Poisson regression

"A family of alternative regression models that is more appropriate for outcome variables with low count"

Count data

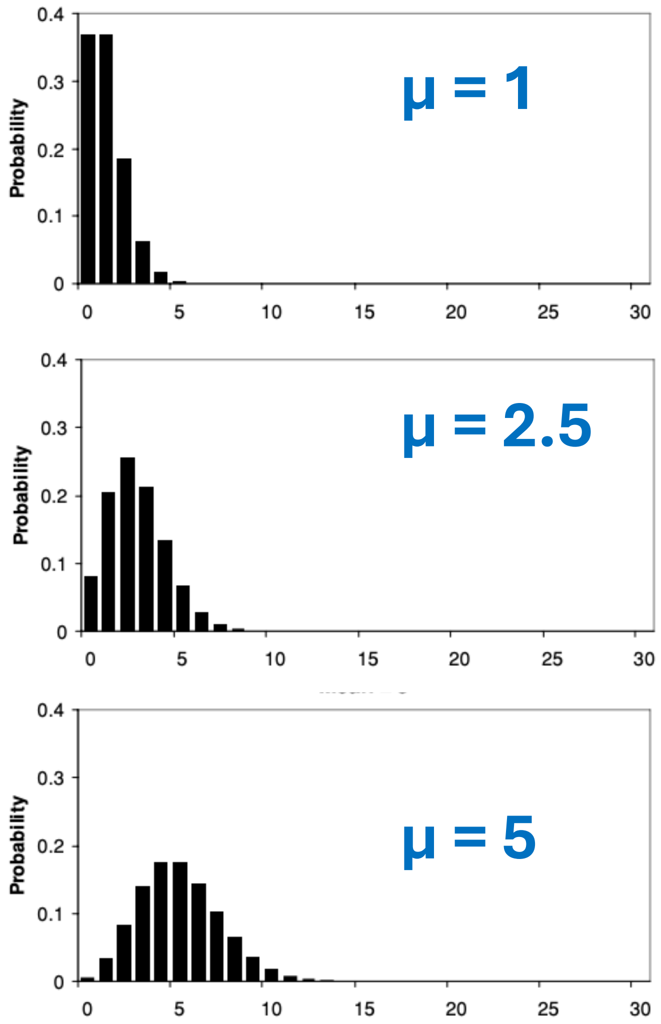
DV that takes on discrete, non-negative values (0, 1, 2, 3, 4, 5…)

Measured during a fixed period of time

Low arithmetic mean (typically <10)

Coxe et al. (2009)

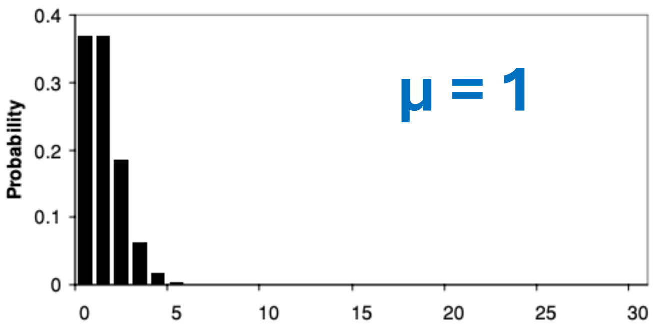
Poisson distributions

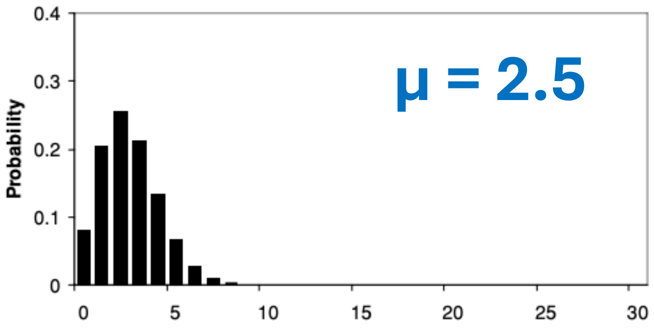


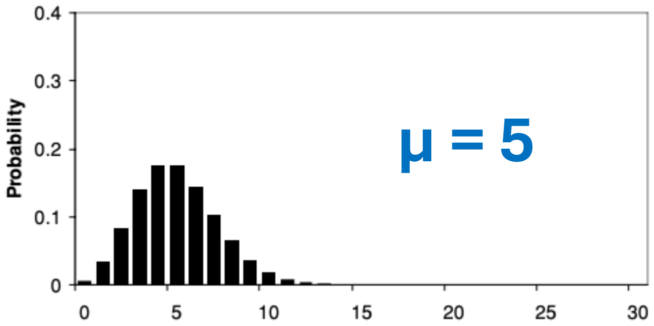
Characterized by a single parameter λ, which defines both the mean (μ) and variance

