

# More Checking Models

Bret Larget

Departments of Botany and of Statistics  
University of Wisconsin—Madison

April 8, 2008

## Checking the Probability of 0

- The Poisson and binomial distributions both are suitable for count data.
- In some situations, the proportion of observed zeros can be larger than the model predicts.
- One solution is to use a model with overdispersion.
- An alternative is to use logistic regression to model presence/absence and then a different model to model the counts when present.
- Here we will examine the use of simulation to check the goodness of fit of real data to data simulated under a Poisson model.
- We will see if using a model with overdispersion is sufficient when a Poisson model fails.

## Case Study:

- The *Dusky Sound* is a large fjord in southwest New Zealand noted for abundant wildlife.
- Biologists conducted a study to examine the relationship between the seaweed *Ecklonia radiata* and the sea urchin *Evechinus chloroticus* in the seas of the fjord.
- The goal is to predict seaweed abundance from urchin abundance and other variables.
- Other predictors include the distance to the mouth of the fjord (in km), the distance to the nearest point of land (in km), and the *fetch*, a sum of distances in km to the nearest land along several radial directions.
- Higher *fetch* values are associated with windier conditions and greater waves.
- There are 103 separate sample locations.
- The number of seaweed and urchins are counted in a 25 m by 1m area.

```
> sw = read.table("seaweed.txt", header = T)
> str(sw)
```

```
'data.frame':      103 obs. of  5 variables:
 $ seaweed: int  0 0 0 2 2 1 1 0 0 3 ...
 $ urchin  : int  5 9 1 1 0 3 2 5 11 2 ...
 $ fjord   : num  11.5 13 9.7 9.6 7.3 9.1 7 7.5 5.6 10.2 ...
 $ land    : num  0.6 0.7 1 0.7 0.8 1 0.5 1.1 0.8 0.2 ...
 $ fetch   : num  28.1 39.7 39.4 33.5 17.4 29.4 27.5 38.9 33.6 36.2 ...
```

# Data Summaries

```
> summary(sw)
```

| seaweed        | urchin         | fjord         | land           |
|----------------|----------------|---------------|----------------|
| Min. : 0.000   | Min. : 0.000   | Min. : 1.30   | Min. :0.2000   |
| 1st Qu.: 0.000 | 1st Qu.: 1.000 | 1st Qu.: 7.30 | 1st Qu.:0.6000 |
| Median : 0.000 | Median : 2.000 | Median : 9.70 | Median :0.7000 |
| Mean : 4.194   | Mean : 2.748   | Mean : 9.58   | Mean :0.7058   |
| 3rd Qu.: 2.000 | 3rd Qu.: 4.000 | 3rd Qu.:12.00 | 3rd Qu.:0.8000 |
| Max. :50.000   | Max. :11.000   | Max. :18.40   | Max. :1.2000   |

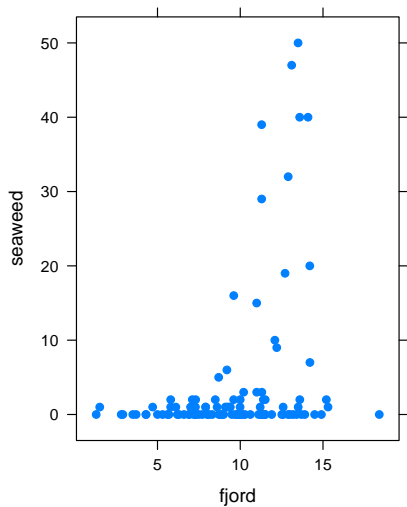
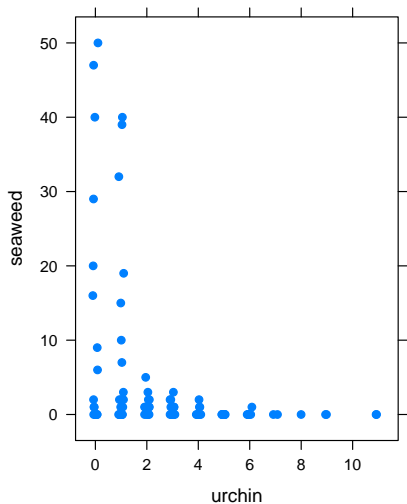
  

| fetch         |
|---------------|
| Min. :17.40   |
| 1st Qu.:28.50 |
| Median :32.30 |
| Mean :32.30   |
| 3rd Qu.:36.05 |
| Max. :48.30   |

