Week 4 assignment: Poisson models

Example solutions

01 February, 2016

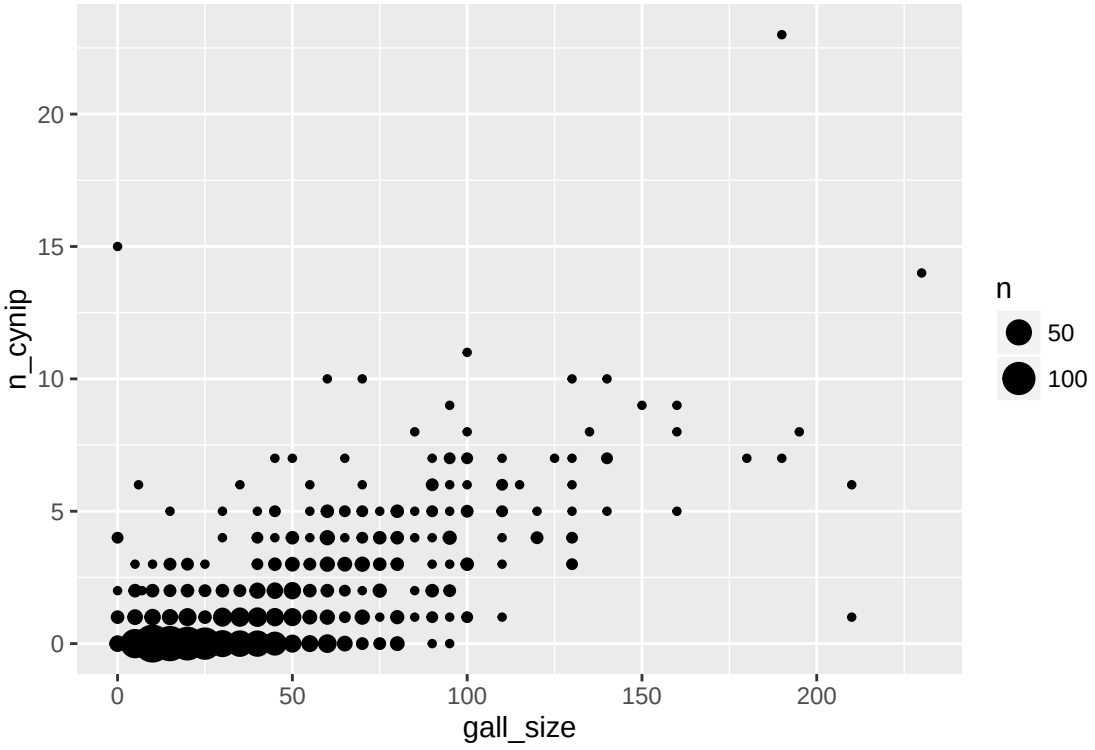
Wasps in the family Cynipidae lay their eggs on plants which form galls around the developing larvae, providing nutrition until the larvae metamorphose and burrow out of the galls, emerging as adults. From any particular gall, there is variation in the number of host wasps that emerge.

Here, you will construct a Bayesian model for the number of emerging cynipid wasps, using features of the galls as explanatory variables. The data are available in the `cleaned_galls.csv` file. Your task is to estimate the parameters of your model, and then to do a posterior predictive check to evaluate overdispersion.

