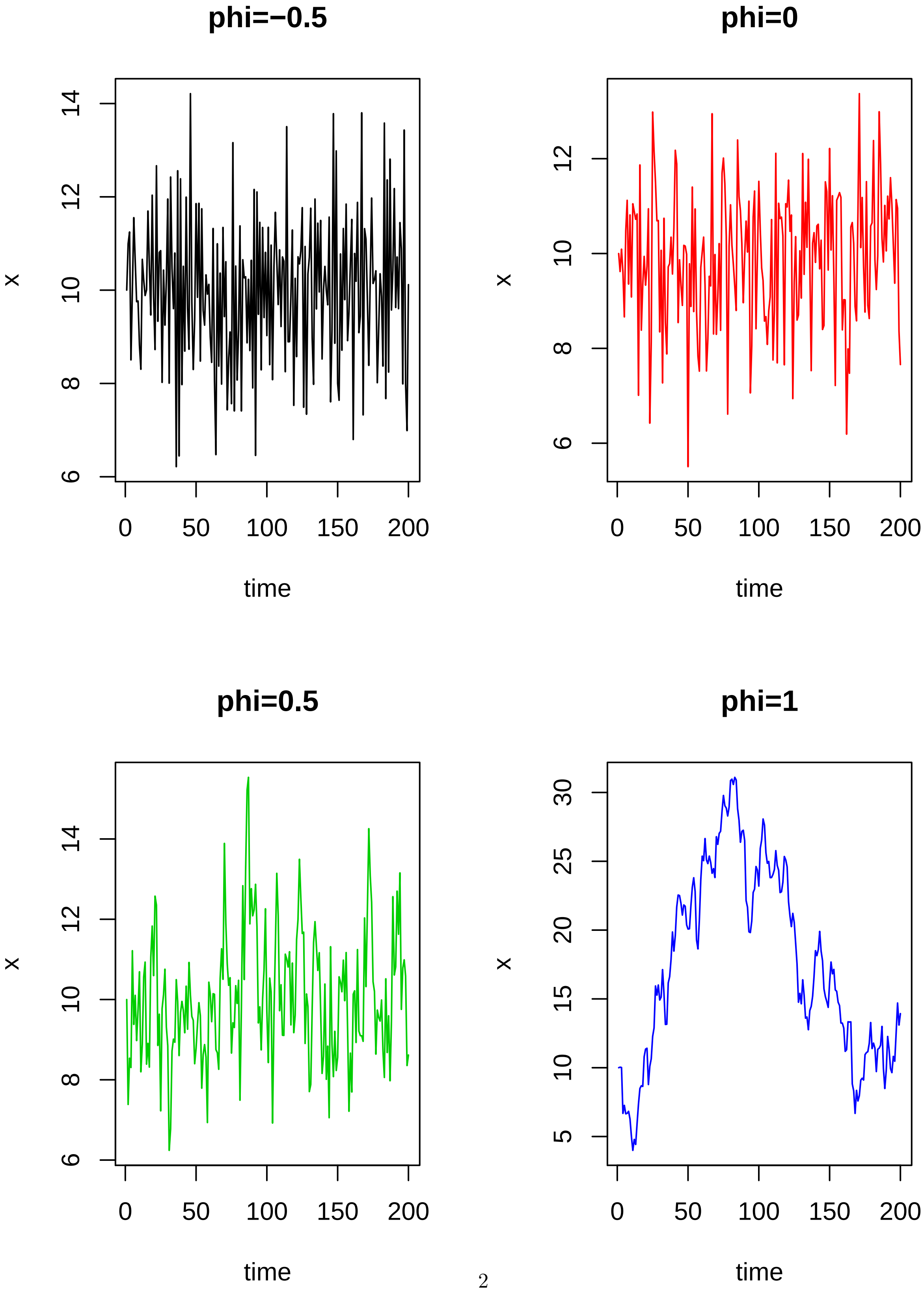
732A91 Bayesian Learning- Computer Lab 4

Namita Sharma, Aman Kumar Nayak

5/26/2020

1. Time series models in Stan

(a) Simulate data from AR process



phi=−0.5

phi=0

phi=0.5

phi=1

We see from the realizations of x for different values of ϕ that the samples are less correlated for low values of $\phi < 0$ i.e. there is quite a bit of variation between consecutively generated samples. As ϕ increases, we see that the samples get more and more correlated. The new sample value is very close to its previous one.

