Computer Lab 4
732A54 - Bayesian Learning

Jooyoung Lee - ***joole336***
Zuxiang Li - ***zuxli371***

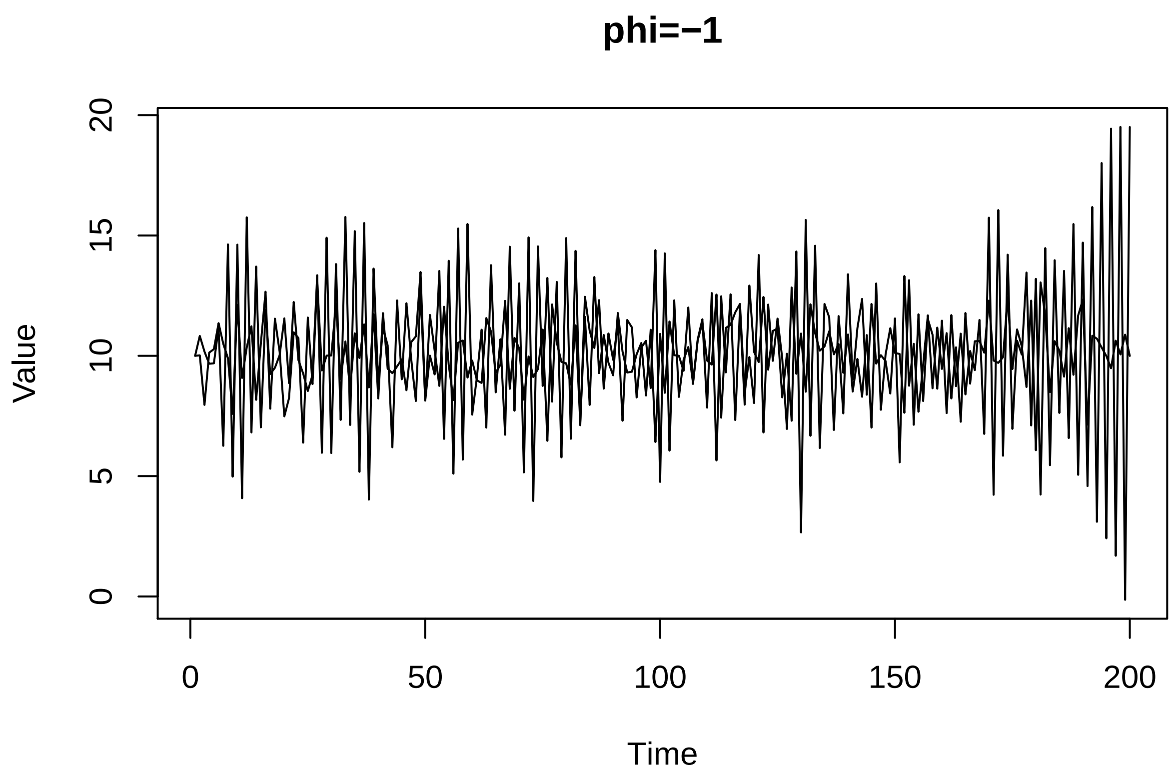
May 24, 2020

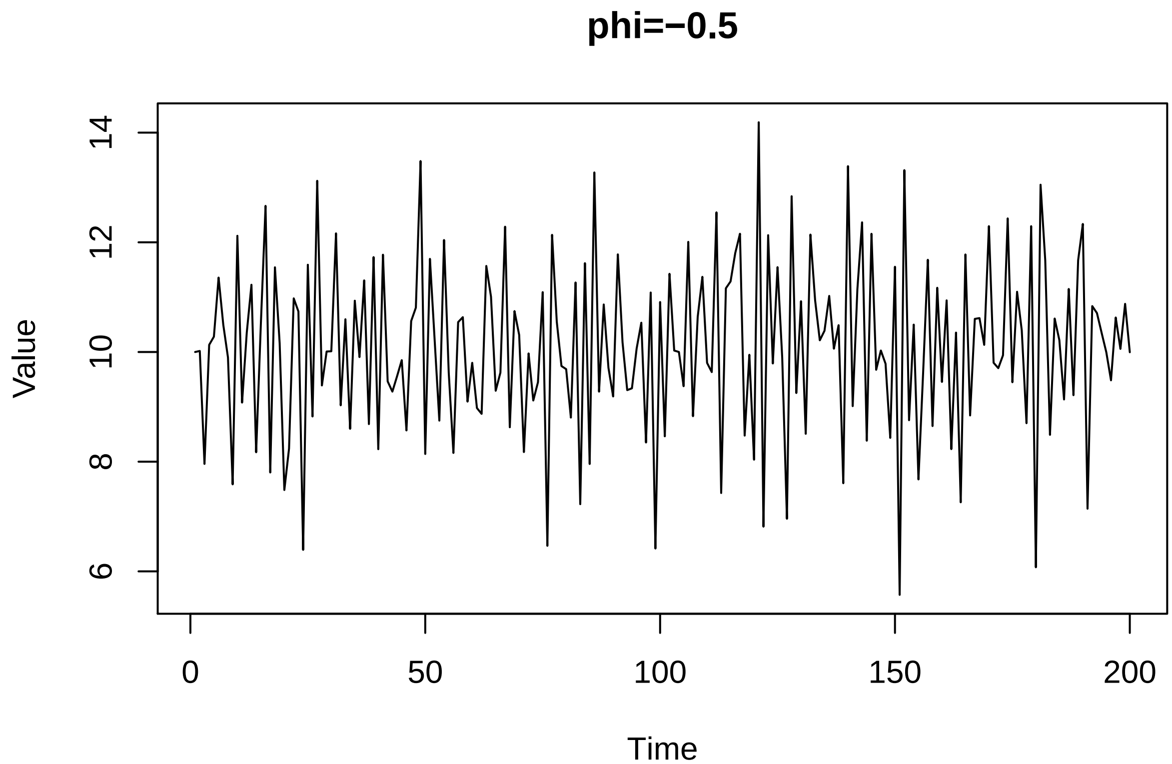
1. Time series models in Stan

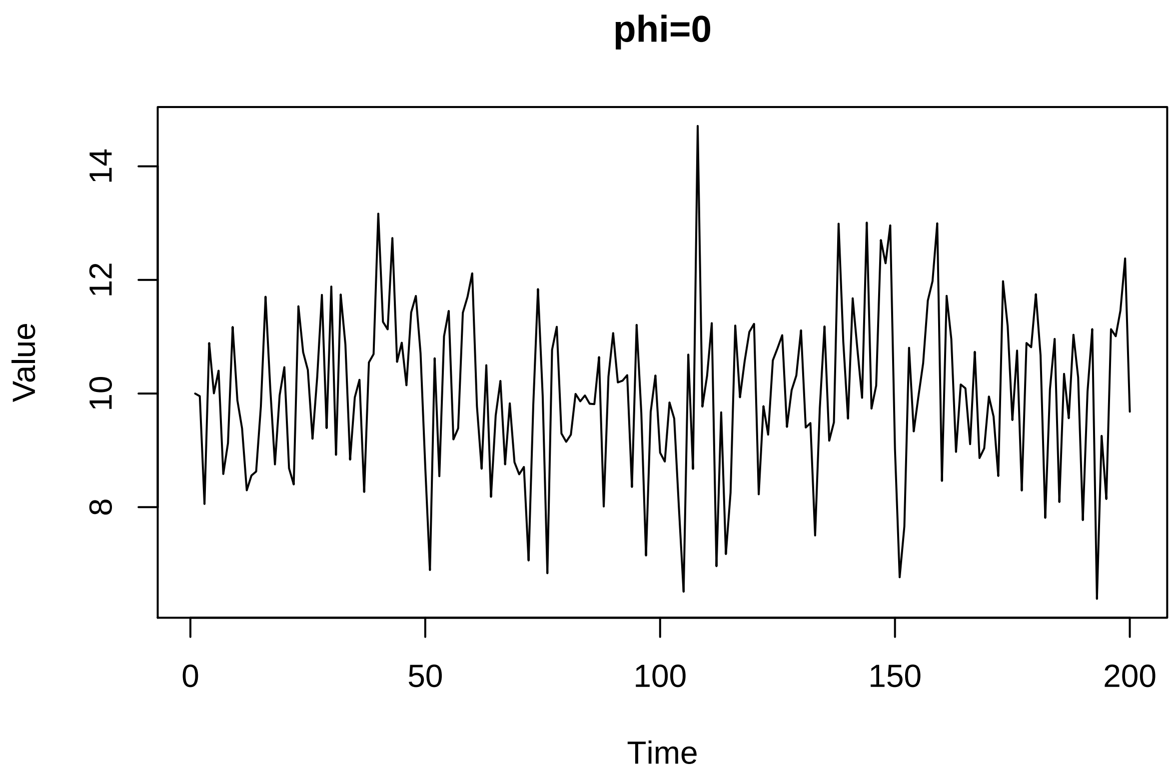
(a) Write a function in R that simulates data from the AR(1)-process

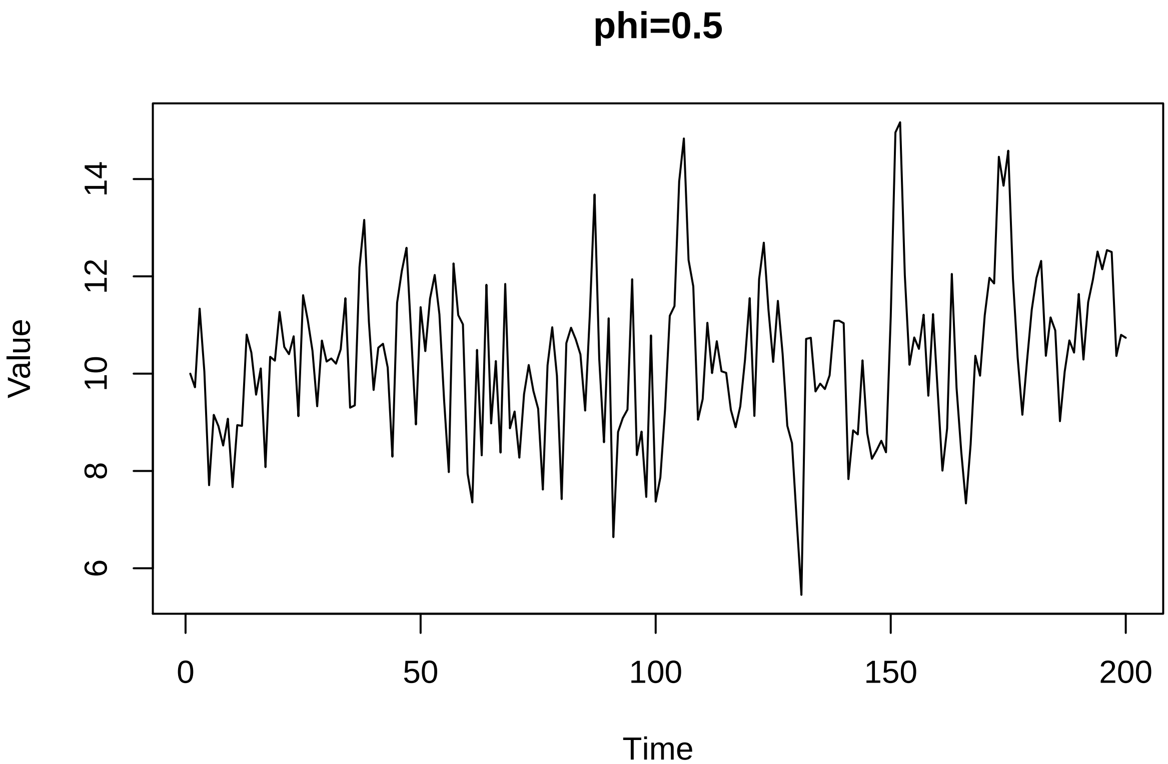
$$x_t = \mu + \phi(x_{t-1} - \mu) + \epsilon_t, \epsilon_t \sim N(0, \sigma^2)$$

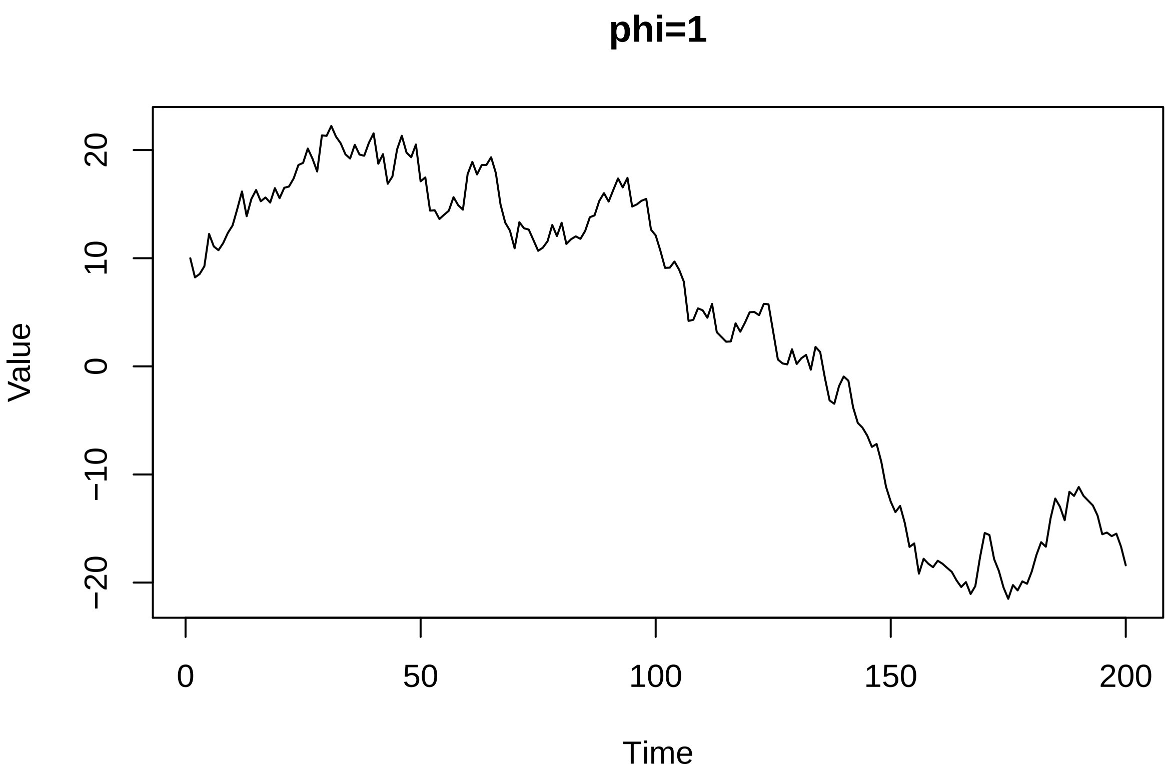
for given values of μ,ϕ and σ^2. Start the process at $x_1 = \mu$ and then simulate values for x_t for $t = 2, 3, \cdots, T$ and return the vector $x_{1:T}$ containing all time points. Use $\mu = 10$, $\sigma^2 = 2$ and $T = 200$ and look at some different realizations (simulations) of $x_{1:T}$ for values of ϕ betweent -1 and 1 (this is the interval of ϕ where the AR(1)-process is stable). Include a plot of at least one realization in the report. What effect does the value of ϕ have on $x_{1:T}$




