Multilevel Fixed and Sequential Acceptance Sampling: The R Package MFSAS

Aaron Childs
McMaster University

Yalin Chen
B.C. Centre for Excellence in HIV/AIDS

Abstract

This introduction to the R package **MFSAS** is a (slightly) modified version of Childs and Chen (2011), published in the *Journal of Statistical Software*.

Multilevel acceptance sampling for attributes is used to decide whether a lot from an incoming shipment or outgoing production is accepted or rejected when the product has multiple levels of product quality or multiple types of (mutually exclusive) possible defects. This paper describes a package which provides the tools to create, evaluate, plot, and display the acceptance sampling plans for such lots for both fixed and sequential sampling. The functions for calculating cumulative probabilities for several common multivariate distributions (which are needed in the package) are provided as well.

Keywords: multilevel quality control, multinomial distribution, multivariate hypergeometric distribution, negative multinomial distribution, negative multivariate hypergeometric distribution.

Acceptance sampling is used to decide whether a lot from a manufacturer should be accepted or rejected by making an inference about the lot quality based on a sample. The product quality can be described by classifying a single characteristic of the product using more than two discrete levels in many circumstances. For example, a food product could be classified as good, marginal, or bad, depending on the concentration of harmful microorganisms in the product. In MIL-STD-105E (1989) the U.S. Department of Defense establishes military standards for sampling plans in which products are classified as critical defective, major defective, minor defective, or nondefective. Other examples of three or more classifications can be found in Radhakrishnan and Sankar (2009), Cassady and Nachlas (2003), Bray, Lyon, and Burr (1973), Newcombe and Allen (1988), Thatcher and Clarke (1978), and Shapiro and Zaheda (1990).

In this paper we describe a new R (R Development Core Team 2011) package for multilevel acceptance sampling in which two types of sampling plans are provided. The first type is `"fixed"` sampling, in which a sample of size n is selected from the lot for inspection. After inspection, the consumer will either accept or reject the lot, depending on the number of defectives of each type that are found in the sample. Let k (≥ 2) denote the number of different levels of product quality, $k-1$ of which are considered to be defective items, and one of which consists of nondefective or good items. Let X_i be the number of defectives of type i in the sample, for $i = 1, 2, \ldots, k-1$, and let X_k denote the number of good items. A fixed sampling plan requires a sample size n and rejection numbers r_i with the property that the lot will be rejected if $X_i \geq r_i$ for any i in $\{1, 2, \ldots, k-1\}$. That is, the sample is rejected if there are too many of any one type of defect.

The second type of plan that we provide is for `"sequential"` sampling, in which items in the lot are selected one at a time for inspection. A sequential sampling plan requires a cell quota m for the good items and cell quotas r_i for the each of the defective items, for $i = 1, 2, \ldots, k-1$. Sampling continues until either the number of good items or any of the $k-1$ types of defectives reaches its respective quota. If the former occurs first, then the lot is accepted; otherwise it is rejected.

Sequential sampling has the advantage that it will require, on average, less items to be inspected than for fixed sampling. This will be advantageous if the testing is expensive, or if the units are destroyed in the testing. Fixed sampling, however, allows for units in the sample to be tested simultaneously, whereas in sequential sampling the items have to be tested one at a time. Therefore sequential sampling will not be as practical if the testing of items is time consuming.

For each type of sampling plan we provide the option for the sampling to be *with* or *without* replacement. These sampling modes are determined by specifying the type of distribution, which can be either `multinomial` (for sampling with replacement) or `hypergeom` (for sampling without replacement). When sampling is without replacement, the lot size N also needs to be specified.

In addition, all plans require a matrix *pd* whose rows contain the proportions of each of the $k-1$ types of defectives in the lot. Thus the rows of the matrix *pd* determine different levels of *lot quality*.

Once a sampling plan has been determined, the probability *pa* of accepting lots can be calculated for various levels of lot quality to obtain the *operating characteristic* (OC) function. For 2-level or 3-level acceptance sampling plans the OC function can then be plotted as a curve or surface, respectively. Two particular levels of lot quality that are used to assess a sampling plan are the *producer's quality level* (PQL), which consists of proportions of each type of defective considered by the producer to be acceptable, and the *consumer's quality level* (CQL), consisting of proportions of each type of defective considered by the consumer to be unacceptable. Since the decision to accept or reject the lot is based only on a sample, there is always the possibility that an incorrect decision is made. This would occur if a lot which has the producer's quality level is rejected by the plan, or if a lot which has the consumer's quality level is accepted by the plan. Then the probability of an incorrect decision is either

- The *producer's risk* α, which is the probability that a lot which has the producer's quality level is rejected by the plan,

or

- The *consumer's risk* β, which is the probability that a lot which has the consumer's quality level is accepted by the plan.

In other words, the producer's risk α is the probability that a batch of high quality is rejected, and the consumer's risk β is the probability that a batch of low quality is accepted. These two risks are analogous to the Type-I and Type-II errors in hypothesis testing.

For each of the four possible combinations of sampling type (fixed and sequential) and sampling mode (with and without replacement) this package allows the user to create sampling plans that have pre-specified maximum values for α and β, given values for the PQL and

CQL. The user can also evaluate whether a given sampling plan meets specified maximum values for α and β, to plot the OC curve or surface for 2-level or 3-level plans, and to display acceptance probabilities for any given levels of lot quality.

A related R package called **AcceptanceSampling** was developed by Kiermeier (2008) and can be used when the sample size is fixed and there are $k = 2$ levels of product quality. The **MFSAS** package described here extends the functionality of the **AcceptanceSampling** package by allowing for more than two levels of product quality, and by allowing the sampling type to be sequential, in addition to fixed. However, fixed sampling in the **MFSAS** package is restricted to single stage sampling for attributes, whereas the **AcceptanceSampling** package allows the sampling to be multi-stage (in which the decision at each stage of sampling can be to accept the lot, reject the lot, or take an additional sample of items) and provides functionality for sampling inspection by variables, in addition to attributes. Further details about the **AcceptanceSampling** package, as well as some background on acceptance sampling in general can be found in Kiermeier (2008). For an up-to-date and complete reference on all aspects of acceptance sampling the reader is referred to Schilling and Neubauer (2009).

1. The MFSAS package

The **MFSAS** package is based on formal S4 classes and methods. It provides functionality for creating, evaluating, and plotting k-level acceptance sampling plans for attributes according to different distributional assumptions.

