732A73: Bayesian Learning

Computer Lab 4

Oriol Garrobé, Dávid Hrabovszki

17 May 2020

Question 1. Time series models in Stan

(a)

We are asked to develop a simulator from the AR(1)-process,

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

for the given values: $\mu = 10$, $\sigma^2 = 2$ and $\phi \in (-1, 1)$. The code of the simulater can be seen in the Appendix.

From this point we are asked to generate 200 values of x_t. In Figure 1 it can be seen one simulation for $\phi = 1$. The parameter ϕ determines how the current value of x_t depends on the previous value. If $\phi > 0$ then they are positively correlated, if $\phi < 0$ they are negatively correlated and if $\phi = 0$ they are not correlated at all.

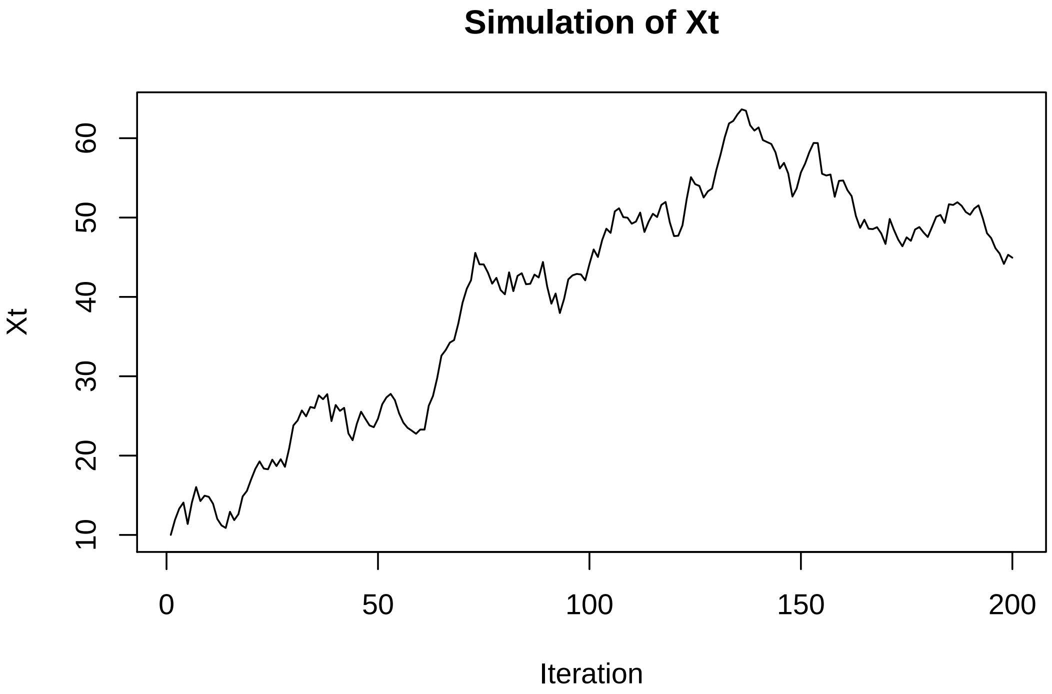

Figure 1: AR(1)-process simulation

(b)

Now we use the simulator from a) to simulate two AR(1)-processes with $\phi = 0.3$ and $\phi = 0.95$. Using this two simulations as synthetic data, we estimate μ, ϕ and σ^2 using MCMC. The priors used for the parameters are: * $\mu \sim N(10, 10)$. This is a non-informative prior as it has a large variance. * $phi \sim N(0, \sqrt{10})$. $\phi \in (-1, 1)$, therefore this is a weakly informative prior. * $\sigma^2 \sim Scale - inv - \chi^2(1, 1.4)$. Degrees of freedom are very small, therefore is non-informative.

The implemented Stan-code can be found in the appendix.

i.

For each of the simulated AR(1)-processes we get the following values for the three inferred parameters.

Parameter	Mean	95% CI	Effective Samples
mu	6.6	$(-56.12, 69.32)$	687
phi	0.97	$(0.9308, 1.0092)$	83
sigma2	1.29	$(1.1724, 1.4076)$	384

Table 1: *Posterior values for y_t.*

Parameter	Mean	95% CI	Effective Samples
mu	10.25	$(9.91, 10.60)$	3159
phi	0.39	$(0.26, 0.44)$	3847
sigma	1.49	$(1.35, 1.65)$	3365

Table 2: *Posterior values for x_t.*

We are able to estimate the true values for the synthetic data x_t but not for y_t. conclusion. WHY?

ii.

