




bizicount: Bivariate Zero-Inflated Count Copula Regression Using R

John M. Niehaus **Lin Zhu** **Scott J. Cook** **Mikyoung Jun** 
Texas A&M University Renmin University Texas A&M University University of Houston
of China
Texas A&M University

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 to estimate bivariate zero-inflated count regression models in R. Therefore, we present an R package, **bizicount**, that enables 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.

Keywords: zero-inflated, bivariate, copula, Poisson, negative binomial, regression, R.

1. Motivation

Count data often contain a higher proportion of zeroes than expected under probability distributions, such as the Poisson or Negative binomial, commonly used in regression modeling. These “excess” zeroes have led to the development of statistical models that account for zero inflation in count data regression (e.g., [Mullahy 1986](#); [Lambert 1992](#)), which are now widely used by applied researchers. These zero-inflated models jointly consider two processes, with the first (often a Bernoulli process) generating zeroes and the second (often a Poisson or Negative binomial) generating counts. For example, if Y a random variable (with $y \in \mathbb{Z}^{\geq 0}$) that follows a zero-inflated Poisson distribution, one assumes Y has a mixture distribution of

a Bernoulli distribution (with probability of zero, ψ , $0 \leq \psi \leq 1$) and a Poisson distribution (with rate $\lambda > 0$):

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

where $f(y; \lambda) = \frac{\lambda^y e^{-\lambda}}{y!}$. This class of zero-inflated count data models are available in most commonly used statistical software packages, however, these are all for single-count (i.e., univariate) processes.¹

In theory, incorporating a zero-inflated component into bivariate (or multivariate) count processes can also be achieved without much issue. For example, if (Y_1, Y_2) follows a bivariate zero-inflated Poisson distribution, one can assume that (i) with probability ψ_1 , $Y_1 = Y_2 = 0$, (ii) with probability ψ_2 , Y_1 is a Poisson random variable with rate $\lambda_1 > 0$ and $Y_2 = 0$, (iii) with probability ψ_3 , $Y_1 = 0$ and Y_2 is a Poisson random variable with rate $\lambda_2 > 0$, and (iv) with probability $1 - \psi_1 - \psi_2 - \psi_3$, Y_1 and Y_2 follow bivariate Poisson distribution, for $0 \leq \psi_i \leq 1$, $i = 1, 2, 3$ and $\sum_{i=1}^3 \psi_i \leq 1$. In practice, however, specifying the bivariate count mass function often require researchers to make assumptions about the joint distribution that are not well motivated by their theory, and/or impose restrictions (e.g., positive dependence across counts) that are not well supported by their data (see Section 2 for a review of approaches for modeling bivariate count distributions). As a result, there are few available statistical software routines to account for excess zeroes for bivariate (or multivariate) count processes.

Due to this lack of software, researchers analyzing bivariate (or multivariate) count data 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 – e.g., 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 parameters, 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 parameter also produces inaccurate (and overconfident) 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, Holan, Wikle, and Wildhaber 2012; Faroughi and Ismail 2017a,b; Gurmu and Elder 2008; Li, Lu,

¹For R, see the **pscl** package (Zeileis, Kleiber, and Jackman 2008); for Stata, see the **zip** command (StataCorp 2021); for SAS, see SAS Institute Inc. (2020); for Python, see Seabold and Perktold (2010). SPSS relies on a R plug-in that calls up the **pscl** package.

Park, Kim, Brinkley, and Peterson 1999; Wang 2003; Wang, Lee, Yau, and Carrivick 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 does not permit negative dependence for zero-inflation, and also assumes a specific joint distribution.

Quantitative research on terrorism provides a useful example of this problem. Here researchers are often interested in modeling variation in terror attack counts or frequencies (for a given spatial or space-time unit) as a function of known inputs (e.g., population, development, etc.). Given the high proportion of zeroes in data on terrorism (see Desmarais and Harden 2013), researchers often utilize (univariate) zero-inflation count models (e.g., Piazza 2017; Savun and Tirone 2018). However, researchers are also often interested in more than one sample of terror events and, potentially, the relationship between these. In country-level analysis of terror counts, for example, this may include domestic and transnational terrorism (e.g., Wilson and Piazza 2013). In sub-national analyses of terrorism, this may instead involve attacks perpetrated by several terror groups (e.g., Boko Haram and Fulani extremists in Nigeria). Given limitations to existing software, terrorism researchers are currently forced to choose between neglecting zero-inflation (e.g., estimating a bivariate Poisson), neglecting cross-sample correlations (e.g., estimating a zero-inflated Poisson for a single sample of attacks), or both (e.g., estimating a Poisson for a single sample of attacks).

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

$$f_{Y_1, Y_2}(y_1, y_2) = P(Y_1 = y_1, Y_2 = y_2) = e^{(-\lambda_X - \lambda_V - \lambda_U)} \sum_{c=0}^{\min(y_1, y_2)} \frac{\lambda_U^c \lambda_X^{y_1-c} \lambda_V^{y_2-c}}{(y_1-c)!(y_2-c)!c!}, \quad (2)$$

for $y_1, y_2 = 0, 1, \dots, \infty$. 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

$$f_{Y_1, Y_2}(y_1, y_2) = \left[\frac{e^{(-\lambda_1 - \lambda_2)} \lambda_1^{y_1} \lambda_2^{y_2}}{y_1! y_2!} \right] \left[1 + \alpha (e^{-y_1} - e^{-\lambda_1 c}) (e^{-y_2} - e^{-\lambda_2 c}) \right], \quad (3)$$

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 2010b; Marshall and Olkin 1990) or bivariate generalized Poisson distributions (Famoye 2010a). 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 given in Equation 1, and the corresponding cumulative distribution function (CDF) is then

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

where $f_C(\cdot)$ denotes the PMF of some arbitrary count distribution (e.g., Poisson, negative binomial, etc.). Then, combining the intuition of the univariate zero-inflated distribution from Equation 1 with the bivariate Poisson PMF from Equation 3, we have a PMF for the BZIP based on multiplicative factors (Faroughi and Ismail 2017b):

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

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 Equation 3.² This can be generalized to a bivariate

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

regression modelling context in the usual way: for observations i and margins j we specify $\psi_{ij} = \Lambda(Z_{ij}\gamma_j)$ 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 j -th marginal count distribution (f_C in Equation 4), 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. As such, no assumptions regarding the form of the joint distribution are required. Instead, 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. 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 Section 2.1). 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 $Y \sim U(0, 1)$:

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

which implies F_Y is the CDF of the standard uniform distribution. Then, given a random vector $\mathbf{X} = (X_1, X_2, \dots, X_n)$ with continuous, monotone increasing marginal CDFs (F_1, F_2, \dots, F_n), we can use the above result to deduce that

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

which is to say that the random vector $\{F_1(X_1), F_2(X_2), \dots, F_n(X_n)\}$ has uniform marginals. The copula function is then defined as the continuous joint CDF of the variables in the vector \mathbf{U} above (Cameron, Li, Trivedi, and Zimmer 2004; Trivedi and Zimmer 2007, p. 10):

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

where $C(\cdot)$ is the copula function. Thus, working backwards we can see that if $\mathbf{U} \sim C$, then $\{F_1^{-1}(u_1), F_2^{-1}(u_2), \dots, F_n^{-1}(u_n)\} \sim F$, 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 $\text{ran}(F_1) \times \text{ran}(F_2) \times \cdots \times \text{ran}(F_n)$, 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)

$$F(y_1, y_2) = C\{F_1(y_1), F_2(y_2); \rho\} = C(u_1, u_2; \rho), \quad (6)$$

where ρ is the dependence parameter to be estimated. In order to obtain the joint PDF from the joint CDF in Equation 6, 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 *et al.* 2011; Trivedi and Zimmer 2007):

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

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 Equation 7, we can proceed with estimating the parameters of the distribution by maximum likelihood in the usual way.³

The only remaining question is on 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

³Note 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, Yang, Allen, and Ravikumar (2017).

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

copula takes the form

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

where F_j are the marginal, zero-inflated count CDFs as defined in Equation 4, 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 permits unbounded dependence, with the bivariate form

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

where the F_j are the same as in Equation 8, and $\rho \neq 0$ is the dependence parameter. Visual depictions of each copula can be found in Equation 1.

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 Equation 4. 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 j -th marginal *count* distribution (f_C in Equation 4), 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 (8) or (9) to the PMF from finite-differencing as shown in Equation 7, and then estimate the parameter vectors γ_j and β_j by maximum 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 the appropriate probability distribution for each marginal model.
2. Use prior theory to choose a set of relevant covariates for each marginal model (can be common to both or unique).
3. Select a copula distribution relating the two marginals.
4. Take finite differences over the copula CDF to arrive at the PMF as in Equation 7.
5. Take logarithms and use maximum likelihood to estimate parameters.
6. Carry out post-estimation diagnostics to test assumptions, model fit, etc.
7. Evaluate input relevance using typical methods (e.g., likelihood ratio, Wald, score tests) and calculate desired quantities of interest (e.g., log expected counts, marginal effects, etc.).