1.1. Object classes

The package consists of a virtual class, `Ocmult`. In this class the two parameters are `type` (the type of distribution) and `stype` (the type of sampling). The distributions that can be specified for `type` are

- `multinomial` which is used if the number of items in the lot is assumed to be large relative to the sample size, or if the sample is taken with replacement.
- `hypergeom` which is used when the lot is finite and the sample is taken without replacement.

The types of sampling that can be specified with `stype` are

- `fixed` for fixed sampling, in which a sample of size n is selected from the lot. For this type of sampling, calculations are based on either the multinomial or multivariate hypergeometric distribution, depending on the value specified for `type`.
- `sequential` for sequential sampling, in which items are selected one at a time for inspection. Here calculations are based on either the negative multinomial or negative multivariate hypergeometric distribution, depending on the value specified for `type` (which is still either `multinomial` or `hypergeom`).

The two actual classes, `Ocmult.multinomial` and `Ocmult.hypergeom` are derived from `Ocmult` (see Figure 1).

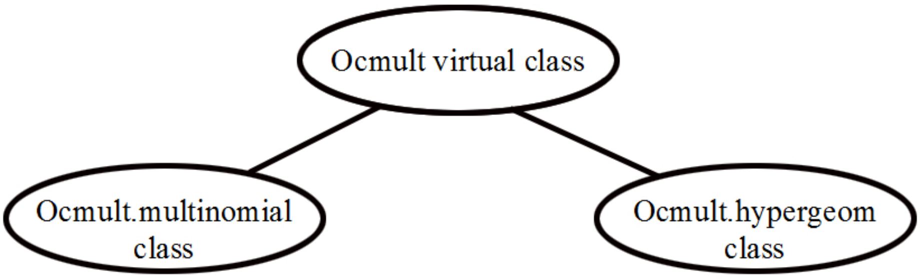

Figure 1: Class structure.

Both classes contain the `Ocmult` virtual class, hence its slots. Objects of the two classes can be generated by the `Ocmult` function which takes the following arguments.

`rn`: A vector of length $k-1$ consisting of rejection numbers for fixed sampling, or cell quotas for the defective items for sequential sampling.

`pd`: A matrix with $k-1$ columns, whose rows contain the proportions of each type of defective in the lot.

`stype`: The type of sampling.

`type`: The type of distribution on which the plans are based.

`...`: Additional arguments which depend on the distribution to be used and the type of sampling. When `type="hypergeom"` the lot size `N` needs to be specified. Since `pd*N` is a matrix containing the actual number of each type of defective in the lot, its entries must be nonnegative integers. The `stype="fixed"` needs `n` (the sample size), and for `stype="seq"` `m` (the cell quota for good items) must be provided.

Arguments for the `Ocmult` function for the different types of distribution and sampling are summarized in Table 1. If the value of `n` is not specified then it takes the default value of 30. The default value for `N` is 100. For the proportion of defectives `pd` the default value depends on k, and on the type of distribution. These default values for `pd` are given in Table 2.

The new object is created and returned after the arguments are initialized and validated by the initialize and validation functions which are part of the class building.

`type`	`stype`	**Arguments**	**Distribution**
`"multinomial"`	`"fixed"`	`rn`, `pd`, `n`	Multinomial
	`"sequential"`	`rn`, `pd`, `m`	Negative Multinomial
`"hypergeom"`	`"fixed"`	`rn`, `pd`, `n`, `N`	Multivariate Hypergeometric
	`"sequential"`	`rn`, `pd`, `m`, `N`	Negative Multivariate Hypergeometric

Table 1: Arguments and sampling distributions for the `Ocmult` class.

k	Default Values for `pd`
2	All values of $p \in \{0, 0.01, 0.02, \ldots, 0.1\}$
3	A 121×2 matrix whose rows consist of all possible combinations (p_1, p_2) where $p_1, p_2 \in \{0, 0.01, 0.02, \ldots, 0.1\}$
> 3	`type="multinom"`
	A $10{\times}(k-1)$ matrix whose i^{th} row is given by $\frac{i}{25k(k-1)}$`c(1:(k-1))` (for $i = 1, 2, \ldots, 10$) (so that the i^{th} row has a total of $2i\%$ defectives)
	`type="hypergeom"`
	`round(N*pd)/N`, where `pd` is the same as for `type="multinom"` (so that `N*pd` gives integer values)

Table 2: Default values for the proportion of defectives `pd`.

1.2. Initialize and validation methods

When creating an object, the initialize function creates the value for each argument according to the specified distribution. Then the validation functions for the virtual class and actual classes validate if the sampling plan makes sense for the specified distribution. The validation functions for the virtual class applies to both actual classes because of inheritance.

The validation function for the virtual class `Ocmult` checks that the vector `rn` contains no NAs, the values in the vector `rn` are greater than zero, the matrix `pd` contains no NAs, none of the entries in the matrix `pd` are less than 0, the sum of the values in each row of the matrix `pd` is not greater than 1, and that the length of the vector `rn` is equal to the number of columns in the matrix `pd`.

In the validation functions for the actual class `Ocmult.multinomial`, the arguments for fixed sampling are validated by checking that the sample size `n` has length one, is not NA, and is greater than 0, and that none of the values in `rn` are greater than `n`. For sequential sampling, the additional checks are that the cell quota `m` for good items has length one, is not NA, and is greater than 0.

For both fixed and sequential sampling, the validation function for `Ocmult.hypergeom` performs the same checks as for `Ocmult.multinomial` given above. In addition, the validation function for `Ocmult.hypergeom` also checks that the lot size `N` is not NA, the value of `N` is greater than `n`, the length of `N` is equal to 1, the entries in the matrix of `pd*N` are all integers, and that the length of the vector `rn` is less then `N`. Note that this last condition is needed due to the fact that the `rn` vector has length $k-1$ and the number k of different types of items in the lot can't be bigger than the lot size `N`.

1.3. Plot methods

The OC function of 2-level acceptance sampling plans can be presented by plotting the 2-dimensional OC curve corresponding to the sampling plans. In this package plot methods have been created for the actual classes with the proportion of defectives `pd` on the horizontal axis and the probabilities of acceptance `pa` on the vertical axis in the graph. To plot the OC curve corresponding to the sampling plan, only an object of the particular class needs to be specified and all relevant details are extracted from the object (see Section 2 for examples).

