Count data and the Poisson distribution

Frank Edwards

Count data

- Counts are cumulative totals of the number of incidences of some event, generally across time or place

Count data

- Counts are cumulative totals of the number of incidences of some event, generally across time or place
- Counts are positive integers $\in [0, \infty]$

Count data

- Counts are cumulative totals of the number of incidences of some event, generally across time or place
- Counts are positive integers $\in [0, \infty]$

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?

Data for today: National Women's Soccer League stats

```
## data from https://github.com/adror1/nwslR
# devtools::install_github("adror1/nwslR")
library(nwslR)
data("player")
data("fieldplayer_overall_season_stats")
head(player, n=2)
```

```
## # A tibble: 2 x 5
##   person_id player        nation pos   name_other
##       <dbl> <chr>         <chr>  <chr> <chr>
## 1       342 Marisa Abegg  USA    DF    <NA>
## 2       117 Danesha Adams USA    FW,MF <NA>
```

```
head(fieldplayer_overall_season_stats, n=2)
```

```
## # A tibble: 2 x 14
##   person_id season nation pos   team_id    mp starts   min   gls   ast    pk
##       <int>  <dbl> <chr>  <chr> <chr>   <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
## 1       342   2013 USA    DF    WAS         5      4    NA     0     0     0
## 2       117   2013 USA    FW,MF NJ         20     20    NA     3     3     1
## # i 3 more variables: p_katt <dbl>, crd_y <dbl>, crd_r <dbl>
```

```
# check the help files with ?(fieldplayer_overall_season_stats) for codebook
```

make a joined table with players names

```r
### attaching names
dat<-fieldplayer_overall_season_stats %>%
  left_join(player)

glimpse(dat)
```

```
## Rows: 1,350
## Columns: 16
## $ person_id  <dbl> 342, 117, 6, 300, 202, 202, 28, 290, 56, 313, 363, 454, 414~
## $ season     <dbl> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013,~
## $ nation     <chr> "USA", "USA", "ESP", "USA", "USA", "USA", "USA", "USA", "US~
## $ pos        <chr> "DF", "FW,MF", "FW", "DF,MF", "DF", "DF", "DF", "DF", "MF",~
## $ team_id    <chr> "WAS", "NJ", "WNY", "KC", "POR", "BOS", "WNY", "SEA", "NJ",~
## $ mp         <dbl> 5, 20, 15, 22, 4, 11, 7, 22, 7, 20, 3, 2, 1, 22, 20, 6, 1, ~
## $ starts     <dbl> 4, 20, 14, 22, 2, 11, 2, 22, 5, 11, 0, 1, 0, 22, 16, 5, 0, ~
## $ min        <dbl> NA, NA, 3, 1900, 212, 990, NA, NA, NA, NA, NA, 123, NA, 198~
## $ gls        <dbl> 0, 3, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 3, 2, 0, 0, 5, 0,~
## $ ast        <dbl> 0, 3, NA, 5, 0, 1, NA, 1, 0, 2, 0, 0, 0, 2, 1, 0, 0, 0, 4, ~
## $ pk         <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ p_katt     <dbl> 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ crd_y      <dbl> 0, 0, 2, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 3, 0,~
## $ crd_r      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ player     <chr> "Marisa Abegg", "Danesha Adams", "Adriana", "Leigh Ann Brow~
## $ name_other <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

Approaches to modeling count data

The Poisson model

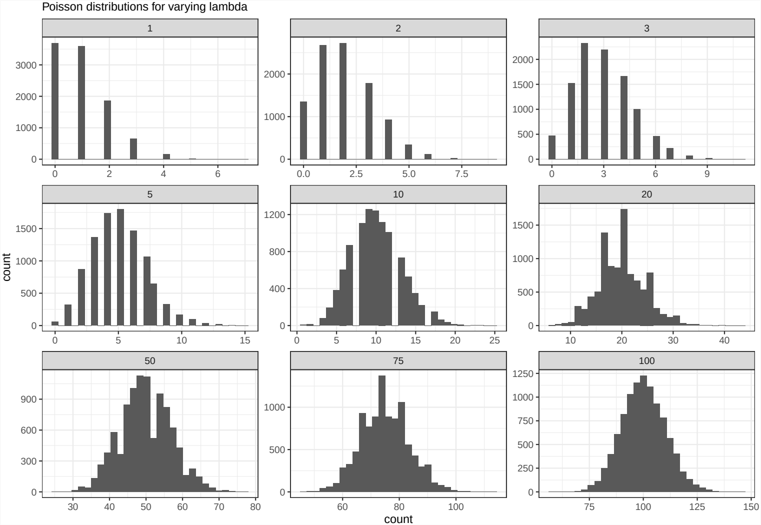
Where y is a non-negative integer (count)

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

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

$$Var(y) = \lambda$$

The Poisson Distribution



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
##    <dbl>  <dbl>    <dbl>
## 1      1   1.01     1.01
## 2      2   2.00     1.99
## 3      3   2.97     2.94
## 4      5   5.01     5.00
## 5     10  10.0     10.0
## 6     20  20.0     20.3
## 7     50  49.9     50.0
## 8     75  75.0     72.8
## 9    100 100.     101.
```