In Figure 2 it can be seen that there is bad mixin in the chain that uses y_t as data, the chains do not oscillate too much and they don't cover the same areas.

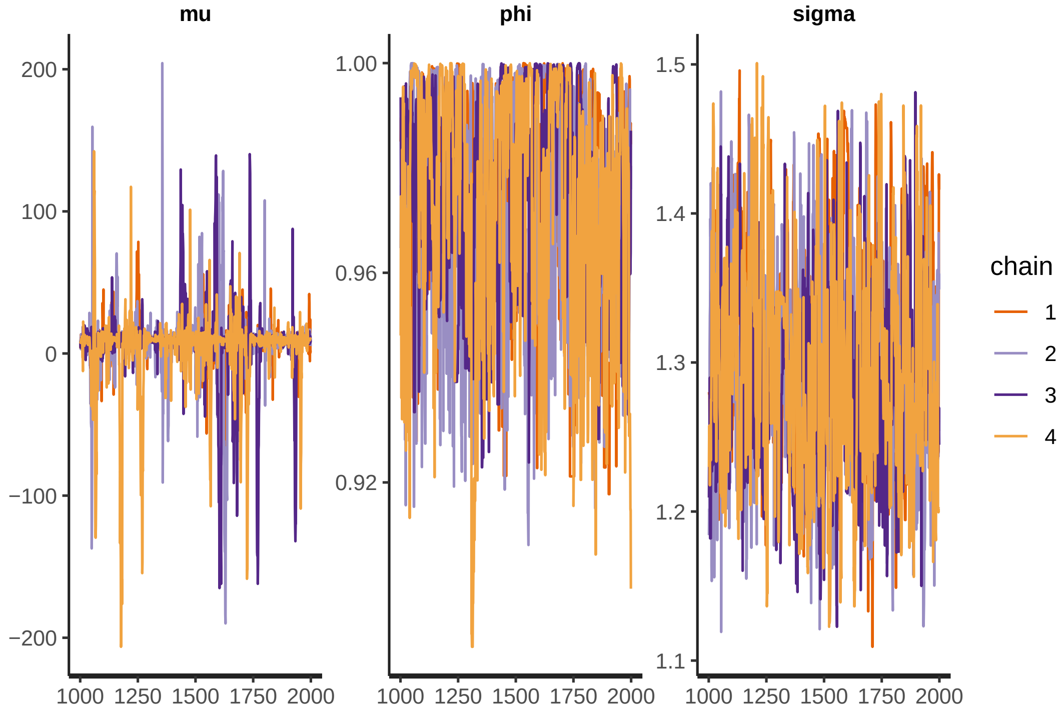

Figure 2: Traceplots of parameters for y_t.

On the other hand Figure 3 confirms that in x_t there is good convergence, because the chains oscillate a lot, and they cover the same areas, therefore they arrived at the same conclusion.

