Airlines - SDS2 Project

Giacomo Maretto

03 gennaio 2019

# Airlines Fatalities

## Introduction

In this project we are analyzing data collected from 1976 to 2001 about airline fatal accidents by the International Civil Aviation Organization in Montreal , Canada (www.icao.int).

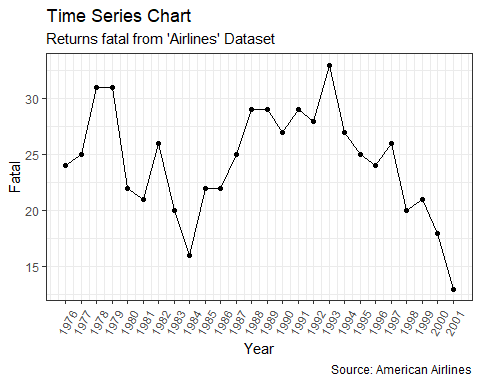
Our goal is to get a good prediction about future fatalitis through a bayesian approach.

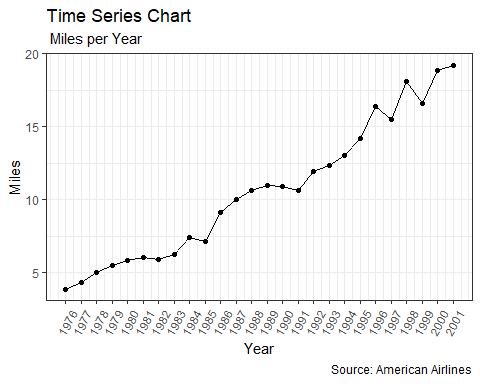
Our data is structured with four columns, year, fatal, miles, rate.

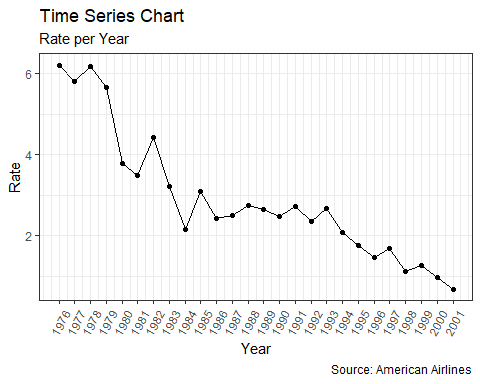
“Passenger miles” are in units of and the “accident rate” is the number of fatal accidents per passenger miles.

|  |  |  |  |
| --- | --- | --- | --- |
| year | fatal | miles | rate |
| 1976 | 24 | 3.863 | 6.213 |
| 1977 | 25 | 4.3 | 5.814 |
| 1978 | 31 | 5.027 | 6.167 |
| 1979 | 31 | 5.481 | 5.656 |
| 1980 | 22 | 5.814 | 3.784 |
| 1981 | 21 | 6.033 | 3.481 |
| 1982 | 26 | 5.877 | 4.424 |
| 1983 | 20 | 6.223 | 3.214 |
| 1984 | 16 | 7.433 | 2.152 |
| 1985 | 22 | 7.107 | 3.096 |
| 1986 | 22 | 9.1 | 2.418 |
| 1987 | 25 | 10 | 2.5 |
| 1988 | 29 | 10.6 | 2.736 |
| 1989 | 29 | 10.99 | 2.639 |
| 1990 | 27 | 10.88 | 2.482 |
| 1991 | 29 | 10.63 | 2.727 |
| 1992 | 28 | 11.96 | 2.342 |
| 1993 | 33 | 12.34 | 2.674 |
| 1994 | 27 | 13.01 | 2.075 |
| 1995 | 25 | 14.22 | 1.758 |
| 1996 | 24 | 16.37 | 1.466 |
| 1997 | 26 | 15.48 | 1.679 |
| 1998 | 20 | 18.08 | 1.106 |
| 1999 | 21 | 16.63 | 1.263 |
| 2000 | 18 | 18.88 | 0.954 |
| 2001 | 13 | 19.23 | 0.676 |

### Following three descriptive time series plot respectively about the number of fatal accidents, the miles flown and the accident rate per each year.





