Journal of Statistical Software

July 2019, Volume 90, Issue 8. doi: 10.18637/jss.v090.i08

Mean and Variance Modeling of Under-Dispersed and Over-Dispersed Grouped Binary Data

David M. Smith
IBM Watson Health

Malcolm J. Faddy
Queensland University of Technology

Abstract

This article describes the R package **BinaryEPPM** and its use in determining maximum likelihood estimates of the parameters of extended Poisson process models for grouped binary data. These provide a Poisson process family of flexible models that can handle unlimited under-dispersion but limited over-dispersion in such data, with the binomial distribution being a special case. Within **BinaryEPPM**, models with the mean and variance related to covariates are constructed to match a generalized linear model formulation. Combining such under-dispersed models with standard over-dispersed models such as the beta binomial distribution provides a very general form of residual distribution for modeling grouped binary data. Use of the package is illustrated by application to several data-sets.

Keywords: binomial distribution, covariate effects, dispersion, Poisson process, precision of estimates.

1. Introduction

Modeling using extended Poisson process models (EPPMs) was originally developed in Faddy (1997) where the construction of discrete probability distributions having very general dispersion properties was described. Smith and Faddy (2016) was concerned with generalizations of the Poisson distribution to deal with over- and under-dispersion. This article is about similar generalizations of the binomial distribution which is another special case of the modeling described in Faddy (1997). Covariate dependence can be incorporated via a re-parameterization using approximate forms of the mean and variance.

The supplementary material for Faddy and Smith (2012) contained R (R Core Team 2019) code illustrating the fitting of these models. This code has been extended and generalized to have inputs and outputs akin to those of the generalized linear model function `glm` from the

packages **stats** and **betareg** (Cribari-Neto and Zeileis 2010; Grün, Kosmidis, and Zeileis 2012). The resulting package **BinaryEPPM** (Smith and Faddy 2019), whose use is described here, is available as a contributed package from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/package=BinaryEPPM.

There exists a number of generalized binomial models that can deal with over-dispersion relative to the binomial: for example, mixed models (Williams 1996) and correlated models (Kupper and Haseman 1978). Although the resulting probability distributions can admit some under-dispersion, where the residual variance is less than that corresponding to a binomial distribution, this may be rather too limited for them to be considered general models for under-dispersed data. **BinaryEPPM** complements these models by modeling under-dispersion (and limited over-dispersion). Both the mean and variance can be formulated in terms of associated covariates. Observed data can then be modeled using these generalized binomial distributions, leading to better fitting models and model checking diagnostics, and more appropriate assessment of the precision of any estimated quantities.

2. Models

2.1. Extended Poisson process models (EPPMs)

The models described in Faddy (1997) can be summarized as describing probability distributions on 0, 1, 2, ..., n in terms of the vector of probabilities

$$\boldsymbol{p} = (1\ 0\ \cdots\ 0)\exp(\boldsymbol{Q}), \tag{1}$$

where $\boldsymbol{Q}$ is an $(n+1)\times(n+1)$ bi-diagonal matrix consisting of (Poisson process) rate parameters $\lambda_i(>0)$ for $i = 0, 1, \ldots, n-1$ on the upper diagonal; and $-\lambda_i$ for $i = 0, 1, \ldots, n$ (with $\lambda_n = 0$) on the diagonal. A function of linearly decreasing λ_i's

$$\lambda_i = a(n - i), \text{ for } i = 0, 1, 2, \ldots, n \text{ with } a > 0 \tag{2}$$

gives rise to the binomial distribution with probability $p = 1 - \exp(-a)$. If covariates, $\boldsymbol{x}$ say, influence the response then having $\log(a) = \boldsymbol{x}^T\boldsymbol{\beta}$ (the usual linear predictor) in this binomial special case corresponds to generalized linear modeling with a complementary log-log link function (Dobson and Barnett 2008, Chapter 7). Other link functions (such as logistic) could be used, but the complementary log-log link function does arise quite naturally from this extended Poisson process modeling.

Faddy and Smith (2008) considered a generalization of Equation 2

$$\lambda_i = a(n - i)^b, \text{ with } b > 0 \tag{3}$$

resulting in distributions analogous to those from correlated binomial modeling (Kupper and Haseman 1978) with concave sequences of λ_i's ($0 < b < 1$) corresponding to positive correlations and over-dispersion, and convex sequences ($b > 1$) to negative correlations and under-dispersion. Here approximations for the mean and variance of these distributions from Faddy (1997) are used to re-parameterize them in terms of the probability of a success p_s in a single Bernoulli trial and scale-factor f_s for the variance of the number of successes in n

trials as in Equations 4 and 5.

$$p_s = \frac{\text{mean}}{n} \approx 1 - \left\{1 - an^{b-1}(1 - b)\right\}^{\frac{1}{1-b}} \tag{4}$$

$$\text{and } f_s = \frac{\text{variance}}{np_s(1 - p_s)} \approx \frac{(1 - p_s)^{2b-1} - 1}{p_s(1 - 2b)} \tag{5}$$

with substantial under-dispersion possible for large b ($f_s \to 0$ as $b \to \infty$) while over-dispersion is limited by $f_s < \frac{1}{1-p_s}$ (the value for $b \to 0$). Since the complementary probability distribution of the number of failures will have (approximately) $f_s < \frac{1}{p_s}$ over-dispersion is effectively limited by $f_s < \max\left(\frac{1}{1-p_s}, \frac{1}{p_s}\right)$ with this modeling. Although technically the scale-factor cannot exceed n, this is unlikely to be a practical limitation, so a simple log link can be used for covariate dependence; i.e., $\log(f_s) = \boldsymbol{x}^\top \boldsymbol{\gamma}$.

Given f_s and p_s Equation 5 can be solved for b using the R root finding function `uniroot`, then Equation 4 can be solved for a leading to

$$\lambda_i = n\left[\frac{1 - (1 - p_s)^{1-b}}{(1 - b)}\right]\left(1 - \frac{i}{n}\right)^b \tag{6}$$

from Equation 3. This parameterization based on approximate forms for the mean and variance results in the exact mean and scale-factor not being described perfectly by their respective link functions of the linear predictors but by some perturbations of these. However, for the examples discussed in the next section the effect of this on moment-based estimates is quite modest. The covariate coefficients $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ describing the mean and scale-factor can be estimated by maximum likelihood from data $y_1, y_2, \ldots, y_k$ using the likelihood $p_{y_1}, p_{y_2}, \ldots, p_{y_k}$ from the probabilities in Equation 1. Alternatively, the parameter b in Equation 6 can be estimated as a nuisance parameter if there is no interest in modeling the variance. Exact calculation of the mean and variance can also be done using the probabilities in Equation 1.

