revengc: An R package to reverse engineer decoupled and censored data

Samantha Duchscherer, Robert Stewart, and Marie Urban
Oak Ridge National Laboratory, 1 Bethel Valley Road, Oak Ridge, TN 37831

Abstract

An issue occurs when authors do not reveal clear information. Decoupled variables (e.g. separate averages) and numeric censoring (e.g. between ages 10-15) are reoccurring instances found in areas ranging from demographic and epidemiological data to ecological inference problems. Decoupled variables provide no availability for cross tabulations while censoring obscures the true underlying values. The **revengc** R package was developed to reverse engineer this unclear information that is continually reported by many well-established organizations (e.g. World Health Organization (WHO), Centers for Disease Control and Prevention (CDC), World Bank, and various national censuses). There are two main functions in **revengc** and both fit data to a Poisson or Quasi-Poisson distribution. The *estimated_ lambda* function takes a univariate censored frequency table and approximates its lambda (average) value. The *rec* function estimates an uncensored bivariate table from decoupled and summarized arguments.

1 Introduction

The **revengc** R package was originally developed to aid in the efforts of modeling building occupancy [1]. Household size and area of residential structures are typically found in any given national census. If a census revealed the raw data or provided a full uncensored contingency table (household size $*$ area), computing interior density as people per area would be straightforward. However, household size and area are often reported as decoupled variables (separate univariate frequency tables, average values, or a combination of the two) or a provided contingency table is left ($<$, $\leq$), right ($>$, $\geq$), and interval (-) censored. This type of information is problematic. How can a people per area ratio be calculated when no affiliation between the variables exist? If a census reports a household size average of 5.3, then how many houses are there with 1 person, 2 people,..., 10 people? If a census reports that there are 100 houses in an area of 26-50 square meters, then how many houses are in 26, 27,..., 50 square meters?

We decided to improve population density estimates by relying on Poisson as our common underlying distribution. Household size follows a univariate Poisson distribution across several countries [2] while modeling area as a Poisson distribution of "counted" units of measure arose as a matter of discussion. Area is a continuous property and its distribution varies in open source information [3, 4, 5]. However, census data rarely reports area with precision greater than simple integer values for any unit of measure. It is therefore not unreasonable to think of these data as unit of measure "counts." It is also known that for large values of the Poisson parameter (e.g. $\lambda > 20$) the Poisson and the Normal distribution are roughly equivalent. If the underlying continuous distribution for area is Gaussian, the Poisson solution here will still provide a very close approximation. Furthermore, the Poisson distribution has the convenient property that its only parameter (λ) is equal to the mean of the count data, a commonly encountered value.

Although we developed **revengc** for a specific purpose, a tool that can estimate a Poisson lambda value from a censored table and estimate interior cells of a contingency table governed by Poisson marginals can be useful for many other applications. For example, the Poisson distribution is very common to population and community ecologist [6]. **revengc** could aid with censored tables consisting of organism counts (e.g. ¿100 species). A Poisson distribution has also shown to be effective for modeling life expectancy and mortality [7], which are two variables that are notorious for being summarized. Life expectancy is typically an average while the number of deaths are usually provided in a censored frequency table. Other summarized examples that could be fit to a Poisson distribution includes the average number of births, the number of new cases of disease (censored table), the number of mutations in a gene (censored table) or average mutation rate,

etc. With various applications resulting in different data formats, **revengc** offers five scenarios that can be reverse engineered into an uncensored contingency table:

1. A user provides a univariate frequency table to estimate a lambda value
2. A user provides decoupled averages to estimate an uncensored contingency table
3. A user provides decoupled frequency tables to estimate an uncensored contingency table
4. A user provides an average and frequency table to estimate an uncensored contingency table
5. A user provides a censored contingency table to estimate an uncensored contingency table

This paper proceeds with our reverse engineering methodology. In the Methods section, we first provide a methodology workflow for *estimated_ lambda* and *rec*. The Methods section continues with an in-depth analysis of how overdispersion and bounds effect our main functions, how a maximum likelihood function estimates a lambda value, and how we utilize the **mipfp** R package to calculate cross tabulations [8]. Since the **revengc** package has specific input requirements, we continue with a Usage and Data entry section. Both aid in explaining how to effectively implement our functions with correctly formatted table(s). We then provide a coded example that implements **revengc** on national census data (household size and area) and end with concluding remarks.

