Categorical data and regression

Frank Edwards

Count data

- Counts are cumulative totals of the number of incidences of some event, generally across time or place
- Counts are positive integers $\in [0, \infty]$
- We can generally express counts as rates by dividing by the desired exposure variable (e.g. population, time, subject)

Counts as extensions of binary data

- Counts can be thought of as repeated binary trials
- $\sum y_i$ where y is equal to 1 or 0 provides a count
- Generally, we could treat `sum(y==1) + sum(y==0)` or `nrow(y)` as the exposure, or denominator for a rate. Why?

Let's return to the Titanic

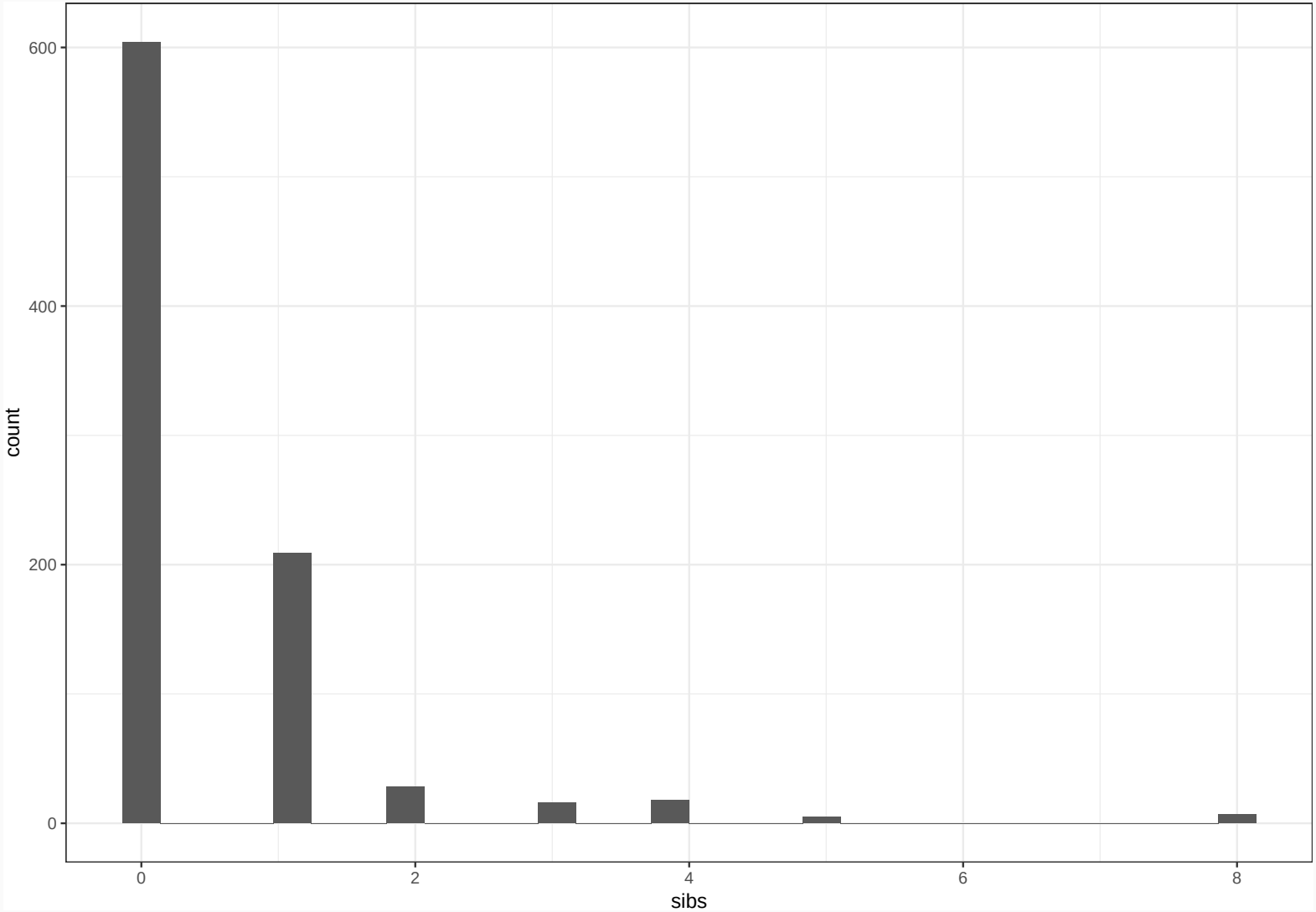
```
titanic <- read_csv("./data/titanic.csv") %>% select(-Name) %>% rename_all(tolower) %>%
    rename(sibs = `siblings/spouses aboard`, kids = `parents/children aboard`)

head(titanic)
```

```
## # A tibble: 6 x 7
##   survived pclass sex      age  sibs  kids  fare
##      <dbl>  <dbl> <chr>  <dbl> <dbl> <dbl> <dbl>
## 1        0      3 male      22     1     0  7.25
## 2        1      1 female    38     1     0 71.3
## 3        1      3 female    26     0     0  7.92
## 4        1      1 female    35     1     0 53.1
## 5        0      3 male      35     0     0  8.05
## 6        0      3 male      27     0     0  8.46
```