2.2. Models other than EPPMs

There are other distributional models available for over-dispersed binary data such as the correlated binomial and beta binomial distributions. These distributions differ in their interpretation with the former allowing the outcomes of successive trials to be correlated, and the latter being a mixed binomial distribution where the success probability p_s is not fixed over the sequence of trials but varies according to a beta distribution. The mean and scale-factor of a simple correlated binomial with correlation ρ between the outcomes of any two trials are np_s and $1 + \rho(n-1)$, with probability mass function as in Kupper and Haseman (1978)

$$\mathsf{P}(X = x) = \binom{n}{x} p_s^x (1 - p_s)^{n-x} \left\{1 + \frac{\rho}{2p_s(1 - p_s)}\left[(x - np_s)^2 + x(2p_s - 1) - np_s^2\right]\right\}.$$

The beta binomial distribution has probability mass function as in Smith (1983)

$$\mathsf{P}(X = x) = \binom{n}{x} \frac{\prod_{r=0}^{x-1}(\mu + r\theta) \prod_{r=0}^{n-x-1}(1 - \mu + r\theta)}{\prod_{r=0}^{n-1}(1 + r\theta)},$$

with mean μ and scale-factor $1 + \frac{\theta}{(1+\theta)}(n-1)$ (Hughes and Madden 1995). Both these distributions do admit some modest levels of under-dispersion; bounds on the scale-factor can be determined from those given for ρ in Kupper and Haseman (1978) for the correlated binomial, and for θ in Prentice (1986) for the beta binomial.

The EPPM generalization of the binomial distribution complements the beta binomial distribution as it allows quite general levels of under-dispersion but only modest levels of over-dispersion. Therefore a distribution formed by a combination of beta binomial for $f_s > 1$ and EPPM generalized binomial for $f_s \leq 1$ will allow for the full range of under- and over-dispersion in observed data. With the mean and scale-factor being dependent on covariates as discussed in the previous sub-section, continuity is assured by both the EPPM generalized binomial and the beta binomial reducing to the simple binomial distribution for $f_s = 1$. Standard likelihood methods would apply as $f_s = 1$ is not on the boundary of the parameter spaces of either of the components forming the residual distribution.

3. Description of the functions

Models with two covariate dependencies linked to p_s and f_s are developed using Equations 1 and 3. The link function between p_s and the linear predictor of covariates is either logit, probit, complementary log-log, cauchit, log, loglog, double exponential, double reciprocal, power logit, or negative complementary log. The last four of these link functions are not available in `glm` or **betareg**. References to them are Ford, Torsney, and Wu (1992), Gaudard, Karson, Linder, and Tse (1993), and Tibshirani and Ciampi (1983). Only a log link function is used for the scale-factor f_s. Fitting to data is done using maximum likelihood, the optimization method used being one of two of the options available in the R function `optim`, i.e., the simplex method of Nelder and Mead (1967) (`"Nelder-Mead"`), or the `"BFGS"` method which uses first derivatives. The first derivatives used in the latter method, and in calculating the hessian matrix, are numerical ones obtained using the `gradient` function of the R package **numderiv** of Gilbert and Varadhan (2019).

The R package **Formula** of Zeileis and Croissant (2010) is used to extract model information from the `formula` input to `BinaryEPPM`. Offsets are included in the formulae specifications. To avoid repeated extractions within subsidiary functions, extraction of model information such as `covariates.matrix.mean` is only done once. As iteration is involved in the model fitting, initial estimates of the parameters are needed. These can be provided in the vector `initial` with a default, if unset, of initial estimates being produced within `BinaryEPPM` by fitting a binomial model using `glm`. The matrix exponential function used for calculating the probabilities of Equation 1 is from the package **expm** of Goulet, Dutang, Maechler, Firth, Shapira, and Stadelmann (2019) which depends on the package **Matrix** of Bates and Maechler (2019). Three pseudo R-squared are available, the first, is the square of the correlation between the observed and predicted GLM linear predictor values; the other two are commonly used in logistic regression, relevant references being Cox and Snell (1989) and Nagelkerke (1991).

The arguments of `BinaryEPPM` are

```
BinaryEPPM(formula, data, subset = NULL, na.action = NULL,
  weights = NULL, model.type = "p and scale-factor",
  model.name = "generalized binomial", link = "cloglog",
  initial = NULL, method = "Nelder-Mead",
  pseudo.r.squared = "square of correlation", control = NULL)
```

Argument	Description	Default
`formula`	paired formulae as in Zeileis and Croissant (2010)	
`data`	a `data.frame` or a `list`	
`subset`	subsetting commands	`NULL`
`na.action`	action taken for NAs in `data`	`NULL`
`weights`	vector if `data` is a `data.frame`	vector of ones
	a `list` if `data` is a `list`	list of lists of ones
	attributes `normalization`, `norm.to.n`	both `NULL`
`model.type`	`"p only"`	`"p and scale-factor"`
	(only p_s in Equation 4 modeled)	
	`"p and scale-factor"`	
	(p_s and f_s modeled)	
`model.name`	`"binomial"` (`"p only"`)	`"generalized binomial"`
	`"beta binomial"`	
	`"correlated binomial"`	
	`"generalized binomial"`	
`link`	the GLM link function for p_s	`"cloglog"`
	`"logit"` `"probit"` `"cloglog"`	
	`"cauchit"` `"log"` `"loglog"`	
	`"doubexp"` `"doubrecip"`	
	`"negcomplog"`	
	`"powerlogit"` attribute `"power"`	`"power"` = 1
`initial`	parameter initial values vector	`glm` fit of binomial
`method`	`"Nelder-Mead"`	`"Nelder-Mead"`
	`"BFGS"` attribute `"grad.method"`	attribute `"simple"`
	which is `"simple"` or `"Richardson"`	
`pseudo.r.squared`	`"square of correlation"`	`"square of correlation"`
	`"R squared"`	
	`"max-rescaled R squared"`	
`control`	list of control parameters	see text for more detail

