library(jagsUI)

library(rstan)

require(loo)

options(scipen=99)

# 365 points in x (Bitcoin price in $000s)

x=c(0.998,1.018,1.031,1.130,1.006,0.896,0.905,0.914,0.899,0.905,0.776,0.803,0.826,0.819,0.822,

0.830,0.904,0.884,0.898,0.892,0.920,0.925,0.913,0.886,0.894,0.915,0.919,0.920,0.914,0.920,0.968,

0.987,1.008,1.018,1.034,1.014,1.025,1.053,1.054,0.989,0.993,1.010,1.000,0.997,1.009,1.009,1.034,

1.053,1.058,1.053,1.083,1.124,1.122,1.178,1.181,1.152,1.180,1.194,1.191,1.230,1.261,1.291,1.268,

1.278,1.281,1.232,1.150,1.192,1.117,1.182,1.230,1.243,1.246,1.260,1.173,1.070,0.971,1.018,1.041,

1.115,1.037,1.030,0.936,0.965,0.965,1.040,1.044,1.040,1.038,1.080,1.090,1.099,1.148,1.144,1.135,

1.191,1.193,1.185,1.210,1.213,1.225,1.217,1.179,1.183,1.181,1.185,1.204,1.218,1.227,1.255,1.257,

1.244,1.248,1.248,1.264,1.285,1.329,1.320,1.327,1.348,1.402,1.444,1.492,1.516,1.512,1.548,1.555,

1.639,1.707,1.757,1.807,1.677,1.760,1.772,1.697,1.718,1.802,1.887,1.968,2.052,2.056,2.139,2.291,

2.476,2.358,2.247,2.106,2.208,2.290,2.197,2.330,2.452,2.517,2.556,2.553,2.737,2.914,2.694,2.825,

2.827,2.942,3.019,2.683,2.739,2.494,2.457,2.528,2.664,2.576,2.642,2.779,2.712,2.741,2.738,2.619,

2.594,2.485,2.593,2.585,2.562,2.500,2.460,2.530,2.581,2.625,2.629,2.619,2.521,2.580,2.526,2.372,

2.332,2.423,2.365,2.233,1.993,1.939,2.244,2.328,2.294,2.877,2.694,2.839,2.763,2.779,2.591,2.550,

2.697,2.805,2.720,2.746,2.874,2.736,2.724,2.814,2.884,3.302,3.255,3.432,3.453,3.378,3.445,3.680,

3.918,4.111,4.383,4.204,4.425,4.316,4.159,4.206,4.111,4.055,4.138,4.191,4.362,4.408,4.387,4.395,

4.440,4.648,4.631,4.765,4.951,4.644,4.632,4.320,4.422,4.627,4.638,4.318,4.292,4.191,4.189,4.148,

3.874,3.226,3.687,3.679,3.673,4.067,3.897,3.858,3.613,3.603,3.777,3.662,3.928,3.896,4.209,4.185,

4.164,4.353,4.395,4.404,4.320,4.226,4.323,4.370,4.437,4.597,4.773,4.755,4.831,5.439,5.640,5.810,

5.697,5.754,5.595,5.572,5.700,5.984,6.013,5.985,5.895,5.519,5.734,5.888,5.768,5.733,6.141,6.122,

6.448,6.750,7.030,7.161,7.387,7.382,6.958,7.119,7.459,7.147,6.570,6.337,5.857,6.518,6.599,7.279,

7.844,7.690,7.777,8.034,8.238,8.096,8.231,8.003,8.201,8.764,9.327,9.739,9.908,9.816,9.917,10.86,

10.895,11.181,11.617,11.696,13.709,16.858,16.057,14.913,15.037,16.700,17.178,16.407,16.531,

17.602,19.343,19.087,18.961,17.608,16.455,15.561,13.857,14.549,13.975,13.917,15.745,15.378,

14.429,14.428,12.630,13.86)

# returns calculation

T=364

y=numeric(T)

for (t in 1:T) {y[t] <- (x[t+1]-x[t])/x[t]}

D=list(y=y,T=T)

#

**# double autoregressive model**

#

**cat("**model { for (t in 1:T) {y[t] ~ dnorm(mu[t],tau[t])

LL[t] <- -0.92+0.5\*log(tau[t])-0.5\*tau[t]\*(y[t]-mu[t])\*(y[t]-mu[t])}

y0 ~ dnorm(0,1)

mu[1] <- rho\*y0

tau[1] <- 1/(gam+alph\*y0\*y0)

for (t in 2:T) { mu[t] <- rho\*y[t-1]

tau[t] <- 1/(gam+alph\*y[t-1]\*y[t-1])}

rho <- sqrt(rho2)

gam ~ dgamma(1,1)

rho2 <- r[1]/sum(r[])

alph <- r[2]/sum(r[])

for (j in 1:3) {r[j] ~ dgamma(1,1)}}

", file="model1.jag")

inits1 <- list(r=c(0.4,0.4,0.4),gam=1)

inits2 <- list(r=c(0.8,0.8,0.8),gam=0.5)

inits=list(inits1,inits2)

pars = c("rho","gam","alph","LL")

R1 = autojags(D, inits, pars,model.file="model1.jag",2,iter.increment=5000, n.burnin=500,Rhat.limit=1.1, max.iter=50000, seed=1234,codaOnly=c("LL"))

R1$summary

**# Fit measures**

loo(as.matrix(R1$sims.list$LL))

