

# GLM(M)s for counts

Bayesian statistics 6 – generalized linear models for count data

Frédéric Barraquand (CNRS, IMB)

05/12/2023

# Some things that we learned the last time

- You can use GLMs to model counts.
  - ① If you want to explain *and* counts are relatively large, you can also transform.
  - ② If your want to predict or counts are small, you have to use GLMs.
- The Poisson distribution is useful to model *small* counts
- A main property is that mean = variance, so small counts have large CV.
- Classical *link function* is the log-link, so Poisson regression looks like  $Y_i \sim \mathcal{P}(\exp(a + bx_i + [\text{stuff}]))$

# The law of small numbers

Book written by [Władysław Bortkiewicz](#) in 1898.



Bortkiewicz, unsung hero of small numbers and weird datasets

- not to be confused with the [law of large numbers](#) which refers to averaging. Here it is a “law of rare events”.
- events with low frequency  $p$  in a large population  $n$  follow a Poisson distribution.  $Y \sim \mathcal{B}(n, p) \rightarrow \mathcal{P}(np)$  for large  $n$  and small  $p$ . Even if actually there are  $n$  Bernoulli trials with varying probability  $p_i$ .

# Prussian army horse-kick data

```
horsekick = read.csv("Prussian_horse-kick_data.csv")  
horsekick
```

##	Year	GC	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C14	C15
## 1	1875	0	0	0	0	0	0	0	1	1	0	0	0	1	0
## 2	1876	2	0	0	0	1	0	0	0	0	0	0	0	1	1
## 3	1877	2	0	0	0	0	0	1	1	0	0	1	0	2	0
## 4	1878	1	2	2	1	1	0	0	0	0	0	1	0	1	0
## 5	1879	0	0	0	1	1	2	2	0	1	0	0	2	1	0
## 6	1880	0	3	2	1	1	1	0	0	0	2	1	4	3	0
## 7	1881	1	0	0	2	1	0	0	1	0	1	0	0	0	0
## 8	1882	1	2	0	0	0	0	1	0	1	1	2	1	4	1
## 9	1883	0	0	1	2	0	1	2	1	0	1	0	3	0	0
## 10	1884	3	0	1	0	0	0	0	1	0	0	2	0	1	1
## 11	1885	0	0	0	0	0	0	1	0	0	2	0	1	0	1
## 12	1886	2	1	0	0	1	1	1	0	0	1	0	1	3	0
## 13	1887	1	1	2	1	0	0	3	2	1	1	0	1	2	0
## 14	1888	0	1	1	0	0	1	1	0	0	0	0	1	1	0
## 15	1889	0	0	1	1	0	1	1	0	0	1	2	2	0	2
## 16	1890	1	2	0	2	0	1	1	2	0	2	1	1	2	2
## 17	1891	0	0	0	1	1	1	0	1	1	0	3	3	1	0
## 18	1892	1	3	2	0	1	1	3	0	1	1	0	1	1	0
## 19	1893	0	1	0	0	0	1	0	2	0	0	1	3	0	0
## 20	1894	1	0	0	0	0	0	0	0	1	0	1	1	0	0

## Btw, conjugate prior = Gamma

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior}$$

The same way we have always

$$\text{Beta} \propto \text{Binomial} \times \text{Beta}$$

here we have

$$\text{Gamma} \propto \text{Poisson} \times \text{Gamma}$$

If you measure  $n$   $\text{Poisson}(\lambda)$ -distributed values  $y_i$  with  $\Gamma(\alpha, \beta)$  prior on  $\lambda$ , the posterior distribution for  $\lambda$  is  $\Gamma(\alpha + \sum_{i=1}^n y_i, \beta + n)$ .