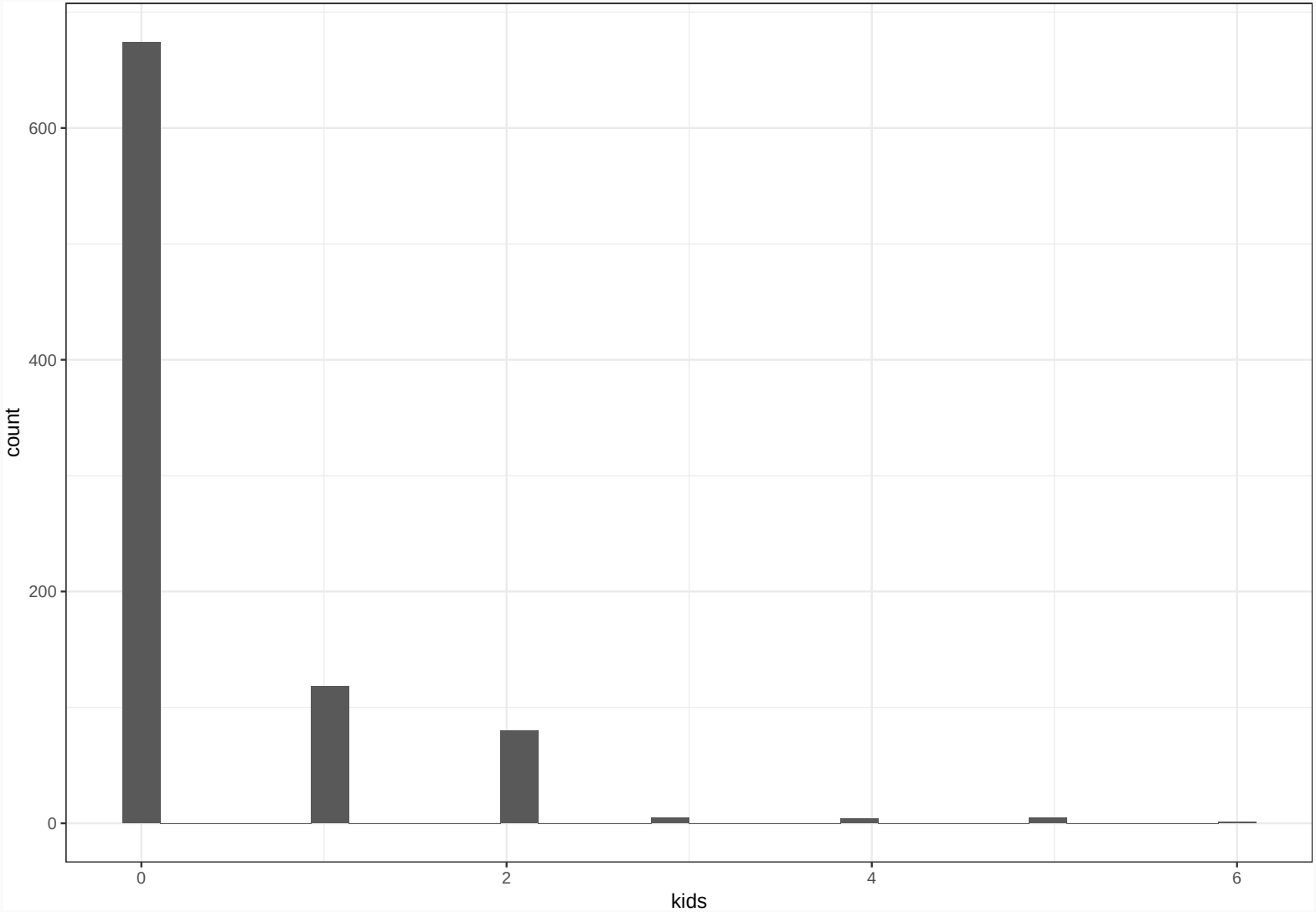
Exploring the count variables

```
ggplot(titanic, aes(x = sibs)) + geom_histogram()
```



Exploring the count variables

```
ggplot(titanic, aes(x = kids)) + geom_histogram()
```



How can we model kids?

- Perhaps the number of kids is a function of age and sex? Seems reasonable

$$kids_i = \beta_0 + \beta_1 age_i + \beta_2 sex_i + \varepsilon_i$$

$$\varepsilon \sim Normal(0, \sigma^2)$$

Estimating the model

```
m0 <- lm(kids ~ sex + age, data = titanic)
m0_out <- tidy(m0)
m0_out
```

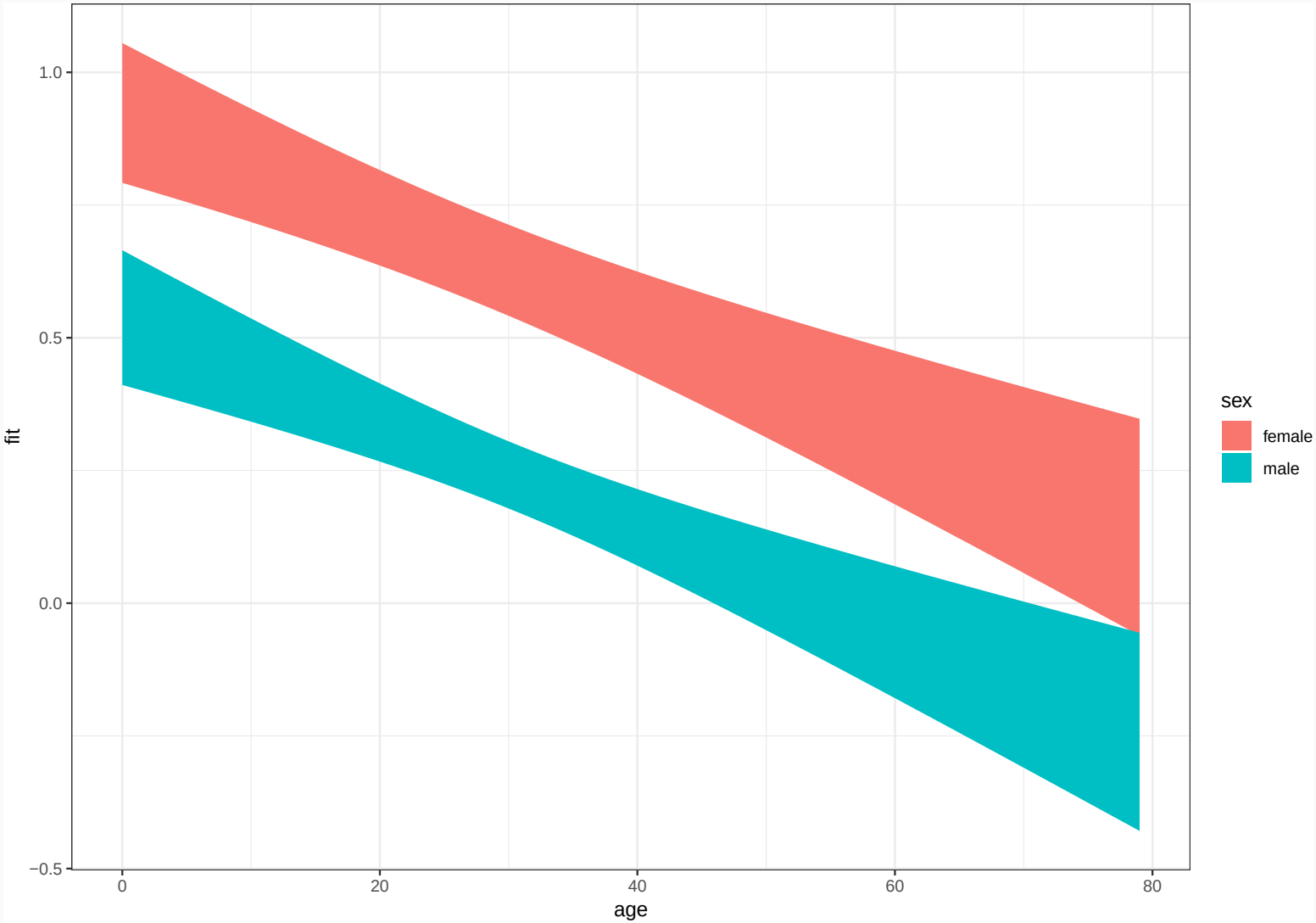
```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)  0.923     0.0671      13.8  3.61e-39
## 2 sexmale     -0.386     0.0544      -7.09 2.78e-12
## 3 age         -0.00988   0.00184     -5.36 1.07e- 7
```

