Stan Regularized Linear Models

Michael Rose

4/23/2019

Introduction

In this document we will estimate a linear model using the STAN Bayesian modeling engine.

The four steps to a Bayesian analysis are the following:

- 1. Specify a joint distribution for the outcome and all the unknowns. This takes the form of a marginal prior distribution for the unknowns multiplied by a likelihood for the outcomes conditional on the unknowns. Our joint distribution is proportional to a posterior distribution of the unknowns conditioned on the observed data.
- 2. Draw from our posterior with Markov Chain Monte Carlo sampling
- 3. Evaluate how well the model fits the data and revise if necessary
- 4. Draw from the posterior predictive distribution of the outcomes given interesting values of the predictors in order to visualize how manipulations in the predictors affect the outcomes.

The Data

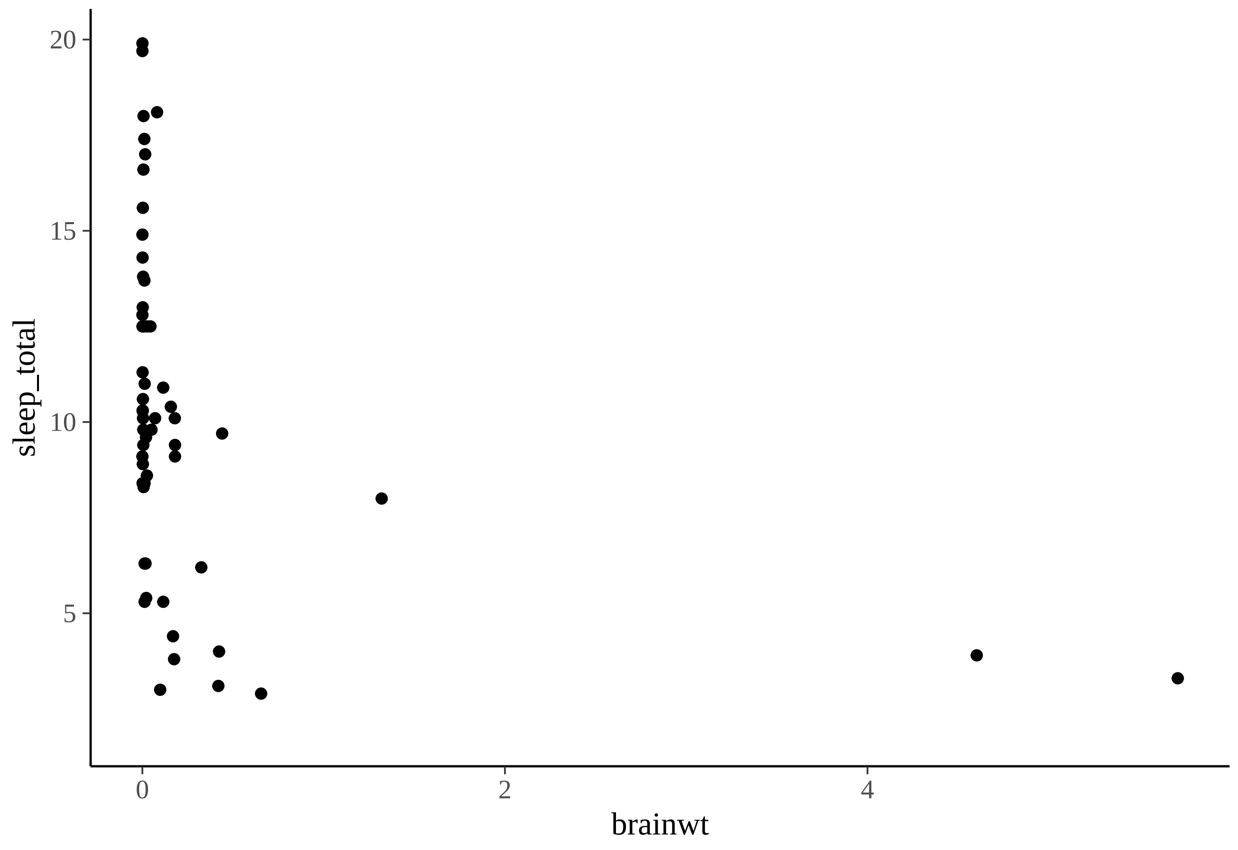
?msleep

```
msleep %>% head()
```

```
## # A tibble: 6 x 11
##   name  genus vore  order conservation sleep_total sleep_rem sleep_cycle
##   <chr> <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl>
## 1 Chee~ Acin~ carni Carn~ lc                  12.1      NA        NA
## 2 Owl ~ Aotus omni  Prim~ <NA>                17         1.8      NA
## 3 Moun~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA
## 4 Grea~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133
## 5 Cow   Bos   herbi Arti~ domesticated         4         0.7       0.667
## 6 Thre~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767
## # ... with 3 more variables: awake <dbl>, brainwt <dbl>, bodywt <dbl>
```

```
# remove NA
msleep %>%
  ggplot(aes(brainwt, sleep_total)) +
  geom_point()
```

