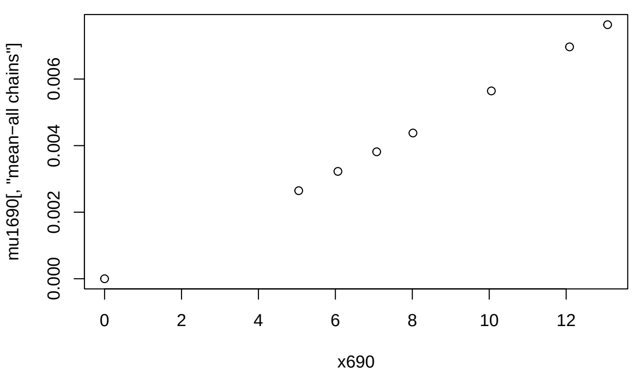

```
mu1 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1")
plot(x1,mu1[,"mean-all chains"])
```

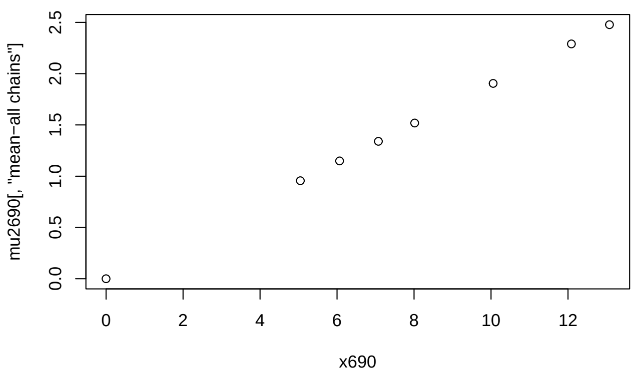



```
mu2 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2")
plot(x1,mu2[,"mean-all chains"])
```

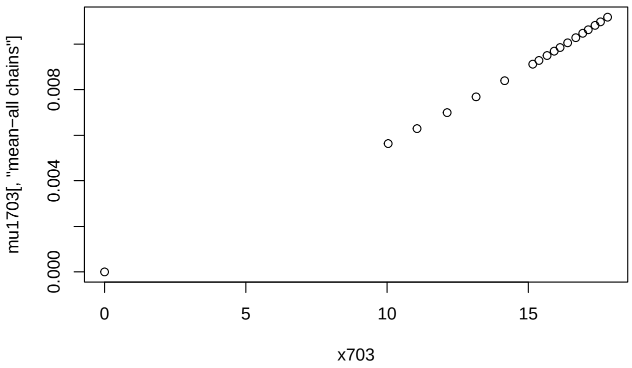


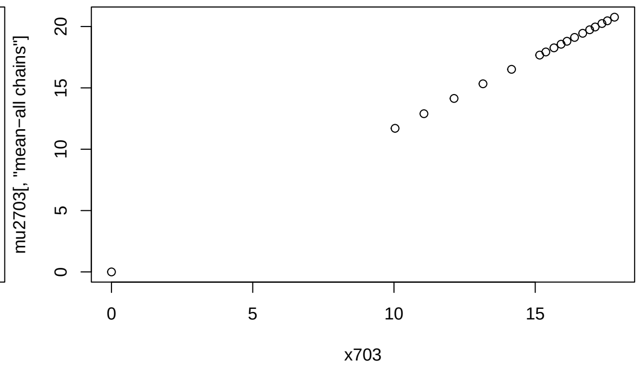

```
mu1690 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1690")
mu2690 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2690")

plot(x690,mu1690[,"mean-all chains"])
plot(x690,mu2690[,"mean-all chains"])
```



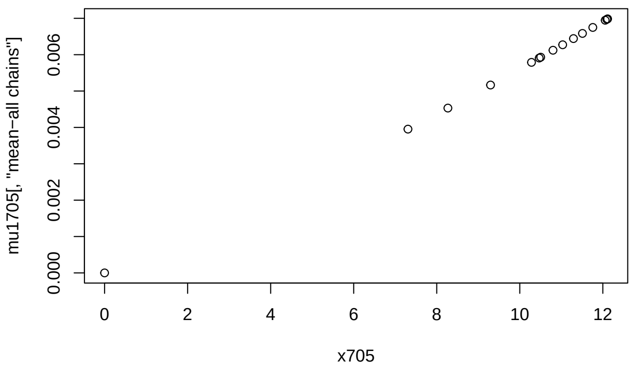


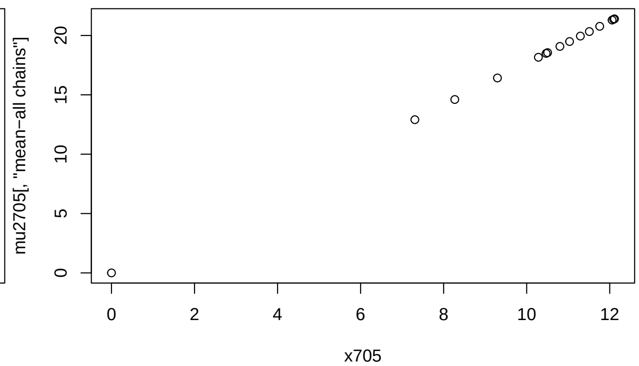

```
mu1703 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1703")
mu2703 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2703")

plot(x703,mu1703[,"mean-all chains"])
plot(x703,mu2703[,"mean-all chains"])
```