Does the model fit the data? Make prediction data

```r
age <- rep(0:79, 2)
sex <- rep(c("male", "female"), each = 80)
newdata <- data.frame(age = age, sex = sex)
yhat <- predict(m0, newdata = newdata, interval = "confidence")
yhat <- as.data.frame(yhat)
yhat$sex <- sex
yhat$age <- age
```

Check the fit of the fake data

```
ggplot(yhat, aes(x = age, y = fit, ymin = lwr, ymax = upr, fill = sex)) + geom_ribbon()
```

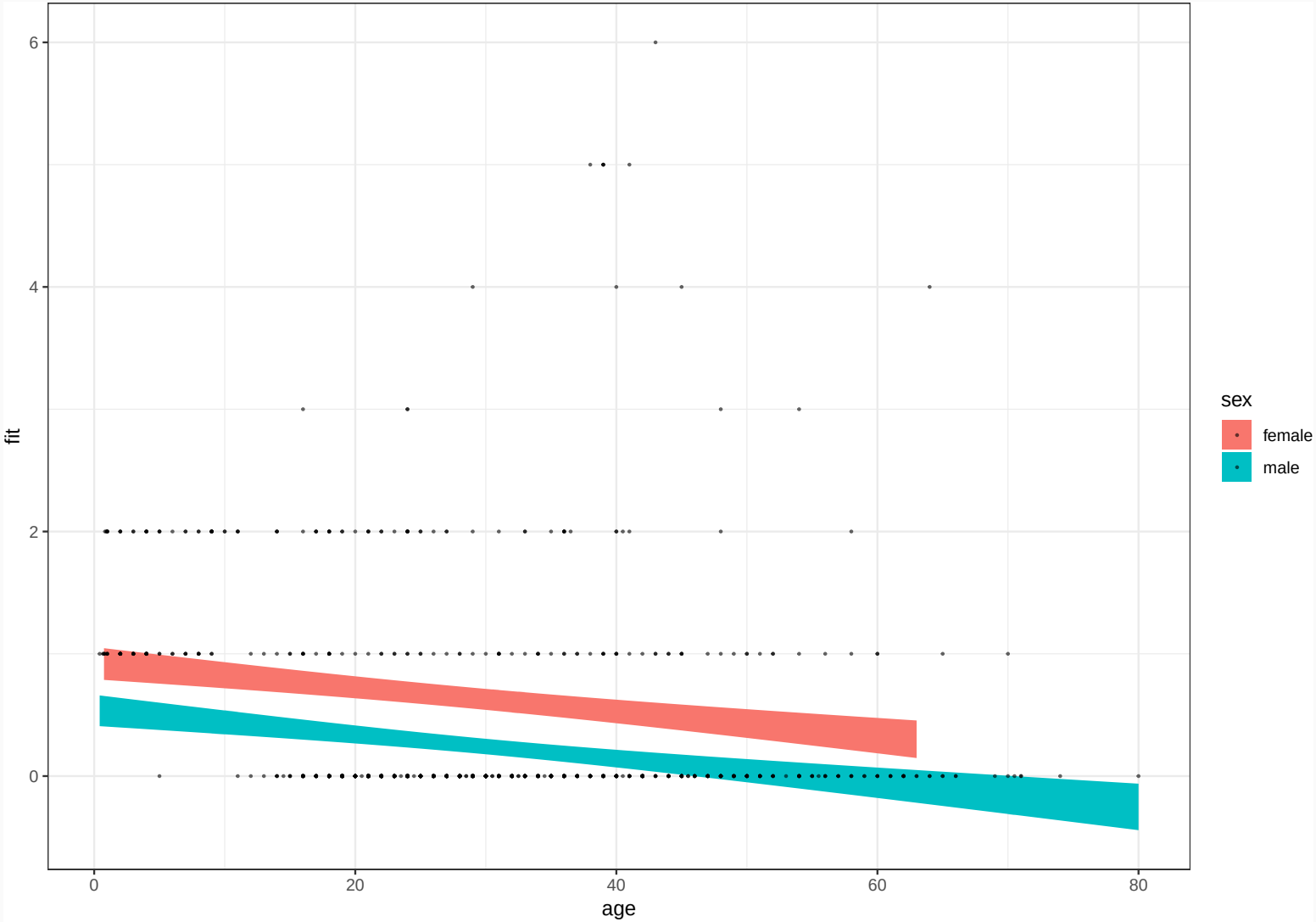


What about with the observed data?

```
newdata <- predict(m0, newdata = titanic, interval = "confidence")
newdata <- as.data.frame(newdata)
newdata$age <- titanic$age
newdata$sex <- titanic$sex
newdata$kids <- titanic$kids
```

Any problems here?

```
ggplot(newdata, aes(x = age, y = fit, ymin = lwr, ymax = upr, fill = sex)) +
    geom_ribbon() + geom_point(aes(x = age, y = kids), alpha = 0.6, size = 0.3)
```