2 Methods

2.1 Workflow

Before we provide an in-depth analysis of the mathematical methods utilized in *estimated_ lambda* and *rec*, it will be helpful to first understand their workflow. The *estimated_ lambda* function methodology is relatively straightforward. To provide an estimated lambda value, a univariate frequency table is fit to a Poisson or Quasi-Poisson distribution using a maximum likelihood function customized to handle censoring and truncation. Since the *rec* function is more complex, we display a workflow of its methodology in Figure 1.

2.2 Poisson and Quasi-Poisson

In a Poisson distribution, the variance equals the mean (1).

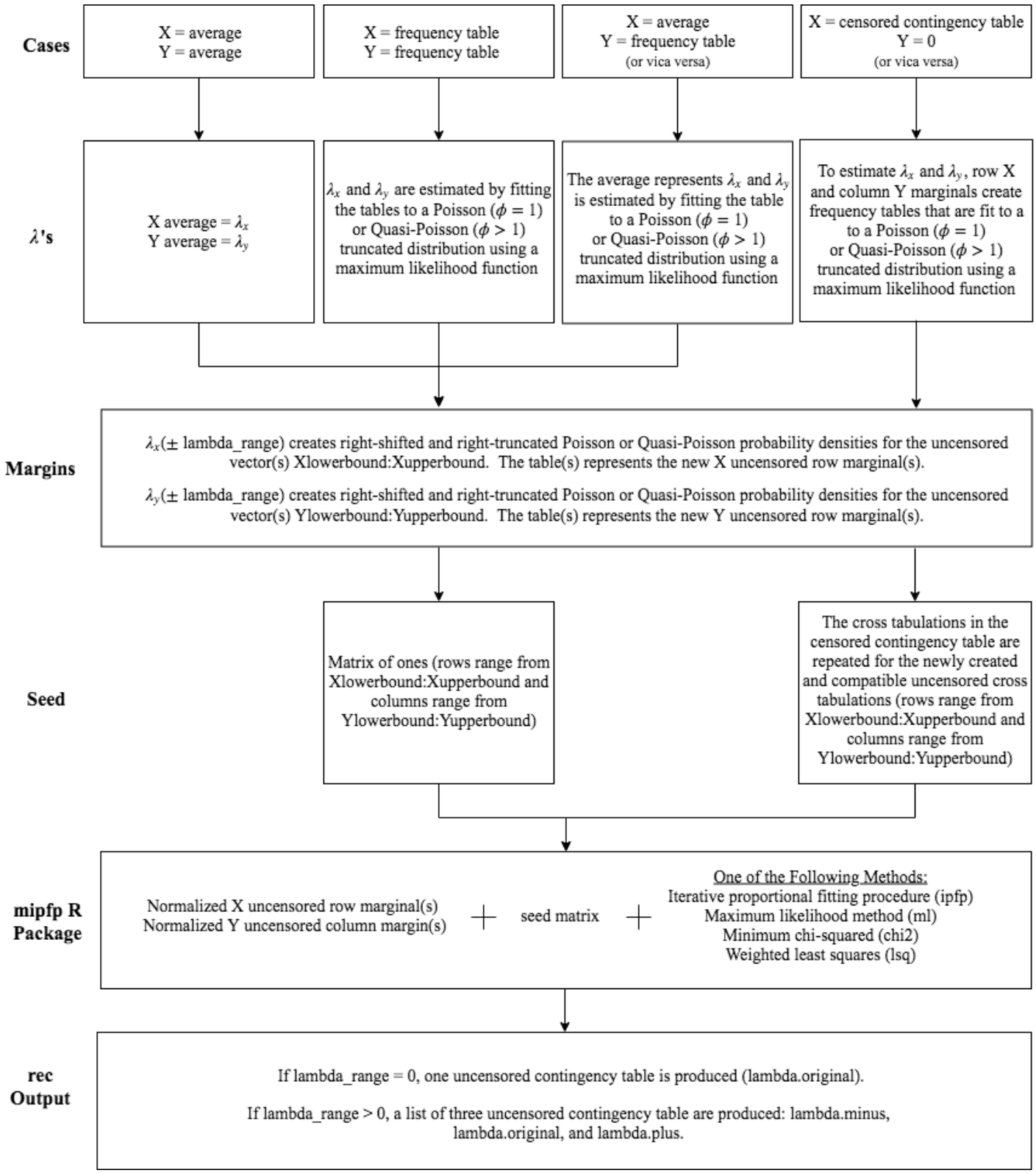
$$Var(Y) = E(Y) = \mu \tag{1}$$

However, there are many cases where data has more variation than what is indicated by the Poisson distribution. Sometimes variance is larger than the mean (overdispersion). To account for overdispersion in count data, **revengc** provides the option of implementing the Quasi-poisson distribution (variance is proportional to the mean by a scalar value (2)). If the scalar value $\phi = 1$, then the variance obviously equals the mean and the original Poisson mean-variance relationship holds. When the user wishes to accommodate for overdispersion in their data ϕ should be set to a value ¿ 1.

$$Var(Y) = \phi E(Y) = \phi\mu \tag{2}$$

revengc does not account for the rarer instances of the variance being smaller than the mean (underdispersion) because we use the negative binomial distribution to generate the Quasi-Poisson distribution. To show this, let's first recall that a negative binomial distribution (size = r, probability = p, and k = 0, 1, 2....) has the following moments (3)

$$\begin{aligned} E(X) &= \frac{p \cdot r}{1-p} = \mu \\ Var(X) &= \frac{p \cdot r}{(1-p)^2} = \frac{\mu}{1-p} \end{aligned} \tag{3}$$



Workflow of rec function.