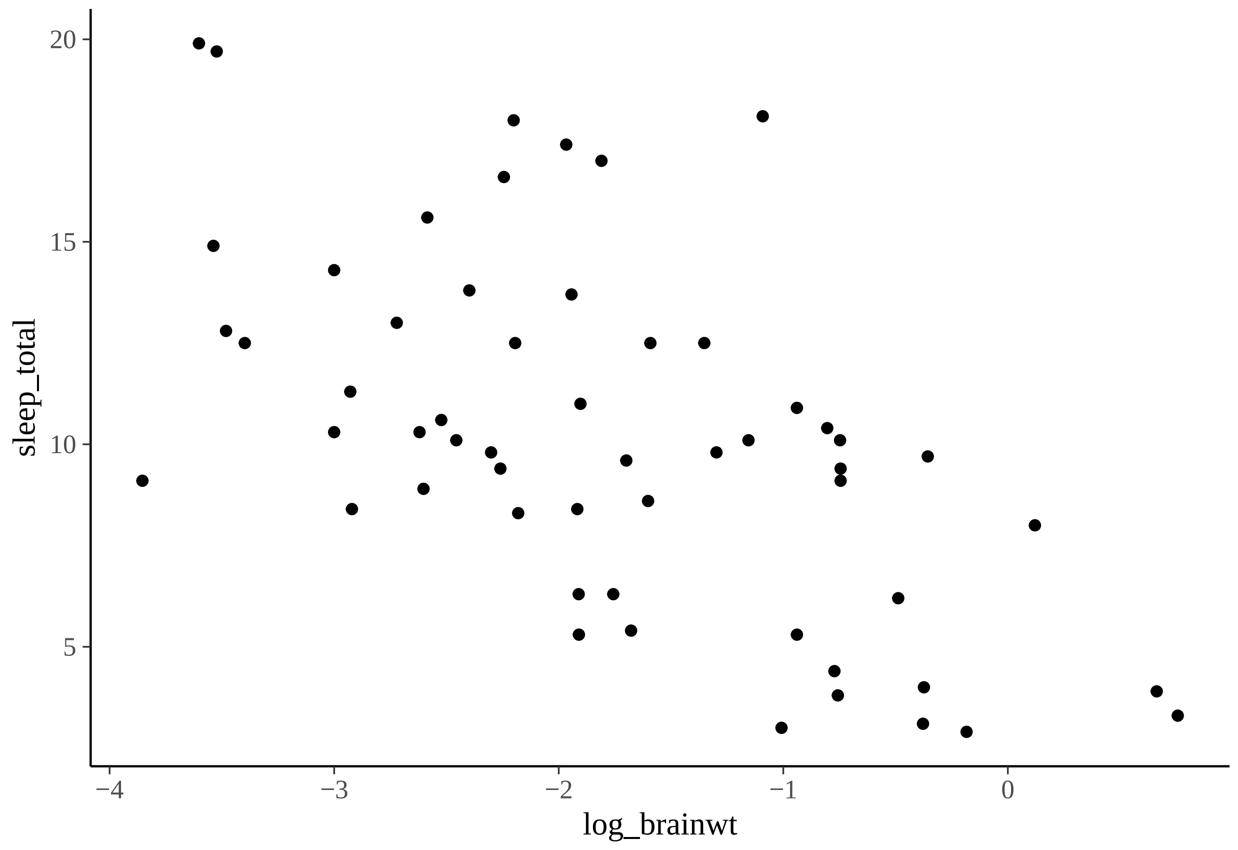
```
## Warning: Removed 27 rows containing missing values (geom_point).
```



