Lab8_STA602

ElenaW.

11/1/2021

```
require(tidyverse)
```

```
## Loading required package: tidyverse

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
require(rstanarm)
```

```
## Loading required package: rstanarm

## Loading required package: Rcpp

## This is rstanarm version 2.21.1

## - See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!

## - Default priors may change, so it's safest to specify priors, even if equivalent to the defaults.

## - For execution on a local, multicore CPU with excess RAM we recommend calling

##   options(mc.cores = parallel::detectCores())
```

```
require(magrittr)
```

```
## Loading required package: magrittr

##
## Attaching package: 'magrittr'
## The following object is masked from 'package:purrr':
##
##     set_names
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
require(rstan)
```

```
## Loading required package: rstan
```

```
## Loading required package: StanHeaders
```

```
## rstan (Version 2.21.2, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
## Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars file
```

```
##
## Attaching package: 'rstan'
```

```
## The following object is masked from 'package:magrittr':
##
##     extract
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
require(bayesplot)
```

```
## Loading required package: bayesplot
```

```
## This is bayesplot version 1.8.1
```

```
## - Online documentation and vignettes at mc-stan.org/bayesplot
```

```
## - bayesplot theme set to bayesplot::theme_default()
```

```
##    * Does _not_ affect other ggplot2 plots
```

```
##    * See ?bayesplot_theme_set for details on theme setting
```

```
require(loo)
```

```
## Loading required package: loo

## This is loo version 2.4.1

## - Online documentation and vignettes at mc-stan.org/loo

## - As of v2.0.0 loo defaults to 1 core but we recommend using as many as possible. Use the 'cores' arg

## - Windows 10 users: loo may be very slow if 'mc.cores' is set in your .Rprofile file (see https://gi

##
## Attaching package: 'loo'

## The following object is masked from 'package:rstan':
##
##     loo
```

```
require(readxl)
```

```
## Loading required package: readxl
```

```
require(plyr)
```

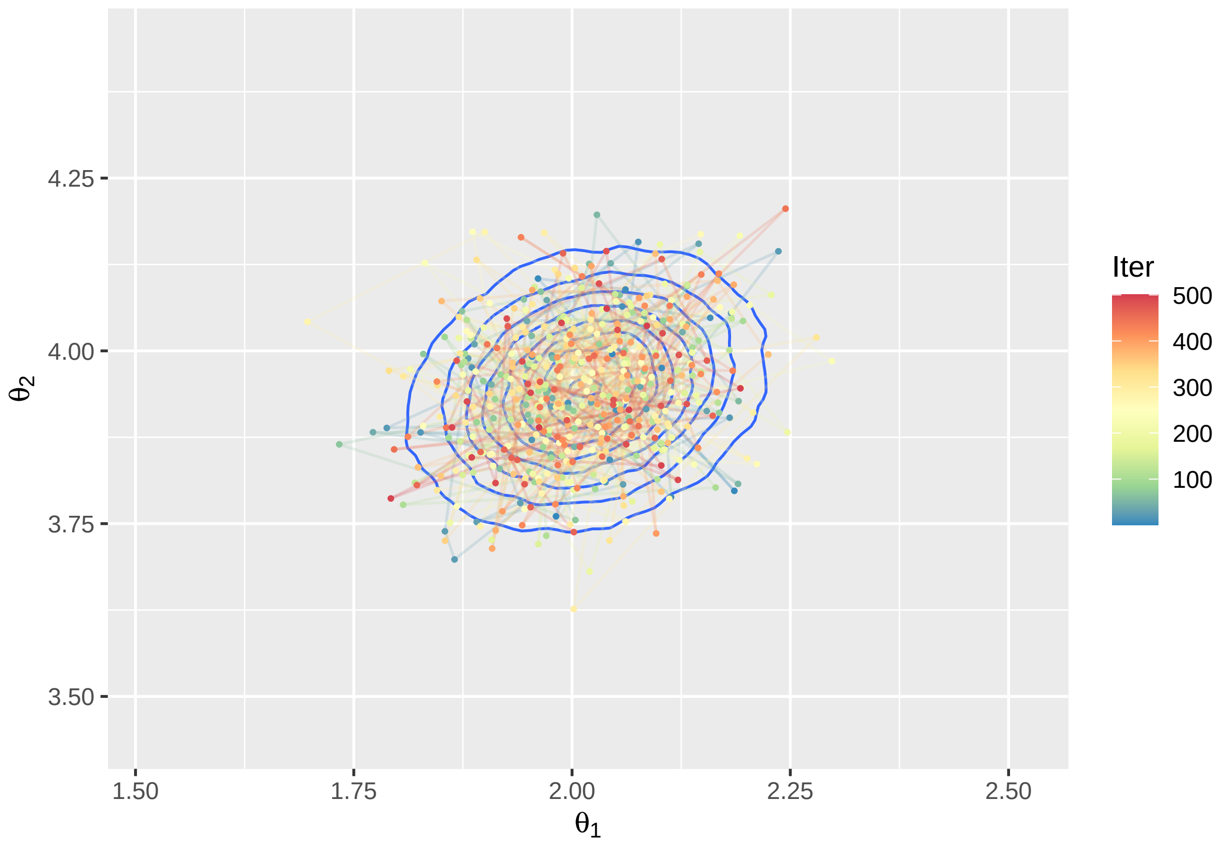
```
## Loading required package: plyr

## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:purrr':
##
##     compact
```