 I started with the simplest model i could immagine, that simply all the years look the same. This means that the number of fatal accidents in each year are independend with a Poisson(/theta) distribution. I set a non informative gamma prior distribution for , that has

The model for the data is:

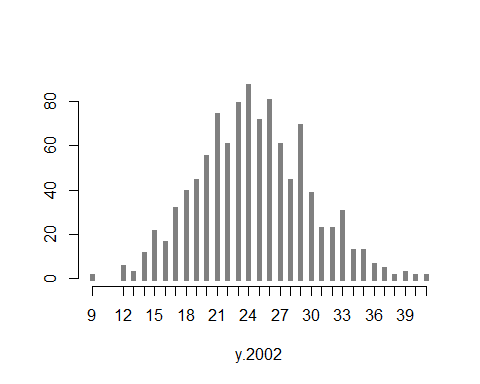
where /theta is the expected number of fatal accidents in an year. If the prior distribution for is then the posterior distrubution is , where in this case and

## Posterior distribution

The posterior distribution for is and the conditional distribution of (the number of fatal accidents in 2002) is .

So to simulate values of all we need to do is first generate a realized value form the posterior distribution of as the mean. Iterating this process will generate values of from the posterior predictive distribution. What we are doing here is integrating numerically, using simulation, over the posterior distrobution in . We can simulate this easily in R.

theta <- rgamma(1000, 634, 26 )  
y.2002 <- rpois(1000,theta)  
#hist( y.2002 )  
plot( table(y.2002), type="h", lwd=5, lend=2, col=gray(0.5), bty="n", ylab="" )

 We can specify the model in R using the package rJags.

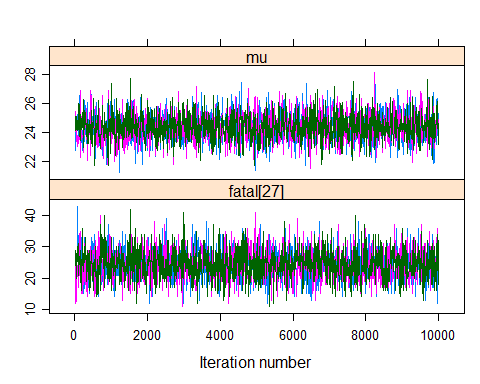
a1.par <- c("mu","fatal[27]")  
  
a1.ini <- list(list( mu=22 ),  
 list( mu=23 ),  
 list( mu=24 ) )  
  
a1.dat <- list( fatal = c(airline$fatal,NA), I=27 )  
  
# Model compilation and burn-in  
a1.mod <- jags.model( file = "a1.jag",  
 data = a1.dat,  
 inits = a1.ini,  
 n.chains = 3,  
 n.adapt = 1000 )

## Compiling model graph  
## Resolving undeclared variables  
## Allocating nodes  
## Graph information:  
## Observed stochastic nodes: 26  
## Unobserved stochastic nodes: 2  
## Total graph size: 30  
##   
## Initializing model

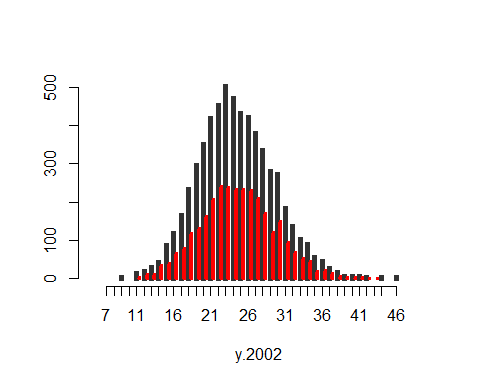
# Sampling from the posterior  
a1.res <- coda.samples( a1.mod,  
 var = a1.par,  
 n.iter = 10000,  
 thin = 10 )  
summary( a1.res )

##   
## Iterations = 10:10000  
## Thinning interval = 10   
## Number of chains = 3   
## Sample size per chain = 1000   
##   
## 1. Empirical mean and standard deviation for each variable,  
## plus standard error of the mean:  
##   
## Mean SD Naive SE Time-series SE  
## fatal[27] 24.36 4.9184 0.08980 0.08830  
## mu 24.38 0.9778 0.01785 0.01711  
##   
## 2. Quantiles for each variable:  
##   
## 2.5% 25% 50% 75% 97.5%  
## fatal[27] 15.00 21.00 24.00 28.00 34.00  
## mu 22.47 23.71 24.35 25.02 26.35

print( xyplot( a1.res[,1:2] ) )