The OC function of 3-level acceptance sampling plans can be presented by two types of plots, *persp* and *contour*. In the `persp` methods for the actual classes, the proportions of defectives `pd` are on the two horizontal axes x, y and the probabilities of acceptance `pa` are on the vertical axis z in the graph. In the contour graph, the curves represent the probabilities of acceptance `pa` for specific values corresponding the proportions of defectives `pd` on the two axes x, y. An example is provided in Section 2.

1.4. Summary methods

The generic `summary` method is used to summarize the object. The `show` method gives a brief summary of the supplied object. For fixed sampling it displays the type of distribution, the sample size `n`, and the rejection number(s) `rn`. For sequential sampling, it displays the type of distribution and the cell quotas `m` and `rn` for the good and the defective item(s). If the type of distribution is hypergeometric, then the lot size `N` is also shown.

The `summary` method shows the same detail as the `show` method by default, but provides the additional logical argument `detail`. If `detail=TRUE`, then all the information for the object is printed, including all values of `pd` and the corresponding values of `pa` as well as `ASN` (average sampling number) for sequential sampling.

Examples of these methods are given in Section 2.

1.5. Assessing a sampling plan

The function `assess.multi` can be used to assess whether a user-specified sampling plan can meet certain criteria. The two types of criteria, the *producer's risk point* (PRP) and the *consumer's risk point* (CRP), can be specified singly or together by the arguments `PRP` and `CRP`, respectively.

`PRP` is a vector of length k which contains two parts. The first part consists of $k-1$ elements that are the proportions of each type of defective in the lot which are considered by the producer to be acceptable, i.e., the PQL. The second part is the kth element which represents a lower limit for the corresponding probability of acceptance. For the producer's risk point to be met, the acceptance probability of the plan must be *at least* equal to the value specified by the user in the kth component of `PRP` when the proportions of defectives are equal to the values specified by the user in the first $k-1$ components of `PRP`. This means that the producer's risk α for the user-specified plan must be *at most* equal to 1 minus the kth component of `PRP` for the risk point to be met.

Similarly the `CRP` vector contains the consumer's quality level (CQL) in the first $k-1$ components and an upper limit for the corresponding probability of acceptance in the last component. For the consumer's risk point to be met, the acceptance probability of the plan must be *at most* equal to the value specified by the user in the kth component of `CRP` when the proportions of defectives are equal to the values specified by the user in the first $k-1$ components of `CRP`. This means that the consumer's risk β for the user-specified plan must be *at most* equal to the kth component of `CRP`

The argument `print` in the function `assess.multi` indicates whether a summary of the assessment should be printed. The default value is print=TRUE. The remaining arguments and parameters are the same as the classes, which are summarized in Table 1.

1.6. Finding a sampling plan

The function find.multi.plan allows the user to find a multilevel acceptance sampling plan which meets specified producer and consumer risk points (see Section 1.5). Both points must be specified in the function and the CRP must have worse quality than the PRP.

In the case of type="multinomial", only the PRP and CRP need to be specified. For type ="hypergeom", the additional argument N (the lot size) must be provided.

For fixed sampling, the function finds the smallest sample size n and the corresponding rejection numbers for each type of defective which will meet the PRP and CRP requirements. The process of finding a plan is through trial starting with $n = 1$ and increasing n until the appropriate plan is found. The rejection numbers for the defectives change appropriately at each step. For the sequential sampling, the function finds the smallest quota m for the good items and the corresponding cell quotas for each type of defective, which will meet the PRP and CRP requirements. The process is through trial starting with $m = 1$ and increasing m until the appropriate plan is found. The cell quotas for defectives change appropriately at each step but do not exceed the quota for the good items.

2. Examples

2.1. Creating Ocmult plans

A new object of the sampling plan can be created by using the Ocmult function. For example, a sampling plan with $n = 20$ and $rn = (2, 4, 3)$ for a large lot size can be obtained as follows:

```
R> library("MFSAS")
R> plan.mf <- Ocmult(rn = c(2, 4, 3), n = 30)
R> plan.mf

  4-Level Acceptance Sampling Plan Multinomial:
 Sample size:    30
 Rej. Number(s): 2 4 3
```

Optionally, the types of defectives can be specified by using the built-in R function names to name the components of the rn vector:

```
R> rn <- c(2, 4, 3)
R> names(rn) <- c("Critical", "Major", "Minor")
R> plan.mf <- Ocmult(rn = rn, n = 30)
R> plan.mf

  4-Level Acceptance Sampling Plan Multinomial:
 Sample size:    30
                 Critical Major Minor
 Rej. Number(s):        2     4     3
```

For a finite lot of size $N = 100$, `type="hyper"` should be specified. (Note that argument matching is used for the `type` and `stype` arguments, so it is only necessary to enter enough letters to uniquely identify the distribution and the type of sampling.) If the sample size is $n = 15$, and if $rn = (2, 3)$, then the sampling plan is obtained as:

```
R> plan.hf <- Ocmult(rn = c(2, 3), n = 15, N = 100, type = "hyper")
R> plan.hf

  3-Level Acceptance Sampling Plan Multivariate Hypergeom: N = 100
 Sample size:    15
 Rej. Number(s): 2 3
```

Now consider a sequential sampling plan in which sampling continues until we obtain either 5 good items or 3 defectives. In this case we use `stype="seq"`, and the target number m of good items must be specified:

```
R> plan.ms <- Ocmult(rn = 3, m = 5, stype = "seq", pd = seq(0,
+      0.99, 0.01))
R> plan.ms

  2-Level Sequential Acceptance Sampling Plan Negative Multinomial:
 Acc. Number:    5
 Rej. Number(s): 3
```

If the defective items are classified as "Marginal" or "Bad" in a lot of size $N = 100$, and we continue to sample until we obtain either 4 good items, 5 Marginal items, or 2 Bad items, then the 3-level sequential sampling plan can be obtained as follows:

```
R> rn <- c(5, 2)
R> names(rn) <- c("Marginal", "Bad")
R> plan.hs <- Ocmult(rn = rn, m = 4, N = 100, type = "hyper",
+      stype = "seq")
R> plan.hs

  3-Level Sequential Acceptance Sampling Plan Negative Multivariate Hypergeom: N = 100
 Acc. Number:    4
                 Marginal Bad
 Rej. Number(s):        5   2
```

2.2. Plotting sampling plans

The OC curve or surface can be plotted when the sampling plans are 2-level or 3-level.

