Package ‘jfa’

June 16, 2022

Title Bayesian and Classical Audit Sampling

Version 0.6.4

Date 2022-06-16

Description Provides statistical audit sampling methods as implemented in JASP for Audit (Derks et al., 2021 <doi:10.21105/joss.02733>). The package makes it easy for an auditor to plan a statistical sample, select the sample from the population, and evaluate the misstatement in the sample compliant with the International Standards on Auditing. Next to classical audit sampling methodology, the package implements Bayesian equivalents of these methods whose statistical underpinnings are described in Derks et al. (2021) <doi:10.1111/ijau.12240>, Derks et al. (2021) <doi:10.31234/osf.io/kzqp5>, and Derks et al. (2

BugReports `https://github.com/koenderks/jfa/issues`

URL `https://koenderks.github.io/jfa/`, `https://github.com/koenderks/jfa`

Imports extraDistr, graphics, stats

Suggests kableExtra, knitr, MUS, rmarkdown, samplingbook, testthat

Language en-US

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

VignetteBuilder knitr

NeedsCompilation no

Author Koen Derks [aut, cre] (<`https://orcid.org/0000-0002-5533-9349`>)

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

Repository CRAN

Date/Publication 2022-06-16 16:10:09 UTC

R topics documented:

`jfa-package` *jfa — Bayesian and Classical Audit Sampling*

Description

`jfa` is an R package for statistical audit sampling. The package provides functions for planning, performing, evaluating, and reporting an audit sample. Specifically, these functions implement standard audit sampling techniques for calculating sample sizes, selecting items from a population, and evaluating the misstatement from a data sample or from summary statistics. Additionally, the `jfa` package allows the user to create a prior probability distribution to perform Bayesian audit sampling using these functions.

The package and its intended workflow are also implemented with a graphical user interface in the Audit module of JASP, a free and open-source statistical software program.

For documentation on `jfa` itself, including the manual and user guide for the package, worked examples, and other tutorial information visit the package website.

Reference tables

Below you can find several links to reference tables that contain statistical sample sizes, upper limits, and Bayes factors. These tables are created using the `planning()` and `evaluation()` functions provided in the package. See the corresponding help files for more information about these functions and how to replicate this output.

Sample sizes

- Sample sizes based on the Poisson distribution
- Sample sizes based on the binomial distribution
- Sample sizes based on the hypergeometric distribution

Upper limits

- Upper limits based on the Poisson distribution
- Upper limits based on the binomial distribution
- Upper limits based on the hypergeometric distribution

One-sided p values

- One sided p values based on the Poisson distribution
- One sided p values based on the binomial distribution
- One sided p values based on the hypergeometric distribution

Bayes factors

- Impartial Bayes factors based on the gamma distribution
- Impartial Bayes factors based on the beta distribution
- Impartial Bayes factors based on the beta-binomial distribution

Author(s)

Koen Derks (maintainer, author) <k.derks@nyenrode.nl>

Please use the citation provided by R when citing this package. A BibTex entry is available from `citation('jfa')`.

See Also

Useful links:

- The cheat sheet for a quick overview of the intended workflow.
- The vignettes for worked examples.
- The issue page to submit a bug report or feature request.

Examples

```
# Load the jfa package
library(jfa)

# Load the BuildIt population
data('BuildIt')

############################################
### Example 1: Classical audit sampling ####
############################################

# Stage 1: Planning
stage1 <- planning(materiality = 0.03, expected = 0.01,
                   likelihood = 'poisson', conf.level = 0.95)
summary(stage1)

# Stage 2: Selection
stage2 <- selection(data = BuildIt, size = stage1,
                    units = 'values', values = 'bookValue',
```

```
                    method = 'interval', start = 1)
summary(stage2)

# Stage 3: Execution
sample <- stage2[['sample']]

# Stage 4: Evaluation
stage4 <- evaluation(materiality = 0.03, method = 'stringer',
                     conf.level = 0.95, data = sample,
                     values = 'bookValue', values.audit = 'auditValue')
summary(stage4)

######################################################################
### Example 2: Bayesian audit sampling using a non-informed prior ####
######################################################################

# Create the prior distribution
prior <- auditPrior(method = 'default', likelihood = 'poisson')
summary(prior)

# Stage 1: Planning
stage1 <- planning(materiality = 0.03, expected = 0.01,
                   likelihood = 'poisson', conf.level = 0.95, prior = prior)
summary(stage1)

# Stage 2: Selection
stage2 <- selection(data = BuildIt, size = stage1,
                    units = 'values', values = 'bookValue',
                    method = 'interval', start = 1)
summary(stage2)

# Stage 3: Execution
sample <- stage2[['sample']]

# Stage 4: Evaluation
stage4 <- evaluation(materiality = 0.03, conf.level = 0.95, data = sample,
                     values = 'bookValue', values.audit = 'auditValue',
                     prior = prior)
summary(stage4)

###################################################################
### Example 3: Bayesian audit sampling using an informed prior ####
###################################################################

# Create the prior distribution
prior <- auditPrior(method = 'arm', likelihood = 'poisson',
                    expected = 0.01, materiality = 0.03, cr = 0.6, ir = 1)
summary(prior)

# Stage 1: Planning
stage1 <- planning(materiality = 0.03, expected = 0.01,
                   likelihood = 'poisson', conf.level = 0.95, prior = prior)
summary(stage1)
```

```
# Stage 2: Selection
stage2 <- selection(data = BuildIt, size = stage1,
                    units = 'values', values = 'bookValue',
                    method = 'interval', start = 1)
summary(stage2)

# Stage 3: Execution
sample <- stage2[['sample']]

# Stage 4: Evaluation
stage4 <- evaluation(materiality = 0.03, conf.level = 0.95, data = sample,
                     values = 'bookValue', values.audit = 'auditValue',
                     prior = prior)
summary(stage4)
```

