`bizicount`: Bivariate Zero-Inflated Count Copula Regression using R

John M. Niehaus
Texas A&M University

Lin Zhu
Renmin University of China
Texas A&M University

Scott J. Cook
Texas A&M University

Mikyoung Jun
University of Houston

Abstract

Two common issues arise in regression modelling of bivariate count data: i) dependence across outcomes, and ii) excess zero counts (i.e., zero inflation). However, there are currently few options for estimating bivariate zero-inflated count regression models in R. Therefore, we present an R package, **bizicount**, to enable researchers to easily estimate bivariate zero-inflated count copula regression models. By using copulas to model the dependence across outcomes, researchers do not have to make assumptions about the multivariate (and zero-inflated) structure relating their count variables to one another. Instead, they are only required to make familiar assumptions about the marginal distribution of each outcome variable, which should enable wider use of our approach. Below we present our proposed estimator, detail its advantages over existing alternatives, and demonstrate the use of the corresponding functions for bivariate modeling of terrorism data from Nigeria.

1 Motivation

Count data often contain a higher proportion of zeroes than expected under commonly used distributions such as the Poisson or Negative binomial. These "excess" zeroes have led to the development of statistical models that account for zero inflation in count data (e.g., Mullahy, 1986; Lambert, 1992), which are now widely used by applied researchers. While these zero-inflated count data models are available in most commonly-used statistical software packages, these are all for single count (i.e., univariate) processes.[1] For bivariate (or multivariate) count processes, there are few options for modelling excess zeroes. As such, researchers are often forced to either: a) ignore the "excess" zeroes in the data, or b) ignore the dependence across counts.

The statistical consequences of erroneously neglecting zero inflation or assuming independence are widely known. When (univariate) count data are zero inflated – ex. if a separate process produces excess "structural" zeroes – one-part count models (i.e., Poisson regression models) that neglect zero inflation are known to be biased and result in inefficient estimators. Similarly, when bivariate count data is assumed to be independent, then any unmodeled dependence produces statistically inefficient estimators of the model paramaters, which can also result in inferential errors.

When researchers with bivariate zero-inflated count data neglect both issues, they inherit both of these problems: unmodeled excess zeroes *and* unmodeled count dependence. Moreover, addressing only one issue – i.e., modeling zero inflation *or* dependence, but not both – does not ensure better-behaved estimators. For example, using widely-available bivariate count estimators – e.g., Stata's **bivcnto**, or R's **GJRM** – in the presences of unmodeled excess zeroes risks biased estimates of both the covariate coefficients and the dependence parameter. The bias in the dependence paramater also produces inaccurate (and over confident) estimates of the standard errors for the covariate coefficients in the marginal count models. That is, by addressing only the cross-outcome dependence, we produce an estimator of the covariate coefficients that is biased *and* overconfident, thereby *increasing* the risk of inferential errors.

Despite recent developments in zero-inflated, bivariate count regression models (Arab et al., 2012; Faroughi and Ismail, 2017*a*,*b*; Gurmu and Elder, 2008; Li et al., 1999; Wang, 2003; Wang et al., 2003), most of these estimators are not yet widely available to applied researchers in commonly used statistical software. The only software implementations we can locate that permit bivariate zero-inflated count regression are the SAS®-macro, `%bicount` (Chou and Steenhard, 2011), which supports only a Frank copula, and the R package **bivpois** (Karlis and Ntzoufras, 2005), which is archived on CRAN, does not permit negative dependence for zero-inflation, and also assumes a specific joint distribution.

[1] For R, see the `pscl` package (Jackman et al., 2020); for Stata, see the `zip` command (StataCorp, 2019); for SAS, see SAS Institute (2020); for Python, see Seabold and Perktold (2010). SPSS relies on a R plug-in that calls up the `pscl` package.

Therefore, in this article we present the **bizicount** R package for estimating bivariate zero-inflated count copula regression models. Our work builds on recent developments in copula regression models for discrete outcomes (e.g., So, Lee and Jung, 2011; Yang, Frees and Zhang, 2020), which do not require assumptions about the form of the joint distribution. Instead, one only needs to make familiar assumptions on the marginal distributions of each outcome. Our copula-based strategy is also advantageous in that allows researchers to specify different marginal distributions of each outcome, naturally accommodating mixed-process models. After reviewing several approaches to modeling bivariate, zero-inflated count distributions, we discuss existing software, detail our implementation, and provide an empirical demonstration analyzing terror attacks in Nigeria.

2 Modeling Dependence in Bivariate Counts

Researchers tend to model bivariate counts by either: i) specifying the full joint distribution, or ii) using copulas to model the dependence between the counts. In the following, we briefly survey existing research on both approaches, then discuss how these can be generalized to allow for zero-inflation.

2.1 Bivariate Count Mass Functions

In the first of these approaches, one often specifies the joint distribution for bivariate count data in one of two ways: i) the trivariate reduction method, or ii) the multiplicative factors method.

The trivariate reduction method (Kocherlakota and Kocherlakota, 1992; Lai, 1995, 8) can be used to arrive at a bivariate Poisson distribution, as detailed by Holgate (1964) and Marshall and Olkin (1985, 334). First let

$$\begin{aligned} Y_1 &= X + U, \\ Y_2 &= V + U, \end{aligned}$$

with X, V, and U independent Poisson random variables with rate λ_X, λ_V, and λ_U, respectively. Then, it follows that the joint distribution of Y_1 and Y_2 is bivariate Poisson with

$$f_{Y_1,Y_2}(y_1, y_2) = P(Y_1 = y_1, Y_2 = y_2) = e^{(-\lambda_X - \lambda_V - \lambda_U)} \sum_{c=0}^{\min(y_1,y_2)} \frac{\lambda_U^c \lambda_X^{y_1-c} \lambda_V^{y_2-c}}{(y_1-c)!(y_2-c)!c!}, \quad (1)$$

for $y_1, y_2 = 0, 1, \ldots, \infty$. The joint distribution has Poisson margins with parameters $\lambda_1 = \lambda_X + \lambda_U$ and $\lambda_2 = \lambda_V + \lambda_U$, and $\text{Cov}(Y_1, Y_2) = \lambda_U$. Thus, this formulation permits only positive dependence in the two counts.

As an alternative to the trivariate reduction approach, and in order to allow the dependence between outcomes to be unrestricted, Lakshminarayana, Pandit and Srinivasa Rao (1999) propose a bivariate Poisson distribution using the approach discussed by Sarmanov (1966) and Ting Lee (1996). Specifically, a bivariate Poisson mass function is arrived at as the product of Poisson marginals with a multiplicative factor

$$f_{Y_1,Y_2}(y_1,y_2) = \left[\frac{e^{(-\lambda_1-\lambda_2)}\lambda_1^{y_1}\lambda_2^{y_2}}{y_1!y_2!}\right]\left[1+\alpha\left(e^{-y_1}-e^{\lambda_1 c}\right)\left(e^{-y_2}-e^{-\lambda_2 c}\right)\right], \tag{2}$$

which has Poisson margins with parameters λ_1 and λ_2, and $\text{Cov}(Y_1,Y_2) = \alpha\lambda_1\lambda_2 c^2 e^{-c(\lambda_1+\lambda_2)}$, where $c = 1-e^{-1}$. In this case, the sign on the covariance depends on the estimated sign of α, which can be positive, negative, or zero.

These two approaches are readily extended to permit over- and under-dispersion in the form of bivariate negative binomial (Famoye, 2010*b*; Marshall and Olkin, 1990) or bivariate generalized Poisson distributions (Famoye, 2010*a*). Each of these bivariate probability mass functions (PMF) can also be further extended to permit zero-inflation in the counts. Recall that the PMF for a *univariate* zero-inflated count distribution is

$$f_{ZI}(y) = \begin{cases} \psi + (1-\psi)\cdot f_C(y), & \text{for} \quad y = 0, \\ (1-\psi)\cdot f_C(y), & \text{for} \quad y > 0, \end{cases} \tag{3}$$

and the corresponding cumulative distribution function (CDF) is then

$$F_{ZI}(y) = \psi + (1-\psi)\sum_{i=0}^{y} f_C(i), \tag{4}$$

with $y \in \mathbb{Z}^{\geq 0}$ and $\psi \in [0,1]$, the probability of zero-inflation. Here $f_C(\cdot)$ denotes the PMF of some arbitrary count distribution, e.g., the Poisson, negative binomial, or generalized Poisson. Then, combining the intuition of the univariate zero-inflated distribution from **??** with the bivariate Poisson PMF from **??**, we have a PMF for the BZIP based on multiplicative factors (Faroughi and Ismail, 2017*b*):

$$f_{Y_1,Y_2}(y_1,y_2) = \begin{cases} \psi + (1-\psi)\cdot e^{-\lambda_1-\lambda_2}\left[1+\alpha\prod_{j=1}^{2}(1-e^{-c\lambda_j})\right], & \text{if} \quad y_1,y_2 = 0, \\ (1-\psi)f_{BP_m}(y_1,y_2), & \text{otherwise.} \end{cases} \tag{5}$$

Here, ψ is again the probability of zero inflation, and f_{BP_m} is the bivariate Poisson PMF based on multiplicative factors as defined in **??**.[2] This can be generalized to a bivariate regression modelling context in the usual way: for observations i and margins j we specify

[2]For a bivariate, zero-inflated distribution based on the trivariate reduction method, see Li et al. (1999).

$\psi_{ij} = \Lambda(Z_{ij}\gamma_j)$ and $\lambda_{ij} = \exp(X_{ij}\beta_j)$, where $\Lambda(\cdot)$ is the CDF of the logistic or standard normal distribution, λ_{ij} is the conditional mean of the jth marginal count distribution (f_C in **??**), and both X_{ij} and Z_{ij} are covariate vectors for the j^{th} margin with parameter vectors β_j and γ_j to be estimated, respectively.

2.2 Copula Regression

In contrast, copula regression provides an alternative approach for modeling dependence and mixed-zero generation in count data that does not require assumptions regarding the joint distribution. In short, copulas are functions that specify an arbitrary joint CDF as a function of uniform marginals and a dependence parameter. This carries the benefit of requiring no assumptions regarding the form of the joint distribution; rather, scholars assume the form of the dependence between the two margins in choosing a copula function, and input uniform marginals into this copula function. These uniform marginals are derived from the probability integral transform, and therefore require assumptions about the marginal distributions rather than the joint distribution. However, because information about the marginals is typically better known than for the joint distribution, researchers often find that the assumptions required for copula regression are more tenable than assuming knowledge on the complete form of a bivariate mass function (as in **??**). Importantly, the marginal distributions can also differ from each other. For example, in the bivariate case, researchers can specify one margin as zero-inflated negative binomial, and the other as Poisson, thus permitting greater flexibility than restricting both margins to be the same.

To see this, note that copulas leverage the probability integral transform in order to express an arbitrary (and often unknown) multivariate CDF as a copula function of uniform marginal distributions (Sklar, 1959, 1973). Specifically, the probability integral transform states that given a strictly increasing, continuous CDF, F_X, and the random variable, $Y = F_X(X)$, then $Y \sim U(0, 1)$:

$$\begin{aligned} F_Y(y) &= \Pr(Y \leq y) \\ &= \Pr\{F_X(X) \leq y\} \\ &= \Pr\{X \leq F_X^{-1}(y)\} \\ &= F_X\{F_X^{-1}(y)\} \\ &= y, \end{aligned}$$

which implies F_Y is the CDF of the standard uniform distribution. Then, given a random vector $\boldsymbol{X} = (X_1, X_2, \ldots, X_n)$ with continuous, monotone increasing marginal CDFs

$(F_1, F_2, \ldots, F_n)$, we can use the above result to deduce that

$$\{F_1(X_1), F_2(X_2), \ldots, F_n(X_n)\} = (U_1, U_2, \ldots, U_n) = \boldsymbol{U}$$

which is to say that the random vector $\{F_1(X_1), F_2(X_2), \ldots, F_n(X_n)\}$ has uniform margins.