We can find an equation between ϕ and p by first letting $\phi = \frac{1}{1-p}$. Arranging $E(X)$ in Equation (3) to $p = \frac{\mu}{r+\mu}$ and substituting p into our defined ϕ gives $\frac{1}{\phi} = \frac{r}{r+\mu}$, which simplifies to $r = \frac{\mu}{\phi-1}$. Therefore, to generate a Quasi-Poisson probability we use a negative binomial distribution where $\mu = \lambda$ and $r = \frac{\mu}{\phi-1}$. The size parameter r is negative when ϕ ¡ 1 (underdispersion) , and this is not possible for the negative binomial distribution.

2.3 Lower and upper bounds

Certain probabilities do not exist (e.g. 1 person per 0 m^2) while other probabilities are a minuscule value worth refining to zero (e.g. 2 people per 1,000,000 m^2). Therefore, this package incorporates bounds. We conduct a right shift to make the probabilities below a lower bound nonexistent. We also omit existing probabilities above an upper bound by calculating a right-truncated distribution.

Conducting a right-shift is simple. Considering a random variable X $\sim$ Poisson(λ) where l = lower bound and u = upper bound

$$P(X = x) = e^{-\lambda}\frac{\lambda^x}{x!} \qquad (x = 0, 1, 2, ..., (u - l)) \tag{4}$$

Now consider $Y = X + l$. This imples Y is just X shifted to the right by the provided lower bound l value

$$P(Y = y) = e^{-\lambda}\frac{\lambda^{y-l}}{(y-l)!} \qquad (y = l, l + 1, l + 2, ..., (u + l)) \tag{5}$$

Building on this concept, Equation (6) displayes a Poisson distribution for a right-truncated variable Y [9]. The provided upper bound u is the truncation point for right-shifted y. The right-shift and right-truncation concept does not change when $\phi \neq 1$. The truncated Poisson probabilities just become truncated probabilities that follows the assumption of the variance being proportional to the mean.

$$P(Y = y) = \frac{e^{-\lambda}\lambda^y}{(y)!\left(1 - \sum_{y=u+1}^{u+l} P(Y = y)\right)} \qquad (y = l, l + 1, l + 2, ..., (u + l)) \tag{6}$$