Coxe et al. (2009)

Poisson distributions

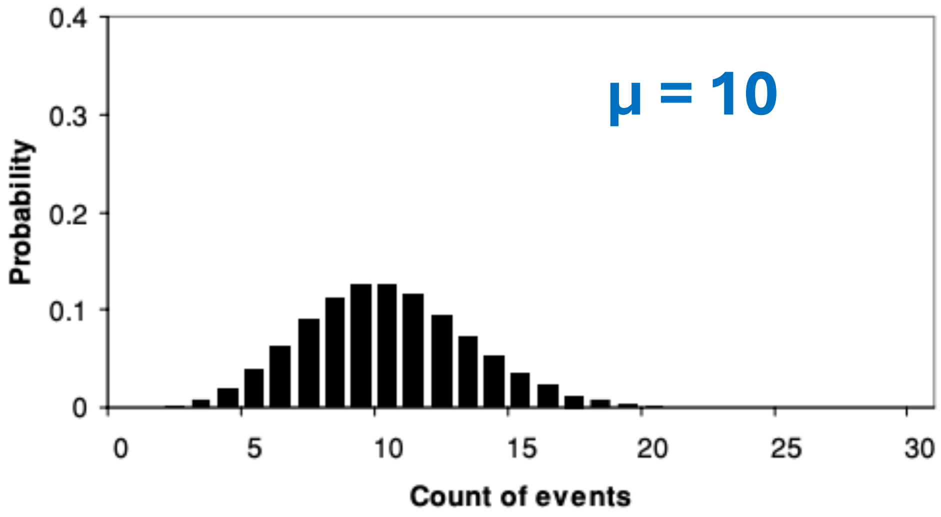






Characterized by a single parameter λ, which defines <u>both</u> the mean and variance

When μ > 10, the Poisson distribution approximates the normal distribution

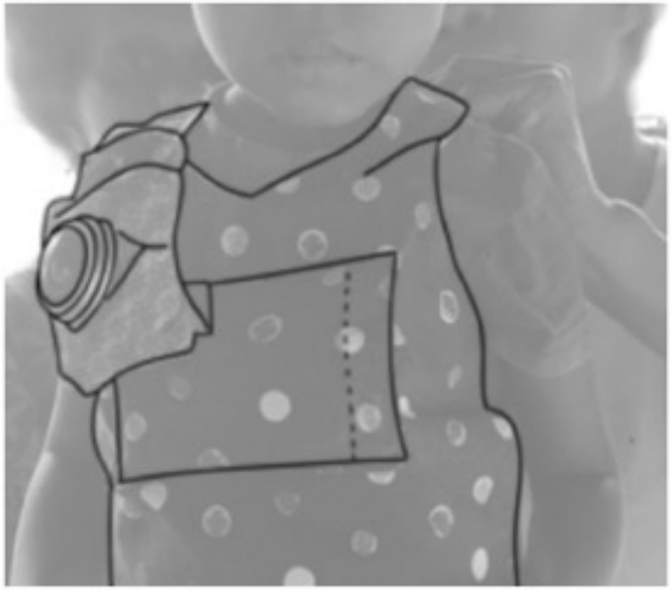


Coxe et al. (2009)