Poisson models as a GLM

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

$$y \sim Poisson(\lambda)$$
$$\lambda = e^{\beta_0 + \beta_1 x_1 \cdots \beta_n x_n}$$

Poisson models as a GLM

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

$$y \sim Poisson(\lambda)$$
$$\lambda = e^{\beta_0 + \beta_1 x_1 \cdots \beta_n x_n}$$

What is $\log(\lambda)$ equal to?

Poisson models as a GLM

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

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

Poisson models as a GLM

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

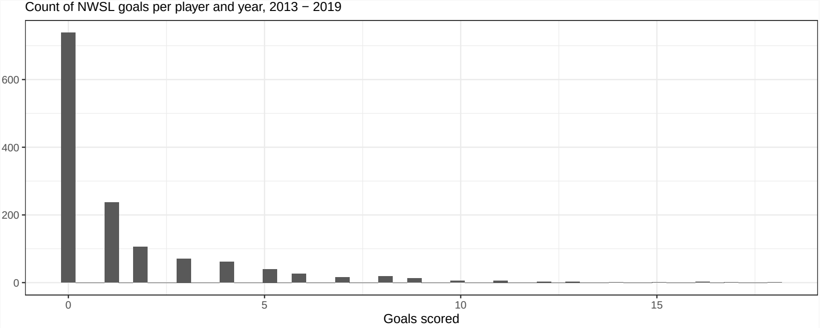
$$log(E(y|x)) = \lambda = X\beta$$

if a GLM is defined as $g(\mu) = X\beta$ with link function g, what is the link function for the Poisson GLM?

Modeling NWSL data using a Poisson GLM

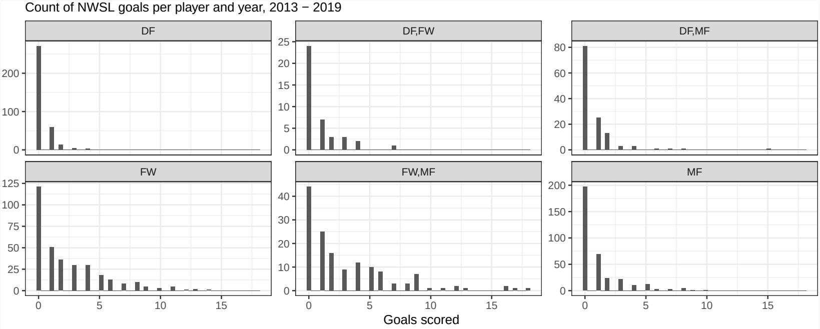
Goal scoring

```
ggplot(dat,
       aes(x = gls)) +
  geom_histogram(bins = 50) +
  labs(x = "Goals scored", y = "",
       subtitle = "Count of NWSL goals per player and year, 2013 - 2019")
```



Goal scoring

```r
ggplot(dat,
       aes(x = gls)) +
  geom_histogram(bins = 50) +
  facet_wrap(~pos, scales = "free_y") +
  labs(x = "Goals scored", y = "",
       subtitle = "Count of NWSL goals per player and year, 2013 - 2019")
```



