STA 602 Lab

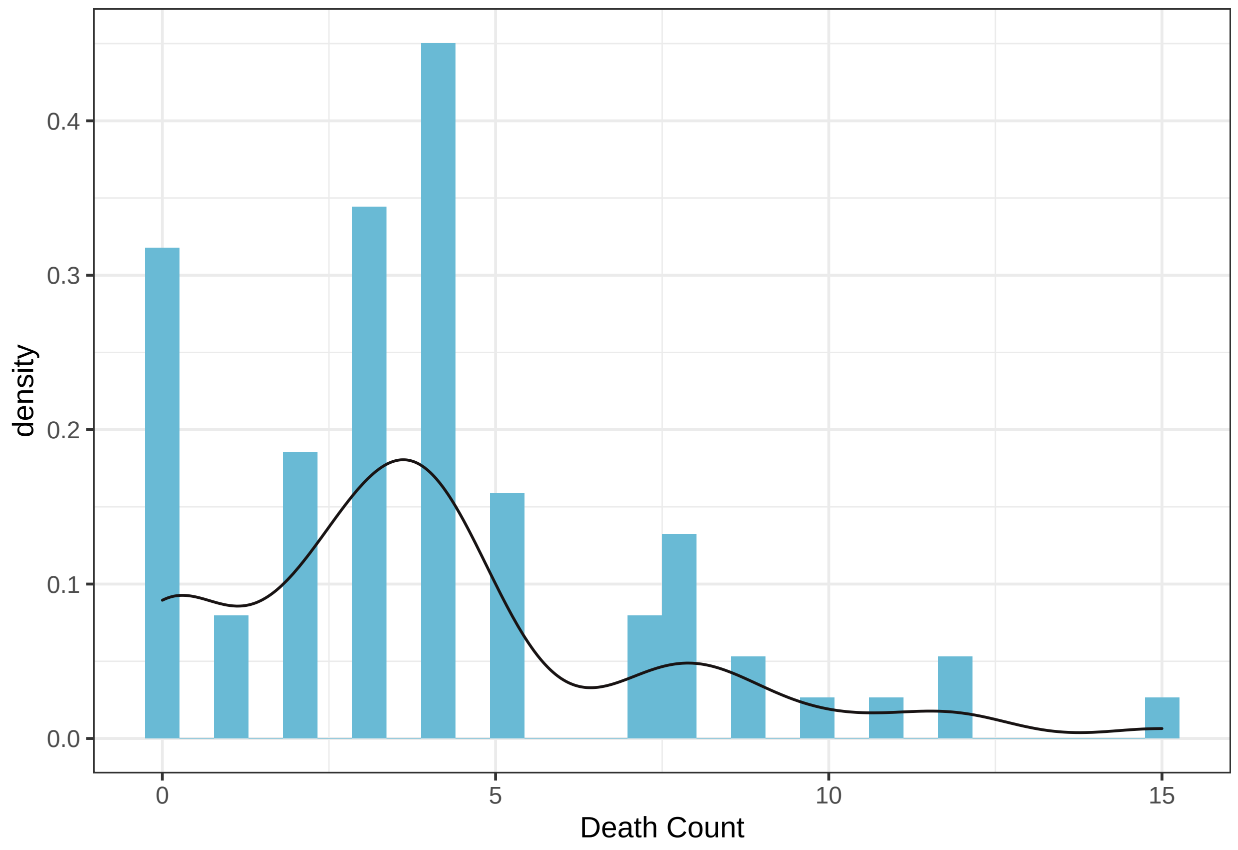
Student

19 September, 2022

Exercise 1

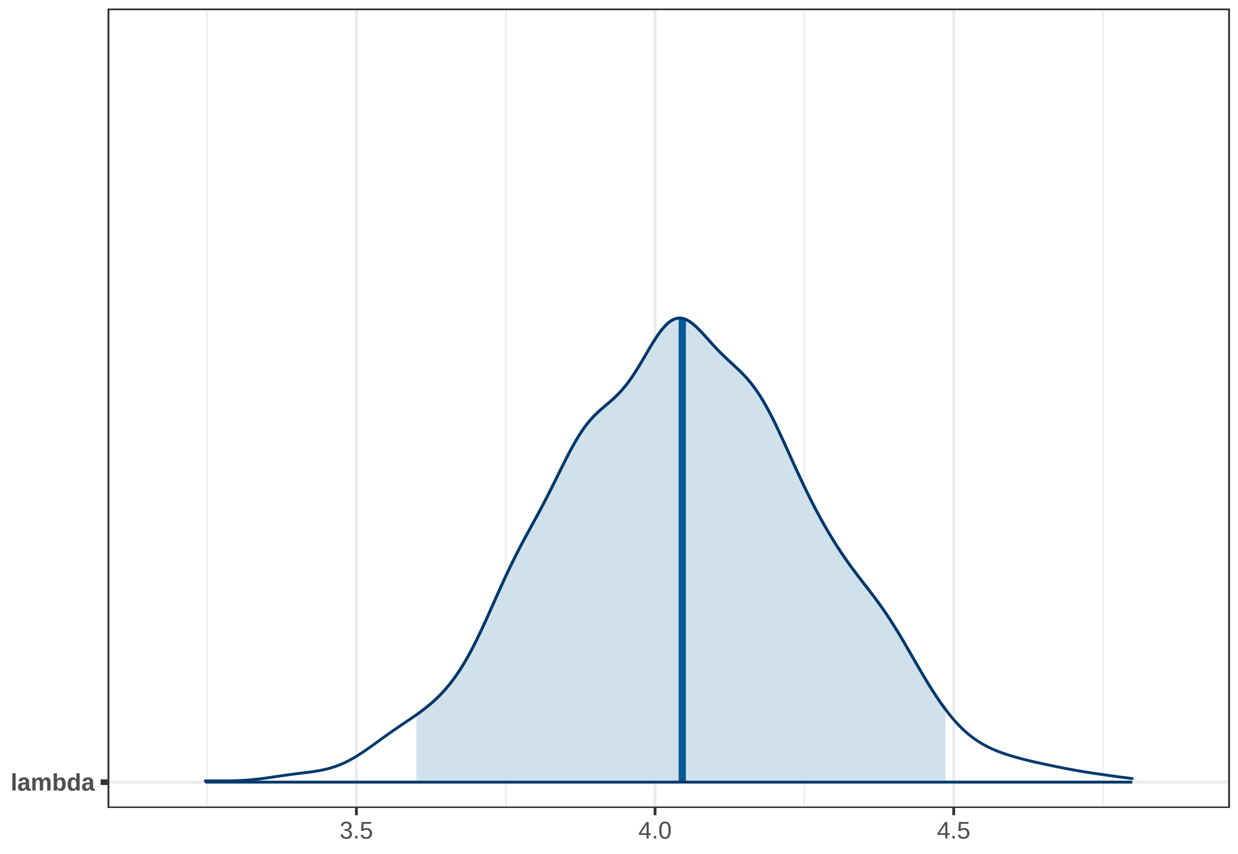
```r
ggplot(got, aes(x = Count)) +
  geom_histogram(aes(y=..density..), fill='#69bad5') +
  geom_density(alpha=.2, color="#191414") +
  xlab("Death Count")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Exercise 2

```r
lambda_draws <- as.matrix(fit, pars = "lambda")
mcmc_areas(lambda_draws, prob = 0.95)
```



```r
print(fit, pars = "lambda")
```

```
## Inference for Stan model: lab-03-poisson-simple.
## 2 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=2000.
##
##        mean se_mean   sd 2.5%  25%  50% 75% 97.5% n_eff Rhat
## lambda 4.05    0.01 0.23  3.6 3.89 4.05 4.2  4.49   789    1
##
## Samples were drawn using NUTS(diag_e) at Mon Sep 19 12:29:19 2022.
## 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).
```

Exercise 3

```r
set.seed(235983)