`auditPrior` *Prior Distributions for Audit Sampling*

Description

`auditPrior()` is used to specify a prior distribution for Bayesian audit sampling. The interface allows a complete customization of the prior distribution as well as a formal translation of pre-existing audit information into a prior distribution. `auditPrior()` returns an object of class `jfaPrior` that can be subsequently used in the `planning()` and `evaluation()` functions via their `prior` argument. Objects with class `jfaPrior` can be used with associated `summary()` and `plot()` methods.

For more details on how to use this function, see the package vignette: `vignette('jfa', package = 'jfa')`

Usage

```
auditPrior(method = 'default', likelihood = c('poisson', 'binomial', 'hypergeometric'),
        N.units = NULL, alpha = NULL, beta = NULL, materiality = NULL, expected = 0,
            ir = NULL, cr = NULL, ub = NULL, p.hmin = NULL, x = NULL,
            n = NULL, factor = NULL, conf.level = 0.95)
```

Arguments

`method` a character specifying the method by which the prior distribution is constructed. Possible options are `default`, `strict`, `impartial`, `param`, `arm`, `bram`, `hyp`, `sample`, and `factor`. See the details section for more information.

`likelihood` a character specifying the likelihood for updating the prior distribution. Possible options are `poisson` (default) for a conjugate gamma prior distribution, `binomial` for a conjugate beta prior distribution, or `hypergeometric` for a conjugate beta-binomial prior distribution. See the details section for more information.

`N.units` a numeric value larger than 0 specifying the total number of units in the population. Only used for the `hypergeometric` likelihood.

`alpha`	a numeric value specifying the α parameter of the prior distribution. Only used for method `param`.
`beta`	a numeric value specifying the β parameter of the prior distribution. Only used for method `param`.
`materiality`	a numeric value between 0 and 1 specifying the performance materiality (i.e., the maximum tolerable misstatement) as a fraction of the total number of units in the population. Only used for methods `impartial`, `arm`, and `hyp`.
`expected`	a numeric value between 0 and 1 specifying the expected errors in the sample relative to the total sample size. Only used for methods `impartial`, `arm`, `bram`, and `hyp`.
`ir`	a numeric value between 0 and 1 specifying the inherent risk in the audit risk model. Only used for method `arm`.
`cr`	a numeric value between 0 and 1 specifying the internal control risk in the audit risk model. Only used for method `arm`.
`ub`	a numeric value between 0 and 1 specifying the `conf.level`-% upper bound for the prior distribution as a fraction of the total number of units in the population. Only used for method `bram`.
`p.hmin`	a numeric value between 0 and 1 specifying the prior probability of the hypothesis of tolerable misstatement (H1: $\theta <$ materiality). Only used for method `hyp`.
`x`	a numeric value larger than, or equal to, 0 specifying the sum of proportional errors (taints) in a prior sample. Only used for methods `sample` and `factor`.
`n`	a numeric value larger than 0 specifying the sample size of a prior sample. Only used for methods `sample` and `factor`.
`factor`	a numeric value between 0 and 1 specifying the weight of the prior sample. Only used for method `factor`.
`conf.level`	a numeric value between 0 and 1 specifying the confidence level.

Details

To perform Bayesian audit sampling you must assign a prior probability distribution to the parameter in the model, i.e., the population misstatement θ. The prior distribution can incorporate pre-existing audit information about θ before seeing a sample, which consequently allows for a more efficient or more accurate estimate of θ. However, the default priors used in `jfa` are purposely indifferent towards the individual values of θ in order to 'let the data speak for themselves'. Note that these default priors are a conservative choice of prior since they assume all possible misstatement to be (roughly) equally likely before seeing a data sample. It is therefore strongly recommended to construct an informed prior distribution based on pre-existing audit information if possible.