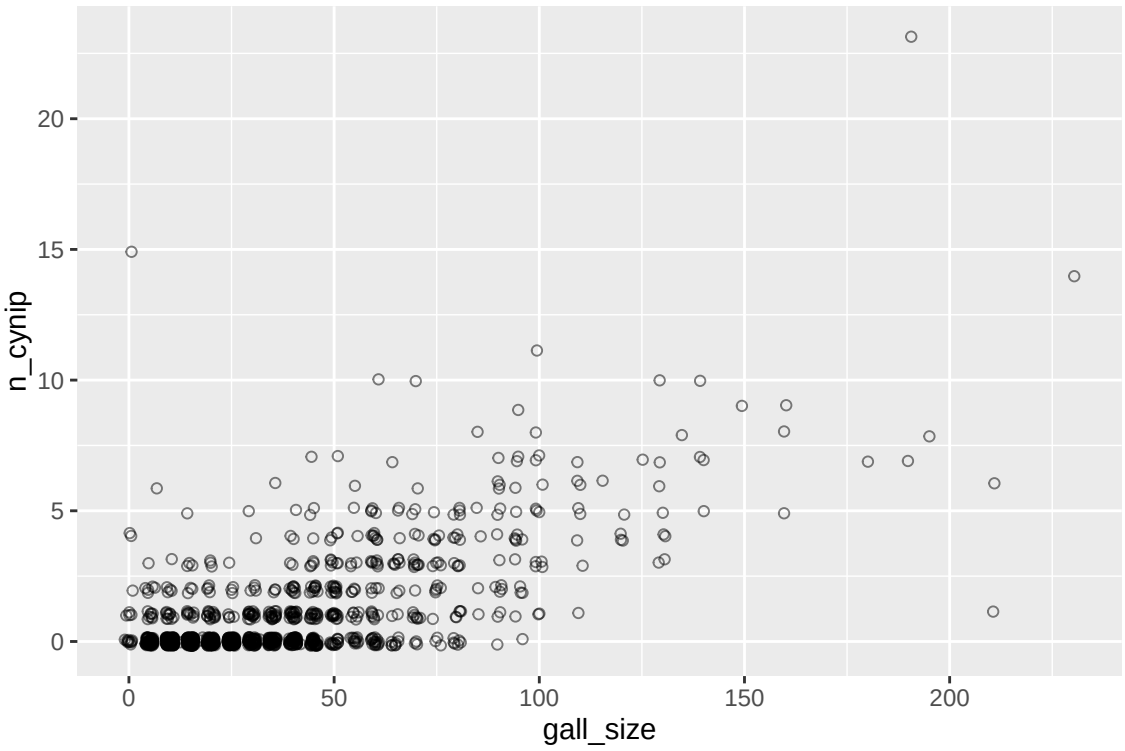
Problem 1: looking at the data

Load the data and explore how the features relate to the response variable.

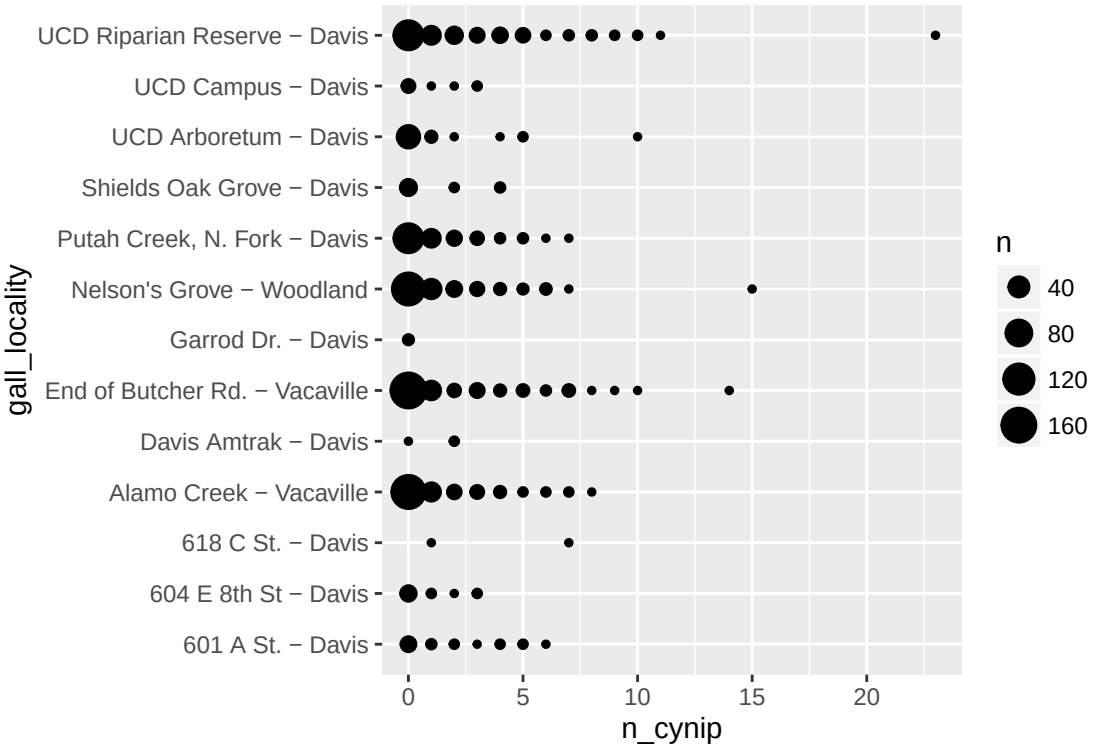
```
library(ggplot2)
d <- read.csv('cleaned_galls.csv')
ggplot(d, aes(x = gall_size, y = n_cynip)) +
  geom_count()
```