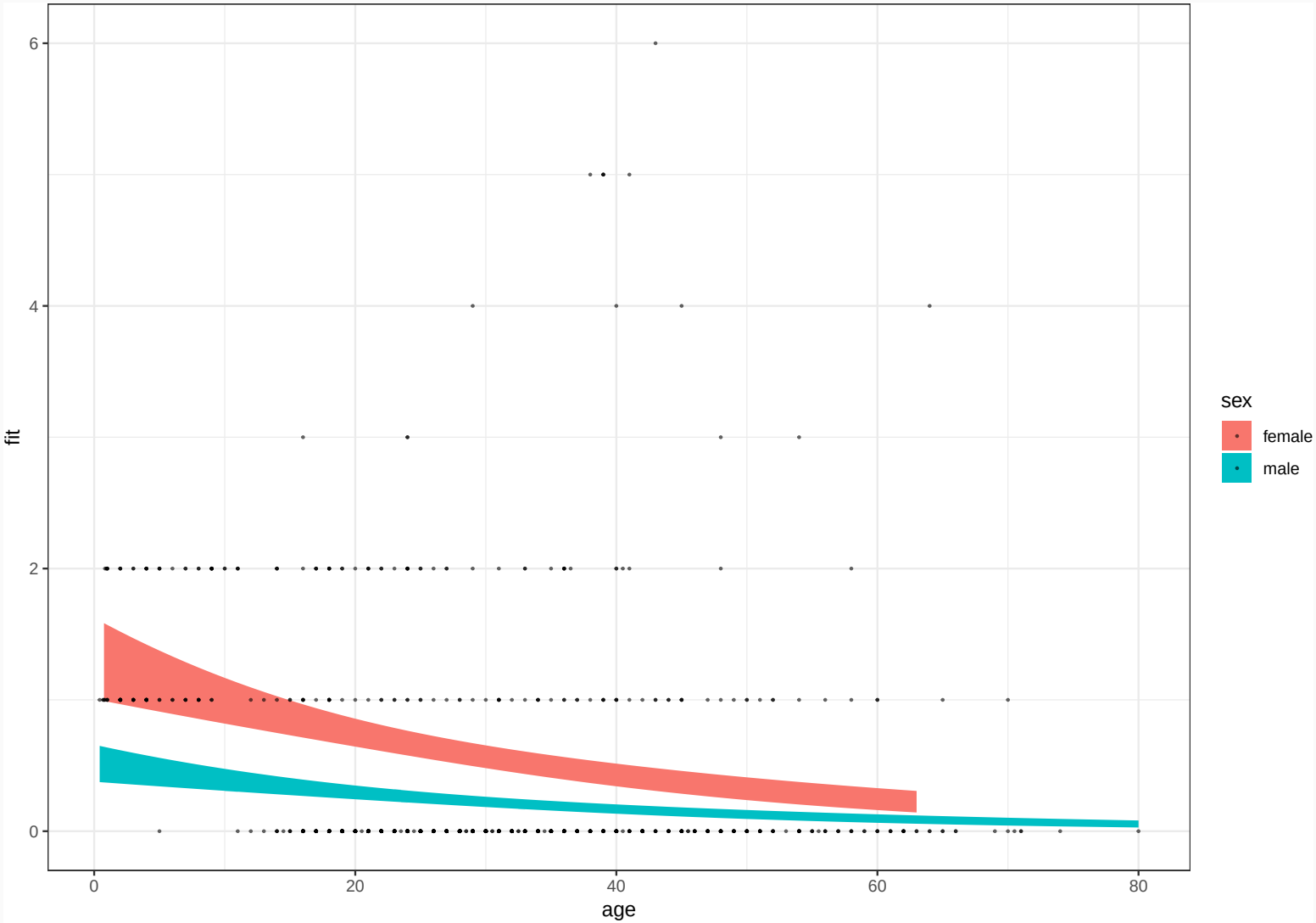


Let's try a different approach

```r
m1 <- glm(kids ~ sex + age, data = titanic, family = "poisson")
newdata <- predict(m1, newdata = titanic, se.fit = TRUE, type = "response")
newdata <- as.data.frame(newdata)
newdata$age <- titanic$age
newdata$sex <- titanic$sex
newdata$kids <- titanic$kids
```

Somewhat better? How?

```
ggplot(newdata, aes(x = age, y = fit, ymin = fit + 2 * se.fit, ymax = fit -
    2 * se.fit, fill = sex)) + geom_ribbon() + geom_point(aes(x = age, y = kids),
    alpha = 0.6, size = 0.3)
```



Approaches to modeling count data

The Poisson model

Where y is a non-negative integer (count)