This section elaborates on the available options for the `method` argument.

- `default`: This method produces a *gamma(1, 1)*, *beta(1, 1)*, or *beta-binomial(N, 1, 1)* prior distribution. These priors are indifferent towards the possible values of the misstatement.
- `strict`: This method produces an improper *gamma(1, 0)*, *beta(1, 0)*, or *beta-binomial(N, 1, 0)* prior distribution. These prior distributions exactly match sample sizes and upper limits from classical methods.

- `impartial`: This method produces an impartial prior distribution. These prior distributions assume that tolerable misstatement ($\theta <$ materiality) and intolerable misstatement ($\theta >$ materiality) are equally likely.
- `param`: This method produces a `gamma(alpha, beta)`, `beta(alpha, beta)`, or `beta-binomial(N, alpha, beta)` prior distribution.
- `hyp`: This method translates an assessment of the prior probability for tolerable misstatement ($\theta <$ materiality) to a prior distribution.
- `arm`: This method translates an assessment of inherent risk and internal control risk (Audit Risk Model, Derks et al., 2021) to a prior distribution.
- `bram`: This method translates an assessment of the expected most likely error and x-% upper bound to a prior distribution.
- `sample`: This method translates sampling results from an earlier sample to a prior distribution.
- `factor`: This method translates and weighs sampling results from an earlier sample to a prior distribution.

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

- `poisson`: The Poisson distribution is an approximation of the binomial distribution. The Poisson distribution is defined as:

  $$f(\theta, n) = \frac{\lambda^\theta e^{-\lambda}}{\theta!}$$

  The conjugate *gamma*(α, β) prior has probability density function:

  $$p(\theta; \alpha, \beta) = \frac{\beta^\alpha \theta^{\alpha-1} e^{-\beta\theta}}{\Gamma(\alpha)}$$

- `binomial`: The binomial distribution is an approximation of the hypergeometric distribution. The binomial distribution is defined as:

  $$f(\theta, n, x) = \binom{n}{x} \theta^x (1-\theta)^{n-x}$$

  The conjugate *beta*(α, β) prior has probability density function:

  $$p(\theta; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

- `hypergeometric`: The hypergeometric distribution is defined as:

  $$f(x, n, K, N) = \frac{\binom{K}{x}\binom{N-K}{n-x}}{\binom{N}{n}}$$

  The conjugate *beta-binomial*(α, β) prior (Dyer and Pierce, 1993) has probability mass function:

  $$f(x, n, \alpha, \beta) = \binom{n}{x} \frac{B(x+\alpha, n-x+\beta)}{B(\alpha, \beta)}$$

Value

An object of class `jfaPrior` containing:

- `prior` — a string describing the functional form of the prior distribution.
- `description` — a list containing a description of the prior distribution, including the parameters of the prior distribution and the implicit sample on which the prior distribution is based.
- `statistics` — a list containing statistics of the prior distribution, including the mean, mode, median, and upper bound of the prior distribution.
- `specifics` — a list containing specifics of the prior distribution that vary depending on the `method`.
- `hypotheses` — if `materiality` is specified, a list containing information about the hypotheses, including prior probabilities and odds for the hypothesis of tolerable misstatement (H1) and the hypothesis of intolerable misstatement (H0).
- `method` — a character indicating the method by which the prior distribution is constructed.
- `likelihood` — a character indicating the likelihood of the data.
- `materiality` — if `materiality` is specified, a numeric value between 0 and 1 giving the materiality used to construct the prior distribution.
- `expected` — a numeric value larger than, or equal to, 0 giving the input for the number of expected errors.
- `conf.level` — a numeric value between 0 and 1 giving the confidence level.
- `N.units` — if `N.units` is specified, the number of units in the population.

Author(s)

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

References

Derks, K., de Swart, J., Wagenmakers, E.-J., Wille, J., & Wetzels, R. (2021). JASP for audit: Bayesian tools for the auditing practice. *Journal of Open Source Software*, *6*(68), 2733.

Derks, K., de Swart, J., van Batenburg, P., Wagenmakers, E.-J., & Wetzels, R. (2021). Priors in a Bayesian audit: How integration of existing information into the prior distribution can improve audit transparency and efficiency. *International Journal of Auditing*, *25*(3), 621-636.