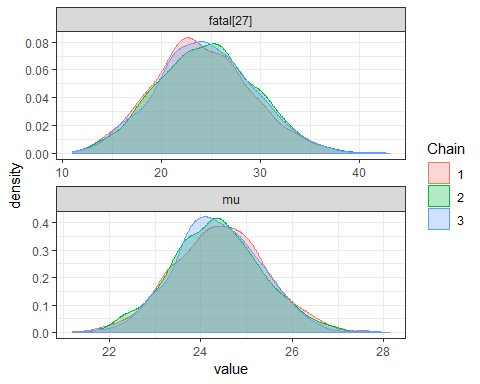


theta <- rgamma(6000, 634, 26 )  
y.2002 <- rpois(6000,theta)  
plot( table(y.2002), type="h", lwd=5, lend=2, col=gray(0.2), bty="n", ylab="", xlim=c(5,50) )  
tpr <- table( as.matrix( a1.res[,"fatal[27]"] ) )  
points( as.numeric(names(tpr))+0.4, tpr, type="h", col="red", lwd=4 )

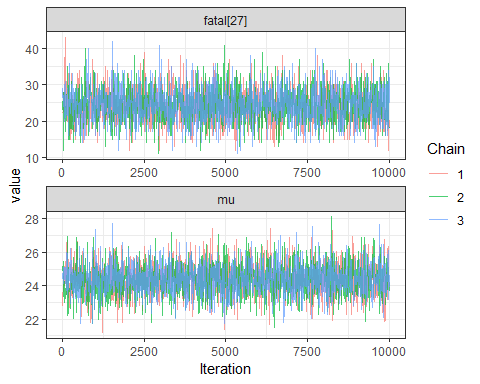


###################

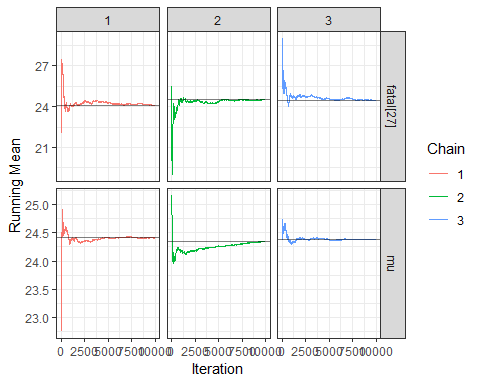
result = ggs(a1.res)  
ggs\_density(result)



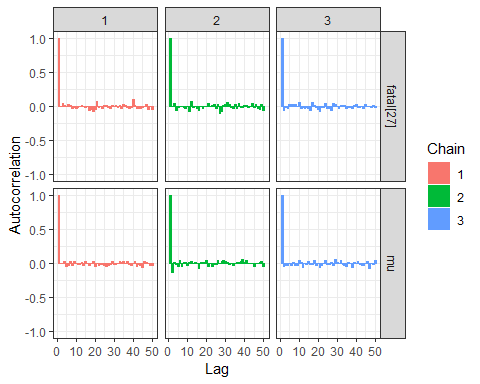
ggs\_traceplot(result)



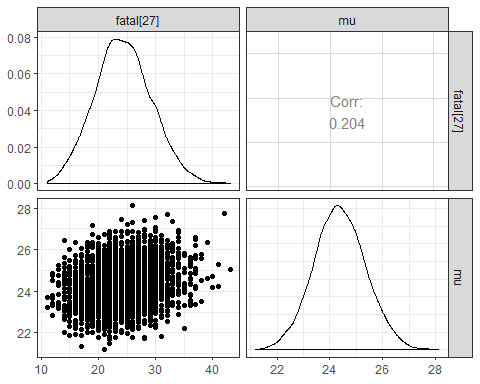
ggs\_running(result)



ggs\_autocorrelation(result)



ggs\_pairs(result) ########mmmmmeh

 ## MODEL2

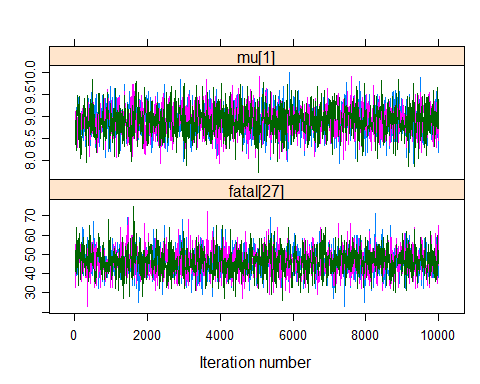
a2.ini <- list( list( lambda=10 ),list( lambda=20 ),list( lambda=30 ) )  
a2.dat <- list( fatal=c(airline$fatal,NA),miles=c(airline$miles,20), I=27 )  
a2.par <- c("mu","fatal[27]")  
  
