Package 'jfa'

January 8, 2020

Title Bayesian and Classical Audit Sampling

Version 0.1.0

Description Implements the audit sampling workflow as discussed in Derks et al. (2019) <doi:10.31234/osf.io/9f6ub>. The package makes it easy for an auditor to plan an audit sample, sample from the population, and evaluating that sample using various confidence bounds according to the International Standards on Auditing. Furthermore, the package implements Bayesian equivalents of these methods.

Language en-US

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

VignetteBuilder knitr

Suggests testthat, knitr, rmarkdown

NeedsCompilation no

Author Koen Derks [aut, cre]

Maintainer Koen Derks <`k.derks@nyenrode.nl`>

Repository CRAN

Date/Publication 2020-01-08 17:10:10 UTC

R topics documented:

`auditPrior` *Create a Prior Distribution*

Description

This function creates a prior distribution according to the audit risk model. The returned object is of class `jfaPrior` and can be used with associated `print()` and `plot()` methods. `jfaPrior` objects can be used as input argument for the `prior` argument in other functions.

Usage

```
auditPrior(materiality, confidence = 0.95, method = "arm", ir = 1, cr = 1,
           expectedError = 0, likelihood = "binomial", N = NULL)
```

Arguments

`materiality`	a value between 0 and 1 representing the materiality of the audit as a fraction of the total size or value.
`confidence`	the confidence level desired from the confidence bound (on a scale from 0 to 1). Defaults to 0.95, or 95% confidence.
`method`	the method by which the prior distribution is constructed. Currently only supports the `arm` method, which uses the audit risk model (Derks et al., 2019).
`ir`	the inherent risk probability from the audit risk model. Defaults to 1 for 100% risk.
`cr`	the inherent risk probability from the audit risk model. Defaults to 1 for 100% risk.
`expectedError`	a fraction representing the percentage of expected mistakes in the sample relative to the total size, or a number (>= 1) that represents the number of expected mistakes.
`likelihood`	can be one of `binomial`, `poisson`, or `hypergeometric`.
`N`	the population size (required for hypergeometric calculations).

Value

An object of class `jfaPrior` containing:

`method`	the method by which the prior distribution is constructed.
`likelihood`	the likelihood by which the prior distribution is updated.
`priorD`	the name of the probability density function of the prior distribution.
`nPrior`	the prior assumed sample size.
`kPrior`	the prior assumed sample errors
`aPrior`	the prior parameter alpha.
`bPrior`	the prior parameter beta.
`materiality`	the materiality that was used to construct the prior distribution.
`N`	if specified as input, the population size.

Author(s)

Koen Derks, <k.derks@nyenrode.nl>

References

Derks, K., de Swart, J., Wagenmakers, E.-J., Wille, J., & Wetzels, R. (2019). JASP for audit: Bayesian tools for the auditing practice.

See Also

planning sampling evaluation

Examples

```
library(jfa)

# Specify the materiality, confidence, and expected errors:
materiality   <- 0.05   # 5%
confidence    <- 0.95   # 95%
expectedError <- 0.025  # 2.5%

# Specify the inherent risk (ir) and control risk (cr):
ir <- 1     # 100%
cr <- 0.6   # 60%

# Create a beta prior distribution according to the Audit Risk Model (arm)
# and a binomial likelihood:
prior <- auditPrior(materiality = materiality, confidence = confidence,
                    method = "arm", ir = ir, cr = cr,
                    expectedError = expectedError, likelihood = "binomial")
print(prior)

# jfa prior distribution for arm method:
#
# Prior sample size:     51
# Prior errors:          1.27
# Prior:                 beta(2.275, 50.725)
```

BuildIt *BuildIt Construction financial statements*

Description

Fictional data from a construction company in the United States, containing 3500 observations identification numbers, book values, and audit values. The audit values are added for illustrative purposes, as these would need to be assessed by the auditor in the execution stage of the audit.

Usage

```
data(BuildIt)
```

Format

A data frame with 3500 rows and 3 variables.

ID unique record identification number.

bookValue book value in US dollars ($14.47–$2,224.40).

auditValue true value in US dollars ($14.47–$2,224.40).

References

Derks, K., de Swart, J., Wagenmakers, E.-J., Wille, J., & Wetzels, R. (2019). JASP for audit: Bayesian tools for the auditing practice.

Examples

```
data(BuildIt)
```

`evaluation` *Evaluation of Audit Samples using Confidence / Credible Bounds*

Description

This function takes a sample data frame or summary statistics about an evaluated audit sample and calculates a confidence bound according to a specified method. The returned object is of class `jfaEvaluation` and can be used with associated `print()` and `plot()` methods.