The copula function is then defined as the continuous joint CDF of the variables in the vector $\boldsymbol{U}$ above (Cameron et al., 2004; Trivedi and Zimmer, 2007, p.10):

$$\begin{aligned} F(X_1, X_2, \ldots, X_n) &= F\{F_1^{-1}(u_1), F_2^{-1}(u_2), \ldots, F_n^{-1}(u_n)\} \\ &= \Pr(U_1 \leq u_1, U_2 \leq u_2, \ldots, U_n \leq u_n) \\ &= C(u_1, u_2, \ldots, u_n) \end{aligned}$$

where $C(\cdot)$ is the copula function. Thus, working backwards we can see that if $\boldsymbol{U} \sim C$, then $\{F_1^{-1}(u_1), F_2^{-1}(u_2), \ldots, F_n^{-1}(u_n)\} \sim F$, and we can write a multivariate CDF in terms of a copula function and uniform marginals, although we still assume the marginal distributions in order to then apply the inverse transform (Trivedi and Zimmer, 2007, p.10).

While the preceding illustration of copula functions assumes continuous marginal distributions, we are interested in discrete (zero-inflated) count marginals. In the discrete case, the lack of continuity gives rise to copulas that are unique only over the Cartesian product of the ranges of the marginals; that is, over $ran(F_1) \times ran(F_2) \times \cdots \times ran(F_n)$, where "$\times$" denotes the Cartesian product, and *ran* is the range of a random variable (Genest and Nešlehová, 2007; Karlis, 2016; Nikoloulopoulos, 2013). These models therefore have the potential for identification problems. Additionally, the dependence between outcomes is now affected by the choice of margins (Genest and Nešlehová, 2007). Fortunately, we are interested in copula *regression* where the issue of identification is less salient because we model the conditional expected mean as a function of (often continuous) covariates, with this mean itself being continuous (Trivedi and Zimmer, 2017). Moreover, even if non-uniqueness persists, the "...lack of uniqueness is not a problem in practical applications as it implies that there may exist two copulas with identical properties" (Karlis, 2016, p.410).

Keeping the above review of copulas in mind, a *bivariate* copula CDF with count margins can be written as (Cameron et al., 2004; Joe, 1997; Marshall and Olkin, 1990)

$$\begin{aligned} F(y_1, y_2) &= C\{F_1(y_1), F_2(y_2); \rho\} \\ &= C(u_1, u_2; \rho), \end{aligned} \tag{6}$$

where ρ is the dependence parameter to be estimated. In order to obtain the joint PDF

from the joint CDF in **??**, we would typically take derivatives in the usual way, i.e., $\partial^2 F / \partial u_1 \partial u_2$; however, the data are assumed to be drawn from a zero-inflated count distribution, which is discrete, and therefore non-differentiable. Instead, we arrive at the joint PMF by taking backwards finite differences as an analog to the derivative of the CDF (So, Lee and Jung, 2011; Trivedi and Zimmer, 2007):

$$\begin{aligned} f(y_1, y_2) &= F(y_1, y_2) - F(y_1 - 1, y_2) - F(y_1, y_2 - 1) + F(y_1 - 1, y_2 - 1) \\ &= C\{F_1(y_1), F_2(y_2); \rho\} - C\{F_1(y_1 - 1), F_2(y_2); \rho\} - \\ &\quad C\{F_1(y_1), F_2(y_2 - 1); \rho\} + C\{F_1(y_1 - 1), F_2(y_2 - 1); \rho\} \\ &= c\{F_1(y_1), F_2(y_2); \rho\}, \end{aligned} \tag{7}$$

where $f(\cdot)$ is the joint PMF of the original data, $c(\cdot)$ is the copula PMF, $F_j(\cdot)$ are the marginal CDFs, F is the joint CDF, and C is the copula CDF. Using the joint PMF as derived above in **??**, we can proceed with estimating the parameters of the distribution by maximum likelihood in the usual way.[3]

The only remaining question at this point regards the form of the copula CDF, C. Although a large literature exists on the various copulas and their properties, we focus on two widely-used bivariate copulas in our discussion (and subsequent software): the Gaussian and Frank copulas.[4] Both of these are symmetric but with different shapes, particularly in the tails. The Gaussian copula takes the form

$$\begin{aligned} C_G(u_1, u_2; \rho) &= \Phi_\rho \left[\Phi^{-1}\{F_1(y_1)\}, \Phi^{-1}\{F_2(y_2)\}; \rho \right] \\ &= \Phi_\rho \left\{ \Phi^{-1}(u_1), \Phi^{-1}(u_2); \rho \right\} \end{aligned} \tag{8}$$

where F_j are the marginal, zero-inflated count CDFs as defined in **??**, Φ^{-1} is the quantile function of the standard normal distribution, and Φ_ρ is the CDF of the standard bivariate normal distribution with correlation $|\rho| < 1$. Thus, the copula CDF is the standard bivariate normal, and the dependence parameter is the correlation for this CDF.

Alternatively, the Frank copula comes from the Archimedean class of copulas and

[3] Note that taking finite differences and then using maximum likelihood is one of a few approaches to solving the problem of discrete marginal distributions. For discussion of additional approaches not utilized here, see Inouye et al. (2017).

[4] For detailed expositions on copulas, see Joe (1997); Nelsen (2007); Trivedi and Zimmer (2007). Regarding count copulas, see Genest and Nešlehová (2007).

permits unbounded dependence, with the bivariate form

$$
\begin{aligned}
C_F(u_1, u_2; \rho) &= -\rho^{-1} \ln \left[1 + \frac{\{e^{-\rho F_1(y_1)} - 1\}\{e^{-\rho F_2(y_2)} - 1\}}{e^{-\rho} - 1} \right] \\
&= -\rho^{-1} \ln \left\{ 1 + \frac{(e^{-\rho u_1} - 1)(e^{-\rho u_2} - 1)}{e^{-\rho} - 1} \right\},
\end{aligned} \tag{9}
$$

where the F_j are the same as in **??**, and $\rho \neq 0$ is the dependence parameter. For interested readers, visual depictions of each copula can be found in **??** in the Appendix.

For both the Gaussian and Frank copulas, we can arrive at a bivariate, zero-inflated count copula regression model if we assume that the F_j are the CDFs of a univariate zero-inflated count distribution as defined in **??**. From here we can link in covariates by specifying $\psi_{ij} = \Lambda(Z_{ij}\gamma_j)$ and $\lambda_{ij} = \exp(X_{ij}\beta)$, where $\Lambda(\cdot)$ is the CDF of the logistic or standard normal distribution, λ_{ij} is the conditional mean of the jth marginal *count* distribution (f_C in **??**), and both X_{ij} and Z_{ij} are covariate observation vectors for the j^{th} margin with parameter vectors β_j and γ_j, respectively. After adding covariates, we can plug-in the resulting equation from (**??**) or (**??**) to the PMF from finite-differencing as shown in **??**, and then estimate the parameter vectors γ_j and β_j by maximimum likelihood.

In sum, to model bivariate zero-inflated count data analysts can use copula regression, which requires only assumptions about the marginal distributions and a copula function. The specific steps for arriving at a particular model are:

1. Specify a pair of marginal distributions
2. Select a copula distribution
3. Take finite differences over the copula CDF to arrive at the PMF as in **??**
4. Take logarithms and use maximum likelihood to estimate parameters

3 Available Software

Current software is available to model bivariate count data using the strategies discussed above. Many of these implementations require researchers to assume a joint distribution of the outcomes, which can be limiting for the reasons given above. Fewer implementations are available using copula-based approaches, and only one that we are aware of – a SAS® -macro – also accounts for zero-inflation. In the following we discuss existing software and detail the benefits of our package over available alternatives.

3.1 Non-Copula Based Software

When zero-inflation is not a concern, researchers with bivariate count data have several non-copula based options available to them. First, the R package **bpglm** (Chowdhury and Islam, 2020) fits bivariate Poisson and zero-truncated bivariate Poisson models. This package is not actively maintained on CRAN at the time of this writing, however, source code is available on GitHub.[5] For similar functionality in Stata, the package **bivcnto** (Xu and Hardin, 2016) uses maximum likelihood to estimate bivariate Poisson and Negative Binomial regression models with the joint PMFs defined in Famoye (2010*b*) and Marshall and Olkin (1985). Lastly, the SAS®-macro **%bicount** (Chou and Steenhard, 2011) fits bivariate count models to a variety of assumed joint PMFs.

When zero-inflation *is* a concern, users can again use the SAS-macro **%bicount**, or can rely on the **bivpois** R package (Karlis and Ntzoufras, 2005). The latter of these two estimates both bivariate Poisson, and bivariate zero-inflated Poisson models to an assumed joint PMF derived from the trivariate reduction method.[6] Despite offering a widely-available means to estimate bivariate zero-inflated count models, the package suffers some drawbacks. Namely, it permits only positive dependence, assumes the form of the joint distribution of the data, and requires both margins to be of the same functional form (i.e., both margins are Poisson or zero-inflated Poisson).[7]

3.2 Copula Based Software

Beyond packages that assume a joint distribution, there are also implementations of copula regression models available in most widely-used statistical software platforms. As copula modeling is a broad literature, we limit ourselves to discussion of copula *regression* software. In SAS®, discrete- and mixed-margins copula regression have been done using the `nlmixed` procedure (Chen and Hanson, 2017). Alternatively, both the SAS **%bicount** macro and Stata **bivcnto** package can again be used, this time for copula-based count regression. In SAS, all common count margins are supported, including Poisson, negative binomial, and generalized Poisson, as well as their zero-inflated, truncated, and censored counterparts; however, only the Frank copula is supported. Stata's **bivcnto** supports Poisson, negative binomial, and generalized Poisson margins, and has options for the Frank, Gaussian, and Kimeldorf-Sampson copulas. Notably, **bivcnto** does not support zero-inflated count margins.

In R, users have several options for copula regression. First, the **copulaRegression** package (Kraemer, 2014) fits gamma and zero-truncated Poisson GLMs via copula, al-

[5]For **bpglm**, visit https://github.com/chowdhuryri/bpglm.
[6]In fact, it permits arbitrary diagonal inflation, not just zero-inflation.
[7]The code for this package is not on CRAN. To use it, visit http://www2.stat-athens.aueb.gr/ jbn/papers/paper14.htm.

though the package is not being maintained on CRAN as of this writing.[8] Second, the **CopulaCenR** package (Sun and Ding, 2019) fits bivariate copula regressions to both parametric and nonparametric censored margins, such as Weibull or Cox distributions, to name only a few. Third, the **GJRM** package fits a wide variety of copula distributions for dependent, sample-selected, censored, and truncated marginals, all using a trust region algorithm for the likelihood search. Available on CRAN, this package also permits Poisson and negative binomial margins, however, it does not have allow for zero-inflation.[9] Additionally, the **copCAR** package (Goren and Hughes, 2021; Hughes, 2015) fits bivariate copula models to areal data with spatial dependence, permitting binomial, Poisson, and negative binomial margins, as well as offering three estimation approaches.[10] Lastly, the **gcmr** package (Masarotto and Varin, 2017) fits Gaussian copula regression models to a single outcome, instead modelling the dependence across observations either serially or spatially.

Despite this seeming abundance of options, for *copula-based* bivariate *zero-inflated* count regression models, the *only* software implementation is the SAS® macro **%bicount**, which, in addition to fitting models to assumed joint distributions, also permits zero-inflated count copula regression models. However, as of this writing it only supports the Frank copula. Moreover, although several open-source R packages exist for copula regression, we are unable to find any such implementation that permits zero-inflated count marginal distributions. Because of the already-mentioned benefits to using copulas for dependence modeling, and to make such tools more widely available for researchers with zero-inflated count margins, we therefore provide a set of R functions to estimate copula-based zero-inflated bivariate count models.

4 Implementation in R

The primary function in the **bizicount** package is `bizicount`:

```
bizicount(fmla1, fmla2, data, cop = "gaus", margins = c("pois", "pois"),
    link.ct = c("log", "log"), link.zi = c("logit", "logit"), starts = NULL,
    keep = F, subset, na.action, weights, frech.min = 1e-7, pmf.min = 1e-7,
    ...)
```

In addition to this function, we provide functions for random number generation, PMF, CDF, and quantile evaluation for the univariate zero-inflated Poisson and negative binomial distributions. Importantly, the **copula** package in R (Yan, 2007) can then be used to generate from the bivariate copula distribution with zero-inflated count margins.

[8] Old source code available at https://rdrr.io/cran/CopulaRegression/.
[9] For further details, see Marra and Radice (2017) and van der Wurp et al. (2019).
[10] Some of these estimation approaches are further discussed in Inouye et al. (2017).

