P8133 Week 8 Session 1: Rstan tutorial and simple linear regression

0. Initialize `Rstan`
(If you have not installed Rstan yet, please check https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started for detail.)

   We start `stan` by executing

   ```
   library(rstan)
   ```

   As the startup message says, if you are using rstan locally on a multicore machine and have plenty of RAM to estimate your model in parallel, at this point execute

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

   In addition, you should follow the second startup message that says to execute

   ```
   rstan_options(auto_write = TRUE)
   ```

   And set your working directory

   ```
   setwd("XXXXXXXX")
   ```

1. Learn about `stan`. Start with the simplest linear regression model, with a single predictor and a slope and intercept coefficient, and normally distributed noise. This model can be written using standard regression notation as

$$y_n = \beta_0 + \beta_1 x_n + \epsilon_n \quad \text{where} \quad \epsilon_n \sim N(0, \sigma^2)$$

   (a). First, generate simulated data

   ```
   set.seed(1)
   n = 30 # Number of observations
   beta0 = 10
   beta1 = 5
   sigma = 2
   x = rnorm(n, mean = 10)
   epsilon = rnorm(n)
   y = beta0 + beta1 * x + epsilon
   plot(y~x)
   ```

   (b). Run stan model

   ```
   stan_data = list(N =30, x = x, y = y)
   stan_fit = stan(file = "1LinearRegression.stan", data = stan_data,
   warmup = 500, iter = 1000, chains = 4, seed = 1)
   print(stan_fit)
   ```

   `extract()` function can be used to put the posterior estimates for each parameter into a list.

```
posterior = extract(stan_fit)
```

Please also run `lm` function with our simulated data and compare them with the outputs of the Stan models. (Point estimate, 95% Confidence Interval, ggplot)

(c). We have the samples from the posterior distribution. How can we visualize the variability in our estimation of the regression line in ggplot?

2. Changing our priors
In the previous part, we used uniform priors for all coefficients. Let's try again, but now with more informative priors for the relationship between y and x. We're going to use normal priors with small standard deviations. If we were to use normal priors with very large standard deviations (say 1000, or 10,000), they would act very similarly to uniform priors.

a). Run stan model

```
stan_data = list(N =30, x = model.matrix(~x), y = y, K = 2)
stan_fit2 = stan(file = "2LinearRegression.stan", data = stan_data,
warmup = 500, iter = 1000, chains = 4, seed = 1)
print(stan_fit2)
```

What is the probability that $\beta_1 > 5.1$?

(b). Change the stan script with different prior distribution. How will this affect posterior distribution?

3. Convergence Diagnostics

For diagnostics, we need to check the Rhat values, and the traceplots of our model parameters to make sure the model has converged and is reliable.

Please create traceplot for each parameter in the stan model.

4. Generate posterior predictive samples

Run stan model

```
stan_fit3 = stan(file = "3LinearRegression.stan", data = stan_data,
warmup = 500, iter = 1000, chains = 4, seed = 1)
y_rep = as.matrix(stan_fit3, pars = "y_rep")
dim(y_rep)
```

Now we have samples from posterior predictive distributions and each row is an iteration (single posterior estimate) from the posterior predictive distributions.

Please comparing density plot of y with densities plot of y over 200 posterior draws.