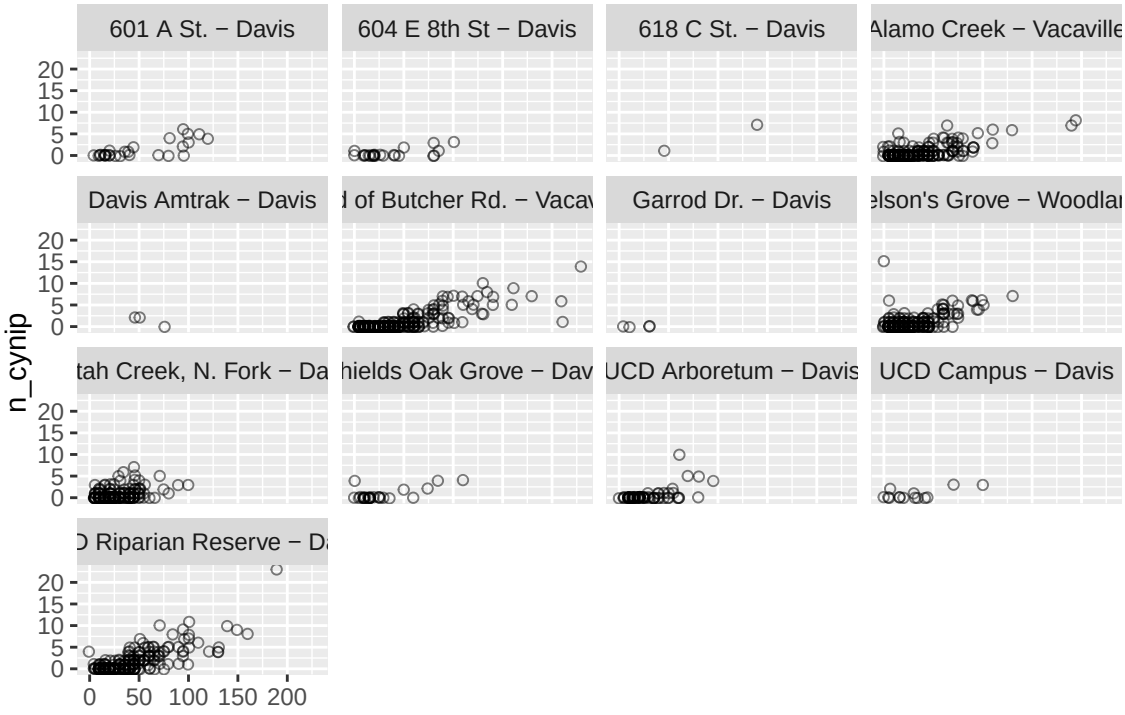

```
ggplot(d, aes(x = gall_size, y = n_cynip)) +
  geom_jitter(shape = 1, position = position_jitter(width=2.5, height=.4),
              alpha=.5)
```




```
ggplot(d, aes(y = gall_locality, x = n_cynip)) + geom_count()
```



```
ggplot(d, aes(x = gall_size, y = n_cynip)) +
  geom_jitter(shape = 1, position = position_jitter(width=2.5, height=.4),
              alpha=.5) +
  facet_wrap(~ gall_locality)
```



Problem 2: model specification

What is your model? Write it in LaTeX.

$y_i \sim Poisson(\lambda_i)$

$\lambda_i = e^{\beta^T X_i}$

What is your Stan model statement?

```
data {
  int n; // sample size
  int p; // number of coefficients
  matrix[n, p] X;
  int y[n];
}

parameters {
  vector[p] beta;
}

model {
  beta ~ normal(0, 5);
  y ~ poisson_log(X * beta);
}
```

Problem 3: parameter estimation

Use the `rstan` package to estimate your model parameters.

```
# center size
mean_size <- mean(d$gall_size)
sd_size <- sd(d$gall_size)
d$size <- (d$gall_size - mean(d$gall_size)) / sd_size

# make glm() construct a design matrix for me with a 2nd degree polynomial
m <- glm(n_cynip ~ size + I(size^2) + gall_locality, data=d, family=poisson)

library(rstan)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())

X <- model.matrix(m)
stan_d <- list(n = nrow(X), p = ncol(X), X = X, y = d$n_cynip)

iter <- 1000
chains <- 4
m_fit <- stan('poisson_glm.stan', data = stan_d,
              chains = chains, iter = iter, cores = 2)
```

Verify convergence using traceplots and the Rhat statistic:

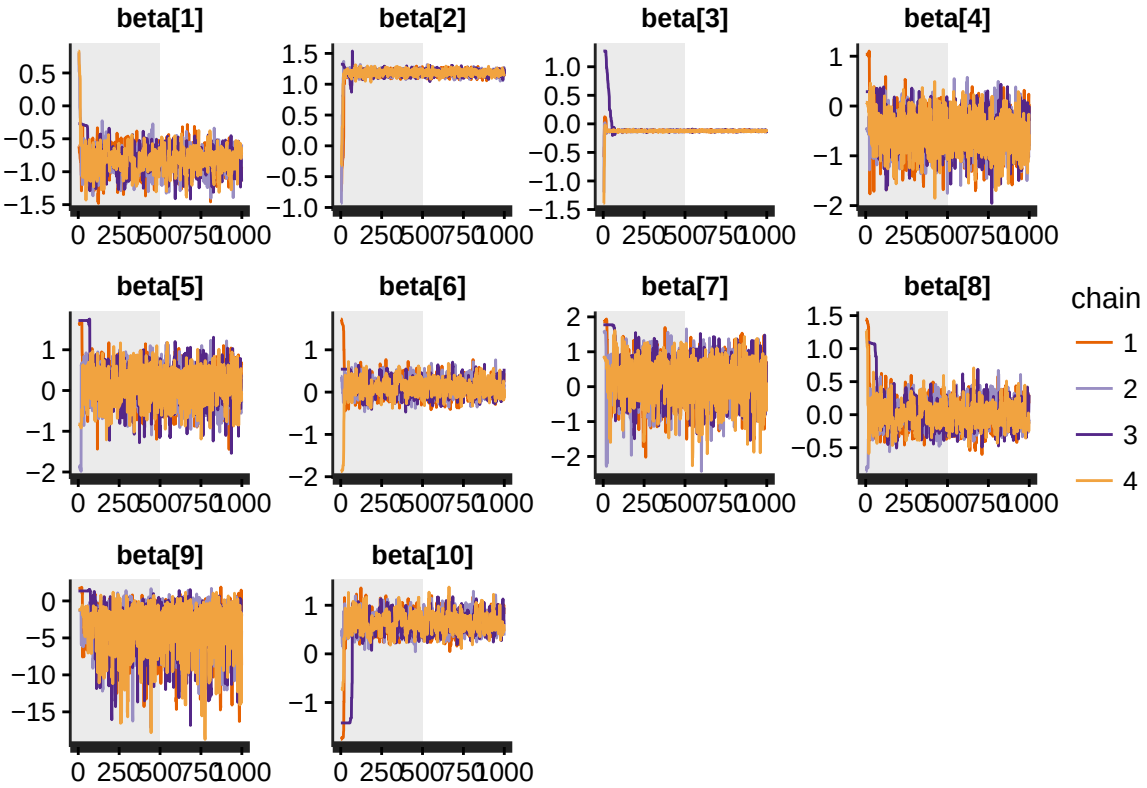
```
m_fit
```

```
## Inference for Stan model: poisson_glm.
## 4 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=2000.
##
##             mean se_mean   sd    2.5%     25%     50%     75%   97.5%
## beta[1]    -0.83    0.01 0.18   -1.21   -0.94   -0.81   -0.71   -0.49
## beta[2]     1.18    0.00 0.04    1.11    1.15    1.18    1.21    1.26
## beta[3]    -0.12    0.00 0.01   -0.14   -0.13   -0.12   -0.12   -0.11
## beta[4]    -0.54    0.01 0.38   -1.31   -0.79   -0.54   -0.28    0.16
## beta[5]     0.11    0.01 0.41   -0.73   -0.16    0.14    0.39    0.87
## beta[6]     0.12    0.01 0.19   -0.25   -0.01    0.10    0.24    0.53
## beta[7]     0.13    0.02 0.58   -1.10   -0.22    0.19    0.53    1.16
## beta[8]     0.01    0.01 0.19   -0.34   -0.12   -0.01    0.12    0.40
## beta[9]    -4.16    0.10 3.13  -11.79   -5.87   -3.54   -1.82    0.27
## beta[10]    0.66    0.01 0.19    0.30    0.53    0.64    0.78    1.06
## beta[11]    0.61    0.01 0.20    0.24    0.47    0.60    0.73    1.04
## beta[12]    0.13    0.01 0.30   -0.50   -0.07    0.14    0.33    0.72
## beta[13]    0.05    0.01 0.25   -0.43   -0.12    0.04    0.21    0.56
## beta[14]    0.06    0.01 0.40   -0.75   -0.21    0.07    0.35    0.80
## beta[15]    0.66    0.01 0.18    0.33    0.53    0.65    0.77    1.03
## lp__     -458.28    0.10 2.76 -464.53 -459.91 -457.95 -456.20 -453.90
##          n_eff Rhat
## beta[1]    321 1.01
## beta[2]    901 1.00
## beta[3]    929 1.00
```

```
## beta[4]    647 1.01
## beta[5]    912 1.00
## beta[6]    376 1.00
## beta[7]    940 1.00
## beta[8]    371 1.01
## beta[9]   1016 1.00
## beta[10]   352 1.00
## beta[11]   393 1.01
## beta[12]   605 1.00
## beta[13]   454 1.00
## beta[14]   790 1.00
## beta[15]   337 1.01
## lp__       815 1.00
##
## Samples were drawn using NUTS(diag_e) at Mon Feb  1 08:28:47 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
traceplot(m_fit, inc_warmup=TRUE)
```

```
## 'pars' not specified. Showing first 10 parameters by default.
```