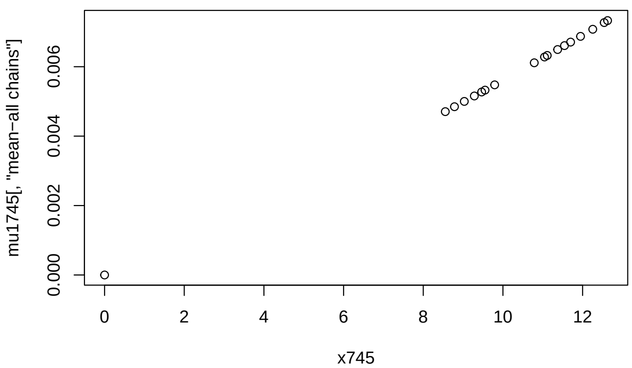


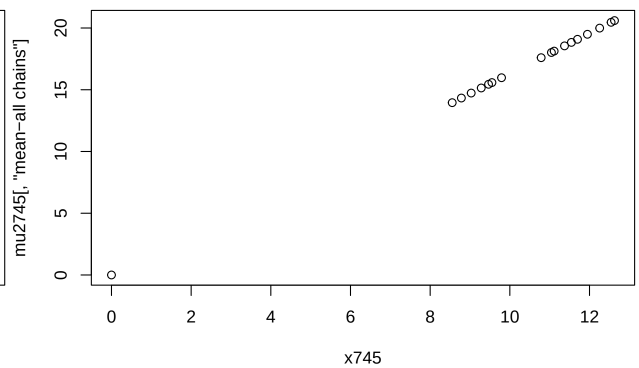

```r
mu1705 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1705")
mu2705 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2705")

plot(x705,mu1705[,"mean-all chains"])
plot(x705,mu2705[,"mean-all chains"])
```



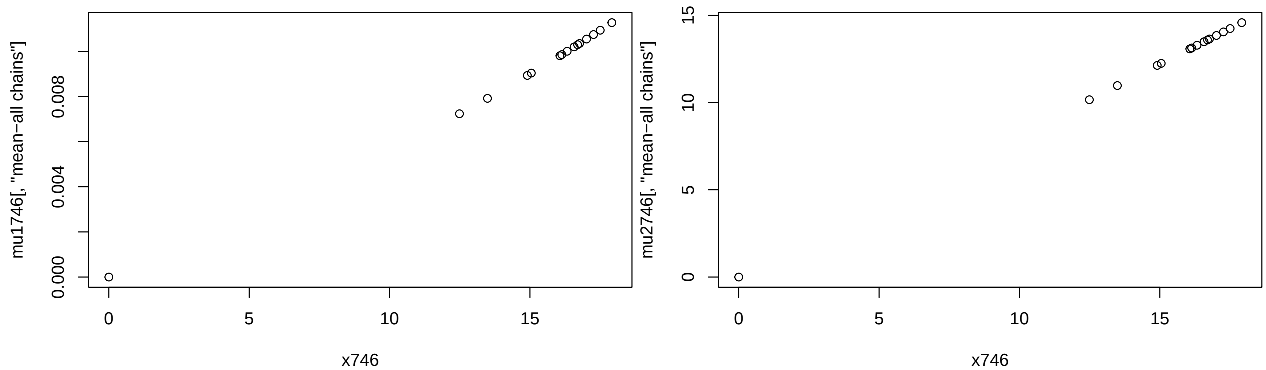



```r
mu1745 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1745")
mu2745 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2745")

plot(x745,mu1745[,"mean-all chains"])
plot(x745,mu2745[,"mean-all chains"])
```

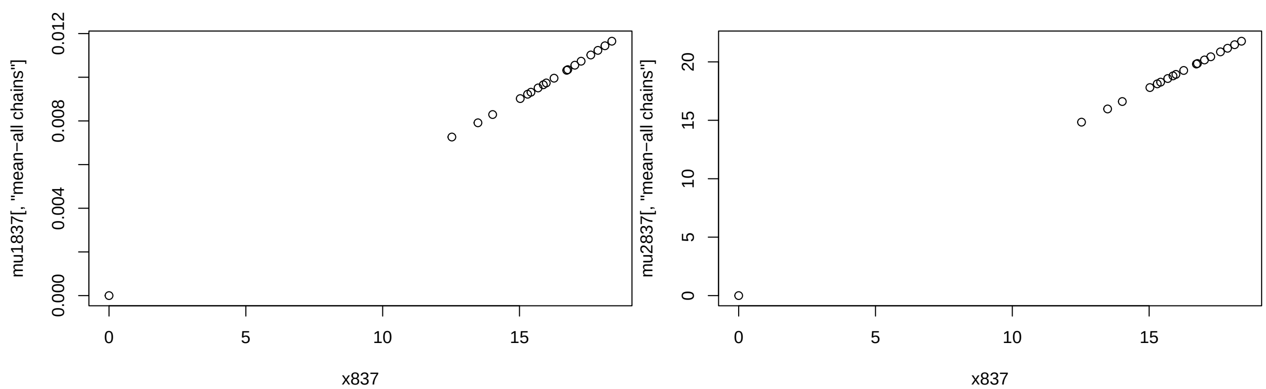





```r
mu1746 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1746")
mu2746 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2746")

plot(x746,mu1746[,"mean-all chains"])
plot(x746,mu2746[,"mean-all chains"])
```




```r
mu1837 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1837")
mu2837 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2837")

plot(x837,mu1837[,"mean-all chains"])
plot(x837,mu2837[,"mean-all chains"])
```



CASES 10.2. LME: Spline con restricciones creciente

```
idx = names(table(aneur3$ptnum))

id690 = which(aneur3$ptnum==690)
n690 = length(id690)
i690 = which(idx==690)
XI690 = XI1[id690,]
x690 = x1[id690]

id703 = which(aneur3$ptnum==703)
n703 = length(id703)
i703 = which(idx==703)
XI703 = XI1[id703,]
x703 = x1[id703]

id705 = which(aneur3$ptnum==705)
n705 = length(id705)
i705 = which(idx==705)
XI705 = XI1[id705,]
x705 = x1[id705]

id745 = which(aneur3$ptnum==745)
n745 = length(id745)
i745 = which(idx==745)
XI745 = XI1[id745,]
x745 = x1[id745]

id746 = which(aneur3$ptnum==746)
n746 = length(id746)
i746 = which(idx==746)
XI746 = XI1[id746,]
x746 = x1[id746]

id837 = which(aneur3$ptnum==837)
n837 = length(id837)
i837 = which(idx==837)
XI837 = XI1[id837,]
x837 = x1[id837]

datos.lme.add.incr.cases <- list( y = y ,
                          id = id ,
               n = length(y) ,
               N = N , Ni = Ni,
               k1=k1,
              XI1 = XI1,
              x1 = x1,
              zero = rep(0,1+k1),
              S1=S1 ,
              n690=n690, i690=i690, XI690=XI690, x690=x690,
              n703=n703, i703=i703, XI703=XI703, x703=x703,
              n705=n705, i705=i705, XI705=XI705, x705=x705,
              n745=n745, i745=i745, XI745=XI745, x745=x745,
```