For the 2-level sequential sampling plan with `type="multinomial"` given in the example above, the plot is shown in Figure 2:

```
R> plot(plan.ms)
```

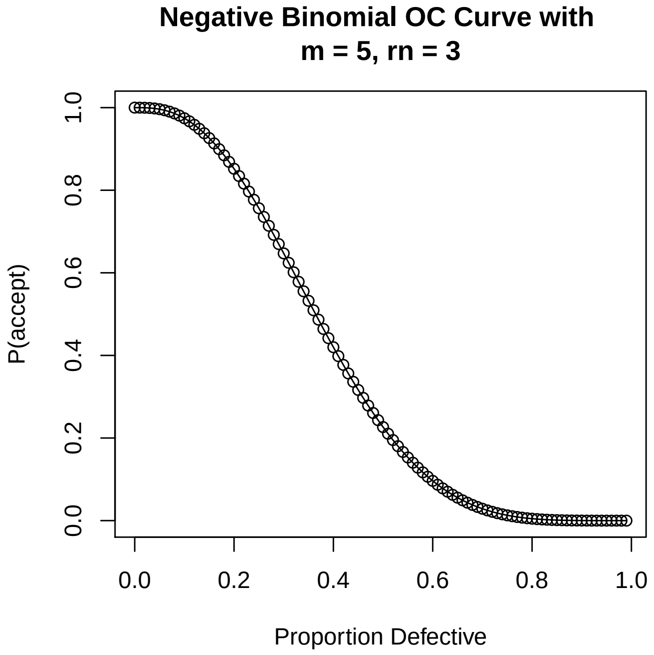

Figure 2: OC curve for sequential sampling, with replacement.

All arguments for the standard plot method can be passed directly to the generic `plot` method in this package. For the binomial and negative binomial distributions the OC curve is, by default, plotted with both lines and points by using `type="o"` in the standard plot type. The OC curve for the hypergeometric and negative hypergeometric distributions are, by default, plotted only with points by using `type="p"` in the standard plot type.

The following example is for 2-level sampling plans for all four combinations of the sampling type and distribution. Since the proportion of defectives in the population is typically small, we use values of *pd* so that the probabilities of acceptance for proportions of defectives between 0 and 0.2 can be more clearly seen on the y-axis of the resulting graphs, which are given in Figure 3.

```
R> x.mn <- Ocmult(n = 8, rn = 3, pd = seq(0, 0.22, 0.01))
R> x.mh <- Ocmult(n = 8, rn = 3, N = 50, type = "hyper",
+      pd = seq(0, 0.2, 0.02))
R> x.nmn <- Ocmult(rn = 2, m = 4, stype = "seq", pd = seq(0,
+      0.22, 0.01))
R> x.nmh <- Ocmult(rn = 2, m = 4, N = 50, type = "hyper",
+      stype = "seq", pd = seq(0, 0.2, 0.02))
R> par(mfrow = c(1, 2))
R> main = "Fixed Sampling Plan \nn = 8, rn = 3"
R> plot(x.mn, type = "l", xlim = c(0, 0.2), ylim = c(0.75,
+      1), main = main)
R> grid(lty = "solid")
R> points(x.mh@pd, x.mh@pa, col = 3)
```

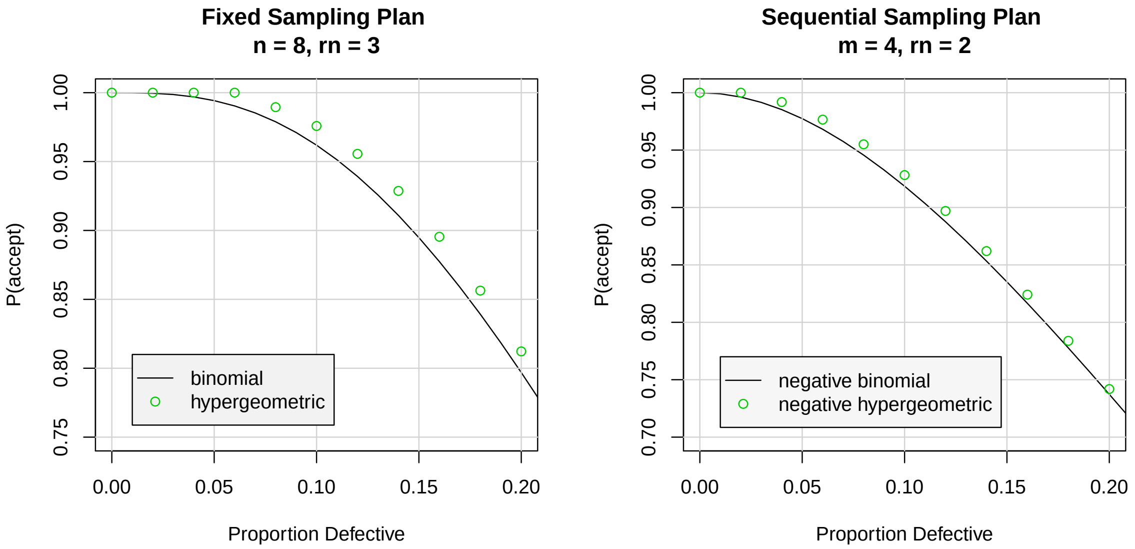

Figure 3: OC curve for fixed sampling and sequential sampling.

```
R> legend(0.01, 0.81, c("binomial", "hypergeometric"), col = c(1,
+      3), lty = c(1, -1), pch = c(-1, 1), bg = "gray95")
R> main = "Sequential Sampling Plan \nm = 4, rn = 2"
R> plot(x.nmn, type = "l", xlim = c(0, 0.2), ylim = c(0.7,
+      1), main = main)
R> grid(lty = "solid")
R> points(x.nmh@pd, x.nmh@pa, col = 3)
R> legend(0.01, 0.77, c("negative binomial", "negative hypergeometric"),
+      col = c(1, 3), lty = c(1, -1), pch = c(-1, 1), bg = "gray96")
```

From both graphs in Figure 3 we see that the probabilities for acceptance are higher for the hypergeometric distribution than for the binomial. So, for example, if the lot size is $N = 50$ and a proportion of defectives of 5% is considered acceptable, then sampling without replacement will result in a smaller value for the producer's risk. On the other hand, if a proportion of defectives of 15% is considered unacceptable, then sampling without replacement will result in a larger value for the consumer's risk. This relationship between binomial and hypergeometric probabilities will generally be true for small values of pd. But for large values of pd the binomial probabilities will be higher.