Modeling goals

```
goals_0<-stan_glm(gls ~ pos,
              data = dat,
              family = "poisson",
              refresh = 0)

goals_0
```

```
## stan_glm
##  family:       poisson [log]
##  formula:      gls ~ pos
##  observations: 1350
##  predictors:   6
## ------
##             Median MAD_SD
## (Intercept) -1.1    0.1
## posDF,FW     1.0    0.2
## posDF,MF     0.9    0.1
## posFW        2.0    0.1
## posFW,MF     2.2    0.1
## posMF        1.2    0.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

So how many goals does our model expect for each position?

We could just do the math: $\lambda_i = E(y_i|X) = e^{\beta_0 + \beta_1 x_1 \ldots \beta_n x_n}$

```
exp(coef(goals_0))
```

```
## (Intercept)    posDF,FW    posDF,MF       posFW    posFW,MF       posMF
##    0.335945    2.729285    2.483760    7.284995    9.422848    3.288571
```

And because $e^{a+b} = e^a \times e^b$

Expected goals for a forward under model 0 are $e^{\beta_0} \times e^{\beta_3}$

```
# intercept is in row 1, b3 is in row 4
exp(coef(goals_0)[4]) * exp(coef(goals_0)[1])
```

```
##    posFW
## 2.447358
```

So how many goals does our model expect for each position?

Or we could have R handle everything using `predict()`

```
sim_dat <- data.frame(pos = unique(dat$pos))
sim_dat <- sim_dat %>%
    mutate(e_gls = predict(goals_0, newdata = sim_dat, type = "response"))

sim_dat
```

```
##     pos     e_gls
## 1    DF 0.3367806
## 2 FW,MF 3.1723728
## 3    FW 2.4509501
## 4 DF,MF 0.8365754
## 5    MF 1.1053956
## 6 DF,FW 0.9243009
```

Regression generates conditional means

Not coincidentally:

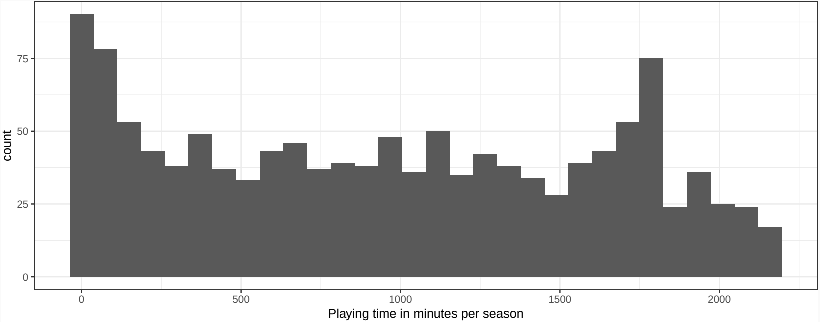
```
dat %>% group_by(pos) %>% summarize(gls = mean(gls))
```

```
## # A tibble: 6 x 2
##   pos     gls
##   <chr> <dbl>
## 1 DF    0.334
## 2 DF,FW 0.925
## 3 DF,MF 0.837
## 4 FW    2.45
## 5 FW,MF 3.17
## 6 MF    1.11
```

Fitting a more complex model

Let's look at playing time as a predictor

```
ggplot(dat,
       aes(x = min)) +
  geom_histogram() +
  labs(x = "Playing time in minutes per season")
```



How does playing time impact scoring?

Theory time

How does playing time impact scoring?

As an opportunity structure - more time = more chances

Theory time

How does playing time impact scoring?

As an opportunity structure - more time = more chances

Is playing time likely to have the same effect on goal scoring for each position?

How does playing time impact scoring?

As an opportunity structure - more time = more chances

Is playing time likely to have the same effect on goal scoring for each position?

Let's evaluate this model:

$$m1 : E(goals|position, minutes) = e^{\beta_0+\beta_1 position+\beta_2 minutes}$$

Estimate the model

```
# 1 minute is a small difference, let's use z-scores
goals_1<-stan_glm(gls ~ pos + scale(min),
              data = dat,
              family = "poisson",
              refresh = 0)
```

Check our fits

```
goals_1
```

```
## stan_glm
##  family:       poisson [log]
##  formula:      gls ~ pos + scale(min)
##  observations: 1271
##  predictors:   7
## ------
##             Median MAD_SD
## (Intercept) -1.5    0.1
## posDF,FW     1.3    0.2
## posDF,MF     0.8    0.1
## posFW        2.3    0.1
## posFW,MF     2.4    0.1
## posMF        1.3    0.1
## scale(min)   0.8    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

What does this show?