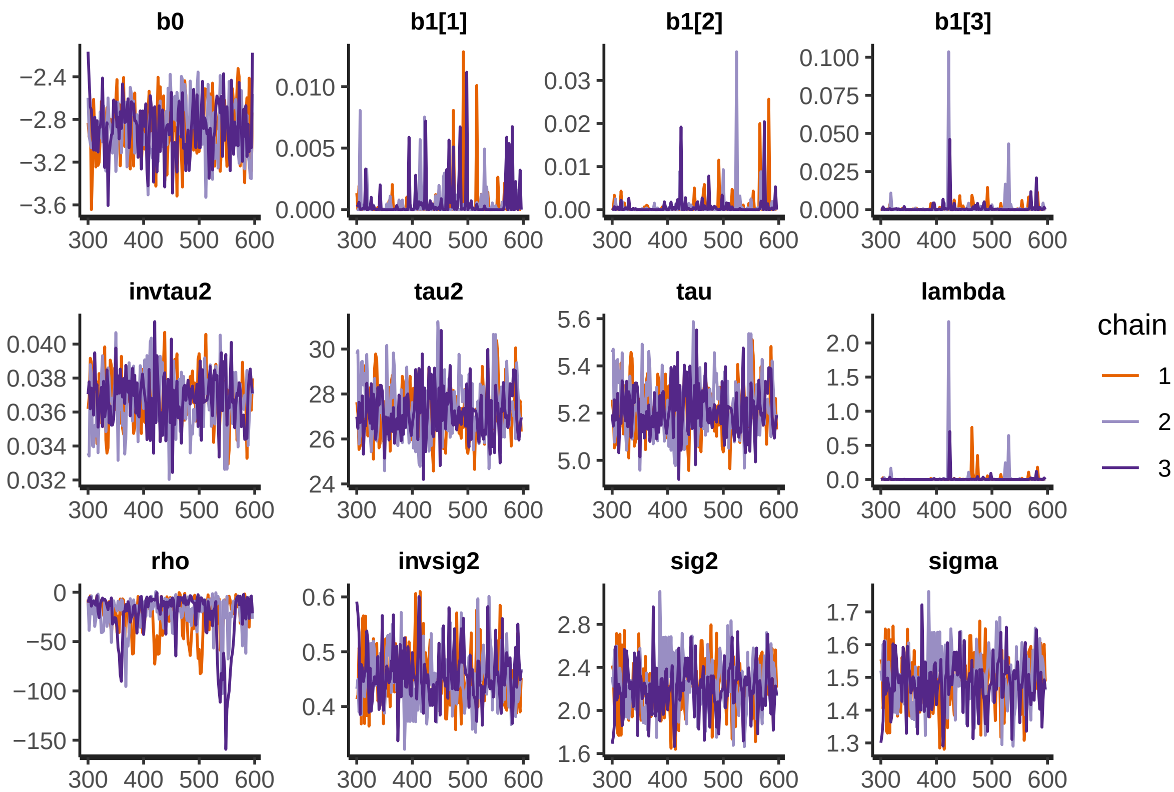
```r
              n746=n746, i746=i746, XI746=XI746, x746=x746,
              n837=n837, i837=i837, XI837=XI837, x837=x837 )
```

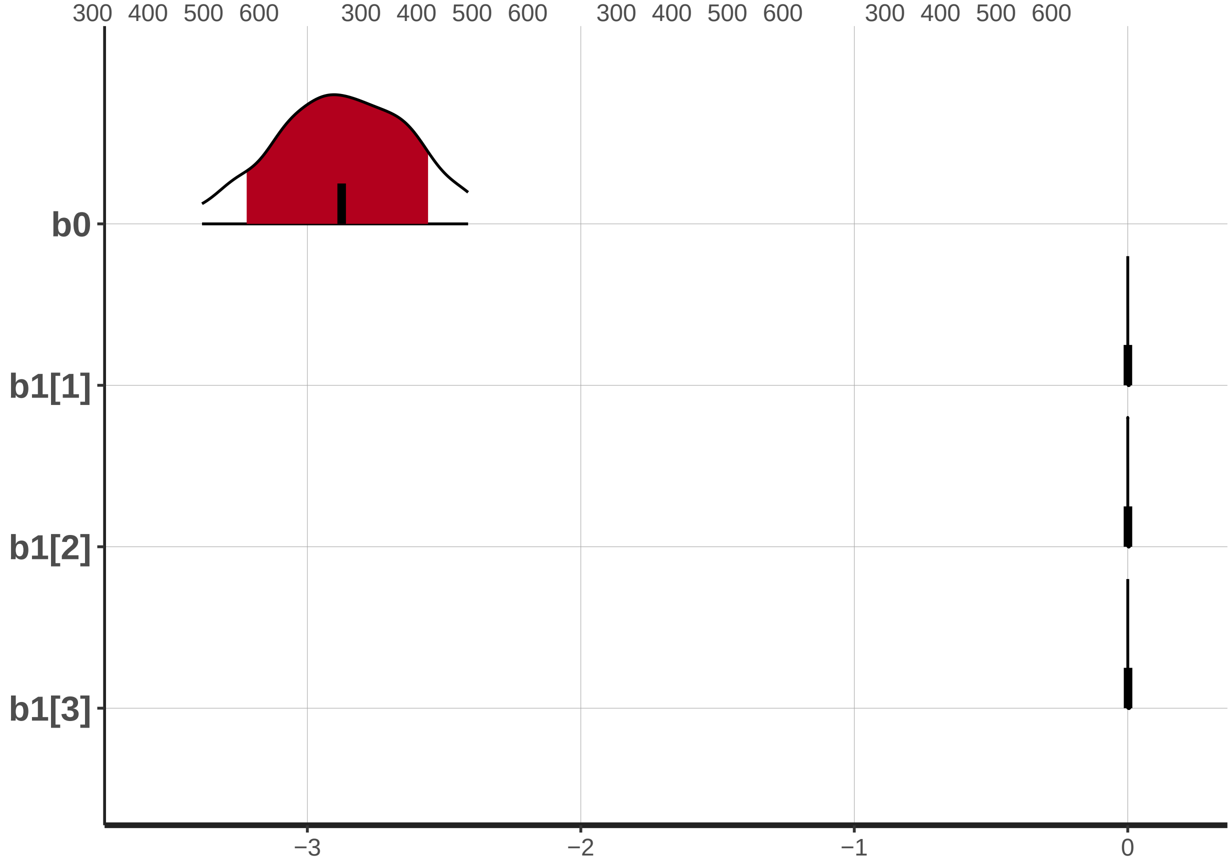
```r
## fit.lme.add.incr.reslope.cases <- stan("jagam_9_aneur_lme_add_incr_reslope_cases_v2.stan",
##              data=datos.lme.add.incr.cases,
##              chains=3,warmup=300,iter=600,thin=2,cores=4,
##              init= inits.lme.add.incr)
```