For 3-level sampling plans, all arguments for the 'persp' and 'contour' methods can be passed directly to the generic `persp` and `contour` functions in this package, respectively. For example, in Figure 4 we use light blue color for the surface and 4 contour levels for the contour plot,

```
R> par(mfrow = c(1, 2))
R> px <- matrix(c(seq(0, 0.2, 0.01), seq(0, 0.8, 0.04)),
+      ncol = 2)
R> p.multinom <- OCmult(n = 15, rn = c(2, 10), pd = px)
```

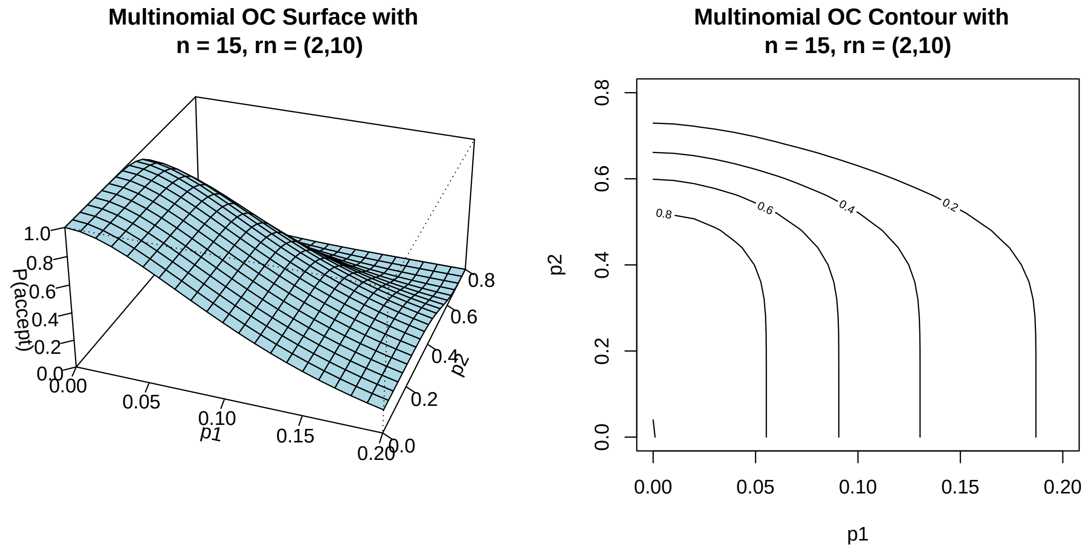

Figure 4: OC surface and contour curve for fixed sampling without replacement, with $N = 100, n = 15, rn = (2,3)$.

```
R> persp(p.multinom, col = "light blue")
R> contour(p.multinom, nlevel = 4)
```

2.3. Sampling plan summary

The summary method gives a summary of the sampling plan with an option for detailed output, as in the following example.

```
R> px <- matrix(c(seq(0, 0.5, 0.1), seq(0, 0.2, 0.04), seq(0,
+     0.1, 0.02)), ncol = 3)
R> rn <- c(5, 4, 2)
R> names(rn) <- c("Critical", "Major", "Minor")
R> p.multinom <- Ocmult(n = 30, rn = rn, pd = px)
R> summary(p.multinom, detail = TRUE)
```

```
  4-Level Acceptance Sampling Plan Multinomial:
 Sample size:    30
                 Critical Major Minor
 Rej. Number(s):        5     4     2

 Detailed acceptance probabilities:
  Critical Major Minor  P.nondef  P(accept)
       0.0  0.00  0.00      1.00  1.0000000
       0.1  0.04  0.02      0.84  0.6977166
       0.2  0.08  0.04      0.68  0.1103746
       0.3  0.12  0.06      0.52  0.0026117
       0.4  0.16  0.08      0.36  0.0000045
       0.5  0.20  0.10      0.20  0.0000000
```

Note that pd is a matrix with $k-1$ columns, where each row contains the proportions for each type of defective in the lot. However, if a vector is provided, then it is converted to a matrix by row according to the length of rn. The vector will be truncated (with a warning) if its length is not an integer multiple of the length of rn. An example follows:

```
R> px <- c(0.02, 0.06, 0.04, 0.06, 0.08, 0.02, 0.02, 0.08,
+      0.04, 0.1)
R> pmh <- Ocmult(rn = c(2, 3, 2, 4), n = 20, N = 100, pd = px,
+      type = "hyper")

Warning message:
In .local(.Object, ...) :
  The length of the pd vector should be an integer
multiple of the length of the rn vector.
The truncated pd in use is:
0.02  0.06  0.04  0.06  0.08  0.02  0.02  0.08
```

2.4. Assessing a sampling plan

The assess.multi function is used to assess a sampling plan given the PRP and/or CRP (see Section 1.5 for a description). For example, suppose that we have two types of defectives, "Marginal" and "Bad", and we want a 3-level plan to meet the producer's risk point which has an acceptance probability of *at least* 0.95 when the proportions of the 2 types of defectives are equal to 0.05 and 0.06, and for the plan to also meet the consumer's risk point, which has an acceptance probability *at most* 0.1 when the proportions of the 2 types of defectives are equal to 0.14, and 0.18. We can assess whether the sequential plan with cell quotas of $m=5$ good items and $rn=(2,3)$ defectives meets the given PRP and CRP as follows:

```
R> rn <- c(2, 3)
R> names(rn) <- c("Marginal", "Bad")
R> assess.multi(rn = rn, m = 5, PRP = c(0.05, 0.06, 0.95),
+      CRP = c(0.14, 0.18, 0.1), stype = "seq")

  3-Level Acceptance Sampling Plan Negative Multinomial:
 Acc. Number:    5
                 Marginal Bad
 Rej. Number(s):        2   3

 Plan CANNOT meet desired risk point(s):
    Marginal  Bad  RP P(accept)  Plan P(accept)        ASN
PRP     0.05 0.06          0.95       0.9564935   5.502048
CRP     0.14 0.18          0.10       0.6278492   5.922620
```

The output shows that the plan cannot meet both risk points. Although the PRP is satisfied since the actual value for P(accept) is 0.956, which exceeds minimum desired level of 0.95, the value of P(accept) for CRP is 0.628 which is greater than the maximum allowable level of 0.1. For sequential sampling, the output also displays the average sampling number(s) ASN for the risk point(s).

