

# Homework Assignment 1

Tao Wu

2026-01-27

1. Write R code to do the following:

- (a) Simulate a random number representing the number of shots that a basketball player makes in 10 tries, if the results of the shots are independent and she has a 40% chance of making each shot.

```
set.seed(23)
shots_made = rbinom(10, 1, 0.4)
sum(shots_made)

## [1] 6

# below is an alternative solution
rbinom(n = 1, size = 10, prob = 0.4)
```

```
## [1] 6
```

- (b) Simulate a random number representing the number of shots that a basketball player makes in 10 tries, if the results of the shots are independent and her chance of making a shot is 10% for the first shot, 20% for the second shot, 30% for the third, etc.

```
set.seed(18)
tries = 10
shots_made = rep(NA, tries)
for (i in 1:tries) {
  shots_made[i] = rbinom(1, 1, 0.1*i)
}
sum(shots_made)
```

```
## [1] 6
```

```
# below is an alternative solution
probs = seq(0.1, 1, by = 0.1)
shots_made_1 = rbinom(10, 1, probs)
sum(shots_made_1)
```

```
## [1] 5
```

2. (a) Write R code to simulate 100 data points from a linear model with intercept 1, slope 2, and residual standard deviation 3, where the predictors are sampled at random uniformly from the range (0,4). Fit a linear regression to these data using `lm`, print the fitted model using the `display` function from the `arm` package, and report whether the estimates of the intercept and slope fall within 1 standard error of the true parameter values.

```
#install.packages("arm")
library(arm)
```

```
## Loading required package: MASS
```

```
## Loading required package: Matrix
```

```
## Loading required package: lme4

##
## arm (Version 1.14-4, built: 2024-4-1)
## Working directory is /Users/tiow/Downloads/1 STAT5293G004 Gelman/STAT_5293/Homeworks

set.seed(21)
n = 100
beta_0 = 1 # true intercept
beta_1 = 2 # true slope

predictor = runif(n, 0, 4)
residuals = rnorm(n, 0, 3) #residual standard deviation 3
y = beta_0 + predictor * beta_1 + residuals

fitted_linear_model = lm(y ~ predictor)
display(fitted_linear_model)

## lm(formula = y ~ predictor)
##               coef.est coef.se
## (Intercept)  0.74      0.62
## predictor    2.03      0.25
## ---
## n = 100, k = 2
## residual sd = 2.95, R-Squared = 0.41
if (abs(coef(fitted_linear_model)[1] - beta_0) <= se.coef(fitted_linear_model)[1]) {
  print("Intercept estimate falls within 1 standard error of the true intercept value.")
}

## [1] "Intercept estimate falls within 1 standard error of the true intercept value."
if (abs(coef(fitted_linear_model)[2] - beta_1) <= se.coef(fitted_linear_model)[2]) {
  print("Slope estimate falls within 1 standard error of the true slope value.")
}

## [1] "Slope estimate falls within 1 standard error of the true slope value."
```