phi=−0.5
Value
Time
0
50
100
150
200
6
8
10
12
14


phi=0
Value
Time
0
50
100
150
200
8
10
12
14





As ϕ increases, the value between two continuous observations become less irregular.

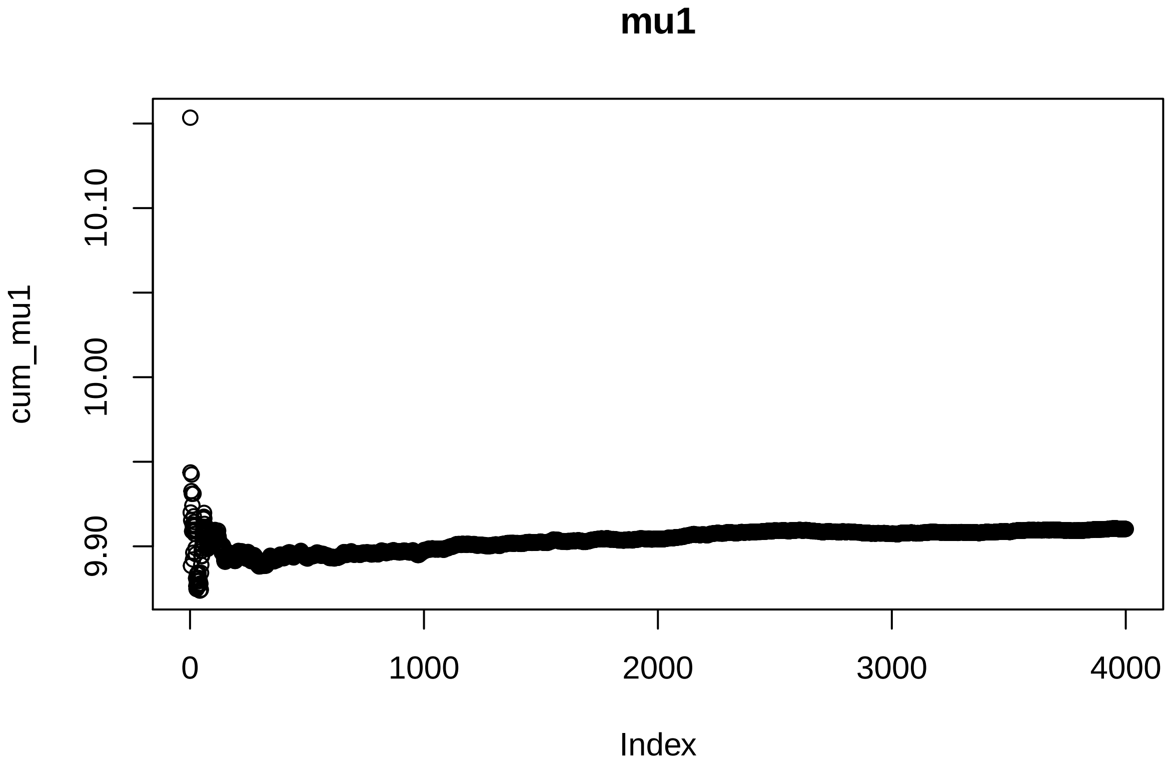
(b) Use your function from a) to simulate two AR(1)-processes, $x_{1:T}$ with $\phi = 0.3$ and $y_{1:T}$ with $\phi = 0.95$. Now, treat your simulated vectors as synthetic data, and treat the values of μ, ϕ and σ^2 as unknown and estimate them using MCMC. Implement Stan-code that samples from the posterior of the three parameters, using suitable non-informative priors of your choice. [Hint: Look at the time-series models examples in the Stan user's guide/reference manual, and note the different parameterization used here.]

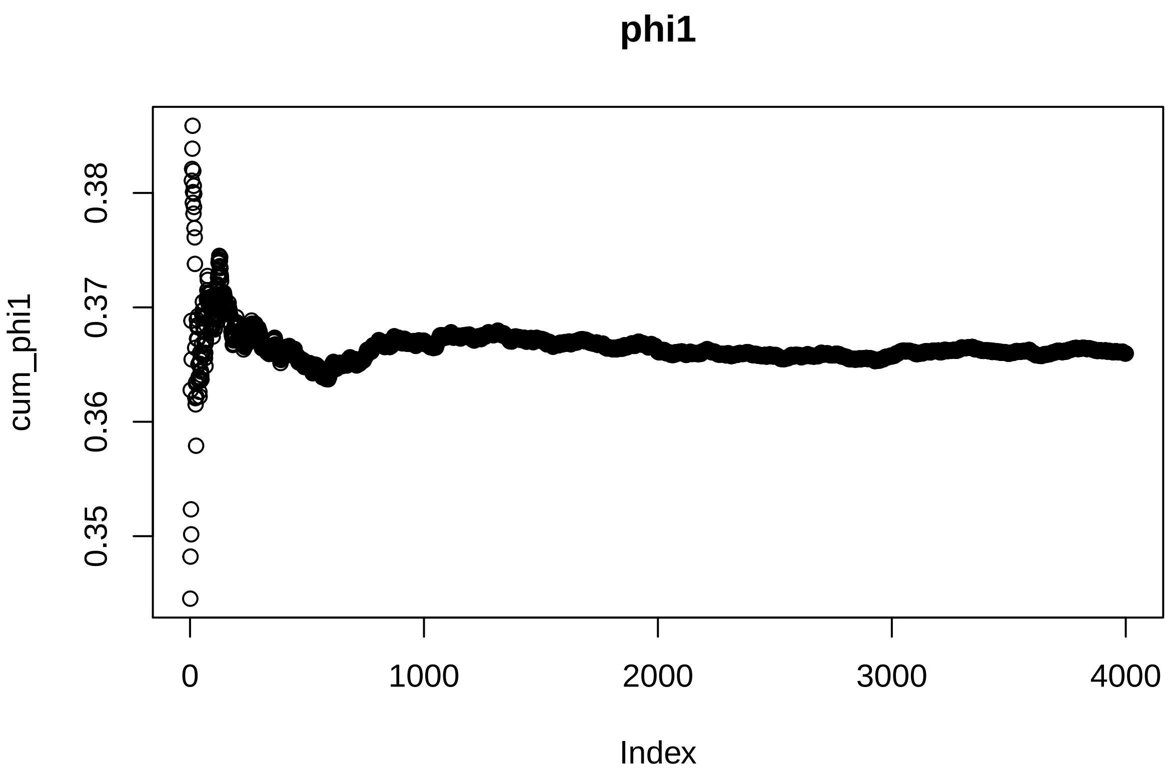
i. Report the posterior mean, 95% credible intervals and the number of effective posterior samples for the three inferred parameters for each of the simulated AR(1)-process. Are you able to estimate the true values?