2.5. Finding a sampling plan

The `find.multi.plan` function provides a method to find a plan which will meet the specified risk points PRP and CRP. For example, the implementation of `find.multi.plan` for sequential sampling from a lot of size 100 is:

```
R> find.multi.plan(PRP = c(0.06, 0.04, 0.06, 0.8), CRP = c(0.14,
+      0.16, 0.2, 0.1), N = 100, type = "hyper", stype = "seq")

 The optimal plan is:

  4-Level Acceptance Sampling Plan Negative Multivariate Hypergeometric: N = 100
 Acc. Number:    7
 Rej. Number(s): 2 2 2

     type 1  type 2  type 3  RP P(accept)  Plan P(accept)        ASN
PRP    0.06    0.04    0.06           0.8       0.8056496   7.589796
CRP    0.14    0.16    0.20           0.1       0.0814709   5.510192
```

This shows that, in order to meet both risk points, we should take observations one at a time until we get either $m = 7$ good items, or 2 of any type of defective. If the former occurs first then the lot should be accepted; otherwise it is rejected. The probability of acceptance at the producer's quality level is 0.806 and at the consumer's quality level is 0.081. The average sampling number is ASN = 7.6 at the producer's quality level, and ASN = 5.5 at the consumer's quality level.

We can also find a plan with the same risk points PRP, CRP, and lot size as above but for fixed sampling:

```
R> find.multi.plan(PRP = c(0.06, 0.04, 0.06, 0.8), CRP = c(0.14,
+      0.16, 0.2, 0.1), N = 100, type = "hyper")

 The optimal plan is:

  4-Level Acceptance Sampling Plan Multivariate Hypergeometric: N = 100
 Sample size:    11
 Rej. Number(s): 2 2 3

     type 1  type 2  type 3  RP P(accept)  Plan P(accept)
PRP    0.06    0.04    0.06           0.8       0.8023994
CRP    0.14    0.16    0.20           0.1       0.0904328
```

The above output shows that if we want to meet both risk points, we need a sample of size $n = 11$. The lot is rejected if the sample contains at least 2 of either of the first two types of defectives, or at least 3 of the third type of defective. Hence to meet both risk points, the sequential sampling procedure requires on average a smaller sample size (ASN = 7.6) than the corresponding fixed sample size procedure ($n = 11$).

3. Cumulative distribution functions

The **MFSAS** package also includes functions to calculate the cumulative distribution functions (CDFs) for the multinomial, negative multinomial, multivariate hypergeometric, and negative multivariate hypergeometric distributions which are required for the calculation of acceptance probabilities for the multilevel sampling plans in this package. A brief description follows, and the reader is referred to Johnson, Kotz, and Balakrishnan (1997) for further details about these distributions.

`pmultinom` - *The multinomial CDF*

The multinomial distribution is used for sampling *with* replacement, or if the lot is large compared to the sample size. If a sample of size n is drawn from a lot whose k classes have probabilities $p_1, \ldots, p_{k-1}, p_k$, let $X_1, \ldots, X_{k-1}, X_k$ denote the number of observations drawn from each of the k classes. Let `x` be a vector of length $k-1$ corresponding to the observed number of items drawn from the first $k-1$ classes, and let `p` be a vector containing the probabilities for each of those $k-1$ classes. Then the cumulative probability `pmultinom(x, size=n, prob=p)` is given by

$$P(X_1 \le x_1, \ldots, X_{k-1} \le x_{k-1})$$

$$= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} P(X_1 = y_1, \ldots, X_{k-1} = y_{k-1})$$

$$= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} \frac{n!}{y_1! \ldots y_{k-1}!(n - \sum_{i=1}^{k-1} y_i)!} p_1^{y_1} \ldots p_{k-1}^{y_{k-1}} \left(1 - \sum_{i=1}^{k-1} p_i\right)^{\left(n - \sum\limits_{i=1}^{k-1} y_i\right)}$$

where the sum is over all values of y such that $\sum\limits_{i=1}^{k-1} y_i \le n$.

`pmultinom` is computed using recursive algorithms for the Dirichlet **J** function given in Sobel, Uppuluri, and Frankowski (1977) and Sobel and Frankowski (2004).

`pnmultinom` - *The negative multinomial CDF*

The negative multinomial distribution is used for sequential sampling *with* replacement. Suppose that the lot has $k-1$ different types of failures, with corresponding probabilities $p_1, \ldots, p_{k-1}$ in each trial. Let $X_1, \ldots, X_{k-1}$ denote the number of failures of each type that are selected in a sequence of trials before a target number m of successes is reached. (Note that success corresponds to selecting a nondefective or good item.) Then `pnmultinom(x, m,` `prob=p)` is the cumulative probability:

$$
\begin{aligned}
&P(X_1 \leq x_1, \ldots, X_{k-1} \leq x_{k-1}) \\
&= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} P(X_1 = y_1, \ldots, X_{k-1} = y_{k-1}) \\
&= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} \frac{\left(\sum_{i=1}^{k-1} y_i + m - 1\right)!}{(m-1)! y_1! \ldots y_{k-1}!} p_1^{y_1} \ldots p_{k-1}^{y_{k-1}} \left(1 - \sum_{i=1}^{k-1} p_i\right)^m
\end{aligned}
$$

Note that for this distribution to makes sense, the probability of success cannot be zero. Therefore we must have that $\sum_{i=1}^{k-1} p_i < 1$.

When $k = 2$ this gives the negative binomial CDF, but is parameterized differently from the `pnbinom` function in the **stats** package in which `prob` denotes the probability of success. Therefore for scalars `x`, `m`, and `p` we have that `pnmultinom(x,m,p)` = `pnbinom(x,m,1-p)`.

`pnmultinom` is computed using recursive algorithms for the Dirichlet **D** function given in Sobel, Uppuluri, and Frankowski (1985) and Sobel and Frankowski (2004).

