A New Versatile Discrete Distribution

by Rolf Turner

Abstract This paper introduces a new flexible distribution for discrete data. Approximate moment estimators of the parameters of the distribution, for use as starting values for a numerical maximum likelihood procedure, are discussed. The quality of the maximum likelihood estimates is assessed via simulation experiments. Several examples are given of fitting instances of the new distribution to real and simulated data. It is noted that the new distribution is a member of the exponential family. Expressions for the gradient and hessian of the log likelihood of the new distribution are derived. The former facilitates the numerical maximisation of the likelihood with `optim()`; the latter provides means of calculating or estimating the covariance matrix of of the parameter estimates. A discrepancy between estimates of the covariance matrix obtained by inverting the hessian and those obtained by Monte Carlo methods is discussed.

Introduction

Modelling the distribution of discrete data sets can be problematic in that it is often the case that none of the "standard" distributions appears to be appropriate. It is possible to use a "completely non-parametric" approach (in other words, to apply multinomial distributions, specified in a very simple manner by means of tables). However this approach often turns out to be a little *too* flexible. In particular, in the context of hidden Markov models for discrete data (**hmm.discnp**, Turner 2020), the number of quantities to estimate rapidly becomes unwieldy. Estimates are unstable, the sensitivity of fitting algorithms to starting values is exacerbated, and problems with the convergence of fitting algorithms arise.

To address these problems I developed a new discrete distribution, termed the "db" ("discretised Beta") distribution. The underlying idea is to define a family of distributions, for discrete data, with shape characteristics as flexible as those of the Beta family of continuous distributions. (See Johnson et al. 1995, Chapter 25, p. 210. See also the help for the `dbeta()` function in the **stats** package, R Core Team 2020, and Abramowitz and Stegun 1972, Chapter 6. The reader may also find it useful to access `https://en.wikipedia.org/wiki/Beta_distribution`.) The db distribution is closely related to the Beta distribution and has "shape" parameters, α and β analogous to the shape parameters of the Beta distribution.

In addition to the shape parameters, the db distribution has two other parameters which specify the "support" of the distribution. These "support parameters" are not estimated from data but must be specified by the user prior to estimating the shape parameters. The support parameters are n_{top} (a positive integer) and ζ (a logical scalar).

The parameter n_{top} is the upper limit of the support of the specified distribution. If the parameter ζ is `TRUE` then zero origin indexing is to be used, i.e. the support of the distribution is the set $\{0, 1, 2, \ldots, n_{\text{top}}\}$. Otherwise the support is $\{1, 2, \ldots, n_{\text{top}}\}$. In the first case we use the notation $n_{\text{bot}} = 0$ and in the second $n_{\text{bot}} = 1$. The first form is convenient if the variable in question may be considered to be a count and zero counts are possible. Of course one could structure the distribution always to have support of the form $\{0, 1, 2, \ldots, n_{\text{top}}\}$, simply by re-coding or shifting the data. However in several of the examples with which I was concerned it seemed more convenient to allow for a non-zero origin.

In some contexts the value of n_{top} may be known (e.g. it may be analogous to the number of trials in a binomial experiment). In other contexts it must be chosen by the user, and the choice may be influenced by the observed values of the data. (See the section **Choosing n_{top}**.)

Like the Beta distribution upon which it is based the db distribution is effectively unimodal. It can have two modes if they occur at the extremes of the support but otherwise can have only one. This characteristic is less than ideal, but seems to be unavoidable. It appears to be difficult to specify multimodal distributions (other than by way of *mixtures*, which are accompanied by other problems). *Wikipedia* (`https://en.wikipedia.org/wiki/Multimodal_distribution`, last accessed 30 March 2021) says "Bimodal distributions, despite their frequent occurrence in data sets, have only rarely been studied [citation needed]. This may be because of the difficulties in estimating their parameters either with frequentist or Bayesian methods."

A referee of an earlier version of this paper suggested that the beta-binomial distribution be considered as an alternative to the new db distribution. The beta-binomial distribution, like, the db distribution, has a flexibility of shape similar to that of the Beta distribution. However after extensive experimentation with the beta-binomial distribution I decided against using it, for reasons discussed in the following section.

The beta-binomial distribution

Although the beta-binomial distribution is very flexible with respect to its shape, it is to a large extent focussed on dealing with data sets which appear, in some way, to arise from binomial distributions but which are in fact overdispersed. (Recall that a data set is overdispersed with respect to the binomial distribution if the variance of that data set is larger than it would be if the underlying distribution were indeed binomial.) The beta-binomial distribution may be conveniently parametrised in terms of a "success probability" m (which must be strictly between 0 and 1) and an overdispersion parameter s (which must be strictly positive). This is the parametrisation chosen in the **rmutil** package, Swihart and Lindsey (2020). The reader may also find it informative to access `https://en.wikipedia.org/wiki/Beta-binomial_distribution` where the parametrisation is expressed in terms of "shape" parameters α and β. These are related to m and s by $m = \alpha/(\alpha+\beta)$ and $s = \alpha+\beta$.

Moment estimators of the parameters of a beta-binomial distribution are explicitly available, but these are often unsatisfactory in that the moment estimate of s turns out to be negative. Maximum likelihood estimates of the parameters may be obtained via numerical maximisation (using e.g. `optim()` from the **stats** package, automatically available in R). Starting values are of course required. If the moment estimates are outside of the required range, e.g. if $\hat{s}$ is negative, rough ad hoc starting values (e.g. $m = \epsilon$ or $m = 1-\epsilon$, and $s = \epsilon$, where ϵ is equal to `sqrt(.Machine$double.eps)`) appear to be adequate most of the time.

However extensive experimentation with simulated data revealed that the maximum likelihood estimate of s is frequently far too large. In one instance I simulated (using `rbetabinom()` from the **rmutil** package) 100 data sets each with 30 observations, with $m = 0.75$, $s = 10$ and `size` (the number of trials) set equal to 10. Maximum likelihood estimates of s, calculated using the true parameter values as starting values, ranged as high as 1038.82. (An "h" plot of the estimates is shown in Figure 1.) The variance of these estimates was 13032.98. In contrast the inverse of the Fisher information, calculated using the true parameter values and the data corresponding to the highest estimate of s was

```
            m           s
m 0.001121638  0.02524533
s 0.025245334 14.66327852
```

which indicates that the variance in question should be of the order of 15. I concluded that parameter estimation for the beta-binomial distribution is not sufficiently reliable and decided not to further pursue its use.

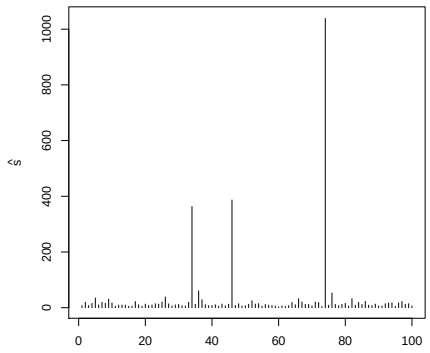

Figure 1: Estimates of the s parameter of the beta-binomial distribution.

Another mildly worrying aspect of the beta-binomial distribution is its focus on overdispersion. Underdispersed pseudo-binomial data sets are rare, but they do exist. Examples are provided in the **dbd** package (see section **Implementation**). It is indeed possible to fit beta-binomial distributions to these examples, but the meaning of the resulting fits is questionable. The estimates of s range from around 640 thousand to 78 million. Such large values of s essentially indicate that there is *no* overdispersion, i.e. that the data are in fact from a binomial distribution. In other words, the estimates are trying as hard as they can to describe the true situation, but cannot actually do so given the constraints of the model.

I encountered other problems — tendencies for errors to be thrown in various circumstances, including the application of `optim()` — in my exploration of the beta-binomial distribution, but there would appear to be no point in going into more detail here.