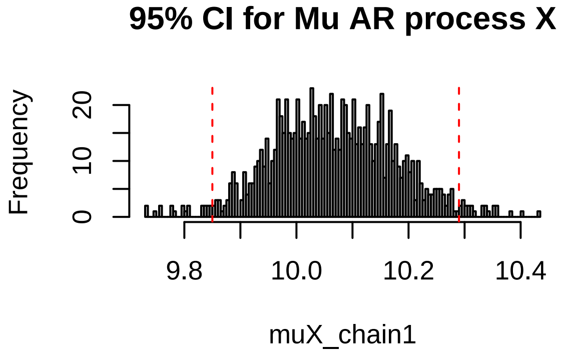
(b) Estimate parameters μ, ϕ and σ^2 using MCMC in stan

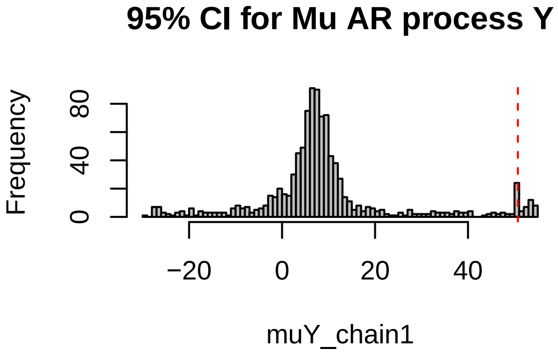
```
## [1] "Posterior mean, 95% credible interval for mu, phi and sigma2 for AR process x"
```

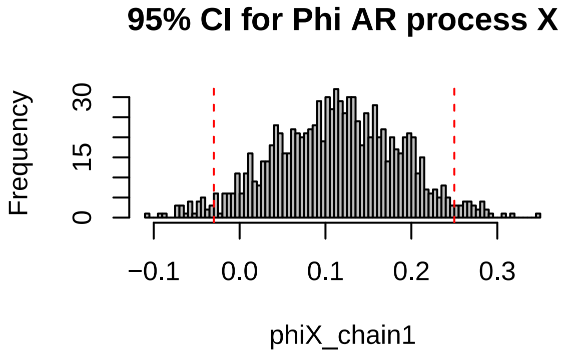
	mean	se_mean	sd	2.5%	97.5%	n_eff	Rhat
mu	10.0679104	0.0016407	0.1119081	9.8453750	10.2946663	4652.352	0.9995124
phi	0.1132213	0.0011213	0.0713348	-0.0258972	0.2525193	4047.405	0.9997709
sigma2	1.9002905	0.0032028	0.1944445	1.5513736	2.3192681	3685.763	0.9994069

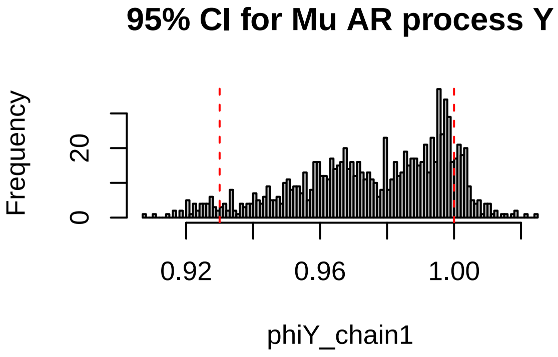
```
## [1] "Posterior mean, 95% credible interval for mu, phi and sigma2 for AR process y"
```

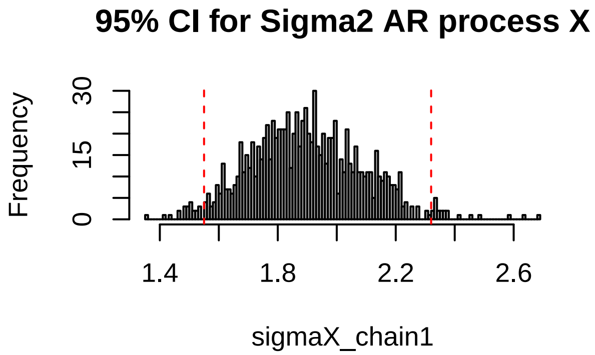
	mean	se_mean	sd	2.5%	97.5%	n_eff	Rhat
mu	6.0891851	1.6239616	18.9921566	-48.9281003	50.719497	136.7720	1.026476
phi	0.9754284	0.0009793	0.0224652	0.9271426	1.004687	526.2515	1.003716
sigma2	1.9994453	0.0061820	0.2058902	1.6351254	2.436736	1109.2220	1.001537