See Also

`planning` `selection` `evaluation` `report`

Examples

```
# Default uniform beta(1, 1) prior distribution
auditPrior(method = "default", likelihood = "binomial")

# Translate inherent risk (ir) and control risk (cr) to a gamma prior distribution
auditPrior(method = "arm", expected = 0.025, materiality = 0.05, ir = 1, cr = 0.6)
```

```
# Impartial beta prior distribution (equal prior probabilities)
auditPrior(method = "impartial", likelihood = "binomial", materiality = 0.05)
```

`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)
```

`carrier` *Carrier Company Financial Statements*

Description

Fictional data from a carrier company in Europe, containing 202 ledger items across 10 company entities.

Usage

```
data(carrier)
```

Format

A data frame with 202 rows and 12 variables.

description description of the ledger item.

entity1 recorded values for entity 1, in US dollars.

entity2 recorded values for entity 2, in US dollars.

entity3 recorded values for entity 3, in US dollars.

entity4 recorded values for entity 4, in US dollars.

entity5 recorded values for entity 5, in US dollars.

entity6 recorded values for entity 6, in US dollars.

entity7 recorded values for entity 7, in US dollars.

entity8 recorded values for entity 8, in US dollars.

entity9 recorded values for entity 9, in US dollars.

entity10 recorded values for entity 10, in US dollars.

total total value, in US dollars.

Source

https://towardsdatascience.com/data-driven-audit-1-automated-sampling-using-python-52e83347add5

Examples

```
data(carrier)
```

`evaluation` *Evaluate a Statistical Audit Sample*

Description

`evaluation()` is used to perform statistical inference about the misstatement in an audit population. It allows specification of statistical requirements for the sample with respect to the performance materiality or the precision. `evaluation()` returns an object of class `jfaEvaluation` which can be used with associated `summary()` and `plot()` methods.

For more details on how to use this function, see the package vignette: `vignette('jfa', package = 'jfa')`

Usage

