GLM(M)s for counts

Bayesian statistics 6 – generalized linear models for count data

Frédéric Barraquand (CNRS, IMB)

22/11/2021

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.

Figure 1: 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. $Y \sim \mathcal{B}(n, p) \to \mathcal{P}(np)$ for large n and small p. Even if actually there are n Bernouilli trials with varying probability p_i.

Prussian army horse-kick data

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

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 Poisson(λ)-distributed values y_i with $\Gamma(\alpha, \beta)$ prior on λ, the posterior distribution for λ 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
```

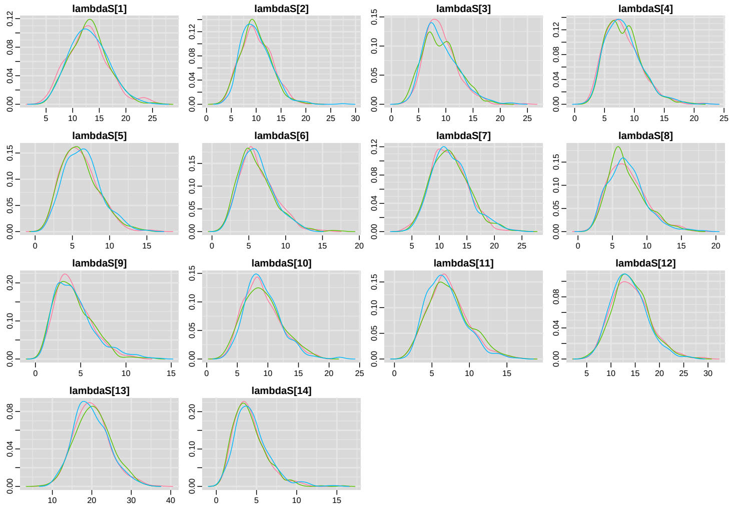
```
## Inference for Bugs model at "poisson.anova.txt", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded), n.thin = 2
##  n.sims = 1500 iterations saved
##             mu.vect sd.vect   2.5%    25%    50%    75%  97.5%  Rhat n.eff
## lambdaS[1]   12.996   3.628  6.642 10.389 12.852 15.257 20.647 1.005   970
## lambdaS[2]   10.008   3.032  4.890  7.928  9.708 12.039 16.313 1.001  1500
## lambdaS[3]    9.108   3.137  4.051  6.856  8.673 10.993 15.989 1.004  1500
## lambdaS[4]    8.171   2.982  3.714  5.947  7.742  9.934 15.181 1.001  1500
## lambdaS[5]    6.180   2.468  2.295  4.343  5.927  7.644 11.601 1.005   390
## lambdaS[6]    6.115   2.384  2.420  4.433  5.784  7.515 11.436 1.002  1500
## lambdaS[7]   11.996   3.372  6.287  9.528 11.692 14.107 19.581 1.001  1400
## lambdaS[8]    7.114   2.705  3.050  5.192  6.681  8.649 13.615 1.000  1500
```

Running the model for horse-kick data III

```
## lambdaS[9]     4.256   2.100   1.244   2.703   3.896   5.417   9.166 1.001  1500
## lambdaS[10]    9.159   3.007   4.296   6.983   8.738  10.901  15.853 1.002  1500
## lambdaS[11]    7.112   2.643   2.884   5.254   6.785   8.626  13.226 1.003   630
## lambdaS[12]   14.031   3.751   7.669  11.435  13.723  16.392  22.148 1.002   860
## lambdaS[13]   20.150   4.440  12.194  16.980  19.804  22.960  29.880 1.003   760
## lambdaS[14]    4.250   2.038   1.256   2.815   3.923   5.311   9.243 1.005   430
## deviance     418.048   5.250 409.450 414.143 417.574 421.259 429.663 1.003   740
##
## 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 = 431.8
## DIC is an estimate of expected predictive error (lower deviance is better).
```