$$y \sim Poisson(\lambda)$$

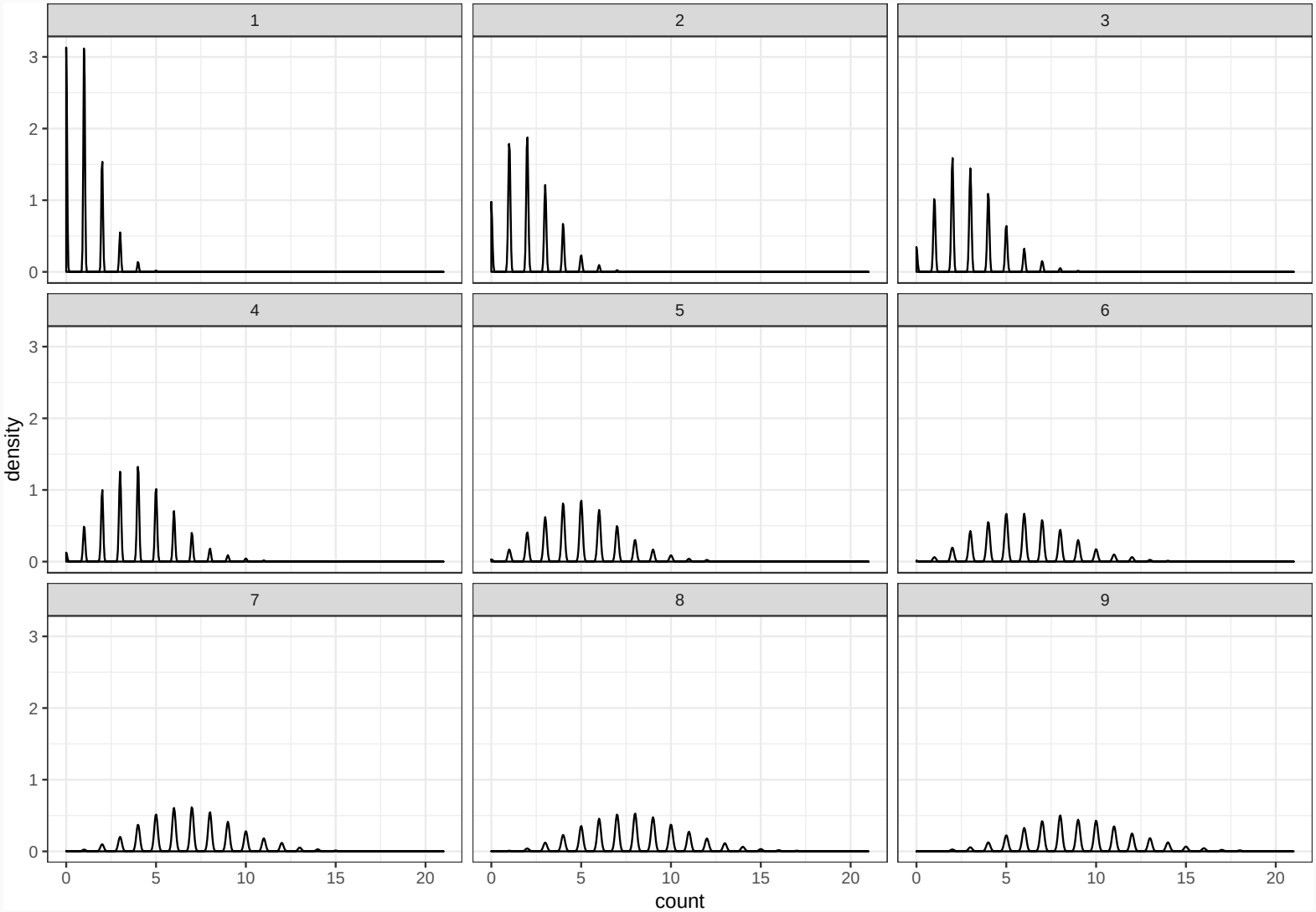
$$E(y) = \bar{y} = \lambda$$

$$Var(y) = \lambda$$

$$Pr(y = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Shape of the Poisson distribution

```
ggplot(pois_demo, aes(x = count)) + geom_density(adjust = 1/4) +
```



Let's look at each Poisson variable

```
pois_demo %>% group_by(lambda) %>%
summarise(mean = mean(count), variance = var(count))
```

```
## # A tibble: 9 x 3
##   lambda  mean variance
##    <int> <dbl>    <dbl>
## 1      1  1.01     1.01
## 2      2  2.00     1.99
## 3      3  2.97     2.94
## 4      4  4.00     3.99
## 5      5  5.02     5.04
## 6      6  6.02     6.27
## 7      7  7.00     6.93
## 8      8  7.98     8.11
## 9      9  8.98     9.20
```

Poisson models as a GLM

For a count variable y, we can specify a Poisson GLM with a log link function

$$y \sim Poisson(\lambda)$$

$$\lambda = \beta X = \beta_0 + \beta_1 x_1 \cdots \beta_n x_n$$

$$E(y|x) = e^{\lambda}$$

$$log(E(y|x)) = \lambda = \beta X$$

Returning to the titanic (again)

```
tidy(m1)
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   0.274    0.118        2.33 1.99e- 2
## 2 sexmale      -0.932    0.111       -8.38 5.37e-17
## 3 age          -0.0281   0.00418     -6.72 1.85e-11
```

$$E(y|age = age_i, sex = sex_i) = exp(0.27 - 0.93sex_i - 0.03age_i$$

Turning this into a prediction

$$E(y|age = age_i, sex = sex_i) = exp(0.27 - 0.93(sex_i = male) - 0.03age_i)$$

If age = 20 and sex = female

Turning this into a prediction

$$E(y|age = age_i, sex = sex_i) = exp(0.27 - 0.93(sex_i = male) - 0.03age_i)$$

If age = 20 and sex = female

$$E(y|x) = exp(0.27 - 0.6) = 0.72 = \lambda$$

What does this look like as a count?

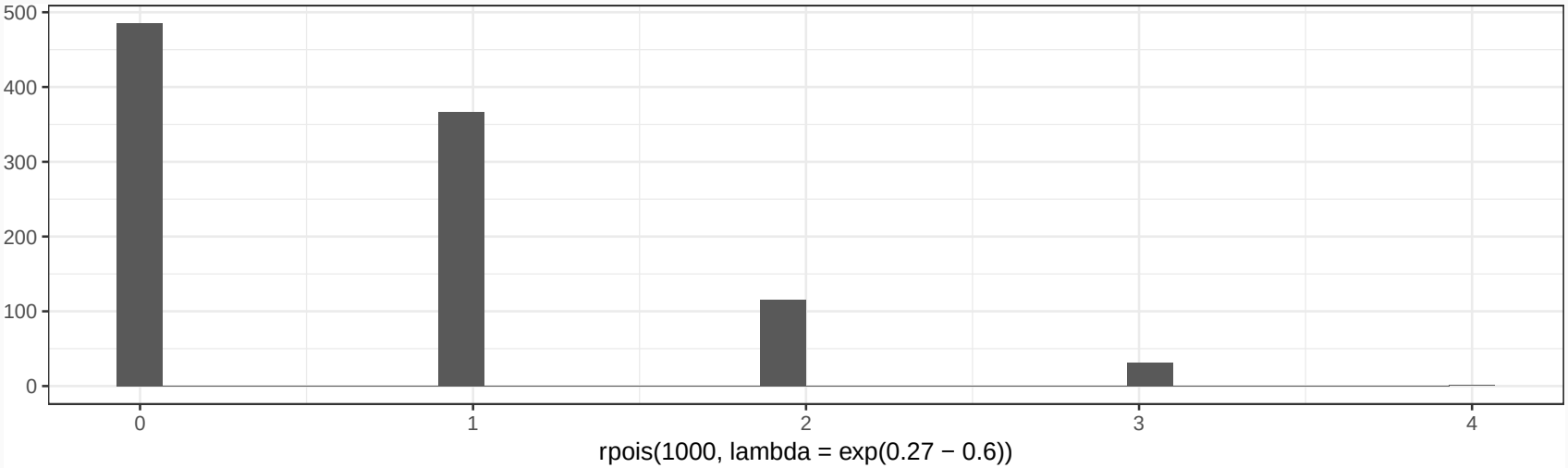
```
rpois(1, lambda = exp(0.27 - 0.6))
```

```
## [1] 1
```

```
rpois(10, lambda = exp(0.27 - 0.6))
```

```
##  [1] 0 2 2 0 0 1 2 1 2 1
```

```
qplot(rpois(1000, lambda = exp(0.27 - 0.6)))
```