The db distribution can be fitted to underdispersed data sets without obvious problems. Fitting the db distribution to the examples of underdispersed data, referred to above, is discussed in **Example 6** in section **Examples**. Goodness of fit tests all yield very large p-values. However the parameter estimates from the fits are rather large, sometime bizarrely so, which raises suspicions. It is not yet clear how the values of the parameter estimates relate to under and over dispersion. This may be a topic to explore in future research.

Definition of the db distribution

Conceptually the probability mass function (PMF) of the db distribution is

$$\begin{aligned} \Pr(X = x \mid \alpha, \beta, n_{\text{top}}, \zeta) &= \frac{1}{\kappa} f\left(\frac{x - n_{\text{bot}} + 1}{n_{\text{top}} - n_{\text{bot}} + 2}\right) \text{ where} \\ \kappa &= \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} f\left(\frac{i - n_{\text{bot}} + 1}{n_{\text{top}} - n_{\text{bot}} + 2}\right) \end{aligned} \tag{1}$$

$x = n_{\text{bot}}, n_{\text{bot}} + 1, \ldots, n_{\text{top}}$. In (1) $f(\cdot)$ is the probability density function (pdf) of the Beta distribution with first shape parameter equal to α and second shape parameter equal to β. The probabilities given by (1) are the values of the corresponding Beta density, evaluated at equispaced points in the interior of the interval $(0, 1)$, normalised to sum to 1. However it is possible, and advantageous, to express the definition of the db distribution in a direct manner without making reference to the Beta distribution.

Deriving the direct expression for the PMF of the db distribution from (1) is facilitated by noting that the Beta distribution is a member of the exponential family. From this it follows that the db distribution as defined by (1) is, for fixed values of the support parameters n_{top} and ζ, also a member. An expression for the PMF of the db distribution, in exponential family form, is derived from the pdf of the Beta distribution in **Appendix I**.

From this derivation it is seen that the PMF of the db distribution, given by (1), can also be expressed as

$$\Pr(X = x \mid \alpha, \beta, n_{\text{top}}, \zeta) = h(x) \exp\{\alpha T_1(x) + \beta T_2(x) - A(\alpha, \beta)\} \tag{2}$$

$x = n_{\text{bot}}, n_{\text{bot}} + 1, \ldots, n_{\text{top}}$, where

$$\begin{aligned} h(x) &= \frac{(n_{\text{top}} - n_{\text{bot}} + 2)^2}{(x - n_{\text{bot}} + 1)(n_{\text{top}} - x + 1)} \\ T_1(x) &= \log((x - n_{\text{bot}} + 1)/(n_{\text{top}} - n_{\text{bot}} + 2)) \\ T_2(x) &= \log((n_{\text{top}} - x + 1)/(n_{\text{top}} - n_{\text{bot}} + 2) \text{ and} \\ A(\alpha, \beta) &= \log\left(\sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) \exp\{\alpha T_1(i) + \beta T_2(i)\}\right) . \end{aligned}$$

We may consequently take the *definition* of the db distribution to be (2). This has the following advantage. The expression given by (2) is well-defined for *all* values of α and β: positive, negative or zero, whereas (1) is well-defined only for α and β strictly greater than zero. The difference is due to the fact that the pdf of the Beta distribution involves the expression $B(\alpha, \beta) = \Gamma(\alpha)\Gamma(\beta)/\Gamma(\alpha + \beta)$. This latter expression evaluates to ∞ (or `Inf` in R) if either α or β is less than or equal to zero. Consequently in such cases (1) is undefined (being equal to `Inf/Inf = NaN` in R). *Algebraically*, $B(\alpha, \beta)$ cancels from (1) due to the division by κ, whence it does not appear in (2).

Although the db distribution is well-defined for negative parameter values, there are indications of problems in respect of such values. In practice it may be advisable to restrict attention to positive values only. There certainly *exist* data sets — as can be demonstrated by simulation — for which the (maximum likelihood) estimates of the parameters are undeniably negative. Cursory experimentation using the **dbd** package indicates that in some instances negative parameters are reasonably well estimated by maximum likelihood. E.g.

```
library(dbd)
set.seed(42)
x  <- rdb(100,-2,-3,10)
fx <- mleDb(x,10)
print(as.vector(fx))
```

The preceding code produces estimates $\hat{\alpha} = -2.1645$ and $\hat{\beta} = -3.1365$. Ninety-five percent confidence intervals for α and β (based on the variance entries of the "analytic" covariance matrix — see section **Implementation**) are $[-3.0585, -1.2706]$ and $[-3.9443, -2.3297]$ which contain the true values, equal to -2 and -3 respectively.

On the other hand there are instances in which the maximum likelihood estimates are very large (either positive or negative) when the true values are relatively small and negative:

```
library(dbd)
set.seed(348)
x  <- rdb(100,alpha=-3,beta=-6,ntop=10,zeta=TRUE)
fx <- mleDb(x,ntop=10,zeta=TRUE,maxit=2000)
print(as.vector(fx))
```

The resulting estimates are $\hat{\alpha} = 73.18$ and $\hat{\beta} = -166.61$ which bear no relation to the "truth".

The (only?) real advantage in the fact that the parameter values are permitted to be negative lies in the avoidance of various numerical issues that might otherwise arise in the estimation of parameters of a db distribution. In particular, the legitimacy of negative parameter values removes the necessity of imposing box constraints on the procedure for maximising the likelihood. It also circumvents difficulties that can otherwise arise in evaluating the hessian of the log likelihood when one or both of the parameter estimates is close to zero.

Plots of the probability functions of a number of db distributions are shown in Figure 2. The shapes of these distributions mimic those of the corresponding Beta distributions.

Choosing n_{top}

In fitting a db distribution to an observed data set, it is often sensible to set n_{top} equal to the maximum of the data. On the other hand, if there is a conceptual least upper bound for the support of the distribution (not found amongst the observed values) then one should set n_{top} equal to this conceptual least upper bound. Finally, if the data are conceptually unbounded, then one might wish to set n_{top} equal to the maximum of the observed data $+1$. In this latter case the value of $\Pr(X = n_{\text{top}})$ might be interpreted as $\Pr(X \geq n_{\text{top}})$.

Implementation

I have written an R package **dbd** (Turner 2021) to provide tools for working with the db distribution. The package supplies the four standard "d", "p", "q" and "r" — density, probability, quantile and random number generation — functions that are as a rule associated in R with any given distribution. In this setting these functions are `ddb()`, `pdb()`, `qdb()` and `rdb()`. The package also contains functions `expValDb()` and `varDb()` to calculate the expected value (mean) and variance of a db distribution given a specification of its parameters. There is no closed form algebraic expression for these quantities.

Another useful tool in the package is a function `mleDb()` which enables the easy estimation from data, of the parameters of the db distribution, via maximum likelihood. The function `mleDb()` calls upon an undocumented function `meDb()` which calculates approximate moment estimates of the parameters to serve as starting values for the maximization of the likelihood. A user would not normally make direct use of `meDb()`. However the approximation used by `meDb()` is of interest in its own right. This approximation is discussed in the section **Estimation of parameters**.

The function `mleDb()` returns an object of class `"mleDb"`, and there is a corresponding `plot()` method to produce plots of the probability mass functions for distributions having parameters equal to the given estimates. There is also a stand-alone plotting function `plotDb()` which plots the PMF of a db distribution given specified parameters.

The package also has functions that provide means of estimating or calculating the covariance matrix of the parameter estimates. These functions enable assessment of the uncertainty in the estimates of the parameters. The functions are: `aHess()` ("analytic hessian"), `nHess()` ("numeric hessian"), `finfo()` and `mcCovMat()` (Monte Carlo based estimate of the covariance matrix). The first three functions provide matrices whose *inverses* are estimates of (or in the case of `finfo()` equal to) the desired covariance matrix, given the supplied parameter values. The function `nHess()` calls upon `optimHess()` from the **stats** package. The function `aHess()` makes use of the expressions set out in **Appendix II**.