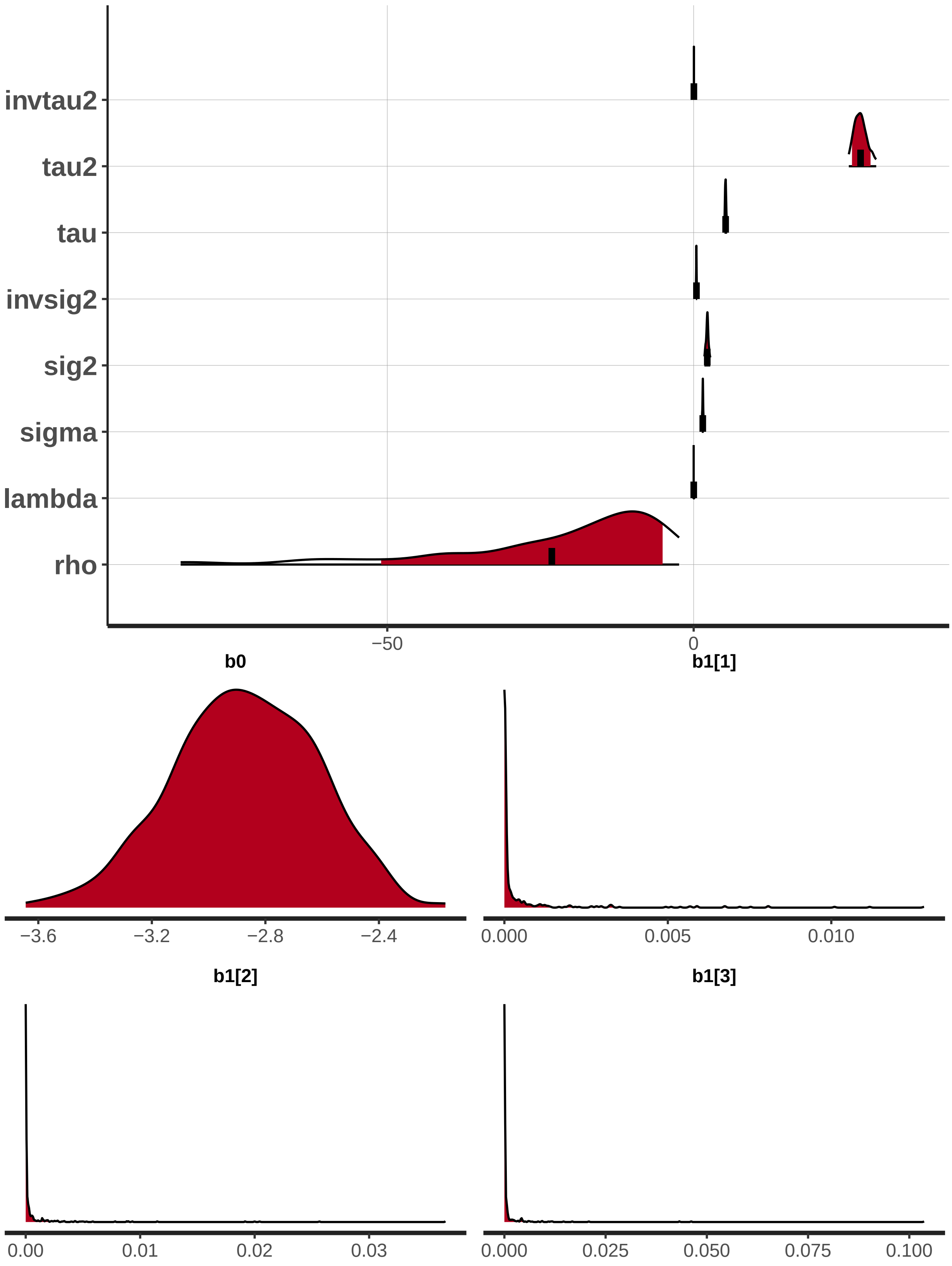
```r
print(fit.lme.add.incr.reslope.cases, pars=param.lme.add)
```