Exercise 4

```
normal_gibbs_sampler = function(S, X, rho){
  theta_1 = 0
  theta_2 = 0
  xbar_1 = colMeans(X)[1]
  xbar_2 = colMeans(X)[2]
  theta = matrix(0, nrow = S, ncol = ncol(X))
  n = nrow(X)
  for (i in 1:S){
    mu_1 = n*(xbar_1 + rho*theta_2 -    rho*xbar_2)/(n+1-rho^2)
    var_1 = (1-rho^2)/(n+1-rho^2)
    theta_1 = rnorm(1, mean = mu_1, sd = sqrt(var_1))
    mu_2 = n*(xbar_2 + rho*theta_1 - rho*xbar_1)/(n+1 - rho^2)
    var_2 = (1-rho^2)/(n+1-rho^2)
    theta_2 = rnorm(1, mean = mu_2, sd = sqrt(var_2))
    theta[i,1] = theta_1
    theta[i,2] = theta_2
  }
  return(theta)
}
```

```
set.seed(0)
n <- 100
rho <- 0.2
X <- MASS::mvrnorm(n = n, mu = c(2, 4), Sigma = matrix(c(1, rho, rho, 1), nrow = 2))
Sigma_post <- matrix(((1-rho^2)/((n+1-rho^2)^2 - (n^2)*(rho^2)))*c(n+1-rho^2, n*rho, n*rho, n+1-rho^2),
mu_post <- n*Sigma_post%*%matrix(c(1/(1-rho^2), -rho/(1-rho^2),
                                                       -rho/(1-rho^2), 1/(1-rho^2)),
                                                       nrow = 2)%*%colMeans(X)
norm_gibbs_samps <- normal_gibbs_sampler(600, X, rho)
#
true_post <- MASS::mvrnorm(n = 100000,
                           mu = mu_post,
                           Sigma = Sigma_post)
data.frame(norm_gibbs_samps) %>%
  magrittr::set_colnames(c("theta_1", "theta_2")) %>%
  dplyr::mutate(iter = 1:n()) %>%
  dplyr::filter(iter > 100) %>%
  dplyr::mutate(iter = 1:n()) %>%
  ggplot2::ggplot() +
  geom_density2d(data = data.frame(true_post) %>%
                        magrittr::set_colnames(c("true_1", "true_2")),
                 aes(x = true_1, y = true_2)) +
  geom_path(aes(x = theta_1, y = theta_2, colour = iter), alpha = 0.2, size = 0.5) +
  geom_point(aes(x = theta_1, y = theta_2, colour = iter), size = 0.5) +
  scale_color_distiller(palette = "Spectral", name = "Iter") +
  labs(x = expression(theta[1]), y = expression(theta[2])) +
  xlim(c(mu_post[1] - 0.5, mu_post[1] + 0.5)) +
  ylim(c(mu_post[2] - 0.5, mu_post[2] + 0.5))
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density2d).
```