y_rep <- get_poisson_sample(n, lambda_draws)
head(y_rep)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
## [1,]   10    6    3    3    9    6    4    8    2     3     4     7     5     5
## [2,]    6    1    9    6    4    2    3    3    2     4     4     7     4     7
## [3,]    2    5    7    4    2    3    7    5    6     6     4     2     2     6
## [4,]    2    9    7    4    3    7    7    2    4     4     4     5     9     5
## [5,]    9    4    3    6    1    3    2    2    3     6     3     5     1     9
## [6,]    5    6    7    6    1    2    8    9    4     5     5     7     2     3
##       [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25] [,26]
## [1,]      1     7     3     1     4     5     5     4     8     4     6     0
## [2,]      5     0     5     4     6     2     6     3     8     1     4     5
## [3,]      2     6     6     4     4     6     9     4     5     8     7     6
## [4,]      1     4     6     5    10     5     5     2     2     2     4     3
## [5,]      9     3     0     2     6     6    10     6     4     6     5     5
## [6,]      3     1     5     4     2     7     4     6     6     5     2     5
##       [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35] [,36] [,37] [,38]
## [1,]      2     3     5     2     5     6     5     6     6     6     1     4
## [2,]      3     2     6     3     5     6     2     5     6     2     4     5
## [3,]      1     4     9     3     9     2     6     5     4     2     3     1
## [4,]      3     5     4     7     1     3     3     7     2     5     4     8
## [5,]      7     3     8    10     6     4     4     6     1     6     4     5
## [6,]      7     7     4     7     5     4     6     3     6     3     4    10
##       [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46] [,47] [,48] [,49] [,50]
## [1,]      1     1     6     3     4     1     3     5     0     4     4     6
## [2,]      4     0     3     1     4     5     4     3     2     5     3     7
## [3,]      1     4     2     5     5     2     2     6     5     5     4     4
## [4,]      3     7     5     3     5     4     4     2     3     3     3     8
## [5,]      4     6     3     6     6     8     5     8     5     4     1     6
## [6,]      5     2     4     2     4    11     5    10     3     3     5     4
##       [,51] [,52] [,53] [,54] [,55] [,56] [,57] [,58] [,59] [,60] [,61] [,62]
## [1,]      6     0     1    10     2     8     5     1     0     5     6     5
## [2,]      4     4     2     4     5     6     5     5     6     7     7     7
## [3,]      8     4     4     2     3     6     5     2     3     2     2     7
## [4,]      2     7     2     2     5     5     2     7     2     4     4     3
## [5,]      6     2     5     6     2     5     5     3     4     5     4     3
## [6,]      3     8     4     5     2     5     3     6     3     6     4     4
##       [,63] [,64] [,65] [,66] [,67] [,68] [,69] [,70] [,71] [,72] [,73]
## [1,]      5     2     2     7     7     7     5     5     5     3     6
## [2,]      6     6     5     7     4     8     6     3     8     4     7
## [3,]      5     4     3     7     4     3     5     4     4     4     5
## [4,]      6     6     2     4     5     2     9     6     4     1     5
## [5,]      3     6     5     5     4     3     6     1     5     4     3
## [6,]      3     3     5     4     4     5     2     9     5     5     5
```

Exercise 4

We have only a reasonable estimate of the mean using the naive Poisson generative model. However, the original data has lots of zeros death that are hard for the naive Poisson distribution to capture. Hence, we see a significant difference in the density comparison plot at 0. I will not say that the naive model is wrong, but it certainly can be made better to account for the bimodality in the death count.

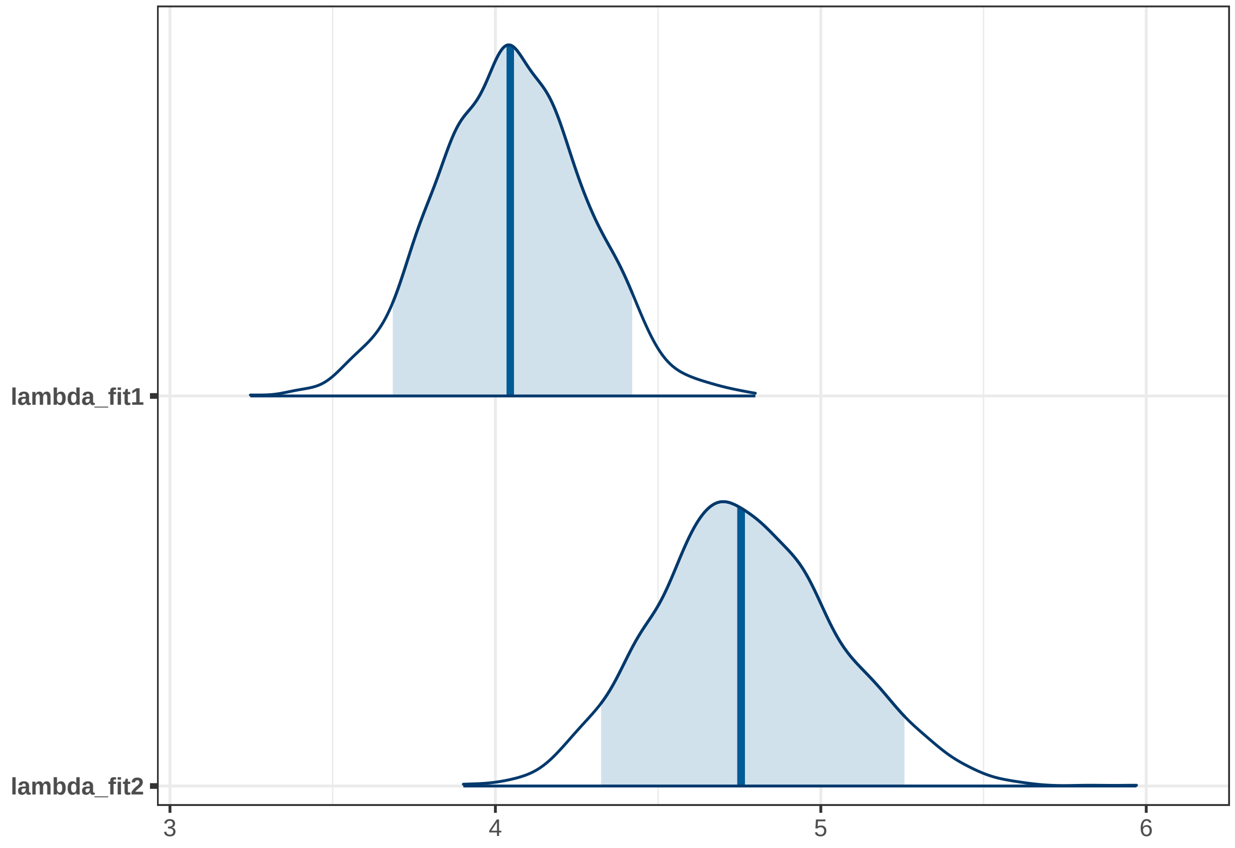
Exercise 5