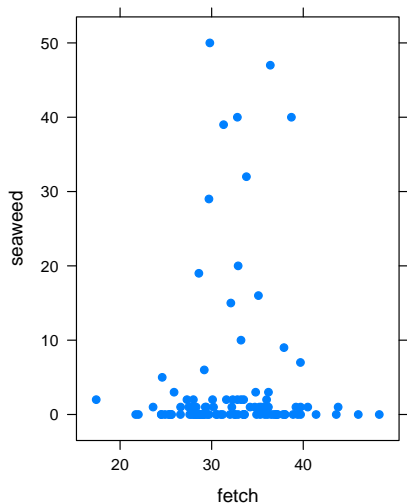
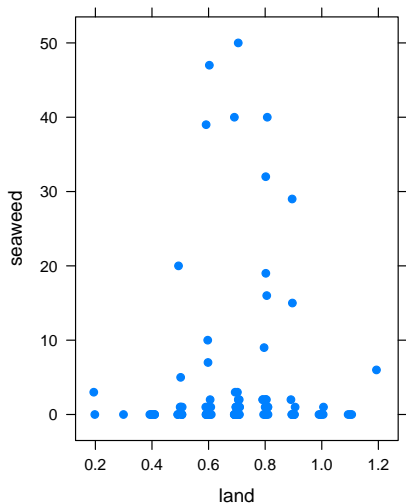
## Figure commands

```
> library(lattice)
> fig1 = xyplot(seaweed ~ urchin, data = sw, jitter.x = T,
+   pch = 16)
> fig2 = xyplot(seaweed ~ fjord, data = sw, jitter.x = T,
+   pch = 16)
> fig3 = xyplot(seaweed ~ land, data = sw, jitter.x = T,
+   pch = 16)
> fig4 = xyplot(seaweed ~ fetch, data = sw, jitter.x = T,
+   pch = 16)
> print(fig1, split = c(1, 1, 2, 1))
> print(fig2, split = c(2, 1, 2, 1), new = F)
> print(fig3, split = c(1, 1, 2, 1))
> print(fig4, split = c(2, 1, 2, 1), new = F)
```