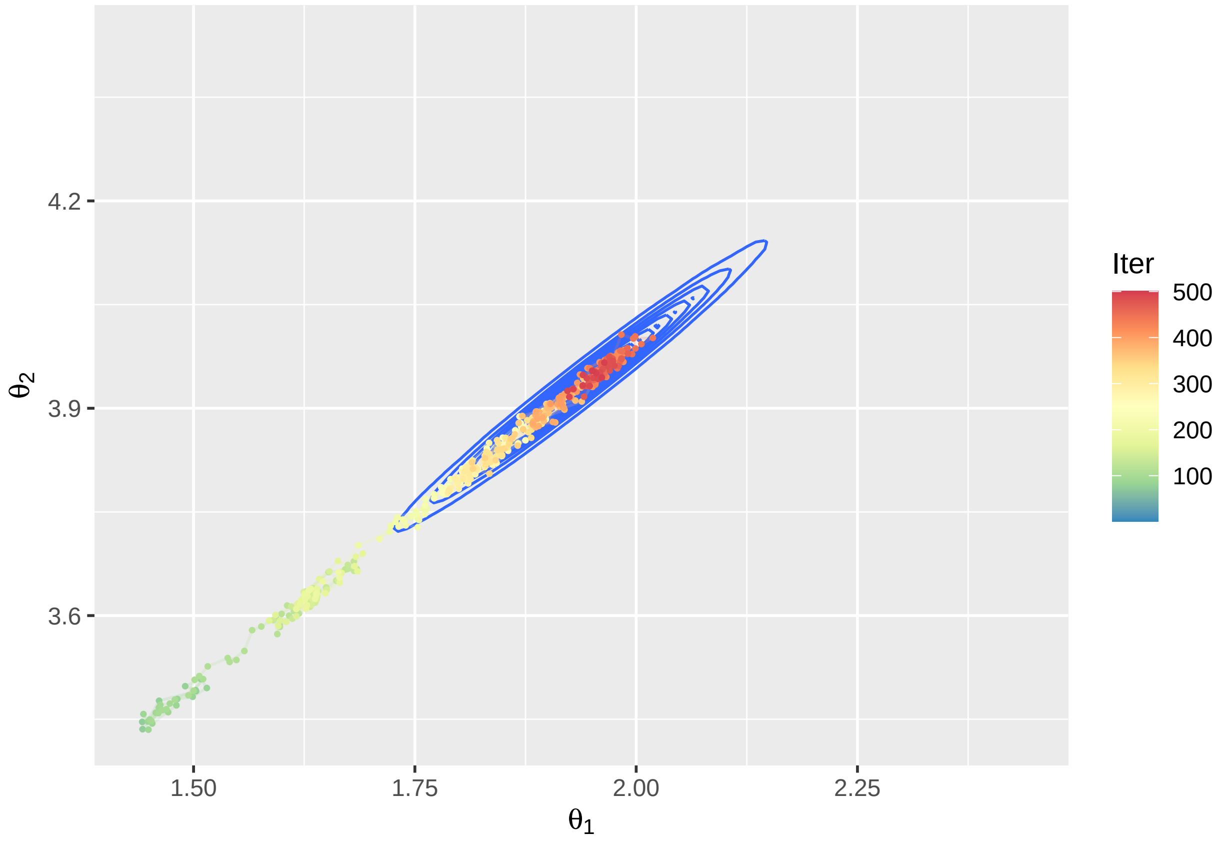


I can't run stan file since my R would be aborted. So i only could compare it with in html.