Table 1: Arguments of `BinaryEPPM`.

with details given in Table 1 together with defaults if any. The dependent variable is either a column, or columns, where `data` is a `data.frame`; or a `list` within `data` where it is a `list`. For the latter, the response variable `list` is one of frequency distributions. Several of the example data sets are available in both forms to illustrate how to deploy them. Table 2 gives details of the fitted model object of class '`BinaryEPPM`' returned. It is a list similar to those of objects with classes '`glm`' and '`betareg`' returned by calls to `glm` and **betareg**. Table 3 gives details of a set of S3 generic extractor functions for objects of class '`BinaryEPPM`'. The set is similar to that of Table 1 of Cribari-Neto and Zeileis (2010) related to package **betareg**, except there are no functions `estfun`, `bread` or `linear.hypothesis`. Also, `gleverage` and `cooks.distance` are variants of the functions `glm.diag` and `glm.diag.plots` from package **boot** (Canty and Ripley 2019) rather than **betareg**. The first four blocks refer to functions specific to **BinaryEPPM**. The last block contains generic functions, the default versions of

Component	Description
`data.type`	`data.frame` or `list`
`list.data`	data as a `list` of frequency distributions
`call`	the call to `BinaryEPPM`
`formula`	the `formula` input
`model.type`	`"p only"` or `"p and scale-factor"`
`model.name`	as in Table 1 according to value of `model.type`
`link`	the GLM link function for p_s
`covariates.matrix.p`	matrix of covariates for p_s
`covariates.matrix.scalef`	matrix of covariates for scale-factor
`offset.p`	offset vector for p_s
`offset.scalef`	offset vector for scale-factor
`coefficients`	the estimated coefficients
`loglik`	the final log likelihood
`vcov`	the estimated variance/covariance matrix
`n` needed for `lmtest`	the number of observations
`nobs` needed for `stats`	the number of observations
`df.null`	null model degrees of freedom
`df.residual`	residual degrees of freedom
`vnmax`	a vector of number of trials
`weights`	a vector of weights
`converged`	whether converged
`iterations`	number of iterations
`method`	`"Nelder-Mead"` or `"BFGS"`
`pseudo.r.squared`	pseudo R squared value
`start`	initial estimates input
`optim`	final estimates of coefficients
`control`	control parameters of `optim`
`fitted.values`	fitted values of p_s
`y`	observed values of p_s
`terms`	model terms

Table 2: Components of object returned by `BinaryEPPM`.

which work because of the information supplied by the functions of the first four blocks. Package **lmtest** (Zeileis and Hothorn 2002) needs to be loaded to use `coeftest` and `lrtest`. Function `AIC` comes from **stats** which is a default package loaded when R is started. In Table 2 both `n` and `nobs` are included, so that functions from both packages **lmtest** and **stats** can use the object returned. The limits on the values of θ (beta binomial) or ρ (correlated binomial) can be obtained from the S3 extractor function `predict` with argument `type = "distribution.parameters"`. For given values of n and p_s tables of limits can be constructed using the subsidiary function `Model.BCBinProb` of **BinaryEPPM**. The supplementary file of examples has code for calculating the table of limits for ρ as given in Kupper and Haseman (1978).

Function	Description
`print()`	a simple printed display
`summary()`	standard regression output (coefficient estimates, standard errors, partial Wald tests); returns an object of class '`summary.BinaryEPPM`' containing the relevant summary statistics (which has a `print()` method)
`coef()`	extract coefficients of model (full, mean, or precision components), a single vector of all coefficients by default
`vcov()`	associated covariance matrix (with matching names)
`predict()`	predictions (response, linear predictor p_s, linear predictor scale-factor, p_s, scale-factor, scale-factor limits, mean, variance, distribution probabilities, distribution parameters) for existing and new data
`fitted()`	fitted means for observed data
`residuals()`	extract residuals (deviance, Pearson, response, standardized deviance, standardized Pearson residuals), defaulting to standardized Pearson residuals
`terms()`	extract terms of model components
`model.matrix()`	extract model matrix of model components
`model.frame()`	extract full original model frame
`logLik()`	extract fitted log-likelihood
`plot()`	diagnostic plots of residuals, predictions, leverages, etc.
`hatvalues()`	hat values (diagonal of hat matrix)
`cooks.distance()`	Cook's distance
`gleverage()`	generalized leverage
`waldtest()`	Wald tests of model parameters
`coeftest()`	partial Wald tests of coefficients
`lrtest()`	likelihood ratio tests of model parameters
`AIC()`	compute information criteria (AIC, BIC, ...)

Table 3: Generic functions for use with objects of class '`BinaryEPPM`'.

4. Examples

To run the examples as shown the package **lmtest** needs to be installed and loaded.

4.1. Data of number of rope spores in a dilution series of potato flour

These dilution series data originate from Finney (1971), where a number of samples ($n = 5$) at each of a series of dilutions of a suspension of potato flour were examined for rope spores. The data are given in Faddy and Smith (2008), Faddy and Smith (2012). Both forms of the data are available with `data("ropespores.grouped", package = "BinaryEPPM")` and `data("ropespores.case", package = "BinaryEPPM")` representing `list` and `data.frame` respectively. All models fitted have the (approximate) p_s modeled according to the series of dilutions using a `cloglog` link function

$$p_s = \frac{\text{mean}}{n} \approx (1 - \exp(-\exp(\beta_0 - \log(\text{dilution})))) .$$