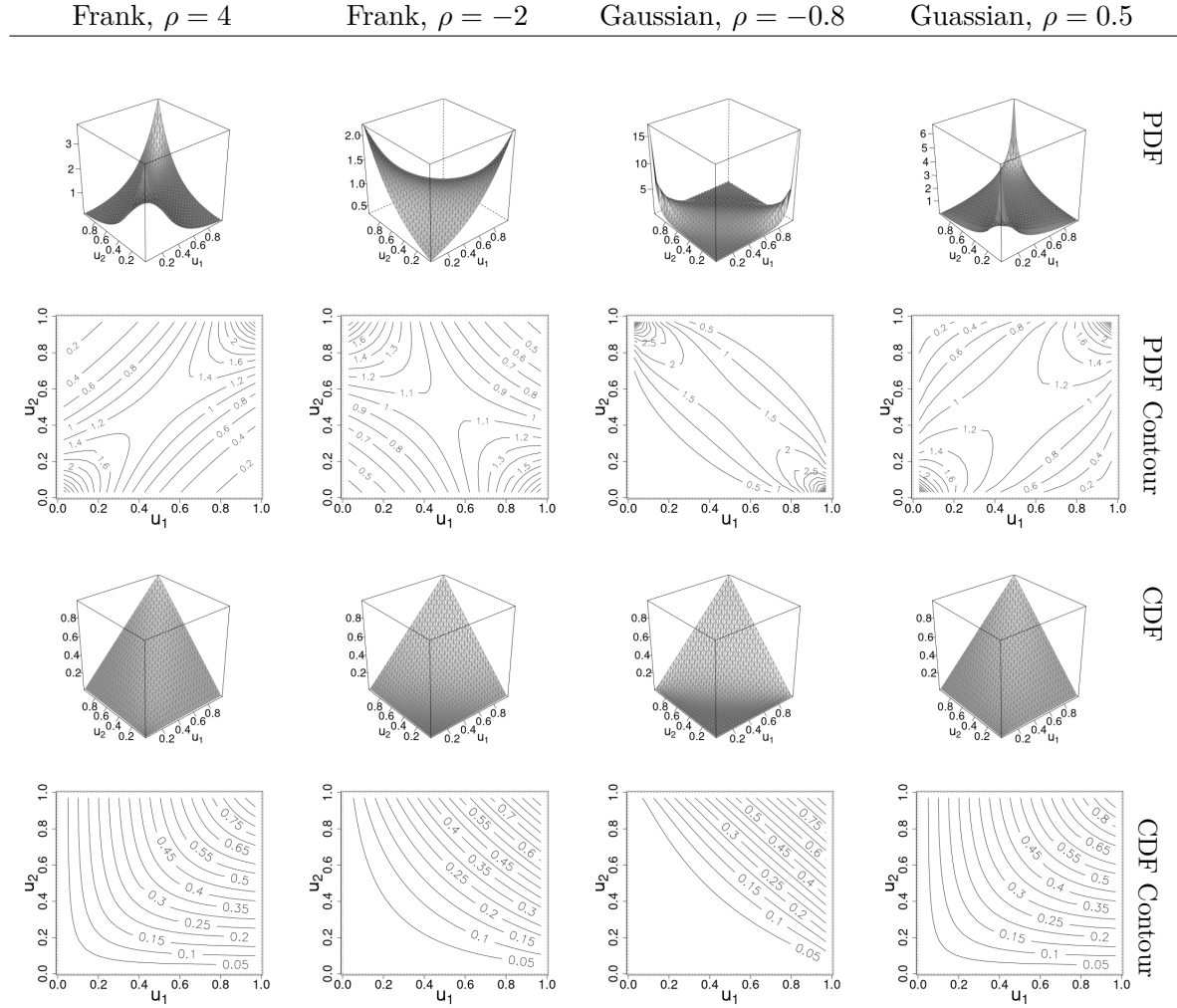


Figure 1: Bivariate Frank and Gaussian copula PDFs and CDFs.

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 2019) fits bivariate Poisson and zero-truncated bivariate Poisson models. This package is not actively maintained on the Comprehensive R Archive Network (CRAN) at the time of this

writing, however, source code is available on GitHub. 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 (2010b) 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.⁵ 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).

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 (Krämer and Silvestrini 2014) fits gamma and zero-truncated Poisson generalized linear models (GLMs) via copula, although the package is not being maintained on CRAN as of this writing. Second, the **CopulaCenR** package (Sun and Ding 2020) 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.⁶ 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.⁷ 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

⁵In fact, it permits arbitrary diagonal inflation, not just zero-inflation.

⁶For further details, see Marra and Radice (2017) and Van der Wurp, Groll, Kneib, Marra, and Radice (2019).

⁷Some of these estimation approaches are further discussed in Inouye *et al.* (2017).

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 provide the R package, **bizicount**, to estimate copula-based zero-inflated bivariate count models, test for zero-modification, determine an appropriate copula function, and then carry out post-estimation diagnostics and produce professional tables.

4. Package contents and implementation

The methods detailed in Section 2 are implemented with a package called **bizicount**, which is available from CRAN at <https://CRAN.R-project.org/package=bizicount>. The main function in the **bizicount** package is `bizicount()` which can be used to estimate a variety of bivariate count data models, including those with zero-inflation in either or both marginal models.⁸

4.1. Main function arguments

A minimal call of the `bizicount()` function requires three user-supplied arguments: `fmla1`, `fmla2`, and `data`. The formulas for the two marginal count processes, `fmla1` and `fmla2`, each require a non-negative integer y on the left-hand side. The right-hand side of the formula must include a set of covariates x for the count process, and can optionally include covariates z for the zero-inflation process. As in the `zeroinfl()` function from the **pscl** package (Zeileis *et al.* 2008), a vertical bar glyph (`|`) is used separate the two sets of covariates using the **Formula** package (Zeileis and Croissant 2010), e.g., `y_j ~ x_j | z_j`. Keeping the assignment of `fmla1` and `fmla2` separate means that users can include common or unique covariates in each, as well as additional supported model options (e.g., offsets).

Additional arguments in the `bizicount()` function specify the marginal probability distributions (`margins`), link functions (`link.zi` and `link.ct`), and copula (`cop`) to be used. Details on the supported options are provided in Table 1 and demonstrated in the next section. In short, if `fmla1` and `fmla2` do not include a zero-inflation component, then users use `margins` to select the Poisson (the default) or negative binomial as the assumed marginal probability distributions, and `cop` to select a Frank (`"frank"`) or Gaussian (`"gaus"`) copula. If instead `fmla1` or `fmla2` *do* include a zero-inflation component, then the available marginals are instead the zero-inflated Poisson (default) and zero-inflated negative binomial. Here users are further able to specify the link function (options given in Table 1) for both the zero-inflation and count process respectively using the `link.zi` and `link.ct` arguments. Use of these arguments is illustrated in detail using a real-data example in the next section.

The `data` argument should indicate the dataframe used in the model. If left unspecified, `bizicount()` will search the global environment for objects with names found in the variable

⁸The package also has implementations for univariate zero-inflated count regression, via `zic.reg()`, as well as functions for evaluating zero-inflated count distributions and simulating from them. These are detailed in Appendix C.

Argument	Description
<code>fmla1, fmla2</code>	formulas specifying the marginal model formulas. For non-inflated margins, these appear as $y_i \sim x_{i1} + x_{i2} + \dots + x_{ik}$, for margins i and covariates $x = 1, 2, \dots, k$. For inflated margins, this appears as $y_i \sim x_{i1} + \dots + x_{ik} \mid z_{i1} + \dots + z_{ip}$, where z are covariates affecting zero-inflation.
<code>data</code>	An optional <code>data.frame</code> containing all variables appearing the model. If not specified, the global environment is searched for variables with the names found in <code>fmla1</code> and <code>fmla2</code> .
<code>cop</code>	A length-one character string specifying the copula to be used. One of "gaus" or "frank". Default is "gaus".
<code>margins</code>	A length-two character string specifying each marginal distribution, with each element being one of "pois", "nbinom", "zip", "zinb". Default is <code>c("pois", "pois")</code>
<code>link.ct</code>	A length-two character string specifying the link function used for the count portion of each marginal distribution, with each element being one of "log", "identity", "sqrt". Default is <code>c("log", "log")</code> .
<code>link.zi</code>	A length-two character string specifying the link function used for the zero-inflation portion of each marginal distribution, with each element being one of "logit", "probit", "cauchit", "log", "cloglog".
<code>starts</code>	An optional vector of starting values for the parameters to be used in numerical optimization. See the help files (<code>?bizicount</code>), for details on ordering these starting values correctly. If not provided, a marginal fit is used to obtain starting values.
<code>keep</code>	Whether to store the source model matrices, offsets, and weights in the model output. Used to conserve memory if necessary. Default is <code>TRUE</code> .
<code>subset</code>	An optional logical vector indicating the subset of observations to be used in fitting.
<code>weights</code>	An optional vector of weights assigned to each observation in fitting.
<code>frech.min</code>	lower boundary for Frechet-Hoeffding bounds on copula CDF. Used for computational purposes to prevent over/underflow in likelihood search. See help files (<code>?bizicount</code>) for greater detail, as well as Section A. Default is <code>1e-7</code> .
<code>pmf.min</code>	Lower boundary on copula PMF evaluations. Used for computational purposes to prevent over/underflow in likelihood search. See help files (<code>?bizicount</code>) for greater detail, as well as Section A. Default is <code>1e-7</code> .
<code>...</code>	Additional arguments that are passed on to the numerical fitting algorithm, <code>nlm()</code> . See <code>?nlm</code> for parameters that may be useful to alter.

Table 1: `bizicount()` argument descriptions.

with for `fmla1` and `fmla2`. Within the `bizicount()` function, users can further refine the data introduced to the model by using the `subset` and `weights` arguments, where the former is a logical vector indicating the subset of observations to be used and the latter is a vector of weights assigned to each observation.

Lastly, with `bizicount()` users have the option to specify features of the estimation process. First, researchers may input a custom set of starting values for the likelihood optimization via the `starts` argument. If no starting values are supplied by the user, the function

Function	Description
<code>print</code>	Prints the simplified vector of coefficient estimates
<code>summary</code>	Returns same model, gives greater details in <code>print</code>
<code>coef</code>	Returns a vector of model coefficients
<code>fitted</code>	Returns $n \times 2$ matrix of fitted values for each margin
<code>logLik</code>	Returns log-likelihood at convergence point
<code>nobs</code>	Returns number of observations used in estimation
<code>AIC</code>	Returns Akaike Information Criterion (Akaike 1973)
<code>BIC</code>	Returns Schwartz-Bayesian Information Criterion (Schwarz 1978)
<code>vcov</code>	Returns the covariance matrix of model parameters
<code>extract</code>	Returns a <code>texreg</code> (Leifeld 2013) object
<code>simulate</code>	Returns simulated matrices for use in <code>DHARMa</code> (Hartig 2022)
<code>make_DHARMa</code>	Returns a list of <code>DHARMa</code> -class objects for the model
<code>zi_test</code>	Carries out He, Zhang, Ye, and Tang (2019) 's test for zero modification.

Table 2: 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.

automatically uses the estimates of coefficients from a univariate fit for each margin, with the dependence parameter set to the Spearman correlation between the two outcomes. Second, since `bizicount()` optimizes the likelihood using R's `nlm()` function, users can pass most any option to `nlm()` via the `...` argument. Among these options, we have found that `stepmax`, which controls the maximum allowable step-size used in optimization, can often be useful to overcome convergence or singularity issues. Details on this can be found in the help files for `nlm()`.

4.2. Main function output and post estimation

The `bizicount()` function returns an S3 object of class `bizicount` that has a host of generic methods, including `print`, `summary`, `coef`, etc. Table 2 below includes short descriptions of each. In addition, `bizicount()` can also output 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.

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 L^AT_EX formats, and coefficient plots drawn. Compatibility with `texreg` is achieved by defining an `extract()` method for `bizicount` objects. Briefly, the `extract()` function (`extract.bizicount(model, CI = NULL, id=T)`) has three arguments when used on a `bizicount` object: `model` is the `bizicount` model object, `CI` is a number on the unit interval specifying the desired two-tailed confidence level, and `id` which indicates whether model identifiers (i.e., `ct_` for count, and `zi_` for zero-inflation) should be prepended to the results.

Finally, users may wish to do various forms of diagnostic tests to assess model fit. In typical

GLM settings, this would involve inspecting residuals of some form, however, it is well known that the discrete nature of most outcomes in GLMs makes residual analysis difficult and often misleading (Dunn and Smyth 1996). Instead, R's **DHARMa** package (Hartig 2022) offers non-parametric, simulation-based techniques to assess model fit, outliers, dispersion, zero-inflation, etc. 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**. Second, because there are two sets of simulations – one for each margin – 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 below in Section 5.2. As this is simply a wrapper around **DHARMa**'s `createDHARMa()` function, users are referred to the corresponding documentation for details.