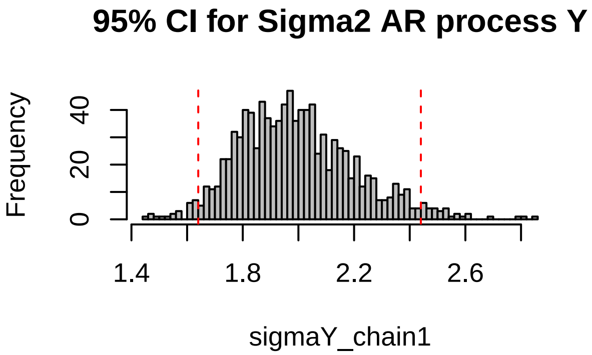

95% CI for Mu AR process X


95% CI for Mu AR process Y


95% CI for Phi AR process X


95% CI for Mu AR process Y

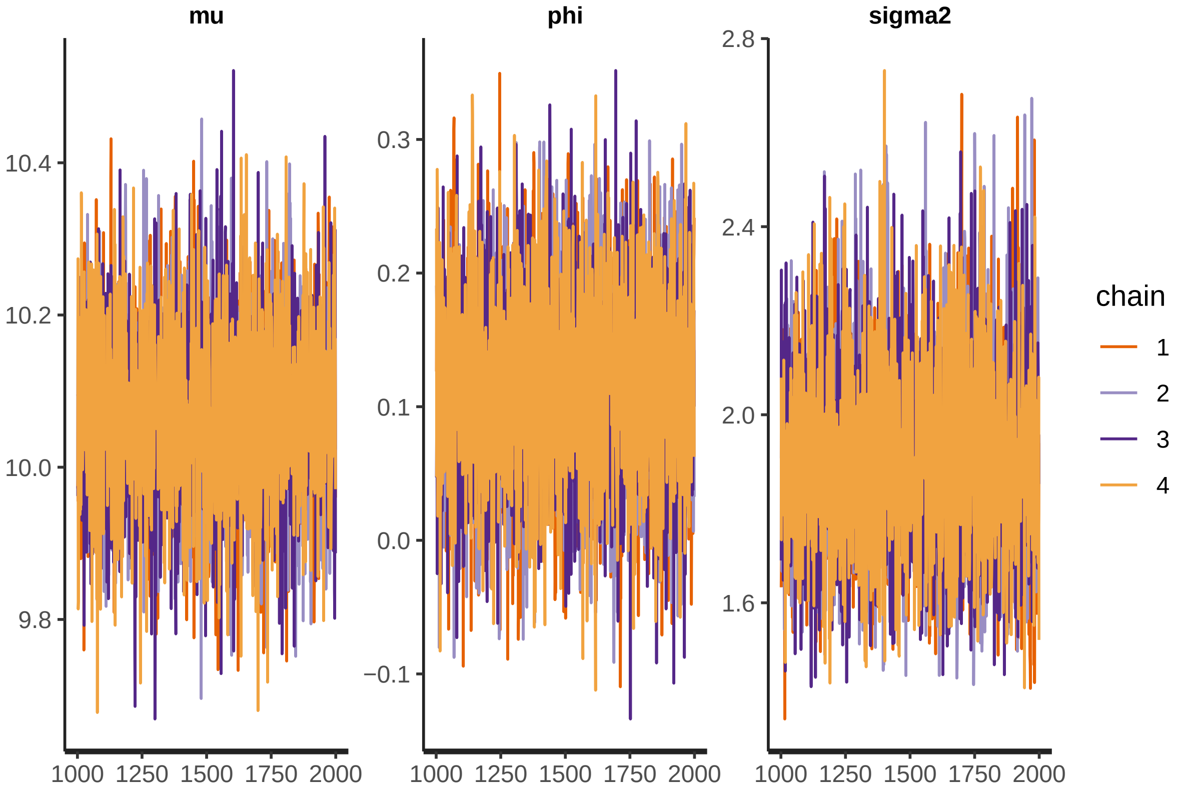


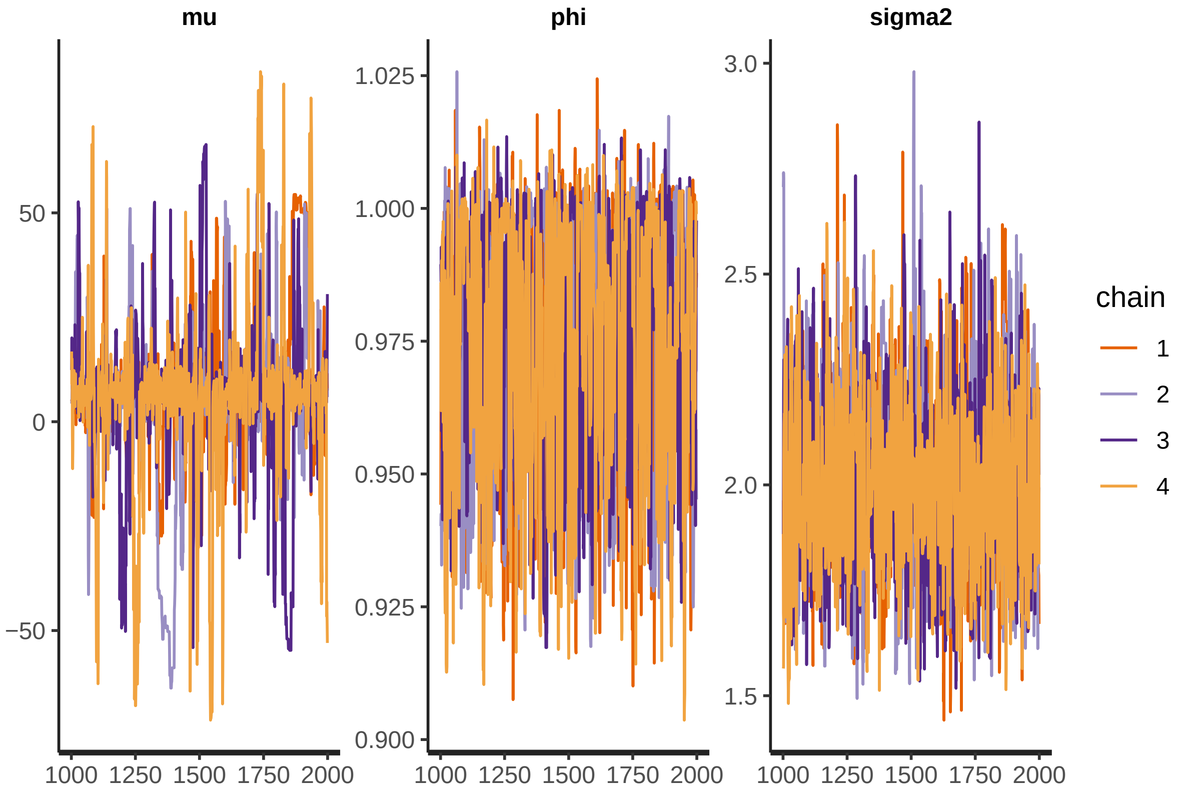


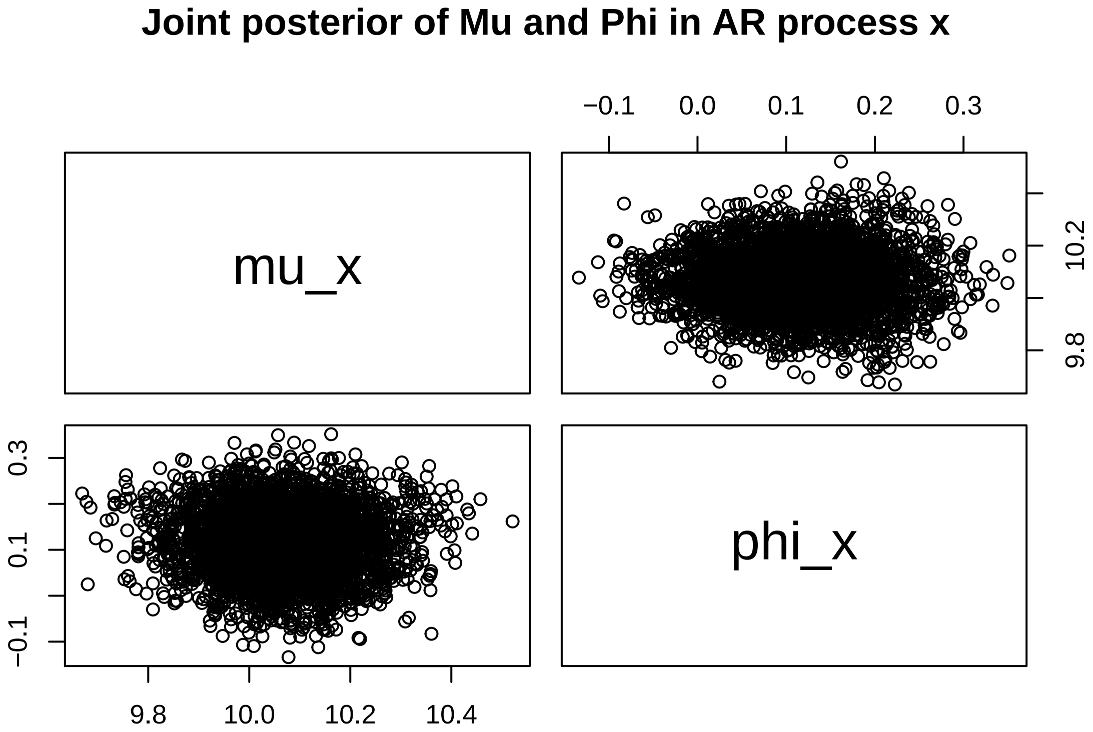
For AR process x, the estimations of parameters μ, ϕ and σ^2 are quite close to the actual values. The 95% credible intervals are narrow indicating less uncertainity in the estimated parameter values. The effective sample size is large which means that the efficiency of MCMC sampling was high.