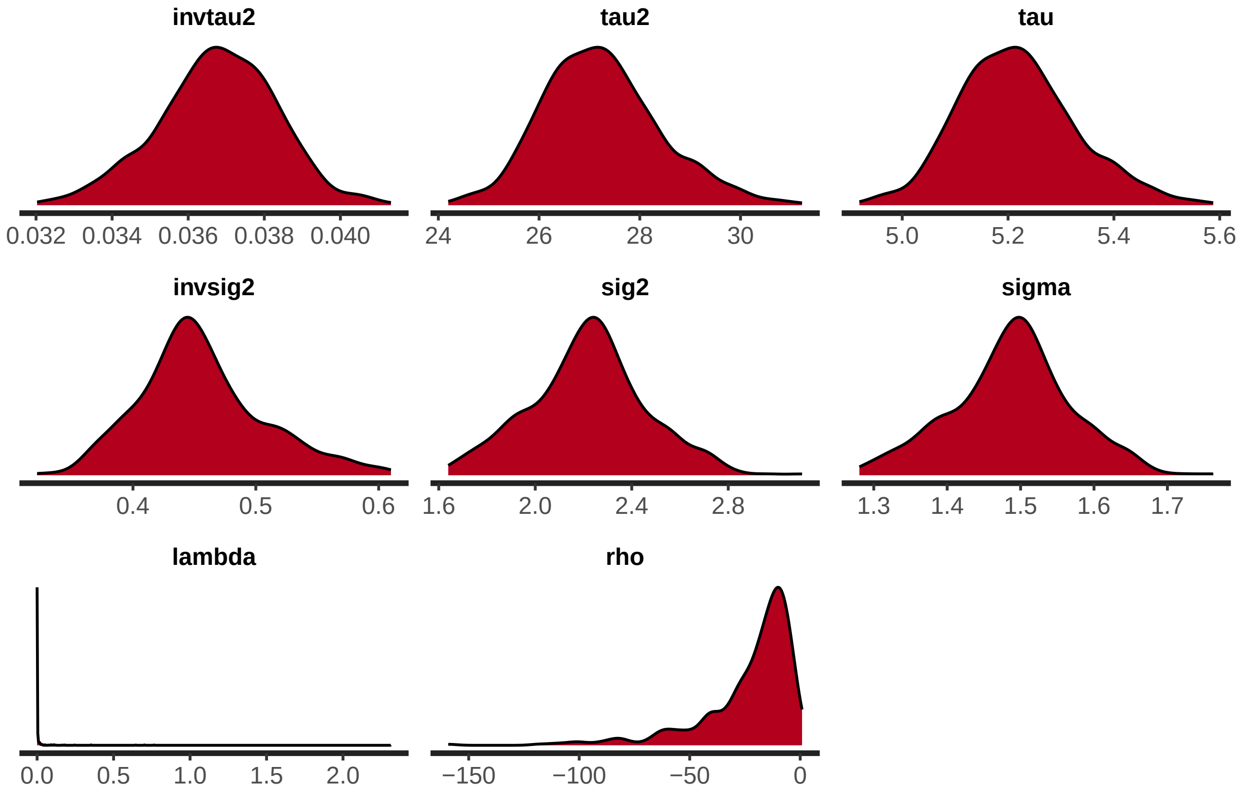
```
## Inference for Stan model: jagam_9_aneur_lme_add_incr_reslope_cases.
## 3 chains, each with iter=600; warmup=300; thin=2;
## post-warmup draws per chain=150, total post-warmup draws=450.
##
##            mean se_mean    sd   2.5%    25%    50%   75% 97.5% n_eff Rhat
## b0        -2.88    0.01  0.25  -3.38  -3.05  -2.87 -2.69 -2.41   288 1.01
## b1[1]      0.00    0.00  0.00   0.00   0.00   0.00  0.00  0.01   461 1.01
## b1[2]      0.00    0.00  0.00   0.00   0.00   0.00  0.00  0.01   474 1.00
## b1[3]      0.00    0.00  0.01   0.00   0.00   0.00  0.00  0.01   469 1.00
## invtau2    0.04    0.00  0.00   0.03   0.04   0.04  0.04  0.04   353 1.00
## tau2      27.24    0.06  1.17  25.32  26.44  27.16 27.93 29.79   352 1.00
## tau        5.22    0.01  0.11   5.03   5.14   5.21  5.28  5.46   352 1.00
## lambda     0.01    0.01  0.13   0.00   0.00   0.00  0.00  0.09   458 1.00
## rho      -23.21    2.62 21.59 -83.63 -29.16 -16.27 -8.93 -2.38    68 1.01
## invsig2    0.46    0.00  0.05   0.37   0.42   0.45  0.48  0.57   336 1.00
## sig2       2.22    0.01  0.24   1.75   2.07   2.23  2.36  2.71   292 1.00
## sigma      1.49    0.00  0.08   1.32   1.44   1.49  1.54  1.65   298 1.00
##
## Samples were drawn using NUTS(diag_e) at Wed Aug 16 11:50:45 2023.
## 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).
```

```r
stan_trace(fit.lme.add.incr.reslope.cases, pars=param.lme.add)
stan_plot(fit.lme.add.incr.reslope.cases, pars=c("b0","b1"), point_est = "mean", show_density = TRUE)
stan_plot(fit.lme.add.incr.reslope.cases, pars=c("invtau2","tau2","tau", "invsig2","sig2","sigma",  "la
stan_dens(fit.lme.add.incr.reslope.cases, pars=c("b0","b1"))
stan_dens(fit.lme.add.incr.reslope.cases, pars=c("invtau2","tau2","tau", "invsig2","sig2","sigma", "lam
```