In summary, the upper bound in *estimated_ lambda* aids calculating the right-truncated Poisson or Quasi-Poisson probabilities in a customized maximum likelihood function. Typically, setting the upper bound value between +10 and +20 of the last categorical value in the table is sufficient for this function. The lower and upper bounds in *rec* implies that the final contingency table has rows of *i = X lower bound, X lower bound + 1,..., X upper bound* and columns of *i = Y lower bound, Y lower bound + 1,..., Y upper bound*. Choosing the bounds in *rec* can be selected intuitively with a brute force method. If *rec* outputs a final contingency table with higher probabilities near the edge(s) of the table, then it would make sense to increase the range of the bound(s). For both variables, this would just involve making the lower bound less, making the upper bound more, or doing a combination of the two. The opposite holds true as well. If the final contingency table in *rec* has very low probabilities near the edge(s) of the table, then a user should decrease the range of the particular bound(s).

2.4 Maximum likelihood function for censored frequency tables

Censored frequency tables are fit to a Poisson or Quasi-Poisson truncated distribution using a maximum likelihood function customized to handle left ($<$, $\leq$), right ($>$, $\geq$), and interval (-) censored data. The primary interested is to find the value for λ that maximizes the log-likelihood of a truncated Poisson or Quasi-Poisson distribution. To see an example function with selective censoring, Equation (7) represents the log-likelihood function for variable x. The cases for uncensored (x), left censored ($x < c$), interval censored ($a \leq x \leq b$), and right censored ($x > d$) are all represented where *a, b, c,* and d represent the censoring limits. Note, u represents the provided upper bound and we did not consider any type of right-shift (lower bound = 0). Since three of the four cases in *rec* use estimated lambda value(s) for further calculations, *rec* has a lambda_ range argument. This argument is added and subtracted from the lambdas that create uncensored marginals.

$$
\begin{aligned}
L\left(\lambda|x\right)_{\log} = & \\
& \sum_{x} \ln\left(P\left(x|\lambda\right)\right) \\
& + \sum_{x<c} \ln\left(P\left(x<c|\lambda\right)\right) \\
& + \sum_{a\leq x\leq b} \ln\left(P\left(a\leq x\leq |\lambda\right)\right) \\
& + \left(\sum_{x>d} \ln\left(P\left(x>d|\lambda\right)\right) - \sum_{x>u} \ln\left(P\left(x>u|\lambda\right)\right)\right)
\end{aligned}
\tag{7}
$$

2.5 Utilizing the mipfp R package in rec

The *rec* function estimates an uncensored contingency table by utilizing the **mipfp** R package. **mipfp** provides four methods of adjusting cell proportions p_{xy} in a X $*$ Y contingency table to known marginal probabilities π_{x+} and π_{+y}. The estimators, $\hat{\pi}_{xy}$, of π_{xy} are calculated with the following marginal restrictions (8) [10].

$$
\begin{aligned}
\sum_{y} \hat{\pi}_{x+} \qquad & (x = 1, ..., X) \\
\sum_{x} \hat{\pi}_{+y} \qquad & (y = 1, ..., Y)
\end{aligned}
\tag{8}
$$

The different estimation methods are listed in Table 1 along with a brief summary. For a better understanding of all methods, please refer the papers by [10] and [11].

Method	**Abbreviation**	**Calculate $\hat{\pi}_{xy}$ by (all are subject to the marginal constraints (8))**
Iterative proportional fitting procedure	ipfp	Minimizing $I(\hat{\pi}; p) = \sum_x \sum_y \hat{\pi}_{xy} ln(\hat{\pi}_{xy}/p_{xy})$
Maximum likelihood method	ml	Maximizing $l(\hat{\pi}) = \sum_x \sum_y p_{xy} ln(\hat{\pi}_{xy})$
Minimum chi-squared	chi2	Minimizing $\sum_x \sum_j (\hat{\pi}_{xy} - p_{xy})^2/\hat{\pi}_{xy}$
Weighted least squares	lsq	Minimizing $\sum_x \sum_y (p_{xy} - \hat{\pi}_{xy})^2/p_{xy}$

mipfp methods to generate estimated cross tabulations [8].