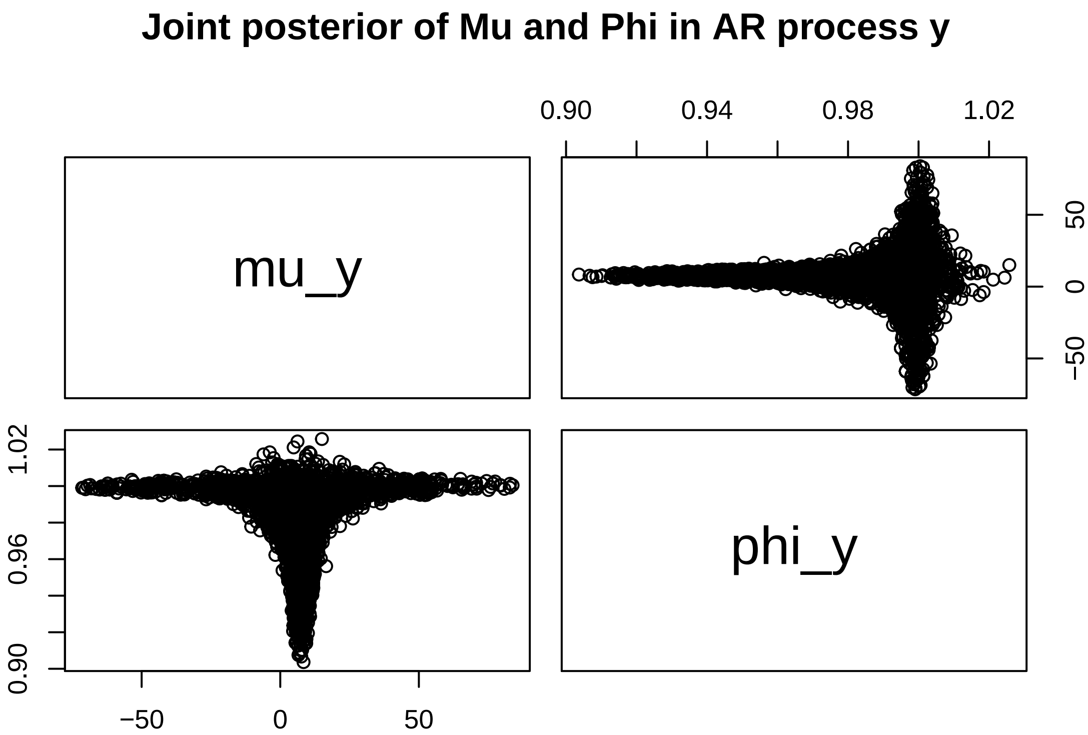
However, for AR process y, the estimation of the mean parameter μ is poor while ϕ and σ^2 estimations seem okay. This is also seen in the 95% credible intervals for the parameters. The interval for μ is very large indicating high uncertainity in the estimated value of mean whereas the other two parameters are tightly bound in their intervals. The effective sample size is also quite small which tells us that the MCMC sampling was not very efficient (due to high auto correlation).

