# STAT 656: Bayesian Data Analysis
# Fall 2024
# Homework 2

Juanwu Lu*

```
library("bayesplot")
```

```
## This is bayesplot version 1.11.1
## - 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
```

```
library("ggplot2")
library("rstan")
```

```
## Loading required package: StanHeaders
##
## rstan version 2.32.6 (Stan version 2.32.2)
## 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)
## For within-chain threading using `reduce_sum()` or `map_rect()` Stan functions,
## change `threads_per_chain` option:
## rstan_options(threads_per_chain = 1)
```

```
options(repr.plot.width = 6, repr.plot.height = 4)
```

## Synthetic Data

The file `hw2_synthetic.csv` is a dataset of count-valued measurements $\boldsymbol{y} = \{y_1, \ldots, y_n\}$, with $y_i \in 0, 1, \ldots$. Each output $y_i$ has an associated $x_i = (x_{i,1}, x_{i,2}) \in \mathbb{R}^2$, and write $\boldsymbol{x} = \{x_1, \ldots, x_n\}$'s as $\boldsymbol{x}$. We model $y_i$ as

$$y_i|\beta \sim \text{Poisson}(e^{f(x_i,\beta)}).$$

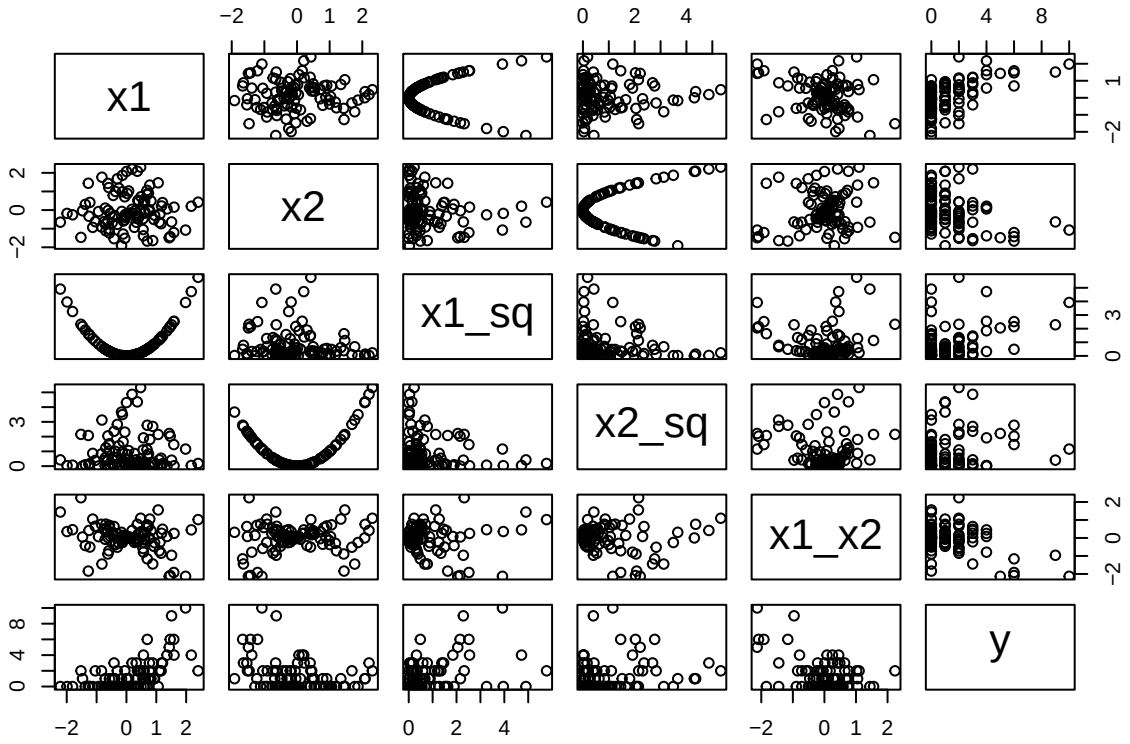
Here, the exponential is to ensure the Poisson rate is always positive, and the function $f(x_i, \beta) = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_3 x_{i,1}^2 + \beta_4 x_{i,2}^2 + \beta_5 x_{i,1} x_{i,2}$.

1. (25 points) **Solution**:

*College of Engineering, Purdue University, West Lafayette, IN, USA

```r
# Read the data
if (file.exists("data/hw2_synthetic.csv")) {
  data <- read.csv("data/hw2_synthetic.csv", header = TRUE)
} else {
  stop("FileNotFound: data file not found at 'data/hw2_synthetic.csv'.")
}

# Preprocess data to create the kernel terms
data$x1_sq <- data$x1^2
data$x2_sq <- data$x2^2
data$x1_x2 <- data$x1 * data$x2
data <- data[, c("x1", "x2", "x1_sq", "x2_sq", "x1_x2", "y")]
plot(data)
```




```r
linreg_poisson <- "
    // Input arguments to the model
    data {
        int<lower=0> n;     // Number of observations
        int<lower=0> k;     // Number of features
        matrix[n, k] X;     // Observation matrix
        int y[n];           // Integer response vector
    }
    parameters {}
    transformed_parameters {}
    model {}
"

linreg_poisson_model <- stan_model(model_code = linreg_poisson)
```