# Formatting the data

```
year = horsekick$Year
count = as.matrix(horsekick[,2:15])

# Bundle data
str(bdata <- list(year=year, count=count,
                  ngroups = ncol(count), T=ncol(count)))
```

# Poisson ANOVA for horse-kick data

```
# Specify model in BUGS language
cat(file = "poisson.anova.txt", "
model {

# Priors
  for (j in 1:ngroups){alpha[j] ~ dnorm(1,0.1)}

# Likelihood
  for (t in 1:T){
    for (i in 1:ngroups){
      count[t,i] ~ dpois(lambda[t,i])
      log(lambda[t,i]) <- alpha[i]
    }
  }

# Derived quantity
mu <- mean(alpha)
for (i in 1:ngroups){
  lambdaS[i] <- sum(lambda[1:T,i])
}

}
")
```

# Running the model for horse-kick data I

```
# Inits function
```

```
inits <- function(){list(alpha = rnorm(14, 0, 1))}
```

```
# Parameters to estimate
```

```
params <- c("lambdaS")
```

```
# MCMC settings
```

```
nc <- 3 ; ni <- 2000 ; nb <- 1000 ; nt <- 2
```

```
# Call JAGS, check convergence and summarize posteriors
```

```
out <- jags(bdata, inits, params, "poisson.anova.txt", n.thin = nt,  
           n.chains = nc, n.burnin = nb, n.iter = ni)
```

```
## Warning in jags.model(model.file, data = data, inits = init.values, n.chains =  
## n.chains, : Unused variable "year" in data
```



# Running the model for horse-kick data II

```
## Compiling model graph
##   Resolving undeclared variables
##   Allocating nodes
## Graph information:
##   Observed stochastic nodes: 196
##   Unobserved stochastic nodes: 14
##   Total graph size: 342
##
## Initializing model
```

```
print(out, dig = 3)      # Bayesian analysis
```

```
## Inference for Bugs model at "poisson.anova.txt", fit using jags,
## 3 chains, each with 2000 iterations (first 1000 discarded), n.thin = 2
## n.sims = 1500 iterations saved
```

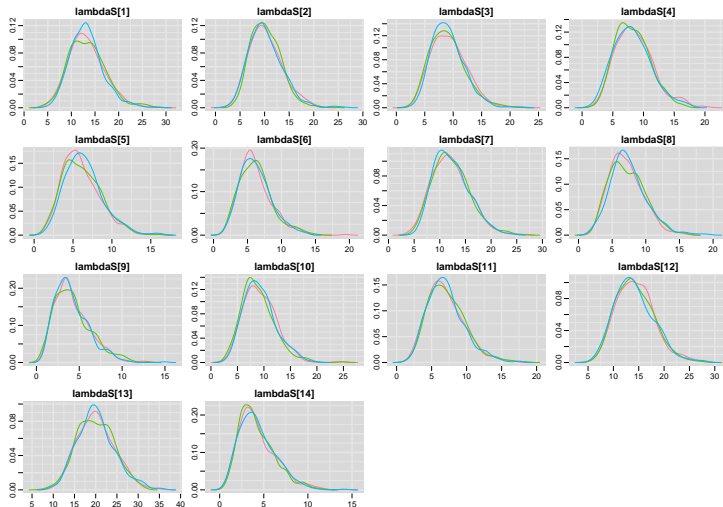
	mu.vect	sd.vect	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
## lambdaS[1]	13.276	3.695	7.070	10.625	12.993	15.400	21.577	1.002	1500
## lambdaS[2]	10.264	3.279	4.899	7.872	9.902	12.291	17.646	1.003	1500
## lambdaS[3]	9.102	2.988	4.272	6.889	8.844	10.837	15.650	1.003	1500
## lambdaS[4]	8.338	2.961	3.535	6.144	8.048	10.188	15.322	1.004	570
## lambdaS[5]	6.206	2.448	2.161	4.379	5.893	7.655	11.729	1.006	360
## lambdaS[6]	6.215	2.373	2.473	4.522	5.883	7.506	11.982	1.000	1500
## lambdaS[7]	12.238	3.519	6.622	9.609	11.836	14.393	19.780	1.002	1500
## lambdaS[8]	7.255	2.608	3.104	5.380	6.942	8.869	12.968	1.003	680