Joint posterior of μ, ϕ





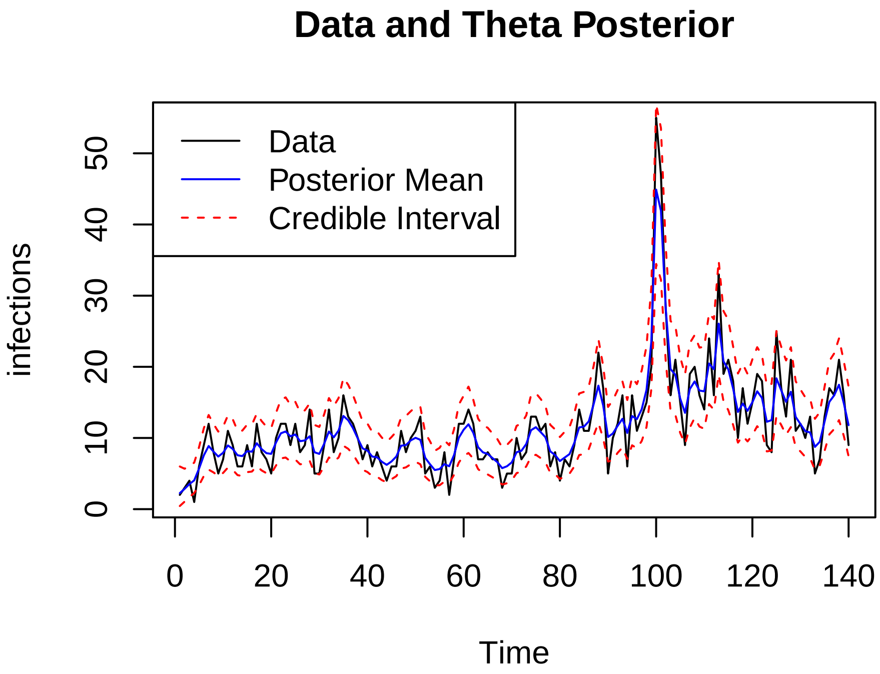

Joint posterior of Mu and Phi in AR process x
mu_x
phi_x
−0.1
0.0
0.1
0.2
0.3
10.2
9.8
0.3
0.1
−0.1
9.8
10.0
10.2
10.4



From the traceplots of parameters μ, ϕ and σ^2, we can see that the convergence was attained in case of AR process x for all parameters. And in case of AR process y, convergence was obtained for parameters ϕ and σ^2 but not μ.

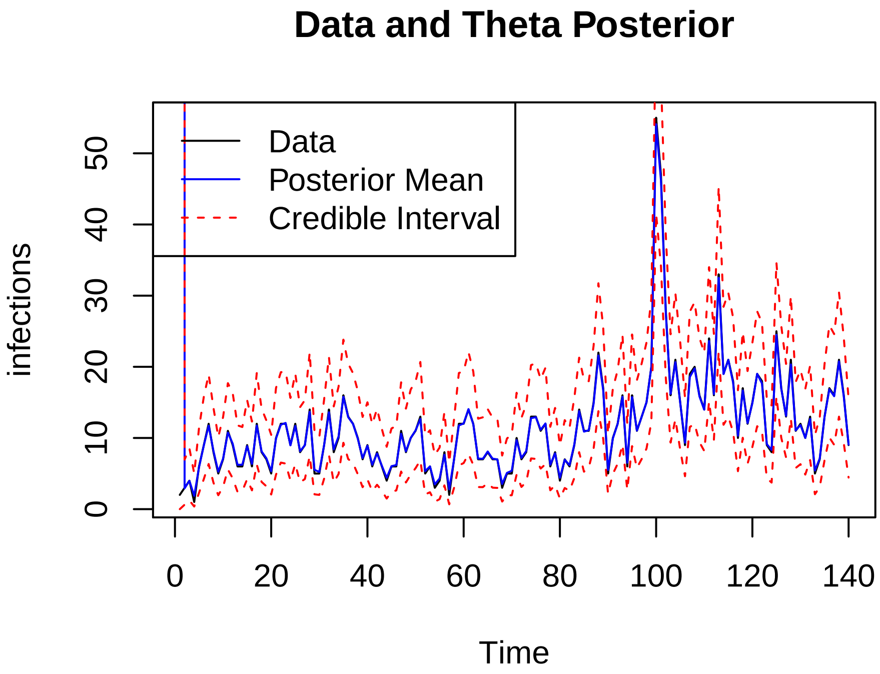
In the joint posterior plot of μ and ϕ for AR process y, it can be seen that the uncertainity along the μ axis is quite high which makes the joint distribution skewed. The joint posterior plot for AR process x shows less uncertainity for both parameters and hence a better convergence.

(c) Estimate Poisson model using Stan



We see from the plot that the posterior mean of θ follows the data quite closely. And the credible interval tightly bounds the posterior mean and the data. Hence, the uncertainity in the model is less and the mean coincides with the data pretty well except at certain peaks and valleys.

(d) Estimate Poisson model with informative prior using Stan



With an informed prior, we see that the posterior mean of θ exactly overlaps with the data distribution at all the peaks and valleys. And the credible interval tightly bounds the posterior mean. The uncertainity in the model is even lesser when compared to the uninformed prior.