# Model compilation and burn-in  
a2.mod <- jags.model( file = "a2.jag",data = a2.dat,inits = a2.ini,n.chains = 3,n.adapt = 1000 )

## Compiling model graph  
## Resolving undeclared variables  
## Allocating nodes  
## Graph information:  
## Observed stochastic nodes: 26  
## Unobserved stochastic nodes: 2  
## Total graph size: 84  
##   
## Initializing model

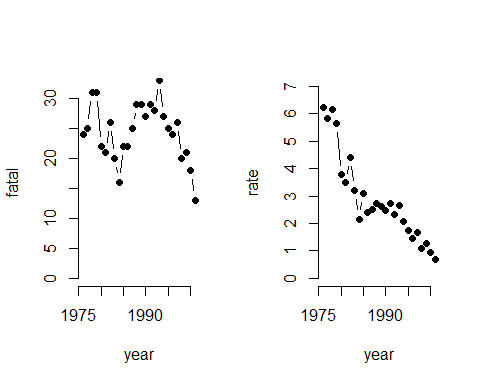
# Sampling from the posterior  
a2.res <- coda.samples( a2.mod,var = a2.par,n.iter = 10000,thin = 10 )  
summary( a2.res )

##   
## Iterations = 10:10000  
## Thinning interval = 10   
## Number of chains = 3   
## Sample size per chain = 1000   
##   
## 1. Empirical mean and standard deviation for each variable,  
## plus standard error of the mean:  
##   
## Mean SD Naive SE Time-series SE  
## fatal[27] 46.233 6.9624 0.127115 0.122617  
## mu[1] 8.897 0.3446 0.006292 0.006251  
## mu[2] 9.903 0.3836 0.007004 0.006958  
## mu[3] 11.577 0.4485 0.008188 0.008134  
## mu[4] 12.623 0.4890 0.008927 0.008869  
## mu[5] 13.390 0.5187 0.009470 0.009407  
## mu[6] 13.894 0.5382 0.009826 0.009762  
## mu[7] 13.535 0.5243 0.009572 0.009509  
## mu[8] 14.332 0.5552 0.010136 0.010069  
## mu[9] 17.118 0.6631 0.012107 0.012027  
## mu[10] 16.368 0.6340 0.011576 0.011500  
## mu[11] 20.958 0.8118 0.014822 0.014724  
## mu[12] 23.030 0.8921 0.016288 0.016181  
## mu[13] 24.412 0.9456 0.017265 0.017151  
## mu[14] 25.306 0.9803 0.017897 0.017779  
## mu[15] 25.057 0.9706 0.017721 0.017604  
## mu[16] 24.488 0.9486 0.017319 0.017205  
## mu[17] 27.535 1.0666 0.019474 0.019345  
## mu[18] 28.426 1.1011 0.020104 0.019972  
## mu[19] 29.965 1.1607 0.021192 0.021053  
## mu[20] 32.749 1.2686 0.023161 0.023009  
## mu[21] 37.703 1.4605 0.026665 0.026489  
## mu[22] 35.658 1.3813 0.025218 0.025052  
## mu[23] 41.639 1.6129 0.029448 0.029254  
## mu[24] 38.306 1.4839 0.027091 0.026913  
## mu[25] 43.470 1.6839 0.030743 0.030541  
## mu[26] 44.294 1.7158 0.031326 0.031120  
## mu[27] 46.061 1.7842 0.032575 0.032361  
##   
## 2. Quantiles for each variable:  
##   
## 2.5% 25% 50% 75% 97.5%  
## fatal[27] 33.000 41.000 46.000 51.000 60.000  
## mu[1] 8.209 8.671 8.898 9.128 9.562  
## mu[2] 9.138 9.652 9.905 10.160 10.643  
## mu[3] 10.683 11.284 11.579 11.878 12.443  
## mu[4] 11.648 12.303 12.625 12.951 13.566  
## mu[5] 12.356 13.051 13.392 13.738 14.391  
## mu[6] 12.821 13.542 13.896 14.255 14.933  
## mu[7] 12.490 13.192 13.537 13.886 14.546  
## mu[8] 13.225 13.969 14.334 14.704 15.403  
## mu[9] 15.796 16.685 17.121 17.563 18.398  
## mu[10] 15.103 15.953 16.370 16.793 17.591  
## mu[11] 19.339 20.427 20.961 21.502 22.524  
## mu[12] 21.251 22.447 23.034 23.628 24.752  
## mu[13] 22.527 23.794 24.416 25.046 26.237  
## mu[14] 23.351 24.665 25.310 25.963 27.197  
## mu[15] 23.122 24.422 25.061 25.708 26.930  
## mu[16] 22.597 23.868 24.492 25.124 26.318  
## mu[17] 25.408 26.838 27.540 28.250 29.593  
## mu[18] 26.231 27.706 28.431 29.165 30.551  
## mu[19] 27.650 29.206 29.970 30.743 32.204  
## mu[20] 30.220 31.920 32.754 33.600 35.197  
## mu[21] 34.791 36.748 37.709 38.682 40.521  
## mu[22] 32.904 34.755 35.664 36.584 38.323  
## mu[23] 38.423 40.584 41.646 42.720 44.751  
## mu[24] 35.348 37.336 38.313 39.301 41.169  
## mu[25] 40.112 42.369 43.477 44.599 46.719  
## mu[26] 40.873 43.172 44.301 45.444 47.605  
## mu[27] 42.503 44.894 46.068 47.257 49.503