5. Data analysis example: Terrorism in Nigeria

To demonstrate the use of the main `bizicount()` regression function and its most essential methods, we now provide an applied example using terrorism in Nigeria. In the following, we model counts of terror attacks in Nigeria by two terrorist groups (Boko Haram and Fulani extremists) using data from the Global Terrorism Database (GTD, LaFree and Dugan 2007; START 2018). The GTD is an event-level data set cataloging individual terror attacks, which we aggregate to the PRIO-GRID square (each square is 0.5×0.5 decimal degrees) level.⁹ As such, the outcomes in our analysis, `att.ful` and `att.bok`, are the number of terror attacks Boko Haram and Fulani extremists, respectively, in 2014 for a given PRIO-GRID square. We model these as a function of covariates using data from the PRIO-GRID 2.0 dataset (Tollefsen, Bahgat, Nordkvelle, and Buhaug 2015; Tollefsen *et al.* 2012), including population (Center for International Earth Science Information Network (CIESIN) & Centro Internacional 2005), percent mountainous terrain (Blyth, Groombridge, Lysenko, Miles, and Newton 2002), and the latitude and longitude of grid centroids. These inputs are mean centered and scaled by one standard deviation in order to avoid ill-conditioning and convergence issues due to the vastly different scales of the covariates. We encourage users to similarly standardize their variables as desired using the `standardize` package in R (Eager 2021), or the `base::scale()` function.

5.1. `bizicount()` regression models and testing for zero-inflation

First, we load the required packages and data:

```
R> library("bizicount")
R> library("texreg")
R> library("DHARMa")
R> data("terror", package = "bizicount")
```

⁹For details on coding methodology and criteria, see the GTD codebook at <https://www.start.umd.edu/gtd/downloads/Codebook.pdf>. For details on PRIO-GRID squares see Tollefsen, Strand, and Buhaug (2012).

```
R> dat <- terror
R> dat[, c("xcoord", "ycoord", "pop", "mtns")] <- scale(
+   dat[, c("xcoord", "ycoord", "pop", "mtns")])
R> fmla.ful <- att.ful ~ pop + mtns + xcoord * ycoord
R> fmla.bok <- att.bok ~ pop + mtns + xcoord * ycoord
```

With the specifications of the marginal models, we can now estimate various bivariate specifications using `bizicount()`. To begin, we consider a bivariate Poisson model without zero inflation as:

```
R> bivpois <- bizicount(fmla.ful, fmla.bok, data = dat,
+   cop = "frank", margins = c("pois", "pois"), keep = TRUE)
```

Here we see that `margins` indicates both marginals should be Poisson, and `cop` indicates that a Frank copula should be used. Using the `summary()` method we obtain details on coefficient estimates, estimated standard errors, test statistics, and corresponding p values:

```
R> summary(bivpois)
```

Call:

```
bizicount(fmla1 = fmla.ful, fmla2 = fmla.bok, data = dat, cop = "frank",
  margins = c("pois", "pois"), keep = TRUE)
```

```
=====  
Count Model: att.ful |  
-----
```

	Estimate	Std. Err.	Z value	Pr(> z)
(Intercept)	-1.26185	0.12291	-10.2661	< 2.2e-16 ***
pop	-0.44110	0.20166	-2.1873	0.02872 *
mtns	-0.44051	0.10896	-4.0429	5.278e-05 ***
xcoord	0.72976	0.14268	5.1147	3.143e-07 ***
ycoord	-0.69300	0.12264	-5.6505	1.600e-08 ***
xcoord:ycoord	-1.03695	0.17133	-6.0523	1.428e-09 ***

```
-----  
                Estimate Std. Err. Z value Pr(>|z|)  
dependence 1.64169 0.75955 2.1614 0.03066 *  
-----
```

```
Count Model: att.bok |  
-----
```

	Estimate	Std. Err.	Z value	Pr(> z)
(Intercept)	-2.378585	0.197090	-12.0685	< 2.2e-16 ***
pop	0.944876	0.062597	15.0945	< 2.2e-16 ***
mtns	-0.050484	0.056749	-0.8896	0.3737


```

xcoord          2.634690  0.161948  16.2688 < 2.2e-16 ***
ycoord          0.952284  0.153673   6.1968 5.762e-10 ***
xcoord:ycoord -0.585055  0.119683  -4.8884 1.017e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
=====

```

We save a full discussion of the results until later when we can compare across models. Briefly, however, we see that population is negatively related with attacks by Fulani extremists, and positively related with attacks by Boko Haram. This is consistent with observed locations of attacks by the two groups; as Fulani extremists tend to operate mainly in rural areas, and Boko Haram tend to target high-population areas to elicit more casualties. The dependence parameter is positive and statistically significant, which indicates that the two groups should be more likely to attack in similar areas. In reality, however, the observed distribution of attacks is more consistent with a repulsive pattern, as these groups tend to operate in different parts of the country.

This latter finding may be a consequence of neglecting zero-inflation in one or both of these marginal processes. As such, we now test for the presence of excess zeroes using `zi_test()` which is an implementation of the test proposed in [He et al. \(2019\)](#). Briefly, the test compares the observed conditional proportion of zeros to that expected under the null hypothesis of a Poisson distribution. In a regression context, the test is preferred over alternatives such as [Vuong \(1989\)](#)'s test, Wald, score or likelihood ratio tests, as the parameters of the zero-inflated Poisson do not meet the assumptions of these tests. Readers are referred to [Wilson \(2015\)](#), as well as [He et al. \(2019\)](#) and [Tang and Tang \(2019\)](#) for greater detail.

```
R> zi_test(bivpois)
```

```

=====
He et al. (2019)'s Test for Zero Modification
-----
H_0: Pr(y = 0 | x) = dpois(0 | x)

      H_a   Z_score   p_value   n
att.ful inflated 10.809096 1.558638e-27 312
att.bok inflated  8.404112 2.155557e-17 312
=====

```

For both marginal models, we reject the null of no-zero inflation at very high levels of significance (p values of $1.56e-27$ and $2.16e-17$). This indicates that a model which accounts for excess zeros may provide a better fit to these data, as such we now modify the specification of our `bizicount()` to allow for that. We can estimate the zero-inflated model in exactly the same way as the earlier non-inflated model, with some minor adjustments to the `fmla1`, `fmla2` and `margins` parameters.

First, defining zero-inflated formulas can be done as follows:

```
R> fmla.zi.ful <- att.ful ~ pop + mtns + xcoord * ycoord |
+   pop + mtns + xcoord * ycoord
R> fmla.zi.bok <- att.bok ~ pop + mtns + xcoord * ycoord |
+   pop + mtns + xcoord * ycoord
```

As seen above, zero-inflated formulas take the form of $y \sim x \mid z$, where y is a response, x are covariates in the count portion of the model, and z are covariates in the zero-inflated portion, separated from the count portion by the $|$ bar symbol.

Next, we alter the `margins` argument to indicate that each margin is zero-inflated Poisson, e.g., `margins = c("zip", "zip")`.

```
R> zi_bivpois = bizicount(fmla.zi.ful, fmla.zi.bok, data = dat,
+   cop = "frank", margins = c("zip", "zip"), keep = TRUE)
```

After running this function, the results can be output using the model summary as before.

```
R> summary(zi_bivpois)
```

Call:

```
bizicount(fmla1 = fmla.zi.ful, fmla2 = fmla.zi.bok, data = dat,
  cop = "frank", margins = c("zip", "zip"), keep = TRUE)
```

```
=====
Count Model: att.ful |
-----

              Estimate Std. Err. Z value Pr(>|z|)
(Intercept)   0.28469   0.12039   2.3647  0.01805 *
pop           -0.76644   0.36025  -2.1275  0.03338 *
mtns          0.25733   0.10566   2.4354  0.01487 *
xcoord       -2.04704   0.26001  -7.8729 3.465e-15 ***
ycoord       -1.00707   0.14800  -6.8044 1.015e-11 ***
xcoord:ycoord -1.57148   0.30833  -5.0967 3.456e-07 ***
```

```
+++++
```

```
Zero Inflation: att.ful |
-----
```

```
              Estimate Std. Err. Z value Pr(>|z|)
(Intercept)  -3.46119   1.07864  -3.2089 0.0013326 **
pop          -0.43413   0.62324  -0.6966 0.4860746
mtns         1.73496   0.48970   3.5429 0.0003957 ***
xcoord     -10.54553   2.38677  -4.4183 9.947e-06 ***
ycoord      -4.74161   1.22946  -3.8567 0.0001150 ***
xcoord:ycoord -6.89191   1.97450  -3.4905 0.0004822 ***
```

```
-----
              Estimate Std. Err. Z value Pr(>|z|)
```

```
dependence -0.76749    1.31529 -0.5835    0.5595
-----
```

```
Count Model: att.bok |
-----
```

	Estimate	Std. Err.	Z value	Pr(> z)	
(Intercept)	0.567623	0.193178	2.9383	0.003300	**
pop	0.232673	0.076434	3.0441	0.002334	**
mtns	-0.019449	0.052042	-0.3737	0.708611	
xcoord	1.217380	0.148355	8.2059	2.29e-16	***
ycoord	0.162435	0.210258	0.7725	0.439790	
xcoord:ycoord	-0.402957	0.127004	-3.1728	0.001510	**

```
+++++
```

```
Zero Inflation: att.bok |
-----
```

	Estimate	Std. Err.	Z value	Pr(> z)	
(Intercept)	1.88502	0.27614	6.8264	8.71e-12	***
pop	-0.75792	0.27215	-2.7849	0.005354	**
mtns	-0.15286	0.20591	-0.7424	0.457857	
xcoord	-0.77939	0.34073	-2.2874	0.022173	*
ycoord	-0.52526	0.27307	-1.9236	0.054407	.
xcoord:ycoord	-0.90781	0.31491	-2.8827	0.003942	**