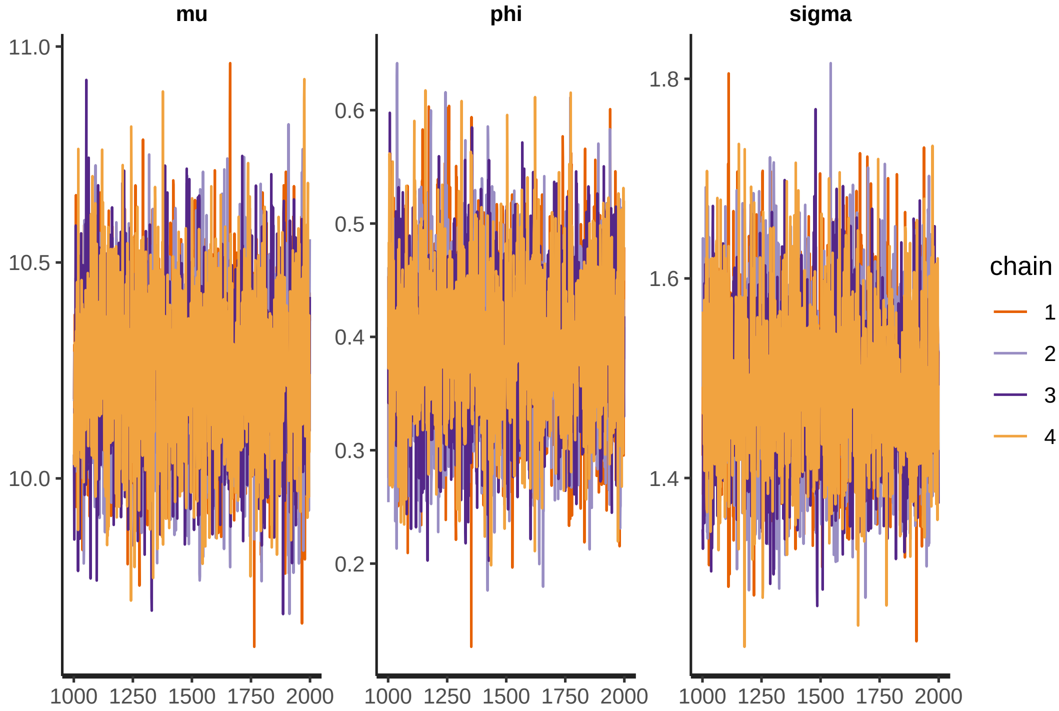

Figure 3: Traceplots of parameters for x_t.

(c)

In this question we use the dataset `campy.dat` that contains the number of cases of campylobacter in Quebec in four weeks intervals from January 1990 to October 2000. We assume that the number of infections c_t at each point follows and independent Poisson distribution when conditioned on a latent AR(1)-process x_t, such as

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

We implement and estimate the model using suitable priors. As we know nothing about the parameters, the priors chosen are the following:

- $\mu \sim N(10, 10)$. This is a non-informative prior as it has a large variance.
- $phi \sim N(0, \sqrt{10})$. $\phi \in (-1, 1)$, therefore this is a weakly informative prior.
- $\sigma^2 \sim Scale - inv - \chi^2(1, 2)$. Degrees of freedom are very small, therefore is non-informative.

The R-Stan code can be found in the Appendix.

In Figure 4 it can be seen the posterior mean and the sample mean along with the Latent intensity $\theta_t = exp(x_t)$ 95% Credible Intervals parameters over time.

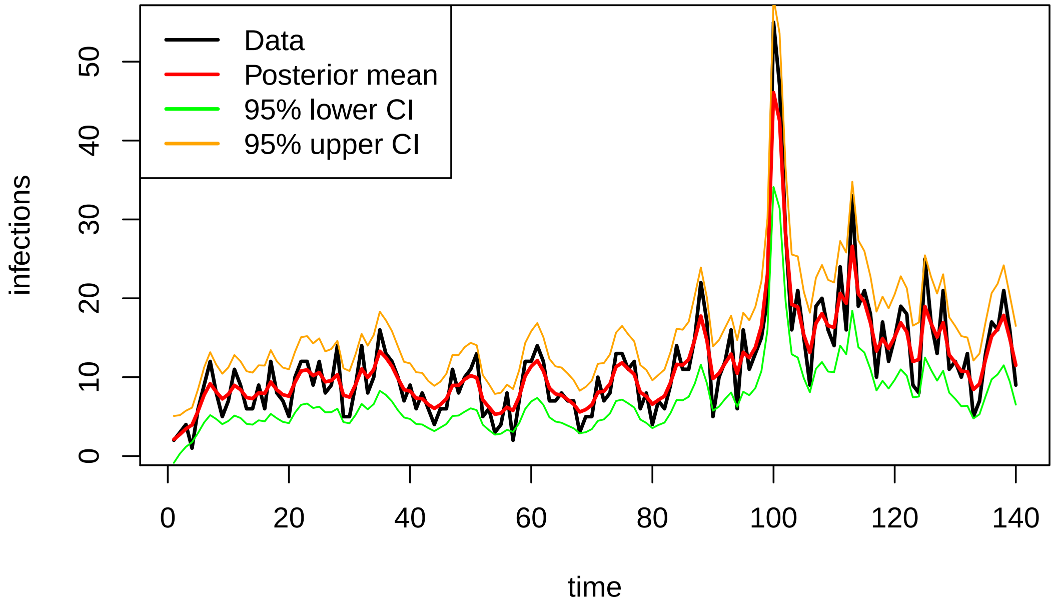

Figure 4: Posterior parameters over time.

(d)

Finally, having the prior belief that the true underlying intensity θ_t varies more smoothly than the data suggests, we change the prior for $sigma^2$ so it is more informative. We re-estimate the model with the new prior. The priors for the parameters are:

- $\mu \sim N(10, 10)$. This is a non-informative prior as it has a large variance.
- $phi \sim N(0, \sqrt{10})$. $\phi \in (-1, 1)$, therefore this is a weakly informative prior.
- $\sigma^2 \sim Scale-inv-\chi^2(100, 1)$. Due to the large value of degrees of freedom, this prior is very informative. It provides us with the information that the error is small.

The R-stan code can be found in the Appendix.

In Figure 5 it can be seen the posterior mean and the sample mean along with the Latent intensity $\theta_t = exp(x_t)$ 95% Credible Intervals parameters over time.