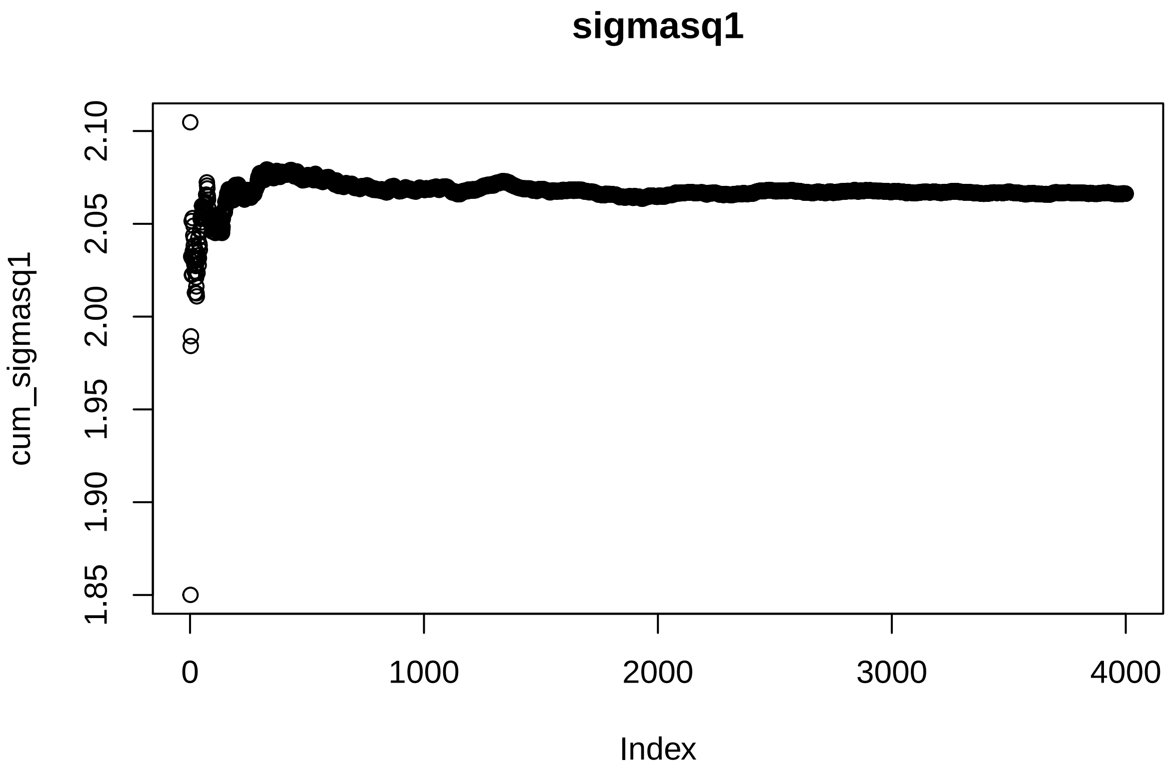
ii. For each of the two data sets, evaluate the convergence of the samplers and plot the joint posterior of μ and ϕ. Comments?

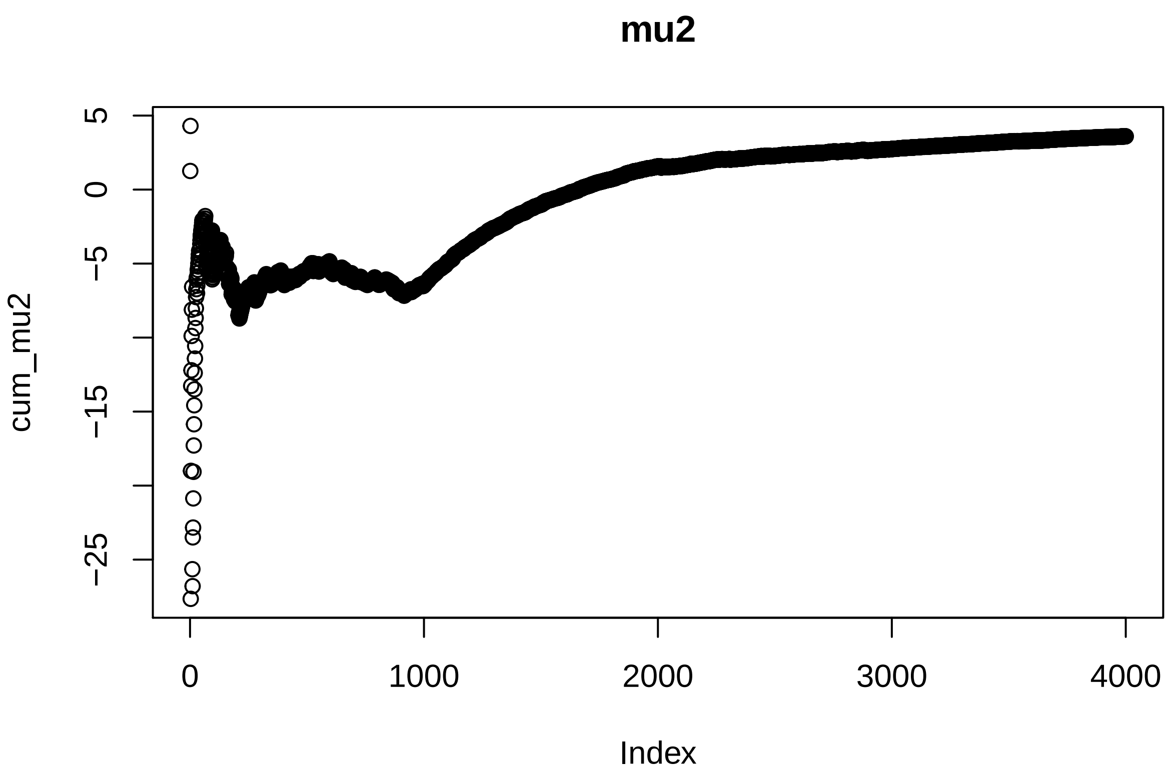
```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Frameworks/R.frame
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/3.6/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/3.6/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/include/Eigen/Core:96:10: fa
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1

## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess

## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess

##                 mean     se_mean         sd         2.5%       97.5%    n_eff
## mu         9.9102375 0.002901337 0.16580712    9.5872953   10.241383 3265.954
## phi        0.3659702 0.001213694 0.06704012    0.2337117    0.496231 3051.061
## sigmasq    2.0662497 0.003638547 0.20894250    1.6970299    2.512204 3297.597
## lp__    -172.8634357 0.030319634 1.27933138 -176.1181671 -171.431126 1780.402
##              Rhat
## mu      1.0006961
## phi     0.9996560
## sigmasq 1.0002414
## lp__    0.9998415

##                 mean     se_mean          sd          2.5%        97.5%
## mu         3.5993348 2.554492254 24.15247831  -46.8175515   19.8991572
## phi        0.9506787 0.001845377  0.03003343    0.8920551    0.9993169
## sigmasq    2.0503557 0.005179958  0.20881355    1.6822719    2.5092713
## lp__    -174.1050826 0.209324646  2.47083279 -180.7208139 -171.5372357
##            n_eff     Rhat
## mu       89.3952 1.066689
```