Additionally, we provide an additional regression function for univariate zero-inflated count models; however, as all of these functions except `bizicount` are not the main focus of this article, we detail them briefly in the Appendix. Finally, the `frech.min` and `pmf.min` parameters are included for computational purposes, but will rarely require alteration; as such, we discuss them further in the Appendix.

4.1 Data Input

To estimate a model using `bizicount`, users must first input a multi-part formula for each margin (`fmla1`, `fmla2`) (Zeileis and Croissant, 2010), and will typically supply a data frame containing the required data. If a margin is zero-inflated (detailed below), its formula will use a vertical "bar" (`|`), and be similar to `y_j ~ x_j | z_j`. If no margin is zero inflated, then the formula simplifies to `y_j ~ x_j`. Each marginal formula supports offsets in each part of the formula specification; for example, `y_j ~ x_j + offset(count_offset_variab` Finally, `weights` can be specified, and analyses can be restricted to a `subset` of observations as desired. All together then, users are unrestricted in terms of which covariates they input in each formula, that is, the covariates for each margin can be the same or different, as well as offsets.

4.2 Copula and Marginal Distributions

After specifying their formulas and any optional features (e.g., offsets, weights, subsets), users then decide on either a Frank (`"frank"`) or Gaussian (`"gaus"`) copula via the `cop` parameter (partial matching supported), with the details of each copula's form specified in **??** and **??**, respectively. Then, a character vector of marginal distributions is specified via the `margins` parameter, including the Poisson as `"pois"`, the negative binomial as `"nbinom"`, and their zero-inflated counterparts, `"zip"` and `"zinb"` respectively. Clearly then, and despite the name `bizicount`, the function permits zero-inflated margins but does not require them, as users are free to select their non-inflated counterparts. After selecting marginal distributions, users specify the count and zero-inflated link-functions from **??** as length-two character vectors via `link.ct` and `link.zi`, with the two elements of each vector corresponding to one of the two margins. The zero-inflated portion permits links from `"logit"`, `"probit"`, `"cauchit"`, `"log"` or `"cloglog"`, while the count portion permits one of `"log"`, `"identity"` or `"sqrt"`.

4.3 Starting Values

Users also have the option to input a custom set of starting values for the likelihood optimization via the `starts` parameter, or by default, have the function choose starting values for the them. If no starting values are supplied by the user, the function auto-

matically uses the estimates of coefficients from a univariate fit for each margin. The dependence parameter's starting value is set as the Spearman correlation between the two outcomes.

4.4 Other Options and Recommendations

Finally, users are permitted increased control over the optimization procedure via the variable arguments `...` parameter. The `bizicount` function optimizes the likelihood using R's `nlm` function, which comes by default with R. Users can pass most any option to `nlm` via the `...` argument in `bizicount`. Exceptions include inverting the function scale, as well as opting not to include the Hessian matrix in output; attempts at alteration of these two options are flagged with a warning, and not heeded.

One key parameter in `nlm` is `stepmax`; this controls the maximum allowable step-size used in the quasi-Newton derivative-based optimization process. We have found that often this parameter may need to be tuned in order to overcome some convergence issues, and invertability or non-negative definiteness of the Hessian matrix. Specifically, users may find it useful to reduce `stepmax` when faced with convergence or singularity issues. To do so, users must simply put `stemax = X` into the `bizicount()` call, where `X` is a positive number. That in mind, users are directed to the help files for `nlm` for further details on fine-tuning the likelihood search, as all arguments to `nlm` can be altered by specifying them in `bizicount`.

Lastly, because of the fact that `bizicount` is using numerical estimates of the gradient and Hessian during the likelihood search, we strongly recommend centering and scaling covariates prior to estimating parameters. Through our own experiences with the function, ensuring that all variables are on the same scale tends to both speed up the optimization process, and prevents some errors that occur when taking excessively large steps along one dimension of the parameter space.

4.5 Generic Methods and Model Object

The `bizicount` function returns an S3 object of class **bizicount** that has a host of generic methods. Table 1 below includes short descriptions of each.

In addition to having these associated methods, the **bizicount** class of objects is a list containing, among other things, the numerical Hessian matrix, the numerical gradient at convergence, the covariance matrix, and several matrices containing the coefficient estimates and their asymptotic standard errors, corresponding p-values, and z-scores. To conserve memory, the `keep` option defaults to `FALSE`, meaning that the design matrices and matrix of outcomes are not kept in the outputted model object. Importantly, `keep` must be set to `TRUE` if users want to do post-estimation diagnostics, or call the `fitted` function.

function	Description
print	Prints the simplified vector of coefficient estimates
summary	Returns same model, gives greater details in print
coef	Returns a vector of model coefficients
fitted	Returns $n \times 2$ matrix of fitted values for each margin
logLik	Returns log-likelihood at convergence point
nobs	Returns number of observations used in estimation
AIC	Returns Akaike Information Criterion (Akaike, 1973)
BIC	Returns Schwartz-Bayesian Information Criterion (Schwarz, 1978)
vcov	Returns the covariance matrix of model parameters
extract	Returns a **texreg** (Leifeld, 2013) object
simulate	Returns simulated matrices for use in **DHARMa** (Hartig, 2020)
make_DHARMa	Returns a list of DHARMa-class objects for the model

Table 1: Generic Methods for **bizicount** objects. Note that the extract function requires an S4 object, so we define an S4 **bizicount** object for use with texreg, while all other generics expect and S3 object. The results of the simulate function can be input to the make.DHARMa function, as detailed later in **??**.

4.6 Producing Tables and Coefficient Plots with texreg

With a desired set of models in hand, **bizicount** objects are fully compatible with the suite of functions from the **texreg** package. Namely, tables of coefficients can be produced in text, Word, or LaTeX formats, and coefficient plots drawn. Compatibility with **texreg** is achieved by defining an extract method for **bizicount** objects. Note that this extract method prepends an identifier to the beginning of each covariate so as to distinguish the same covariate appearing in each part of the model (ct_ for count, and zi_ for zero-inflation). For example, the var variable in the count portion of a **bizicount** model would be extracted as ct_var, while the same variable appearing in the zero-inflated portion is extracted as zi_var. This approach makes **texreg**ing additional models (i.e., GLMs) alongside **bizicount** objects more easy, as the coefficient vector for the other models can simply be renamed to be consistent with the identifier-prepended names produced by extract. An example of this entire process is demonstrated in **??**, although if users desire to suppress the identifiers, that option is available as well. However, note that suppressing the identifier when **texreg**ing additional models alongside **bizicount** models will often result in a larger than desired table.

Briefly, the extract function has three parameters when used on a **bizicount** object:

```
extract.bizicount(model, CI=NULL, id=T)
```

with model being the **bizicount** model object, CI is a number on the unit interval specifying the desired two-tailed confidence level, and id being TRUE tells the extraction to prepend the identifiers outlined previously.

4.7 Model Diagnostics with DHARMa

After estimating a desired model, users may wish to do various forms of diagnostic testing to assess model fit. In typical GLM settings, this would involve inspecting residuals of some form, such as deviance, Pearson, or Anscombe (Cameron and Trivedi, 2013). However, it is well known that the discrete nature of most outcomes in GLMs makes residual analysis difficult, and often times misleading (Dunn and Smyth, 1996). In an attempt to rectify this gap, R's **DHARMa** package (Hartig, 2020) offers non-parametric, simulation-based techniques to assess model fit, outliers, dispersion, zero-inflation, as well as independence in a host of models. Briefly, this is accomplished by simulating outcomes using the parameters of the fitted model, and inspecting the quantiles of the fitted values in relation to the quantiles of these simulated datasets to determine deviation from what would be expected if the model were true. Users are referred to the detailed vignette on **DHARMa** for additional information and examples.

For **bizicount**-class objects, we provide two useful functions in relation to **DHARMa**'s diagnostic tools. First, the generic `simulate` function is given a corresponding method in our package to simulate the datasets required by **DHARMa**. A list containing two matrices of simulated outcome data for each margin is returned, each of which can be used with **DHARMa**, particularly the `createDHARMa` function. However, because there are two sets of simulations—one for each margin—we we provide a `make.DHARMa` wrapper around `createDHARMa` to first create this list of simulated datasets, and then create a list of **DHARMa** objects from them. That is, users can simply call `make.DHARMa` on a **bizicount** object to then obtain a list of **DHARMa** objects that interface with all of that package's methods, as shown in **??**. However, if users require access to only the simulated values, the `simulate` function makes this simple.

In short, the function has four parameters:

```
make.DHARMa( model, n = 250, seed = 123, method = c("PIT", "traditional") )
```

where users must input a **bizicount**-class `model`, a number of simulations, `n`, a `seed` for these simulations, and the `method` used for obtaining the quantiles of a discrete distribution. As this is simply a wrapper around **DHARMa**'s `createDHARMa` function, users are referred to the corresponding documentation for details on these parameters.

5 Empirical Application: Terrorism in Nigeria

Terrorism scholars are often interested in the determinants of terror attacks perpetrated by a particular terrorist group or set of groups. Because attack frequencies (in a given discrete space-time unit) tend to be the outcome of interest in terrorism research, scholars typically model these attacks using a count distribution. However, two

common issues that arise in this research are often underaddressed: i) dependence between terrorist organizations' attacks, and ii) attacks being generated by a mixture of distinct processes. Regarding dependence, terrorist groups are theorized to both compete (Conrad and Greene, 2015; Kydd and Walter, 2006) and form alliances with other organizations (Horowitz and Potter, 2014), implying that the attacks by one terror organization are likely dependent on those of others. Secondly, researchers may observe zero terrorist attacks for two reasons: terrorist groups may be present in a location but inactive (i.e., stochastic zeroes), or alternatively terror attacks go unreported, no terror groups are present, etc.(i.e., structural zeroes). Thus, studies of terrorism face both problems discussed here: dependence across outcomes and mixed generation of zero counts.

As such, our preferred copula regression approach appears well suited to modeling the attacks of two or more groups. As such, we now demonstrate the utility of `bizicount` in modeling these type of data with a cross-sectional analysis of terror attacks in Nigeria during 2014. During this time period, there were two main terror groups active in Nigeria: Boko Haram and Fulani extremists. We focus on terror attacks from these two respective groups in our subsequent bivariate analyses.

5.1 Data

We analyze two sets of terrorist attacks in Nigeria during 2014, one perpetrated by Boko Haram (`att.bok`), the other perpetrated by Fulani extremists (`att.ful`). These terrorism data come from the National Consortium for the Study of Terrorism and Responses to Terrorism's (START) Global Terrorism Database (GTD) (LaFree and Dugan, 2007; START, 2018), which is an event-level data set cataloguing individual terror attacks.[11] The GTD data provides a wealth of information, including, importantly for us, the perpetrator and the spatial coordinates (i.e., latitude and longitude) of each attack. Using this information, we aggregate the number of events perpetrated by each group in Nigeria during 2014 up to the PRIO-GRID square level (0.5×0.5 decimal degrees). Using the PRIO-GRID as the unit of analysis allows us to easily incorporate existing sub-national covariates from PRIO-GRID 2.0 dataset (Tollefsen et al., 2015; Tollefsen, Strand and Buhaug, 2012). Specifically, we include grid-level data on population (`pop`) (CIESIN & CIAT 2005), percent mountainous terrain (`mtns`) (Blyth et al., 2002), and the latitude and longitude of grid centroids (`xcoord`, `ycoord`), all of which are common in terrorism research. Note that all covariates are mean-centered and scaled by one standard deviation.

[11] For details on coding methodology and criteria, see the GTD codebook.