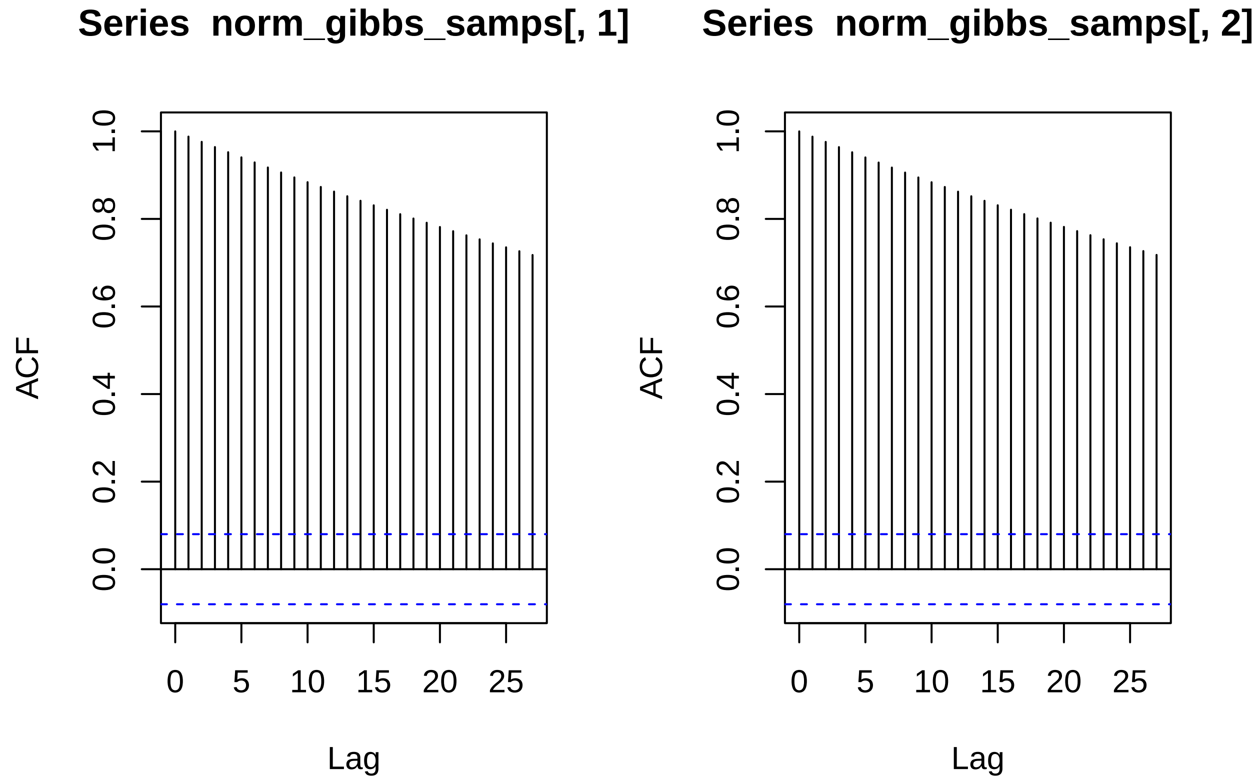
```
n <- 100
rho <- 0.995
X <- MASS::mvrnorm(n = n, mu = c(2, 4), Sigma = matrix(c(1, rho, rho, 1), nrow = 2))
Sigma_post <- matrix(((1-rho^2)/((n+1-rho^2)^2 - (n^2)*(rho^2)))*c(n+1-rho^2, n*rho, n*rho, n+1-rho^2),
mu_post <- n*Sigma_post%*%matrix(c(1/(1-rho^2), -rho/(1-rho^2),
                                    -rho/(1-rho^2), 1/(1-rho^2)),
                                    nrow = 2)%*%colMeans(X)
norm_gibbs_samps <- normal_gibbs_sampler(600, X, rho)
#
true_post <- MASS::mvrnorm(n = 100000,
                           mu = n*Sigma_post%*%(matrix(c(1/(1-rho^2), -rho/(1-rho^2),
                                                 -rho/(1-rho^2), 1/(1-rho^2)),
                                                 nrow = 2)%*%colMeans(X)),
                           Sigma = Sigma_post)
#
data.frame(norm_gibbs_samps) %>%
  magrittr::set_colnames(c("theta_1", "theta_2")) %>%
  dplyr::mutate(iter = 1:n()) %>%
  dplyr::filter(iter > 100) %>%
  dplyr::mutate(iter = 1:n()) %>%
  ggplot2::ggplot() +
  geom_density2d(data = data.frame(true_post) %>%
                      magrittr::set_colnames(c("true_1", "true_2")),
                 aes(x = true_1, y = true_2)) +
  geom_path(aes(x = theta_1, y = theta_2, colour = iter), alpha = 0.2, size = 0.5) +
  geom_point(aes(x = theta_1, y = theta_2, colour = iter), size = 0.5) +
  scale_color_distiller(palette = "Spectral", name = "Iter") +
  labs(x = expression(theta[1]), y = expression(theta[2])) +
  xlim(c(mu_post[1] - 0.5, mu_post[1] + 0.5)) +
  ylim(c(mu_post[2] - 0.5, mu_post[2] + 0.5))
```

```
## Warning: Removed 75 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 75 rows containing missing values (geom_point).
```



```
#
par(mfrow = c(1,2))
acf(norm_gibbs_samps[,1])
acf(norm_gibbs_samps[,2])
```



Exercise 5

With increasing rho to 0.995, the center of generated plot from Gibbs sampling is further away from the (0,0) and it has less concentration. The reason of this is that we set our prior means is (0,0), and the correlation is increasing while rho is increasing, which means that we get high prior correlation and the thetas in Gibbs moved away from (0,0) in a lower pace. However, the generated plot from Stan file has nicer concentration and tends to concentrate to the true well density curve.

Exercise 6

Because we have larger rho in the Gibbs sampling, which means that the dependency between theta1 and theta2 is stronger. And the larger dependency causes the thetas moving very slowly, which is also a reason why we see that there are many simulated points are concentrated on the lower left corner and upper right corner, and the curve is really thin.