The values produced by `aHess()` and `nHess()` generally appear to be in very good agreement. However if the parameter values are "unreasonably large" (e.g. $\alpha = 150$, $\beta = 400$) then there can be

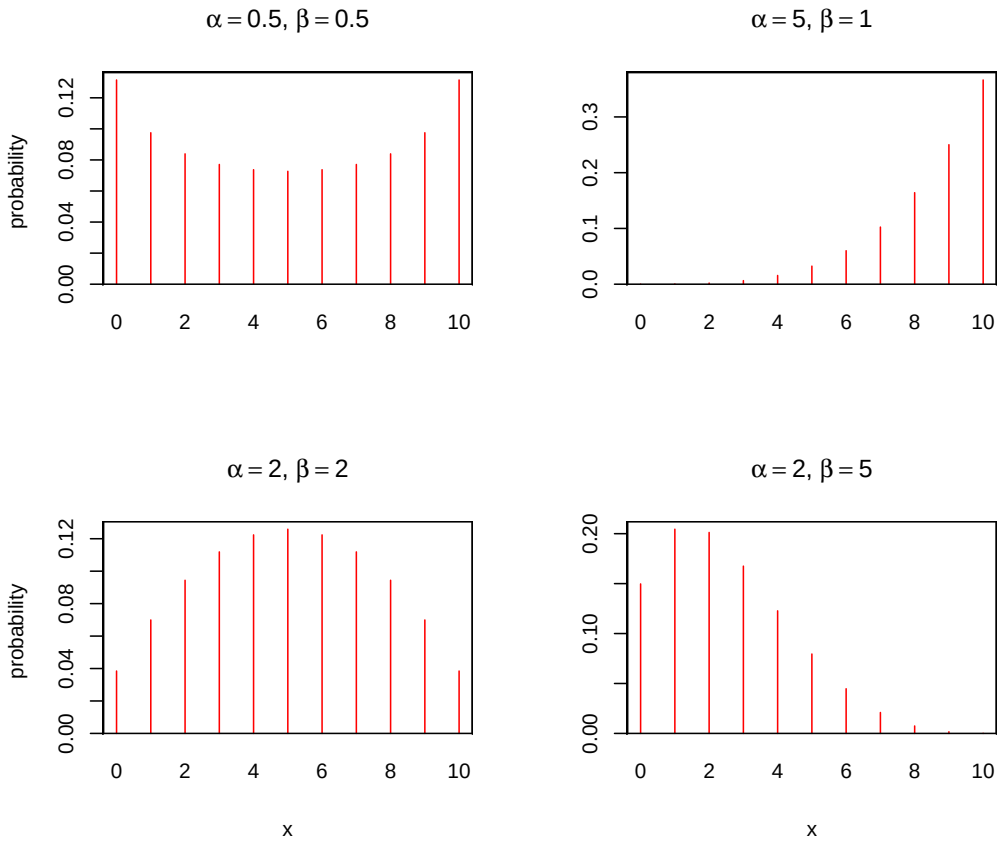

Figure 2: Probability mass functions of db distributions

substantial disparity between the values. In such instances it would be inadvisable to trust either result.

Of course the real reason for calculating the hessian is to obtain an estimate of the covariance matrix of the parameter estimates. Another way to obtain an estimated covariance matrix is to use the Monte Carlo methods conveniently provided by the function `mcCovMat()` referred to above. Again there is generally good agreement between the value produced by `mcCovMat()` and the inverse of the hessian produced, e.g., by `aHess()`. However, unless the sample size is quite large, the variance entries of the inverse hessian are substantially smaller than those of the matrix returned by `mcCovMat()`:

```
set.seed(25)
x  <- rdb(n=30,alpha=3,beta=4,ntop=10)
fx <- mleDb(x,ntop=10)
solve(aHess(fx))
          alpha     beta
alpha 0.7712032 1.047111
beta  1.0471108 1.790513
 mcCovMat(fx)
         alpha     beta
alpha 1.342672 1.889880
beta  1.889880 3.291645
```

Further discussion of this phenomenon is to be found in **Appendix III**.

The **dbd** package also contains a function `llPlot()` for plotting log likelihood surfaces and a function `gof()` for performing tests of goodness of fit for the db distribution. The `llPlot()` function may be useful in diagnosing problems with parameter estimation should these arise. The tests effected by `gof()` may be either chi-squared based tests or Monte Carlo tests. Users should be aware that Monte Carlo tests (which use a relatively small number of simulations) are *random*. Performing a Monte Carlo test is *not* the same as simulating a large number of test statistics (under the null hypothesis) in order to approximate the null distribution of the statistic. See for example Baddeley et al. (2015, Section 10.6, p. 384) for some discussion of such tests.

Estimation of parameters

There is, unsurprisingly, no closed form for any sort of estimates of the shape parameters of a db distribution. Estimates may however be calculated reasonably easily via (numerical) maximum likelihood. The function `mleDb()` from the **dbd** package, discussed in the section **Implementation** makes use of the `optim()` function, from the **stats** package (automatically available in R) to effect the maximisation.

The `optim()` function requires starting values for the parameters being estimated. It turns out that adequate starting values can be produced, as indicated in the section **Implementation**, via a (very!) rough approximation to the method of moments. To develop the approximation we need to go back to the conceptual definition of the db distribution expressed in terms of the Beta distribution (1). In terms of the conceptual definition, the mean and variance of a db distribution with shape parameters α and β may be written as

$$\mu = \frac{1}{\kappa}\sum_{i=0}^{n} i f((i+1)/(n+1))$$
$$\sigma^2 = \frac{1}{\kappa}\sum_{i=0}^{n} (i-\mu)^2 f((i+1)/(n+1))$$

where $f(\cdot)$ is the probability density function of the Beta distribution with shape parameters α and β, n is equal to n_{top} (the maximum value of the support of the distribution) and κ is the normalising constant

$$\kappa = \sum_{i=0}^{n} f((i+1)/(n+1)) \; .$$

In the foregoing, zero origin indexing (i.e. that ζ is `TRUE`) is assumed. This is of no real consequence given that the method is so rough to start with.

To get approximate expressions for the mean and variance of the distribution, one may manipulate the sums in the expressions for μ, σ^2 and κ into the form of Riemann sums that approximate integrals. This reveals that

$$\mu \approx \frac{(n+2)^2}{\kappa}\int_0^1 x f(x)\, dx - 1$$

and that

$$\kappa \approx (n+2)\int_0^1 f(x)\, dx \; .$$

The integral of $xf(x)$ is the mean of the associated Beta distribution, $\alpha/(\alpha+\beta)$ (Johnson et al. 1995, Chapter 25, equation 25.15a) and the integral of $f(x)$ is of course just 1, whence $\kappa \approx n+2$. Therefore

$$\mu \approx \frac{(n+2)\alpha}{\alpha+\beta} - 1 \; .$$

Proceeding similarly one finds that

$$\sigma^2 \approx \frac{(n+2)^2\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \; .$$

This latter result makes use of the fact that the variance of the associated Beta distribution is

$$\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

(Johnson et al. 1995, Chapter 25, equation 25.15b).

To calculate the approximate method of moments estimates, which I shall denote by $\tilde{\alpha}$ and $\tilde{\beta}$ respectively, equate the foregoing approximate expressions for μ and σ^2 to the observed sample mean and variance ($\bar{x}$ and s^2) and solve for α and β. One obtains

$$\tilde{\alpha} = \frac{(n+2)^2 a}{s^2(a+1)^3} - \frac{1}{a+1}$$
$$\tilde{\beta} = a\tilde{\alpha}$$

where for convenience I have set $a = (n+1-\bar{x})/(1+\bar{x})$.