5.2 Models and Output

In order to compare the bivariate models to their univariate counterparts, we first estimate univariate Poisson and zero-inflated Poisson regression models for attacks by Fulani Extremists (FE) and by Boko Haram (BH). First, defining the base formula and estimating the univariate Poisson models using the `glm` function:

```
 load("empirical_replication/data_replication.RData")

# Univariate Poisson Models
fmla.ful = att.ful ~ pop + mtns + xcoord * ycoord
fmla.bok = update.formula(fmla.ful, att.bok ~ .)

pois.ful = glm(fmla.ful, data=dat, family=poisson(link="log"))
pois.bok = glm(fmla.bok, data=dat, family=poisson(link="log"))
```

Next, we update the formulas for use with our univariate zero-inflated count regression function, `zic.reg` (which is detailed in the Appendix), in order to compare the Poisson to its zero-inflated counterpart, and eventually compare the two to their corresponding bivariate models:

```
# Univariate Zero-Inflated Poisson Models
library("bizicount")

fmla.zi.ful = att.ful ~ pop + mtns + xcoord * ycoord | pop + mtns + xcoord * ycoord
fmla.zi.bok = att.bok ~ pop + mtns + xcoord * ycoord | pop + mtns + xcoord * ycoord

unizi.ful = zic.reg(fmla = fmla.zi.ful,
                    data = dat)
unizi.bok = zic.reg(fmla = fmla.zi.bok,
                    data = dat)
```

The same formulas from the univariate Poisson and univariate zero-inflated Poisson models can then be used as the formulas for each margin of the bivariate Poisson and zero-inflated bivariate Poisson models. In this case, we set `cop="frank"`, meaning we use a Frank copula. Finally, note that the margins are set as `c("pois", "pois")` by default, but are explicitly input here for the marginal Poisson case for clarity.

```
# Bivariate Poisson
bivpois = bizicount(fmla.ful,
                    fmla.bok,
                    data = dat,
```

```
                    cop = "frank",
                    margins = c("pois", "pois"),
                    keep = T
                )

# Bivariate, zero-inflated Poisson, Frank copula
zibivpois = bizicount(fmla.zi.ful,
                      fmla.zi.bok,
                      data = dat,
                      cop = "frank",
                      margins = c("zip", "zip"),
                      keep = T
                )
```

Binding all of the univariate and bivariate models just estimated to a list, we can apply the `texreg` function to this list, which will automatically use the `extract` method that we have defined for **bizicount** objects in order to obtain the necessary information, and output it to a table. Note that due to name mis-matches on covariates, we change the names vector for the `glm` objects to be consistent with the names of the covariates from `zicreg` and `bizicount`, as noted in **??**:

```
# Texreg
library("texreg")

# Rename model coefficients for table
names(pois.ful$coefficients) = paste0("ct_", names(pois.ful$coefficients))
names(pois.bok$coefficients) = paste0("ct_", names(pois.bok$coefficients))

mods = list(pois.ful, pois.bok, unizi.ful, unizi.bok, bivpois, zibivpois)

texreg(mods,
  groups = list("Count Model" = 1:6, "Zero Inflation" = 7:12),
  dcolumn = T,
  digits = 3,
  custom.model.names = c(
    "Poisson: Fulani",
    "Poisson: BH",
    "Univariate ZIP: Fulani",
    "Univariate ZIP: BH",
    "Bivariate Poisson: Fulani",
```

```
    "Bivariate Poisson: BH",
    "Bivariate ZIP: Fulani",
    "Bivariate ZIP: BH"
  )
)
```

The resulting output from `texreg` is presented in Table 2. Focusing first on the respective univariate models (columns 1 - 4), we see that the models accounting for zero inflation (columns 3 and 4) produce significantly different results from those that did not (columns 1 and 2). For the FE sample, observe that failing to account for zero inflation suppresses the effect of `Population` and `Mountains`, as both are insignificant in column 1 and positive and significant in column 3. The zero inflation coefficients in column 3 are also consistent with this interpretation, as both `Population` and `Mountains` are positively related to excess zeroes; as such, when this process is neglected in column 1 the recovered coefficients conflate the positive zero-generating effect and the positive count-generating effect, producing a null finding. Similar patterns obtain in the BH samples (columns 2 and 4), except now the coefficients on `Population` and `Mountains` in the uncorrected case (column 2) are inflated, because these covariates are correlated with *fewer* zeroes for BH (as seen in the zero inflation results of column 4).

Continuing to the bivariate models, we see that accounting for dependence but not zero inflation (columns 5 and 6) produces results quite different to the univariate analog. This is especially true for the FE sample, as both `Population` and `Mountains` are *negative* and significant (column 5), as opposed to the positive (but insignificant) finding in column 1. Note that the dependence parameter is also positive and significant, suggesting that the bivariate estimator is appropriate in this case. Intuitively, however, this does not make much sense, as we know that these respective terror groups operate in different parts of the country. As such, they should have *negative* dependence. This is, in fact, observed in the bivariate zero-inflated Poisson results (columns 7 and 8), which produce a negatively signed, yet insignificant dependence paramater. It seems, therefore, that the earlier positive dependence paramater was due to the unmodeleded zero inflation in these processes.

Here we demonstrate the with applied data that covariate results can be sensitive to modeling choices made by researchers on outcome error dependence and zero inflation. However, with a single sample of observed data, we cannot make strong statements about the relative performance of these estimators. To explore this, we offer a set of simulation experiments in the Appendix. Briefly summarizing those findings, we observe that, as expected, neglecting dependence (i.e., estimating univariate zero-inflated models) results in inefficient estimators, while omitting zero-inflation (i.e., estimating bivariate Poisson models) results in bias for both covariate effects and the dependence parameter. Together, these results suggest that unless researchers can rule out both of these issues ex-ante, our

	Univariate Models				Bivariate Models			
	Poisson		ZIP		Poisson		ZIP	
	FE	BH	FE	BH	FE	BH	FE	BH
Count								
Intercept	-0.691^{***}	-1.954^{***}	0.776^{***}	0.151	-1.262^{***}	-2.379^{***}	0.285^{*}	0.568^{**}
	(0.085)	(0.162)	(0.116)	(0.210)	(0.122)	(0.195)	(0.121)	(0.194)
Population	0.119	0.869^{***}	1.216^{***}	0.443^{***}	-0.441^{*}	0.945^{***}	-0.767^{*}	0.233^{**}
	(0.096)	(0.056)	(0.260)	(0.064)	(0.202)	(0.062)	(0.371)	(0.076)
Mountains	0.056	-0.109^{*}	0.484^{***}	-0.033	-0.441^{***}	-0.050	0.257^{*}	-0.019
	(0.077)	(0.050)	(0.079)	(0.046)	(0.108)	(0.057)	(0.106)	(0.052)
Longitude	0.308^{**}	2.659^{***}	-1.499^{***}	1.627^{***}	0.730^{***}	2.635^{***}	-2.047^{***}	1.217^{***}
	(0.118)	(0.139)	(0.257)	(0.149)	(0.144)	(0.161)	(0.260)	(0.149)
Latitude	-0.213^{*}	0.770^{***}	-0.708^{***}	0.388	-0.693^{***}	0.952^{***}	-1.007^{***}	0.162
	(0.092)	(0.133)	(0.142)	(0.206)	(0.122)	(0.154)	(0.146)	(0.211)
Longitude × Latitude	-0.719^{***}	-0.609^{***}	-1.658^{***}	-0.543^{***}	-1.037^{***}	-0.585^{***}	-1.571^{***}	-0.403^{**}
	(0.128)	(0.108)	(0.280)	(0.120)	(0.170)	(0.120)	(0.307)	(0.127)
Zero Inflation								
Intercept			0.170	1.595^{***}			-3.462^{***}	1.885^{***}
			(0.497)	(0.304)			(0.523)	(0.276)
Population			1.146^{*}	-0.726^{*}			-0.434	-0.758^{**}
			(0.486)	(0.306)			(0.609)	(0.269)
Mountains			0.988^{**}	-0.216			1.735^{***}	-0.153
			(0.318)	(0.204)			(0.407)	(0.207)
Longitude			-3.638^{**}	-0.525			-10.546^{***}	-0.779^{*}
			(1.183)	(0.368)			(1.199)	(0.343)
Latitude			-0.784	-0.386			-4.742^{***}	-0.525
			(0.529)	(0.316)			(0.591)	(0.273)
Longitude × Latitude			-1.359	-1.066^{**}			-6.893^{***}	-0.908^{**}
			(0.986)	(0.344)			(0.890)	(0.317)
Dependence					1.642^{*}		-0.768	
					(0.762)		(1.304)	
BIC	861.562	952.230	540.451	778.600	1442.682		1097.948	
Log Likelihood	-413.552	-458.886	-235.768	-354.842	-716.836		-540.310	
N	312	312	312	312	312		312	

Table 2: Univariate and Bivariate Poisson and Zero-Inflated Poisson Models of Terrorist Attacks, Frank Copula. Standard errors are in parentheses, with $^{***}p < 0.001$; $^{**}p < 0.01$; and $^{*}p < 0.05$.

`bizicount` function should be utilized.

5.3 Diagnostics

To then carry out post-estimation diagnostics on the preceding models, we can use **DHARMa**'s residual analysis by invoking **bizicount**'s `make_DHARMa` wrapper, and then use any of **DHARMa**'s functions on the resulting list of objects, as discussed in **??**. For example, testing the marginal uniformity of the residuals using a KS test can be done as follows:

```
# DHARMa
library("DHARMa")
nsims = 500
dharma_bivpois = make_DHARMa(bivpois, nsims, seed=78941)
dharma_zibivpois  = make_DHARMa(zibivpois, nsims, seed=97941)

bivpois_ks = lapply(dharma_bivpois, testUniformity, plot=F)
zibivpois_ks = lapply(dharma_zibivpois, testUniformity, plot=F)
names(bivpois_ks) = names(zibivpois_ks) = bivpois$outcomes
```

Which then has the corresponding output as follows.

```
DHARMa Output: Bivariate Poisson

> print(bivpois_ks)
$att.ful

        One-sample Kolmogorov-Smirnov test

data:  simulationOutput$scaledResiduals
D = 0.10385, p-value = 0.002389
alternative hypothesis: two-sided



$att.bok

        One-sample Kolmogorov-Smirnov test

data:  simulationOutput$scaledResiduals
D = 0.090717, p-value = 0.01177
alternative hypothesis: two-sided
```

```
DHARMa Output: Bivariate Zero Inflated Poisson

> print(zibivpois_ks)
$att.ful

        One-sample Kolmogorov-Smirnov test

data:  simulationOutput$scaledResiduals
D = 0.039506, p-value = 0.7149
alternative hypothesis: two-sided


$att.bok

        One-sample Kolmogorov-Smirnov test

data:  simulationOutput$scaledResiduals
D = 0.040695, p-value = 0.6797
alternative hypothesis: two-sided
```

Inspecting the output from the KS tests, we see that failing to account for zero-inflation yields marginal models that do not fit a Poisson distribution very well. However, after accounting for zero-inflation, we fail to reject the KS test's null hypothesis, suggesting a better fit when accounting for excess zeros. Note also that **DHARMa** has a host of additional functions, and default plotting techniques for its objects. For clarity, we only include the above KS tests for marginal uniformity, but some additional tests are shown in the Appendix for interested readers. Alternatively, readers are referred to the detailed documentation on **DHARMa**, as any of its functions is compatible with the **DHARMa** objects that we created above.

6 Conclusion and Future Directions

As we have demonstrated here, with multivariate count data researchers should often consider both dependence across outcomes and zero inflation (i.e., an increased probability mass on zero). While many solutions exist for each of these problems separately, when faced with both simultaneously researchers have far fewer options. Currently, researchers

interested in estimating zero-inflated bivariate count models are often required to specify a particular multivariate zero-inflated count distribution, even when the assumptions for such a specification are not well supported. Here we have discussed how copulas can be used to provide an alternative that only requires assumptions about the *marginal* distributions and the form of the dependence, assumptions which are often more tenable. Using this strategy, we express an arbitrary joint distribution as a known copula distribution with uniform marginals and an additional parameter to capture the dependence between outcomes. Because researchers here are only required to supply the marginal distributions, it is straightforward to allow for zero inflation: one simply specifies zero-inflated count margins for the joint distribution.

To make copula-based bivariate zero-inflated count regression models more accessible, we present the **bizicount** package using R. Our package supports both Frank and Gaussian copula regression for either Poisson or negative binomial marginal distributions, and importantly, their zero-inflated counterparts. Moreover, to make the transition from modeling to professional writing less costly for users, we extend functions from the **texreg** package to output **bizicount** models alongside other models as one table. To facilitate additional post-estimation diagnostic tests, we include functions compatible with the well-known **DHARMa** package, thus permitting marginal residual analysis, dispersion testing, as well as a host of other tools. In this way, we offer authors an alternative means of accounting for dependence between their zero-inflated counts, as well as additional tools to aid in the analysis and presentation of model results.