Let’s look at model expectations with simulation

```r
## all positions with varying play time
pos<-unique(dat$pos)
min<-seq(from = 0,
        to = max(dat$min, na.rm=T),
        by = 5)
sim_dat<-expand_grid(pos, min)

head(sim_dat)
```

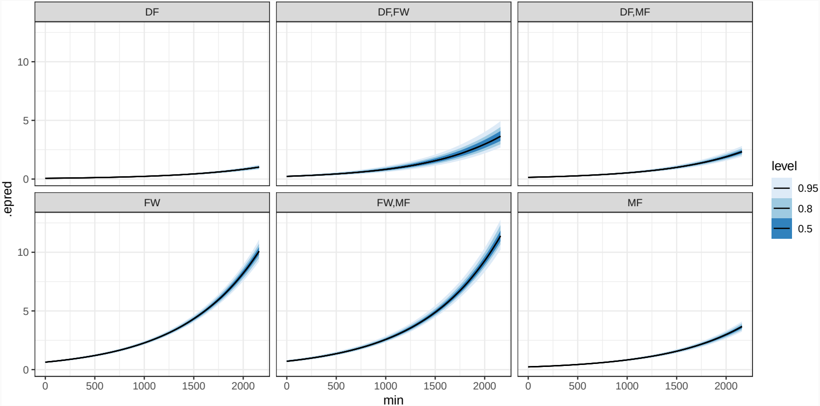
```
## # A tibble: 6 x 2
##   pos     min
##   <chr> <dbl>
## 1 DF        0
## 2 DF        5
## 3 DF       10
## 4 DF       15
## 5 DF       20
## 6 DF       25
```

Now simulate

```r
## add epred draws adds on the expected value scale
library(tidybayes)
sim_dat<-sim_dat %>%
  add_epred_draws(goals_1,
                  ndraws = 500)
```

Now to visualize expected values

```
ggplot(sim_dat,
       aes(y = .epred, x = min)) +
  stat_lineribbon(size = 0.5) + # to make the line thinner
  facet_wrap(~pos) +
  scale_fill_brewer()
```



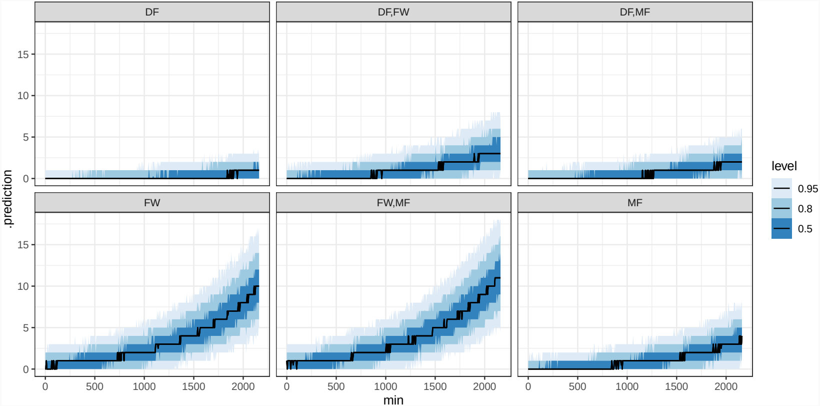
Let's run the simulations again, but now with predicted goals (rather than expected)

```
pos<-unique(dat$pos)
min<-seq(from = 0,
         to = max(dat$min, na.rm=T),
         by = 5)
sim_dat<-expand_grid(pos, min)

sim_dat<-sim_dat %>%
  add_predicted_draws(goals_1,
                      ndraws = 500)
```

Visualize predicted values

```
ggplot(sim_dat,
       aes(y = .prediction, x = min)) +
  stat_lineribbon(size = 0.5) + # to make the line thinner
  facet_wrap(~pos) +
  scale_fill_brewer()
```



Advantages of the Poisson distribution for regression

1. Constrained to non-negative integers
2. Variance scales with the expectation of y (non-constant error variance!)
3. Relatively simple to interpret