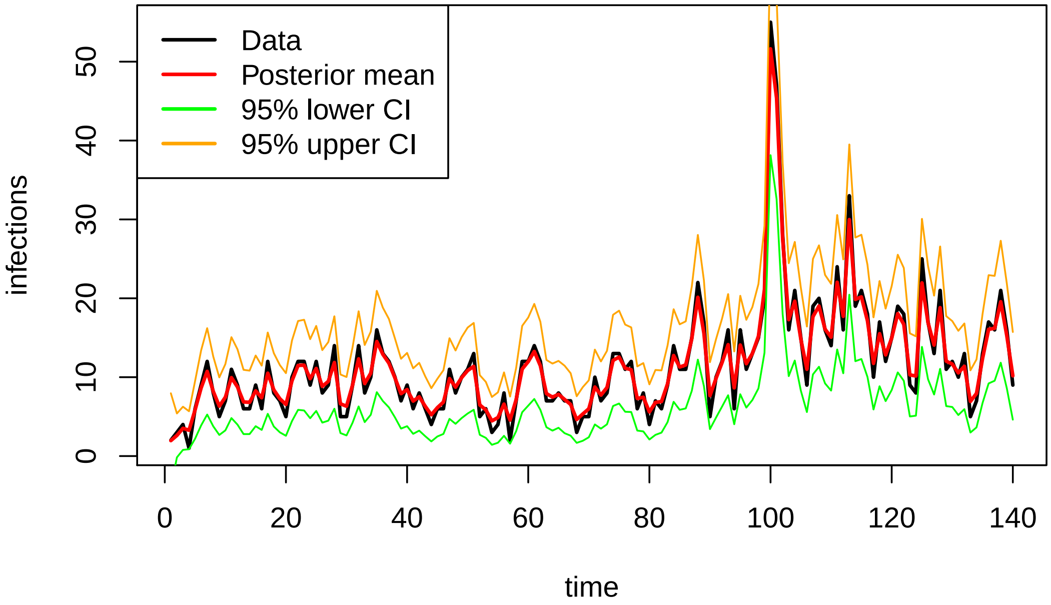

Figure 5: Posterior parameters over time.

Changing the prior for σ^2 changes the posterior in the following way. As the error is smaller the posterior mean is closer to the sample mean, for the same reason the shape of the Credible Intervals for the Latent intensity $\theta_t = exp(x_t)$ is very similar to the one from the data. However, the Credible Interval bounds are larger than with the non-informative prior-WHY?.