Advantages of the Poisson distribution for regression

1. Constrained to non-negative integers
2. Variance scales with the expectation of y
3. Relatively simple to interpret

However:

$$\lambda = E(y|x) = var(y)$$

Is a pretty strong assumption that is rarely true. Let's check it on the Titanic

Overdispersion in count models

What if we did this:

$$\lambda = E(y|x)$$

$$var(y) = \phi\lambda$$

We can scale the variance by a parameter ϕ to create overdispersion, or more variance than we might expect under a standard model.

In real world settings, there is virtually always overdispersion

Quick check for overdispersion

First let's check the mean:

```
mean(titanic$kids)  #observed
```

```
## [1] 0.3833145
```

```
mean(exp(predict(m1)))  #predicted
```

```
## [1] 0.3833145
```

Quick check for overdispersion

First let's check the mean:

```
mean(titanic$kids)  #observed
```

```
## [1] 0.3833145
```

```
mean(exp(predict(m1)))  #predicted
```

```
## [1] 0.3833145
```

Now let's check the variance:

```
var(titanic$kids)  #observed variance
```

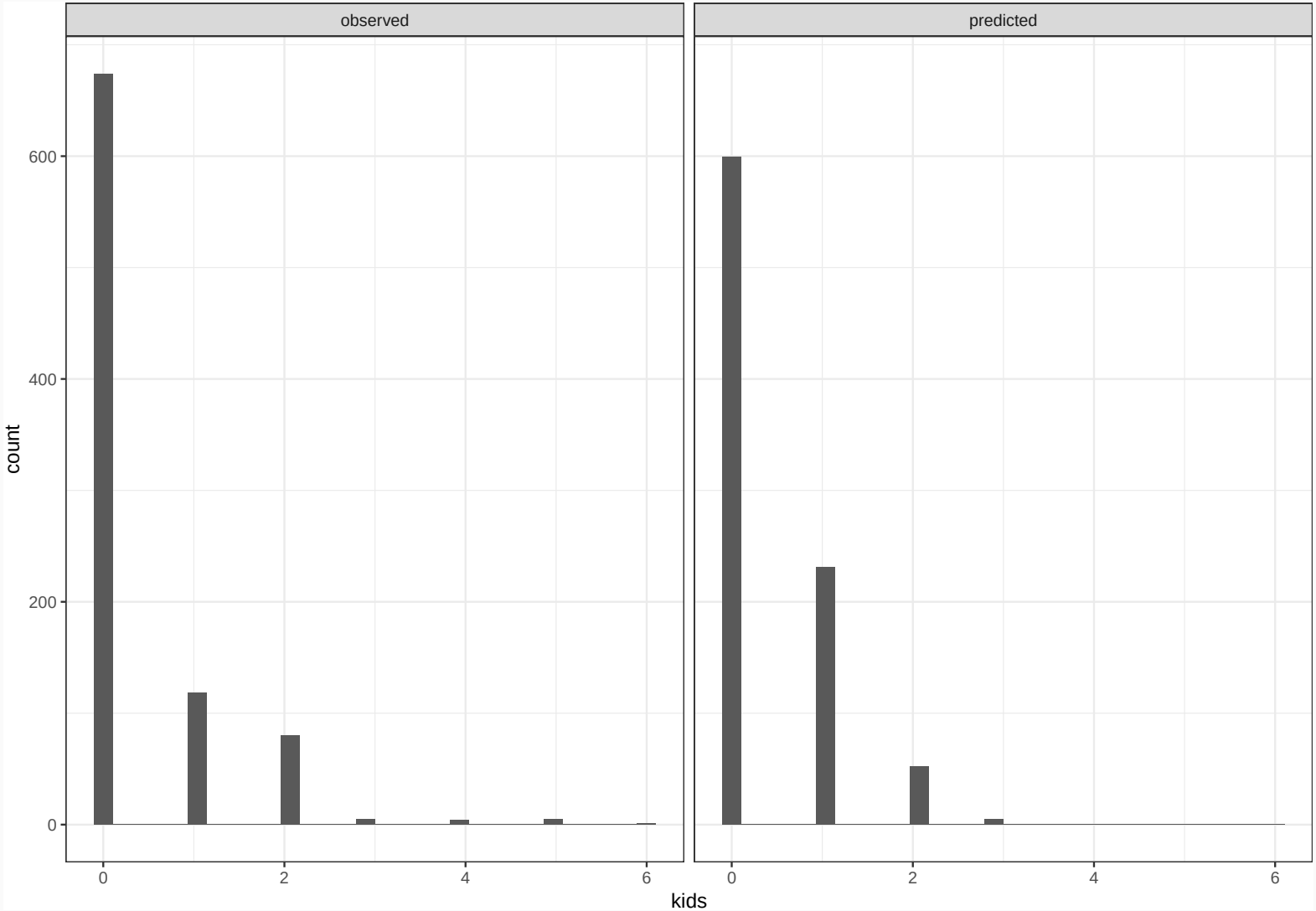
```
## [1] 0.6520012
```

```
var(exp(predict(m1)))  #predicted variance
```

```
## [1] 0.06549625
```

Notice any problems?