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
=====
```

Looking at the results, we observe several differences as compared to the non-inflated, bivariate Poisson model from before. Perhaps most importantly, after accounting for zero-inflation, we fail to reject the null of no dependence between the two groups' attacks. Despite this null result, the dependence parameter is in the negative direction, which is consistent with the spatial patterns of attacks by the two groups.

In each of the sample-specific results, we see different relationships from what we found previously in the non-inflated models. For attacks by Fulani extremists, we see that population affects the odds of an inflated zero (i.e., of units not being at risk of an attack), but not expected count of attacks. Mountainous terrain, on the other hand, both decreases the odds of an inflated zero, and increases the expected number of attacks. This is in contrast to the first set of results in Section 5.1, which indicated that mountainous terrain was *negatively* related to attacks by Fulani extremists. Finally, the latitude and longitude coefficients are consistent with theoretical expectations for Fulani extremism, as they suggest that as we move further north and east, the chances of an inflated zero are increased, and the expected number of attacks are reduced.

For the Boko Haram sample of attacks, population is associated with an increase in the expected number of attacks, as was the case in the the first set of results from Section 5.1.

However, the magnitude of the coefficient is nearly 75% lower than before, with a one-standard deviation increase in population corresponding to a 0.945 increase in the expected log count in the non-zero inflated models as compared to only 0.233 in the zero-inflated adjusted results here.¹⁰ As before, mountains are not associated with attacks by Boko Haram. Finally, as we move further north and east, Boko Haram is more likely to experience a structural zero, and is more likely to attack up to a certain degree of northward movement.

In summary then, by testing and accounting for zero inflation, substantive conclusions are changed regarding the effects of covariates for both of the terror groups attacks. Moreover, after accounting for zero-inflated counts, we now fail to reject the null of independence across the two groups' attack patterns. This is in contrast to our earlier finding that their attacks are *positively* related, which contradicts subject matter knowledge on the two groups' areas of operation. This highlights the importance of jointly account for zero-inflation and cross-sample dependence, as the former would have caused us to erroneously conclude in support of the latter.

5.2. **bizicount** post-estimation diagnostics with **DHARMa**

In Section 5.1, we demonstrate **bizicount**'s implementation of He *et al.* (2019)'s test for zero modification. However, this test can only indicate how well a Poisson model fits zero counts. Outliers, dispersion, and the suitability of a marginal zero-inflated count model are often also of interest. As such, we now detail additional tools for post-estimation diagnostics on **bizicount** objects.

In typical GLM settings, post-estimation diagnostics might 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 some outcomes in GLMs makes residual analysis difficult if not misleading (Dunn and Smyth 1996). In an attempt to rectify this gap, R's **DHARMa** package (Hartig 2022) offers non-parametric, simulation-based techniques to assess model fit, outliers, dispersion, zero-inflation, and independence across a range of models. 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 `make_DHARMa()` function, which, when called on a **bizicount**-class object, returns a list of **DHARMa**-class objects that will interface with all of that package's methods. Second, because most of **DHARMa**'s methods rely on simulated datasets, we provide a `simulate()` method that can be called on **bizicount**-class objects, in the event that users need access to the raw simulation output used in **DHARMa**'s methods.

Using the `make_DHARMa()` method, we can diagnose the bivariate, zero-inflated Poisson regression model from the previous section. As demonstrated below, we create the list of **DHARMa** object, and then call the associated `testResiduals` method for residual diagnostics. This allows users to see a set of potentially useful results with a single line of code.

```
R> zip_dharmas = make_DHARMa(zi_bivpois, nsim = 5000, seed = 789443)
R> class(zip_dharmas)
```

¹⁰As in any GLM, the coefficients may not directly reflect the quantities of interest desired by a researcher. Here since we have used a logit to model the zero inflation and a Poisson for the count process, some direct interpretation of results – as log odds and expected log count – is easily done. Calculation of other quantities of interest (e.g., marginal effects) are not produced directly by the `bizicount()` function.

```
[1] "list"
```

```
R> lapply(zip_dharmas, class)
```

```
[[1]]
```

```
[1] "DHARMa"
```

```
[[2]]
```

```
[1] "DHARMa"
```

```
R> lapply(zip_dharmas, testResiduals, plot = FALSE)
```

```
$uniformity
```

```
Asymptotic one-sample Kolmogorov-Smirnov test
```

```
data: simulationOutput$scaledResiduals
```

```
D = 0.041527, p-value = 0.6549
```

```
alternative hypothesis: two-sided
```

```
$dispersion
```

```
DHARMa nonparametric dispersion test via sd of residuals fitted vs.  
simulated
```

```
data: simulationOutput
```

```
dispersion = 4.0577, p-value = 4e-04
```

```
alternative hypothesis: two.sided
```

```
$outliers
```

```
DHARMa bootstrapped outlier test
```

```
data: simulationOutput
```

```
outliers at both margin(s) = 2, observations = 312, p-value < 2.2e-16
```

```
alternative hypothesis: two.sided
```

```
percent confidence interval:
```

```
0.000000000 0.003205128
```

```
sample estimates:
```

```
outlier frequency (expected: 0.00016025641025641 )
```

```
0.006410256
```

```
$uniformity
```

```
Asymptotic one-sample Kolmogorov-Smirnov test
```

```
data: simulationOutput$scaledResiduals
D = 0.067001, p-value = 0.1214
alternative hypothesis: two-sided
```

```
$dispersion
```

```
      DHARMA nonparametric dispersion test via sd of residuals fitted vs.
      simulated
```

```
data: simulationOutput
dispersion = 4.0115, p-value < 2.2e-16
alternative hypothesis: two.sided
```

```
$outliers
```

```
      DHARMA bootstrapped outlier test
```

```
data: simulationOutput
outliers at both margin(s) = 4, observations = 312, p-value < 2.2e-16
alternative hypothesis: two.sided
percent confidence interval:
 0 0
sample estimates:
outlier frequency (expected: 0 )
          0.01282051
```

Although the output may appear daunting, the interpretation of these residual tests is straightforward.¹¹ First, note that each item is labeled with either `[[1]]` or `[[2]]`, indicating the appropriate margin. Starting with margin 1, we notice that the Kolmogorov-Smirnov (KS) test for the residuals is insignificant, suggesting that the zero-inflated Poisson distribution may be an appropriate assumption for the Fulani extremists' counts (as we fail to reject in favor of the alternative). The overdispersion test for margin 1, however, is consistent with over-dispersion when fitting a zero-inflated Poisson distribution for Fulani extremists.¹² The size of the estimated overdispersion parameter is 4, which is roughly 4 times that expected under the null hypothesis. Finally, the test for marginal outliers for Fulani extremists is significant. However, with only 312 observations, the actual frequency of these outliers is still quite small. The same set of diagnostic test results are then repeated for margin 2 (here Boko Haram). Briefly, these results are similar to above: the KS test is insignificant, while the overdispersion and outlier tests are significant.

5.3. Accounting for overdispersion

Based on this review of the post-estimation diagnostics, further model revision is suggested to

¹¹The **DHARMA** package also has numerous methods for residual *plots*. To conserve space, we print only the console output here. Example code for residual plots is available in Appendix B; however, readers are referred to the **DHARMA** documentation for greater details on residual plotting.

¹²This is despite the fact that the zero-inflated Poisson accommodates some forms of overdispersion inherently, as $\text{Var}(Y_i; \lambda_i, \psi_i) = (1 - \psi_i)(\lambda_i + \psi_i \lambda_i^2) > \mathbb{E}(Y_i; \lambda_i, \psi_i) = (1 - \psi_i)\lambda_i, \forall \psi_i \in (0, 1)$.

account for excess variance. One possible approach is to fit a zero-inflated negative binomial model, which will better account for overdispersion. We therefore fit a **bizicount** model to the terrorism data below using zero-inflated negative binomial margins. We then carry out a likelihood ratio test to determine if adding the inverse dispersion parameters to the distribution improves our fit, and again undertake **DHARMA**'s residual tests. To conserve space, we do not print **bizicount**'s raw regression output for the zero-inflated negative binomial model here, as we intend only to show how readers might proceed with subsequent model diagnosis.

```
R> zi_bivnb = bizicount(fmla.zi.ful, fmla.zi.bok, data = dat, cop = "frank",
+   margins = c("zinb", "zinb"), keep = TRUE)
R> chi_stat_nb = -2 * (logLik(zi_bivpois) - logLik(zi_bivnb))
R> pchisq(chi_stat_nb, df = 2, lower.tail = FALSE)
```

```
[1] 1.202813e-53
```

```
R> zinb_dharmas = make_DHARMA(zi_bivnb, nsim = 5000, seed = 12473)
R> lapply(zinb_dharmas, testResiduals, plot = FALSE)
```

```
$uniformity
```

```
Asymptotic one-sample Kolmogorov-Smirnov test
```

```
data: simulationOutput$scaledResiduals
D = 0.021347, p-value = 0.9989
alternative hypothesis: two-sided
```

```
$dispersion
```

```
DHARMA nonparametric dispersion test via sd of residuals fitted vs.
simulated
```

```
data: simulationOutput
dispersion = 1.4655, p-value = 0.2976
alternative hypothesis: two.sided
```

```
$outliers
```

```
DHARMA bootstrapped outlier test
```

```
data: simulationOutput
outliers at both margin(s) = 1, observations = 312, p-value = 0.14
alternative hypothesis: two.sided
percent confidence interval:
0.000000000 0.003205128
sample estimates:
outlier frequency (expected: 0.000224358974358974 )
0.003205128
```

`$uniformity`

Asymptotic one-sample Kolmogorov-Smirnov test

```
data: simulationOutput$scaledResiduals
D = 0.039582, p-value = 0.7126
alternative hypothesis: two-sided
```

`$dispersion`

DHARMA nonparametric dispersion test via sd of residuals fitted vs. simulated

```
data: simulationOutput
dispersion = 0.53827, p-value = 0.5048
alternative hypothesis: two.sided
```

`$outliers`

DHARMA bootstrapped outlier test

```
data: simulationOutput
outliers at both margin(s) = 0, observations = 312, p-value = 1
alternative hypothesis: two.sided
percent confidence interval:
 0.000000000 0.003205128
sample estimates:
outlier frequency (expected: 0.000192307692307692 )
0
```

Both from the likelihood ratio test and from **DHARMA**'s output, we can see clearly that accounting for overdispersion vastly improves the model's fit, with overdispersion, outliers, and quantile deviations no longer detectable.

In summary, using diagnostics contained within the **bizicount** package we arrive at a bivariate zero-inflated negative binomial model. [He *et al.* \(2019\)](#)'s test demonstrate the need to account for zero inflation in the marginals, and subsequent residual diagnostics indicate alack of fit (namely excess variance) in the zero-inflated Poisson margins, leading us to better account for overdispersion using the zero-inflated negative binomial model.

Keeping this example in mind, it is important to mention that poor fit is often a result of a poorly specified model mean structure. For space considerations here, we do not further investigate adding additional covariates, transforming them, or adding interactions/polynomials, as the main goal here is an exposition of the tools offered by the package. However, in general this type of exploration would offer a much richer analysis of any data, and is often the correct approach (rather than accounting for variation in a dispersion parameter). The copula nature of these models does not alter typical approaches to model exploration, including like-

	Bivariate Poisson		Bivariate ZIP		Bivariate ZINB	
	Fulani	Boko Haram	Fulani	Boko Haram	Fulani	Boko Haram
<i>Count model</i>						
(Intercept)	-1.26*** (0.12)	-2.38*** (0.20)	0.28* (0.12)	0.57** (0.19)	0.20 (0.24)	-0.17 (0.31)
Population	-0.44* (0.20)	0.94*** (0.06)	-0.77* (0.36)	0.23** (0.08)	-0.72 (0.77)	0.51** (0.16)
% Mountains	-0.44*** (0.11)	-0.05 (0.06)	0.26* (0.11)	-0.02 (0.05)	0.17 (0.17)	-0.17 (0.19)
Longitude	0.73*** (0.14)	2.63*** (0.16)	-2.05*** (0.26)	1.22*** (0.15)	-2.13*** (0.48)	2.24*** (0.29)
Latitude	-0.69*** (0.12)	0.95*** (0.15)	-1.01*** (0.15)	0.16 (0.21)	-1.08** (0.33)	-1.26*** (0.37)
Lat. × Long.	-1.04*** (0.17)	-0.59*** (0.12)	-1.57*** (0.31)	-0.40** (0.13)	-1.12 (0.60)	-0.22 (0.26)
<i>Zero inflation</i>						
(Intercept)			-3.46** (1.08)	1.89*** (0.28)	-6.04** (2.19)	-0.67 (1.02)
Population			-0.43 (0.62)	-0.76** (0.27)	-0.52 (0.91)	-1.81* (0.72)
% Mountains			1.73*** (0.49)	-0.15 (0.21)	2.20** (0.82)	0.25 (0.51)
Longitude			-10.55*** (2.39)	-0.78* (0.34)	-14.38*** (4.05)	0.47 (0.71)
Latitude			-4.74*** (1.23)	-0.53 (0.27)	-6.58** (2.12)	-5.33** (1.72)
Lat. × Long.			-6.89*** (1.97)	-0.91** (0.31)	-8.79** (2.96)	-1.85 (1.17)
Dispersion ⁻¹					0.64** (0.21)	0.68*** (0.16)
ρ		1.64* (0.76)		-0.77 (1.32)		0.04 (1.35)
N		312		312		312
BIC		1442.68		1097.95		855.63
AIC		1459.67		1130.62		890.91
LogLik		-716.84		-540.31		-418.46

Table 3: Bivariate Poisson, zero-inflated Poisson, and zero-inflated negative binomial models of terrorist attacks, Frank copula. Standard errors are in parentheses, with *** $p < 0.001$; ** $p < 0.01$; and * $p < 0.05$.

likelihood ratio, Wald, and score testing for nested models.¹³ As such, variable selection can (and

¹³However, as noted earlier, it is incorrect to test the Poisson as being nested in the zero-inflated Poisson in a regression context (Wilson 2015; Tang and Tang 2019).

should) proceed in the same way as it would for any generalized linear model. We recommend readers consult basic econometrics texts (Greene 2003) or specific treatments of count data analysis (Cameron and Trivedi 2013) for further guidance on these points.

5.4. Tabularizing **bizicount** models using **texreg**

Although inspecting the raw console output is useful for model building, after arriving at a set of models to be presented to a wider audience, users will likely want to export the results in a more compact and neat form. For this reason, we provide methods for the **texreg** package's generic functions (Leifeld 2013), which allows users to output **bizicount** models as \LaTeX code, plain-text, or a Microsoft Word document. For example, we can more succinctly present the results of the preceding regression models in a \LaTeX table using the following code, which results in the \LaTeX code to produce Table 3.