Note that although the Beta distribution is undefined for non-positive values of α and β, the foregoing approximate moment estimation procedure can be applied without problem to data for which non-positive parameter estimates are appropriate.

It must be emphasised here that the moment estimation procedure is not intended to be applied by users. It exists solely for the purpose of providing starting values for the maximum likelihood procedure. The estimates produced via the moment estimation procedure are not very good, and in general appear to be substantially biased. Despite this they seem to be adequate as starting values.

The quality of the maximum likelihood estimates

I investigated the question of how well maximum likelihood estimation performs by means of simulation experiments. In these experiments 100 samples, each of size 100, were generated (using `rdb()`) from distributions

$$\mathrm{db}(\alpha_i, \beta_j, n_{\mathrm{top}} = 10, \zeta = \mathtt{TRUE})$$

with α_i and β_j varying over the set $\{0.5, 1.0, 1.5, 2.0, \ldots, 9.5, 10.0\}$. The maximum likelihood estimates of the underlying parameters appeared to be substantially biased. The bias was positive most of the time and sometimes assumed relatively large positive values but not large negative ones. The bias tended to grow larger as the true parameter values grew larger and the bias in $\hat{\alpha}$ tended to be larger when the true value of β was large and similarly (*mutatis mutandis*) for the bias in $\hat{\beta}$. The mean bias in the maximum likelihood estimates $\hat{\alpha}$ and $\hat{\beta}$ ranged between -0.09 and 2.38, and -0.16 and 2.07 respectively.

The bias in the maximum likelihood estimates does not seem to be induced by the use of the (biased) approximate moment estimates as starting values. In a simulation experiment it is possible to use the *true* values of the parameters as starting values. Doing so gave rise to maximum likelihood estimates that were virtually identical to those obtained when the approximate moment estimates were used to start the maximisation.

The bias discussed above arose in a setting in which the sample size was 100. The problem appears to diminish, albeit slowly, as the sample size increases. I undertook a further simulation experiment in which I set α and β both to have the "particularly bad" true value of 10 and in which the sample size was allowed to range over the set $\{100, 200, 300, 500, 1000, 5000\}$. One hundred simulated samples were generated in each instance. Notched boxplots of the resulting individual biases are shown in Figure 3. Note that there is substantial evidence of bias, in this instance, when the sample size takes the rather large value of 500. However it appears that the "asymptotically unbiased" property of maximum likelihood estimation does *eventually* take effect.

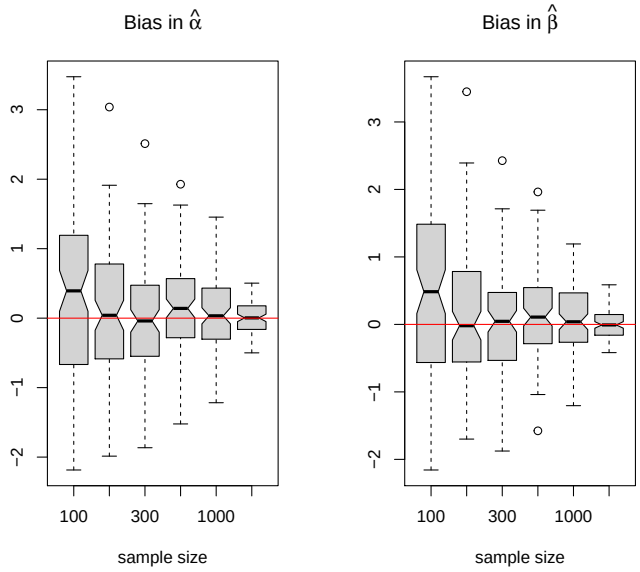

Figure 3: Notched boxplots of the bias in $\hat{\alpha}$ and $\hat{\beta}$. The horizontal red lines are at the zero level.

Table 1 gives sample mean values of estimates of α and β for a small grid of true values of these parameters. One hundred samples were generated for each combination of the true parameter values, and the means and standard errors of these estimates were calculated. Each sample was of size of 100. For each combination of the true parameters 95% confidence intervals (mean $\pm 1.96\times$ standard error) for the individual parameters were also calculated.

Half of the 18 confidence intervals failed to cover the true values. For the $(3, 3)$ combination, the

β interval failed to cover. For the $(3,6)$, $(6,6)$, $(9,3)$ and $(9,9)$ combinations, both intervals failed to cover. Where the confidence intervals failed to cover, they missed on high side, as one might expect on the basis of Figure 3. The amounts by which they missed were not egregiously large. The worst case was for the $(9,3)$ combination where the lower endpoint of the confidence interval for α was greater than 9 by 0.294.

Table 1: Some sample mean parameter estimates and 95% confidence intervals for the true values.

α	β	$\bar{\hat{\alpha}}$	95% CI for α	$\bar{\hat{\beta}}$	95% CI for β
3	3	3.075	(2.991, 3.159)	3.088	(3.003, 3.174)
3	6	3.161	(3.061, 3.261)	6.335	(6.146, 6.524)
3	9	3.029	(2.944, 3.115)	9.054	(8.793, 9.316)
6	3	6.184	(5.981, 6.387)	3.090	(2.990, 3.190)
6	6	6.228	(6.051, 6.404)	6.246	(6.076, 6.416)
6	9	6.078	(5.913, 6.242)	9.106	(8.858, 9.353)
9	3	9.597	(9.294, 9.899)	3.204	(3.103, 3.306)
9	6	9.145	(8.903, 9.388)	6.097	(5.936, 6.257)
9	9	9.334	(9.074, 9.595)	9.335	(9.073, 9.598)

Examples

Example 1: Simulated binomial data

For an initial example, I simulated 100 data from a binomial distribution $\text{Bin}(10, 0.25)$ and fitted a db distribution to that. I set ζ =TRUE and $n_{\text{top}} = 10$ (the "conceptual" upper bound). A plot of the resulting fit is shown in Figure 4. The fit is consistent with the other possible values of the probabilities of the various counts.

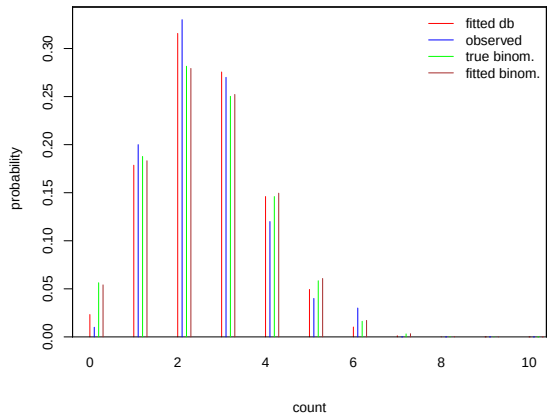

Figure 4: Fit of a db distribution to a sample from a $\text{Bin}(10, 0.25)$ distribution. Also shown are the observed proportions, the true binomial probabilities and the fitted binomial probabilities.

Example 2: The Downloads data

The Downloads data from Weiß (2018) consist of a time series (of length 267) of the observed daily number of downloads of a TEX editor for the period from June 2006 to February 2007. These data are available as `Downloads` in the CRAN package **hmm.discnp** They can also be obtained from the Wiley web site `https://www.wiley.com/en-gb/An+Introduction+to+Discrete+Valued+Time+Series-p-9781119096962` by clicking on "Downloads" and then on the "Download" button next to "Datasets". This will provide a zip archive of all of the data sets from Weiß's book.