```
evaluation(materiality = NULL, min.precision = NULL, method = 'poisson',
           alternative = c('less', 'two.sided', 'greater'), conf.level = 0.95,
           data = NULL, values = NULL, values.audit = NULL, times = NULL,
           x = NULL, n = NULL, N.units = NULL, N.items = NULL,
           r.delta = 2.7, m.type = 'accounts', cs.a = 1, cs.b = 3, cs.mu = 0.5,
           prior = FALSE)
```

Arguments

- `materiality` a numeric value between 0 and 1 specifying the performance materiality (i.e., the maximum tolerable misstatement) as a fraction of the total number of units in the population. Can be `NULL`, but `min.precision` should be specified in that case. Not used for methods `direct`, `difference`, `quotient`, and `regression`.
- `min.precision` a numeric value between 0 and 1 specifying the minimum precision (i.e., upper bound minus most likely error) as a fraction of the total population size. Can be `NULL`, but `materiality` should be specified in that case.
- `method` a character specifying the inference method. Possible options are `poisson` (default), `binomial`, `hypergeometric`, `mpu`, `stringer`, `stringer.meikle`, `stringer.lta`, `stringer.pvz`, `rohrbach`, `moment`, `direct`, `difference`, `quotient`, or `regression`. See the details section for more information.
- `alternative` a character indicating the alternative hypothesis and the type of confidence / credible interval. Possible options are `less` (default), `two.sided`, or `greater`.
- `conf.level` a numeric value between 0 and 1 specifying the confidence level.
- `data` a data frame containing a data sample.
- `values` a character specifying name of a column in `data` containing the book values of the items.
- `values.audit` a character specifying name of a column in `data` containing the audit (true) values of the items.
- `times` a character specifying name of a column in `data` containing the number of times each item in `data` should be counted (e.g., due to being selected multiple times for the sample).
- `x` a numeric value larger than 0 specifying the sum of (proportional) misstatements in the sample. If specified, overrides the `data`, `values` and `values.audit` arguments and assumes that the data come from summary statistics specified by both `x` and `n`.
- `n` an integer larger than 0 specifying the number of items in the sample. If specified, overrides the `data`, `values` and `values.audit` arguments and assumes that the data come from summary statistics specified by both `x` and `n`.
- `N.units` an integer larger than 0 specifying the number of units in the population. Only used for methods `hypergeometric`, `direct`, `difference`, `quotient`, and `regression`.
- `N.items` an integer larger than 0 specifying the number of items in the population. Only used for methods `direct`, `difference`, `quotient`, and `regression`.
- `r.delta` a numeric value specifying Δ in Rohrbach's augmented variance bound (Rohrbach, 1993). Only used for method `rohrbach`.
- `m.type` a character specifying the type of population (Dworin and Grimlund, 1984). Possible options are `accounts` and `inventory`. Only used for method `moment`.
- `cs.a` a numeric value specifying the α parameter of the prior distribution on the mean taint. Only used for method `coxsnell`.
- `cs.b` a numeric value specifying the β parameter of the prior distribution on the mean taint. Only used for method `coxsnell`.

`cs.mu`	a numeric value between 0 and 1 specifying the mean of the prior distribution on the mean taint. Only used for method `coxsnell`.
`prior`	a logical specifying whether to use a prior distribution, or an object of class `jfaPrior` or `jfaPosterior` containing the prior distribution. If `FALSE` (default), performs classical planning. If `TRUE`, performs Bayesian planning using a default conjugate prior.

Details

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

- `poisson`: Evaluates the sample with the Poisson distribution. If combined with `prior = TRUE`, performs Bayesian evaluation using a *gamma* prior and posterior.
- `binomial`: Evaluates the sample with the binomial distribution. If combined with `prior = TRUE`, performs Bayesian evaluation using a *beta* prior and posterior.
- `hypergeometric`: Evaluates the sample with the hypergeometric distribution. If combined with `prior = TRUE`, performs Bayesian evaluation using a *beta-binomial* prior and posterior.
- `mpu`: Evaluates the sample with the mean-per-unit estimator.
- `stringer`: Evaluates the sample with the Stringer bound (Stringer, 1963).
- `stringer.meikle`: Evaluates the sample with the Stringer bound with Meikle's correction for understatements (Meikle, 1972).
- `stringer.lta`: Evaluates the sample with the Stringer bound with LTA correction for understatements (Leslie, Teitlebaum, and Anderson, 1979).
- `stringer.pvz`: Evaluates the sample with the Stringer bound with Pap and van Zuijlen's correction for understatements (Pap and van Zuijlen, 1996).
- `rohrbach`: Evaluates the sample with Rohrbach's augmented variance bound (Rohrbach, 1993).
- `moment`: Evaluates the sample with the modified moment bound (Dworin and Grimlund, 1984).
- `coxsnell`: Evaluates the sample with the Cox and Snell bound (Cox and Snell, 1979).
- `direct`: Evaluates the sample with the direct estimator (Touw and Hoogduin, 2011).
- `difference`: Evaluates the sample with the difference estimator (Touw and Hoogduin, 2011).
- `quotient`: Evaluates the sample with the quotient estimator (Touw and Hoogduin, 2011).
- `regression`: Evaluates the sample with the regression estimator (Touw and Hoogduin, 2011).

Value

An object of class `jfaEvaluation` containing:

`conf.level`	a numeric value between 0 and 1 giving the confidence level.
`mle`	a numeric value between 0 and 1 giving the most likely error in the population.
`ub`	a numeric value between 0 and 1 giving the upper bound for the population misstatement.

`lb`	a numeric value between 0 and 1 giving the lower bound for the population misstatement.
`precision`	a numeric value between 0 and 1 giving the difference between the most likely error and the upper bound.
`p.value`	for classical tests, a numeric value giving the one-sided p-value.
`x`	an integer larger than, or equal to, 0 giving the number of sample errors.
`t`	a value larger than, or equal to, 0, giving the sum of proportional sample errors.
`n`	an integer larger than 0 giving the sample size.
`materiality`	if `materiality` is specified, a numeric value between 0 and 1 giving the performance materiality as a fraction of the number of units in the population.
`min.precision`	if `min.precision` is specified, a numeric value between 0 and 1 giving the minimum precision as a fraction of the number of units in the population.
`alternative`	a character indicating the alternative hypothesis.
`method`	a character indicating the inference method.
`N.units`	if `N.units` is specified, in integer larger than 0 indicating the number of units in the population.
`N.items`	if `N.items` is specified, in integer larger than 0 indicating the number of items in the population.
`K`	if `method = 'hypergeometric'`, an integer indicating the assumed total errors in the population.
`prior`	an object of class 'jfaPrior' that contains the prior distribution.
`posterior`	an object of class 'jfaPosterior' that contains the posterior distribution.
`data`	a data frame containing the relevant columns from the `data`.
`data.name`	a character giving the name of the data.

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.

Derks, K., de Swart, J., van Batenburg, P., Wagenmakers, E.-J., & Wetzels, R. (2021). Priors in a Bayesian audit: How integration of existing information into the prior distribution can improve audit transparency and efficiency. *International Journal of Auditing*, 25(3), 621-636.

Dworin, L. D. and Grimlund, R. A. (1984). Dollar-unit sampling for accounts receivable and inventory. *The Accounting Review*, 59(2), 218–241

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. *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 selection report

Examples

```
data("BuildIt")

# Draw a sample of 100 monetary units from the population using
# fixed interval monetary unit sampling
sample <- selection(
  data = BuildIt, size = 100, units = "values",
  method = "interval", values = "bookValue"
)$sample

# Classical evaluation using the Stringer bound
evaluation(
  materiality = 0.05, method = "stringer", conf.level = 0.95,
  data = sample, values = "bookValue", values.audit = "auditValue"
)

# Classical evaluation using the Poisson likelihood
evaluation(
  materiality = 0.05, method = "poisson", conf.level = 0.95,
  data = sample, values = "bookValue", values.audit = "auditValue"
)