Let’s compare the histograms



Modeling the overdispersion

We have two primary options for modeling overdispersion

- The quasipoisson model (including ϕ to scale variance as an overdispersion parameter)
- The negative binomial model (more on this in a bit)

Quasipoisson

```r
m2 <- glm(kids ~ sex + age, family = "quasipoisson", data = titanic)
summary(m2)
```

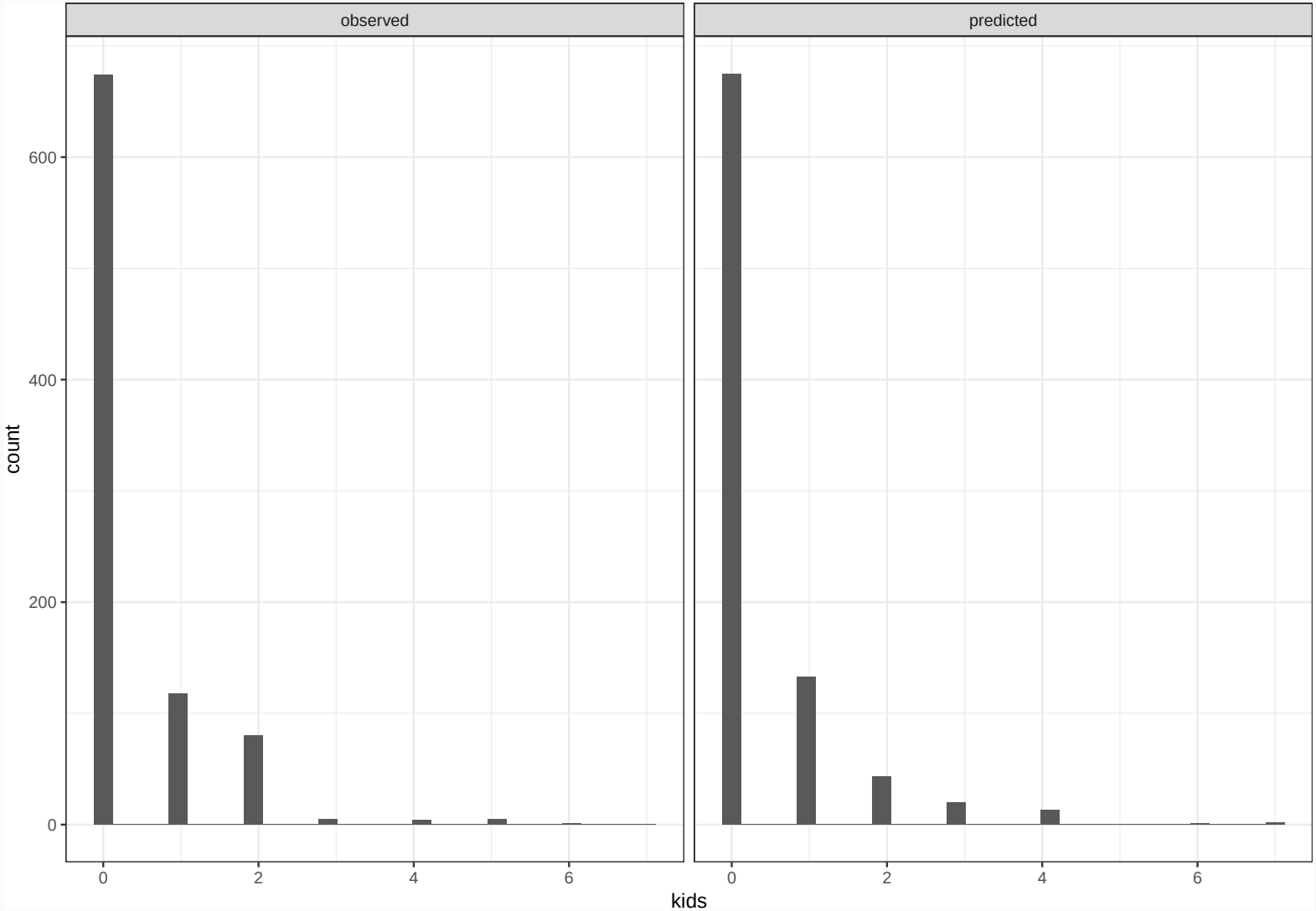
```
##
## Call:
## glm(formula = kids ~ sex + age, family = "quasipoisson", data = titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5119  -0.7795  -0.6680  -0.4060   4.8956
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.273845   0.159232   1.720   0.0858 .
## sexmale     -0.932247   0.150581  -6.191 9.14e-10 ***
## age         -0.028056   0.005652  -4.964 8.30e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 1.831483)
##
##     Null deviance: 1053.15  on 886  degrees of freedom
## Residual deviance:  920.88  on 884  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 6
```

The regular Poisson model on the same data

```
summary(m1)
```

```
##
## Call:
## glm(formula = kids ~ sex + age, family = "poisson", data = titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5119  -0.7795  -0.6680  -0.4060   4.8956
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.273845   0.117660   2.327   0.0199 *
## sexmale     -0.932247   0.111268  -8.378  < 2e-16 ***
## age         -0.028056   0.004177  -6.717 1.85e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1053.15  on 886  degrees of freedom
## Residual deviance:  920.88  on 884  degrees of freedom
## AIC: 1421.1
##
## Number of Fisher Scoring iterations: 6
```