In future work we plan to consider adding functionality to the **bizicount** package including: the "distributional transform" (Inouye et al., 2017; Kazianka, 2013; Kazianka and Pilz, 2010; Rüschendorf, 2013), the addition of non-copula based approaches, and an extension of the package to higher dimensional settings (e.g., trivariate models). For the extension to multivariate settings – i.e., with dimension $d > 2$ – we would likely use vine pair copula constructions (vine PCCs), due both to their computational superiority and accuracy relative to other approaches (Panagiotelis, Czado and Joe, 2012). As needed, we will also consider adding support for additional marginal count distributions, including censored and truncated Poisson, negative binomial distributions, generalized Poisson distribution, etc. For now, however, we feel that our base **bizicount** package is sufficiently general to assist in estimating the models most often encountered by applied researchers.

Acknowledgments

The authors acknowledge support by NSF DMS-1925119 and DMS-2123247. Mikyoung Jun also acknowledges support by NIH P42ES027704. The authors thank the Texas

A&M University High Capacity Research Computing office for computing support for the simulations found in the appendix.

References

Akaike, Hirotogu. 1973. Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*. Springer pp. 199–213.

Arab, Ali, Scott H Holan, Christopher K Wikle and Mark L Wildhaber. 2012. "Semiparametric bivariate zero-inflated Poisson models with application to studies of abundance for multiple species." *Environmetrics* 23(2):183–196.

Blyth, Simon, Brian Groombridge, Igor Lysenko, Lera Miles and Adrian Newton. 2002. "Mountain watch." *Cambridge, UK: UNEP World Conservation Monitoring Centre* .

Cameron, A Colin and Pravin K Trivedi. 2013. *Regression analysis of count data*. Vol. 53 Cambridge university press.

Cameron, A Colin, Tong Li, Pravin K Trivedi and David M Zimmer. 2004. "Modelling the differences in counted outcomes using bivariate copula models with application to mismeasured counts." *The Econometrics Journal* 7(2):566–584.

Center for International Earth Science Information Network (CIESIN) & Centro Internacional de Agricultura Tropical (CIAT). 2005. "Gridded Population of the World, Version 3 (GPWv3): Population Count Grid.".

Chen, Yuhui and Timothy Hanson. 2017. "Copula regression models for discrete and mixed bivariate responses." *Journal of Statistical Theory and Practice* 11(4):515–530.

Chou, Nan-Ting and D Steenhard. 2011. "Bivariate count data regression models-a SAS® macro program." *SAS Global Forum: Statistics and Data Analysis* 355.

Chowdhury, R.I. and M.A. Islam. 2020. "bpglm: R package for Bivariate Poisson GLM withCovariates." Unpublished.

Conrad, Justin and Kevin Greene. 2015. "Competition, differentiation, and the severity of terrorist attacks." *The Journal of Politics* 77(2):546–561.

Dunn, Peter K and Gordon K Smyth. 1996. "Randomized quantile residuals." *Journal of Computational and Graphical Statistics* 5(3):236–244.

Famoye, Felix. 2010*a*. "A new bivariate generalized Poisson distribution." *Statistica Neerlandica* 64(1):112–124.

Famoye, Felix. 2010*b*. "On the bivariate negative binomial regression model." *Journal of Applied Statistics* 37(6):969–981.

Faroughi, Pouya and Noriszura Ismail. 2017*a*. "Bivariate zero-inflated generalized Poisson regression model with flexible covariance." *Communications in Statistics-Theory and Methods* 46(15):7769–7785.

Faroughi, Pouya and Noriszura Ismail. 2017*b*. "Bivariate zero-inflated negative binomial regression model with applications." *Journal of Statistical Computation and Simulation* 87(3):457–477.

Fréchet, Maurice. 1951. "Sur les tableaux de corrélation dont les marges sont données." *Ann. Univ. Lyon, 3ˆ e serie, Sciences, Sect. A* 14:53–77.

Genest, Christian and Johanna Nešlehová. 2007. "A primer on copulas for count data." *ASTIN Bulletin: The Journal of the IAA* 37(2):475–515.

Goren, Emily and John Hughes. 2021. ***copCAR**: Fitting the copCAR regression model for discrete areal data*. Frederick, MD: . R package version 2.0-4.

Gurmu, Shiferaw and John Elder. 2008. "A bivariate zero-inflated count data regression model with unrestricted correlation." *Economics Letters* 100(2):245–248.

Hartig, Florian. 2020. *DHARMa: Residual Diagnostics for Hierarchical (Multi-Level / Mixed) Regression Models*. R package version 0.3.3.0.
URL: *https://CRAN.R-project.org/package=DHARMa*

Hoeffding, Wassily. 1940. Masstabinvariante Korrelationstheorie. In *The Collected Works of Wassily Hoeffding*, ed. P.K. Sen N.I. Fisher. Springer pp. 57–107.

Hoeffding, Wassily. 1941. Masstabinvariante korrelationsmasse für diskontinuierliche Verteilungen. In *The Collected Works of Wassily Hoeffding*, ed. P.K. Sen N.I. Fisher. Springer pp. 49–70.

Holgate, Philip. 1964. "Estimation for the bivariate Poisson distribution." *Biometrika* 51(1-2):241–287.

Horowitz, Michael C and Philip BK Potter. 2014. "Allying to kill: Terrorist intergroup cooperation and the consequences for lethality." *Journal of Conflict Resolution* 58(2):199–225.

Hughes, John. 2015. "copCAR: A flexible regression model for areal data." *Journal of Computational and Graphical Statistics* 24(3):733–755.

Inouye, David I, Eunho Yang, Genevera I Allen and Pradeep Ravikumar. 2017. "A review of multivariate distributions for count data derived from the Poisson distribution." *Wiley Interdisciplinary Reviews: Computational Statistics* 9(3).

Jackman, Simon, with contributions from Alex Tahk, Achim Zeileis, Christina Maimone, Jim Fearon and Zoe Meers. 2020. *pscl: Political Science Computational Laboratory*. R package version 1.5.5.
URL: *https://CRAN.R-project.org/package=pscl*

Joe, Harry. 1997. *Multivariate models and multivariate dependence concepts*. CRC Press.

Karlis, Dimitris. 2016. Models for Multivariate Count Time Series. In *Handbook of Discrete-Valued Time Series*, ed. Richard A Davis, Scott H Holan, Robert Lund and Nalini Ravishanker. CRC Press chapter 19, pp. 407–424.

Karlis, Dimitris and Ioannis Ntzoufras. 2005. "Bivariate Poisson and diagonal inflated bivariate Poisson regression models in R." *Journal of Statistical Software* 14(10):1–36.

Kazianka, Hannes. 2013. "Approximate copula-based estimation and prediction of discrete spatial data." *Stochastic Environmental Research and Risk Assessment* 27(8):2015–2026.

Kazianka, Hannes and Jürgen Pilz. 2010. "Copula-based geostatistical modeling of continuous and discrete data including covariates." *Stochastic environmental research and risk assessment* 24(5):661–673.

Kocherlakota, Subrahmaniam and Kathleen Kocherlakota. 1992. *Bivariate Discrete Distributions*. Routledge.

Kraemer, Nicole. 2014. ***copulaRegression**: Bivariate copula-based regression models.* **URL:** *https://rdrr.io/cran/CopulaRegression/man/CopulaRegression-package.html*

Kydd, Andrew H and Barbara F Walter. 2006. "The strategies of terrorism." *International security* 31(1):49–80.

LaFree, Gary and Laura Dugan. 2007. "Introducing the global terrorism database." *Terrorism and political violence* 19(2):181–204.

Lai, CD. 1995. "Construction of bivariate distributions by a generalised trivariate reduction technique." *Statistics & probability letters* 25(3):265–270.

Lakshminarayana, J, SNN Pandit and K Srinivasa Rao. 1999. "On a bivariate Poisson distribution." *Communications in Statistics-Theory and Methods* 28(2):267–276.

Lambert, Diane. 1992. "Zero-inflated Poisson regression, with an application to defects in manufacturing." *Technometrics* 34(1):1–14.

Leifeld, Philip. 2013. "**texreg**: Conversion of Statistical Model Output in R to LaTeX and HTML Tables." *Journal of Statistical Software* 55(8):1–24.
URL: *http://dx.doi.org/10.18637/jss.v055.i08*

Li, Chin-Shang, Jye-Chyi Lu, Jinho Park, Kyungmoo Kim, Paul A Brinkley and John P Peterson. 1999. "Multivariate zero-inflated Poisson models and their applications." *Technometrics* 41(1):29–38.

Marra, Giampiero and Rosalba Radice. 2017. "A joint regression modeling framework for analyzing bivariate binary data in R." *Dependence Modeling* 5(1):268–294.

Marshall, Albert W and Ingram Olkin. 1985. "A family of bivariate distributions generated by the bivariate Bernoulli distribution." *Journal of the American Statistical Association* 80(390):332–338.

Marshall, Albert W and Ingram Olkin. 1990. "Multivariate distributions generated from mixtures of convolution and product families." *Lecture Notes-Monograph Series* pp. 371–393.

Masarotto, Guido and Cristiano Varin. 2017. "Gaussian copula regression in R." *Journal of Statistical Software* 77(8):1–26.

Mullahy, John. 1986. "Specification and testing of some modified count data models." *Journal of econometrics* 33(3):341–365.

Nelsen, Roger B. 2007. *An Introduction to Copulas*. Springer Science & Business Media.

Nikoloulopoulos, Astridis. 2013. Copula-Based Models for Multivariate Discrete Response Data. In *Copulae in Mathematical and Quantitative Finance*, ed. Piotr Jaworski, Fabrizio Durante and Wolfgang Karl Härdle. Springer chapter 11, pp. 231–250.

Panagiotelis, Anastasios, Claudia Czado and Harry Joe. 2012. "Pair copula constructions for multivariate discrete data." *Journal of the American Statistical Association* 107(499):1063–1072.

Rüschendorf, Ludger. 2013. Copulas, Sklar's theorem, and distributional transform. In *Mathematical risk analysis*. Springer pp. 3–34.

Sarmanov, Oleg Vasil'evich. 1966. Generalized normal correlation and two-dimensional Fréchet classes. In *Doklady Akademii Nauk*. Vol. 168 Russian Academy of Sciences pp. 32–35.

SAS Institute, Inc. 2020. "Fitting Zero-Inflated Count Data Models byUsing PROC GENMOD.".
URL: *https://support.sas.com/rnd/app/stat/examples/GENMODZIP/roots.pdf*

Schwarz, Gideon. 1978. "Estimating the dimension of a model." *Annals of statistics* 6(2):461–464.

Seabold, Skipper and Josef Perktold. 2010. Statsmodels: Econometric and statistical modeling with python. In *Proceedings of the 9th Python in Science Conference*. Vol. 57 Austin, TX p. 61.

Sklar, Abe. 1973. "Random variables, joint distribution functions, and copulas." *Kybernetika* 9(6):449–460.

Sklar, M. 1959. "Fonctions de repartition an dimensions et leurs marges." *Publ. inst. statist. univ. Paris* 8:229–231.

So, Sunha, Dong-Hee Lee and Byoung Cheol Jung. 2011. "An alternative bivariate zero-inflated negative binomial regression model using a copula." *Economics Letters* 113(2):183–185.

START. 2018. "Global Terrorism Database [Data file]." Retrieved from https://www.start.umd.edu/gtd.

StataCorp, LLC. 2019. "Stata statistical software: release 16.".

Sun, Tao and Ying Ding. 2019. ***CopulaCenR**: Copula-Based Regression Models for Bivariate Censored Data*. R package version 1.1.2.
URL: *https://CRAN.R-project.org/package=CopulaCenR*

Ting Lee, Mei-Ling. 1996. "Properties and applications of the Sarmanov family of bivariate distributions." *Communications in Statistics-Theory and Methods* 25(6):1207–1222.

Tollefsen, Andreas Forø, Håvard Strand and Halvard Buhaug. 2012. "PRIO-GRID: A unified spatial data structure." *Journal of Peace Research* 49(2):363–374.

Tollefsen, Andreas Forø, Karim Bahgat, Jonas Nordkvelle and Halvard Buhaug. 2015. "PRIO-GRID v. 2.0 codebook.".

Trivedi, Pravin and David Zimmer. 2017. "A note on identification of bivariate copulas for discrete count data." *Econometrics* 5(1):10.