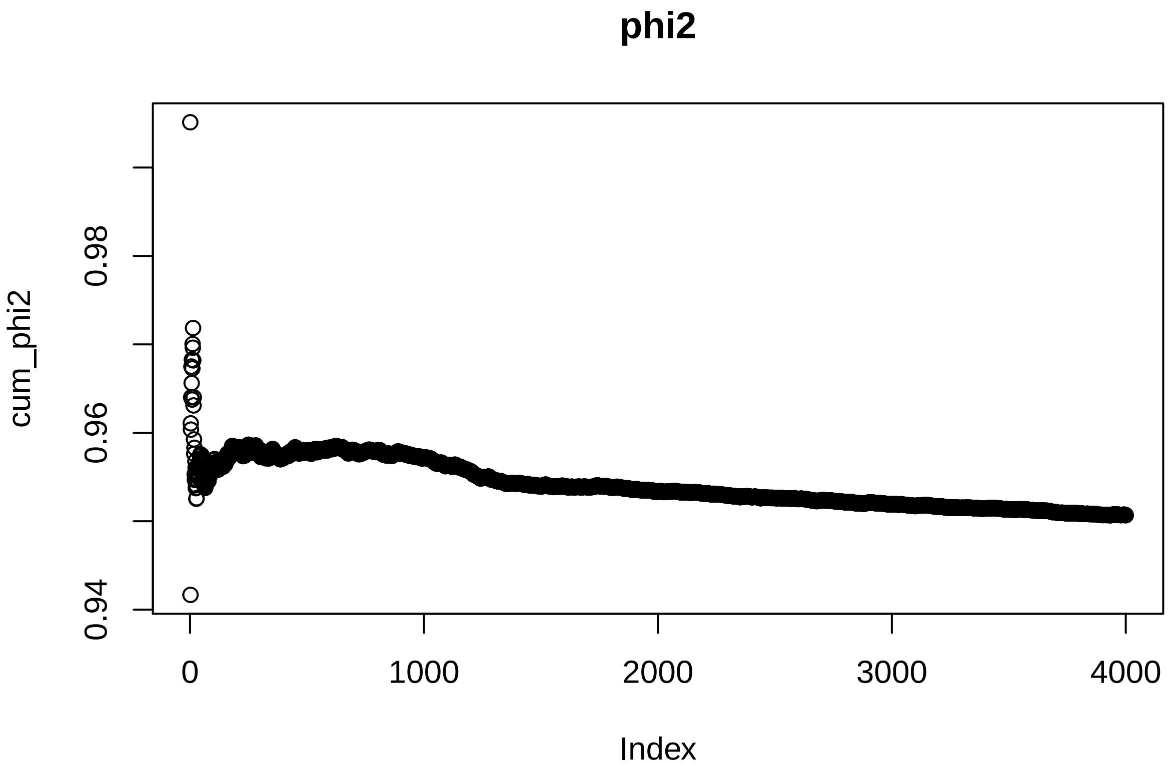
```
## phi       264.8741 1.015448
## sigmasq 1625.0431 1.000582
## lp__      139.3304 1.037456
```