# Scatter plots



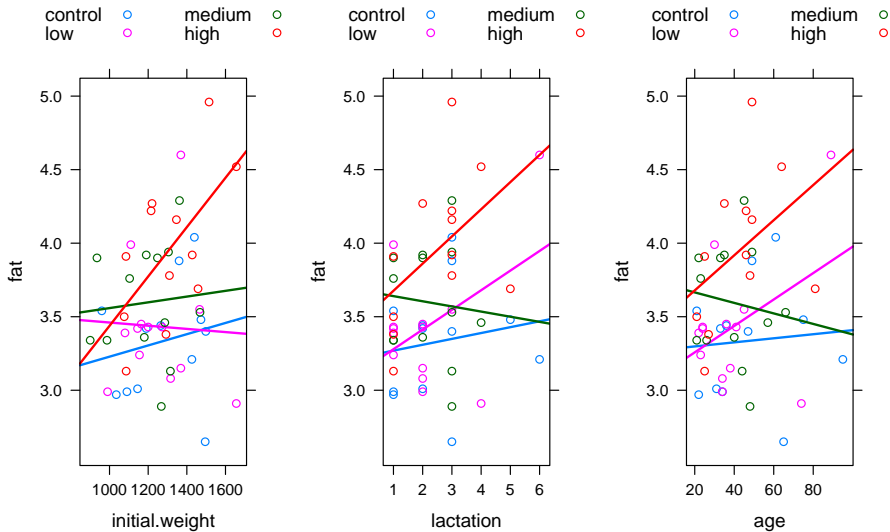
# Scatter plots



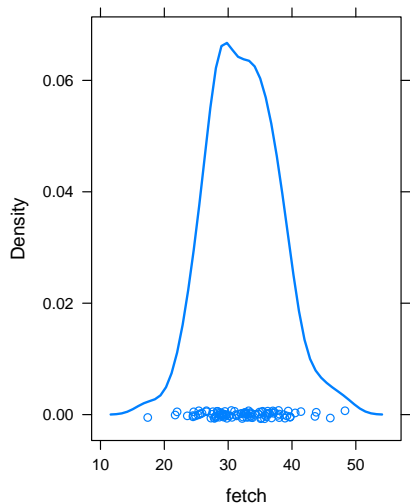
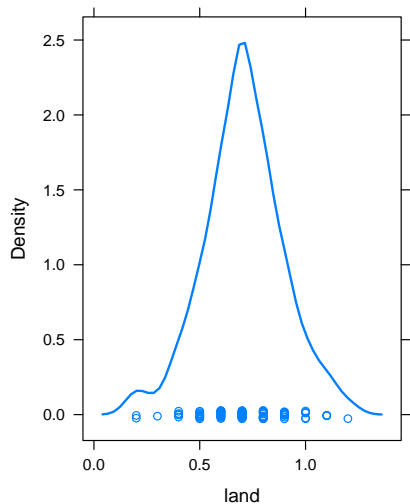
## More Figure Commands

```
> fig5 = histogram(~urchin, data = sw)
> fig6 = densityplot(~fjord, data = sw, lwd = 2)
> fig7 = densityplot(~land, , data = sw, lwd = 2)
> fig8 = densityplot(~fetch, data = sw, lwd = 2)
> print(fig5, split = c(1, 1, 2, 1))
> print(fig6, split = c(2, 1, 2, 1), new = F)
> print(fig7, split = c(1, 1, 2, 1))
> print(fig8, split = c(2, 1, 2, 1), new = F)
```

# Plots



# Plots

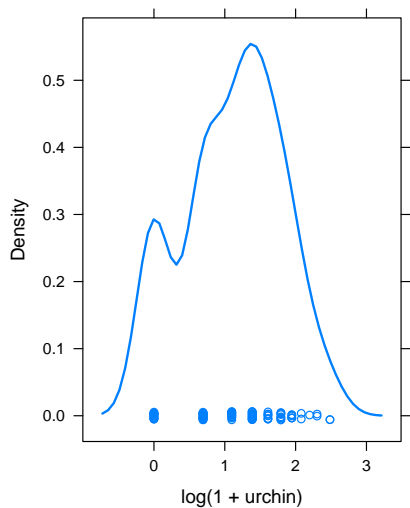
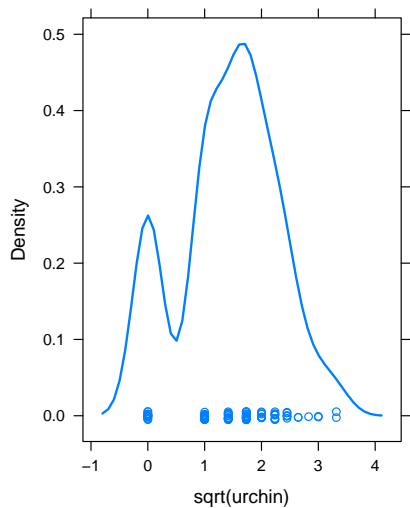


# Transformations

- Consider transformations to the urchin variable to decrease the effect of the skewness.
- As there are zeros, we cannot take logarithms directly.
- Square root is a possibility.
- We can also try adding one and then taking logs.
- See the graphs and that both are more symmetric.
- We will use the  $\log(1 + x)$  transformation to follow the authors.

```
> fig9 = densityplot(~sqrt(urchin), data = sw, lwd = 2)
> fig10 = densityplot(~log(1 + urchin), data = sw, lwd = 2)
> print(fig9, split = c(1, 1, 2, 1))
> print(fig10, split = c(2, 1, 2, 1), new = F)
```