```
R> mods <- list(bivpois, zi_bivpois, zi_bivnb)
R> texreg(mods, groups = list("Count Model" = 1:6, "Zero Inflation" = 7:12),
+   reorder.coef = c(1:6, 8:13, 14, 7), digits = 2,
+   custom.model.names = c("Poisson: Fulani", "Poisson: BH", "ZIP: Fulani",
+   "ZIP: BH", "ZINB: Fulani", "ZINB: BH"))
```

The **texreg** package comes with a host of additional options for customizing output, only a few of which we utilize in this example. Users are referred to its documentation for greater detail on modifying output to suit their specific needs.

6. Handling convergence problems

Although it did not happen in our terrorism data example here, since **bizicount** uses likelihood-based methods users may encounter convergence issues in their models. First, **bizicount** utilizes numerical methods for optimization, implying several layers of finite difference approximations, first to obtain the copula PMF from its CDF, and then to obtain approximate gradients and Hessians for the quasi-Newton fitting procedure. Second, covariates may often be on different scales, which can drastically affect the gradient vector if these scales are very different and result in an ill-conditioned (and therefore non-invertible) Hessian. Third, users may also face convergence issues if their marginal distributions are misspecified, although this is rare in application. For example, the negative binomial distribution collapses to a Poisson when the inverse dispersion parameter goes to infinity.¹⁴ Similarly, if there is no zero-inflation in the data-generating process and users specify zero-inflated marginal distributions, then the theoretical zero-inflation coefficients tend to negative infinity. This fact underscores the necessity of adequate prior testing for zero-inflation – namely using He *et al.* (2019)'s test – as it requires only a Poisson regression fit, thereby foregoing the issue of fitting a non-convergent zero-inflated model prior to testing for zero-inflation.

Although none of these presented a problem in the above analysis, several warnings and errors may indicate non-convergence. They include statements like:

¹⁴We take some steps to mitigate this by warning users when the inverse dispersion parameter is suspiciously large.

- **Convergence code other than 1...** which indicates general convergence problems, with different codes having different meanings. A code of 2 – which is defined by the `nlm()` documentation as “successive iterates within tolerance, current iterate is probably solution” – might be acceptable, but others are not. The documentation for the `nlm()` function includes descriptions of each convergence code, although in general convergence codes 3 through 5 are problematic, and model results are not to be trusted.
- **Probable coding error in analytical gradient...** which indicates issues in obtaining starting values using a univariate fit. This is often solved by scaling covariates due to the reasons mentioned above.
- **nlm was unable to obtain a Hessian; NumDeriv was used...** which indicates that the built in second-order finite difference method for the fitting procedure was unable to obtain an approximate Hessian, so the `numDeriv` package was used (successfully) to obtain a Hessian (Gilbert and Varadhan 2019). Despite this success, the warning indicates potential issues to be resolved.
- **Hessian of LogLik is not negative-definite...** which indicates that the Hessian was either unobtainable, or it exists but is not negative-definite. If the Hessian was not obtainable, this includes the additional attempts by `numDeriv` in place of `nlm()`.

To mitigate non-convergence that is not due to a theoretically infinite coefficient, users have two primary options:

1. Iteratively tuning the `stepmax` parameter in `bizicount()`, which is passed on to the `nlm()` fitter. What number this should be is context dependent, but we have had success anywhere in the range of 0.05 to 25. This can be done by specifying `stepmax` in a call to `bizicount()`, e.g. `bizicount(fmla1, fmla2, stepmax = 0.1)`, along with any other desired parameter settings. The parameter controls the maximum allowable step length along the gradient in the likelihood search, which will slow down the fitting procedure but will avoid taking large steps that may simply jump over the minimum (or maximum). When adjusting this parameter, users will need to be especially wary of reaching the maximum number of iterations (code 4) and exceeding the maximum step length 5 consecutive times (code 5). For code 4, the iteration limit should be increased, e.g., `bizicount(fmla1, fmla2, iterlim = 50000)`, and for code 5, the user has set `stepmax` too small, and should therefore increase it slightly and try fitting again, ideally until no warnings occur.
2. Putting covariates on a similar scale, e.g., by mean-centering and dividing by one standard deviation.¹⁵ This is simple to do in R using its built-in `scale()` function, which can be called directly on a `data.frame` object to scale all of its columns, e.g., `df_scaled = scale(df_unscaled)`. In fact, we use this exact approach in the applied example, with our code found in Section 5.1. Alternatively, individual covariates can be scaled as needed, either by subsetting the user’s `data.frame` appropriately, or by specifying the `scale()` function in the model formula, e.g., `y ~ scale(x) + scale(z)`. Scaling comes with the obvious tradeoff of model interpretability as practitioners are probably not interested in the effects of a change of one standard deviation, but it will often improve numerical stability.

¹⁵Or dividing by two standard deviations, if binary variables are present (Gelman 2008).

Beyond these approaches, users can also consider altering additional parameters that are passed on to `nlm()`, which can all be passed simply by specifying them in the call to `bizicount()`, identically to the `stepmax` parameter discussed in (1) above. Although we have had the most success by adjusting `stepmax`, details on these additional parameters can be found in the `nlm()` function’s documentation.

7. Simulation evidence

The preceding analysis of terrorism data demonstrates the general methods of the **bizicount** package, as well as some of the potential changes to our conclusions that occur when accounting for various features of the data-generating process, e.g., zero-inflation and overdispersion. In that demonstration, for example, we observed that the signs and significance on the dependence parameter and some covariates changed after allowing for zero-inflation, and even more so after accounting for overdispersion. However, this is only a single example and, more importantly, we do not know the true data-generating process to benchmark against. Therefore, to provide a more comprehensive analysis of the consequences of omitting dependence and/or zero-inflation, we provide evidence from a series of Monte Carlo experiments below. Anticipating our results, we see that neglecting dependence results in over-confident and inefficient estimators, omitting zero-inflation results in biased estimators, and omitting both of these processes results in both biased and overconfident estimators. This is consistent with our theoretical expectations mentioned in the introduction, and again demonstrates the importance of having a model which can simultaneously accommodate both zero-inflation and dependence across the marginals.

7.1. Data-generating process

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

$$Y_{ij} \sim ZIP \{ \lambda_{ij} = \exp(\mathbf{X}_{ij}\boldsymbol{\beta}_j), \psi_j = \psi_j \}$$

$$\text{cor} \left[\Phi^{-1}\{F_1(Y_{.1})\}, \Phi^{-1}\{F_2(Y_{.2})\} \right] = \text{cor}(U_1, U_2) = \rho$$

Where

- $i = 1, 2, \dots, N$ are observations.
- $N = 500$ is the sample size, which is constant across our simulations.
- $j \in \{1, 2\}$ are the margin indices.
- $\boldsymbol{\beta}_1 = [1, 3.25, -2.3]^\top$ is a vector of parameters for the count portion of margin 1, including an intercept. This vector is held constant across simulations.
- $\boldsymbol{\beta}_2 = [2, -1.75, 3.5]^\top$ is a vector of parameters for the count portion of margin 2, including an intercept. This vector is held constant across simulations.
- 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.
- $\mathbf{X}_{i1}^{1 \times 2} \sim \text{Bernoulli}(p = 0.5)$ is an observation vector for margin 1, so both covariates have the same parameter p . A 1 is inserted as the first element for the intercept.
- $\mathbf{X}_{i2}^{1 \times 2} \sim \text{Exponential}(\text{rate} = 3)$ is an observation vector for margin 2, so both covariates have the same rate. A 1 is inserted as the first element for the intercept.

Using these data, we then estimate three models:

1. Our preferred bivariate model with zero-inflated Poisson margins using a Gaussian copula. This is labeled as the correct model below as it reflects the true DGP.
2. A bivariate model with Poisson margins and a Gaussian copula – i.e., a bivariate model omitting zero-inflation in the margins.
3. A univariate zero-inflated Poisson model for each outcome, \mathbf{Y}_j – i.e., a univariate model for each outcome vector that omits dependence, but which correctly specifies the marginal distribution.

7.2. Bias and inefficiency of count parameter estimators

The densities in Figure 2 are for the second element of β_1 in margin 1, $\beta_{1,2} = 3.25$, while holding margin 2's zero-inflation parameter constant at $\psi_2 = 0.1$. The densities of the estimates are centered on the true value, i.e., they are $\hat{\beta}_{21} - \beta_{21}$. A few conclusions are immediately apparent:

- The correct bivariate model is unbiased in each set of reported conditions.
- The incorrect bivariate model is biased upward, with the magnitude of this bias increasing in the extent of zero-inflation. This is consistent with our expectations, 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.

7.3. Overconfidence of count parameter estimators

The plot in Figure 3 demonstrates the level of overconfidence in the estimated parameter $\beta_{1,2}$ compared to its empirical sampling distribution, again holding margin 2's zero-inflation fixed at $\psi_2 = 0.1$. We measure overconfidence as

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

where $MAD(\cdot)$ denotes the median absolute deviation from the median.

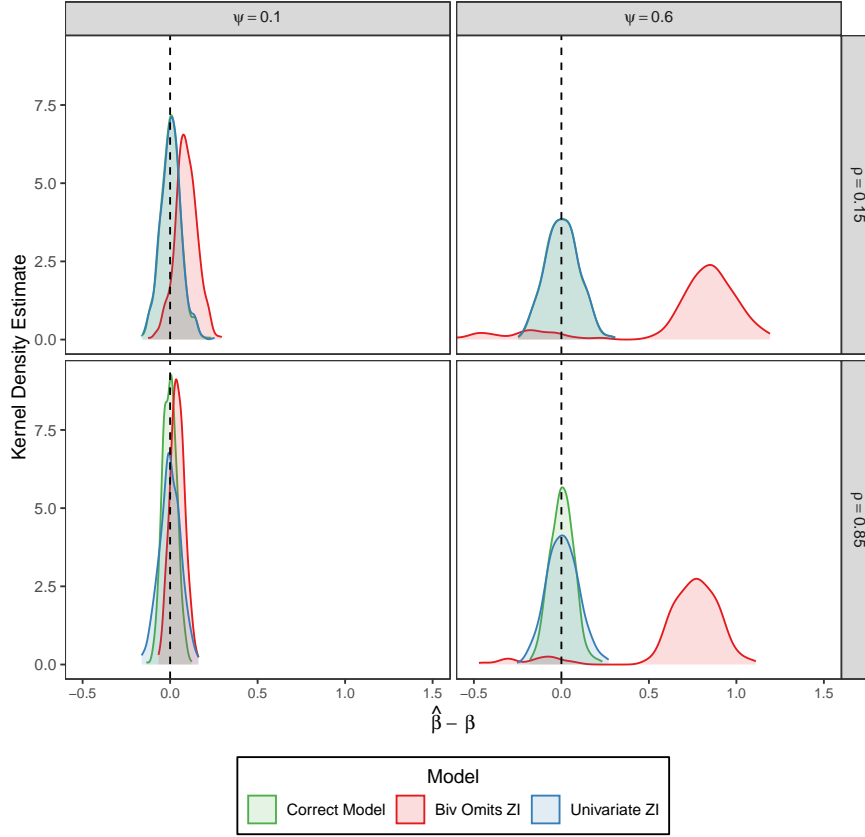


Figure 2: Density of centered count slope parameter estimates ($\beta_{1,2} = 3.25$) in margin 1, with $\psi_2 = 0.1$.

That is, we compare an estimate of the dispersion of $\hat{\beta}$ (using its sampling distribution) 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 holds (i.e., a measure less than 1), then our estimator is underconfident. Thus, an ideal estimator will have a value close to 1. A widely-used 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.¹⁶ 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 with the correct model by using robust estimates of the center.

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

¹⁶For an unbiased estimator $\hat{\beta}$, this quantity would be the inverse of a Monte Carlo estimate of the efficiency, $\mathcal{I}^{-1}(\beta)/\text{Var}(\hat{\beta})$, with \mathcal{I} denoting the Fisher information. Thus, our measure of overconfidence can loosely be seen as a robust estimator of the inverse efficiency provided an estimator is unbiased.

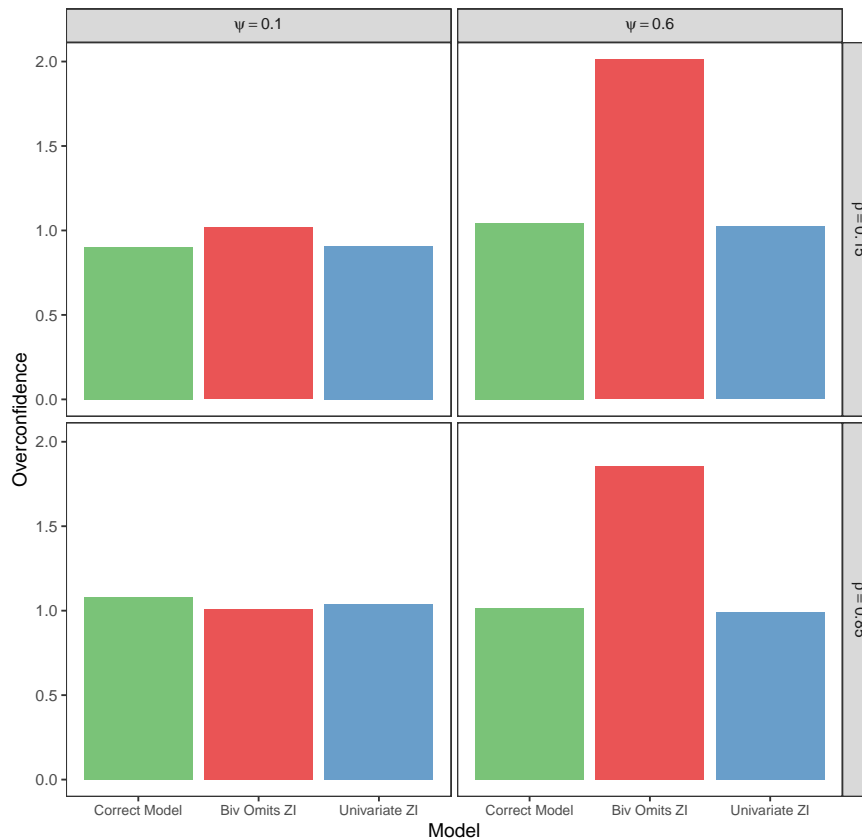


Figure 3: Overconfidence of centered count slope parameter estimate ($\beta_{1,2} = 3.25$) in margin 1, holding $\psi_2 = 0.1$ constant.

degree of zero-inflation. At low levels of zero-inflation (ex. $\psi = 0.1$), however, the bivariate non-inflated model's overconfidence does not differ considerably from the zero-inflated bivariate model. Further investigation would be required to determine at what exact levels of zero-inflation these differences become appreciable. The results here suffice to indicate that with observational data the conservative strategy would be to account for zero inflation when suspected. Somewhat surprisingly, the univariate ZIP model – which does not account for model dependence – does not underestimate its standard errors. However, the univariate ZIP model estimator is less efficient (e.g., greater standard errors) relative to the true zero-inflated bivariate model under high levels of dependence (see the bottom-left panel of Figure 2 above).

7.4. Bias of dependence parameter estimators

Finally, users may also be interested in the performance of the dependence parameter estimators. The density plots in Figure 4 are the truth-centered estimates of the dependence parameter, again holding the zero-inflation parameter for margin 2 fixed at $\psi_2 = 0.1$.¹⁷ In short, there are two main conclusions:

¹⁷We do not report the results from the univariate models since these constrain ρ to zero by assumption.

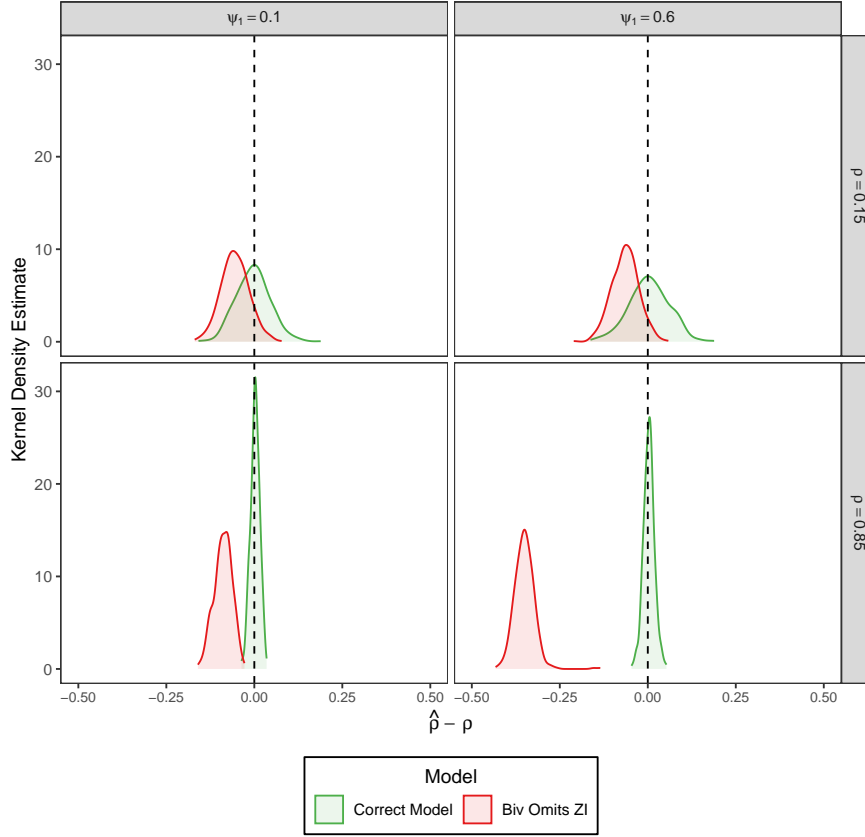


Figure 4: Density of centered dependence parameter estimate with $\psi_2 = 0.1$.

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.

8. Conclusion and future directions

As we demonstrate above, 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 challenges simultaneously researchers have far fewer options.

Currently, researchers often address only one of these problems at a time, risking either inefficiency or bias. Even where implementations of zero-inflated bivariate count models are found, these require users to specify a particular multivariate zero-inflated count distribution, regardless of whether the assumptions for such a specification are not well supported. Here we have discussed how copulas can be used as an alternative that only requires assumptions about the *marginal* distributions and the form of the dependence, which are often better supported theoretically. Using this strategy it is also 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; Rüschenendorf 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. Mikiyoung 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.

References

- Akaike H (1973). “Information Theory and an Extension of the Maximum Likelihood Principle.” In *Selected Papers of Hirotugu Akaike*, pp. 199–213. Springer-Verlag.
- Arab A, Holan SH, Wikle CK, Wildhaber ML (2012). “Semiparametric Bivariate Zero-Inflated Poisson Models with Application to Studies of Abundance for Multiple Species.” *Environmetrics*, **23**(2), 183–196. doi:10.1002/env.1142.
- Blyth S, Groombridge B, Lysenko I, Miles L, Newton A (2002). *Mountain Watch*. UNEP World Conservation Monitoring Centre, Cambridge.
- Cameron AC, Li T, Trivedi PK, Zimmer DM (2004). “Modelling the Differences in Counted Outcomes Using Bivariate Copula Models with Application to Mismeasured Counts.” *The Econometrics Journal*, **7**(2), 566–584. doi:10.1111/j.1368-423x.2004.00144.x.
- Cameron AC, Trivedi PK (2013). *Regression Analysis of Count Data*, volume 53. Cambridge University Press.

- Center for International Earth Science Information Network (CIESIN) & Centro Internacional (2005). “Gridded Population of the World, Version 3 (GPWv3): Population Count Grid.”
- Chen Y, Hanson T (2017). “Copula Regression Models for Discrete and Mixed Bivariate Responses.” *Journal of Statistical Theory and Practice*, **11**(4), 515–530. doi:10.1080/15598608.2016.1278059.
- Chou NT, Steenhard D (2011). “Bivariate Count Data Regression Models – A SAS Macro Program.” *SAS Global Forum: Statistics and Data Analysis*, **355**.
- Chowdhury RI, Islam MA (2019). *bpglm: Bivariate Poisson Generalized Linear Models with Covariate Dependence*. R package version 0.1.0, URL <https://github.com/chowdhuryri/bpglm>.
- Desmarais BA, Harden JJ (2013). “Testing for Zero Inflation in Count Models: Bias Correction for the Vuong Test.” *The Stata Journal*, **13**(4), 810–835. doi:10.1177/1536867x1301300408.
- Dunn PK, Smyth GK (1996). “Randomized Quantile Residuals.” *Journal of Computational and Graphical Statistics*, **5**(3), 236–244. doi:10.1080/10618600.1996.10474708.
- Eager CD (2021). *standardize: Tools for Standardizing Variables for Regression in R*. R package version 0.2.2, URL <https://CRAN.R-project.org/package=standardize>.
- Famoye F (2010a). “A New Bivariate Generalized Poisson Distribution.” *Statistica Neerlandica*, **64**(1), 112–124. doi:10.1111/j.1467-9574.2009.00446.x.
- Famoye F (2010b). “On the Bivariate Negative Binomial Regression Model.” *Journal of Applied Statistics*, **37**(6), 969–981. doi:10.1080/02664760902984618.
- Faroughi P, Ismail N (2017a). “Bivariate Zero-Inflated Generalized Poisson Regression Model with Flexible Covariance.” *Communications in Statistics – Theory and Methods*, **46**(15), 7769–7785. doi:10.1080/03610926.2016.1165846.
- Faroughi P, Ismail N (2017b). “Bivariate Zero-Inflated Negative Binomial Regression Model with Applications.” *Journal of Statistical Computation and Simulation*, **87**(3), 457–477. doi:10.1080/00949655.2016.1213843.
- Fréchet M (1951). “Sur les Tableaux de Corrélacion dont les Marges Sont Données.” *Annales de l’Université de Lyon, III. Série, Section A*, **14**, 53–77.
- Gelman A (2008). “Scaling Regression Inputs by Dividing by Two Standard Deviations.” *Statistics in Medicine*, **27**(15), 2865–2873. doi:10.1002/sim.3107.
- Genest C, Nešlehová J (2007). “A Primer on Copulas for Count Data.” *ASTIN Bulletin: The Journal of the IAA*, **37**(2), 475–515. doi:10.1017/s0515036100014963.
- Gilbert P, Varadhan R (2019). “**numDeriv**: Accurate Numerical Derivatives.” R package version 2016.8-1.1, URL <https://CRAN.R-project.org/package=numDeriv>.
- Goren E, Hughes J (2021). *copCAR: Fitting the copCAR Regression Model for Discrete Areal Data*. R package version 2.0-4, URL <https://CRAN.R-project.org/package=copCAR>.

- Greene WH (2003). *Econometric Analysis*. 5th edition. Prentice Hall, Upper Saddle River.
- Gurmu S, Elder J (2008). “A Bivariate Zero-Inflated Count Data Regression Model with Unrestricted Correlation.” *Economics Letters*, **100**(2), 245–248. doi:[10.1016/j.econlet.2008.02.001](https://doi.org/10.1016/j.econlet.2008.02.001).
- Hartig F (2022). **DHARMA**: *Residual Diagnostics for Hierarchical (Multi-Level / Mixed) Regression Models*. R package version 0.4.6, URL <https://CRAN.R-project.org/package=DHARMA>.
- He H, Zhang H, Ye P, Tang W (2019). “A Test of Inflated Zeros for Poisson Regression Models.” *Statistical Methods in Medical Research*, **28**(4), 1157–1169. doi:[10.1177/0962280217749991](https://doi.org/10.1177/0962280217749991).
- Hoeffding W (1940). “Maßstabinvariante Korrelationstheorie.” In PK Sen, N Fisher (eds.), *The Collected Works of Wassily Hoeffding*, pp. 57–107. Springer-Verlag.
- Hoeffding W (1941). “Maßstabinvariante Korrelationsmaße für diskontinuierliche Verteilungen.” In PK Sen, N Fisher (eds.), *The Collected Works of Wassily Hoeffding*, pp. 49–70. Springer-Verlag.
- Holgate P (1964). “Estimation for the Bivariate Poisson Distribution.” *Biometrika*, **51**(1-2), 241–287. doi:[10.1093/biomet/51.1-2.241](https://doi.org/10.1093/biomet/51.1-2.241).
- Hughes J (2015). “copCAR: A Flexible Regression Model for Areal Data.” *Journal of Computational and Graphical Statistics*, **24**(3), 733–755. doi:[10.1080/10618600.2014.948178](https://doi.org/10.1080/10618600.2014.948178).