The mipfp R package estimates cross tabulations with a selected seed estimation method, row and column marginals, and a seed matrix. For the decoupled cases (two averages, two tables, or a combination of a table and average), provided (average) or estimated (maximum likelihood function) lambdas first create right-shifted and right-truncated Poisson or Quasi-Poisson X and Y probability densities for uncensored vectors ranging from Xlowerbound:Xupperbound and Ylowerbound:Yupperbound, respectively. The Xlowerbound:Xupperbound vector with its corresponding normalized density values now represents the row marginal and the Ylowerbound:Yupperbound vector with its corresponding normalized density values represents the new column marginal. The absence of additional information makes it difficult to say much about the joint distribution, so the X $*$ Y seed is set to a matrix of ones. We understand this is not ideal but

offering the different methods in updating the seed as well as offering a range on the lambdas used in making the marginals does offer the opportunity to conduct sensitivity analysis.

When provided a censored contingency table, the row marginals create a univariate X frequency table while the column marginals create a univariate Y frequency table. Both tables estimate a lambda value with a customized maximum likelihood function. The process of creating right-shifted and right-truncated Poisson or Quasi-Poisson X and Y probability densities to represent the marginals is repeated. However, the seed differs here because there are already provided cross tabulations. The X $*$ Y seed matrix repeats the cross tabulations in the censored contingency table for the newly created and compatible uncensored cross tabulations. To see the seed for this case, refer to the Example section (Indonesia).

3 Usage

3.1 estimated_ lambda

The *estimated_ lambda* function has the following format with a description of each argument directly below. The output is a lambda (numeric class).

```
estimated_lambda(censoredtable, upperbound, quasipoisson_phi)
```

censoredtable: A frequency table (censored and/or uncensored). A data.frame and matrix are acceptable table classes.

upperbound: A numeric class value to represent a right-truncated point. The value cannot be less than the highest category value (e.g. the upper bound cannot be 90 if a table has a '>100' category).

quasipoisson_ phi: A numeric class value to help with overdispersion found in the censored table. If the value = 1, the original Poisson mean-variance relationship holds (mean = variance). When the value is >1, the user is accounting for overdispersion (variance becomes proportional to the mean by quasipoisson_ phi value). The value must strictly be $\geq$1.

3.2 rec

The *rec* function has the following format with a description of each argument directly below. The output is an uncensored contingency table. By an increment of one, the rows range from Xlowerbound:Xupperbound and the columns range from Ylowerbound:Yupperbound. If the lambda_ range argument = 0 then one uncensored contingency table is provided in a data.frame format. If the user sets the lambda_ range argument $>$ 0, then a list of three uncensored contingency tables is provided (lambda.minus, lambda.original, and lambda.plus).

```
rec(X,Y,Xlowerbound,Xupperbound,Ylowerbound,Yupperbound,
 quasipoisson_phiX, quasipoisson_phiY, seed_estimation_method, lambda_range)
```

X: Argument can be an average, a univariate frequency table, or a censored contingency table. The average value should be a numeric class while a data.frame or matrix are acceptable table classes. If a user inputs a censored contingency table, make Y = 0.

Y: Same description as X but this argument is for the Y variable. Make X = 0 if Y argument is a censored contingency table.

Xlowerbound: A numeric class value to represent a right-shift variable for X. The value must strictly be $\geq$ 0 and cannot be greater than the lowest category/average value provided for X (e.g. the lower bound cannot be 6 if a table has '$\geq$ 5' as a X or row category).

Xupperbound: A numeric class value to represent a right-truncated point for X. The value cannot be less than the highest category/average value provided for X (e.g. the upper bound cannot be 90 if a table has '> 100' as a X or row category).

Ylowerbound: Same description as Xlowerbound but this argument is for Y (column variable in contingency table).