# Plots



# Fitting a Poisson Regression

```
> fit1.glm = glm(seaweed ~ log(1 + urchin) + fjord + land +
+   fetch, data = sw, family = poisson)
> display(fit1.glm)

glm(formula = seaweed ~ log(1 + urchin) + fjord + land + fetch,
     family = poisson, data = sw)

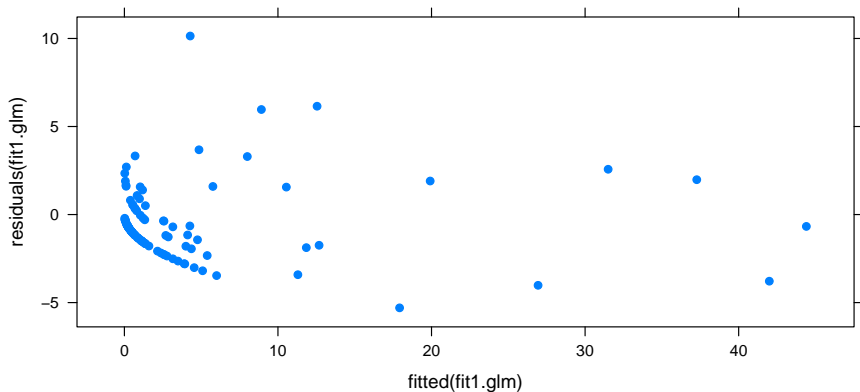
```

|                 | coef.est | coef.se |
|-----------------|----------|---------|
| (Intercept)     | -1.89    | 0.46    |
| log(1 + urchin) | -1.91    | 0.09    |
| fjord           | 0.35     | 0.02    |
| land            | 0.75     | 0.30    |
| fetch           | 0.01     | 0.01    |

```
---
n = 103, k = 5
residual deviance = 491.8, null deviance = 1407.1 (difference = 915.3)
```

# Residual Plot

```
> print(xyplot(residuals(fit1.glm) ~ fitted(fit1.glm),  
+           pch = 16))
```



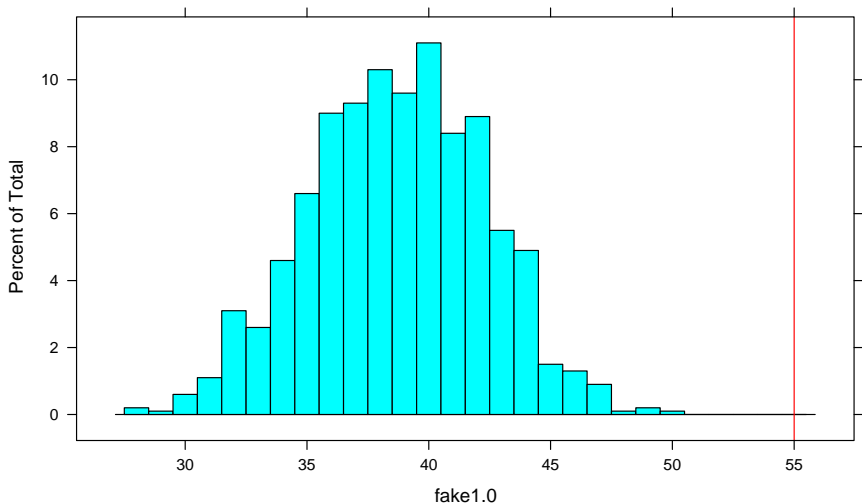
## Checking 0 probability

- Biological count data often has an excess of zeros relative to what a standard model predicts.
- We can examine this by simulation and compare the number of zeros in simulated data to the number in the real data.
- We can do this by generating Poisson random variables with means as the fitted values.