This doesn't look too good. Lets try a transformation

```
# transform
msleep %<>%
  filter(!(is.na(brainwt))) %>%
  mutate(log_brainwt = brainwt %>% log10(),
         log_bodyweight = bodywt %>% log10(),
         log_sleep_total = sleep_total %>% log10())

# plot
msleep %>%
  ggplot(aes(log_brainwt, sleep_total)) +
  geom_point()
```

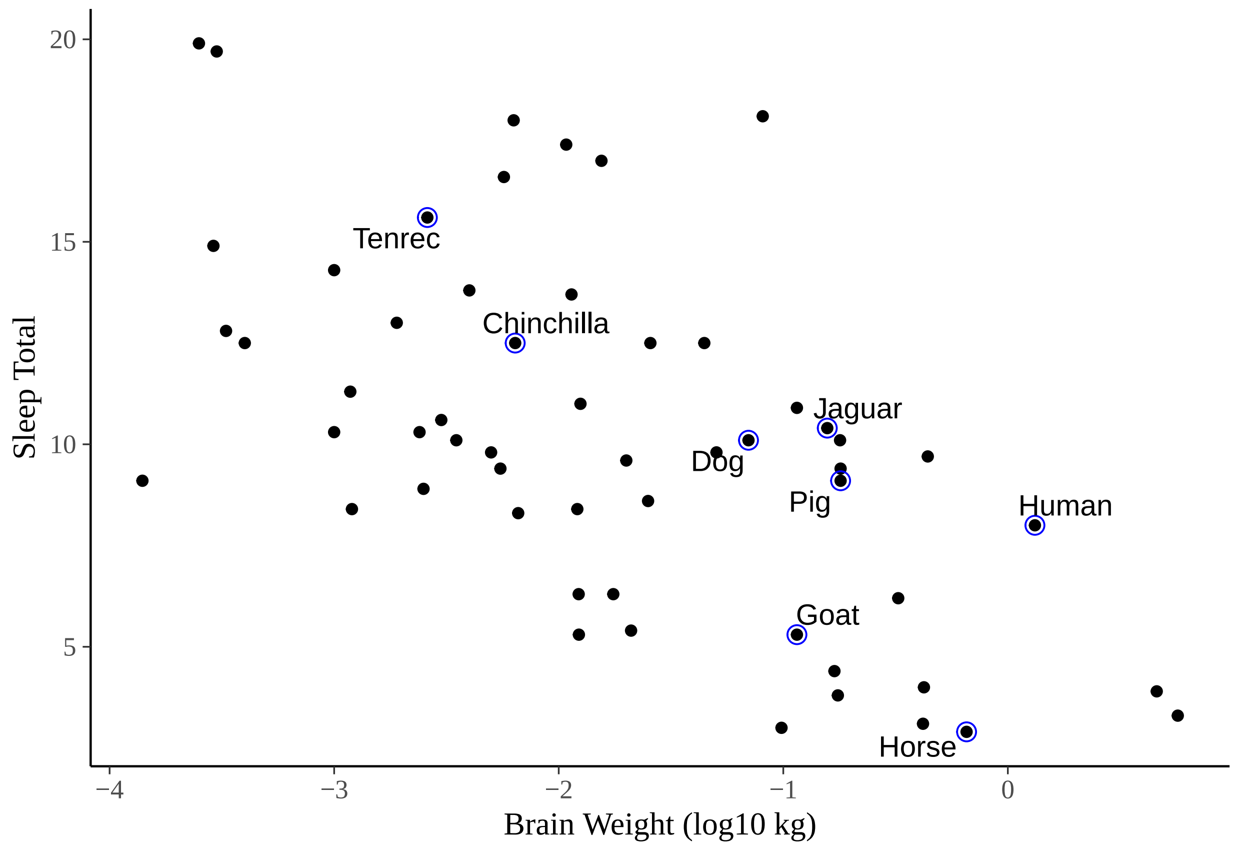


This looks much better. Lets see if we can get a better idea of what is going on here.

```r
# choose species to highlight
species <- c("Goat", "Horse", "Dog", "Human", "Jaguar", "Chinchilla", "Pig", "Tenrec")

# make df of chosen animals
species_df <- msleep %>%
  filter(name %in% species)

# plot
msleep %>%
  ggplot(aes(log_brainwt, sleep_total)) +
  geom_point() +
  geom_point(size = 3, shape = 1, color = "blue", data = species_df) +
  ggrepel::geom_text_repel(aes(label = name), data = species_df) +
  xlab(paste0("Brain Weight (", expression(log10), " kg)")) +
  ylab("Sleep Total")
```



Classical Regression

```r
# fit model
lm_classical <- lm(log_sleep_total ~ log_brainwt, data = msleep)