```r
# setting up the stan model
fit2 <- stan(
  "lab-03-poisson-hurdle.stan",
  data = stan_dat, refresh = 0, chains = 2
)
```

Exercise 6

```
# compare lambda posterial
lambda_draws2 <- as.matrix(fit2, pars = "lambda")

mcmc_areas(
  cbind(lambda_fit1 = lambda_draws[, 1], lambda_fit2 = lambda_draws2[, 1]),
  prob = 0.9
)
```

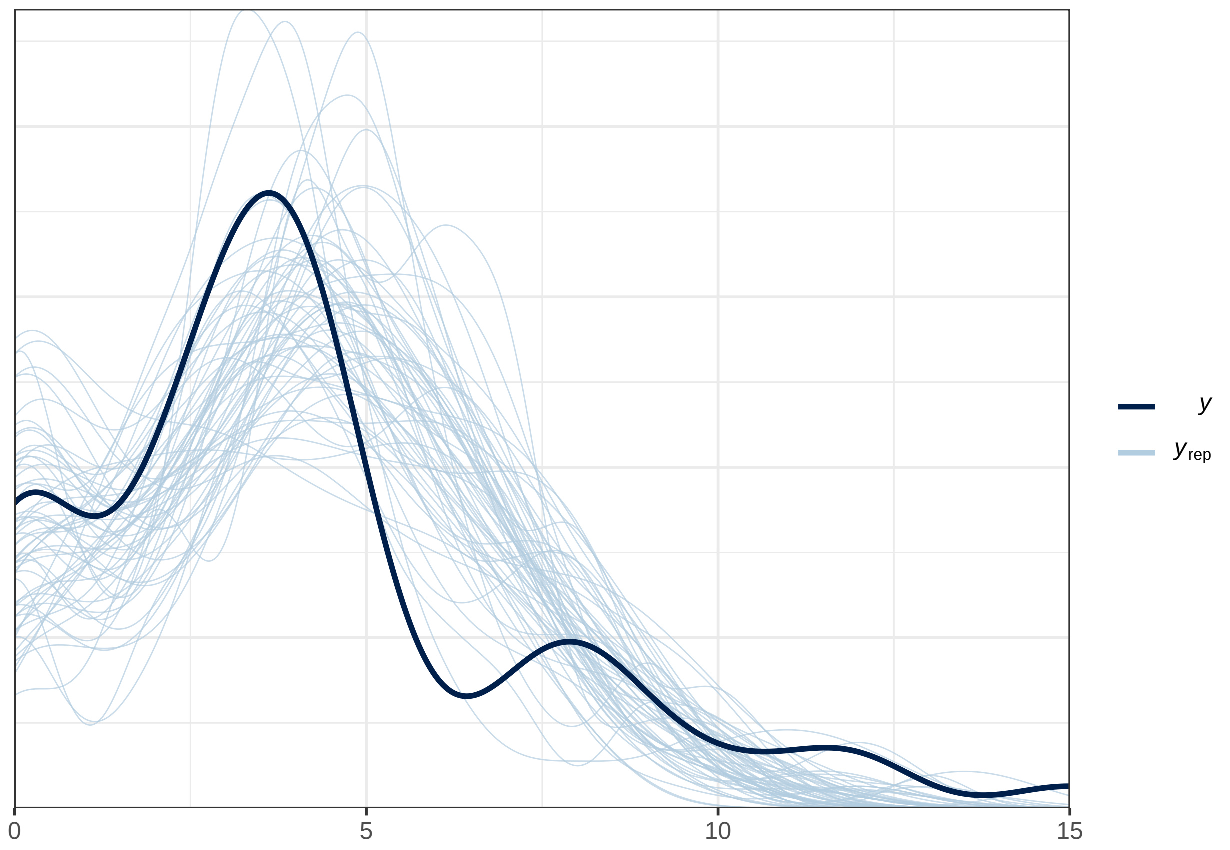