Yupperbound: Same description as Xupperbound but this argument is for Y (column variable in contingency table).

quasipoisson_ phiX: A numeric class value to help with overdispersion found in X. If the value = 1, the original Poisson mean-variance relationship for X (row of contingency table) holds, which implies mean = variance. When the is > 1, the user is accounting for overdispersion in X (variance becomes proportional to the mean by quasipoisson_ phiX value). The value must strictly be ≥ 1.

quasipoisson_ phiY: Same description as quasipoisson_ phiX but this argument is for Y (column variable in contingency table).

seed_ estimation_ method: A character string indicating which method is used for updating the cross tabulations. The choices are: "ipfp", "ml", "chi2", or "lsq".

lambda_ range: A numeric class value to represent a range on the X lambda and Y lambda. The value is added (lambda.plus) and subtracted (lambda.minus) from the given (average) or estimated (maximum likelihood function) X lambda and Y lambda (lambda.original). Set to 0 if generating one table (lambda.original) is the desired output. Set value > 0 to calculate three separate uncensored contingency tables with their corresponding X and Y marginal lambdas (lambda.minus, lambda.original, lambda.plus).

4 Data entry

The censored table for the *estimated_ lambda* function has restrictions. The univariate frequency table, which can be a data.frame or matrix class, must have two columns and n number of rows. The categories must be in the first column with the frequencies in the second column. Row names should never be placed in this table (the default row names should always be 1:n). Column names can be any character string. The only symbols accepted for censored data are listed below. Less than or equal to ($\leq$ and LE) is not equivalent to less than ($<$ and L) and greater than or equal to ($\geq$, +, and GE) is not equivalent to greater than ($>$ and G).

- **Left censored symbols:** $<$, $\leq$, L, and LE
- **Interval censored symbols:** $-$ and I (symbol has to be placed in the middle of the two category values)
- **Right censored symbols:** $>$, $\geq$, +, G, and GE
- **Uncensored symbol:** no symbol (only provide category value)

To provide examples, the three tables below use different censored symbols yet give the same *estimated_ lambda* output.

The censored contingency table for *rec* also has a strict format. The censored symbols should follow the requirements listed above. The table's class can be a data.frame or a matrix. The column names should be

Category	Frequency	Category	Frequency	Category	Frequency
≤ 6	11800	LE 6	11800	< 7	11800
7-12	57100	7 I 12	57100	7 I 12	57100
13-19	14800	13 I 19	14800	13-19	14800
20+	3900	GE 20	3900	≥ 20	3900

Examples of correctly formatted univariate tables.

the Y category values. Row names should never be placed in this table, the default should always be 1:n. The first column should be the X category values. The inside of the table are X $*$ Y cross tabulation, which are either non-negative if seed_estimation_method is "ipfp" or strictly positive when method is "ml", "lsq" or "chi2". The row and column marginal totals corresponding to their X and Y category values need to be placed in this table. The top left, top right, and bottom left corners of the table should be NA or blank. The bottom right corner can be a total cross tabulation sum value, NA, or blank. Table 2 is a formatted example with percentages as the cross tabulations and the bottom right corner as a total sum.

NA	<20	20-30	>30	NA
<5	0.18	0.19	0.08	0.45
5-9	0.13	0.08	0.12	0.33
$\geq$10	0.07	0.05	0.10	0.21
NA	0.38	0.32	0.31	NA

Example of a correctly formatted bivariate table.