Prima facie it might seem plausible that these data are Poisson distributed, but they are in fact much too overdispersed to be Poisson; the sample mean is 2.401 whereas the sample variance is 7.506. Weiß

finds that an INAR(1) (integer valued autoregressive, $p = 1$) model provides a good fit. In fitting a db distribution to these data I took $\zeta =$ `TRUE` (zero counts are observed) and $ntop = 15$ (1+ the observed maximum of the data which is 14). This db fit yields a mean of 2.451 and a variance of 7.461. A plot of this fit is shown in Figure 5. Of course simply fitting a db distribution is not really appropriate, since this treats the data as being i.i.d., and as Weiß's analysis shows, there is strong evidence of serial dependence in these data. Moreover a goodness of fit test yields a p-value of 0.03 (see the help for `gof()` in the **dbd** package). We would thereby reject the hypothesis that the fit of the db distribution is adequate, at the 0.05 significance level.

In other analyses of these data, the details of which it is inexpedient to discuss here, I have fitted hidden Markov models with marginal db distributions and varying numbers of states to these data. I used both AIC and BIC to select the number of states, and both criteria indicate that a two state model is optimal, i.e. a model involving serial dependence is chosen over the model in which the data are i.i.d.

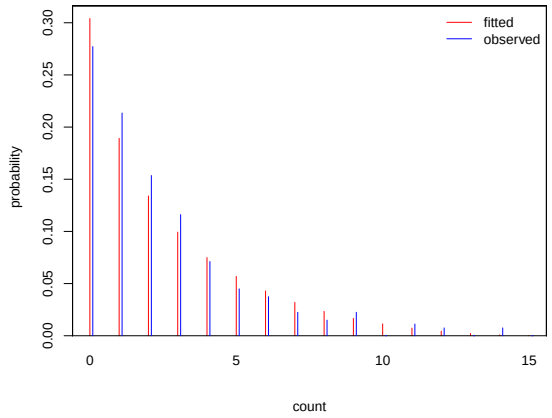

Figure 5: Fit of a db distribution to the Downloads data from Weiß (2018). Also shown are the observed proportions.

Example 3: The Sydney Coliform Count data

These data were analysed in Turner et al. (1998). In that paper the data were modelled, after a certain transformation had been applied, as having a hidden Markov model structure with marginal Poisson distributions. I have since discretised these data into five (ordered) categories. The resulting data set is available as `SycColDisc` in the CRAN package **hmm.discnp**. I fitted db distributions to subsets of these data, treating them as having the numeric values $1, 2, \ldots, 5$, taking $n_{\text{top}} = 5$ and $\zeta =$ `FALSE`.

Plots of the fits, together with the observed proportions, are shown in Figure 6, for the Bondi East data at each of the four depths, 0, 20, 40 and 60 metres.

Example 4: The Sydney Coliform Count data (continued)

In general it may be of interest to provide graphical representations of the uncertainty in the estimates of the parameters of db distributions. This can be done by plotting (say) 95% confidence ellipses around the point estimates. The **ellipse** package (Murdoch and Chow (2018)) provides convenient means of plotting such ellipses.

In the context of the current example it is also of interest to examine whether there are differences amongst the distributions associated with the various depth and location combinations. Such examination can also be effected by means of plotting confidence ellipses. In this setting one would plot confidence ellipses for the differences between the parameters corresponding to different combinations. Assuming that the samples are independent (possibly problematic here) the covariance matrix on which to base the confidence ellipse for the difference between the parameters is the sum of the two individual covariance matrices.

We can also test the hypothesis that the distributions corresponding to a number of different combinations of depths and locations are all identical (again assuming the samples to be independent)

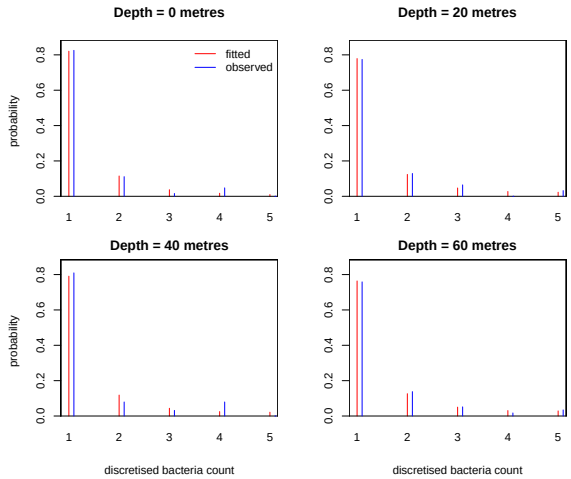

Figure 6: Fits of db distributions to discretised Sydney coliform count data from the Bondi East location, at depths 0, 20, 40 and 60 metres. Also shown are the observed proportions.

by means of a likelihood ratio test. We illustrate the foregoing ideas using the four different depths at the "BondiE" location. Confidence ellipses for the parameters corresponding to the four depths are shown in Figure 7. Confidence ellipses for the six pairwise differences are shown in Figure 8. Code for effecting the likelihood ratio test is as follows:

```
library(dbd)
X       <- hmm.discnp::SydColDisc
X$y     <- as.numeric(X$y)
X       <- split(X,f=with(X,interaction(locn,depth)))
X       <- X[c("BondiE.0","BondiE.20","BondiE.40","BondiE.60")]
fitz    <- lapply(X,function(x){mleDb(x$y,ntop=5)})
x.all   <- unlist(lapply(X,function(x){x$y}))
fit.all <- mleDb(x.all,ntop=5)
ll0     <- logLik(fit.all) # Two parameters.
ll1     <- sum(sapply(fitz,logLik)) # Eight parameters.
print(pchisq(2*(ll1-ll0),6,lower=FALSE)) # Df = 8 - 2.
```

The resulting p-value is 0.9781; i.e. there is no evidence at all of any differences. This conclusion is confirmed by Figure 8 wherein we see that the 95% confidence ellipses all contain the point $(0,0)$. It is also in accordance with the visual impression given by Figure 6 in which the plots of the four distributions all look very similar.

An analogous exercise was done involving measurements all made at a depth of 60 metres, at four of the seven locations (Longreef, Bondi East, Malabar Offshore and North Head Offshore; two "controls" and two "outfalls"). The likelihood ratio test yielded a p-value equal to 6.37×10^{-9}, i.e. effectively zero. The origin $(0,0)$ was exterior to the 95% confidence ellipses for the pairwise differences in four of the six instances,

Note that the foregoing analyses of the Sydney Coliform data are superficial in that they take no account of the serial dependence of these data. Undertaking analyses that accommodate serial dependence would lead us much too far afield.

Example 5: Monocyte counts and psychosis ratings

The monocyte counts and psychosis ratings data arose in a study initiated by Jonathan Williams, who was at the start of the study working for the Northland District Health Board in New Zealand. The study is still ongoing, and the results have not yet been published. The data involved cannot be publicly released due to patient confidentiality issues.

This example is really what motivated the development of the db distribution. The data consist of pairs of sequences of observations of monocyte blood counts, discretised to a 1 to 5 scale, and ratings of severity of psychosis on a 0 to 4 scale. The observations were made on 1258 patients. They were made at irregularly spaced times and there were varying numbers of observations per patient. The different types of sequence were not observed at the same times and there were usually different numbers of observation between the types.