Appendix

```
##################################################################
# 1. Time series models in Stan
##################################################################
library(rstan)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())

#--------------------------------------------
# (a) Simulate data from AR process
simulate_AR <- function(mu, phi, sigma2, T) {
  x    <- numeric(T)
  x[1] <- mu

  for (t in 2:T) {
    epsilon_t <- rnorm(n=1, mean=0, sd=sqrt(sigma2))
    x[t]      <- mu + phi*(x[t-1]-mu) + epsilon_t
  }
  return(x)
}
```

```
# Simulate from AR process for different values of phi
PHI = seq(-0.5, 1, length=4)
i = 1
par(mfrow = c(1,2))
for (phi in PHI) {
  # Simulate from AR process
  x <- simulate_AR(mu=10, phi=phi, sigma2=2, T=200)

  # Plot the realizations
  plot(1:200, x, type="l", col=i, xlab="time", ylab="x", main=paste0("phi=", phi))
  i <- i+1
}

#-------------------------------------------
# (b) Estimate parameters using MCMC
x <- simulate_AR(mu=10, phi=0.3, sigma2=2, T=200)
y <- simulate_AR(mu=10, phi=0.95, sigma2=2, T=200)

# Stan code to sample from posterior mu, phi and sigma2
stanNormal = '
data {
  int<lower=0> T;
  real x[T];
}
parameters {
  real mu;
  real phi;
  real<lower=0> sigma2;
}
model {
  mu ~ normal(0, 100);
  sigma2 ~ scaled_inv_chi_square(0.01, 2);
  phi ~ normal(0, 1);

  for (t in 2:T)
    x[t] ~ normal(mu + phi*(x[t-1]-mu), sqrt(sigma2));
}'

# Data (synthetic)
N <- length(x)
dataX <- list(T=N, x=x)
dataY <- list(T=N, x=y)

# Fit stan model
burnin <- 1000
niter  <- 2000

# For AR(1)-process 1
fitX <- stan(model_code=stanNormal, data=dataX, warmup=1000, iter=2000)
# Posterior of mu, phi and sigma2
postDrawsX <- extract(fitX)

# For AR(1)-process 2
```

```
fitY <- stan(model_code=stanNormal, data=dataY, warmup=1000, iter=2000)
# Posterior of mu, phi and sigma2
postDrawsY <- extract(fitY)

# Summary Stats - Posterior mean, 95% credible interval for mu, phi and sigma2
paste("Posterior mean, 95% credible interval for mu, phi and sigma2 for AR process x")
summaryX = summary(fitX,
                   pars=c("mu", "phi", "sigma2"),
                   probs = c(0.025, 0.975))$summary
knitr::kable(summaryX)

paste("Posterior mean, 95% credible interval for mu, phi and sigma2 for AR process y")
summaryY = summary(fitY,
                   pars=c("mu", "phi", "sigma2"),
                   probs = c(0.025, 0.975))$summary
knitr::kable(summaryY)

# 95% credible interval for mu, phi and sigma2
quanX = round(summaryX[, c(4,5)], 2)
quanY = round(summaryY[, c(4,5)], 2)

# Plot of first chain of mu, phi and sigma2 with 95% credible interval
par(mfrow = c(2,2))

muX_chain1 = postDrawsX$mu[1:(niter-burnin)]
hist(muX_chain1, breaks=100, col="gray", main="95% CI for Mu AR process X")
abline(v=quanX[1,1], lty="dashed", col="red")
abline(v=quanX[1,2], lty="dashed", col="red")

muY_chain1 = postDrawsY$mu[1:(niter-burnin)]
hist(muY_chain1, breaks=100, col="gray", main="95% CI for Mu AR process Y")
abline(v=quanY[1,1], lty="dashed", col="red")
abline(v=quanY[1,2], lty="dashed", col="red")

phiX_chain1 = postDrawsX$phi[1:(niter-burnin)]
hist(phiX_chain1, breaks=100, col="gray", main="95% CI for Phi AR process X")
abline(v=quanX[2,1], lty="dashed", col="red")
abline(v=quanX[2,2], lty="dashed", col="red")

phiY_chain1 = postDrawsY$phi[1:(niter-burnin)]
hist(phiY_chain1, breaks=100, col="gray", main="95% CI for Mu AR process Y")
abline(v=quanY[2,1], lty="dashed", col="red")
abline(v=quanY[2,2], lty="dashed", col="red")

sigmaX_chain1 = postDrawsX$sigma[1:(niter-burnin)]
hist(sigmaX_chain1, breaks=100, col="gray", main="95% CI for Sigma2 AR process X")