# Bayesian evaluation using a noninformative gamma prior distribution
evaluation(
  materiality = 0.05, method = "poisson", conf.level = 0.95,
  data = sample, values = "bookValue", values.audit = "auditValue",
  prior = TRUE
)

# Bayesian evaluation using an informed prior distribution
evaluation(
  materiality = 0.05, method = "poisson", conf.level = 0.95,
  data = sample, values = "bookValue", values.audit = "auditValue",
  prior = auditPrior(method = "param", alpha = 1, beta = 10)
)
```

jfa-methods *Methods for jfa objects*

Description

Methods defined for objects returned from the `auditPrior`, `planning`, `selection`, and `evaluation` functions.

Usage

```
## S3 method for class 'jfaPrior'
print(x, ...)

## S3 method for class 'jfaPosterior'
print(x, ...)

## S3 method for class 'jfaPlanning'
print(x, ...)

## S3 method for class 'jfaSelection'
print(x, ...)

## S3 method for class 'jfaEvaluation'
print(x, digits = getOption("digits"), ...)

## S3 method for class 'summary.jfaPrior'
print(x, digits = getOption("digits"), ...)

## S3 method for class 'summary.jfaPosterior'
print(x, digits = getOption("digits"), ...)

## S3 method for class 'summary.jfaPlanning'
print(x, digits = getOption("digits"), ...)

## S3 method for class 'summary.jfaSelection'
print(x, digits = getOption("digits"), ...)

## S3 method for class 'summary.jfaEvaluation'
print(x, digits = getOption("digits"), ...)

## S3 method for class 'jfaPrior'
summary(object, digits = getOption("digits"), ...)

## S3 method for class 'jfaPosterior'
summary(object, digits = getOption("digits"), ...)

## S3 method for class 'jfaPlanning'
```

```
summary(object, digits = getOption("digits"), ...)

## S3 method for class 'jfaSelection'
summary(object, digits = getOption("digits"), ...)

## S3 method for class 'jfaEvaluation'
summary(object, digits = getOption("digits"), ...)

## S3 method for class 'jfaPrior'
predict(object, n, lim = NULL, cumulative = FALSE, ...)

## S3 method for class 'jfaPosterior'
predict(object, n, lim = NULL, cumulative = FALSE, ...)

## S3 method for class 'jfaPrior'
plot(x, xlim = c(0, 1), ...)

## S3 method for class 'jfaPosterior'
plot(x, xlim = c(0, 1), ...)

## S3 method for class 'jfaPlanning'
plot(x, xlim = c(0, 1), ...)

## S3 method for class 'jfaSelection'
plot(x, ...)

## S3 method for class 'jfaEvaluation'
plot(x, xlim = c(0, 1), ...)
```

Arguments

`...`	further arguments, currently ignored.
`digits`	an integer specifying the number of digits to which output should be rounded. Used in `summary`.
`object, x`	an object of class `jfaPrior`, `jfaPosterior`, `jfaPlanning`, `jfaSelection`, or `jfaEvaluation`.
`n`	used in `predict`. Specifies the sample size for which predictions should be made.
`lim`	used in `predict`. Limits the number of errors for which predictions should be made.
`cumulative`	used in `predict`. Specifies whether cumulative probabilities should be shown.
`xlim`	used in `plot`. Specifies the x limits (x1, x2) of the plot.

Value

The `summary` methods return a `data.frame` which contains the input and output.

The `print` methods simply print and return nothing.

planning *Plan a Statistical Audit Sample*

Description

planning() is used to calculate a minimum sample size for audit samples. It allows specification of statistical requirements for the sample with respect to the performance materiality or the precision. planning() returns an object of class jfaPlanning which can be used with associated summary() and plot() methods.

For more details on how to use this function, see the package vignette: vignette('jfa', package = 'jfa')

Usage