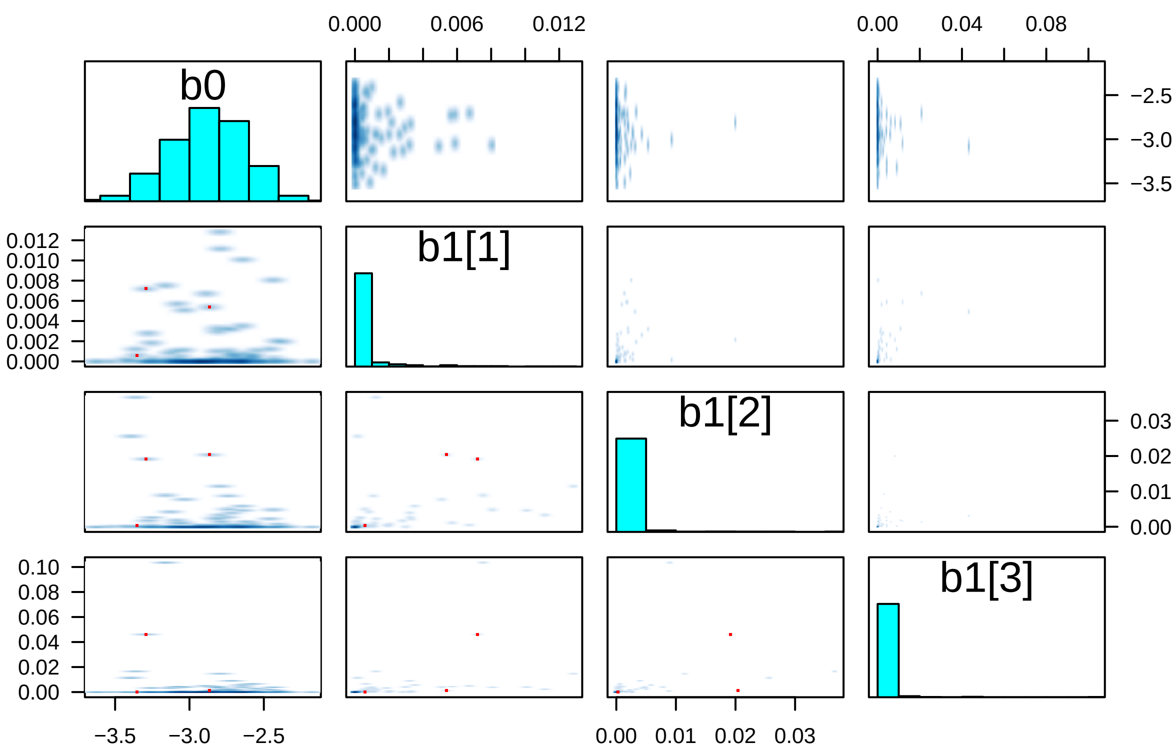

b0
b1[1]
b1[2]
b1[3]
invtau2
tau2
tau
lambda
rho
invsig2
sig2
sigma
chain
1
2
3


b0
b1[1]
b1[2]
b1[3]
−3
−2
−1
0

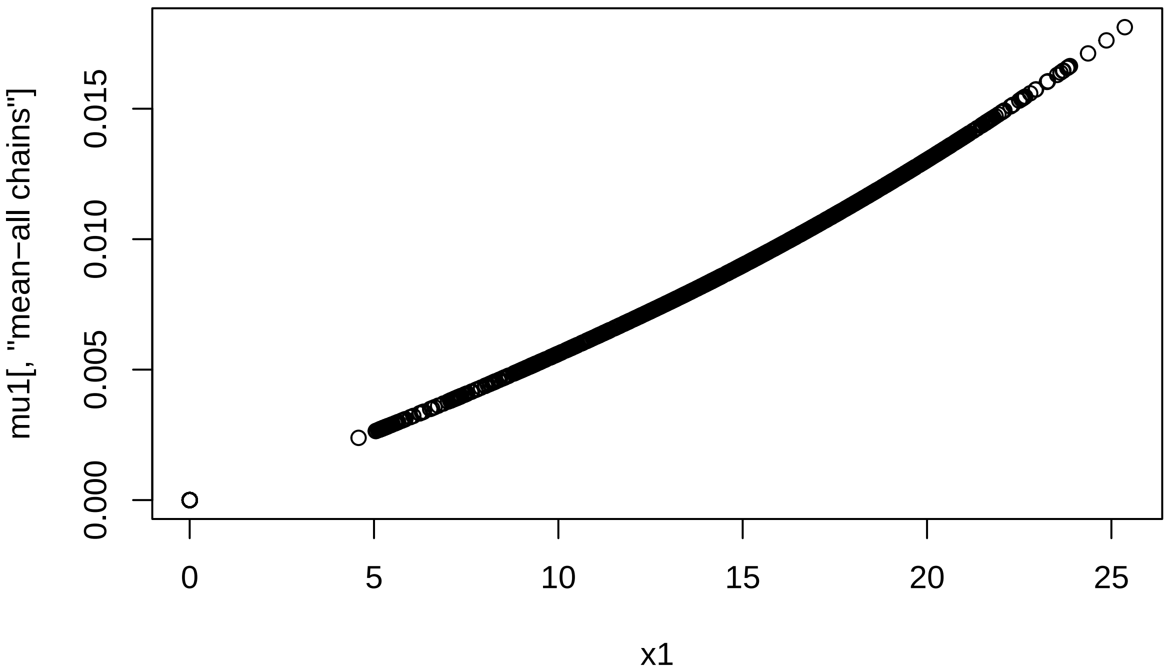

invtau2
tau2
tau
invsig2
sig2
sigma
lambda
rho
−50
0
b0
b1[1]
−3.6
−3.2
−2.8
−2.4
0.000
0.005
0.010
b1[2]
b1[3]
0.00
0.01
0.02
0.03
0.000
0.025
0.050
0.075
0.100