```
set.seed(235983)

y_rep2 <- as.matrix(fit2, pars='y_rep')
head(y_rep2)
```

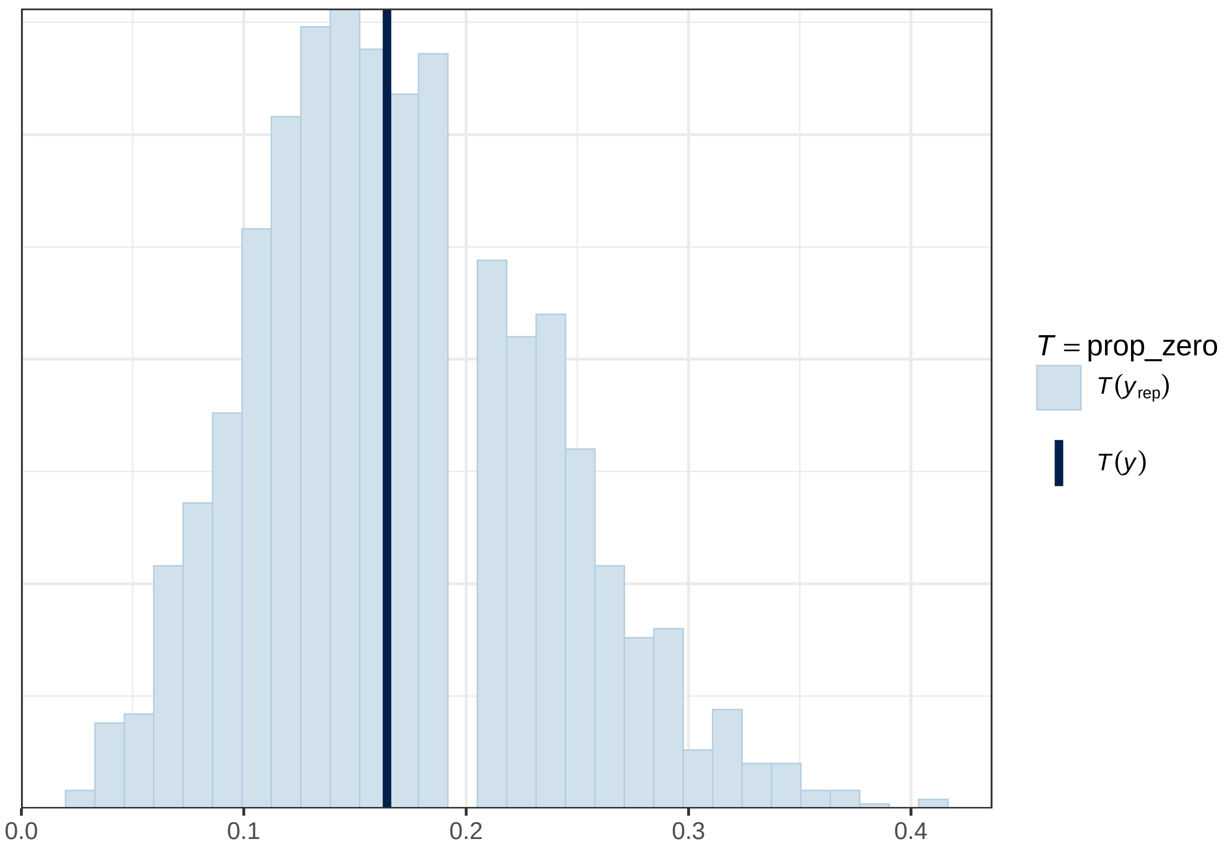
```
##           parameters
## iterations y_rep[1] y_rep[2] y_rep[3] y_rep[4] y_rep[5] y_rep[6] y_rep[7]
##       [1,]        5        6        1       14        4        2        1
##       [2,]        0        8        4        8        0        3        0
##       [3,]        4        8        6        5        0        3        0
##       [4,]        0        2        0        5        4        6        9
##       [5,]        7        6        5        0        6        4        6
##       [6,]        3        4        0        0        4        0        4
##           parameters
## iterations y_rep[8] y_rep[9] y_rep[10] y_rep[11] y_rep[12] y_rep[13] y_rep[14]
##       [1,]        6        9         8         7         8         8         4
##       [2,]       10        5         3         3         4         0         2
##       [3,]        2        0         9         3         2         8         0
##       [4,]       0        4        0        3        4        0        0
##       [5,]       4        5        5        6        7        5        9
##       [6,]       5        7        0        2        4        2        9
##           parameters
## iterations y_rep[15] y_rep[16] y_rep[17] y_rep[18] y_rep[19] y_rep[20]
##       [1,]         3         4         5         2         4         5
##       [2,]         8         0         5         5         0         0
##       [3,]         6         8         0         4         7         5
##       [4,]         4         3         0         8         0         0
##       [5,]         4         0         0         0         2         2
##       [6,]         5         6         4         6         3         0
##           parameters
## iterations y_rep[21] y_rep[22] y_rep[23] y_rep[24] y_rep[25] y_rep[26]
##       [1,]         5         6         1         4         6         0
##       [2,]         4         4         4         7         5         5
##       [3,]         5         0         2         0         3         4
##       [4,]         7         8         2         6         7         3
##       [5,]         4         2         6         8         4         7
##       [6,]         4        11         8         2         2         6
##           parameters
## iterations y_rep[27] y_rep[28] y_rep[29] y_rep[30] y_rep[31] y_rep[32]
##       [1,]         3         9         5         8         0         5
##       [2,]         4         3         6         3         3         3
##       [3,]         6         4         3         2         8         0
##       [4,]         6         0         2         9         4         0
##       [5,]         7         8        10         8         6         4
##       [6,]         9         3         7         3         4         0
##           parameters
## iterations y_rep[33] y_rep[34] y_rep[35] y_rep[36] y_rep[37] y_rep[38]
##       [1,]         3         7         6         6         6         2
##       [2,]         6         5         0         3         0         2
##       [3,]         0         9         2         0         6         0
##       [4,]         8         7         0         3         1         6
##       [5,]         4         8         0         6         8         5
##       [6,]         