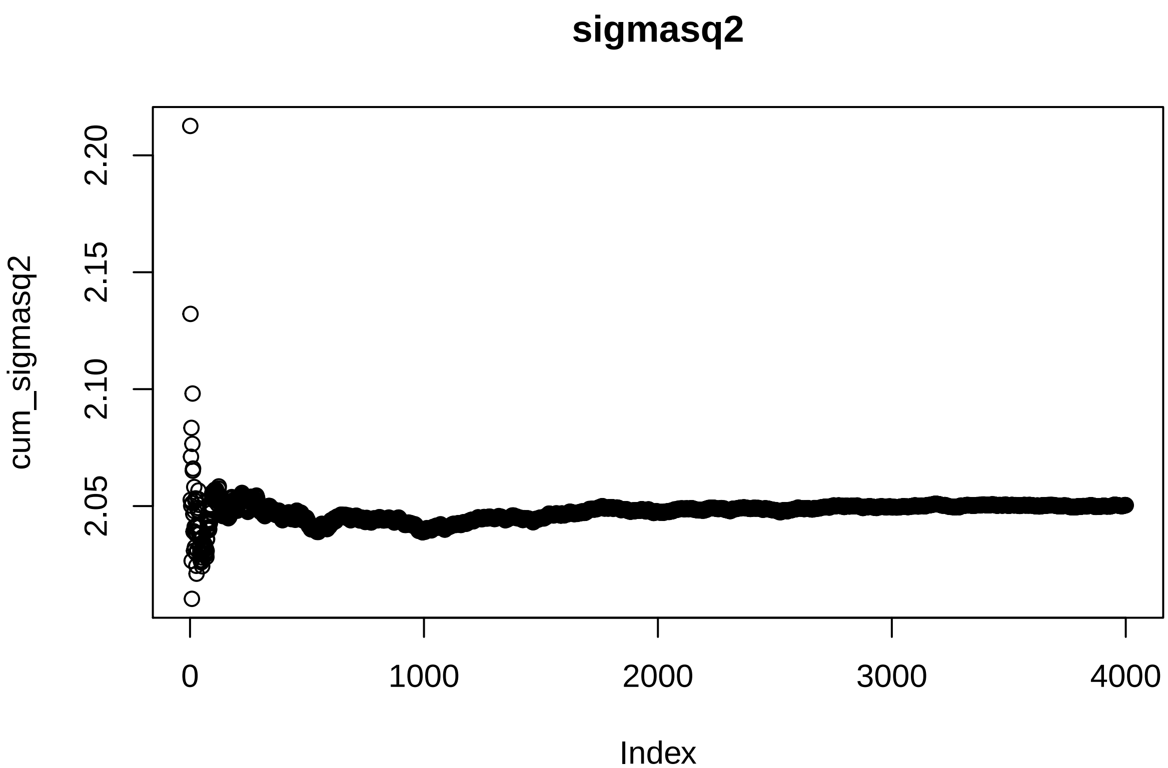


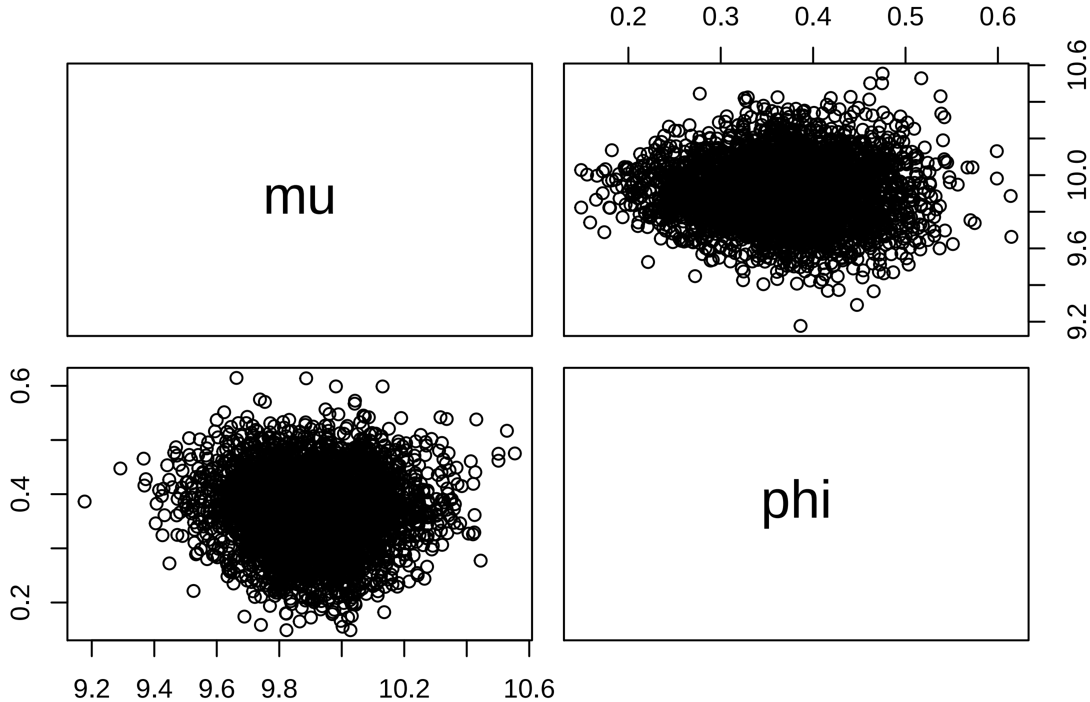


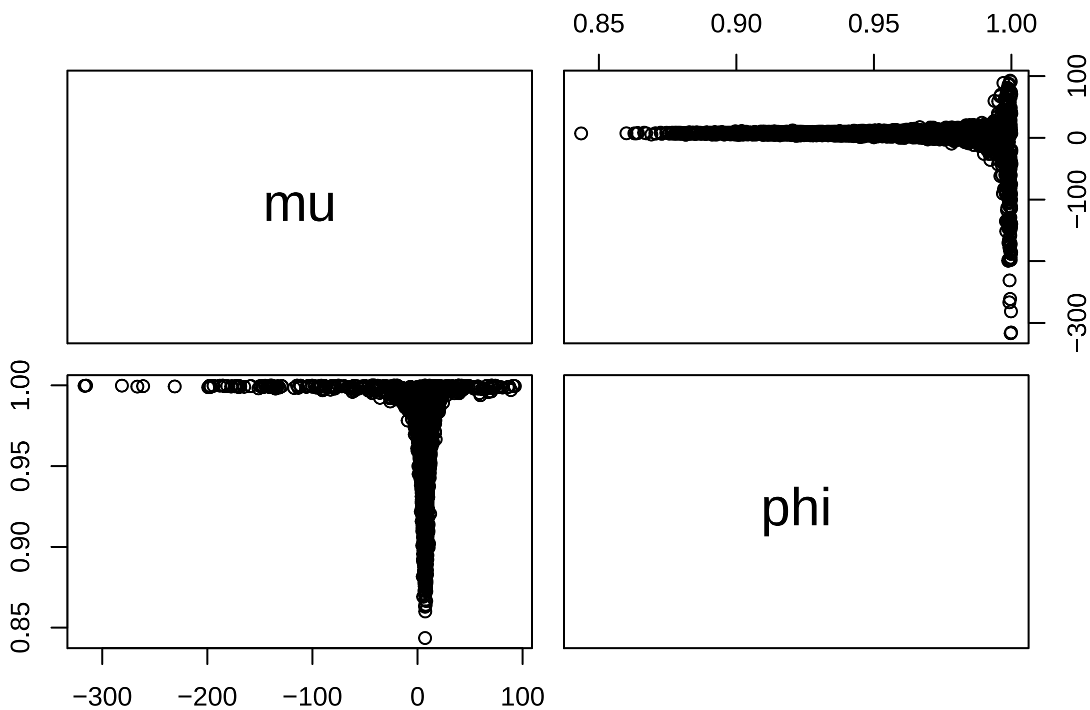

sigmasq1
cum_sigmasq1
1.85
1.90
1.95
2.00
2.05
2.10
0
1000
2000
3000
4000
Index


mu2
cum_mu2
−25
−15
−5
0
5
0
1000
2000
3000
4000
Index


phi2
cum_phi2
0.94
0.96
0.98
0
1000
2000
3000
4000
Index


sigmasq2
cum_sigmasq2
2.05
2.10
2.15
2.20
0
1000
2000
3000
4000
Index





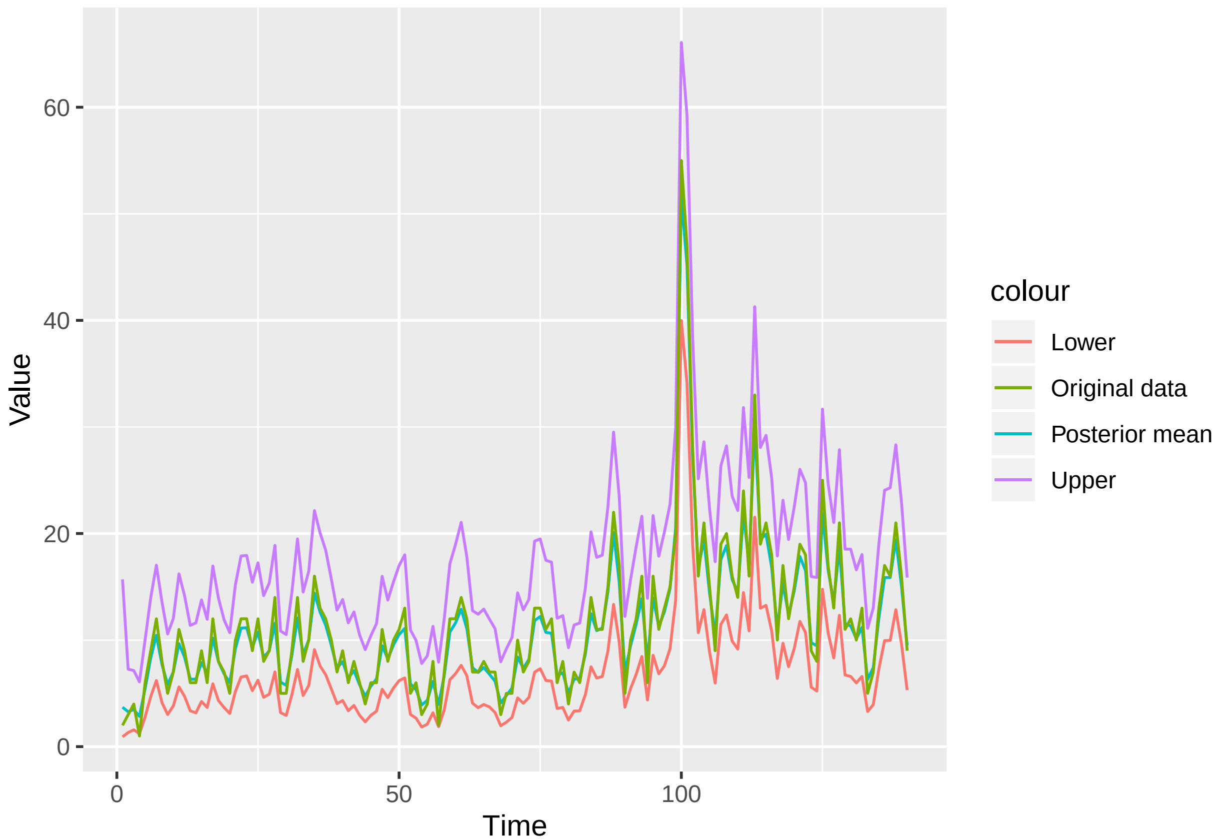
The joint posterior of μ and ϕ are concentrated around parameters' mean. The joint posterior distribution seems normally distributed, since we know the data is simulated using normal distribution, it is possible to conclude that the model constructed for sampling described the data well.

(c) The data campy.dat contain the number of cases of campylobacter infections in the north of the province Quebec (Canada) in four week intervals from January 1990 to the end of October 2000. It has 13 observations per year and 140 observations in total. Assume that the number of infections c_t at each time point follows an independent Poisson distribution when conditioned on a latent AR(1)-process x_t, that is

$$c_t|x_t \sim Poisson(exp(x_t))$$

where x_t is an AR(1)-process as in a). Implement and estimate the model in Stan, using suitable priors of your choice. Produce a plot that contains both the data and the posterior mean and 95% credible intervals for the latent intensity $\theta_t = exp(x_t)$ over time. [Hint: Should x_t be seen as data or parameters?]

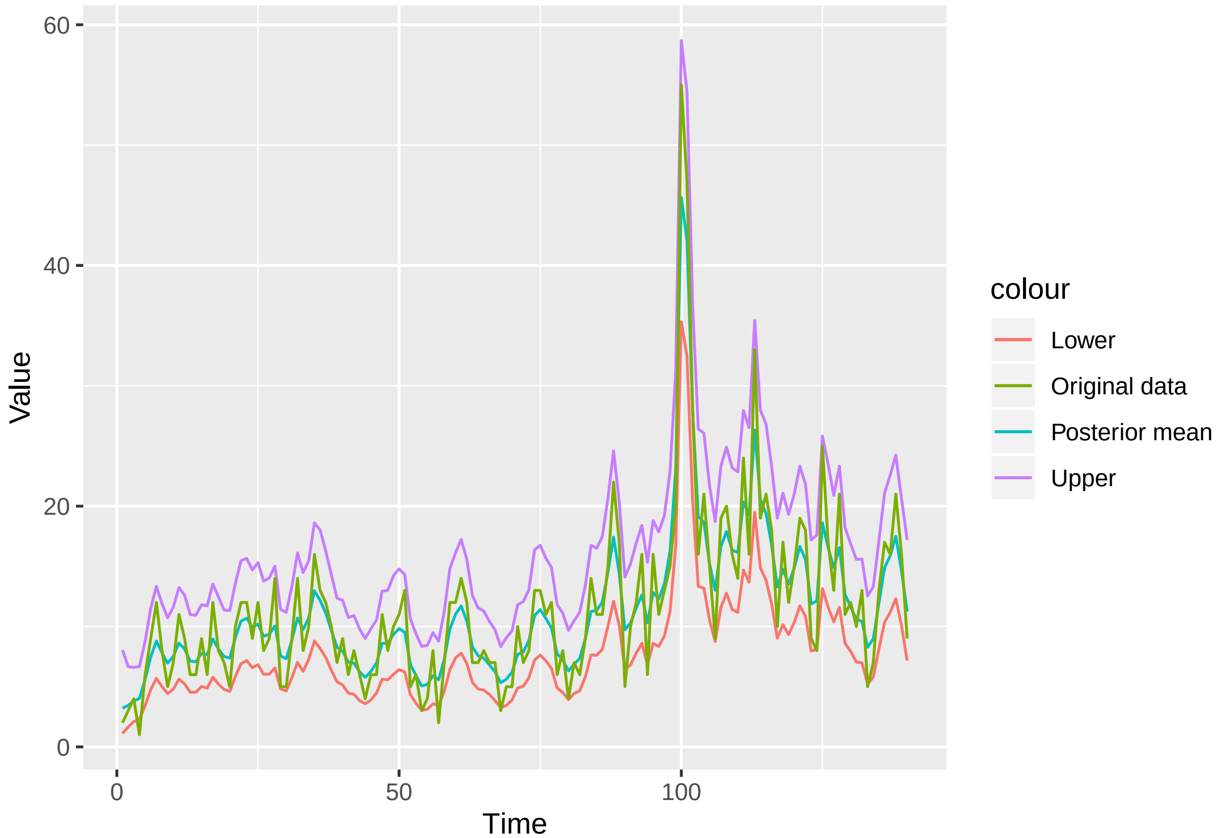
```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Frameworks/R.frame
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/3.6/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/3.6/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/include/Eigen/Core:96:10: fa
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```



In this case, x_t should be seen as parameter.

(d) Now, assume that we have a prior belief that the true underlying intensity θ_t varies more smoothly than the data suggests. Change the prior for σ^2 so that it becomes informative about that the AR(1)-process increments ϵ_t should be small. Re-estimate the model using Stan with the new prior and produce the same plot as in c). Has the posterior for θ_t changed?

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Frameworks/R.frame
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/include/Eigen/src/Core/util/
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/3.6/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/3.6/Resources/library/StanHeaders/incl
## In file included from /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/3.6/Resources/library/RcppEigen/include/Eigen/Core:96:10: fa
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1
```



Increased degrees of freedom in simulating sigmasq represents more informative prior for it. In this question we suppose nu = 20 on prior.Means of posterior thetas are more similar to actual data. 95% credible interval seems wider than before; but such credible interval covers almost all actual data points.