```
planning(materiality = NULL, min.precision = NULL, expected = 0,
         likelihood = c('poisson', 'binomial', 'hypergeometric'),
         conf.level = 0.95, N.units = NULL, by = 1, max = 5000,
         prior = FALSE)
```

Arguments

materiality	a numeric value between 0 and 1 specifying the performance materiality (i.e., the maximum tolerable misstatement) as a fraction of the total number of units in the population. Can be NULL, but min.precision should be specified in that case.
min.precision	a numeric value between 0 and 1 specifying the minimum precision (i.e., upper bound minus most likely error) as a fraction of the total population size. Can be NULL, but materiality should be specified in that case.
expected	a numeric value between 0 and 1 specifying the expected / tolerable errors in the sample relative to the total sample size, or a number (>= 1) specifying the expected / tolerable number of errors in the sample. It is advised to set this value conservatively to minimize the probability of the observed errors exceeding the expected errors, which would imply that insufficient work has been done in the end.
likelihood	a character specifying the likelihood for the data. Possible options are poisson (default) for the Poisson likelihood, binomial for the binomial likelihood, or hypergeometric for the hypergeometric likelihood. See the details section for more information about the available likelihoods.
conf.level	a numeric value between 0 and 1 specifying the confidence level.
N.units	a numeric value larger than 0 specifying the total number of units in the population. Only used for the hypergeometric likelihood.
by	an integer larger than 0 specifying the increment between possible sample sizes.
max	an integer larger than 0 specifying the sample size at which the algorithm terminates.

prior — a logical specifying whether to use a prior distribution, or an object of class `jfaPrior` or `jfaPosterior`. If `FALSE` (default), performs classical planning. If `TRUE`, performs Bayesian planning using a default conjugate prior.

Details

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

- `poisson`: The Poisson distribution is an approximation of the binomial distribution. The Poisson distribution is defined as:

  $$f(\theta, n) = \frac{\lambda^\theta e^{-\lambda}}{\theta!}$$

  The conjugate *gamma*(α, β) prior has probability density function:

  $$p(\theta; \alpha, \beta) = \frac{\beta^\alpha \theta^{\alpha-1} e^{-\beta\theta}}{\Gamma(\alpha)}$$

- `binomial`: The binomial distribution is an approximation of the hypergeometric distribution. The binomial distribution is defined as:

  $$f(\theta, n, x) = \binom{n}{x} \theta^x (1-\theta)^{n-x}$$

  The conjugate *beta*(α, β) prior has probability density function:

  $$p(\theta; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

- `hypergeometric`: The hypergeometric distribution is defined as:

  $$f(x, n, K, N) = \frac{\binom{K}{x}\binom{N-K}{n-x}}{\binom{N}{n}}$$

  The conjugate *beta-binomial*(α, β) prior (Dyer and Pierce, 1993) has probability mass function:

  $$f(x, n, \alpha, \beta) = \binom{n}{x} \frac{B(x+\alpha, n-x+\beta)}{B(\alpha, \beta)}$$

Value

An object of class `jfaPlanning` containing:

conf.level — a numeric value between 0 and 1 giving the confidence level.

x — a numeric value larger than, or equal to, 0 giving (the proportional sum of) the tolerable errors in the sample.

n — an integer larger than 0 giving the minimal sample size.

ub — a numeric value between 0 and 1 giving the expected upper bound.

precision — a numeric value between 0 and 1 giving the expected precision.

p.value	a numeric value giving the expected one-sided p-value.
K	if likelihood = 'hypergeometric', an integer larger than 0 giving the assumed population errors.
N.units	an integer larger than 0 giving the number of units in the population (only returned if N.units is specified).
materiality	a numeric value between 0 and 1 giving the performance materiality if specified.
min.precision	a numeric value between 0 and 1 giving the minimum precision if specified.
expected	a numeric value larger than, or equal to, 0 giving the expected errors input.
likelihood	a character indicating the likelihood.
errorType	a character indicating whether the expected errors input type.
iterations	an integer giving the number of iterations of the algorithm.
prior	if a prior distribution is specified, an object of class jfaPrior that contains information about the prior distribution.
posterior	if a prior distribution is specified, an object of class jfaPosterior that contains information about the expected posterior distribution.

Author(s)

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

References

Derks, K., de Swart, J., van Batenburg, P., Wagenmakers, E.-J., & Wetzels, R. (2021). Priors in a Bayesian audit: How integration of existing information into the prior distribution can improve audit transparency and efficiency. *International Journal of Auditing*, 25(3), 621-636.

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 selection evaluation report

Examples

```
# Classical planning using a Poisson likelihood
planning(materiality = 0.03, expected = 0.01, likelihood = "poisson")

# Bayesian planning using a noninformative beta prior distribution
planning(
  materiality = 0.05, expected = 0.025, likelihood = "binomial",
  prior = TRUE
)

# Bayesian planning using an impartial gamma prior distribution
planning(
  materiality = 0.05, expected = 0, likelihood = "poisson",
  prior = auditPrior(method = "impartial", materiality = 0.05)
)
```

report — *Create an Audit Sampling Report*

Description

report() takes an object of class jfaEvaluation as returned by the evaluation() function automatically generates a html or pdf report of the results.

For more details on how to use this function, see the package vignette: vignette('jfa', package = 'jfa')

Usage

```
report(object, file = 'report.html', format = c('html_document', 'pdf_document'))
```

Arguments

object	an object of class jfaEvaluation as returned by the evaluation() function.
file	a character specifying the name of the report (e.g. report.html).
format	a character specifying the output format of the report. Possible options are html_document (default) and pdf_document, but compiling to pdf format requires a local version of MikTex.

Value

A html or pdf file containing a report of the evaluation.

Author(s)

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

See Also

auditPrior planning selection evaluation

Examples