The preliminary analysis of these data in Faddy and Smith (2008) was based on a binomial distribution from Equation 2 with $\log(a) = \beta_0 - \log(\text{dilution})$, corresponding to the parameter a being proportional to the reciprocal of the dilution, and log(dilution) an offset. Here, $1 - \exp(-a)$ is the probability of a single sample being fertile for rope spores and $\exp(-a)$ the probability of a single sample being sterile. Fitting a binomial followed by generalized binomial Equation 3 with constant b using the `data.frame` form of input

```
R> data("ropespores.case", package = "BinaryEPPM")
R> output.fn <- BinaryEPPM(number.spores / number.tested ~
+    1 + offset(logdilution), data = ropespores.case,
+    model.name = "binomial")
R> output.fn.one <- update(output.fn, model.type = "p only",
+    model.name = "generalized binomial")
R> summary(output.fn.one)

 Dependent variable a vector of numerator / denominator.

Call:
BinaryEPPM(formula = number.spores/number.tested ~ 1 + offset(logdilution),
  data = ropespores.case, model.type = "p only",
  model.name = "generalized binomial")

Model type        : p only
Model name        : generalized binomial
Link p            : cloglog
non zero offsets in linear predictors
Coefficients (model for p with cloglog  link):
Coefficient of GB parameter has 1 subtracted from it
so the test is against 1 i.e., a binomial.

t test of coefficients:

             Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.86624    0.14352 13.0036 1.16e-06 ***
GB parameter  8.49031    6.39009  1.3287   0.2206
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1' ' 1

 Type of estimator: ML (maximum likelihood)
 Log-likelihood: -3.244071 on 2 Df
 Pseudo R-squared: 0.892522 type square of correlation
 Number of iterations: 67 of optim method Nelder-Mead
 return code 0 successful
```

A likelihood ratio test can be performed and AIC values produced to compare the models.

```
R> lrtest(output.fn, output.fn.one)
R> AIC(output.fn, output.fn.one)
```

In the following, model 1 (`output.fn`) is a binomial and model 2 (`output.fn.one`) a generalized binomial.

```
Likelihood ratio test

Model 1: number.spores/number.tested ~ 1 + offset(logdilution)
Model 2: number.spores/number.tested ~ 1 + offset(logdilution)
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   1 -5.5942
2   2 -3.2441  1 4.7003    0.03016 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1' ' 1

              df      AIC
output.fn      1 13.18843
output.fn.one  2 10.48814
```

The generalized binomial model with constant b is superior to the binomial with significant under-dispersion apparent according to the likelihood ratio test, although not according to the Wald test due to considerable asymmetry in the profile log-likelihood as a function of this parameter. The estimates of the other parameter β_0 are rather different due to the formulation of the generalized binomial model in terms of the approximate mean (Equation 4), but this has only a small effect on the actual means of the fitted model.

The complementary log–log link function is asymmetric about the 50% (ED50) as compared to the symmetric logit link function. To assess how a more general asymmetric link function might perform, the profile likelihood can be optimized for a power logit link function.

```
R> output.fn.two <- update(output.fn.one, link = "powerlogit")
R> Results  <- optim(par = 1, fn = function(par, input.data, ...) {
+    local.link <- "powerlogit"
+    attr(local.link, which = "power") <- par
+    sum.logL <- logLik(update(output.fn.two, link = local.link))
+    return(sum.logL)}, input.data = ropespores.case,
+    method = "Brent", lower = 1/3, upper = 3,
+    control = list(fnscale = -1), hessian = TRUE)
R> se <- sqrt(-solve(Results$hessian)[1, 1])
R> data.frame(name = "power", Results$par, se, name = "log likelihood",
+    Results$value)
R> cat(paste("\n", "power", round(Results$par, digits = 4), "se",
+    round(sqrt(-solve(Results$hessian)[1, 1]), digits = 4),
+   "log likelihood", round(Results$value, digits = 4), "\n", sep = " "))

 power 2.5677 se 2.4504 log likelihood -2.5956
```

The difference in log-likelihoods here is insufficient for AIC to favor a model with a power logit link over one with the complementary log-log link.

4.2. Frequency of sex combinations in litters of pigs

The title of Brooks, James, and Gray (1991) suggests that the data they consider show underdispersion relative to the binomial distribution. Of the three data sets mentioned, only those for the Yorkshire breed will be used here. The fitting of a binomial distribution to these data with litter size treated as a factor with 9 levels suggests that such a model might be a satisfactory fit.

```
R> output.fn <- BinaryEPPM(data = Yorkshires.litters,
+    model.name = "binomial",  number.success ~ 0 + fsize)
R> cat(paste("\n", "generalized Pearson goodness of fit statistic",
+    round(sum(residuals(output.fn, type = "pearson")^2), digits = 4),
+    "on", sum(sapply(1:length(Yorkshires.litters$number.success),
+    function(i) { sum(c(Yorkshires.litters$number.success[[i]]))})) -
+    length(attr(Yorkshires.litters$fsize, which = "levels")), "df","\n",
+    sep = " "))

generalized Pearson goodness of fit statistic 2614.2181 on 2602 df
```

Fitting binomial and generalized binomial models with probability p_s dependent on litter size, the latter with a constant scale-factor f_s would support this. However, there is quite an improvement in fit by allowing the scale-factor f_s also to depend on litter size.

```
R> output.fn <- BinaryEPPM(data = Yorkshires.litters,
+    model.name = "binomial", number.success ~ 1 + vsize)
R> output.fn.one <- BinaryEPPM(data = Yorkshires.litters,
+    number.success ~ 1 + vsize | 1)
R> output.fn.two <- BinaryEPPM(data = Yorkshires.litters,
+    number.success ~ 1 + vsize | 1 + vsize)
R> lrtest(output.fn, output.fn.one, output.fn.two)

Model 1: number.success ~ 1 + vsize
Model 2: number.success ~ 1 + vsize | 1
Model 3: number.success ~ 1 + vsize | 1 + vsize
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   2 -4776.6
2   3 -4776.5  1 0.0726     0.7876
3   4 -4774.6  1 3.8115     0.0509 .
```

A `data.frame` of predicted summary statistics can be printed.

```
R> print(data.frame(size = Yorkshires.litters$vsize,
+    mean = predict(output.fn.two, type = "mean"),
+    variance = predict(output.fn.two, type = "variance"),
+    p = predict(output.fn.two, type = "p"),
+    scale.factor = predict(output.fn.two, type = "scale.factor"),
+    lower = predict(output.fn.two,
```