## Checking 0 probability

```
> count0 = function(x) {  
+   return(sum(x == 0))  
+ }  
> with(sw, count0(seaweed))  
  
[1] 55  
  
> n = nrow(sw)  
> mu = fitted(fit1.glm)  
> fake1.0 = replicate(1000, count0(rpois(n, mu)))
```

# Checking 0 probability



- Notice that the actual data set has 55 zeros, but the fake simulated data sets never have more than 50 zeros.
- We can try to fit a model with an overdispersion parameter.
- An alternative would be to fit a logistic regression model for the zeros and fit a different model for the positive outcomes.
- We can use simulation to see if the overdispersed-Poisson model produces as many zeros as we observe in the real data.

# Negative Binomial Distribution

- An example of the overdispersed-Poisson model is the *negative binomial distribution*.
- We can parameterize this distribution with a parameter  $\mu$  which is the mean and  $\alpha$  which is a shape parameter.
- In R we estimate  $\mu$  with the fitted values and  $\alpha = \mu / (w - 1)$  where  $w$  is the overdispersion parameter.
- The function `rnbinom()` generates random variables with arguments `mu` and `size` for  $\alpha$ .

# Fitting an Over-dispersed Poisson Regression

```
> fit2.glm = glm(seaweed ~ log(1 + urchin) + fjord + land +  
+ fetch, data = sw, family = quasipoisson)  
> display(fit2.glm)
```

```
glm(formula = seaweed ~ log(1 + urchin) + fjord + land + fetch,  
     family = quasipoisson, data = sw)
```

|                 | coef.est | coef.se |
|-----------------|----------|---------|
| (Intercept)     | -1.89    | 1.27    |
| log(1 + urchin) | -1.91    | 0.25    |
| fjord           | 0.35     | 0.06    |
| land            | 0.75     | 0.81    |
| fetch           | 0.01     | 0.03    |

---

n = 103, k = 5

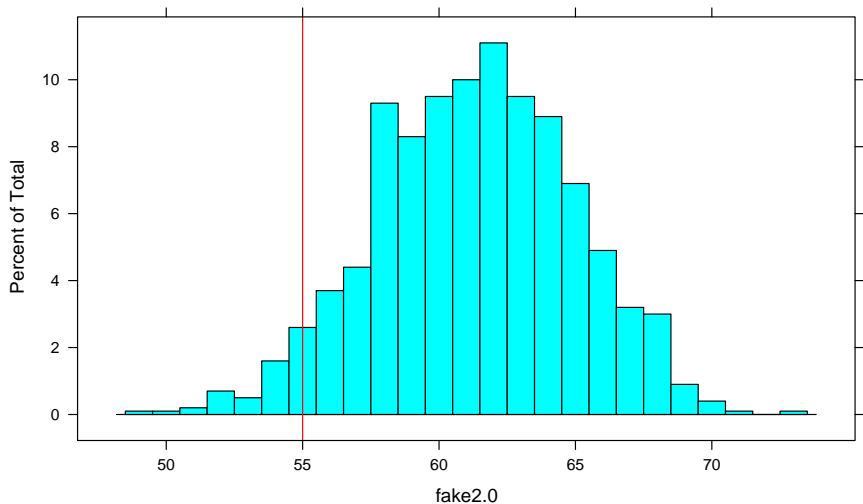
residual deviance = 491.8, null deviance = 1407.1 (difference = 915.3)

overdispersion parameter = 7.4

## Checking 0 probability

```
> with(sw, count0(seaweed))  
  
[1] 55  
  
> n = nrow(sw)  
> mu = fitted(fit2.glm)  
> alpha = mu/(7.4 - 1)  
> fake2.0 = replicate(1000, count0(rnbinom(n, mu = mu,  
+   size = alpha)))
```

# Checking 0 probability



- The actual number of zeros, 55, is fairly typical for this distribution.
- We could also test other aspects of the data to see if the simulated data is similar to the actual data.