```
data("BuildIt")

# Draw a sample of 100 monetary units from the population using
# fixed interval monetary unit sampling
sample <- selection(
  data = BuildIt, size = 100, method = "interval",
  units = "values", values = "bookValue"
)$sample

# Evaluate using the Stringer bound
result <- evaluation(
  conf.level = 0.95, materiality = 0.05, method = "stringer",
```

```
  data = sample, values = "bookValue", values.audit = "auditValue"
)
## Not run:
report(result)

## End(Not run)
```

---

selection *Select a Statistical Audit Sample*

Description

`selection()` is used to perform statistical selection of audit samples. It offers flexible implementations of the most common audit sampling algorithms for attributes sampling and monetary unit sampling. `selection()` returns an object of class `jfaSelection` which can be used with associated `summary()` and a `plot()` methods.

For more details on how to use this function, see the package vignette: `vignette('jfa', package = 'jfa')`

Usage

```
selection(data, size, units = c('items', 'values'),
          method = c('interval', 'cell', 'random', 'sieve'), values = NULL,
          order = NULL, decreasing = FALSE, randomize = FALSE,
          replace = FALSE, start = 1)
```

Arguments

`data`	a data frame containing the population data.
`size`	an integer larger than 0 specifying the number of units to select. Can also be an object of class `jfaPlanning`.
`units`	a character specifying the type of sampling units. Possible options are `items` (default) for selection on the level of items (rows) or `values` for selection on the level of monetary units.
`method`	a character specifying the sampling algorithm. Possible options are `interval` (default) for fixed interval sampling, `cell` for cell sampling, `random` for random sampling, or `sieve` for modified sieve sampling.
`values`	a character specifying the name of a column in `data` containing the book values of the items.
`order`	a character specifying the name of a column in `data` containing the ranks of the items. The items in the `data` are ordered according to these values in the order indicated by `decreasing`.
`decreasing`	a logical specifying whether to order the items from smallest to largest. Only used if `order` is specified.

`randomize`	a logical specifying if items should be randomly shuffled prior to selection. Note that `randomize = TRUE` overrules `order`.
`replace`	a logical specifying if sampling units should be selected with replacement. Only used for method `random`.
`start`	an integer larger than 0 specifying index of the unit that should be selected. Only used for method `interval`.

Details

This section elaborates on the possible options for the `units` argument:

- `items`: In attributes sampling each item in the population is a sampling unit. An item with a book value of $5000 is therefore equally likely to be selected as an item with a book value of $500.
- `values`: In monetary unit sampling each monetary unit in the population is a sampling unit. An item with a book value of $5000 is therefore ten times more likely to be selected as an item with a book value of $500.

This section elaborates on the possible options for the `method` argument:

- `interval`: In fixed interval sampling the sampling units are divided into a number of equally large intervals. In each interval, a single sampling unit is selected according to a fixed starting point (specified by `start`).
- `cell`: In cell sampling the sampling units in the population are divided into a number (equal to the sample size) of equally large intervals. In each interval, a single sampling unit is selected randomly.
- `random`: In random sampling all sampling units are drawn with equal probability.
- `sieve`: In modified sieve sampling items are selected with the largest sieve ratio (Hoogduin, Hall, & Tsay, 2010).

Value

An object of class `jfaSelection` containing:

`data`	a data frame containing the population data.
`sample`	a data frame containing the selected data sample.
`n.req`	an integer giving the requested sample size.
`n.units`	an integer giving the number of obtained sampling units.
`n.items`	an integer giving the number of obtained sample items.
`N.units`	an integer giving the number of sampling units in the population data.
`N.items`	an integer giving the number of items in the population data.
`interval`	if `method = 'interval'`, a numeric value giving the size of the selection interval.
`units`	a character indicating the type of sampling units.
`method`	a character indicating the sampling algorithm.

`values`	if `values` is specified, a character indicating the book value column.
`start`	if `method = 'interval'`, an integer giving the index of the selected unit in each interval.
`data.name`	a character indicating the name of the population data.

Author(s)

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

References

Hoogduin, L. A., Hall, T. W., & Tsay, J. J. (2010). Modified sieve sampling: A method for single- and multi-stage probability-proportional-to-size sampling. *Auditing: A Journal of Practice & Theory*, 29(1), 125-148.

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.

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

See Also

`auditPrior` `planning` `evaluation` `report`

Examples

```
data("BuildIt")

# Select 100 items using random sampling
selection(data = BuildIt, size = 100, method = "random")

# Select 150 monetary units using fixed interval sampling
selection(
  data = BuildIt, size = 150, units = "values",
  method = "interval", values = "bookValue"
)
```

Index