```r
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

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

(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])
    }
 }
```

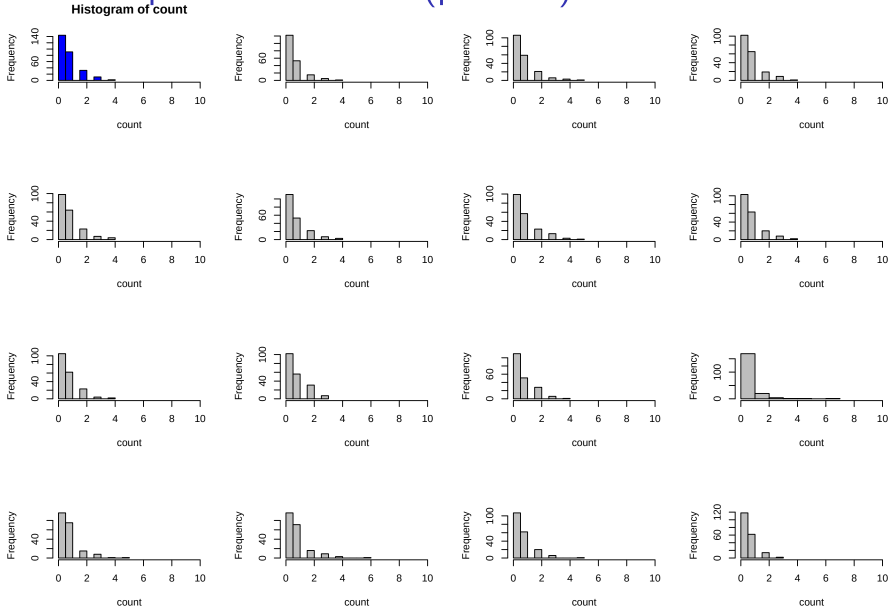
Posterior predictive checks (practice) I

```r
#library(RColorBrewer)
str(out$BUGSoutput$sims.list$count.rep)
```

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

```r
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



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 $Y_i \sim \mathrm{NB}(r, p)$ 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]))
```

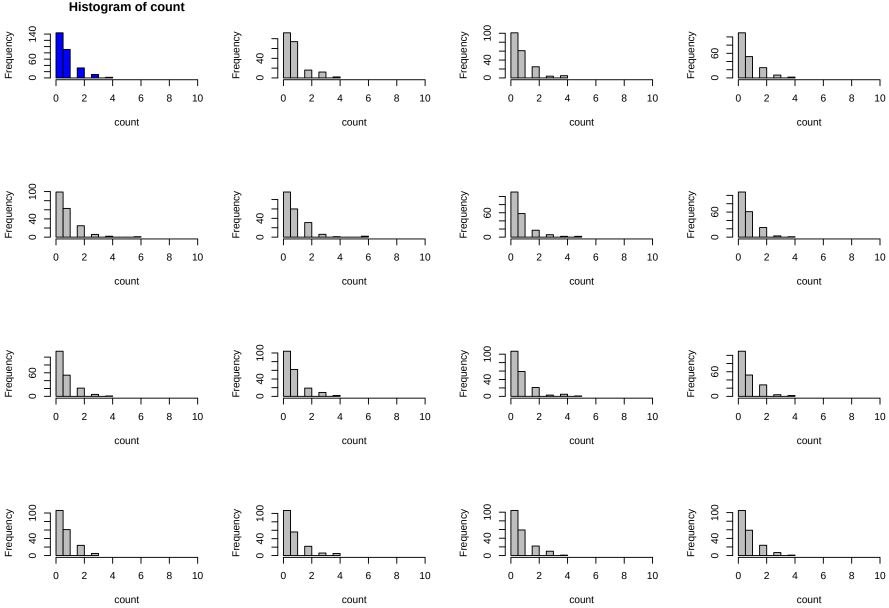
Posterior predictive checks again I

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

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

```r
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



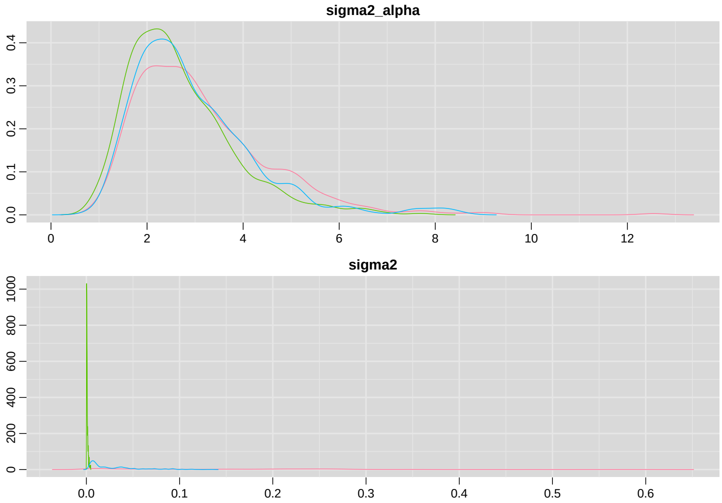
PLN mixed model: estimating intercorps variance I

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

```r
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(total number of reads as an offset) $= o_i$. What does that mean?

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

Offsets: models

We code log(total number of reads as an offset) $= o_i$. What does that mean?

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

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

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

$$Y_{i,j} = \mathcal{P}(N_i \exp(\alpha_{j[i]}))$$

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

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