Usage

```
evaluation(sample = NULL, bookValues = NULL, auditValues = NULL,
           confidence = 0.95, nSumstats = NULL, kSumstats = NULL,
           method = "binomial", materiality = NULL, N = NULL,
           prior = FALSE, nPrior = 0, kPrior = 0,
           rohrbachDelta = 2.7, momentPoptype = "accounts",
           populationBookValue = NULL,
           csA = 1, csB = 3, csMu = 0.5)
```

Arguments

`sample`	a data frame containing at least a column of book values and a column of audit (true) values.
`bookValues`	the column name for the book values in the sample.
`auditValues`	the column name for the audit (true) values in the sample.
`confidence`	the required confidence level for the bound.
`nSumstats`	the number of observations in the sample. If specified, overrides the `sample`, `bookValues` and `auditValues` arguments and assumes that the data comes from summary statistics specified by `nSumstats` and `kSumstats`.

`kSumstats`	the sum of the errors found in the sample. If specified, overrides the `sample`, `bookValues` and `auditValues` arguments and assumes that the data comes from summary statistics specified by `kSumstats` and `nSumstats`.
`method`	can be either one of `poisson`, `binomial`, `hypergeometric`, `stringer`, `stringer-meikle`, `stringer-lta`, `stringer-pvz`, `rohrbach`, `moment`, `direct`, `difference`, `quotient`, or `regression`.
`materiality`	if specified, the function also returns the conclusion of the analysis with respect to the materiality. This value must be specified as a fraction of the total value of the population (a value between 0 and 1). The value is discarded when `direct`, `difference`, `quotient`, or `regression` method is chosen.
`N`	the total population size.
`prior`	whether to use a prior distribution when evaluating. Defaults to `FALSE` for frequentist evaluation. If `TRUE`, the prior distribution is updated by the specified likelihood. Chooses a conjugate gamma distribution for the Poisson likelihood, a conjugate beta distribution for the binomial likelihood, and a conjugate beta-binomial distribution for the hypergeometric likelihood.
`nPrior`	the prior parameter α (number of errors in the assumed prior sample).
`kPrior`	the prior parameter β (total number of observations in the assumed prior sample).
`rohrbachDelta`	the value of Δ in Rohrbach's augmented variance bound.
`momentPoptype`	can be either one of `accounts` or `inventory`. Options result in different methods for calculating the central moments, for more information see Dworin and Grimlund (1986).
`populationBookValue`	the total value of the audit population. Required when `method` is one of `direct`, `difference`, `quotient`, or `regression`.
`csA`	if `method = "coxsnell"`, the α parameter of the prior distribution on the mean taint. Default is set to 1, as recommended by Cox and Snell (1979).
`csB`	if `method = "coxsnell"`, the β parameter of the prior distribution on the mean taint. Default is set to 3, as recommended by Cox and Snell (1979).
`csMu`	if `method = "coxsnell"`, the mean of the prior distribution on the mean taint. Default is set to 0.5, as recommended by Cox and Snell (1979).

Details

This section lists the available options for the `methods` argument.

- `poisson`: The confidence bound taken from the Poisson distribution. If combined with `prior = TRUE`, performs Bayesian evaluation using a *gamma* prior and posterior.
- `binomial`: The confidence bound taken from the binomial distribution. If combined with `prior = TRUE`, performs Bayesian evaluation using a *beta* prior and posterior.
- `hypergeometric`: The confidence bound taken from the hypergeometric distribution. If combined with `prior = TRUE`, performs Bayesian evaluation using a *beta-binomial* prior and posterior.
- `stringer`: The Stringer bound (Stringer, 1963).

- `stringer-meikle`: Stringer bound with Meikle's correction for understatements (Meikle, 1972).
- `stringer-lta`: Stringer bound with LTA correction for understatements (Leslie, Teitlebaum, and Anderson, 1979).
- `stringer-pvz`: Stringer bound with Pap and van Zuijlen's correction for understatements (Pap and van Zuijlen, 1996).
- `rohrbach`: Rohrbach's augmented variance bound (Rohrbach, 1993).
- `moment`: Modified moment bound (Dworin and Grimlund, 1986).
- `coxsnell`: Cox and Snell bound (Cox and Snell, 1979).
- `direct`: Confidence interval using the direct method (Touw and Hoogduin, 2011).
- `difference`: Confidence interval using the difference method (Touw and Hoogduin, 2011).
- `quotient`: Confidence interval using the quotient method (Touw and Hoogduin, 2011).
- `regression`: Confidence interval using the regression method (Touw and Hoogduin, 2011).