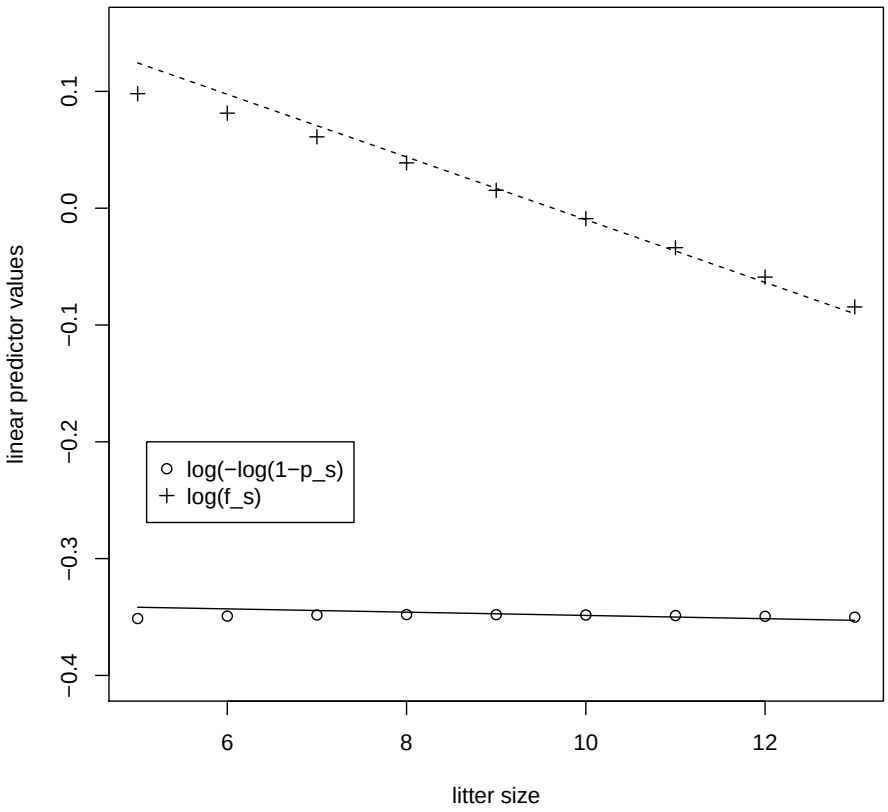

Figure 1: Linear predictor plots.

```
+      type = "scale.factor.limits")[["lower"]],
+    upper = predict(output.fn.two,
+      type = "scale.factor.limits")[["upper"]]),
+    row.names = FALSE)
```

size	mean	variance	p	scale.factor	lower	upper
5	2.526440	1.378628	0.5052879	1.1030256	0	2.035225
6	3.036085	1.626883	0.5060141	1.0847454	0	2.033201
7	3.544445	1.859951	0.5063493	1.0630006	0	2.031182
8	4.051727	2.078820	0.5064658	1.0395836	0	2.029168
9	4.558034	2.284360	0.5064482	1.0154399	0	2.027159
10	5.063419	2.477295	0.5063419	0.9910774	0	2.025154
11	5.567904	2.658235	0.5061731	0.9667782	0	2.023155
12	6.071499	2.827718	0.5059583	0.9427066	0	2.021160
13	6.574205	2.986238	0.5057081	0.9189622	0	2.019169

The calculation of these exact summary statistics is done using the probabilities in Equation 1 for the generalized binomial, rather than the approximate formulae in Equations 4 and 5. The following code uses `predict` to compare these approximate forms with the above predicted values of p_s and f_s. Figure 1 shows plots of these where the lines represent the approximate (linear) values and the symbols the exact (non linear) values.

```
R> approx.lp.p <- predict(output.fn.two, type = "linear.predictor.p")
R> approx.lp.sf <- predict(output.fn.two,
+    type = "linear.predictor.scale.factor")
R> exact.lp.p <- log( -log(1 - predict(output.fn.two, type = "p")))
R> exact.lp.sf <- log(predict(output.fn.two, type = "scale.factor"))
R> plot(x = c(5, 13), y = c(-0.40, 0.15), xlab = "litter size",
+    ylab = "linear predictor values", type = "n")
R> lines(x = Yorkshires.litters$vsize, y = approx.lp.p, lty = 1)
R> points(x = Yorkshires.litters$vsize, y = exact.lp.p, pch = 1)
R> lines(x = Yorkshires.litters$vsize, y = approx.lp.sf, lty = 2)
R> points(x = Yorkshires.litters$vsize, y = exact.lp.sf, pch = 3)
R> legend(5.1, -0.2, legend = c("log( -log(1 - p_s)", "log(f_s)"),
+    pch = c(1, 3), cex = 1.0)
```

At least for these data, the approximations are numerically close, but more importantly the exact values show only minor perturbations from linearity.

The data from the first five litters sizes show scale-factors greater than one and the data from the last four show scale-factors less than one. The following shows how a combined model with beta binomial for the over-dispersed litter sizes and generalized binomial for the under-dispersed litter sizes can be fitted.

```
R> in.par <- c(output.fn.two$coefficients$p.est,
+    output.fn.two$coefficients$scalef.est)
R> Results <- optim(par = in.par, fn = function(in.par, in.data,
+    model.names, subsets, ...) {
+    subset1 <- BinaryEPPM(data = in.data, model.name = model.names[1],
+    subset = subsets[[1]], initial = in.par,
+    number.success ~ 1 + vsize | 1 + vsize, control = list(maxit = 1))
+    subset2 <- BinaryEPPM(data = in.data, model.name = model.names[2],
+    subset = subsets[[2]], initial = in.par,
+    number.success ~ 1 + vsize | 1 + vsize, control = list(maxit = 1))
+    slogL <- logLik(subset1) + logLik(subset2)
+    attr(slogL, which = "df") <- attr(logLik(subset1), which = "df") +
+    attr(logLik(subset2), which = "df")
+    attr(slogL, which = "nobs") <- attr(logLik(subset1), which = "nobs") +
+    attr(logLik(subset2), which = "nobs")
+    return(slogL) }, in.data = Yorkshires.litters,
+    model.names = c("beta binomial", "generalized binomial"),
+    subsets = list(1:5, 6:9), control = list(fnscale = -1),
+    hessian = TRUE)
R> cat(paste("\n", "log likelihood", round(Results$value, digits = 4),
+    "\n", sep = " "))

 log likelihood -4775.0018
```