Problem 4: posterior predictive check

Does your model adequately capture the variance in the emergence data, or is there overdispersion?

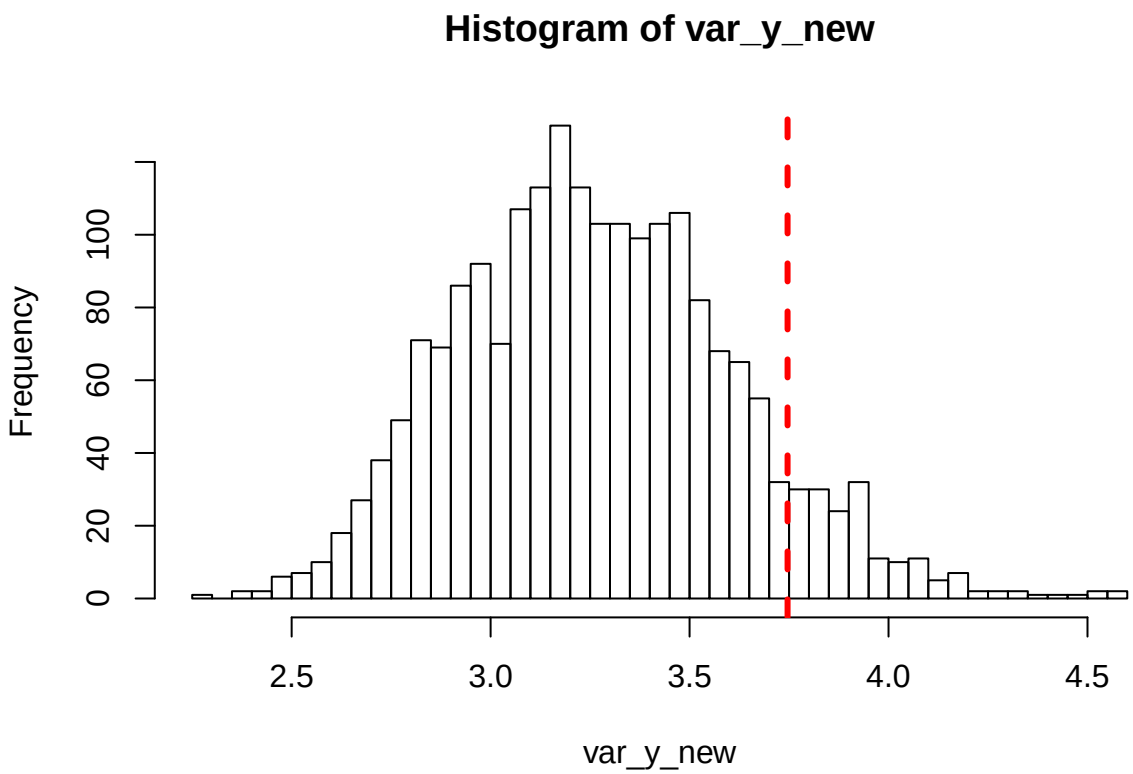
```r
# need to simulate datasets for each posterior draw
# the test statistic is var(y)
post <- rstan::extract(m_fit)
n_draws <- length(post$lp__)

sim_y <- function(X, beta) {
  # little function to simulate response data
  n <- nrow(X)
  lambda <- c(exp(X %*% beta))
  y <- rpois(n, lambda)
  return(y)
}

# create object to store variances of y
var_y_new <- rep(NA, n_draws)

# iterate through posterior draws and store variances of new data
for (i in 1:n_draws) {
  y_new <- sim_y(X, beta = post$beta[i, ])
  var_y_new[i] <- var(y_new)
}

# plot histogram of simulated variances
hist(var_y_new, breaks = 50)
# add line to indicate observed value
abline(v = var(d$n_cynip), col = 'red', lty = 2, lwd = 3)
```



The model seems to have adequately captured the variance in y.