Example: Naturalistic object handling frequency

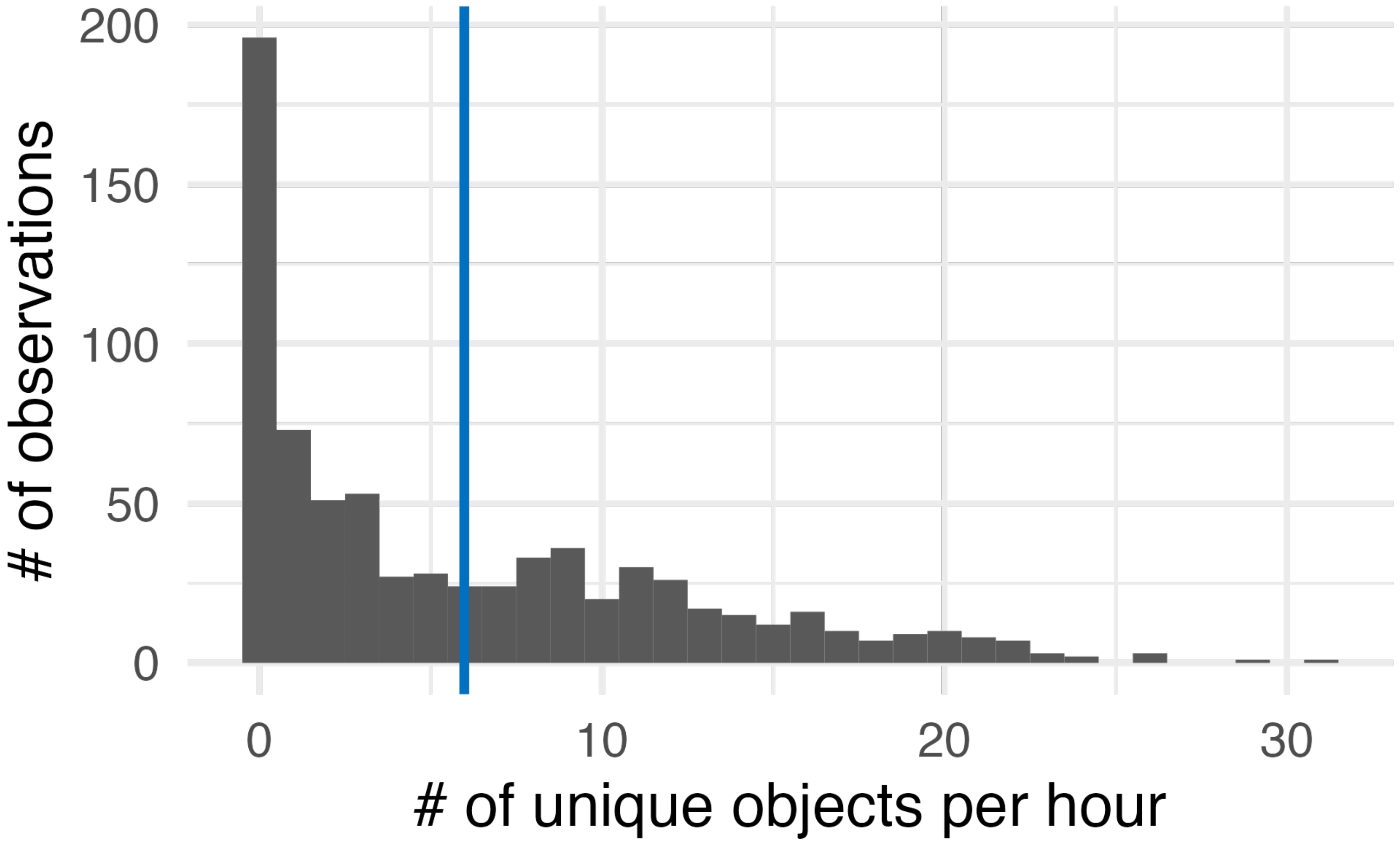
DV: count of unique objects handled per hour

IVs: age, cultural context/site, sex



Example: Naturalistic object handling frequency

DV: count of unique objects handled per hour (μ = 5.98)



Why OLS regression doesn't work

Count variable as IV:

- If variance is low, then coefficient estimates are unstable and have high SEs

Count variable as DV:

- Can return negative λ values (predicted mean counts) which don't make sense
- Biased SEs and significance tests
- Violations of linear model assumptions…

Coxe et al. (2009)

Two key assumptions of OLS error structure often violated by count data

data = model + error

```
m_ols <- lm(n_objects ~ age*site + sex, data = data)
```