Trivedi, Pravin K and David M Zimmer. 2007. *Copula modeling: an introduction for practitioners*. Now Publishers Inc.

van der Wurp, Hendrik, Andreas Groll, Thomas Kneib, Giampiero Marra and Rosalba Radice. 2019. "Generalised Joint Regression for Count Data with a Focus on Modelling Football Matches." *arXiv preprint arXiv:1908.00823* .

Wang, Kui, Andy H Lee, Kelvin KW Yau and Philip JW Carrivick. 2003. "A bivariate zero-inflated Poisson regression model to analyze occupational injuries." *Accident Analysis & Prevention* 35(4):625–629.

Wang, Peiming. 2003. "A bivariate zero-inflated negative binomial regression model for count data with excess zeros." *Economics Letters* 78(3):373–378.

Xu, Xinling and James W Hardin. 2016. "Regression models for bivariate count outcomes." *The Stata Journal* 16(2):301–315.

Yan, Jun. 2007. "Enjoy the Joy of Copulas: With a Package copula." *Journal of Statistical Software* 21(4):1–21.
URL: *https://www.jstatsoft.org/v21/i04/*

Yang, Lu, Edward W Frees and Zhengjun Zhang. 2020. "Nonparametric estimation of copula regression models with discrete outcomes." *Journal of the American Statistical Association* 115(530):707–720.

Zeileis, Achim and Yves Croissant. 2010. "Extended model formulas in R: Multiple parts and multiple responses." *Journal of Statistical Software* 34(1).

A Visual Depictions of Copulas

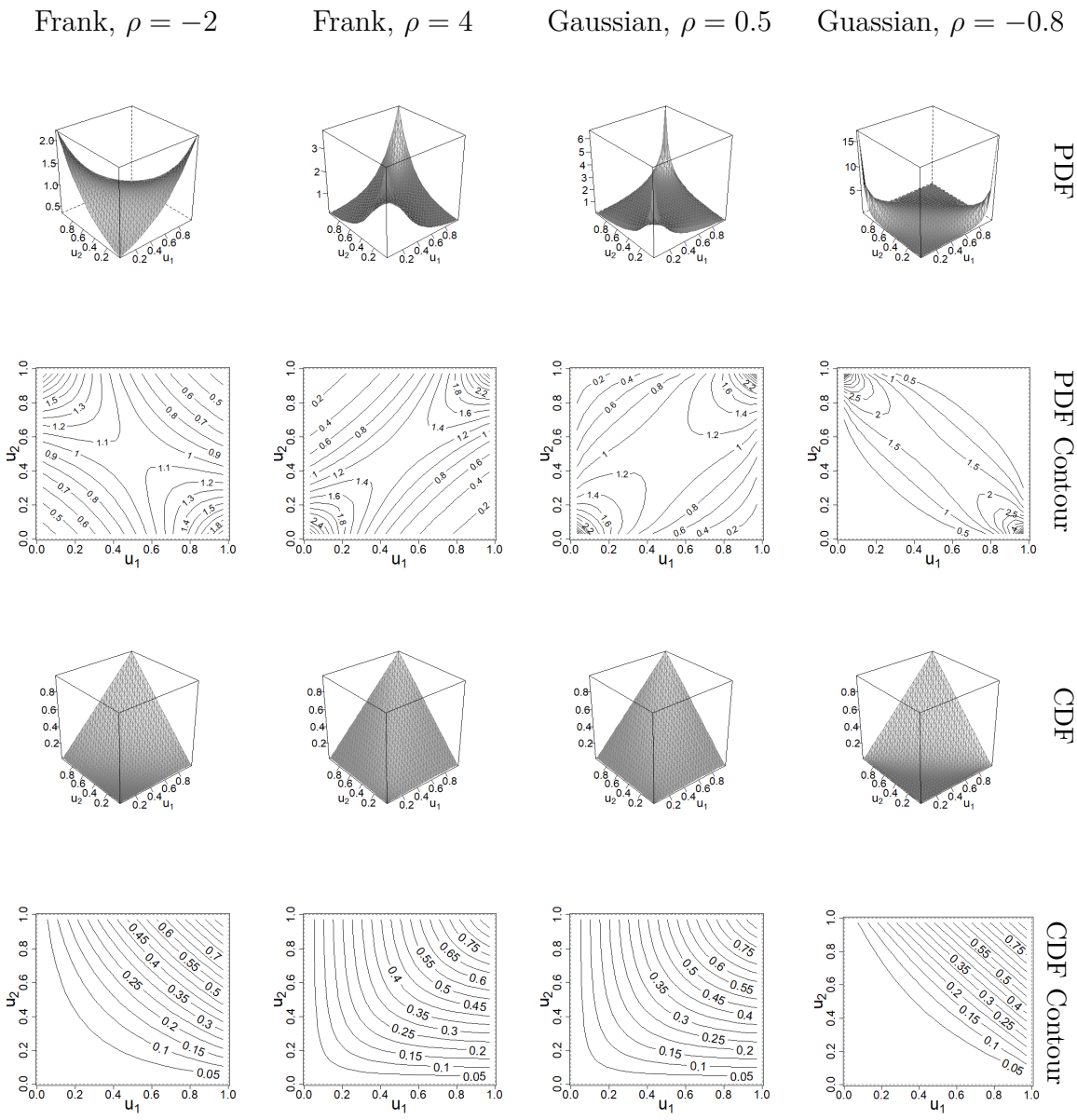

Figure A.1: Bivariate Frank and Gaussian Copula PDFs and CDFs

B Monte Carlo Evidence

Below we provide some monte carlo evidence for claims made in the introductory portion of this article. The data from these simulations are available with our replication materials.

The data-generating process for the below simulations is as follows:

$$\boldsymbol{Y}_j \sim ZIP\left\{\boldsymbol{\lambda}_j = exp(\boldsymbol{X}_j\boldsymbol{\beta}_j), \psi_j = \psi_j\right\}$$

$$\text{cor}\left[\Phi^{-1}\{F_1(Y_1)\}, \Phi^{-1}\{F_2(Y_2)\}\right] = \text{cor}(U_1, U_2) = \rho$$

Where

- $j \in \{1, 2\}$ are the margin indices
- $N = 500$ is the sample size, which is constant for our simulations
- $\boldsymbol{\beta}_1 = [1, 3.25, -2.3]^\top$ is a vector of parameters for the count portion of margin 1, including an intercept. This is not varied.
- $\boldsymbol{\beta}_2 = [2, -1.75, 3.5]^\top$ is a vector of parameters for the count portion of margin 2, including an intercept. This is not varied.
- F_j is the CDF of the respective marginal ZIP distribution
- Φ^{-1} is the univariate standard-normal quantile function
- $\rho \in \{0.15, 0.85\}$ is the dependence parameter to be varied in the Gaussian copula.
- $\psi_1, \psi_2 \in \{0.1, 0.6\}$ are the zero-inflation parameters to be varied in the Gaussian copula for each margin
- $\boldsymbol{X}_1^{N\times 3} \sim Bernoulli(p = 0.5)$ is the matrix of covariates for margin 1, with an intercept included
- $\boldsymbol{X}_2^{N\times 3} \sim Exponential(rate = 3)$ is the matrix of covariates for margin 2, with an intercept

Given these data, we then estimate three models:

1. Our preferred bivariate model with zero-inflated Poisson margins using a Gaussian copula.
2. A bivariate model with Poisson margins using a Gaussian copula; that is, a bivariate model omitting zero-inflation in the margins.
3. A univariate zero-inflated Poisson model for each outcome, $\boldsymbol{Y}_j$. That is, a univariate model for each outcome vector that omits dependence, but which correctly specifies the marginal distribution.

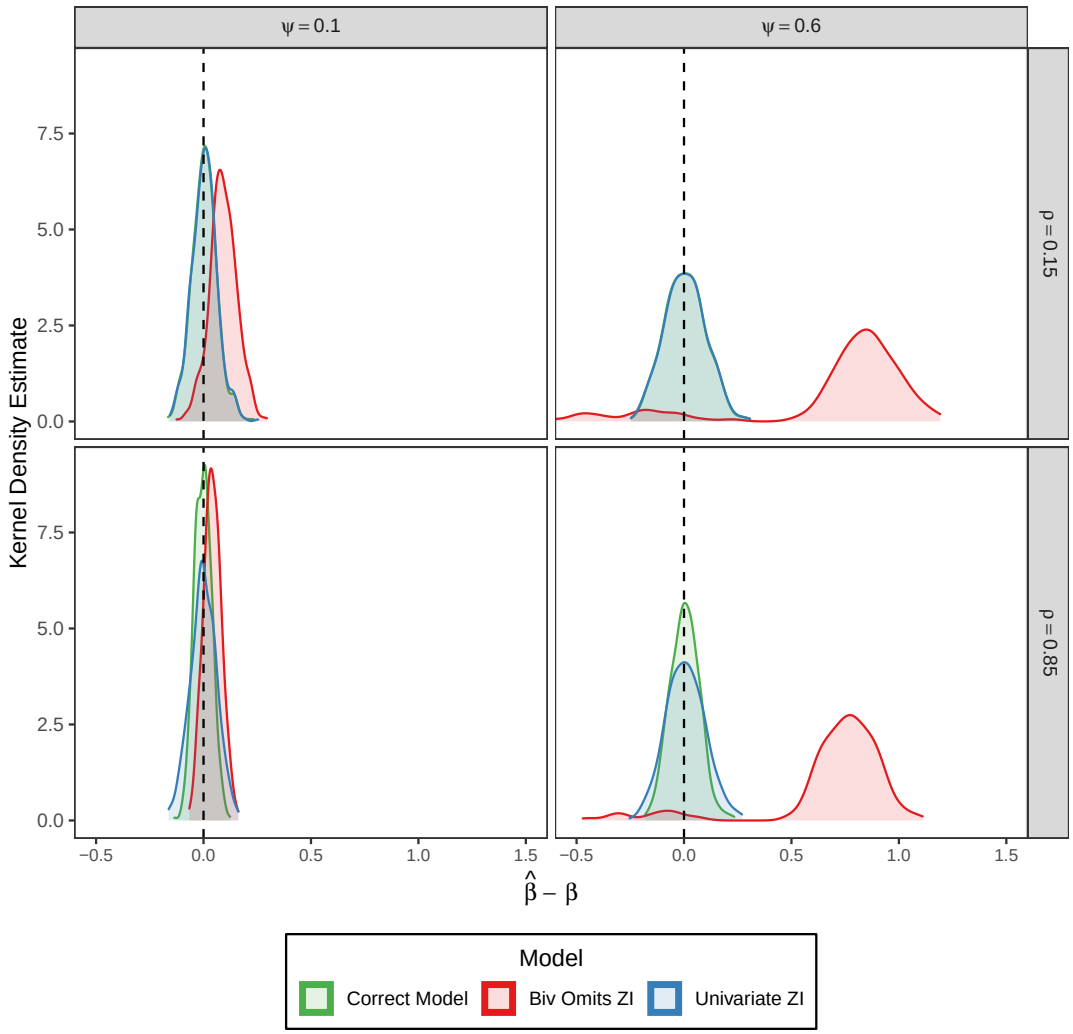

Figure B.1: Density of Centered Count Slope Parameter Estimates ($\beta_{21} = 3.25$) in Margin 1, with $\psi_2 = .1$

The above densities are for the second count slope parameter in margin 1, $\beta_{21} = 3.25$, while holding margin 2's zero-inflation parameter constant at $\psi_2 = .1$. The densities of the estimates are centered on the true value, i.e., they are $\hat{\beta}_{21} - \beta_{21}$. A few conclusions are apparent:

- The correct bivariate model is unbiased.
- The incorrect bivariate model is biased upward, with the degree of this bias increasing in the degree of zero-inflation. This is to be expected, as Poisson margins have a mean of λ, while ZIP margins have a mean of $(1 - \psi)\lambda \leq \lambda$.
- The univariate ZIP model that omits dependence has greater variance than the correct bivariate model. This difference in variance increases with the degree of dependence in the data generating process, which is to be expected as there is less information upon which to base the univariate inferences when compared to their bivariate counterpart.