#

**# Stochastic Volatility**

**#**

SV.stan <- "

data {

int<lower=0> T; // # time points (equally spaced)

vector[T] y; // mean corrected return at time t

}

parameters {

real mu; // mean log volatility

real<lower=-1,upper=1> phi; // persistence of volatility

real<lower=0> sigma; // white noise shock scale

vector[T] h; // log volatility at time t

}

transformed parameters {

vector[T] V;

V = exp(h);

}

model {

phi ~ uniform(-1,1);

sigma ~ cauchy(0,5);

mu ~ cauchy(0,10);

h[1] ~ normal(mu, sigma / sqrt(1 - phi \* phi));

for (t in 2:T)

h[t] ~ normal(mu + phi \* (h[t - 1] - mu), sigma);

for (t in 1:T)

y[t] ~ normal(0, exp(h[t] / 2));

}

generated quantities {

vector[T] log\_lik;

for (t in 1:T) { log\_lik[t] = normal\_lpdf(y[t] |0 , exp(h[t]/2));

}}

"

**# Compilation and Estimation**

sm =stan\_model(model\_code=SV.stan)

R2 =sampling(sm,data=D,pars=c("mu","phi","log\_lik","V"),iter=2000,warmup=200,chains=2,

seed=12345)

print(R2)

**# Fit**

loo(as.matrix(R2,pars="log\_lik"))

**# Variance through time**

V= apply(as.matrix(R2,pars="V"),2,mean)

plot(V,ylab="Variance")

**#**

**# Student t Volatility (Scale Mixture)**

**#**

SV\_student.stan <- "

data {

int<lower=0> T; // # time points (equally spaced)

vector[T] y; // mean corrected return at time t

}

parameters {

real mu; // mean log volatility

real<lower=-1,upper=1> phi; // persistence of volatility

real<lower=0> sigma; // white noise shock scale

vector[T] h; // log volatility at time t

real<lower=0> inv\_lam[T]; // precision scaling factors

real<lower=1> nu;

}

transformed parameters { real lam[T];

for (t in 1:T) {lam[t] = 1/inv\_lam[t];}

}

model {

phi ~ uniform(-1,1);

sigma ~ cauchy(0,5);

inv\_lam ~ gamma(nu/2,nu/2);

mu ~ cauchy(0,10);

nu ~ exponential(0.1);

h[1] ~ normal(mu, sigma / sqrt(1 - phi \* phi));

for (t in 2:T) {h[t] ~ normal(mu + phi \* (h[t - 1] - mu), sigma);}

for (t in 1:T) {y[t] ~ normal(0, sqrt(lam[t])\*exp(h[t] / 2));}

}

generated quantities {

vector[T] log\_lik;

for (t in 1:T) { log\_lik[t] = normal\_lpdf(y[t] |0 , sqrt(lam[t])\*exp(h[t]/2));

}}

"

sm =stan\_model(model\_code=SV\_student.stan)

R3=sampling(sm,data=D,iter=2000,warmup=200,chains=2,seed=12345)

print(R3)

**# Fit**

loo(as.matrix(R3,pars="log\_lik"))

**# Ten lowest precision adjustments**

inv\_lam\_mn <- apply(as.matrix(R3,pars="inv\_lam"),2,mean)

index=seq(1:364)

list <- data.frame(index,inv\_lam\_mn,y)

list=list[order(list$inv\_lam\_mn),]

head(list,10)

#

**# state space stochastic volatility with jumps**

#

**cat("**model { for (t in 1:T) { tau[t] <- 1/exp(th[t])

y[t] ~ dnorm(J[t]\*N[t],tau[t])

N[t] ~ dbern(lambda)

J[t] ~ dnorm(0,tau.J)

LL[t] <- -0.92+0.5\*log(tau[t])-0.5\*tau[t]\*(y[t]-J[t]\*N[t])\*(y[t]-J[t]\*N[t])

V[t] <- exp(th[t])}

# priors

lambda ~ dbeta(2,48)

Nsum <- sum(N[])

tau.J ~ dgamma(1,0.001)

mu ~ dnorm(0,0.1)

phistar ~ dbeta(20,1.5);

phi <- 2\*phistar-1

invW ~ dgamma(2.5,0.025)

W <- 1/invW

th0 ~ dnorm(mu,invW)

th.1 <- mu + phi\*(th0-mu)

th[1] ~ dnorm(th.1,invW)

for (t in 2:T) {th.m[t] <- mu + phi\*(th[t-1]-mu)

th[t] ~ dnorm(th.m[t],invW)}}

", file="model2.jag")

inits1 <- list(invW=0.1,phistar=0.5,mu=-6,th=rep(-6,T),tau.J=1,N=rep(0,T),lambda=0.01)

inits2 <- list(invW=0.2,phistar=0.6,mu=-6.5,th=rep(-6.5,T),tau.J=2,N=rep(0,T),lambda=0.02)

inits=list(inits1,inits2)

pars = c("phi","mu","LL","Nsum","N","lambda")

R4 = autojags(D, inits, pars,model.file="model2.jag",2,iter.increment=5000, n.burnin=500,Rhat.limit=1.1, max.iter=50000, seed=1234,codaOnly=c("LL","N"))

R4$summary

**# Fit measures**

loo(as.matrix(R4$sims.list$LL))

# N is estimated probability that N[t]=1

N=as.matrix(R4$sims.list$N)

N <- apply(N,2,mean)

**# Ten highest outlier probabilities**

index=seq(1:364)

list <- data.frame(index,N,y)

list=list[order(-list$N),]

head(list,10)