$$\hat{e}_i = Y_i - \hat{Y}_i$$

(1) Normally distributed errors
(2) Homoskedasticity of errors

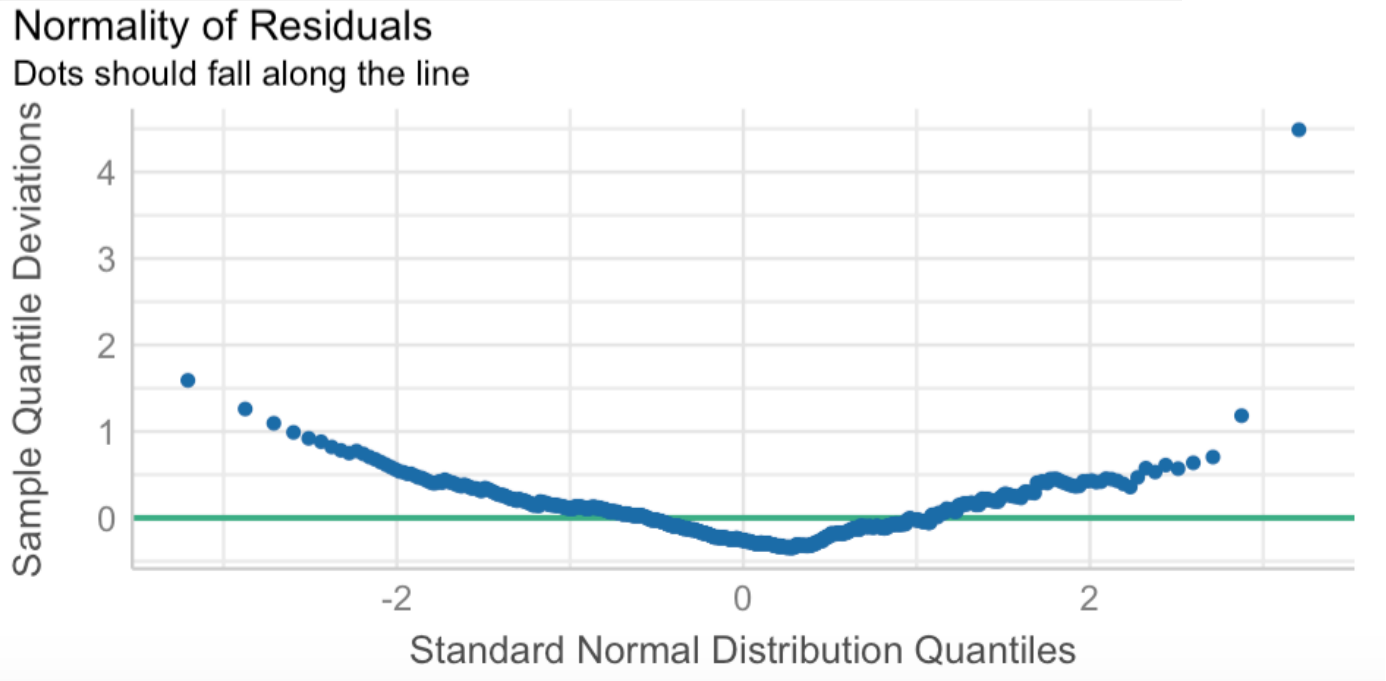
Two key assumptions of OLS error structure often violated by count data

(1) Normally distributed errors

```
performance::check_normality(m_ols)
```

```
Warning: Non-normality of residuals detected (p < .001).
```

```
performance::check_model(m_ols, check = c("qq"))
```



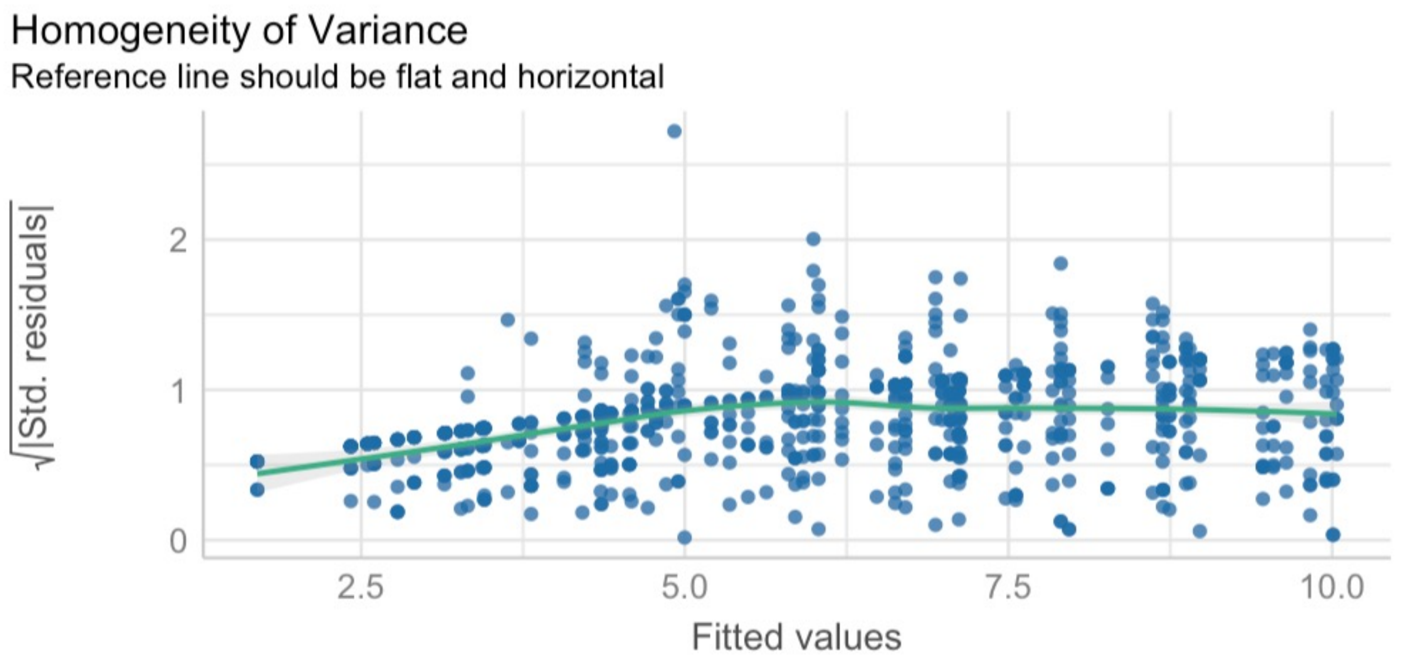
Two key assumptions of OLS error structure often violated by count data