## Running the model for horse-kick data III

```
## lambdaS[9]      4.211    2.079    1.194    2.695    3.818    5.382    9.080 1.001 1500
## lambdaS[10]     9.132    3.054    4.227    6.936    8.796   10.965   15.945 1.005  440
## lambdaS[11]     7.107    2.651    3.026    5.203    6.776    8.603   13.298 1.001 1500
## lambdaS[12]    14.264    3.721    8.227   11.511   13.907   16.587   22.551 1.001 1500
## lambdaS[13]    20.075    4.365   12.513   16.904   19.835   22.901   29.291 1.000 1500
## lambdaS[14]     4.277    2.062    1.325    2.778    3.875    5.480    9.284 1.000 1500
## deviance      417.916    5.249 409.754 414.065 417.279 421.041 429.314 1.003  760
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 13.8 and DIC = 431.7
## DIC is an estimate of expected predictive error (lower deviance is better).

library(mcmcplots)
denplot(out,parms="lambdaS")
```

# Running the model for horse-kick data IV



```
## Warning in jags.model(model.file, data = data, inits = init.values, n.chains =  
## n.chains, : Unused variable "year" in data
```

# Posterior predictive checks

## Posterior predictive distribution

$$p(y^{\text{rep}}|y) = \int \underbrace{p(y^{\text{rep}}|y, \theta)}_{\text{new model draws}} \underbrace{p(\theta|y)}_{\text{posterior}} d\theta$$

(Negative-Binomial distributed in Poisson ANOVA or regression).

Much easier to obtain as code than write out

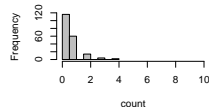
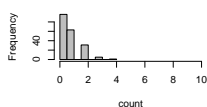
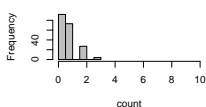
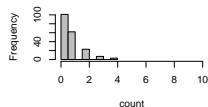
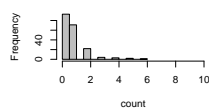
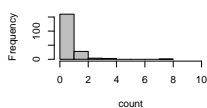
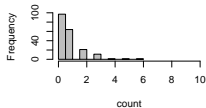
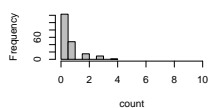
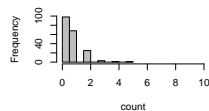
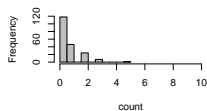
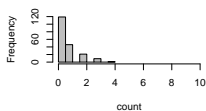
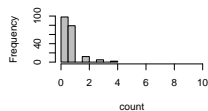
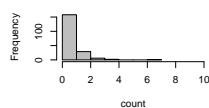
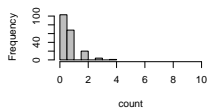
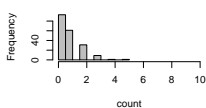
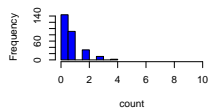
```
# New derived quantity
for (t in 1:T){
  for (i in 1:ngroups){
    count.rep[t,i] ~ dpois(lambda[t,i])
  }
}
```

# Posterior predictive checks (practice) I

```
#library(RColorBrewer)  
str(out$BUGSoutput$sims.list$count.rep)  
  
##  num [1:1500, 1:14, 1:14] 1 0 0 1 0 0 0 3 0 2 ...  
par(mfrow=c(4,4))  
hist(count,col="blue",xlim=c(0,10),xlab = "count")  
for (i in 1:15){  
  hist(out$BUGSoutput$sims.list$count.rep[i,,],  
       col="gray",xlim=c(0,10),main="",xlab = "count")  
}
```

## Posterior predictive checks (practice) II

### Histogram of count



# What if the data is over-dispersed?

What do we mean?  $\mathbb{V}(Y_i) \propto \mathbb{E}(Y_i)^b$  with  $b > 1$  ( $b = 1$ ) for Poisson.

- Remember: We can obtain  $b = 2$  for Gamma or Log-Normal

# What if the data is over-dispersed?

What do we mean?  $\mathbb{V}(Y_i) \propto \mathbb{E}(Y_i)^b$  with  $b > 1$  ( $b = 1$ ) for Poisson.

- Remember: We can obtain  $b = 2$  for Gamma or Log-Normal
- Logical (and historical) strategy: Poisson-mixture



# Gamma–Poisson aka Negative Binomial

Compound or mixture distribution

$$Y_i | \lambda_i \sim \mathcal{P}(\lambda_i)$$

and

$$\lambda_i \sim \Gamma(\alpha, \beta)$$

is equivalent to  $Y_i \sim \text{NB}(r, p)$  with  $\alpha = r$  and  $\beta = \frac{p}{1-p}$ . [Proof](#).