The resulting parameter estimates together with their standard errors can be printed.

```
R> print(data.frame(parameters = c("intercept p", "slope p",
+    "intercept scale factor", "slope scale factor"), Results$par,
+    se = c(sqrt(diag(solve(Results$hessian))))), row.names = FALSE)
```

```
             parameters   Results.par          se
            intercept p -0.3443024694 0.015661870
                slope p -0.0005205837 0.001835825
 intercept scale factor  0.2083432559 0.076138071
     slope scale factor -0.0220887354 0.008843641
```

A `data.frame` of predicted summary statistics can be printed.

```
R> first.subset <- BinaryEPPM(data = Yorkshires.litters,
+    model.name = "beta binomial", subset = 1:5, initial = Results$par,
+    number.success ~ 1 + vsize | 1 + vsize, control = list(maxit = 1))
R> second.subset <- BinaryEPPM(data = Yorkshires.litters,
+    model.name = "generalized binomial", subset = 6:9,
+    initial = Results$par, number.success ~ 1 + vsize | 1 + vsize,
+    control = list(maxit = 1))
R> print(data.frame(size = Yorkshires.litters$vsize,
+    mean = c(predict(first.subset, type = "mean"),
+      predict(second.subset, type = "mean")),
+    variance = c(predict(first.subset, type = "variance"),
+      predict(second.subset, type = "variance")),
+    p = c(predict(first.subset, type = "p"),
+      predict(second.subset, type = "p")),
+    scale.factor = c(predict(first.subset, type = "scale.factor"),
+      predict(second.subset, type = "scale.factor")),
+    lower = c(predict(first.subset,
+        type = "scale.factor.limits")[["lower"]],
+      predict(second.subset, type = "scale.factor.limits")[["lower"]]),
+    upper = c(predict(first.subset,
+        type = "scale.factor.limits")[["upper"]],
+      predict(second.subset, type = "scale.factor.limits")[["upper"]])),
+    row.names = FALSE)
```

```
 size     mean variance         p scale.factor     lower    upper
    5 2.534078 1.378309 0.5068156    1.1028520 0.4374562 5.000000
    6 3.039805 1.617853 0.5066342    1.0787585 0.4526227 6.000000
    7 3.545169 1.846277 0.5064527    1.0551913 0.4622157 7.000000
    8 4.050170 2.063953 0.5062713    1.0321390 0.4688047 8.000000
    9 4.554809 2.271241 0.5060899    1.0095903 0.4735900 9.000000
   10 5.060765 2.471189 0.5060765    0.9886216 0.0000000 2.023917
   11 5.567677 2.663229 0.5061524    0.9685937 0.0000000 2.023174
   12 6.074282 2.845601 0.5061901    0.9486791 0.0000000 2.022432
   13 6.580583 3.018647 0.5061987    0.9289574 0.0000000 2.021691
```

The predicted p_s and scale factor with its limits for a litter size of 14 can be produced from the fitted model, illustrating use of the `newdata` argument of `predict`.

```
R> newdata <- data.frame(vsize = 14, vnmax = c(14),
+    mean.p = Results$par[[1]], mean.scalef = Results$par[[2]])
R> print(data.frame(size = newdata$vsize,
+    p = predict(subset2, newdata, type = "p"),
+    scale.factor = predict(second.subset, newdata, type = "scale.factor"),
+    lower = predict(second.subset, newdata = newdata,
+    type = "scale.factor.limits")[["lower"]],
+    upper = predict(second.subset, newdata = newdata,
+    type = "scale.factor.limits")[["upper"]]), row.names = FALSE)

 size         p scale.factor lower    upper
   14 0.5061843      0.90948     0 2.025047
```

4.3. Chromosome aberrations

The two data sets are of chromosome aberrations amongst survivors of the atomic bombs exploded over Japan in 1945. The response variable is the number of cells that show chromosome aberrations out of one hundred tested. Although nominally the same data there are differences between the two data sets. The Prentice (1986) set `Hiroshima.grouped` consists of four frequency distributions, i.e., one for a zero dose and three others where the doses are of ranges, and it is assumed that every survivor has had one hundred cells tested. The Morel and Neerchal (2012) set `Hiroshima.case` is for individual survivors and not all survivors had one hundred cells tested. The doses of `Hiroshima.grouped` have been transformed to a standard normal gz to match those of `Hiroshima.case` which are represented by z, with zz and gzz representing dose^2. Morel and Neerchal (2012, Section 5.4) fit a beta binomial model similar to that of `"p and scale-factor"` but to p_s and the over-dispersion parameter θ of the beta binomial, both having a logit link function. The following sequence of commands replicates this model fit, but to a `"p and scale-factor"` model with log link for the scale-factor, using `Hiroshima.grouped` to provide initial estimates for fitting the model to `Hiroshima.case`.

```
R> output.group <- BinaryEPPM(number.aberrations ~ gz + gzz | gz + gzz,
+    data = Hiroshima.grouped, model.type = "p and scale-factor",
+    model.name = "beta binomial", link = "logit",
+    pseudo.r.squared.type = "max-rescaled R squared")
R> initial <- output.group$optim$par
R> names(initial) <- c("(Intercept)", "z", "zz", "(Intercept)", "z", "zz")
R> output.case <- BinaryEPPM(t/m ~ z + zz | z + zz, data = Hiroshima.case,
+    initial=initial, model.type = "p and scale-factor",
+    model.name = "beta binomial", link = "logit",
+    pseudo.r.squared.type = "max-rescaled R squared")
R> summary(output.case)

 Dependent variable a vector of numerator / denominator.
```

```
Call:
BinaryEPPM(formula = t/m ~ z + zz | z + zz, data = Hiroshima.case,
 model.type = "p and scale-factor", model.name = "beta binomial",
 link = "logit", initial = initial,
 pseudo.r.squared.type = "max-rescaled R squared")

Model type        : p and scale-factor
Model name        : beta binomial
Link p            : logit
Link scale-factor : log

 Coefficients (model for p with logit  link)

t test of coefficients:

             Estimate Std. Error t value  Pr(>|t|)
(Intercept) -3.012536   0.044732 -67.347 < 2.2e-16 ***
z            1.375235   0.055280  24.878 < 2.2e-16 ***
zz          -0.348296   0.033072 -10.532 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1' ' 1

 Coefficients (model for scale factor with log link)

t test of coefficients:

             Estimate Std. Error t value  Pr(>|t|)
(Intercept)  1.044516   0.077455 13.4855 < 2.2e-16 ***
z            0.847877   0.097408  8.7044 < 2.2e-16 ***
zz          -0.153439   0.054284 -2.8266  0.004851 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1' ' 1

 Type of estimator: ML (maximum likelihood)
 Log-likelihood: -1428.126 on 6 Df
 Pseudo R-squared: 0.557056 type max-rescaled R squared
 Number of iterations: 351 of optim method Nelder-Mead
 return code 0 successful
```

Morel and Neerchal (2012) report a lower log-likelihood value of -1429.6 for their model which had the parameter θ of the beta binomial distribution, rather than the scale-factor, dependent on the covariates.