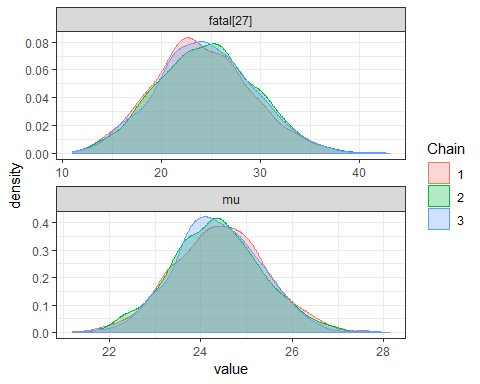
print( xyplot( a2.res[,1:2] ) )



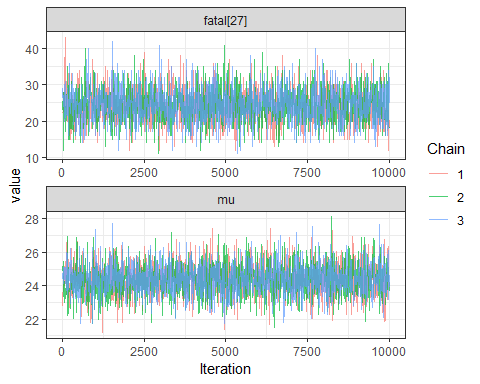
################################   
#A closer inspection of the number of fatal airline crashes can be dome by:  
par(mfrow=c(1,2))  
with(airline, plot( year, fatal, pch=16, type="b", ylim=c(0,32), bty="n" ) )  
with(airline, plot( year, rate, pch=16, type="b", ylim=c(0,7), bty="n" ) )



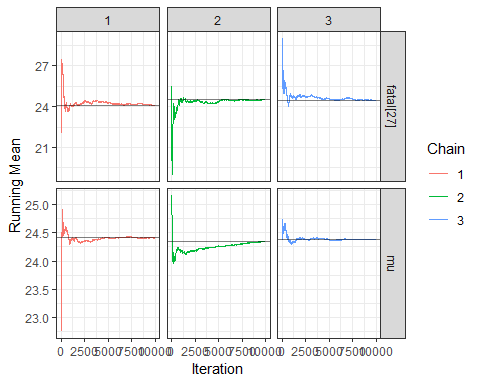
result2 = ggs(a2.res)  
ggs\_density(result)



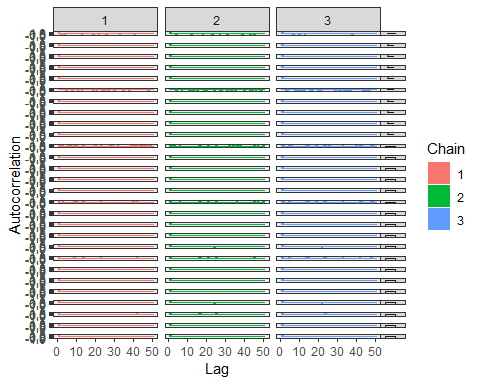
ggs\_traceplot(result)



ggs\_running(result)



ggs\_autocorrelation(result2)