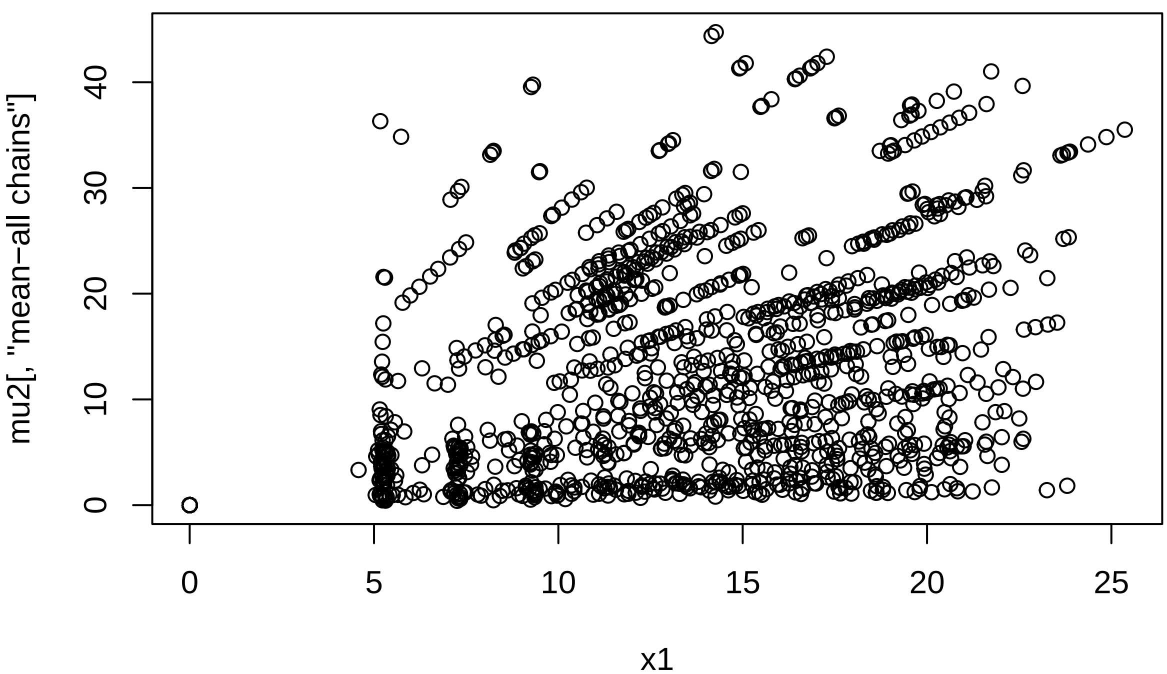

```
pairs(fit.lme.add.incr.reslope.cases, pars = c("b0","b1"), las = 1)
```



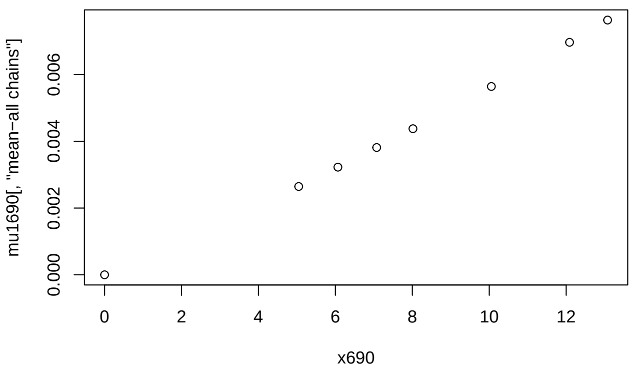

```
mu1 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1")
plot(x1,mu1[,"mean-all chains"])
```

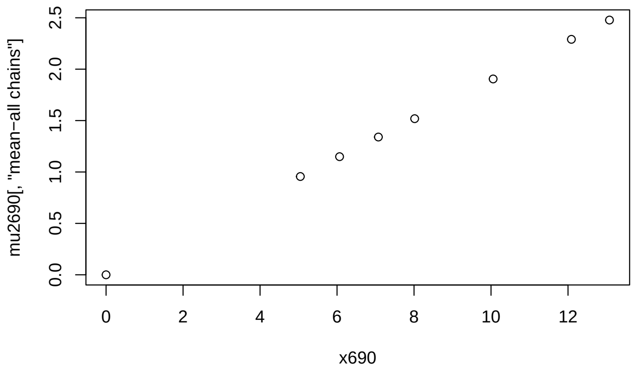



```r
mu2 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2")
plot(x1,mu2[,"mean-all chains"])
```

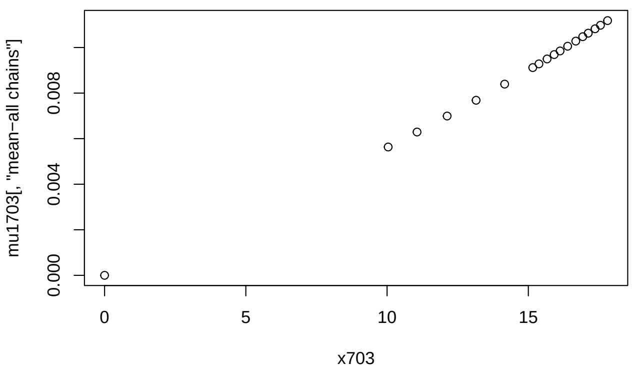


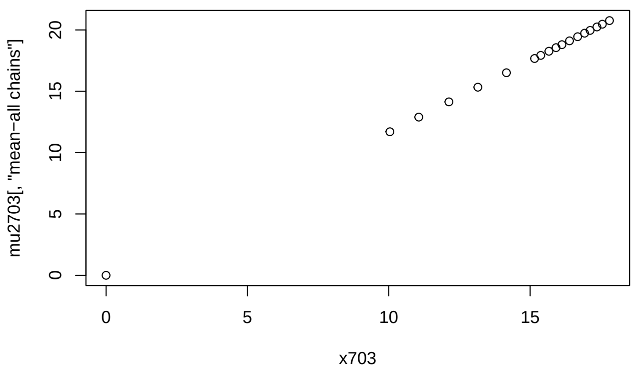

```r
mu1690 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1690")
mu2690 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2690")

plot(x690,mu1690[,"mean-all chains"])
plot(x690,mu2690[,"mean-all chains"])
```