- Inouye DI, Yang E, Allen GI, Ravikumar P (2017). “A Review of Multivariate Distributions for Count Data Derived from the Poisson Distribution.” *Wiley Interdisciplinary Reviews: Computational Statistics*, **9**(3). doi:[10.1002/wics.1398](https://doi.org/10.1002/wics.1398).
- Joe H (1997). *Multivariate Models and Multivariate Dependence Concepts*. Chapman & Hall/CRC. doi:[10.1201/b13150](https://doi.org/10.1201/b13150).
- Karlis D (2016). “Models for Multivariate Count Time Series.” In RA Davis, SH Holan, R Lund, N Ravishanker (eds.), *Handbook of Discrete-Valued Time Series*, chapter 19, pp. 407–424. Chapman & Hall/CRC.
- Karlis D, Ntzoufras I (2005). “Bivariate Poisson and Diagonal Inflated Bivariate Poisson Regression Models in R.” *Journal of Statistical Software*, **14**(10), 1–36. doi:[10.18637/jss.v014.i10](https://doi.org/10.18637/jss.v014.i10).
- Kazianka H (2013). “Approximate Copula-Based Estimation and Prediction of Discrete Spatial Data.” *Stochastic Environmental Research and Risk Assessment*, **27**(8), 2015–2026. doi:[10.1007/s00477-013-0737-7](https://doi.org/10.1007/s00477-013-0737-7).
- Kazianka H, Pilz J (2010). “Copula-Based Geostatistical Modeling of Continuous and Discrete Data Including Covariates.” *Stochastic Environmental Research and Risk Assessment*, **24**(5), 661–673. doi:[10.1007/s00477-009-0353-8](https://doi.org/10.1007/s00477-009-0353-8).
- Kocherlakota S, Kocherlakota K (1992). *Bivariate Discrete Distributions*. Routledge. doi:[10.1201/9781315138480](https://doi.org/10.1201/9781315138480).

- Krämer N, Silvestrini D (2014). **CopulaRegression**: *Bivariate Copula-Based Regression Models*. R package version 0.1-5, URL <https://CRAN.R-project.org/package=CopulaRegression>.
- LaFree G, Dugan L (2007). “Introducing the Global Terrorism Database.” *Terrorism and Political Violence*, **19**(2), 181–204. doi:10.1080/09546550701246817.
- Lai C (1995). “Construction of Bivariate Distributions by a Generalised Trivariate Reduction Technique.” *Statistics & Probability Letters*, **25**(3), 265–270. doi:10.1016/0167-7152(94)00230-6.
- Lakshminarayana J, Pandit SNN, Srinivasa Rao K (1999). “On a Bivariate Poisson Distribution.” *Communications in Statistics – Theory and Methods*, **28**(2), 267–276. doi:10.1080/03610929908832297.
- Lambert D (1992). “Zero-Inflated Poisson Regression, with an Application to Defects in Manufacturing.” *Technometrics*, **34**(1), 1–14. doi:10.2307/1269547.
- Leifeld P (2013). “**texreg**: Conversion of Statistical Model Output in R to L^AT_EX and HTML Tables.” *Journal of Statistical Software*, **55**(8), 1–24. doi:10.18637/jss.v055.i08.
- Li CS, Lu JC, Park J, Kim K, Brinkley PA, Peterson JP (1999). “Multivariate Zero-Inflated Poisson Models and Their Applications.” *Technometrics*, **41**(1), 29–38. doi:10.2307/1270992.
- Marra G, Radice R (2017). “A Joint Regression Modeling Framework for Analyzing Bivariate Binary Data in R.” *Dependence Modeling*, **5**(1), 268–294. doi:10.1515/demo-2017-0016.
- Marshall AW, Olkin I (1985). “A Family of Bivariate Distributions Generated by the Bivariate Bernoulli Distribution.” *Journal of the American Statistical Association*, **80**(390), 332–338. doi:10.1080/01621459.1985.10478116.
- Marshall AW, Olkin I (1990). “Multivariate Distributions Generated from Mixtures of Convolution and Product Families.” In HW Block, AR Sampson, TH Savits (eds.), *IMS Lecture Notes – Monograph Series*, volume 16, pp. 371–393. Institute of Mathematical Statistics. doi:10.1214/lnms/1215457574.
- Masarotto G, Varin C (2017). “Gaussian Copula Regression in R.” *Journal of Statistical Software*, **77**(8), 1–26. doi:10.18637/jss.v077.i08.
- Mullahy J (1986). “Specification and Testing of Some Modified Count Data Models.” *Journal of Econometrics*, **33**(3), 341–365. doi:10.1016/0304-4076(86)90002-3.
- Nelsen RB (2007). *An Introduction to Copulas*. Springer-Verlag.
- Nikoloulopoulos A (2013). “Copula-Based Models for Multivariate Discrete Response Data.” In P Jaworski, F Durante, WK Härdle (eds.), *Copulae in Mathematical and Quantitative Finance*, chapter 11, pp. 231–250. Springer-Verlag.
- Panagiotelis A, Czado C, Joe H (2012). “Pair Copula Constructions for Multivariate Discrete Data.” *Journal of the American Statistical Association*, **107**(499), 1063–1072. doi:10.1080/01621459.2012.682850.

- Piazza JA (2017). “Repression and Terrorism: A Cross-National Empirical Analysis of Types of Repression and Domestic Terrorism.” *Terrorism and Political Violence*, **29**(1), 102–118. doi:10.1080/09546553.2014.994061.
- Rüschendorf L (2013). “Copulas, Sklar’s Theorem, and Distributional Transform.” In *Mathematical Risk Analysis*, pp. 3–34. Springer-Verlag.
- Sarmanov OV (1966). “Generalized Normal Correlation and Two-Dimensional Fréchet Classes.” In *Doklady Akademii Nauk*, volume 168, pp. 32–35. Russian Academy of Sciences.
- SAS Institute Inc (2020). “Fitting Zero-Inflated Count Data Models by Using PROC GENMOD.” URL <https://support.sas.com/rnd/app/stat/examples/GENMODZIP/roots.pdf>.
- Savun B, Tirone DC (2018). “Foreign Aid as a Counterterrorism Tool: More Liberty, Less Terror?” *Journal of Conflict Resolution*, **62**(8), 1607–1635. doi:10.1177/0022002717704952.
- Schwarz G (1978). “Estimating the Dimension of a Model.” *The Annals of Statistics*, **6**(2), 461–464. doi:10.1214/aos/1176344136.
- Seabold S, Perktold J (2010). “statsmodels: Econometric and Statistical Modeling with Python.” In *Proceedings of the 9th Python in Science Conference*, volume 57, p. 61. Austin.
- Sklar A (1973). “Random Variables, Joint Distribution Functions, and Copulas.” *Kybernetika*, **9**(6), 449–460.
- Sklar M (1959). “Fonctions de Repartition à n Dimensions et Leurs Marges.” *Publications de l’Institut de Statistique de l’Université de Paris*, **8**, 229–231.
- So S, Lee DH, Jung BC (2011). “An Alternative Bivariate Zero-Inflated Negative Binomial Regression Model Using a Copula.” *Economics Letters*, **113**(2), 183–185. doi:10.1016/j.econlet.2011.07.017.
- START (2018). “Global Terrorism Database [Data File].” Retrieved from <https://www.start.umd.edu/gtd>.
- StataCorp (2021). *Stata Statistical Software: Release 17*. StataCorp LLC, College Station. URL <https://www.stata.com/>.
- Sun T, Ding Y (2020). “CopulaCenR: Copula Based Regression Models for Bivariate Censored Data in R.” *The R Journal*, **12**(1), 266–282. doi:10.32614/RJ-2020-025.
- Tang Y, Tang W (2019). “Testing Modified Zeros for Poisson Regression Models.” *Statistical Methods in Medical Research*, **28**(10-11), 3123–3141. doi:10.1177/0962280218796253.
- Ting Lee ML (1996). “Properties and Applications of the Sarmanov Family of Bivariate Distributions.” *Communications in Statistics – Theory and Methods*, **25**(6), 1207–1222. doi:10.1080/03610929608831759.
- Tollefsen AF, Bahgat K, Nordkvelle J, Buhaug H (2015). *PRIO-GRID v. 2.0 Codebook*.
- Tollefsen AF, Strand H, Buhaug H (2012). “PRIO-GRID: A Unified Spatial Data Structure.” *Journal of Peace Research*, **49**(2), 363–374. doi:10.1177/0022343311431287.

- Trivedi P, Zimmer D (2017). “A Note on Identification of Bivariate Copulas for Discrete Count Data.” *Econometrics*, **5**(1), 10. doi:10.3390/econometrics5010010.
- Trivedi PK, Zimmer DM (2007). *Copula Modeling: An Introduction for Practitioners*. Now Publishers.
- Van der Wurp H, Groll A, Kneib T, Marra G, Radice R (2019). “Generalised Joint Regression for Count Data with a Focus on Modelling Football Matches.” *arXiv 1908.00823*, arXiv.org E-Print Archive. doi:10.48550/arXiv.1908.00823.
- Vuong QH (1989). “Likelihood Ratio Tests for Model Selection and Non-Nested Hypotheses.” *Econometrica*, pp. 307–333. doi:10.2307/1912557.
- Wang K, Lee AH, Yau KKW, Carrivick PJW (2003). “A Bivariate Zero-Inflated Poisson Regression Model to Analyze Occupational Injuries.” *Accident Analysis & Prevention*, **35**(4), 625–629. doi:10.1016/s0001-4575(02)00036-2.
- Wang P (2003). “A Bivariate Zero-Inflated Negative Binomial Regression Model for Count Data with Excess Zeros.” *Economics Letters*, **78**(3), 373–378. doi:10.1016/s0165-1765(02)00262-8.
- Wilson MC, Piazza JA (2013). “Autocracies and Terrorism: Conditioning Effects of Authoritarian Regime Type on Terrorist Attacks.” *American Journal of Political Science*, **57**(4), 941–955. doi:10.1111/ajps.12028.
- Wilson P (2015). “The Misuse of the Vuong Test for Non-Nested Models to Test for Zero-Inflation.” *Economics Letters*, **127**, 51–53. doi:10.1016/j.econlet.2014.12.029.
- Xu X, Hardin JW (2016). “Regression Models for Bivariate Count Outcomes.” *The Stata Journal*, **16**(2), 301–315. doi:10.1177/1536867x1601600203.
- Yang L, Frees EW, Zhang Z (2020). “Nonparametric Estimation of Copula Regression Models with Discrete Outcomes.” *Journal of the American Statistical Association*, **115**(530), 707–720. doi:10.1080/01621459.2018.1546586.
- Zeileis A, Croissant Y (2010). “Extended Model Formulas in R: Multiple Parts and Multiple Responses.” *Journal of Statistical Software*, **34**(1), 1–13. doi:10.18637/jss.v034.i01.
- Zeileis A, Kleiber C, Jackman S (2008). “Regression Models for Count Data in R.” *Journal of Statistical Software*, **27**(8), 1–25. doi:10.18637/jss.v027.i08.

A. Further computational details

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

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

where we require that $\text{frech.min} \in (0, 0.00001]$, 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 $1e-7$.

B. Residual plots using 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.