Facts about the NB distribution:  $\mathbb{E}(Y_i) = \mu = \frac{\alpha}{\beta} = \frac{r(1-p)}{p}$  and we can show that  $\mathbb{V}(Y_i) = \mu + \mu^2/r$ .

# Poisson–Log-Normal

$Y_i | \epsilon_i \sim \mathcal{P}(\exp(a + bx_i + \epsilon_i))$  with  $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$  for regression

$Y_i | \epsilon_i \sim \mathcal{P}(\exp(\alpha_{j[i]} + \epsilon_i))$  with  $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$  for ANOVA

Denoting  $m_i = \exp(a + bx_i + \sigma^2/2)$  the mean of the log-normal distribution, we can show that  $\mathbb{V}(Y_i) = m_i + (e^{\sigma^2} - 1)m_i^2$ .

# Applying to horsekick data I

# Applying to horsekick data II

*# Specify model in BUGS language*

```
cat(file = "poisson.ln.anova.txt", "  
model {
```

```
# Priors
```

```
  for (j in 1:ngroups){alpha[j] ~ dnorm(1,0.1)}  
  sigma ~ dexp(1)  
  tau <-pow(sigma,-2)  
  sigma2 <-pow(sigma,2)
```

```
# Likelihood
```

```
  for (t in 1:T){  
    for (i in 1:ngroups){  
      count[t,i] ~ dpois(lambda[t,i])  
      epsilon[t,i] ~ dnorm(0,tau)  
      log(lambda[t,i]) <- alpha[i] + epsilon[t,i]  
    }  
  }
```

```
# Derived quantity
```

```
mu <- mean(alpha)  
  for (t in 1:T){  
    for (i in 1:ngroups){  
      epsilon.rep[t,i] ~ dnorm(0,tau)  
      count.rep[t,i] ~ dpois(exp(alpha[i]+epsilon.rep[t,i]))
```

# Posterior predictive checks again I

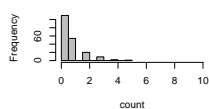
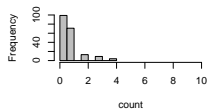
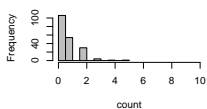
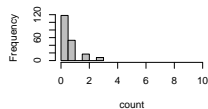
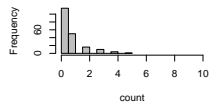
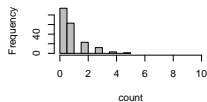
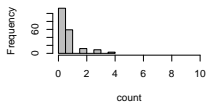
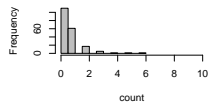
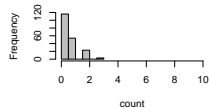
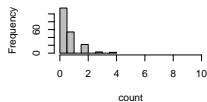
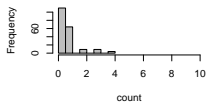
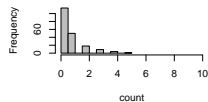
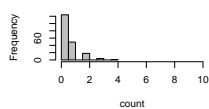
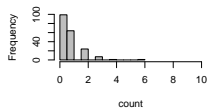
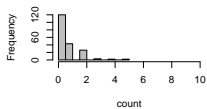
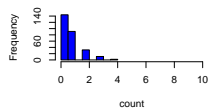
```
str(out$BUGSoutput$sims.list$count.rep)
```

```
##  num [1:1500, 1:14, 1:14] 1 0 0 1 0 0 0 3 0 2 ...
```

```
par(mfrow=c(4,4))  
hist(count,col="blue",xlim=c(0,10),xlab = "count")  
for (i in 1:15){  
  hist(out2$BUGSoutput$sims.list$count.rep[i,,],  
       col="gray",xlim=c(0,10),main="",xlab = "count")  
}
```