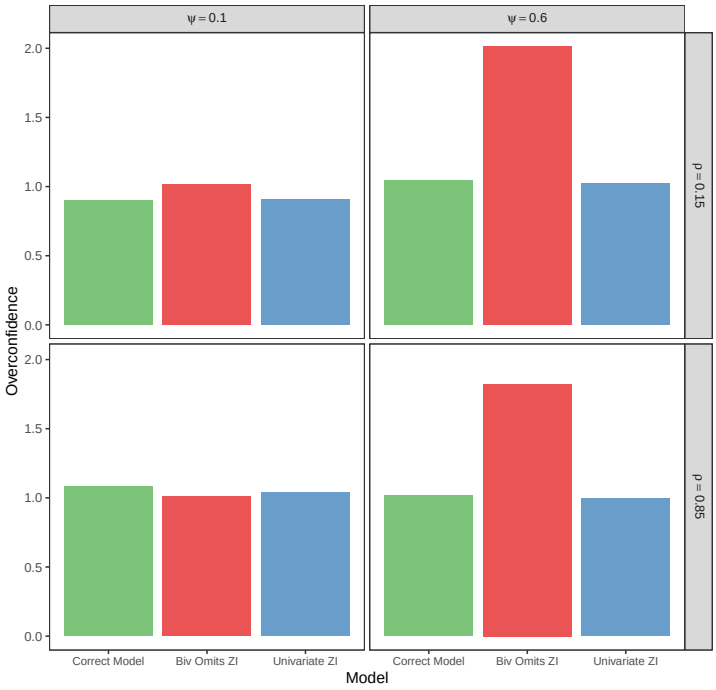

Figure B.2: Overconfidence of Centered Count Slope Parameter Estimate ($\beta_{21} = 3.25$) in Margin 1, with $\psi_2 = .1$

The above plot demonstrates the level of overconfidence in the estimated parameter β_{21} compared to its empirical sampling distribution, again while holding margin 2's zero-inflation fixed at $\psi_2 = .1$. We measure overconfidence as

$$\frac{MAD(\hat{\beta})}{med[\widehat{SE}(\hat{\beta})]}$$

In words, we take an estimate of the dispersion of $\hat{\beta}$ relative to a measure of the center of its estimated standard errors. If the standard errors are underestimated on average relative to the dispersion of the empirical sampling distribution of the estimator, then this measure exceeds 1. If the opposite is true, we are underconfident, and the measure is less than 1. Thus, an ideal estimator will have a value close to 1 for overconfidence. An alternative is to replace the numerator with the standard deviation rather than the median absolute deviation, and the denominator with the mean of the estimated standard errors. However, due to some outliers from simulation leading to long tails—particularly for the incorrectly specified bivariate model—this results in an unreasonable level of overconfidence. Thus, we use robust measures of the center and spread, which only affects the incorrect bivariate model specifically by *reducing* its overconfidence. In other words, we make it easier for the misspecified model to compete.

The primary conclusion from this plot is that the incorrect bivariate model tends to underestimate its standard errors, and that the severity of this underestimation increases with the degree of zero-inflation. Surprisingly, the univariate ZIP model does not underestimate its standard errors in spite of failing to model dependence.

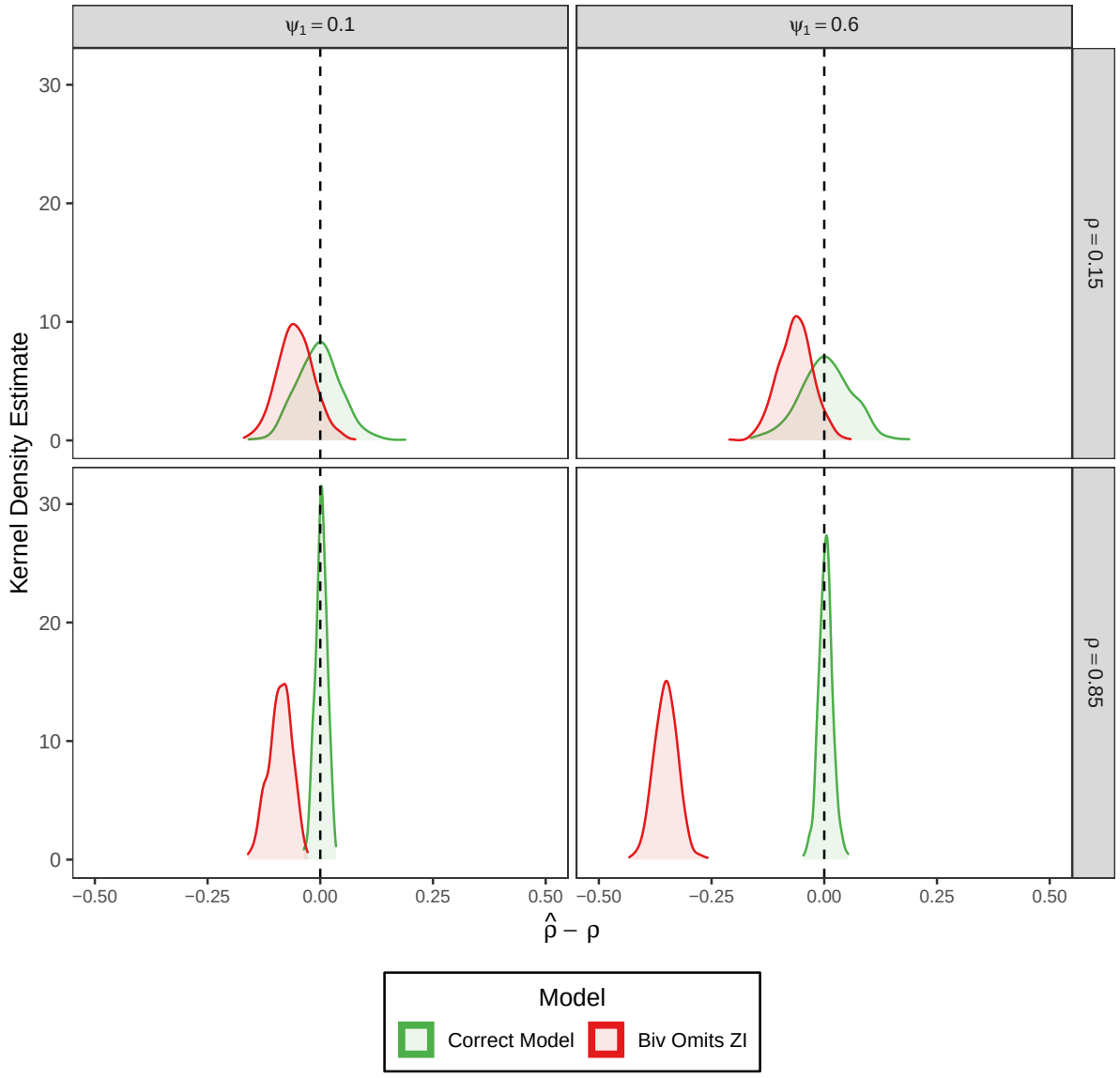

Figure B.3: Density of Centered Dependence Parameter Estimate with $\psi_2 = .1$

The above density plots are for the centered estimates of the dependence parameter, again holding the zero-inflation parameter for margin 2 fixed at $\psi_2 = .1$. There are two primary conclusions from this plot:

1. The incorrect bivariate model underestimates the dependence by about a factor of 1/2.

2. The incorrect bivariate model's variance on the dependence parameter increases with the degree of zero-inflation.

C Computation

In this section we detail the `frech.min` and `pmf.min` parameters to `bizicount`.

As Fréchet (1951) and Hoeffding (1940, 1941) show, any two-dimensional copula distribution function has the following bounds:

$$\max\{u + v - 1,\, 0\} \leq C(u, v) \leq \min\{u, v\}$$

where u and v are realizations from the two uniform marginal distributions of the copula, and $C(\cdot)$ is the copula CDF. Alternatively, this can be written as:

$$\max\{F_1(y_1) + F_2(y_2)1, 0\} \leq F(y_1, y_2) \leq \min\{F_1(y_1), F_2(y_2)\}$$

using the notation from the main text. Accordingly, we impose these bounds on each of the four CDF evaluations used in the finite difference approximation to the PMF. However, to avoid numerical issues with floating-point representations (over/underflow), in practice we use

$$\max\{u + v - 1,\, \texttt{frechmin}\} \leq C(u, v) \leq \min\{u, v, 1 - \texttt{frechmin}\}$$

where we require that `frech.min` $\in (0, .00001]$, with a default value of 1e-7. Effectively, this constrains all copula CDF evaluations to the unit interval.

Second, extremely small values for the PMF—and therefore the likelihood of a each observation individually—can be rounded off to zero due to numerical underflow. These zeros will then result in infinite-valued logarithms in the log-likelihood, which when summed with other, finite log-likelihood values, still returns an infinite log-likelihood for all observations. Thus, to avoid this issue, we threshold PMF values at `pmf.min` to ensure stability in evaluating the log-likelihood during numerical optimization. By default, this is also 1e-7.

D Further Diagnostics in DHARMa

For clarity, we did not print DHARMa's diagnostic plots in the main text, so they are included here. However, readers should also read the documentation for that package, as there are several other functions that may be of interest.

```
## Appendix Dharma stuff

name = c("fulani", "bh")

for(i in seq_along(name)){

  pdf(paste0("replication/appendix_dharma_bivpois_", name[i], ".pdf"), width=8, heig
    plot(dharma_bivpois[[i]])
  dev.off()

  pdf(paste0("replication/appendix_dharma_ZIbivpois_", name[i], ".pdf"), width=8, he
    plot(dharma_zibivpois[[i]])
  dev.off()
}
```