(2) Homoskedasticity of errors (constant error variance)

```
performance::check_homogeneity(m_ols)
```

```
Warning: Variances differ between groups
(Bartlett Test, p = 0.000).
```

```
performance::check_model(m_ols, check = c("homogeneity"))
```



Poisson regression

$$\log(\hat{\mu}) = b_0 + b_1X_1 + b_2X_2 + \cdots + b_pX_p$$

where:

$\hat{\mu}$ is the predicted mean count

b_0 is the log of the predicted mean count when all predictors are 0 (if dummy coded/not centered) or at their mean (if deviation coded/centered)

b_p is the change in the log of the predicted count for each one-unit change in predictor X_p holding all other predictors constant

```r
m_poisson <- glm(n_objects ~ age*site + sex,
                 family = poisson(link = "log"),
                 data = data)

summary(m_poisson)
```

```
Coefficients:
              Estimate Std. Error z value              Pr(>|z|)    
(Intercept)   1.719951   0.016107 106.780 < 0.0000000000000002 ***
age           0.027605   0.001148  24.046 < 0.0000000000000002 ***
site          0.185340   0.032364   5.727          0.0000000102 ***
sex          -0.074721   0.030846  -2.422               0.01542 *  
age:site     -0.007414   0.002265  -3.273               0.00106 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 5290.5  on 741  degrees of freedom
Residual deviance: 4667.3  on 737  degrees of freedom
AIC: 6646.3

Number of Fisher Scoring iterations: 6
```

Poisson regression

$$\log(\hat{\mu}) = b_0 + b_1X_1 + b_2X_2 + \cdots + b_pX_p$$

where:

$\hat{\mu}$ is the predicted mean count

b_0 is the log of the predicted mean count when all predictors are 0 (if dummy coded/not centered) or at their mean (if deviation coded/centered)

b_p is the change in the log of the predicted count for each one-unit change in predictor X_p holding all other predictors constant

b_{age}: For each one-unit increase in age, there is a 0.03 unit increase in the log of the # of objects handled per hour

```
m_poisson <- glm(n_objects ~ age*site + sex,
                 family = poisson(link = "log"),
                 data = data)

summary(m_poisson)
```

```
Coefficients:
             Estimate Std. Error z value             Pr(>|z|)
(Intercept)  1.719951   0.016107 106.780 < 0.0000000000000002 ***
age          0.027605   0.001148  24.046 < 0.0000000000000002 ***
site         0.185340   0.032364   5.727         0.0000000102 ***
sex         -0.074721   0.030846  -2.422              0.01542 *
age:site    -0.007414   0.002265  -3.273              0.00106 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 5290.5  on 741  degrees of freedom
Residual deviance: 4667.3  on 737  degrees of freedom
AIC: 6646.3

Number of Fisher Scoring iterations: 6
```

Poisson regression

$$\log(\hat{\mu}) = b_0 + b_1X_1 + b_2X_2 + \dots + b_pX_p$$

where:

$\hat{\mu}$ is the predicted mean count

b_0 is the log of the predicted mean count when all predictors are 0 (if dummy coded/not centered) or at their mean (if deviation coded/centered)

b_p is the change in the log of the predicted count for each one-unit change in predictor X_p holding all other predictors constant

or exponentiate both sides of the equation to interpret in original units (i.e., count):

$$\hat{\mu} = \exp(b_0 + b_1X_1 + b_2X_2 + \dots + b_pX_p)$$

incidence rate ratio (IRR)

```
m_poisson <- glm(n_objects ~ age*site + sex,
                 family = poisson(link = "log"),
                 data = data)

tidy(m_poisson, exponentiate = TRUE) %>%
  kable(digits = 3, format = "markdown")
```

term	estimate	std.error	statistic	p.value
(Intercept)	5.584	0.016	106.780	0.000
age	1.028	0.001	24.046	0.000
site	1.204	0.032	5.727	0.000
sex	0.928	0.031	-2.422	0.015
age:site	0.993	0.002	-3.273	0.001

Poisson regression

$$\log(\hat{\mu}) = b_0 + b_1X_1 + b_2X_2 + \cdots + b_pX_p$$

where:

$\hat{\mu}$ is the predicted mean count

b_0 is the log of the predicted mean count when all predictors are 0 (if dummy coded/not centered) or at their mean (if deviation coded/centered)

b_p is the change in the log of the predicted count for each one-unit change in predictor X_p holding all other predictors constant

b_{age}: For each one-unit increase in age, the # of objects handled per hour increases by a rate of 1.03

incidence rate ratio (IRR) > 1

```
m_poisson <- glm(n_objects ~ age*site + sex,
                 family = poisson(link = "log"),
                 data = data)

tidy(m_poisson, exponentiate = TRUE) %>%
  kable(digits = 3, format = "markdown")
```

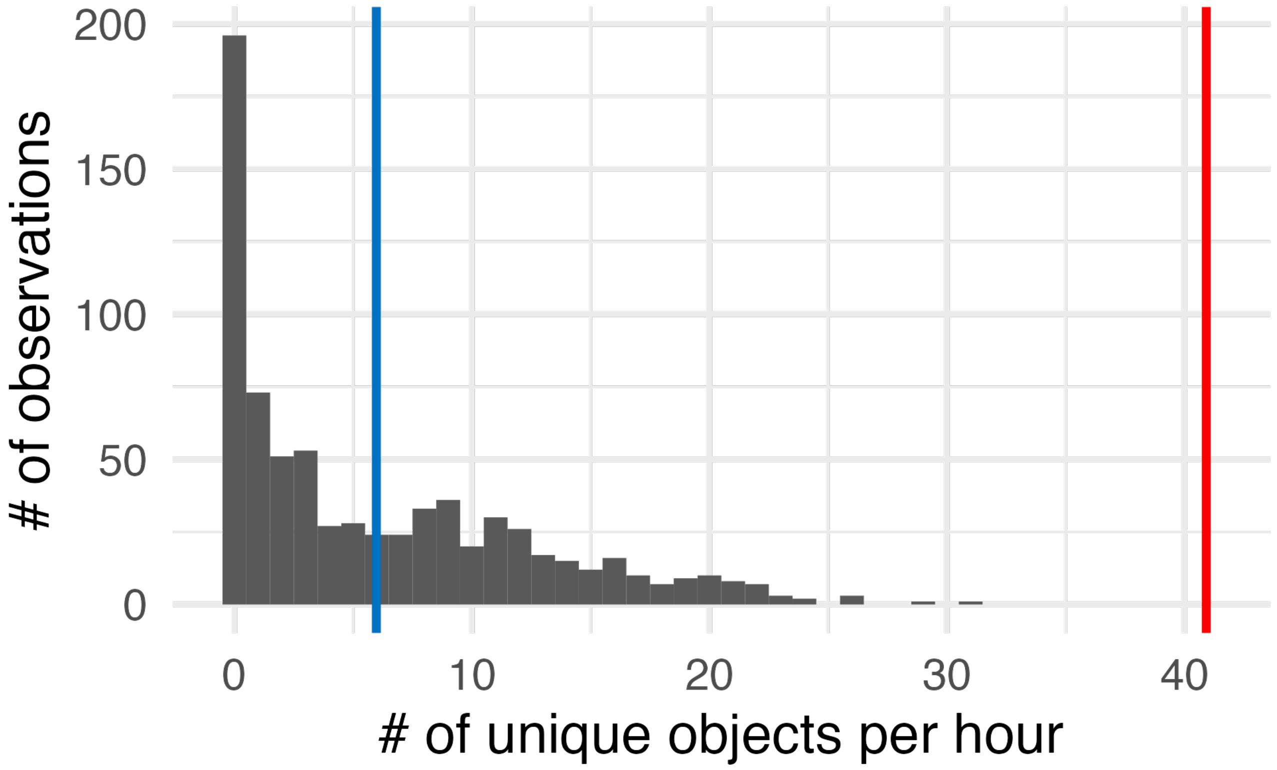
term	estimate	std.error	statistic	p.value
(Intercept)	5.584	0.016	106.780	0.000
age	1.028	0.001	24.046	0.000
site	1.204	0.032	5.727	0.000
sex	0.928	0.031	-2.422	0.015
age:site	0.993	0.002	-3.273	0.001