4.1 Formatting table in R

We will now show how to format these tables properly in R. The following code shows how tables can be created. If a user wants to read in a file, the table format must be identical.

```
# table with one variable (univariate frequency table)
# univariatetable.csv is a preloaded example that provides the same table
univariatetable<-cbind(as.character(c("1-2", "3-4", "5-6", "7-8", ">=9")),
  c(16.2, 41.7, 29.0, 9.0, 4.1))


# table with two variables (contingency table)
# contingencytable.csv is a preloaded example that provides the same table
contingencytable<-matrix(c(6185,9797,16809,11126,6156,3637,908,147,69,4,
  5408,12748,26506,21486,14018,9165,2658,567,196,78,
  7403,20444,44370,36285,23576,15750,4715,994,364,136,
  4793,17376,44065,40751,28900,20404,6557,1296,555,228,
  2354,11143,32837,33910,26203,19301,6835,1438,618,245,
  1060,6038,19256,21298,17774,13864,4656,1039,430,178,
  273,2521,9110,11188,9626,7433,2608,578,196,112,
  119,1130,4183,5566,5053,3938,1367,318,119,66,
  33,388,1707,2367,2328,1972,719,171,68,37,
  38,178,1047,1672,1740,1666,757,193,158,164),
  nrow=10,ncol=10, byrow=TRUE)
rowmarginal<-apply(contingencytable,1,sum)
contingencytable<-cbind(contingencytable, rowmarginal)
```

```
colmarginal<-apply(contingencytable,2,sum)
contingencytable<-rbind(contingencytable, colmarginal)
row.names(contingencytable)[row.names(contingencytable)=="colmarginal"]<-""
contingencytable<-data.frame(c("1","2","3","4","5","6", "7", "8","9","10+", NA),
  contingencytable)
colnames(contingencytable)<-c(NA,"<20","20-29","30-39","40-49","50-69","70-99",
  "100-149","150-199","200-299","300+", NA)
```

5 Worked examples

5.1 Nepal

A Nepal Living Standards Survey [12] provides both a censored table and average for urban household size. We use the censored table to show that the estimated_ lambda function calculates a close approximation to the provided average household size (4.4 people). Below we set the upper bound to 15 (a reasonable guess from the provided table) and assume the household size table follows the original Poisson mean-variance relationship (quasipoisson_ phi = 1).

```
# revengc has the Nepal houshold table preloaded as univariatetable.csv
# result = 4.37
estimated_lambda(censoredtable = univariatetable.csv,
  upperbound = 15, quasipoisson_phi = 1)
```

5.2 Indonesia

In 2010, the Population Census Data - Statistics Indonesia provided over 60 censored contingency tables containing Floor Area of Dwelling Unit (square meter) by Household Member Size. The tables are separated by province, urban, and rural. Here we use the household size by area contingency table for Indonesia's rural Aceh Province to show the multiple coding steps and functions implemented inside *rec*. This allows the user to see a methodology workflow in code form. The final uncensored household size by area contingency table, which was estimated with the "ipfp" method, has rows ranging from 1 (Xlowerbound) to 15 (Xupperbound) people and columns ranging from 10 (Ylowerbound) to 310 (Yupperbound) square meters. For this example, we assume the original Poisson mean-variance relationship (both quasipoisson_ phi arguments = 1) and only estimate one uncensored contingency table (lambda_ range = 0).

```
# data = Indonesia 's rural Aceh Province censored contingency table
# preloaded as 'contingencytable.csv'
contingencytable.csv

# provided upper and lower bound values for table
# X=row and Y=column
Xlowerbound=1
Xupperbound= 15
Ylowerbound=10
Yupperbound=310

# table of row marginals provides lambda x
row.marginal.table<-row_marginal(contingencytable.csv)
lambdax<-estimated_lambda(row.marginal.table, upperbound=Xupperbound, quasipoisson_phi=1)
# table of column marginals provides lambda y
column.marginal.table<-column_marginal(contingencytable.csv)
lambday<-estimated_lambda(column.marginal.table, upperbound=Yupperbound, quasipoisson_phi=1)

# incorporate right shift from lower bounds
```

```
rowrange<-0:(Xupperbound-Xlowerbound)
rowrange<-rowrange+ Xlowerbound
colrange<-0:(Yupperbound-Ylowerbound)
colrange<-colrange+Ylowerbound

# new uncensored row marginal table
new.row.marginals<-dqpois_trunc(rowrange,lambda=lambdax, quasipoisson_phi=1, upperbound=15)
# row marginal == column marginal == 1 (marginal sums have to be equal)
new.row.marginals<-new.row.marginals/sum(new.row.marginals)
# new uncensored column margin table
new.column.marginals<-dqpois_trunc(colrange,lambda=lambday, quasipoisson_phi=1, upperbound=310)
# column marginal == row marginal == 1 (marginal sums have to be equal)
new.column.marginals<-new.column.marginals/sum(new.column.marginals)

# create uncensored seed from censored table
seed.output<-seedmatrix(contingencytable.csv , Xlowerbound=1,
  Xupperbound=15, Ylowerbound=10, Yupperbound=310)

# run mipfp
# store the new margins in a list
tgt.data<-list(new.row.marginals, new.column.marginals)
# list of dimensions of each marginal constrain
tgt.list<-list(1,2)
# calling the estimated function
## seed has to be in array format for mipfp package
## ipfp is the selected seed_estimation_method
final1<-Estimate(array(seed.output,dim=c(length(Xlowerbound:Xupperbound),
  length(Ylowerbound:Yupperbound))), tgt.list, tgt.data, method="ipfp")$x.hat

# rec function outputs the same table
final2<-rec(X= contingencytable.csv,
  Y = 0,
  Xlowerbound = 1,
  Xupperbound = 15,
  Ylowerbound = 10,
  Yupperbound = 310,
  quasipoisson_phiX = 1,
  quasipoisson_phiY =  1,
  seed_estimation_method= "ipfp",
  lambda_range =0)

# check that both data.frame results have same values
all(final1 == final2)
```