4         0         4         7         0         8
##           parameters
## iterations y_rep[39] y_rep[40] y_rep[41] y_rep[42] y_rep[43] y_rep[44]
##       [1,]         5         3         3         7         5         5
##       [2,]         0         0         9         0         0         9
##       [3,]         3         4         0         5         4         2
##       [4,]         0         4         0         5         0         7
##       [5,]         0         0         4         8         9         6
##       [6,]         9         0         5         2         3         3
##           parameters
## iterations y_rep[45] y_rep[46] y_rep[47] y_rep[48] y_rep[49] y_rep[50]
##       [1,]         3         7         6         4         6        13
##       [2,]         2         6         5         5         5         5
##       [3,]         6         3         4         8         4         6
##       [4,]         2         6         0         0         5         5
##       [5,]         0         0         1         6         7         0
##       [6,]         5         6         9         5         3         9
##           parameters
## iterations y_rep[51] y_rep[52] y_rep[53] y_rep[54] y_rep[55] y_rep[56]
##       [1,]         3         9         2         1         3         2
##        [2,]          4          4          5          2          1          1
##        [3,]          5          3          4          4          5          2
##        [4,]          3          0          4          0          4          3
##        [5,]          0          4          0          2          8          8
##        [6,]          7          0          4          8          3          4
##           parameters
## iterations y_rep[57] y_rep[58] y_rep[59] y_rep[60] y_rep[61] y_rep[62]
##        [1,]          4          4          7          5          0          4
##        [2,]          2          4          6          5          3          5
##        [3,]          0          0          2          4          1          5
##        [4,]          7          4          0          1          4          6
##        [5,]          3          7          4          4          3          6
##        [6,]          3          0          0          8          0          6
##           parameters
## iterations y_rep[63] y_rep[64] y_rep[65] y_rep[66] y_rep[67] y_rep[68]
##        [1,]          6          5          1          5          7          2
##        [2,]          4          1          4          7          0          4
##        [3,]          9          4          4          0          7          0
##        [4,]          4          1          0          3          1          8
##        [5,]          3          9          4          0          5          0
##        [6,]          3          3          4          8         10          4
##           parameters
## iterations y_rep[69] y_rep[70] y_rep[71] y_rep[72] y_rep[73]
##        [1,]          8          1          4          4          7
##        [2,]          5          9          6          7          4
##        [3,]          4          5          3          1          5
##        [4,]          0          0          9          0          2
##        [5,]          4          0          5          0          3
##        [6,]          8          5          2          6         10
```