abline(v=quanX[3,1], lty="dashed", col="red")
abline(v=quanX[3,2], lty="dashed", col="red")

sigmaY_chain1 = postDrawsY$sigma[1:(niter-burnin)]
hist(sigmaY_chain1, breaks=100, col="gray", main="95% CI for Sigma2 AR process Y")
abline(v=quanY[3,1], lty="dashed", col="red")
abline(v=quanY[3,2], lty="dashed", col="red")
```

```r
# Convergence of the samplers- Trace plot of mu, phi and sigma2
traceplot(fitX)
traceplot(fitY)

# Joint posterior of mu and phi
jointPostX = cbind(mu_x=postDrawsX$mu, phi_x=postDrawsX$phi)
jointPostY = cbind(mu_y=postDrawsY$mu, phi_y=postDrawsY$phi)

# Plot joint posterior
pairs(jointPostX, main="Joint posterior of Mu and Phi in AR process x")
pairs(jointPostY, main="Joint posterior of Mu and Phi in AR process y")

#--------------------------------------------
# (c) Estimate Poisson model using Stan
campy <- read.table(
  file="C:/Users/namit/Downloads/Bayesian Learning/R files/Lab4/campy.dat",
  header=TRUE)

# Stan code to sample from posterior of x[t]
stanPoisson = '
data {
  int<lower=0> T;
  int c[T];
}
parameters {
  real mu;
  real phi;
  real<lower=0> sigma2;
  real x[T];
}
model {
  mu ~ normal(0, 100);
  sigma2 ~ scaled_inv_chi_square(0.01, 2);
  phi ~ normal(0, 1);

  for (t in 2:T){
    x[t] ~ normal(mu + phi*(x[t-1]-mu), sqrt(sigma2));
    c[t] ~ poisson(exp(x[t]));
  }
}'

# Data (campy file)
N <- length(campy$c)
data <- list(T=N, c=campy$c)

# Fit stan model to poisson process
burnin <- 1000
niter  <- 2000
fit    <- stan(model_code=stanPoisson, data=data, warmup=1000, iter=2000)

# Posteriors of mu, phi, sigma2, x
postDraws <- extract(fit)
```

```r
# Posterior of Latent intensity theta
thetaPost <- exp(postDraws$x)

# Mean of posterior of theta
thetaPostMean = apply(thetaPost, 2, mean)

# 95% credible interval for theta
credI = apply(thetaPost, 2, quantile, probs=c(0.025, 0.975))

# Plot data and posterior of theta
plot(campy$c, type="l", main="Data and Theta Posterior",
     xlab="Time", ylab="infections")
lines(thetaPostMean, col="blue")
lines(credI[1, ], lty="dashed", col="red")
lines(credI[2, ], lty="dashed", col="red")
legend("topleft", c("Data", "Posterior Mean", "Credible Interval"),
       col=c("black", "blue", "red"),
       lty=c("solid", "solid", "dashed"))

#-------------------------------------------
# (d) Estimate Poisson model with informative prior using Stan

# Stan code to sample from posterior of x[t]
stanPoisson = '
data {
  int<lower=0> T;
  int c[T];
}
parameters {
  real mu;
  real phi;
  real<lower=0> sigma2;
  real x[T];
}
model {
  mu ~ normal(0, 100);
  sigma2 ~ scaled_inv_chi_square(2, 10);
  phi ~ normal(0, 1);

  for (t in 2:T){
    x[t] ~ normal(mu + phi*(x[t-1]-mu), sqrt(sigma2));
    c[t] ~ poisson(exp(x[t]));
  }
}'

# Data (campy file)
N <- length(campy$c)
data <- list(T=N, c=campy$c)

# Fit stan model to poisson process
burnin <- 1000
niter  <- 2000
fit    <- stan(model_code=stanPoisson, data=data, warmup=1000, iter=2000)
```

```r
# Posteriors of mu, phi, sigma2, x
postDraws <- extract(fit)

# Posterior of Latent intensity theta
thetaPost <- exp(postDraws$x)

# Mean of posterior of theta
thetaPostMean = apply(thetaPost, 2, mean)

# 95% credible interval for theta
credI <- apply(thetaPost, 2, quantile, probs=c(0.025, 0.975))

# Plot data and posterior of theta
plot(campy$c, type="l", main="Data and Theta Posterior",
     xlab="Time", ylab="infections")
lines(thetaPostMean, col="blue")
lines(credI[1, ], lty="dashed", col="red")
lines(credI[2, ], lty="dashed", col="red")
legend("topleft", c("Data", "Posterior Mean", "Credible Interval"),
       col=c("black", "blue", "red"),
       lty=c("solid", "solid", "dashed"))
```