Value

An object of class `jfaEvaluation` containing:

`n`	the sample size.
`k`	an integer specifying the number of observed errors.
`t`	a number specifying the sum of observed taints.
`confidence`	the confidence level of the result.
`popBookvalue`	if specified as input, the total book value of the population.
`pointEstimate`	if method is one of `direct`, `difference`, `quotient`, or `regression`, the value of the point estimate.
`lowerBound`	if method is one of `direct`, `difference`, `quotient`, or `regression`, the value of the lower bound of the interval.
`upperBound`	if method is one of `direct`, `difference`, `quotient`, or `regression`, the value of the upper bound of the interval.
`confBound`	the upper confidence bound on the error percentage.
`method`	the evaluation method that was used.
`materiality`	the materiality.
`conclusion`	if `materiality` is specified, the conclusion about whether to approve or not approve the population.
`N`	if specified as input, the population size.
`populationK`	the assumed total errors in the population. Used for inferences with `hypergeometric` method.
`prior`	a logical, indicating whether a prior was used in the analysis.
`nPrior`	if a prior is specified, the prior assumed sample size.
`kPrior`	if a prior is specified, the prior assumed sample errors.
`multiplicationFactor`	if `method = "coxsnell"`, the multiplication factor for the F-distribution.
`df1`	if `method = "coxsnell"`, the df1 for the F-distribution.
`df2`	if `method = "coxsnell"`, the df2 for the F-distribution.

Author(s)

Koen Derks, <k.derks@nyenrode.nl>

References

Cox, D. and Snell, E. (1979). On sampling and the estimation of rare errors. *Biometrika*, 66(1), 125-132.

Dworin, L., and Grimlund, R. A. (1986). Dollar-unit sampling: A comparison of the quasi-Bayesian and moment bounds. *Accounting Review*, 36-57.

Leslie, D. A., Teitlebaum, A. D., & Anderson, R. J. (1979). *Dollar-unit sampling: a practical guide for auditors*. Copp Clark Pitman; Belmont, Calif.: distributed by Fearon-Pitman.

Meikle, G. R. (1972). *Statistical Sampling in an Audit Context: An Audit Technique*. Canadian Institute of Chartered Accountants.

Pap, G., and van Zuijlen, M. C. (1996). On the asymptotic behavior of the Stringer bound 1. *Statistica Neerlandica*, 50(3), 367-389.

Rohrbach, K. J. (1993). Variance augmentation to achieve nominal coverage probability in sampling from audit populations. *Auditing*, 12(2), 79.

Stringer, K. W. (1963). Practical aspects of statistical sampling in auditing. *In Proceedings of the Business and Economic Statistics Section* (pp. 405-411). American Statistical Association.

Touw, P., and Hoogduin, L. (2011). *Statistiek voor Audit en Controlling*. Boom uitgevers Amsterdam.

See Also

auditPrior planning sampling

Examples

```
library(jfa)
set.seed(1)

# Generate some audit data (N = 1000):
data <- data.frame(ID = sample(1000:100000, size = 1000, replace = FALSE),
                   bookValue = runif(n = 1000, min = 700, max = 1000))

# Using monetary unit sampling, draw a random sample from the population.
s1 <- sampling(population = data, sampleSize = 100, units = "mus",
               bookValues = "bookValue", algorithm = "random")
s1_sample <- s1$sample
s1_sample$trueValue <- s1_sample$bookValue
s1_sample$trueValue[2] <- s1_sample$trueValue[2] - 500 # One overstatement is found

# Using summary statistics, calculate the upper confidence bound according
# to the binomial distribution:

e1 <- evaluation(nSumstats = 100, kSumstats = 1, method = "binomial",
                 materiality = 0.05)
print(e1)
```

```
# jfa evaluation results for binomial method:
#
# Materiality:          5%
# Confidence:           95%
# Upper bound:          4.656%
# Sample size:          100
# Sample errors:        1
# Sum of taints:        1
# Conclusion:           Approve population

# Evaluate the raw sample using the stringer bound:

e2 <- evaluation(sample = s1_sample, bookValues = "bookValue", auditValues = "trueValue",
                 method = "stringer", materiality = 0.05)
print(e2)

# jfa evaluation results for stringer method:
#
# Materiality:          5%
# Confidence:           95%
# Upper bound:          3.952%
# Sample size:          100
# Sample errors:        1
# Sum of taints:        0.587
# Conclusion:           Approve population
```

planning *Frequentist and Bayesian Planning for Audit Samples*

Description

This function calculates the required sample size for an audit, based on the poisson, binomial, or hypergeometric likelihood. A prior can be specified to perform Bayesian planning. The returned object is of class `jfaPlanning` and can be used with associated `print()` and `plot()` methods.