6 Conclusion

Summarized and decoupled variables unfortunately lead to unclear results. **revengc** was designed to reverse engineer data that fits a Poisson or Quasi-Poisson truncated distribution. If provided a univariate censored and/or uncensored table, *estimated_ lambda* provides the lambda value (average) for that table. When provided two variables, which could consist of decouples average(s) and frequency table(s) as well as a censored contingency table, *rec* outputs an uncensored contingency table with cross tabulation estimated by the **mipfp** R package.

References

[1] R. Stewart, M. Urban, S. Duchscherer, J. Kaufman, A. Morton, G. Thakur, J. Piburn, and J. Moehl, "A bayesian machine learning model for estimating building occupancy from open source data," *Natural Hazards*, vol. 81, no. 3, pp. 1929–1956, 2016.

[2] V. Jennings, B. Lloyd-Smith, and D. Ironmonger, "Household size and the poisson distribution," *Journal of the Australian Population Association*, vol. 16, pp. 65–84, 1999.

[3] H. A. Klaiber Jr, *Valuing open space in a locational equilibrium model of the Twin Cities*. Proquest, Umi Dissertation Publishing, 9 2011.

[4] A. Shuman, "Behrens ranch round rock tx market report 2009-2010," 1 2011.

[5] T. Ohnishi, T. Mizuno, C. Shimizu, and T. Watanabe, "On the evolution of the house price distribution," 2010.

[6] R. B. O'hara and D. J. Kotze, "Do not log-transform count data," *Methods in Ecology and Evolution*, vol. 1, no. 2, pp. 118–122, 2010.

[7] J. Li, "A poisson common factor model for projecting mortality and life expectancy jointly for females and males," *Population studies*, vol. 67, no. 1, pp. 111–126, 2013.

[8] Johan Barthélemy, Thomas Suesse, Mohammad Namazi-Rad, *Multidimensional Iterative Proportional Fitting and Alternative Models*. R Foundation for Statistical Computing, 2018.

[9] S. E. Saffari, R. Adnan, and W. Greene, "Handling of over-dispersion of count data via truncation using poisson regression model," *Journal of Computer Science and Computational Mathematics*, vol. 1, no. 1, pp. 1–4, 2011.

[10] R. J. Little and M.-M. Wu, "Models for contingency tables with known margins when target and sampled populations differ," *Journal of the American Statistical Association*, vol. 86, no. 413, pp. 87–95, 1991.

[11] T. Suesse, M.-R. Namazi-Rad, P. Mokhtarian, and J. Barthélemy, "Estimating cross-classified population counts of multidimensional tables: an application to regional australia to obtain pseudo-census counts," vol. 33, no. 4, 2017.

[12] N. P. C. S. Government of Nepal, "Nepal living standards survey," tech. rep., Central Bureau of Statistics, 11 2011.