```
R> name <- c("fulani", "bh")
R> for(i in seq_along(name)) {
+   pdf(paste0("Figures/appendix_figure_B", i, ".pdf"),
+       width = 8, height = 4.5)
+   plot(zip_dharmas[[i]])
+   dev.off()
+ }
```

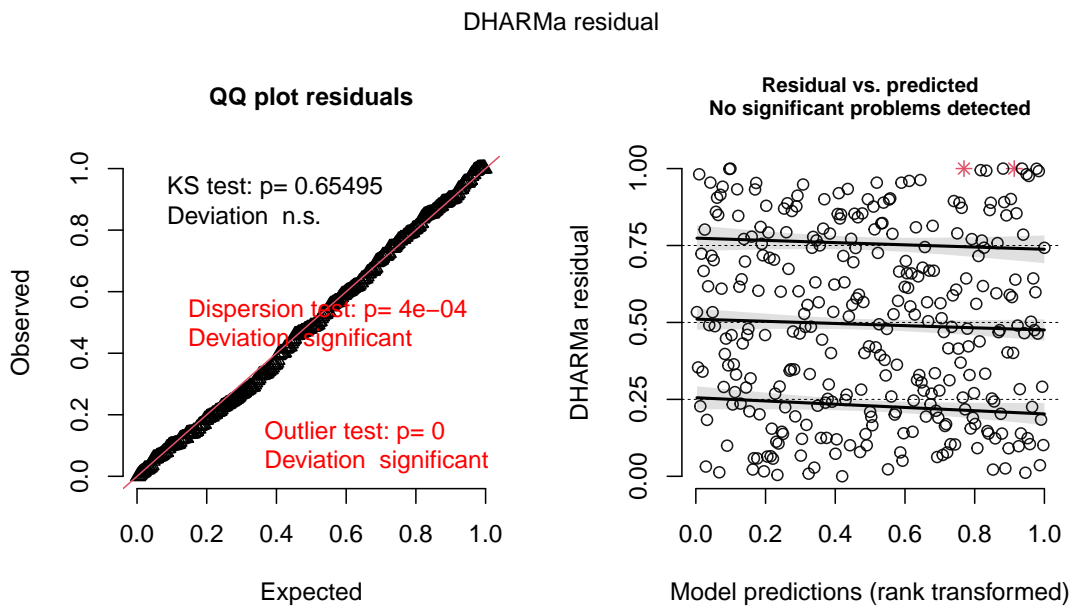


Figure 5: Zero-inflated Poisson, Boko Haram.

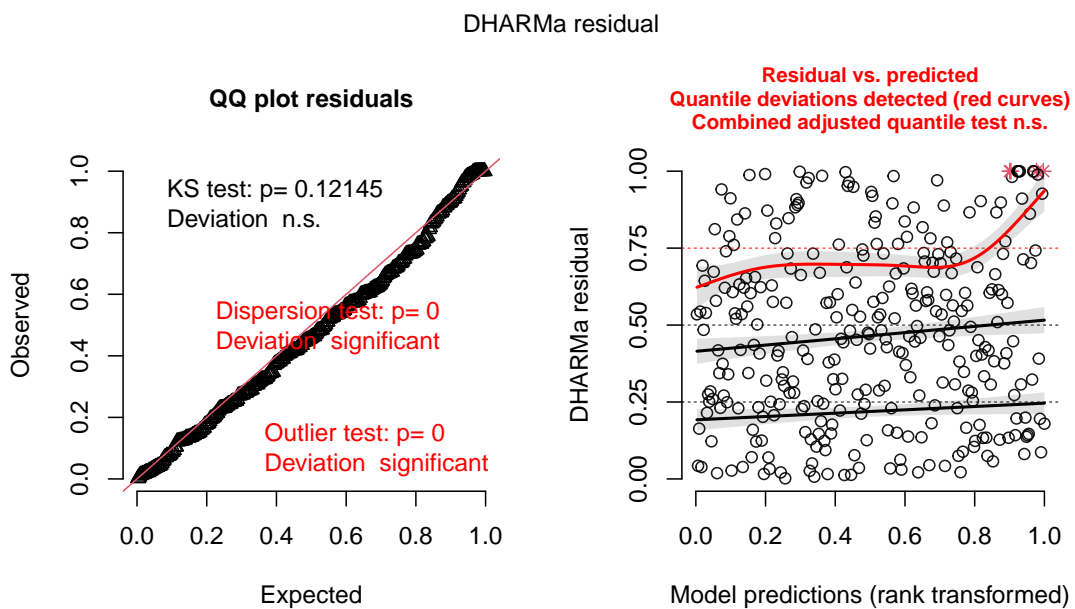


Figure 6: Zero-inflated Poisson, Fulani.

C. Univariate functions

C.1. Random number generation: `rzip()`, `rzinb()`

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

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

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 desirable 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 conformity. 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.

C.2. CDF evaluation: `pzip()`, `pzinb()`

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

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

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 probabilities ought to be returned on the (natural) log scale.

C.3. PMF evaluation: `dzip()`, `dzinb()`

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

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

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

C.4. Quantile function evaluation: `qzip()`, `qzinb()`

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

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

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.

C.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 = TRUE, keep = FALSE, ...)
```

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 "sqrt", "identity", "log".
- `link.zi` determines the link function for the zero-inflation portion, and is one of "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 assumed 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.

```
R> library("DHARMa")
R> library("texreg")
R> unizi.bok <- zic.reg(fmla = fmla.zi.bok, data = dat, keep = TRUE)
R> uni.dharma <- createDHARMa(simulatedResponse = simulate(unizi.bok),
+   observedResponse = dat$att.bok,
+   fittedPredictedResponse = fitted(unizi.bok),
+   integerResponse = TRUE, seed = 123, method = "PIT")
R> plot(uni.dharma)
R> texreg(unizi.bok)
```

Affiliation:

John M. Niehaus

Department of Statistics

Texas A&M University

3143 TAMU, College Station, TX 77843-3143, United States of America

E-mail: jniehaus2257@gmail.com