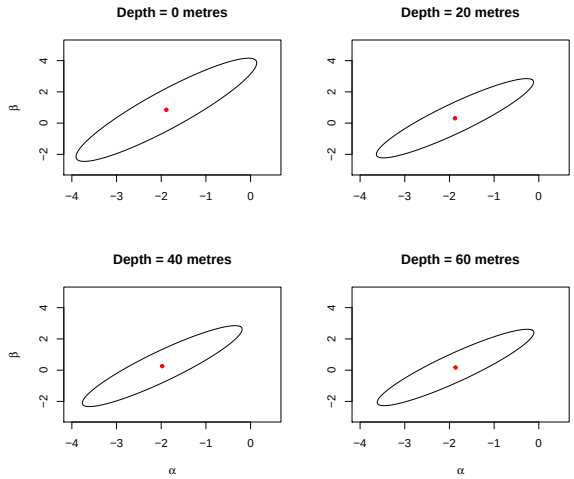

Figure 7: 95% confidence ellipses for the parameters of the db distributions fitted to the data from the Bondi East location, at depths 0, 20, 40 and 60 metres.

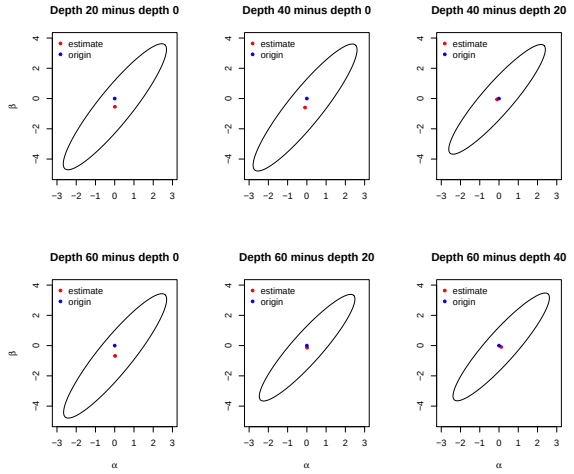

Figure 8: 95% confidence ellipses for pairwise differences between the parameter vectors of the db distributions fitted to the data from the Bondi East location, at depths 0, 20, 40 and 60 metres.

The analysis of these data was intricate and the details cannot be gone into here. The crucial feature of the analysis is that hidden Markov models, whose marginal distributions were db distributions were fitted to the both types of sequence. The value of n_{top} was taken to be 5 for the monocyte counts and 4 for the psychosis ratings, and that of ζ to be `FALSE` for the monocyte counts and `TRUE` for the psychosis ratings.

For each type a three state model was chosen (by means of a cross validation technique). Plots of the db distributions corresponding to each of the three states are shown for the monocyte counts in Figure 9 and for the psychosis ratings in Figure 10.

Example 6: Underdispersed data

As mentioned in the introduction, the fit of db distributions to underdispersed data is less problematic than the fit of beta-binomial distributions. Here are some examples to illustrate fitting the db distribution to underdispersed data.

The horse race prediction data from the **dbd** package (see the help for `hrsRcePred`) provides four such examples.

```
library(dbd)
X     <- hrsRcePred
top1e <- X[X$sbjType=="Expert","top1"]
top1n <- X[X$sbjType=="NonXpert","top1"]
top3e <- X[X$sbjType=="Expert","top3"]
top3n <- X[X$sbjType=="NonXpert","top3"]
fit1e <- mleDb(top1e,ntop=10,zeta=TRUE)
fit1n <- mleDb(top1n,ntop=10,zeta=TRUE)
fit3e <- mleDb(top3e,ntop=10,zeta=TRUE)
fit3n <- mleDb(top3n,ntop=10,zeta=TRUE)
```

The four fitting procedures all proceeded without complaint. Moreover goodness of fit tests indicate that all four fits are adequate:

```
set.seed(42) # To get repeatable Monte Carlo p-values.
pv1e <- gof(fit1e,obsd=top1e,MC=TRUE)$pval) # 0.72
pv1n <- gof(fit1n,obsd=top1n,MC=TRUE)$pval) # 0.75
pv3e <- gof(fit3e,obsd=top3e,MC=TRUE)$pval) # 0.37
pv3n <- gof(fit3n,obsd=top3n,MC=TRUE)$pval) # 0.38
```

The p-values are all large, indicating that there is no evidence of any problems with any of the fits. However the parameter estimates are inordinately large. The worst-case example is `fit1e`:

```
    alpha      beta
145.14659  21.23249
```

This may be indicative of problems.

Another two examples are provided by the visual recognition data from the **dbd** package (see the help for `visRecog`).

```
library(dbd)
fit5  <- with(visRecog,mleDb(tot5,20,TRUE))
fit10 <- with(visRecog,mleDb(tot10,20,TRUE))
set.seed(42) # To get repeatable Monte Carlo p-values.
pv5   <- gof(fit5,obsd=visRecog[["tot5"]],MC=TRUE)$pval   # 0.86
pv10  <- gof(fit10,obsd=visRecog[["tot10"]],MC=TRUE)$pval # 0.68
```

Again the p-values are large, indicating no problem with the fits but the parameter estimates are again somewhat larger than what might be considered "reasonable".

Concluding remarks

The db distribution is a new distribution which can be applied to any sort of data which takes values in a finite discrete set. It is an ad hoc distribution and does not require any theoretical justification in terms of properties that the data may have. It is very flexible, with the restriction that (as remarked in the **Introduction**) it is effectively unimodal. The values of the distribution are integers varying from 0 to n or from 1 to n for some n, and data to which a db distribution is to be fitted must be converted (recoded) into that form. In order that the fit should make practical sense, the data should, generally speaking, bear some relation to counts, or at least be ordered. However there is no theoretical requirement that this should be the case.

A number of examples have been given in this paper illustrating the fit of the db distribution to different data sets. These examples show that the db distribution may reasonably be expected to be useful to data analysts who need to deal with discrete data that do not conform to one of the standard distributions.

Appendix I

Here we derive, from the conceptual definition (1) of the db distribution, an expression for the PMF of this distribution which, for fixed values of the support parameters n_{top} and ζ, is in exponential family form. A distribution is in the exponential family if its probability density or mass function has

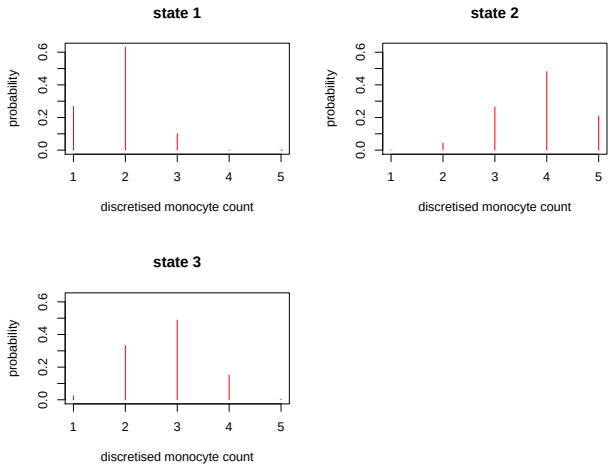

Figure 9: Marginal db distributions corresponding to each of the three states of a hidden Markov model fitted to the monocyte count data.

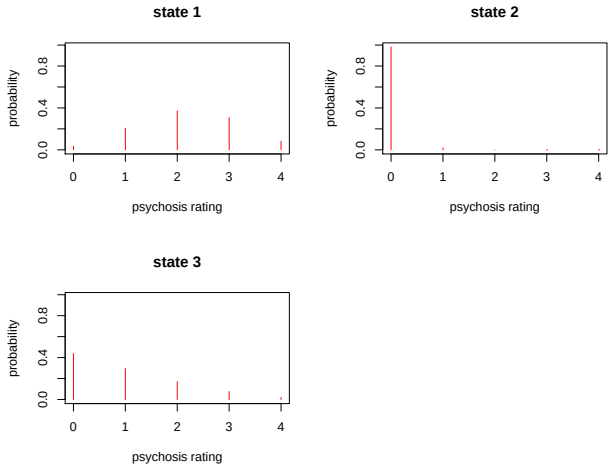

Figure 10: Marginal db distributions corresponding to each of the three states of a hidden Markov model fitted to the psychosis rating data.

a particular structure. Different authors and books express this structure in a variety of equivalent ways. (See e.g. Cox and Hinkley 1974, p.94, Davidson 2003, p. 168, Hogg et al. 2005, p. 400. Liero and Zwanzig 2012, p. 15, Abramovich and Ritov 2013, p. 13. The reader may also find it useful to access `https://en.wikipedia.org/wiki/Exponential_family`.) Almost all of the commonly used distributions (with the notable exception of the uniform distribution) are in the exponential family.