# check summary
summary(lm_classical)
```

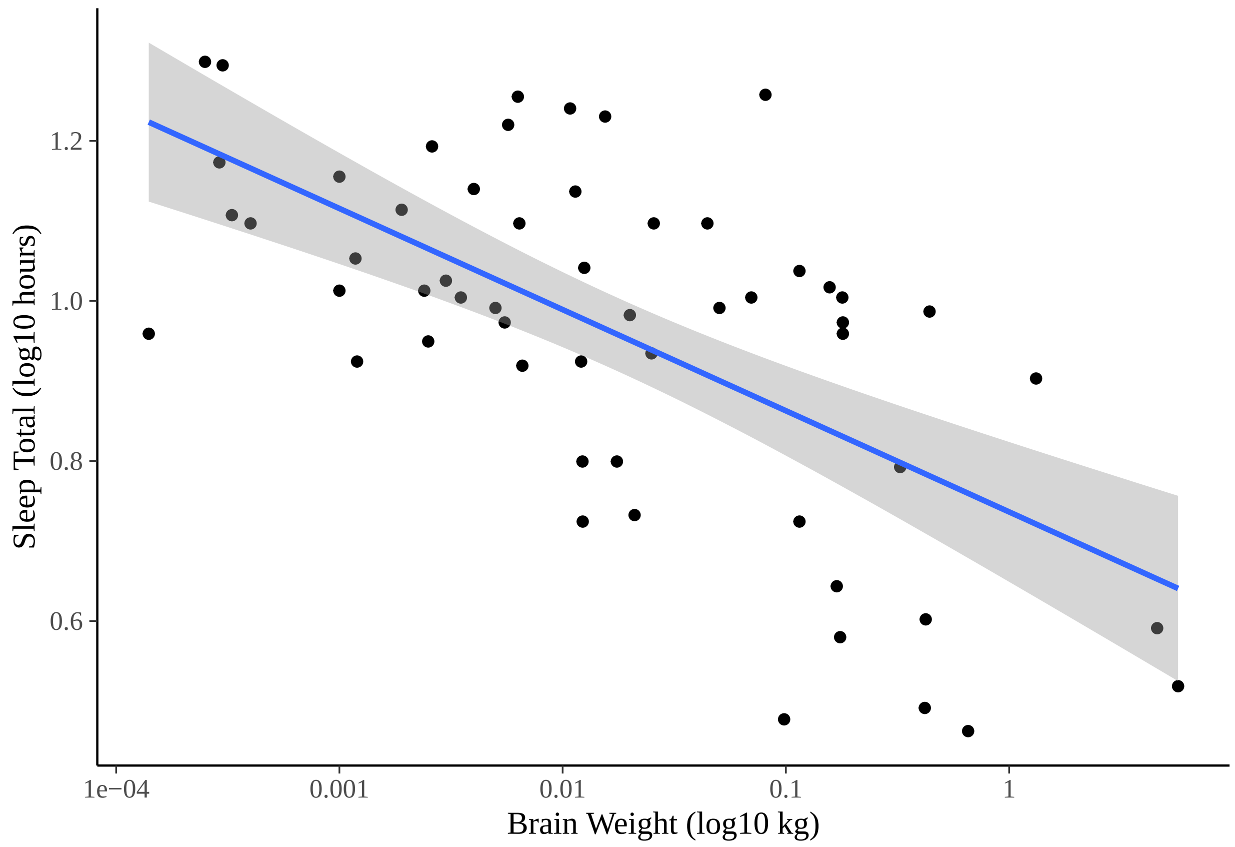
```
##
## Call:
## lm(formula = log_sleep_total ~ log_brainwt, data = msleep)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.3866 -0.1174 -0.0080  0.1332  0.3834
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.73635    0.04359  16.892  < 2e-16 ***
## log_brainwt -0.12640    0.02103  -6.011 1.64e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.1712 on 54 degrees of freedom
## Multiple R-squared:  0.4009, Adjusted R-squared:  0.3898
## F-statistic: 36.13 on 1 and 54 DF,  p-value: 1.637e-07
```

```
# grab coefficients
coef(lm_classical)
```

```
## (Intercept) log_brainwt
##   0.7363492  -0.1264049
```

```
msleep %>%
  ggplot(aes(log_brainwt, log_sleep_total)) +
  geom_point() +
  stat_smooth(method = "lm", level = 0.95) +
  scale_x_continuous(labels = function(x) {10^x}) +
  xlab(paste0("Brain Weight (", expression(log10), " kg)")) +
  ylab(paste0("Sleep Total (", expression(log10), " hours)"))
```