```
ppc_dens_overlay(y, y_rep2[1:60, ])
```

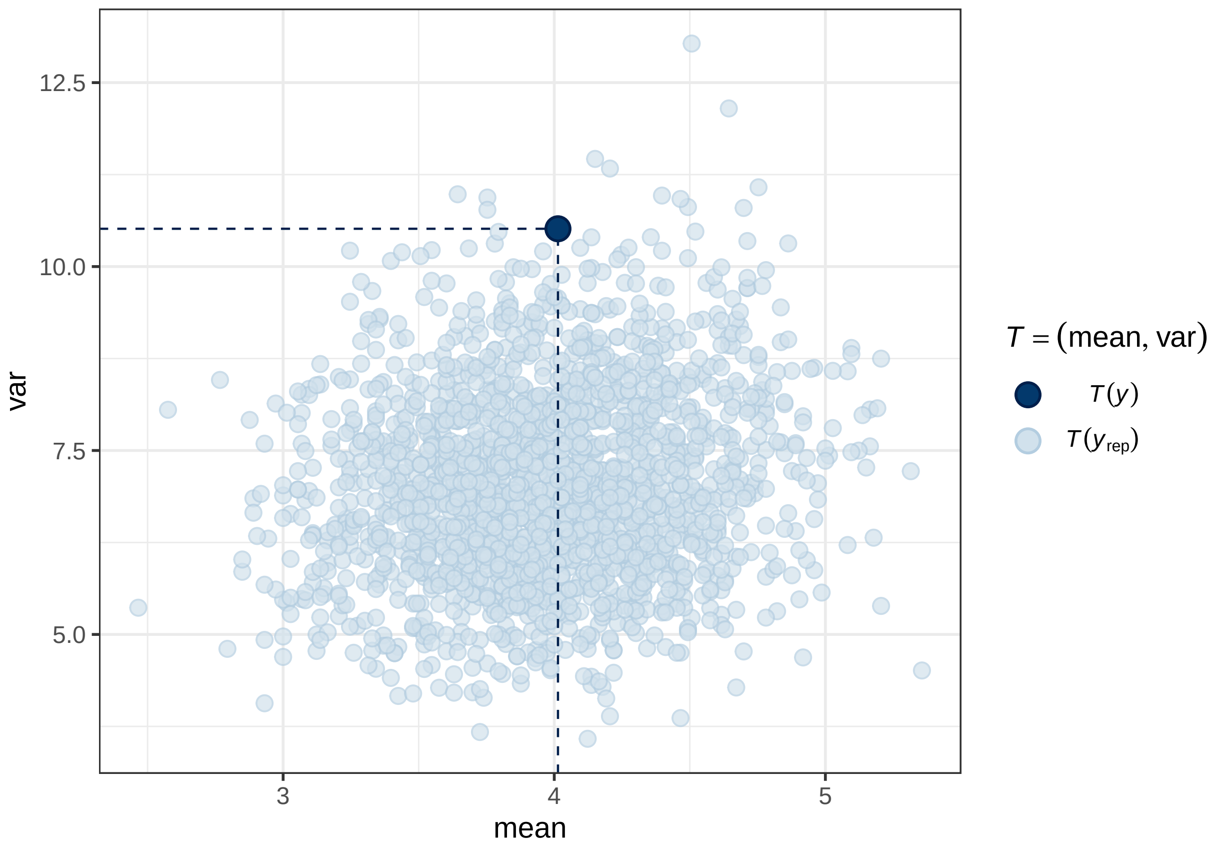


```
ppc_stat(y, y_rep2, stat = "prop_zero")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