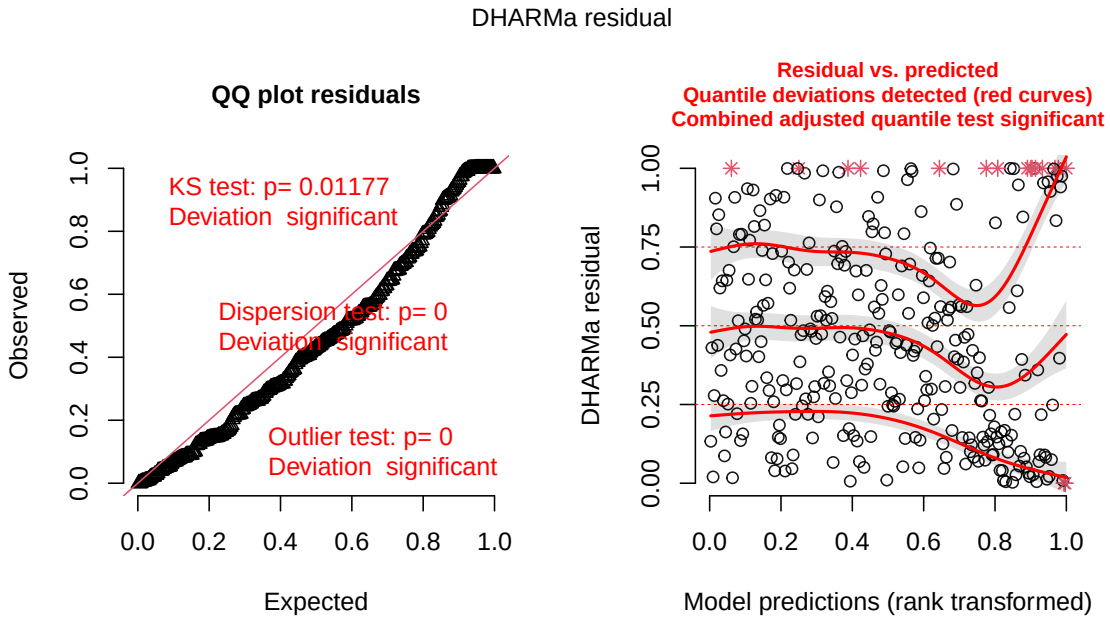

Figure D.1: Bivariate Poisson, Boko Haram

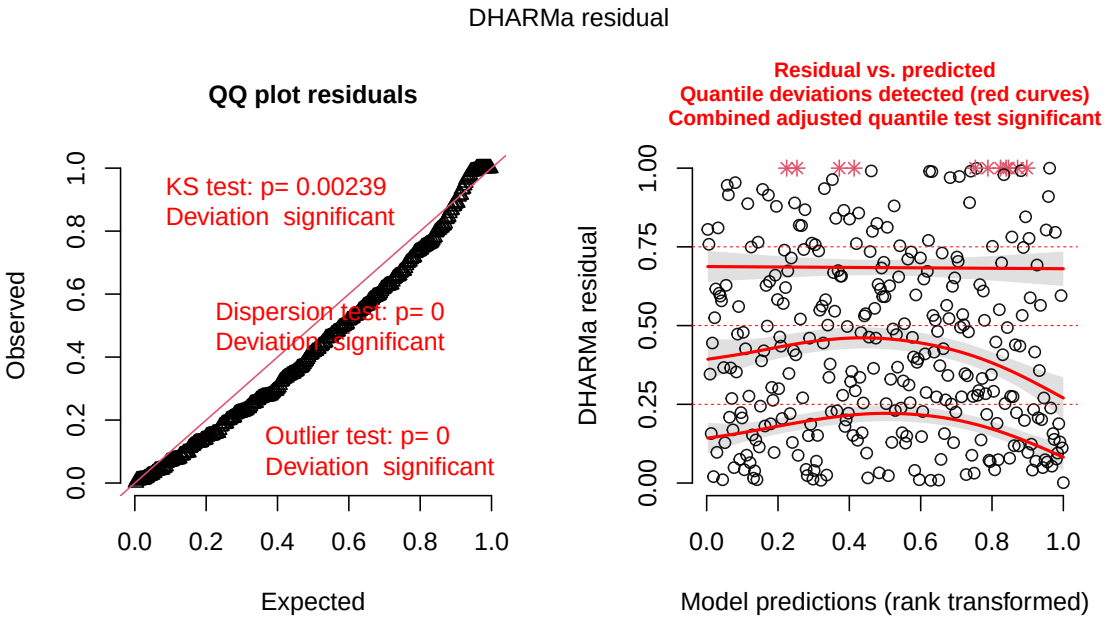

Figure D.2: Bivariate Poisson, Fulani

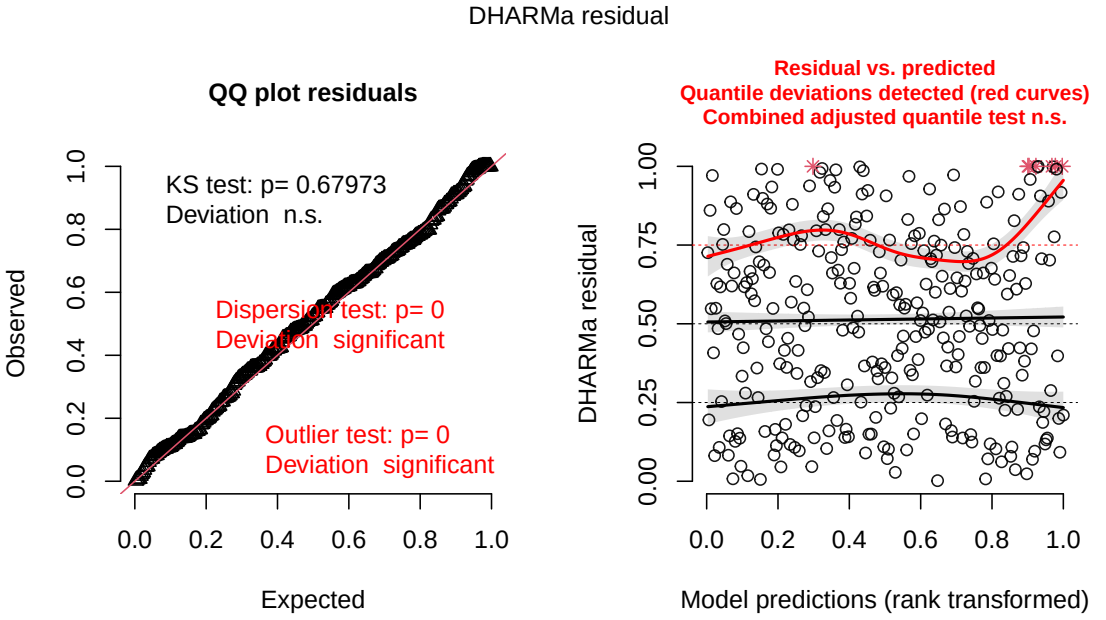

Figure D.3: Zero-Inflated Bivariate, Boko Haram

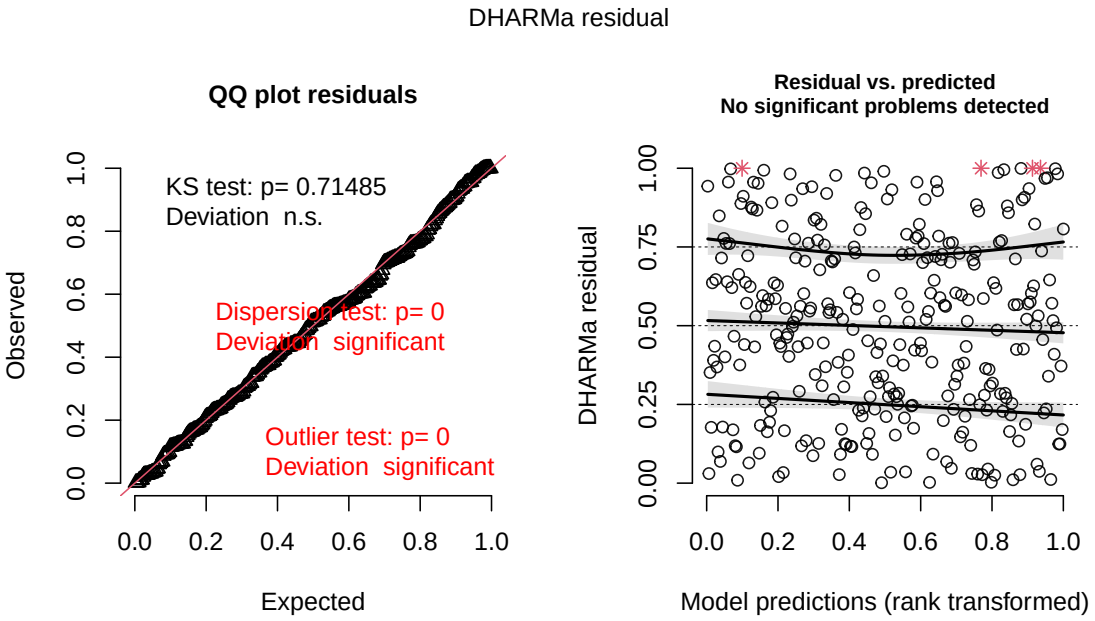

Figure D.4: Zero-Inflated Bivariate, Fulani

E Univariate Functions

E.1 Random Number Generation: rzip, rzinb

To generate samples from the ZIP and ZINB distributions, we define two functions:

```
rzip(n, lambda, psi, recycle=F)
rzinb(n, size, psi, prob=NULL, mu=NULL, recycle=F)
```

Where:

- `n` is the sample size
- `lambda` or `mu` are the conditional mean of the *count* distribution. In other words, they are *not* the conditional mean of the ZIP/ZINB distributions, but the mean of the constituent count distribution. Mathematically, they are the λ and μ found in $(1-\psi)\lambda$ for the mean of the ZIP, and $(1-\psi)\mu$ for the ZINB.
- `psi` is the conditional probability of zero-inflation
- `prob` is an alternative way to specify the mean to the negative binomial distribution; see `?nbinom` for details. For regression purposes, it is often desireable to use `mu` rather than `prob`. One of `mu` or `prob` must be specified.
- `size` is the inverse dispersion parameter of the negative binomial distribution. This is often denoted θ in the literature, and is inverted to obtain a dispersion parameter estimate. See `?nbinom` for more details.
- `recycle` allows recycling of unequal length vectors that are input as arguments for the other parameters in the function. Eg, if `mu` and `psi` are of different lengths, `recycle` allows these vectors to be recycled to ensure comformity. Note that recycling will occur by default *even if recycle is set to FALSE* when one argument is length-one. Eg, if `psi` is length-one, and `lambda` is length-n, `psi` will be recycled.

E.2 CDF Evaluation: ZIP, ZINB

To evaluate the univariate ZIP and ZINB CDFs, we define the following:

```
pzip(q, lambda, psi, lower.tail=T, log.p=F, recycle=F)
pzinb(q, size, psi, prob=NULL, mu=NULL, lower.tail=T, log.p=F, recycle=F)
```

Where parameters with the same name as previous functions are the same, and:

- `q` is the vector of quantiles for evaluation
- `lower.tail` indicates whether the lower-tail, or alternatively its complement should be returned.
- `log.p` indicates whether the preobabilities ought to be returned on the (natural) log scale

E.3 PMF Evaluation: ZIP, ZINB

To evaluate the univariate ZIP and ZINB PMFs, we define the following:

```
dzip(x, lambda, psi, log=F, recycle=F)
dzinb(x, size, psi,  prob=NULL, mu=NULL, log=F, recycle=F)
```

Where parameters with the same name as previous functions are the same, and:

- `x` is the non-negative integer-valued location at which the mass function is to be evaluated

E.4 Quantile Function Evaluation: ZIP, ZINB

To evaluate the univariate ZIP and ZINB quantile functions, we define the following:

```
qzip(p, lambda, psi, log=F, recycle=F)
qzinb(p, size, psi,  prob=NULL, mu=NULL, log=F, recycle=F)
```

Where parameters with the same name as previous functions are the same, and:

- `p` is a vector of probabilities $\in [0, 1]$ at which to evaluate the quantile function.

E.5 Univariate Regression Function: zic.reg

We also include a univariate, zero-inflated count regression function, `zic.reg`, which is not only used to obtain marginal starting values, but also is compatible with **DHARMa** and **texreg**, similarly to `bizicount`. The parameters to the function are:

```
zic.reg(fmla = NULL, data, subset, na.action,
                    weights = rep(1, length(y)), X = NULL,
                    z = NULL, y = NULL, offset.ct = NULL,
                    offset.zi = NULL, dist = "pois", link.ct = "log",
                    link.zi = "logit", starts = NULL, optimizer = "nlm",
                    warn.parent = T, keep = F, ...)
```

Where:

- `fmla` is a two-part formula, eg `y x|z`. If using the `data` function below, then offsets should be put directly into this formula.
- `data` is a dataframe containing the needed variables for the regression
- `subset`, `na.action`, codeweights determine the subset of the data to use, how to handle missingness, and what weights to attach to the observations
- `X`, `z`, `y`, `offset.ct`, `offset.zi` are, respectively, the design matrix for the count portion, the design matrix for the zero-inflation portion, the outcome vector, the offset vector for the count portion, and the offset vector of the zero-inflation portion. NOTE: these should only be used when `data=NULL`, and are primarily advantageous when using large datasets as they bypass some preliminary data manipulation prior to optimization. Also, no missingness can be present in any of these quantities.

- `dist` is one of either `"pois"` or `"nbinom"`, determining the count distribution
- `link.ct` determines the link function for the count portion, and is one of `c("sqrt", "identity",`
- `link.zi` determines the link function for the zero-inflation portion, and is one of `c("logit", "probit", "cauchit", "log", "cloglog")`
- `starts` is a vector of starting values. If NULL, these are automatically obtained.
- `optimizer` is the optimization package used for the likelihood search, one of either `"nlm"` or `"optim"`. Default: `"nlm"`
- `warn.parent` logical indicating whether to warn about using data from the parent environment when parameter `data` is not supplied.
- `keep` logical indicating whether to keep the model frame used for estimation, or discard it prior to returning. Note: this must be TRUE to obtain fitted values, and for diagnostics in DHARMa.
- `...` further arguments passed to the chosen optimizer. See `?nlm` or `?optim` for more details.

Finally, we provide a quick example of how to use the above function's output for diagnostics in **DHARMa** and with **texreg**. Note that output is suppressed here for clarity, as this is only intended to show how users can do this with their own models. Additionally, this code is assumes that the code from the main text has already been run, as it relies on the formula and data from there. The `simulatedResponse` parameter to `createDHARMa` is obtained by using the `simulate` function that we have defined for `zicreg`-class objects, as seen below.

```
library("DHARMa")
library("texreg")
unizi.bok = zic.reg(fmla = fmla.zi.bok,
                    data = dat,
                    keep=T) # make sure keep=T

uni.dharma = createDHARMa(simulatedResponse= simulate(unizi.bok),
                          observedResponse = dat$att.bok,
                          fittedPredictedResponse = fitted(unizi.bok),
                          integerResponse = T,
                          seed = 123,
                          method = "PIT")
plot(uni.dharma)
texreg(unizi.bok)
```