A suitable expression for the exponential family form of a probability density or mass function, of a (scalar) distribution depending on a parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)^\top$, is

$$f(x \mid \boldsymbol{\theta}) = h(x) \exp\left(\sum_{i=1}^{k} \eta_i(\boldsymbol{\theta}) T_i(x) - A(\boldsymbol{\theta}) \right) .$$

The "natural parameters" of the distribution are the $\eta_i(\boldsymbol{\theta})$.

The pdf of the Beta distribution can be written in exponential family form (with natural parameters equal to α and β) as

$$f(x \mid \alpha, \beta) = h_{\mathrm{B}}(x) \exp\{\alpha \log(x) + \beta \log(1 - x) - \log B(\alpha, \beta)\}$$

where $h_{\mathrm{B}}(x) = (x(1-x))^{-1}$ and $B(\alpha, \beta)$ is the beta function, equal to $\Gamma(\alpha)\Gamma(\beta)/\Gamma(\alpha + \beta)$ (where in turn $\Gamma(\cdot)$ is the gamma function).

The PMF of the db distribution, expressed in terms of the Beta distribution (1) is

$$\Pr(X = x \mid \alpha, \beta) = \frac{1}{\kappa} f\left(\frac{x - n_{\text{bot}} + 1}{n_{\text{top}} - n_{\text{bot}} + 2}\right)$$

where $f(\cdot)$ is the pdf of the Beta distribution and

$$\kappa = \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} f\left(\frac{i - n_{\text{bot}} + 1}{n_{\text{top}} - n_{\text{bot}} + 2}\right) .$$

Hence if we set $h(x) = h_{\text{B}}((x - n_{\text{bot}} + 1)/(n_{\text{top}} - n_{\text{bot}} + 2))$, $T_1(x) = \log((x - n_{\text{bot}} + 1)/(n_{\text{top}} - n_{\text{bot}} + 2))$ and $T_2(x) = \log((n_{\text{top}} - x + 1)/(n_{\text{top}} - n_{\text{bot}} + 2)$, then the PMF of the db distribution can be written as

$$\Pr(X = x \mid \alpha, \beta) = h(x) \exp\left\{\alpha T_1(x) + \beta T_2(x) - \log\left[\sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) \exp\{\alpha T_1(i) + \beta T_2(i)\}\right]\right\}$$

Note that the $\log B(\alpha, \beta)$ terms, present in $f(\cdot)$, cancel when

$$f\left(\frac{x - n_{\text{bot}} + 1}{n_{\text{top}} - n_{\text{bot}} + 2}\right)$$

is divided by κ.

The foregoing expression for the PMF is equal to

$$\Pr(X = x \mid \alpha, \beta) = h(x) \exp\{\alpha T_1(x) + \beta T_2(x) - A(\alpha, \beta)\}$$

where

$$A(\alpha, \beta) = \log\left(\sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) \exp\{\alpha T_1(i) + \beta T_2(i)\}\right) .$$

This is the expression given in (2) and is of exponential family form, although the constant $A(\alpha, \beta)$ might appear to be somewhat unorthodox.

Obviously the value of $A(\alpha, \beta)$ is such that the values of $\Pr(X = i \mid \alpha, \beta)$, $i = n_{\text{bot}}, \ldots, n_{\text{top}}$, sum to 1.

Appendix II

When the PMF of a db distribution is expressed in the form (2), it is a relatively simple matter to derive an analytic expression for the gradient of the log likelihood. Such an expression can be passed to `optim()` obviating the need for approximating the gradient numerically, via finite differencing. The derivation of an analytic expression for the hessian is equally easy. The `optim()` function makes no provision for using an analytically calculated hessian, but the availability of such an expression permits the calculation or estimation of the covariance matrix of the parameter estimates in an analytic manner. The derivations of the expressions for the gradient and hessian are as follows.

The log likelihood is

$$\begin{aligned} \ell &= \log \Pr(X = x \mid \alpha, \beta, n_{\text{top}}, \zeta) \\ &= \log h(x) + \alpha T_1(x) + \beta T_2(x) - A(\alpha, \beta) . \end{aligned}$$

Consequently the gradient is given by

$$\frac{\partial \ell}{\partial \alpha} = T_1(x) - \frac{\partial A}{\partial \alpha} \quad \frac{\partial \ell}{\partial \beta} = T_2(x) - \frac{\partial A}{\partial \beta} .$$

The hessian is given by

$$\frac{\partial^2 \ell}{\partial \alpha^2} = -\frac{\partial^2 A}{\partial \alpha^2} \quad \frac{\partial^2 \ell}{\partial \alpha \partial \beta} = -\frac{\partial^2 A}{\partial \alpha \partial \beta}$$

$$\frac{\partial^2 \ell}{\partial \beta^2} = -\frac{\partial^2 A}{\partial \beta^2}$$

Now let

$$E = \exp(A) = \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) \exp\{\alpha T_1(i) + \beta T_2(i)\} .$$

Clearly

$$\frac{\partial A}{\partial \alpha} = \frac{1}{E}\frac{\partial E}{\partial \alpha} \quad \frac{\partial A}{\partial \beta} = \frac{1}{E}\frac{\partial E}{\partial \beta}$$

$$\frac{\partial^2 A}{\partial \alpha^2} = \frac{1}{E}\frac{\partial^2 E}{\partial \alpha^2} - \frac{1}{E^2}\left(\frac{\partial E}{\partial \alpha}\right)^2 \quad \frac{\partial^2 A}{\partial \alpha \partial \beta} = \frac{1}{E}\frac{\partial^2 E}{\partial \alpha \partial \beta} - \frac{1}{E^2}\left(\frac{\partial E}{\partial \alpha}\frac{\partial E}{\partial \beta}\right)$$

$$\frac{\partial^2 A}{\partial \beta^2} = \frac{1}{E}\frac{\partial^2 E}{\partial \beta^2} - \frac{1}{E^2}\left(\frac{\partial E}{\partial \beta}\right)^2 .$$

It remains to calculate the relevant partial derivatives of E. These are given by:

$$\frac{\partial E}{\partial \alpha} = \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) T_1(i) \exp(\alpha T_1(i) + \beta T_2(i))$$

$$\frac{\partial E}{\partial \beta} = \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) T_2(i) \exp(\alpha T_1(i) + \beta T_2(i))$$

$$\frac{\partial^2 E}{\partial \alpha^2} = \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) T_1(i)^2 \exp(\alpha T_1(i) + \beta T_2(i))$$

$$\frac{\partial^2 E}{\partial \alpha \partial \beta} = \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) T_1(i) T_2(i) \exp(\alpha T_1(i) + \beta T_2(i))$$

$$\frac{\partial^2 E}{\partial \beta^2} = \sum_{i=n_{\text{bot}}}^{n_{\text{top}}} h(i) T_2(i)^2 \exp(\alpha T_1(i) + \beta T_2(i)) .$$

The foregoing calculations have been translated into R code in the (undocumented) functions `grad()` and `hess()` in the **dbd** package.