 ## MODEL 3

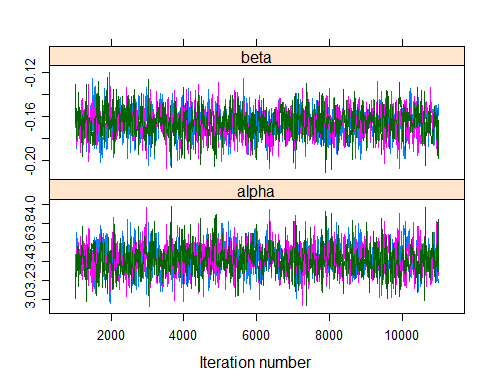
a3.ini <- list( list( alpha=10, beta=-0.5 ),list( alpha=20, beta=-0.6 ),list( alpha=30, beta=-0.4 ) )  
  
a3.dat <- list( fatal=c(airline$fatal,NA),miles=c(airline$miles,20), I=27 )  
  
a3.par <- c("alpha","beta","fatal[27]")  
  
# Model compilation and burn-in  
a3.mod <- jags.model( file = "a3.jag",data = a3.dat,inits = a3.ini,n.chains = 3,n.adapt = 1000 )

## Compiling model graph  
## Resolving undeclared variables  
## Allocating nodes  
## Graph information:  
## Observed stochastic nodes: 26  
## Unobserved stochastic nodes: 3  
## Total graph size: 194  
##   
## Initializing model

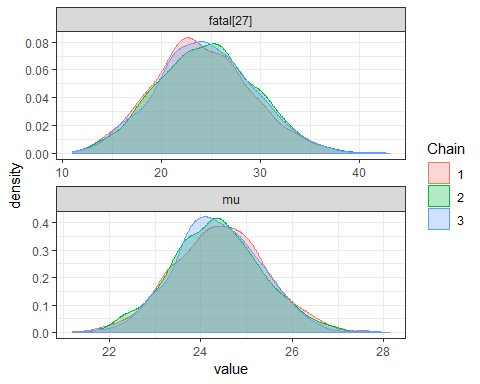
# Sampling from the posterior  
a3.res <- coda.samples( a3.mod,var = a3.par,n.iter = 10000,thin = 10 )  
summary( a3.res )

##   
## Iterations = 1010:11000  
## Thinning interval = 10   
## Number of chains = 3   
## Sample size per chain = 1000   
##   
## 1. Empirical mean and standard deviation for each variable,  
## plus standard error of the mean:  
##   
## Mean SD Naive SE Time-series SE  
## alpha 3.4305 0.15686 0.0028638 0.0035699  
## beta -0.1662 0.01325 0.0002419 0.0002932  
## fatal[27] 12.1023 4.17824 0.0762839 0.0751099  
##   
## 2. Quantiles for each variable:  
##   
## 2.5% 25% 50% 75% 97.5%  
## alpha 3.1328 3.3272 3.4241 3.5334 3.7497  
## beta -0.1936 -0.1749 -0.1661 -0.1572 -0.1406  
## fatal[27] 5.0000 9.0000 12.0000 15.0000 21.0000

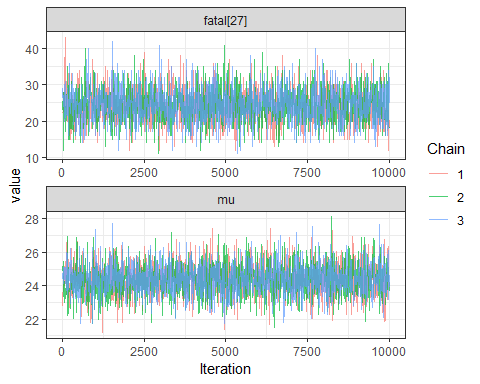
print( xyplot( a3.res[,1:2] ) )



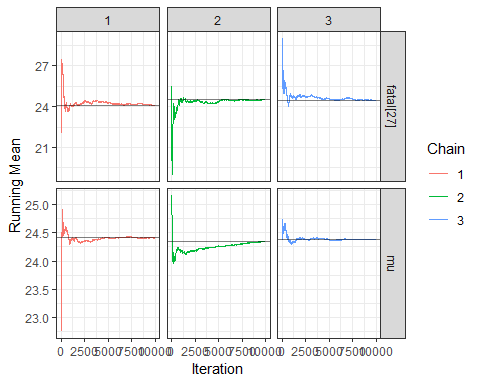
result3 = ggs(a3.res)  
ggs\_density(result)



ggs\_traceplot(result)



ggs\_running(result)



ggs\_autocorrelation(result3)