(b) Fit the regression model using `stan_glm` and print the results.

```
#install.packages("rstanarm")
library(rstanarm)

## Loading required package: Rcpp
## This is rstanarm version 2.32.1
## - 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())
##
## Attaching package: 'rstanarm'
## The following objects are masked from 'package:arm':
##
##   invlogit, logit
```

```

stan_linear_model = stan_glm(y ~ predictor, data = data.frame(y, predictor), refresh = 0) # add refresh

print(stan_linear_model)

## stan_glm
## family:      gaussian [identity]
## formula:     y ~ predictor
## observations: 100
## predictors:   2
## -----
##              Median MAD_SD
## (Intercept)  0.7      0.6
## predictor    2.0      0.2
##
## Auxiliary parameter(s):
##              Median MAD_SD
## sigma 3.0      0.2
##
## -----
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
# quick linear/logistic regression bayesian analysis, use stan_glm

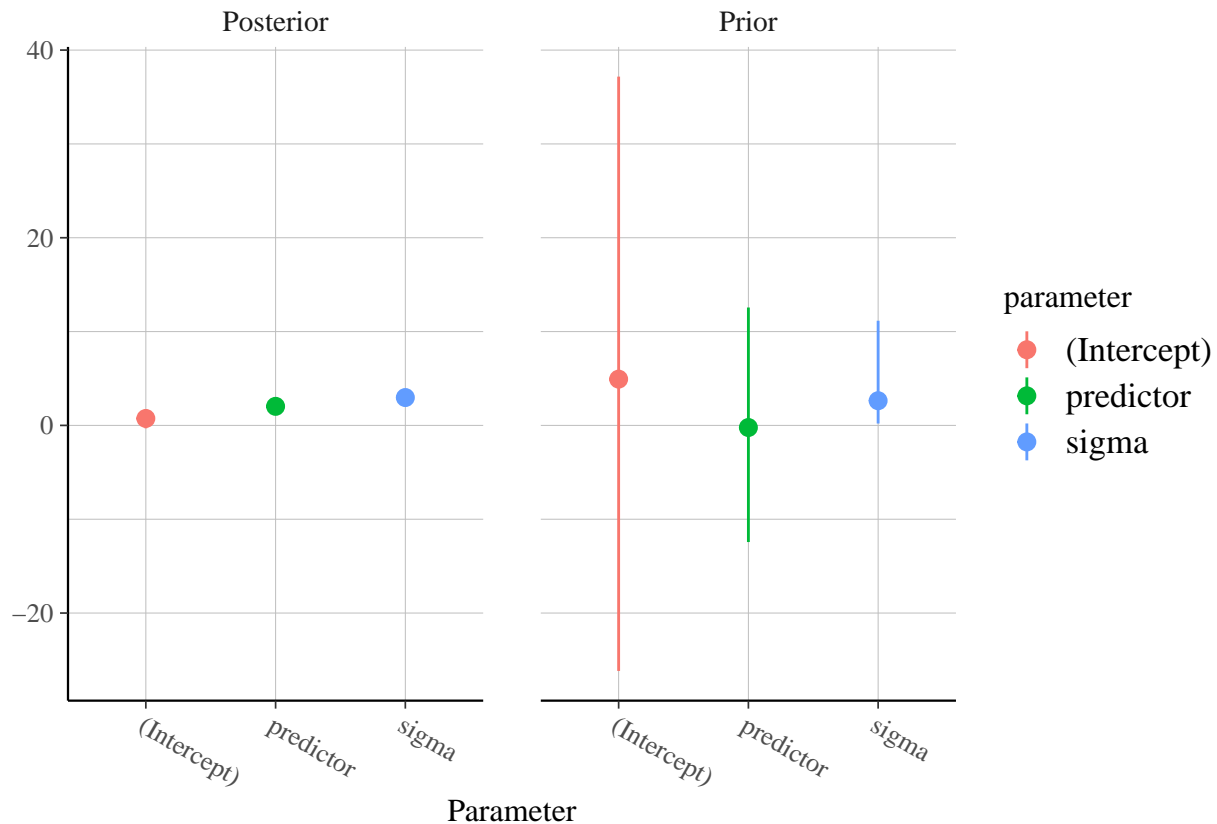
prior_summary(stan_linear_model)

## Priors for model 'stan_linear_model'
## -----
## Intercept (after predictors centered)
## Specified prior:
## ~ normal(location = 5.3, scale = 2.5)
## Adjusted prior:
## ~ normal(location = 5.3, scale = 9.6)
##
## Coefficients
## Specified prior:
## ~ normal(location = 0, scale = 2.5)
## Adjusted prior:
## ~ normal(location = 0, scale = 7.9)
##
## Auxiliary (sigma)
## Specified prior:
## ~ exponential(rate = 1)
## Adjusted prior:
## ~ exponential(rate = 0.26)
## -----
## See help('prior_summary.stanreg') for more details

posterior_vs_prior(stan_linear_model)

##
## Drawing from prior...

```



(c) Write a Stan program, fit the model using the *cmdstanr* package, and print the results.

```
#cmdstanr::install_cmdstan(cores = 4) #install the C++ toolchain and the CmdStan engine

#install.packages("cmdstanr", repos = c("https://mc-stan.org/r-packages/", getOption("repos")))
library(cmdstanr)

## This is cmdstanr version 0.8.0
## - CmdStanR documentation and vignettes: mc-stan.org/cmdstanr
## - CmdStan path: /Users/tiow/.cmdstan/cmdstan-2.38.0
## - CmdStan version: 2.38.0

#example(stan_model, package = "rstan", run.dontrun = TRUE)

stan_data = list(n = 100, x = predictor, y = y) #prepare the data
mod = cmdstan_model("linear_model.stan")        #compile the model

## Warning in readLines(stan_file): incomplete final line found on
## 'linear_model.stan'

fit_cmdstan = mod$sample(                        #fit model using MCMC
  data = stan_data,
  seed = 21,
  chains = 4,
  parallel_chains = 4,
  refresh = 0
)

## Running MCMC with 4 parallel chains...
```

```
## Chain 4 Informational Message: The current Metropolis proposal is about to be rejected because of th
## Chain 4 Exception: normal_lpdf: Scale parameter is 0, but must be positive! (in '/var/folders/7h/m_g
## Chain 4 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 4 but if this warning occurs often then your model may be either severely ill-conditioned or m
## Chain 4
## Chain 1 finished in 0.1 seconds.
## Chain 2 finished in 0.0 seconds.
## Chain 3 finished in 0.0 seconds.
## Chain 4 finished in 0.0 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.0 seconds.
## Total execution time: 0.2 seconds.
```

```
fit_cmdstan$print()
```

```
##   variable    mean  median   sd  mad      q5      q95 rhat ess_bulk ess_tail
## lp__        -157.81 -157.48 1.25 1.06 -160.31 -156.42 1.00    1430    2411
## intercept     0.74   0.74 0.65 0.64  -0.33   1.81 1.00    1730    1611
## slope         2.03   2.03 0.26 0.25   1.60   2.46 1.00    1704    1616
## sigma         3.00   2.99 0.22 0.21   2.65   3.39 1.00    1796    1804
```

```
# want full control on custom likelihoods, unusual priors, or complex hierarchical structures that aren
```