Appendix

```
knitr::opts_chunk$set(echo=FALSE, eval=TRUE)
# R version
RNGversion('3.5.1')

# libraries
library(rstan)

# Seed
set.seed(1234567890)
################# QUESTION 1 ###################


#### (a)

# Given values
mu = 10
sigma2  = 2
t = 200

# AR simulator
sim_AR_1 = function(start, mu, sigma2, phi, t){
  x = numeric(t)
  x[1] = start
  for (i in 2:t) {
    eps = rnorm(1, mean = 0, sd = sqrt(sigma2))
    x[i] = mu + phi*(x[i-1] - mu) + eps
  }
  return(x)
}

# Plot of one simulation for phi=1
set.seed(1234567890)
x1 = sim_AR_1(mu, mu, sigma2, 1, t)
plot(x1, type = 'l', main = "Simulation of Xt", xlab="Iteration", ylab="Xt")

### (b)

# Synthetic data
set.seed(1234567890)
x = sim_AR_1(mu, mu, sigma2, 0.3, t)
y = sim_AR_1(mu, mu, sigma2, 0.95, t)

# Stan model
StanModel = '
data {
  int<lower=0> N;
  vector[N] y;
}
parameters {
  real mu;
  real<lower=-1, upper=1> phi; // this is not recommended, but runs better than with a restrictive prio
  real<lower=0> sigma;
}
model {
  mu ~ normal(10,100); //non-informative, because of large sd
  phi ~ normal(0,10); //weakly informative, because we know it has to be between -1 and 1
  sigma ~ scaled_inv_chi_square(1,1.4); //non-informative, because of small df
  for (n in 2:N)
    y[n] ~ normal(mu+phi*(y[n-1]-mu),sigma);
}'

# i.

# Simulation for y_t
N = length(y)
data_y = list(N=N, y=y)
burnin = 1000
niter = 2000
fit_y = stan(model_code = StanModel, data = data_y, warmup = burnin, iter = niter, chains = 4)

# Simulation for x_t
data_x = list(N=N, y=x)
fit_x = stan(model_code = StanModel, data = data_x, warmup = burnin, iter = niter, chains = 4)

# ii.

traceplot(fit_y)

traceplot(fit_x)

### (c)

# read data
data_campy = read.table('campy.dat', header = T)
plot(x=c(1:140), y=data_campy$c, type = 'l')

# Poisson Stan Model
StanModel_poisson = '
data {
  int<lower=0> N;
  int<lower=0> c[N];
}
parameters {
  real mu;
  real<lower=-1, upper=1> phi; // this is not recommended, but runs better than with a restrictive prio
  real<lower=0> sigma;
  real x[N];
}
model {
  mu ~ normal(0,100); // non-informative prior, we know nothing about mu
  phi ~ normal(0,10); //weakly informative, because we know it has to be between -1 and 1
  sigma ~ scaled_inv_chi_square(1,2); //non-informative, because of small df, we know nothing about sig
  for (n in 2:N){
    x[n] ~ normal(mu+phi*(x[n-1]-mu),sigma);
    c[n] ~ poisson(exp(x[n]));
  }
}'

N = length(data_campy$c)
data = list(N=N, c=data_campy$c)
burnin = 1000
niter = 2000
fit_poisson = stan(model_code = StanModel_poisson, data = data, warmup = burnin, iter = niter, chains =

# Extract posterior samples
postDraws <- extract(fit_poisson)
#plot
intensity_posterior = data.frame(exp(postDraws[["x"]]))
intensity_posterior_means = colMeans(intensity_posterior)
intensity_posterior_sd = apply(intensity_posterior, 2, sd)

#first row = lower bound of 95% interval
#second row = upper bound of 95% interval
intensity_95_intervals = rbind(intensity_posterior_means - 1.96*intensity_posterior_sd,
                               intensity_posterior_means + 1.96*intensity_posterior_sd)

plot(x=c(1:140), y=data_campy$c, type = 'l', lwd = 2, xlab = 'time', ylab = 'infections')
lines(x=c(1:140), y=intensity_posterior_means, col = 'red', lwd = 2)
lines(x=c(1:140), y=intensity_95_intervals[1,], col = 'green')
lines(x=c(1:140), y=intensity_95_intervals[2,], col = 'orange')
legend(x='topleft', legend=c('Data', 'Posterior mean', '95% lower CI', '95% upper CI'),
       col = c('black', 'red', 'green', 'orange'), lwd = 2)

### (d)

# Poisson Stan Model with informative prior
StanModel_poisson2 = '
data {
  int<lower=0> N;
  int<lower=0> c[N];
}
parameters {
  real mu;
  real<lower=-1, upper=1> phi; // this is not recommended, but runs better than with a restrictive prio
  real<lower=0> sigma;
  real x[N];
}
model {
  mu ~ normal(0,100); // non-informative prior, we know nothing about mu
  phi ~ normal(0,10); //weakly informative, because we know it has to be between -1 and 1
  sigma ~ scaled_inv_chi_square(100,1); // informative, because of big df, we know that the error is sma
  for (n in 2:N){
    x[n] ~ normal(mu+phi*(x[n-1]-mu),sigma);
    c[n] ~ poisson(exp(x[n]));
  }
}'
```

```r
N = length(data_campy$c)
data = list(N=N, c=data_campy$c)
burnin = 1000
niter = 2000
fit_poisson2 = stan(model_code = StanModel_poisson2, data = data, warmup = burnin, iter = niter, chains

# Extract posterior samples
postDraws2 <- extract(fit_poisson2)

#plot
intensity_posterior2 = data.frame(exp(postDraws2[["x"]]))
intensity_posterior_means2 = colMeans(intensity_posterior2)
intensity_posterior_sd2 = apply(intensity_posterior2, 2, sd)

#first row = lower bound of 95% interval
#second row = upper bound of 95% interval
intensity_95_intervals2 = rbind(intensity_posterior_means2 - 1.96*intensity_posterior_sd2,
                                intensity_posterior_means2 + 1.96*intensity_posterior_sd2)

plot(x=c(1:140), y=data_campy$c, type = 'l', lwd = 2, xlab = 'time', ylab = 'infections')
lines(x=c(1:140), y=intensity_posterior_means2, col = 'red', lwd = 2)
lines(x=c(1:140), y=intensity_95_intervals2[1,], col = 'green')
lines(x=c(1:140), y=intensity_95_intervals2[2,], col = 'orange')
legend(x='topleft', legend=c('Data', 'Posterior mean', '95% lower CI', '95% upper CI'),
       col = c('black', 'red', 'green', 'orange'), lwd = 2)
```