`pmultihyper` - *The multivariate hypergeometric CDF*

The multivariate hypergeometric distribution is used for sampling from a finite lot *without* replacement. If a sample of size n is drawn from a lot of size N which has M_i objects of type i (for $i = 1, 2, \ldots, k$), let X_i be the number of objects of type i in the sample (for $i = 1, 2, \ldots, k$). Let `x` be a vector of length $k - 1$ corresponding to the observed number of objects drawn from each of the first $k - 1$ classes, and let `M` be a vector containing the sizes for each of those $k - 1$ classes. Then the cumulative probability `pmultihyper(x, n, M, N)` is given by,

$$
\begin{aligned}
&P(X_1 \leq x_1, \ldots, X_{k-1} \leq x_{k-1}) \\
&= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} P(X_1 = y_1, \ldots, X_{k-1} = y_{k-1}) \\
&= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} \frac{\binom{M_1}{y_1} \cdots \binom{M_{k-1}}{y_{k-1}} \binom{N - \sum_{i=1}^{k-1} M_i}{n - \sum_{i=1}^{k-1} y_i}}{\binom{N}{n}}
\end{aligned}
$$

where the sum is over all values of y such that $\sum_{i=1}^{k-1} y_i \leq n$ and $n - \sum_{i=1}^{k-1} y_i \leq N - \sum_{i=1}^{k-1} M_i$.

`pmultihyper` is computed using recursive algorithms for the Dirichlet **HJ** function given in Sobel and Frankowski (1994).

Note that if M_i and N approach infinity in such a way that the ratio $M_i/N = p_i$ is fixed (for $i = 1, 2, \ldots, k$) then the multivariate hypergeometric density function converges to the multinomial density function. Therefore the multinomial distribution can be used as an approximation to the multivariate hypergeometric distribution when sampling is without replacement and the population size is large. When $k = 2$, a general guideline for the use of this approximation is that the population size N should be at least 20 times the sample size n. For $k > 2$ the situation is not as straightforward, but error bounds for this approximation can be found in Loh (1992). A numerical example illustrating this convergence is given in Section 3.1 below.

`pnmultihyper` - *The negative multivariate hypergeometric CDF*

The negative multivariate hypergeometric distribution is used for sequential sampling from a finite lot *without* replacement. Suppose that the lot of size N has $k-1$ different types of failures represented $M_1, \ldots, M_{k-1}$ times, respectively. Let $X_1, \ldots, X_{k-1}$ denote the number of failures of each type that are selected in a sequence of trials before a target number m of successes is reached. Then `pnmultihyper(x, m, M, N)` is the cumulative probability:

$$P(X_1 \leq x_1, \ldots, X_{k-1} \leq x_{k-1})$$

$$= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} P(X_1 = y_1, \ldots, X_{k-1} = y_{k-1})$$

$$= \sum_{y_1=0}^{x_1} \cdots \sum_{y_{k-1}=0}^{x_{k-1}} \frac{\binom{M_1}{y_1} \cdots \binom{M_{k-1}}{y_{k-1}} \binom{N - \sum_{i=1}^{k-1} M_i}{m-1}}{\binom{N}{m-1+\sum_{i=1}^{k-1} y_i}} \frac{N-m+1-\sum_{i=1}^{k-1} M_i}{N-m+1-\sum_{i=1}^{k-1} y_i}$$

where the sum is over all values of y such that $\sum_{i=1}^{k-1} y_i \leq N - m$.

Note that the target number of successes m cannot be bigger than the total number of successes in the population. Therefore for this distribution to makes sense, we must have that $0 < m \leq N - \sum_{i=1}^{k-1} M_i$.

As with the multivariate hypergeometric and multinomial distributions, if M_i and N approach infinity in such a way that the ratio $M_i/N = p_i$ is fixed (for $i = 1, 2, \ldots, k$) then the negative multivariate hypergeometric density function converges to the negative multinomial density function. Therefore the negative multinomial distribution can be used as an approximation to the negative multivariate hypergeometric distribution when sampling is without replacement and the population size is large. See Section 3.1 below for an example.

`pnmultihyper` is computed using recursive algorithms for the Dirichlet **HD** function. See Childs (2010) and Sobel and Frankowski (1995).

3.1. Examples of calculating cumulative probabilities

The following are examples of how to calculate the lower tail cumulative probabilities for each of the above distributions.

- Multivariate hypergeometric distribution:

  Suppose that a lot of size $N = 100$ contains products which are classified according to 4 levels of product quality, one of which is good items. If $M = (8, 10, 14)$ is the number of items in the lot which fall into each of the three categories of defectives, and we draw a sample of size $n = 15$ (without replacement) from the lot, then the probability that the sample contains no more than 1 item with the first type of defect, no more than 3 items with the second type of defect, and no more than 4 items with the third type of defect is calculated as follows:

```
R> X <- c(1, 3, 4)
R> n <- 15
R> M <- c(8, 10, 14)
R> N <- 100
R> pr <- pmultihyper(X, n, M, N)
R> pr

[1] 0.599595
```

- Multinomial distribution:

  In the same setting as above, if the sampling is done with replacement, or equivalently if the lot size is large and the proportions of each of the 3 types of defects in the lot is given by $p = (0.08, 0.10, 0.14)$, then the corresponding probability is calculated using the multinomial CDF:

```
R> X <- c(1, 3, 4)
R> n <- 15
R> pr <- c(0.08, 0.1, 0.14)
R> cdf <- pmultinom(x = X, size = n, prob = pr)
R> cdf

[1] 0.5816256
```

  Alternatively, since the multinomial distribution is the limiting form of the multivariate hypergeometric distribution, the same result can be obtained using `pmultihyper` and a numerical limiting process by letting M and N in the previous example approach infinity in such a way that the ratio $M/N = (0.08, 0.10, 0.14)$ is fixed:

```
R> for (i in c(0:6)) print(pmultihyper(X, n, 10^i * M, 10^i *
+      N))

[1] 0.599595
[1] 0.5832931
[1] 0.5817911
[1] 0.5816421
[1] 0.5816272
[1] 0.5816257
[1] 0.5816256
```

- Negative multivariate hypergeometric distribution:

  Suppose that a lot of size $N = 130$ contains items with 4 different types of product defects with $M = (5, 7, 8, 3)$ of each type. If we select and inspect items one at a time (without replacement) until we obtain 5 good items then the probability that we end up with no more than $X = (2, 3, 4, 1)$ items with each type of defect (at the time when the 5^{th} good item is selected) is calculated as follows:

```
R> X <- c(2, 3, 4, 1)
R> m <- 5
R> M <- c(5, 7, 8, 3)
R> N <- 130
R> cdf <- pnmultihyper(X, m, M, N)
R> cdf

[1] 0.990882
```