Multiple Plausible Regression Lines

```
# fit models
many_lines_model <- stan_glm(
  log_sleep_total ~ log_brainwt,
  family = gaussian(),
  data = msleep,
  prior = normal(0, 3),
  prior_intercept = normal(0, 3)
)

# check summary
summary(many_lines_model)
```

```
##
## Model Info:
##
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      log_sleep_total ~ log_brainwt
##  algorithm:    sampling
##  priors:       see help('prior_summary')
##  sample:       4000 (posterior sample size)
##  observations: 56
##  predictors:   2
##
## Estimates:
##                 mean   sd   2.5%   25%   50%   75%   97.5%
## (Intercept)    0.7    0.0  0.7    0.7   0.7   0.8   0.8
## log_brainwt   -0.1    0.0 -0.2   -0.1  -0.1  -0.1  -0.1
## sigma          0.2    0.0  0.1    0.2   0.2   0.2   0.2
## mean_PPD       1.0    0.0  0.9    0.9   1.0   1.0   1.0
## log-posterior 15.3    1.3 11.9   14.8  15.6  16.2  16.7
##
## Diagnostics:
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  3550
## log_brainwt   0.0  1.0  3160
## sigma         0.0  1.0  3704
## mean_PPD      0.0  1.0  3953
## log-posterior 0.0  1.0  1585
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
# check median parameter estimates
coef(many_lines_model)
```

```
## (Intercept) log_brainwt
##   0.7365420  -0.1256876
```