Appendix A : Code Question 1

```
ar1_process=function(t,mu,phi,sigmasq){
  res=vector(length=t)
  res[1]=mu
  for(i in 2:t){
    res[i]=mu+phi*(res[i-1]-mu)+rnorm(1,0,sqrt(sigmasq))
  }
  return(res)
}
mu=10
sigmasq=2
t=200

tmp2=ar1_process(t,mu,-1,sigmasq)
plot(x=seq(1,t),y=tmp2,type="l",xlab = "Time",ylab="Value",main = "phi=-1")
tmp3=ar1_process(t,mu,-0.5,sigmasq)
lines(tmp3)
plot(x=seq(1,t),y=tmp3,type="l",xlab = "Time",ylab="Value",main = "phi=-0.5")
tmp4=ar1_process(t,mu,0,sigmasq)
plot(x=seq(1,t),y=tmp4,type="l",xlab = "Time",ylab="Value",main = "phi=0")
tmp1=ar1_process(t,mu,0.5,sigmasq)
plot(x=seq(1,t),y=tmp1,type="l",xlab = "Time",ylab="Value",main = "phi=0.5")
tmp5=ar1_process(t,mu,1,sigmasq)
plot(x=seq(1,t),y=tmp5,type="l",xlab = "Time",ylab="Value",main = "phi=1")
x1t=ar1_process(200,10,0.3,2)
y1t=ar1_process(200,10,0.95,2)


StanModel = '
data {
  int<lower=0> N;
  vector[N] y;
}
parameters {
  real mu;
  real <lower=-1,upper=1>phi;
  real<lower=0> sigmasq;
}
model {
  mu~ normal(1,100);
  phi~ normal(1,100);
  sigmasq ~ scaled_inv_chi_square(1,2);
  for (n in 2:N)
    y[n] ~ normal(mu + phi * (y[n-1]-mu), sqrt(sigmasq));
}
'

data1 = list(N=length(x1t), y=x1t)
data2 = list(N=length(y1t), y=y1t)
burnin = 1000
niter = 2000
fit1 = stan(model_code=StanModel,data=data1, warmup=burnin,iter=niter,chains=4,refresh=0)
```

```r
fit2 = stan(model_code=StanModel,data=data2, warmup=burnin,iter=niter,chains=4,refresh=0)

df1=summary(fit1,probs = c(0.025, 0.975))$summary
print(df1)
df2=summary(fit2,probs = c(0.025, 0.975))$summary
print(df2)

# phi=0.3
mu1<- extract(fit1, 'mu')$mu
cum_mu1 = cumsum(mu1)
for (i in 1:length(cum_mu1)){
  cum_mu1[i] = cum_mu1[i]/i
}
plot(cum_mu1,main="mu1")

phi1 <- extract(fit1, 'phi')$phi
cum_phi1 = cumsum(phi1)
for (i in 1:length(cum_phi1)){
  cum_phi1[i] = cum_phi1[i]/i
}
plot(cum_phi1,main="phi1")

sigmasq1 <- extract(fit1, 'sigmasq')$sigmasq
cum_sigmasq1 = cumsum(sigmasq1)
for (i in 1:length(cum_sigmasq1)){
  cum_sigmasq1[i] = cum_sigmasq1[i]/i
}
plot(cum_sigmasq1,main="sigmasq1")

# phi=0.95
mu2<- extract(fit2, 'mu')$mu
cum_mu2 = cumsum(mu2)
for (i in 1:length(cum_mu2)){
  cum_mu2[i] = cum_mu2[i]/i
}
plot(cum_mu2,main="mu2")

phi2 <- extract(fit2, 'phi')$phi
cum_phi2 = cumsum(phi2)
for (i in 1:length(cum_phi2)){
  cum_phi2[i] = cum_phi2[i]/i
}
plot(cum_phi2,main="phi2")

sigmasq2 <- extract(fit2, 'sigmasq')$sigmasq
cum_sigmasq2= cumsum(sigmasq2)
for (i in 1:length(cum_sigmasq2)){
  cum_sigmasq2[i] = cum_sigmasq2[i]/i
}
plot(cum_sigmasq2,main="sigmasq2")

pairs(extract(fit1, c('mu','phi')))
pairs(extract(fit2, c('mu','phi')))
```

```
data=read.table("campy.dat",header = TRUE)
data = list(N=length(data$c), ct=data$c)
StanModel ='
data {
  int<lower=0> N;
  int ct[N];
}
parameters {
  real mu;
  real <lower=-1,upper=1>phi;
  real<lower=0> sigma;
  real xt[N];
}
model {
  mu~ normal(1,100);
  phi~ normal(1,100);
  sigma ~ scaled_inv_chi_square(20,2);
  xt[1]~normal(mu,1);
  ct[1]~poisson(exp(mu));
  for (n in 2:N){
    xt[n]~normal(mu + phi * (xt[n-1]-mu), sigma);
    ct[n] ~ poisson(exp(xt[n]));
  }
}
'

burnin = 1000
niter = 2000
fit1 = stan(model_code=StanModel,data=data, warmup=burnin,iter=niter,chains=4,refresh=0)

df=as.data.frame(summary(fit1,probs = c(0.025, 0.975))$summary)
df=df[c(-1,-2,-3,-144),]
df$mean=exp(df$mean)
df$`2.5%`=exp(df$`2.5%`)
df$`97.5%`=exp(df$`97.5%`)
ggplot()+geom_line(aes(x=c(1:length(df$mean)),y=df$mean,colour="Posterior mean"))+geom_line(aes(x=c(1:le
data=read.table("campy.dat",header = TRUE)
data = list(N=length(data$c), ct=data$c)
StanModel ='
data {
  int<lower=0> N;
  int ct[N];
}
parameters {
  real mu;
  real <lower=-1,upper=1>phi;
  real<lower=0> sigma;
  real xt[N];
}
model {
  mu~ normal(1,100);
  phi~ normal(1,100);
  sigma ~ scaled_inv_chi_square(1,2);
```

```
  xt[1]~normal(mu,1);
  ct[1]~poisson(exp(mu));
  for (n in 2:N){
    xt[n]~normal(mu + phi * (xt[n-1]-mu), sigma);
    ct[n] ~ poisson(exp(xt[n]));
  }
}
'

burnin = 1000
niter = 2000
fit1 = stan(model_code=StanModel,data=data, warmup=burnin,iter=niter,chains=4,refresh=0)

df=as.data.frame(summary(fit1,probs = c(0.025, 0.975))$summary)
df=df[c(-1,-2,-3,-144),]
df$mean=exp(df$mean)
df$`2.5%`=exp(df$`2.5%`)
df$`97.5%`=exp(df$`97.5%`)
ggplot()+geom_line(aes(x=c(1:length(df$mean)),y=df$mean,colour="Posterior mean"))+geom_line(aes(x=c(1:le
```