Appendix III

Covariance matrices of the estimates of the parameters of a db distribution may calculated both by theoretical means (using e.g. `aHess()` from the **dbd** package) or by a Monte Carlo procedure using the `mcCovMat()` function from the same package. Doing such calculations in a number of examples has indicated that there can be substantial discrepancies between the theoretical and Monte Carlo results. To investigate this issue further I calculated the variance of $\hat{\beta}$ from *known* (rather than estimated) parameters, using both the theoretical (inversion of the Fisher information matrix) and Monte Carlo procedures. The essential part of the code used to do this is:

```
obj       <- makeDbdpars(alpha=3,beta=3,ntop=10,zeta=TRUE,ndata=<some value>)
varBeta.mc <- mcCovMat(obj,nsim=500)[2,2]
varBeta.fi <- solve(do.call(finfo,obj))[2,2]
```

I effected the calculations for a range of sample sizes ("ndata"). The results are plotted in Figure 11.

The behaviour depicted in Figure 11 is typical. The theoretical covariance matrices for the parameter estimates generally include variance entries which (for relatively small sample sizes) are appreciably smaller than the corresponding entries of the covariance matrices produced by Monte Carlo methods. Since the Monte Carlo covariance matrices are unbiased estimates of the true covariances, it would appear that the theoretical variances tend to underestimate the truth. This phenomenon is not peculiar to the db distribution. Such underestimation occurs in the context of the Beta distribution and very likely in other contexts as well. As illustrated by Figure 11, the level of underestimation (as would be expected) diminishes as the sample size increases. In the illustrated instance, the underestimation disappeared when the sample size reached 200.

The fact that variances are underestimated by the theoretical covariance estimates implies that inference about the shape parameters based on the theoretical values should be treated with a certain amount of circumspection. Unless the sample size is large confidence intervals may be somewhat too narrow, and hypothesis tests too liberal. It is advisable, when conducting inference about the shape parameters, to estimate the covariance matrix using `mcCovMat()`. The procedure is not too computationally demanding and is thus reasonably quick in normal circumstances. It is probably a good idea, in circumstances in which inferential conclusions are critical, to calculate a number of Monte Carlo

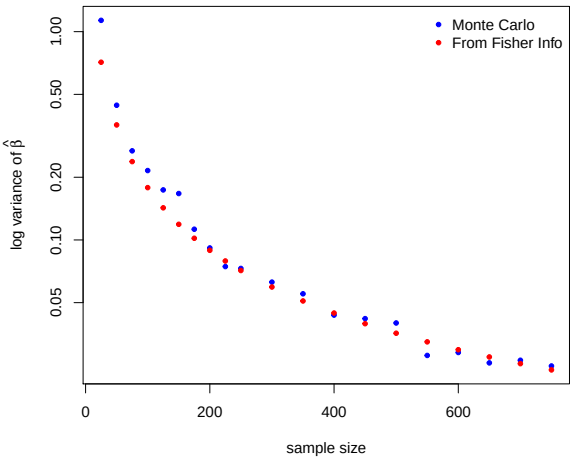

Figure 11: Monte Carlo and analytic estimates of the log variance of $\hat{\beta}$ for various sample sizes.

covariance matrix estimates (using different seeds) and to compare these with each other and with the "analytic" value of the covariance matrix.

The difference between results from using an "analytic" covariance matrix and those from using a Monte Carlo covariance matrix is also illustrated in Figure 12. The examples used are from the Bondi East Sydney Coliform Count data. (See Figures 7 and 8.) The chosen examples evince the most striking difference between the confidence ellipses based on the two different calculation methods.

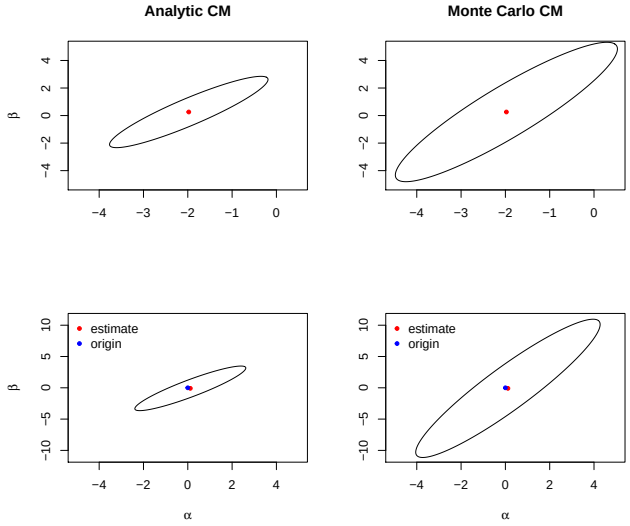

Figure 12: The top two panels show 95% confidence ellipses for the parameters of the db distribution for the Bondi East data at depth 40 metres. The left hand panel is based on the "analytic" covariance matrix, the right hand panel on a Monte Carlo covariance matrix. The bottom two panels show 95% confidence ellipses for the difference in parameters between depth 60 metres and depth 40 metres. Again the left hand panel is based on the "analytic" covariance matrix and the right hand panel on a Monte Carlo covariance matrix. The inferential conclusions with respect to the differences are unchanged.

Acknowledgements

The author would like to thank an anonymous referee for useful comments which led to substantial improvements in the paper.

The author also wishes to acknowledge the use of the New Zealand eScience Infrastructure (NeSI) high performance computing facilities, which made possible the analysis of the monocyte counts and psychosis ratings discussed in **Example 5**. New Zealand's national facilities are provided by NeSI and funded jointly by NeSI's collaborator institutions and through the Ministry of Business, Innovation & Employment's Research Infrastructure programme. URL https://www.nesi.org.nz.

Bibliography

F. Abramovich and Y. Ritov. *Statistical Theory: A Concise Introduction*. CRC Press, Boca Raton, 2013. [p13]

M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions*. Dover, New York, 1972. [p1]

A. Baddeley, E. Rubak, and R. Turner. *Spatial Point Patterns: Methodology and Applications with R*. Chapman and Hall/CRC Press, London, 2015. [p5]

D. R. Cox and D. V. Hinkley. *Theoretical Statistics*. Chapman and Hall, London, 1974. [p13]

A. C. Davidson. *Statistical Models*. Cambridge University Press, Cambridge, 2003. [p13]

R. V. Hogg, J. W. McKean, and A. T. Craig. *Introduction to Mathematical Statistics*. Pearson/Prentice Hall, Upper Saddle River, New Jersey, 2005. [p13]

N. L. Johnson, S. Kotz, and N. Balakrishnan. *Continuous Univariate Distributions*, volume 2. Wiley, New York, 1995. [p1, 6]

H. Liero and S. Zwanzig. *Introduction to the Theory of Statistical Inference*. CRC Press, Boca Raton, 2012. [p13]

D. Murdoch and E. D. Chow. *ellipse: Functions for Drawing Ellipses and Ellipse-Like Confidence Regions*, 2018. URL https://CRAN.R-project.org/package=ellipse. R package version 0.4.1. [p9]

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020. URL https://www.R-project.org/. [p1]

B. Swihart and J. Lindsey. *rmutil: Utilities for Nonlinear Regression and Repeated Measurements Models*, 2020. URL https://CRAN.R-project.org/package=rmutil. R package version 1.1.4. [p2]

R. Turner. *hmm.discnp: Hidden Markov models with discrete non-parametric observation distributions*, 2020. R package version 3.0-6. [p1]

R. Turner. *The db Distribution*, 2021. R package version 0.0-26. [p4]

T. R. Turner, M. A. Cameron, and P. J. Thomson. Hidden Markov chains in generalized linear models. *Canadian Journal of Statistics*, 26:107 – 125, 1998. [p9]

C. H. Weiß. *An Introduction to Discrete-Valued Time Series*. John Wiley & Sons, Chichester, 2018. [p8, 9]

Rolf Turner
Honorary Research Fellow
Department of Statistics
The University of Auckland
New Zealand
ORCID: 0000-0001-5521-5218
r.turner@auckland.ac.nz