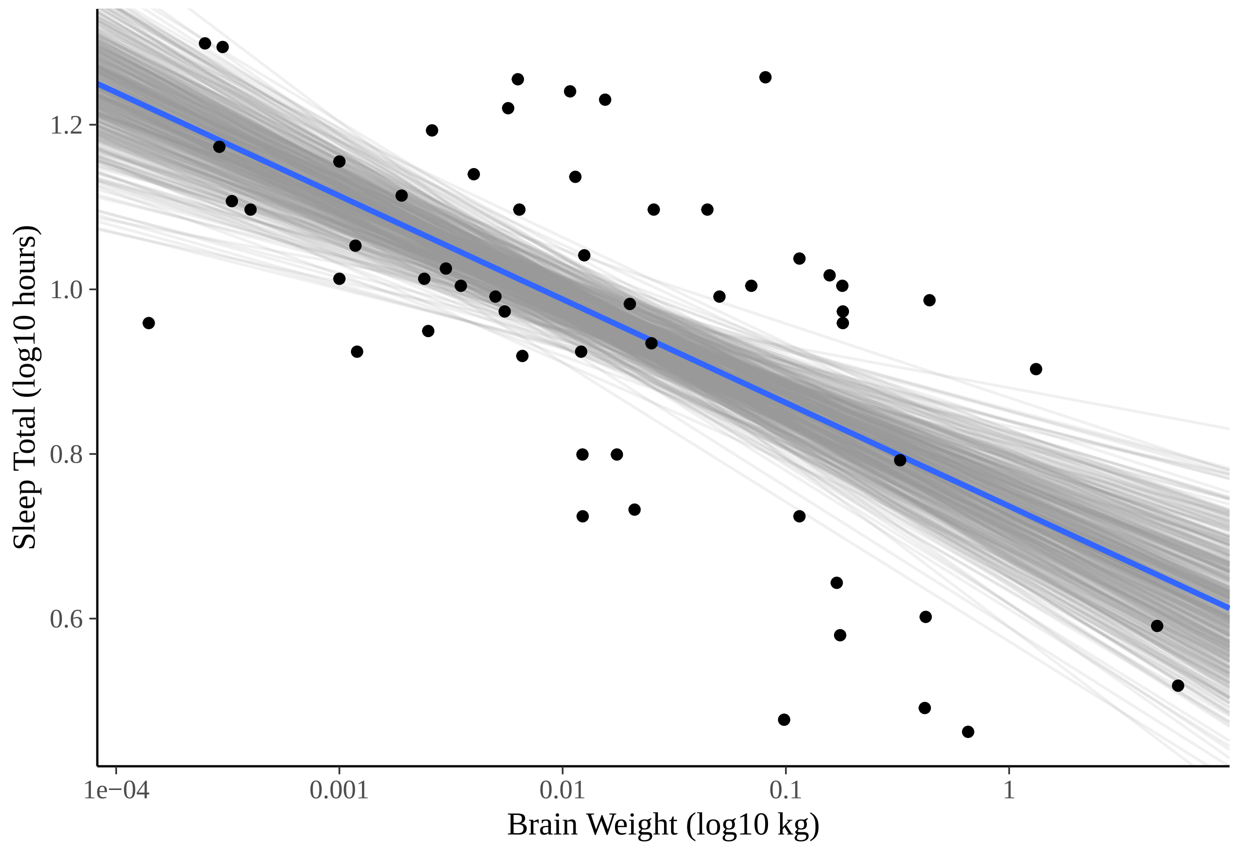
```
# sample the posterior and place each model into a dataframe
model_fits <- many_lines_model %>%
  as_tibble() %>%
  rename(intercept = "(Intercept)")

# look at data frame
model_fits %>% head()
```

```
## # A tibble: 6 x 3
##   intercept log_brainwt sigma
##       <dbl>       <dbl> <dbl>
## 1     0.680      -0.163 0.175
## 2     0.711      -0.145 0.166
## 3     0.706      -0.140 0.170
## 4     0.758      -0.119 0.180
## 5     0.704      -0.139 0.163
## 6     0.769      -0.116 0.146
```

```
# sample lines
n_draws <- 500
alpha_level <- 0.15
col_draw <- "grey60"
col_median <- "#3366FF"

msleep %>%
  ggplot(aes(log_brainwt, log_sleep_total)) +
  # plot sample of linear models
  geom_abline(aes(intercept = intercept, slope = log_brainwt),
              data = sample_n(model_fits, n_draws), color = col_draw, alpha = alpha_level) +
  # plot median values
  geom_abline(intercept = model_fits$intercept %>% median(),
              slope = model_fits$log_brainwt %>% median(),
              size = 1, color = col_median) +
  geom_point() +
  scale_x_continuous(labels = function(x) {10^x}) +
  xlab(paste0("Brain Weight (", expression(log10), " kg)")) +
  ylab(paste0("Sleep Total (", expression(log10), " hours)"))
```



Mean and 95% Confidence Interval

We can also draw a line of best fit and the 95% uncertainty interval around it.

```r
# get log_brainwt range
x_range <- range(msleep$log_brainwt)

# break the range into 80 steps
x_steps <- seq(x_range[1], x_range[2], length.out = 80)

# simulate data
sim_data <- tibble(
  observation = seq_along(x_steps),
  log_brainwt = x_steps
)
```

The function `posterior_linpred` returns the means of a model fitted on a data frame of new data.

```r
pred_lin_models <- posterior_linpred(many_lines_model, newdata = sim_data)

pred_lin_models %>% dim()
```

```
## [1] 4000   80
```