Checking predictions again

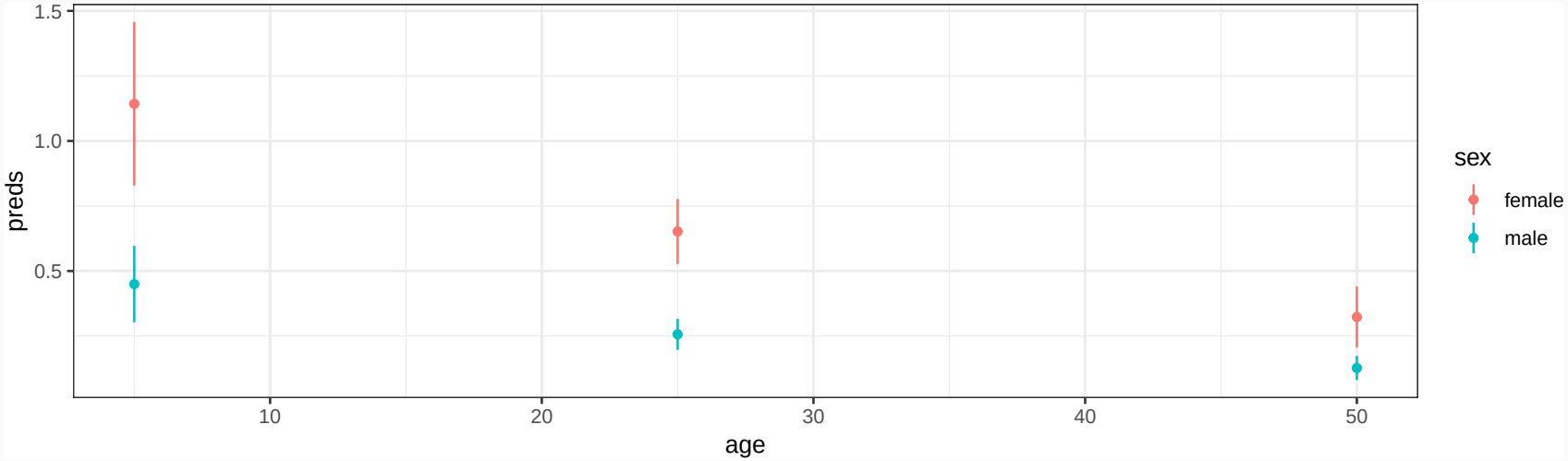


A simple set of predictions to interpret the model

```
newdata <- data.frame(sex = rep(c("male", "female")), age = rep(c(5, 25, 50),
    each = 2))

preds <- predict(m2, newdata, type = "response", se.fit = TRUE)

newdata$preds <- preds$fit
newdata$se <- preds$se.fit
ggplot(newdata, aes(x = age, y = preds, color = sex, ymax = preds + se * 2,
    ymin = preds - se * 2)) + geom_point() + geom_linerange()
```



But we can also get the gist from the coefficients

```
tidy(m2)
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   0.274    0.159        1.72 8.58e- 2
## 2 sexmale      -0.932    0.151       -6.19 9.14e-10
## 3 age          -0.0281   0.00565     -4.96 8.30e- 7
```

An alternative: negative binomial regression

The Negative Binomial distribution is analagous to an overdispersed Poisson regression

```
library(MASS)
select <- dplyr::select
m3 <- glm.nb(formula = kids ~ sex + age, data = titanic)
tidy(m3)
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   0.239    0.163        1.47 1.42e- 1
## 2 sexmale      -0.991    0.139       -7.15 8.82e-13
## 3 age          -0.0256   0.00525     -4.88 1.07e- 6
```

One more example

This is the data I use for this paper: **https://ajph.aphapublications.org/doi/abs/10.2105/AJPH.2018.304559**

```
fe <- read_csv("./data/fe_division_rural.csv")
head(fe)
```

```
## # A tibble: 6 x 15
##    fips state black.men white.men latino.men tot.men ur.code division
##   <dbl> <chr>     <dbl>     <dbl>      <dbl>   <dbl> <chr>   <chr>
## 1  1001 AL         3397     15446        483   19710 3: med~ East So~
## 2  1003 AL         6380     61953       3218   73125 4: sma~ East So~
## 3  1005 AL         5201      5618        473   11501 6: non~ East So~
## 4  1007 AL         2426      6708        163    9541 2: lar~ East So~
## 5  1009 AL          357     19176       1713   21584 2: lar~ East So~
## 6  1011 AL         2801      1211        359    4543 6: non~ East So~
## # ... with 7 more variables: d.asian <dbl>, d.black <dbl>, d.latino <dbl>,
## #   d.other <dbl>, d.white <dbl>, d.na <dbl>, d.total <dbl>
```

Variables: county fips code, state, male population by race/ethnicity, urban/rural code, census division, men killed by police use of force by race/ethnicity

What if counts have a theoretical limit that varies by unit of observation?

How many many men could theoretically be killed in county x in year y?

What if counts have a theoretical limit that varies by unit of observation?

How many many men could theoretically be killed in county x in year y?

We may want to bound our regression model by the size of the population.

Exposure variables in count models

We can adjust our model to offset for exposure, or roughly how many trials there could be across units.

$$log\big(E\big(Y|X\big)\big) - log\big(exposure\big) log \frac{E\big(Y|X\big)}{exposure} = \beta X$$

Exposure variables in count models

We can adjust our model to offset for exposure, or roughly how many trials there could be across units.

$$log\big(E\big(Y|X\big)\big) - log\big(exposure\big) log\frac{E\big(Y|X\big)}{exposure} = \beta X$$

If exposure = population, then conveniently, we are modeling per capita rates

Using offset to include exposure in a model

We estimate a model that predicts total police killings as a function of county metropolitan type.

Why would we want an offset in this model?

Using offset to include exposure in a model

We estimate a model that predicts total police killings as a function of county metropolitan type.

Why would we want an offset in this model?

```
m1<-glm(d.total ~ ur.code +

          offset(log(tot.men)),

        data = fe, family = "quasipoisson")
```

Zero-inflated count models

If you suspect your data may have two processes, one that determines if the outcome is zero or greater than zero, and one that determines the value of a count we can use a two-stage model.

1. First, estimate a logistic regresion where your outcome y = 1 if the count z > 0
2. Second, estimate a count model on the subset of the data where z > 0

Example: Predicting incarceration length in days from a sample of the general population. We may first want to predict the likelihood that incarceration days > 0, as people must be arrested or conviceted to be incarcerated. There are two data generating processes here: 1) arrest/conviction, 2) sentence/pre-trial detention

Count models: general guidance

- If your data are non-zero integers, count models are generally more appropriate than OLS
- Default to an overdispersed model (quasipoisson, negative binomial)
- Unless you are certain the outcome is not overdispersed, do not trust standard errors or p-values from Poisson models without overdispersion parameters
- Count models use a log link function - β can generally be interpreted as a unit change in x predicts a percent change in y.
- Include an offset with an exposure variable when the number of trials differs across units (e.g. population size varies)