Using an alternative generalized binomial model with complementary log-log link and log link functions for p_s and f_s respectively, resulted in a log-likelihood of -1755.506 for the `Hiroshima.case` data set, showing a much poorer fit than the beta binomial which would generally be preferred for over-dispersed data.

4.4. Food stamps

These data on food stamps are used as an example in Künsch, Stefanski, and Carroll (1989) available from package **robustbase** (Maechler *et al.* 2019; Todorov and Filzmoser 2009). Here they are used to illustrate how to use weights. The methodology used in **BinaryEPPM** is maximum weighted likelihood estimation, which is associated with robust estimation. The weights used come from use of `glmrob` from **robustbase** although their use here does not reproduce the analysis of `glmrob`. It reproduces the analysis of `glm` using the same weights. The weights are those of Example 5.2 of Künsch *et al.* (1989).

```
R> output.fn <- BinaryEPPM(participation / n ~ tenancy + suppl.income +
+    income, data = foodstamp.case, weights = foodstamp.case$weights1,
+    model.type = "p only", model.name = "binomial", link = "logit",
+    pseudo.r.squared.type = "max-rescaled R squared")
R> summary(output.fn)


 Dependent variable a vector of numerator / denominator.

Call:
BinaryEPPM(formula = participation/n ~ tenancy + suppl.income + income,
  data = foodstamp.case, weights = foodstamp.case$weights1,
  model.type = "p only",  model.name = "binomial", link = "logit",
  pseudo.r.squared.type = "max-rescaled R squared")

Model type        : p only
Model name        : binomial
Link p            : logit
Coefficients (model for p with logit  link):

t test of coefficients:

                  Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.1764381  0.7035860  0.2508 0.802345
tenancy1      -2.3639420  0.7097606 -3.3306 0.001097 **
suppl.income1  0.8515868  0.5835291  1.4594 0.146611
income        -0.0035141  0.0016648 -2.1109 0.036488 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1' ' 1

 Maximum weighted likelihood regression.
 Vector of weights used.

 Type of estimator: ML (maximum likelihood)
 Log-likelihood: -38.11763 on 4 Df
 Pseudo R-squared: 0.4045576 type max-rescaled R squared
 Number of iterations: 97 of optim method Nelder-Mead
 return code 0 successful
```

These data are also available in frequency distribution form with the dependent variable now `l.participation` and the weights variable `l.weights1` defined as `list`. Use of a list for `data` expects there to be a `list` within the list `data` which is named the dependent variable in `Formula`. The two normalization attributes of `weights` which is also expected to be a `list`, have been set to illustrate how normalization can be done. The code below is for analyzing data where the dependent variable and the weights are `list`. The output is essentially the same as that above and hence not shown.

```
R> attr(l.weights1, which = "normalize") <- TRUE
R> attr(l.weights1, which = "norm.to.n") <- 150
R> output.fn <- BinaryEPPM(l.participation ~ tenancy + suppl.income + income,
+    data = foodstamp.grouped, model.name = "binomial",
+    link = "logit", weights = foodstamp.grouped$l.weights1,
+    pseudo.r.squared.type = "max-rescaled R squared")
R> summary(output.fn)
```

4.5. Other data sets

Testing of **BinaryEPPM** used other data sets which have been included in **BinaryEPPM** but not reported here. Code for use with these other data sets is available in a supplementary file. These data sets are from Kupper and Haseman (1978), Williams (1996), Hilbe (2011) (Titanic data of Table 9.37), and Prater (1956) (gasoline yield). The last of these has gasoline yield as a binomial variable with $n = 1000$. The code reproduces the analyses of Cribari-Neto and Zeileis (2010) on these data, where the response is (gasoline yield)/n and is treated as a continuous beta distributed variable between 0 and 1. The analyses as a beta binomial variable were done to compare how closely the two analyses agree. Models with an n of such size stress **BinaryEPPM** due to the sizes of the matrices involved, and the time taken to run them, so they are not recommended. However, the close agreement of the results does illustrate the similarity of a beta binomial analysis of a discrete variable with a beta distribution analysis of the analogous continuous variable.

5. Concluding remarks

This article has described the use of package **BinaryEPPM** to fit EPPMs and other distributional models to grouped binary data exhibiting under- and/or over-dispersion relative to the binomial distribution. A variety of covariate dependencies and data structures are covered in examples that provide illustrations of the ways in which **BinaryEPPM** can be used in the analysis of grouped binary data. It complements the similar modeling in Smith and Faddy (2016) of count data using EPPMs. Package **CountsEPPM** (Smith and Faddy 2016) is available on the Comprehensive R Archive Network (CRAN) as a contributed package at https://CRAN.R-project.org/package=CountsEPPM.

References

Bates D, Maechler M (2019). ***Matrix****: Sparse and Dense Matrix Classes and Methods*. R package version 1.2-17, URL https://CRAN.R-project.org/package=Matrix.

Brooks RJ, James WH, Gray E (1991). "Modelling Sub-Binomial Variation in the Frequency of Sex Combinations in Litters of Pigs." *Biometrics*, **47**(2), 403–417. doi:10.2307/2532134.

Canty A, Ripley BD (2019). ***boot**: Bootstrap R (S-PLUS) Functions*. R package version 1.3-23, URL https://CRAN.R-project.org/package=boot.