## Posterior predictive checks again II

Histogram of count



# PLN mixed model: estimating intercorps variance I

# PLN mixed model: estimating intercorps variance II

```
# Specify model in BUGS language
cat(file = "poisson.lmm.txt", "
model {

  # Priors
  for (j in 1:ngroups){alpha[j] ~ dnorm(1,tau_alpha)}

  # Residual variance
  sigma ~ dexp(1)
  tau <-pow(sigma,-2)
  sigma2 <-pow(sigma,2)

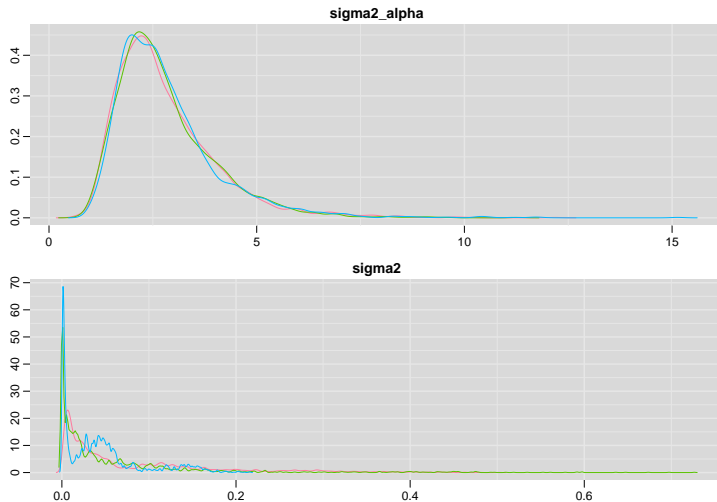
  # Group-level variance
  sigma_alpha ~ dexp(1)
  tau_alpha <-pow(sigma_alpha,-2)
  sigma2_alpha <-pow(sigma_alpha,2)

# Likelihood
  for (t in 1:T){
    for (i in 1:ngroups){
      count[t,i] ~ dpois(lambda[t,i])
      epsilon[t,i] ~ dnorm(0,tau)
      log(lambda[t,i]) <- alpha[i] + epsilon[t,i]
    }
  }
}
```



# Partitioning results

```
library(mcmcplots)  
denplot(out3, parms=c("sigma2_alpha", "sigma2"))
```



## Offsets: a sequencing example

We have 5 samples of 1245, 1145, 987, 1342, and 1012 sequence reads total. Each sample contains DNA sequence counts for 15 species. The total number of counts are determined by the sequencing depth – not how much DNA we have.

- The data reads for the first sample (sorted by size):

c(1056, 103,44, 35, 2, 1, 1, 1 1,1,0,0,0,0,0)

- Second sample

c(821,248,37,17,12, 5, 3, 1, 1, 0,0,0,0,0,0)

## Offsets: models

We code  $\log(\text{total number of reads as an offset}) = o_i$ . What does that mean?  $i$  = sample index,  $j$  = species index

$$Y_{i,j} = \mathcal{P}(\exp(o_i + \alpha_j))$$

## Offsets: models

We code  $\log(\text{total number of reads as an offset}) = o_i$ . What does that mean?  $i$  = sample index,  $j$  = species index

$$Y_{i,j} = \mathcal{P}(\exp(o_i + \alpha_j))$$

$o_i$  is not estimated. It is plugged-in. What does it mean?

Let's say  $N_i = \sum_j Y_{i,j}$ . We have then

$$Y_{i,j} = \mathcal{P}(N_i \exp(\alpha_j))$$

Thus we model  $\frac{Y_{i,j}}{\sum_j Y_{i,j}}$  the fraction of species  $j$  in sample  $i$ .

## Goodness of fit – more info

- We have seen *graphical* posterior predictive checks
- Bayesian p-value  $\mathbb{P}(T(y^{\text{rep}}) > T(y)|\text{model})$ . Should be around 0.5, close to 0 or 1 is bad. *A worked example*

```
# Calculate RSS
for (i in 1:ndata){
  resid[i] <- (Y[i] - lambda[i])/sqrt(lambda[i])
  SS[i] <- pow(resid[i],2)
}
# Calculate RSS for replicated data
for (i in 1:ndata){
  resid.rep[i] <- (Y.rep[i] - lambda[i])/sqrt(lambda[i])
  SS.rep[i] <- pow(resid.rep[i],2)
}
bayes_pval <- mean(sum(SS)>sum(SS.rep))
```

- DHARMA R package with more ideas on model checking, including Dunn-Smyth residuals