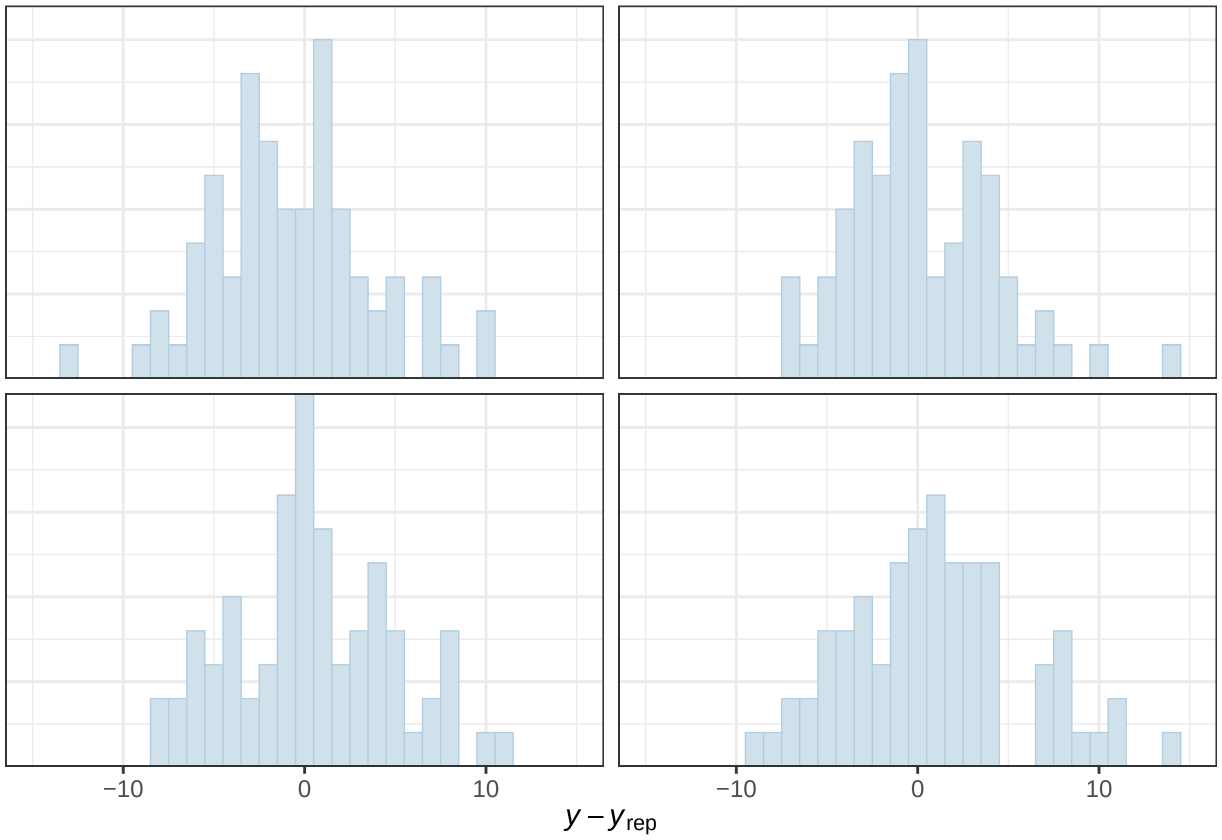


```r
ppc_stat_2d(y, y_rep2, stat = c("mean", "var"))
```



```
ppc_error_hist(y, y_rep2[1:4, ], binwidth = 1) + xlim(-15, 15)
```

```
## Warning: Removed 8 rows containing missing values (geom_bar).
```



Exercise 7

We can see that the new Poisson hurdle model performs much better than the original naive Poisson model.

```r
loo1 <- loo(fit)
loo2 <- loo(fit2)

loo_compare(loo1, loo2)
```

```
##        elpd_diff se_diff
## model2   0.0       0.0  
## model1 -19.8       8.5  
```

Excercise 8

PPC is important, of course. Modeling is a process establishing assumptions and testing those assumptions. One directly way of checking if our assumptions are accurate is to compare the data resulting from the generative model and the observed data.

Excercise 9

The Poisson hurdle model fits quite well, although it can be better. With the hurdle model, we do a good job modeling the frequent zero-death observations. However, looking at the mean-variance 2-D comparison, we can still see that the actual data has a higher variance than our model. It can mean the Poisson assumption might not be the best fit because it assumes the data has equal mean and variance.

Excercise 10

A single LOOCV error is still useful. At least I can get a sense of the prediction error of the model in interest.