Usage

```
planning(materiality, confidence = 0.95, expectedError = 0, likelihood = "poisson",
         N = NULL, maxSize = 5000, prior = FALSE, kPrior = 0, nPrior = 0)
```

Arguments

`materiality`	a value between 0 and 1 representing the materiality of the audit as a fraction of the total size or value.
`confidence`	the confidence level desired from the confidence bound (on a scale from 0 to 1). Defaults to 0.95, or 95% confidence.

`expectedError`	a fraction representing the percentage of expected mistakes in the sample relative to the total size, or a number (>= 1) that represents the number of expected mistakes.
`likelihood`	can be one of `binomial`, `poisson`, or `hypergeometric`.
`N`	the population size (required for hypergeometric calculations).
`maxSize`	the maximum sample size that is considered for calculations. Defaults to 5000 for efficiency. Increase this value if the sample size cannot be found due to it being too large (e.g., for a low materiality).
`prior`	whether to use a prior distribution when planning. Defaults to `FALSE` for frequentist planning. If `TRUE`, the prior distribution is updated by the specified likelihood. Chooses a conjugate gamma distribution for the Poisson likelihood, a conjugate beta distribution for the binomial likelihood, and a conjugate beta-binomial distribution for the hypergeometric likelihood.
`kPrior`	the prior parameter α (number of errors in the assumed prior sample).
`nPrior`	the prior parameter β (total number of observations in the assumed prior sample).

Details

This section elaborates on the available likelihoods and corresponding prior distributions for the `likelihood` argument.

- `poisson`: The Poisson likelihood is used as a likelihood for monetary unit sampling (MUS). Its likelihood function is defined as:
$$p(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$
The conjugate *gamma(α, β)* prior has probability density function:
$$f(x; \alpha, \beta) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$$
- `binomial`: The binomial likelihood is used as a likelihood for record sampling *with* replacement. Its likelihood function is defined as:
$$p(x) = \binom{n}{k} p^k (1-p)^{n-k}$$
The conjugate *beta(α, β)* prior has probability density function:
$$f(x; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1} (1-x)^{\beta-1}$$
- `hypergeometric`: The hypergeometric likelihood is used as a likelihood for record sampling *without* replacement. Its likelihood function is defined as:
$$p(x = k) = \frac{\binom{K}{k}\binom{N-K}{n-k}}{\binom{N}{n}}$$
The conjugate *beta-binomial(α, β)* prior (Dyer and Pierce, 1993) has probability density function:
$$f(k|n, \alpha, \beta) = \binom{n}{k} \frac{B(k+\alpha, n-k+\beta)}{B(\alpha, \beta)}$$

Value

An object of class `jfaPlanning` containing:

`materiality`	the value of the specified materiality.
`confidence`	the confidence level for the desired population statement.
`sampleSize`	the resulting sample size.
`expectedSampleError`	the number of full errors that are allowed to occur in the sample.
`expectedError`	the specified number of errors as a fraction or as a number.
`likelihood`	the specified likelihood.
`errorType`	whether the expected errors where specified as a percentage or as an integer.
`N`	the population size (only returned in case of a hypergeometric likelihood).
`populationK`	the assumed population errors (only returned in case of a hypergeometric likelihood).
`prior`	a list containing information on the prior parameters.

Author(s)

Koen Derks, <`k.derks@nyenrode.nl`>

References

Dyer, D. and Pierce, R.L. (1993). On the Choice of the Prior Distribution in Hypergeometric Sampling. *Communications in Statistics - Theory and Methods*, 22(8), 2125 - 2146.

See Also

`auditPrior` `sampling` `evaluation`

Examples

```
library(jfa)

# Using the binomial distribution, calculates the required sample size for a
# materiality of 5% when 2.5% mistakes are expected to be found in the sample.

# Frequentist planning with binomial likelihood:

p1 <- planning(materiality = 0.05, confidence = 0.95, expectedError = 0.025,
               likelihood = "binomial")
print(p1)

# jfa planning results for binomial likelihood:
#
# Materiality:             5%
# Confidence:              95%
# Sample size:             234
# Allowed sample errors:   6
# Bayesian planning with uninformed prior:

p2 <- planning(materiality = 0.05, confidence = 0.95, expectedError = 0.025,
               likelihood = "binomial", prior = TRUE)
print(p2)

# jfa planning results for beta prior with binomial likelihood:
#
# Materiality:             5%
# Confidence:              95%
# Sample size:             220
# Allowed sample errors:   5.5
# Prior parameter alpha:   1
# Prior parameter beta:    1

# Bayesian planning with informed prior:

prior <- auditPrior(materiality = 0.05, confidence = 0.95, cr = 0.6,
                    expectedError = 0.025, likelihood = "binomial")

p3 <- planning(materiality = 0.05, confidence = 0.95, expectedError = 0.025,
               prior = prior)
print(p3)

# jfa planning results for beta prior with binomial likelihood:
#
# Materiality:             5%
# Confidence:              95%
# Sample size:             169
# Allowed sample errors:   4.23
# Prior parameter alpha:   2.275
# Prior parameter beta:    50.725
```