Cox DR, Snell E (1989). *Analysis of Binary Data*. 2nd edition. Chapman & Hall.

Cribari-Neto F, Zeileis A (2010). "Beta Regression in R." *Journal of Statistical Software*, **34**(2), 1–24. doi:10.18637/jss.v034.i02.

Dobson AJ, Barnett A (2008). *An Introduction to Generalized Linear Models*. 3rd edition. Chapman & Hall.

Faddy MJ (1997). "Extended Poisson Process Modelling and Analysis of Count Data." *Biometrical Journal*, **39**(4), 431–440. doi:10.1002/bimj.4710390405.

Faddy MJ, Smith DM (2008). "Extended Poisson Process Modelling of Dilution Series Data." *Journal of the Royal Statistical Society C*, **57**(4), 461–471. doi:10.1111/j.1467-9876.2008.00622.x.

Faddy MJ, Smith DM (2012). "Extended Poisson Process Modelling and Analysis of Grouped Binary Data." *Biometrical Journal*, **54**(3), 426–435. doi:10.1002/bimj.201100214.

Finney DJ (1971). *Statistical Methods in Biological Assay*. 2nd edition. Griffin.

Ford I, Torsney B, Wu C (1992). "The Use of a Canonical Form in the Construction of Locally Optimal Designs for Non–Linear Problems." *Journal of the Royal Statistical Society B*, **54**(2), 569–583. doi:10.1111/j.2517-6161.1992.tb01897.x.

Gaudard MA, Karson MJ, Linder E, Tse SK (1993). "Efficient Designs for Estimation in the Power Logistic Quantal Response Model." *Statistica Sinica*, **3**(1), 233–243.

Gilbert P, Varadhan R (2019). ***numDeriv**: Accurate Numerical Derivatives*. R package version 2016.8-1.1, URL https://CRAN.R-project.org/package=numDeriv.

Goulet V, Dutang C, Maechler M, Firth D, Shapira M, Stadelmann M (2019). ***expm**: Matrix Exponential, Log, 'etc.'*. R package version 0.999-4, URL https://CRAN.R-project.org/package=expm.

Grün B, Kosmidis I, Zeileis A (2012). "Extended Beta Regression in R: Shaken, Stirred, Mixed, and Partitioned." *Journal of Statistical Software*, **48**(11), 1–25. doi:10.18637/jss.v048.i11.

Hilbe JM (2011). *Negative Binomial Regression*. 2nd edition. Cambridge University Press.

Hughes G, Madden LV (1995). "Some Methods Allowing for Aggregated Patterns of Disease Incidence in the Analysis of Data from Designed Experiments." *Plant Pathology*, **44**(6), 927–943. doi:10.1111/j.1365-3059.1995.tb02651.x.

Künsch HR, Stefanski LA, Carroll RJ (1989). "Conditionally Unbiased Bounded-Influence Estimation in General Regression Models, with Applications to Generalized Linear Models." *Journal of the American Statistical Association*, **84**(406), 460–466. doi:10.1080/01621459.1989.10478791.

Kupper LL, Haseman JK (1978). "The Use of a Correlated Binomial Model for the Analysis of Toxicological Experiments." *Biometrics*, **34**(1), 69–76. doi:10.2307/2529589.

Maechler M, Rousseeuw P, Croux C, Todorov V, Ruckstuhl A, Salibian-Barrera M, Verbeke T, Koller M, Conceicao ELT, di Palma MA (2019). ***robustbase**: Basic Robust Statistics*. R package version 0.93-5, URL https://CRAN.R-project.org/package=robustbase.

Morel JG, Neerchal NK (2012). *Overdispersion Models in SAS*. SAS Press.

Nagelkerke NJD (1991). "A Note on a General Definition of the Coefficient of Determination." *Biometrika*, **78**, 691–692.

Nelder JA, Mead R (1967). "A Simplex Method for Function Minimisation." *The Computer Journal*, **7**(4), 308–313.

Prater NH (1956). "Estimate Gasoline Yields from Crudes." *Petroleum Refiner*, **35**(5), 236–238.

Prentice RL (1986). "Binary Regression Using an Extended Beta-Binomial Distribution, with Discussion of Correlation Induced by Covariate Measurement Errors." *Journal of the American Statistical Association*, **81**(394), 321–327. doi:10.1080/01621459.1986.10478275.

R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Smith DM (1983). "Algorithm AS189. Maximum Likelihood Estimation of the Parameters of the Beta Binomial Distribution." *Journal of the Royal Statistical Society C*, **32**(2), 196–204.

Smith DM, Faddy MJ (2016). "Mean and Variance Modeling of Under- and Overdispersed Count Data." *Journal of Statistical Software*, **69**(6), 1–23. doi:10.18637/jss.v069.i06.

Smith DM, Faddy MJ (2019). ***BinaryEPPM**: Mean and Variance Modeling of Binary Data*. R package version 2.3, URL https://CRAN.R-project.org/package=BinaryEPPM.

Tibshirani RJ, Ciampi A (1983). "A Family of Proportional- and Additive-Hazards Models for Survival Data." *Biometrics*, **39**(1), 141–147. doi:10.2307/2530814.

Todorov V, Filzmoser P (2009). "An Object-Oriented Framework for Robust Multivariate Analysis." *Journal of Statistical Software*, **32**(3), 1–47. doi:10.18637/jss.v032.i03.

Williams DA (1996). "Overdispersion in Logistic Linear Models." In BJT Morgan (ed.), *Statistics in Toxicology*, pp. 75–84. Oxford Science Publications.

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, Hothorn T (2002). "Diagnostic Checking in Regression Relationships." *R News*, **2**(3), 7–10.

Affiliation:

David M.Smith
IBM Watson Health
7700 Old Georgetown Road, 6th floor
Bethesda, MD 20814, United States of America
E-mail: smithdm1@us.ibm.com

Malcolm J. Faddy
School of Mathematical Sciences
Queensland University of Technology
G.P.O. Box 2434
Brisbane Qld 4001, Australia
E-mail: m.faddy@qut.edu.au

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
July 2019, Volume 90, Issue 8 *Submitted:* 2016-12-15
doi:10.18637/jss.v090.i08 *Accepted:* 2018-05-08