- Negative multinomial distribution:

  In the same setting as above, if the sampling is done with replacement then the corresponding probability is calculated using the negative multinomial distribution:

```
R> X <- c(2, 3, 4, 1)
R> m <- 5
R> pr <- c(5/130, 7/130, 8/130, 3/130)
R> pnmultinom(x = X, m = m, prob = pr)

[1] 0.9860325
```

  As with the multinomial distribution, the same result can be obtained using `pnmultihyper` and a numerical limiting process:

```
R> for (i in c(0:6)) print(pnmultihyper(X, m, 10^i * M,
+      10^i * N))

[1] 0.990882
[1] 0.9865194
[1] 0.9860812
[1] 0.9860374
[1] 0.986033
[1] 0.9860325
[1] 0.9860325
```

4. Discussion

The **MFSAS** package provides the user with the tools to create, evaluate, plot, and display multilevel acceptance sampling plans for both fixed and sequential sampling. We have also provided cumulative distribution functions for several discrete multivariate distributions.

Our sampling plans are currently restricted to one-stage sampling. Multi-stage sampling may be incorporated into future versions of the package. For users interested in multi-stage fixed sampling plans for two levels of product quality, the **AcceptanceSampling** package can be used.

We should also note that all of the procedures used for the calculations in this package are exact (aside from possible rounding errors) and are written entirely in R. As a result there is the potential for the calculations to be slow for large values of k, m, and n. Consequently the `find.multi.plan` function can be slow if the probability given in the last component of the producer's risk point is too close to 1 or if the corresponding probability in `CRP` is too close to 0. Therefore the package could be improved by developing and incorporating asymptotic approximations into the calculations, or by using compiled code for the CDFs.

Our cumulative distribution functions currently do not have the option for calculating upper tail probabilities. This option is available for most (if not all) of the currently available R functions for discrete univariate CDFs. However, the relationship between upper and lower tail probabilities is not as straight forward in the multivariate case. Efficient calculation of upper tail probabilities would require a different (but related) set of recursive algorithms. As a result of this, and the fact that upper tail probabilities are not needed in this package, our CDFs do not currently have this functionality. Since upper tail probabilities may be useful in other applications, this is another area in which the package can be improved.

Finally, we hope that practitioners will find this package useful and we welcome any other suggestions for improvements, or for added functionality.

Acknowledgements

The first author would like to thank the Natural Sciences and Engineering Research council of Canada for funding this research. We would also like to thank the editor and an anonymous referee for providing suggestions which let to an improvement in this paper, and in the package as a whole.

References

Bray D, Lyon D, Burr I (1973). "Three Class Attributes Plans in Acceptance Sampling." *Technometrics*, **15**(3), 575–585.

Cassady CR, Nachlas AJ (2003). "Evaluating and Implementing 3-Level Acceptance Sampling Plans." *Quality Engineering*, **15**(3), 361–369.

Childs A (2010). "Vector Extensions of the Dirichlet HC and HD Functions, with Applications to the Sharing Problem." *Methodology and Computing in Applied Probability*, **12**, 91–109.

Childs A, Chen Y (2011). "Multilevel Fixed and Sequential Acceptance Sampling: The R Package **MFSAS**." *Journal of Statistical Software*, **43**(6), 1–20. URL `http://www.jstatsoft.org/v43/i06/`.

Johnson NL, Kotz S, Balakrishnan N (1997). *Discrete Multivariate Distributions*. John Wiley & Sons, New York.

Kiermeier A (2008). "Visualizing and Assessing Acceptance Sampling Plans: The R Package **AcceptanceSampling**." *Journal of Statistical Software*, **26**(6), 1–20. URL http://www.jstatsoft.org/v26/i06/.

Loh WL (1992). "Stein's Method and Multinomial Approximation." *The Annals of Applied Probability*, **2**(3), 536–554.

MIL-STD-105E (1989). *Sampling Procedures and Tables for Inspection by Attributes.* Department of Defense, Washington, DC.

Newcombe P, Allen O (1988). "A Three-Class Procedure for Acceptance Sampling by Variables." *Technometrics*, **30**(4), 415–421.

Radhakrishnan R, Sankar SR (2009). "Single Sampling Plan for Three Attribute Classes Indexed Through Acceptable Quality Level." *International Journal of Statistics and Management System*, **4**(1), 150–164.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Schilling EG, Neubauer DV (2009). *Acceptance Sampling in Quality Control.* 2nd edition. CRC Press, New York.

Shapiro S, Zaheda H (1990). "Bernoulli Trials and Discrete Distributions." *Journal of Quality Technology*, **22**(3), 193–205.

Sobel M, Frankowski K (1994). "Hypergeometric Analogues of Multinomial Type-1 Dirichlet Problems." *Congressus Numerantium*, **101**, 65–82.

Sobel M, Frankowski K (1995). "Hypergeometric Analogues of Multinomial Type-2 Problems via Dirichlet Methodology." *Congressus Numerantium*, **106**, 171–191.

Sobel M, Frankowski K (2004). *Handbook of Beta Distribution and its Applications*, chapter Extensions of Dirichlet Integrals: Their Computation and Probability Applications, pp. 319–360. Marcel Dekker, New York.

Sobel M, Uppuluri VRR, Frankowski K (1977). *Dirichlet Integrals of Type-1 and Their Application*, volume 4 of *Selected Tables in Mathematical Statistics.* American Mathematical Society, Providence, Rhode Island.

Sobel M, Uppuluri VRR, Frankowski K (1985). *Dirichlet Integrals of Type-2 and Their Application*, volume 9 of *Selected Tables in Mathematical Statistics.* American Mathematical Society, Providence, Rhode Island.

Thatcher FS, Clarke DS (1978). *Micro-Organisms in Food 2: Sampling for Microbiological Analysis, Principles and Specific Applications.* University of Toronto Press, Toronto.

Affiliation:

Aaron Childs
Department of Mathematics and Statistics

McMaster University
Hamilton, Ontario, Canada L8S 4K1
Telephone: 905-525-9140 Ext: 23426
E-mail: childsa@mcmaster.ca
URL: http://www.math.mcmaster.ca/childsa/childs.html