`sampling` *Sampling from Audit Populations*

Description

This function takes a data frame and performs sampling according to one of three popular algorithms: random sampling, cell sampling, or fixed interval sampling. Sampling is done in combination with one of two sampling units: records or monetary units The returned object is of class `jfaSampling` and can be used with associated `print()` and `plot()` methods.

Usage

```
sampling(population, sampleSize, bookValues = NULL, units = "records",
         algorithm = "random", intervalStartingPoint = 1, ordered = TRUE,
         ascending = TRUE, withReplacement = FALSE, seed = 1)
```

Arguments

`population`	a data frame containing the population the auditor wishes to sample from.
`sampleSize`	the number of observations that need to be selected from the population. Can also be an object of class `jfaPlanning`.
`bookValues`	a character specifying the name of the column containing the book values (as in the population data).
`units`	can be either `records` (default) for record sampling, or `mus` for monetary unit sampling.
`algorithm`	can be either one of `random` (default) for random sampling, `cell` for cell sampling, or `interval` for fixed interval sampling.
`intervalStartingPoint`	the starting point in the interval (used only in fixed interval sampling)
`ordered`	if `TRUE` (default), the population is first ordered according to the value of their book values.
`ascending`	if `TRUE` (default), order the population in ascending order.
`withReplacement`	whether sampling should be performed with replacement. Defaults to `FALSE`.
`seed`	seed to reproduce results. Default is 1.

Details

This first part of this section elaborates on the possible options for the `units` argument:

- `records`: In record sampling, each observation in the population is seen as a sampling unit. An observation of $5000 is therefore equally likely to be selected as an observation of $500.
- `mus`: In monetary unit sampling, each monetary unit in the population is seen as a sampling unit. An observation of $5000 is therefore ten times more likely to be selected as an observation of $500.

This second part of this section elaborates on the possible options for the `algorithm` argument:

- `random`: In random sampling each sampling unit in the population is drawn with equal probability.
- `cell`: In cell sampling the sampling units in the population are divided into a number (equal to the sample size) of intervals. From each interval one sampling unit is selected with equal probability.
- `interval`: In fixed interval sampling the sampling units in the population are divided into a number (equal to the sample size) of intervals. From each interval one sampling unit is selected according to a fixed starting point (`intervalStartingPoint`).

Value

An object of class `jfaSampling` containing:

`population`	a data frame containing the input population.

`sample`	a data frame containing the selected observations.
`bookValues`	if specified, the name of the specified book value column.
`algorithm`	the algorithm that was used for sampling.
`units`	the sampling units that were used for sampling.

Author(s)

Koen Derks, <`k.derks@nyenrode.nl`>

References

Wampler, B., & McEacharn, M. (2005). Monetary-unit sampling using Microsoft Excel. *The CPA journal*, 75(5), 36.

See Also

`auditPrior` `planning` `evaluation`

Examples

```
library(jfa)
set.seed(1)

# Generate some audit data (N = 1000).
population <- data.frame(ID = sample(1000:100000, size = 1000, replace = FALSE),
                         bookValue = runif(n = 1000, min = 700, max = 1000))

# Draw a custom sample of 100 from the population (via random record sampling):

s1 <- sampling(population = population, sampleSize = 100, algorithm = "random",
               units = "records", seed = 1)
print(s1)

# jfa sampling results for random random record sampling:
#
# Population size:         1000
# Sample size:             100
# Proportion n/N:          0.1

# Use the result from the planning stage in the sampling stage:

p1 <- planning(materiality = 0.05, confidence = 0.95, expectedError = 0.025,
               likelihood = "binomial")

# Draw a sample via random monetary unit sampling:
s2 <- sampling(population = population, sampleSize = p1, algorithm = "random",
               units = "mus", seed = 1, bookValues = "bookValue")
print(s2)

# jfa sampling results for random monetary unit sampling:
#
```

```
# Population size:         1000
# Sample size:             234
# Proportion n/N:          0.234
# Percentage of value:     23.3%
```

Index