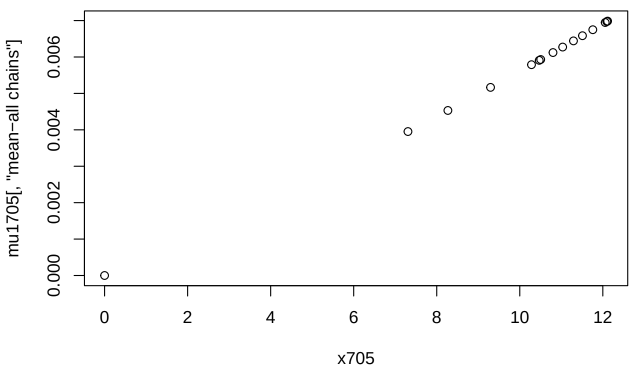


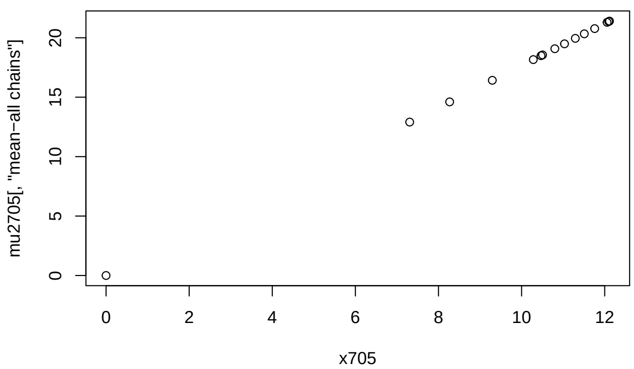

```r
mu1703 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1703")
mu2703 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2703")

plot(x703,mu1703[,"mean-all chains"])
plot(x703,mu2703[,"mean-all chains"])
```



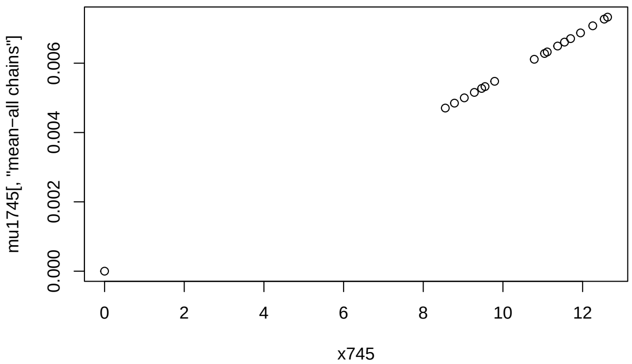


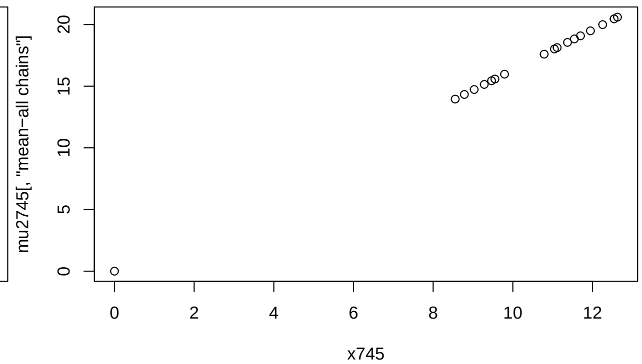

```r
mu1705 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1705")
mu2705 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2705")

plot(x705,mu1705[,"mean-all chains"])
plot(x705,mu2705[,"mean-all chains"])
```



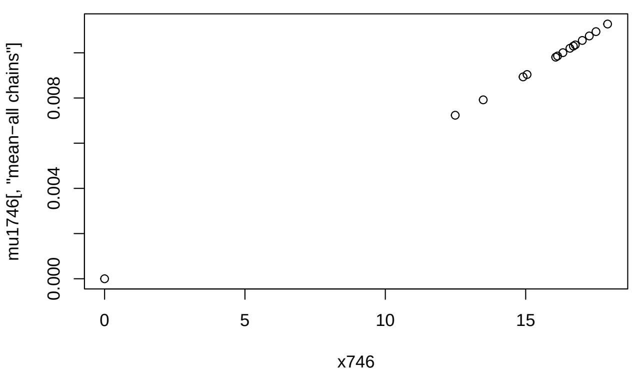


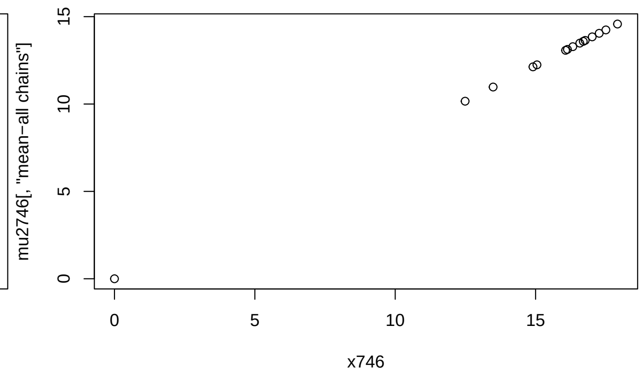

```r
mu1745 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1745")
mu2745 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2745")

plot(x745,mu1745[,"mean-all chains"])
plot(x745,mu2745[,"mean-all chains"])
```






```
mu1746 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1746")
mu2746 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2746")

plot(x746,mu1746[,"mean-all chains"])
plot(x746,mu2746[,"mean-all chains"])
```






```
mu1837 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu1837")
mu2837 = get_posterior_mean(fit.lme.add.incr.reslope.cases,"mu2837")

plot(x837,mu1837[,"mean-all chains"])
plot(x837,mu2837[,"mean-all chains"])
```