Two common problems

Overdispersion: more variability in counts than expected

Zero inflation: more zero counts than expected

Overdispersion (variance > mean)



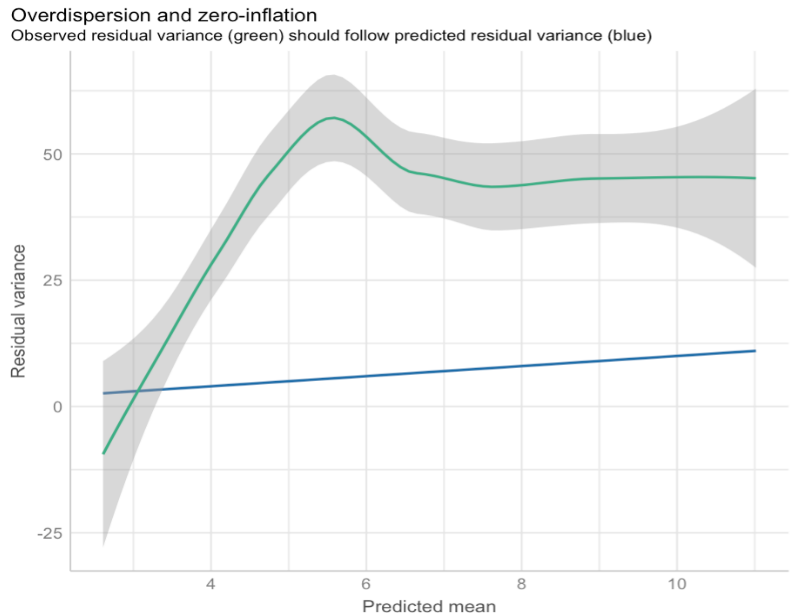

```
performance::check_overdispersion(m_poisson)
```

```
# Overdispersion test

       dispersion ratio =    6.214
  Pearson's Chi-Squared = 4579.988
                p-value =  < 0.001

Overdispersion detected.
```

```
performance::check_model(m_poisson, check = c("overdispersion"))
```



Overdispersion (variance > mean)

Commonly occurs when:

(1) An important predictor is not included in the model

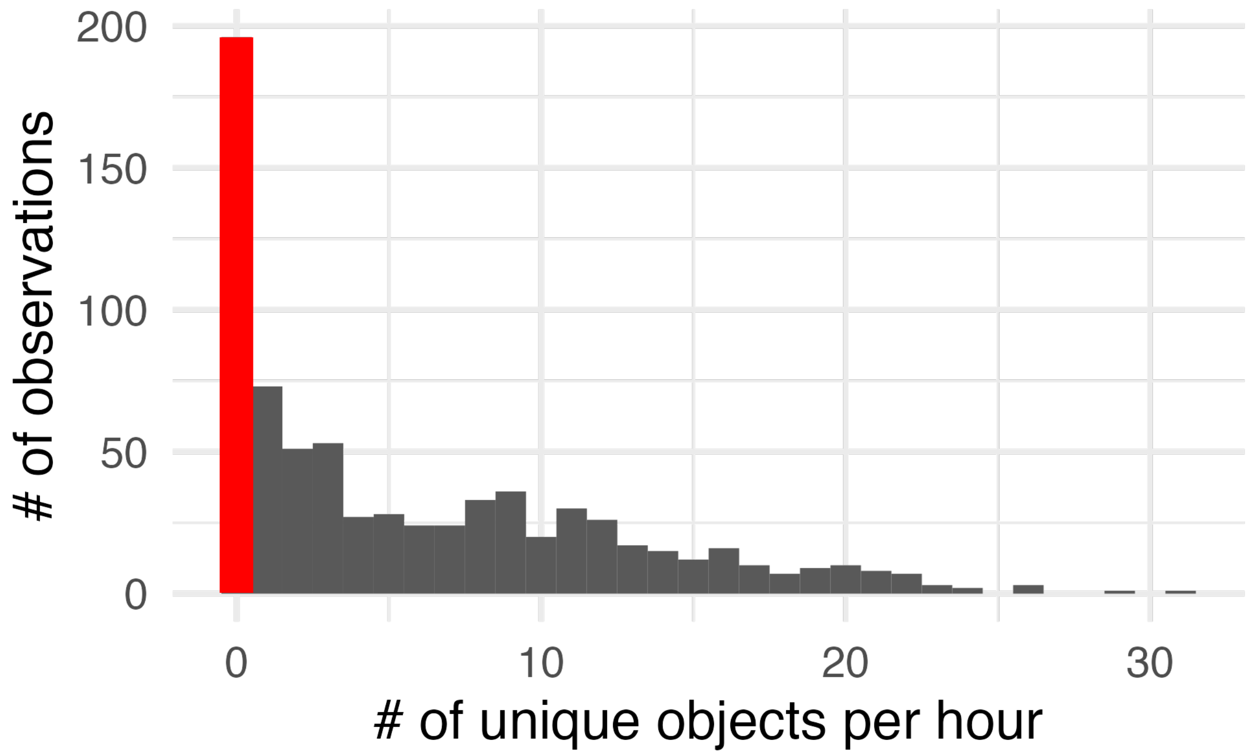
(2) Observations are not independent (i.e., contagion/state dependence)

How to deal with this:

(1) Overdispersed Poisson regression model that includes a dispersion parameter, ϕ

(2) Negative binomial regression model that accounts for variability among individuals who have the same predicted value (variance is a quadratic function of the mean)

Zero inflation (structural zeroes ↑ positive skew)




```
performance::check_zeroinflation(m_poisson)
```

```
# Check for zero-inflation

   Observed zeros: 196
  Predicted zeros: 9
            Ratio: 0.05

Model is underfitting zeros (probable zero-inflation).
```

Zero inflation (structural zeroes ↑ positive skew)

Commonly occurs when:

(1) Structural zeroes are not anticipated in original study design

alcoholic drinks per week
non-drinkers vs. drinkers

How to deal with this:

(1) Eliminate zero counts (ideally beforehand by excluding certain groups as needed)

(2) Zero inflated Poisson model

(3) Zero inflated negative binomial model

Dealing with multiple Poisson assumption violations

Overdispersion: more variability in counts than expected

Zero inflation: more zero counts than expected*

State dependence: non-independent observations*

```r
best_model <- glmmTMB(n_objects ~ age*site + sex + (1|child),
                      data = data,
                      ziformula = ~age*site+sex,
                      family = nbinom2)
```

```
Dispersion parameter for nbinom2 family (): 3.18

Conditional model:
               Estimate Std. Error z value              Pr(>|z|)
(Intercept)    1.654968   0.158146  10.465 < 0.0000000000000002 ***
age            0.038671   0.009571   4.040            0.0000534 ***
siteTseltal   -0.216756   0.183825  -1.179                0.238
sexM           0.153828   0.183978   0.836                0.403
age:siteTseltal 0.013103  0.013748   0.953                0.341
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Zero-inflation model:
            Estimate Std. Error z value              Pr(>|z|)
(Intercept) -1.62033    0.13753 -11.782 <0.0000000000000002 ***
age         -0.01617    0.01126  -1.437               0.151
site         0.00598    0.25998   0.023               0.982
sex          0.25438    0.25703   0.990               0.322
age:site     0.00917    0.02189   0.419               0.675
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Dealing with multiple Poisson assumption violations

Overdispersion: more variability in counts than expected
Zero inflation: more zero counts than expected*
State dependence: non-independent observations*

```
best_model <- glmmTMB(n_objects ~ age*site + sex + (1|child),
                       data = data,
                       ziformula = ~age*site+sex,
                       family = nbinom2)
```

```
ggemmeans(best_model, terms=c("age", "sex", "site")) %>%
  plot() +
  labs(x = "Age (scaled)", y = "Predicted counts of handled objects per hour",
       color = "Sex", fill = "Sex", title = "")
```