Package: deltapif (via r-universe)

May 14, 2026

Title Estimate Potential Impact and Population Attributable Fractions with Aggregated Data

Version 0.4.6

Description Uses the delta-method to estimate the Potential Impact Fraction (PIF) and the Population Attributable Fraction (PAF) from summary data. It creates point-estimates, confidence intervals, and estimates of the variance. Provides an extension to the aggregated data method in Chan, Zepeda-Tello et al (2025) <doi:10.1002/sim.70214>.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports cli, Deriv, S7, scales

URL https://rodrigozepeda.github.io/deltapif/

Suggests EValue, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Depends R (>= 3.5)

LazyData true

Config/testthat/edition 3

Config/testthat/parallel false

Repository https://rodrigozepeda.r-universe.dev

Date/Publication 2026-03-11 18:09:08 UTC

RemoteUrl https://github.com/rodrigozepeda/deltapif

RemoteRef HEAD

RemoteSha ebe1cab40ea08f9bc69665c4f4413ccb16854e64

Contents

`alcohol` *Alcohol consumption in Australia from Pandeya et al*

Description

Alcohol consumption among Australian adults in grams/day

Usage

```
alcohol
```

Format

A data frame with 12 rows and 11 columns:

sex Whether individuals were male or female

alcohol_g Category for the measure in grams of alcohol consumption

median_intake The median intake for the group

age_18_24, age_25_34, age_35_44, age_45_54, age_55_64, age_65_74, age_75_plus, age_18_plus Proportion of adults in each of the `alcohol_g` categories (by age group).

References

Pandeya, Nirmala, et al. "Cancers in Australia in 2010 attributable to the consumption of alcohol." Australian and New Zealand journal of public health 39.5 (2015): 408-413.

See Also

dementiarisk for the relative risks and prevalence estimates

`as.data.frame` *Transform an object into a data.frame*

Description

Gets the fraction (`pif()`/`paf()`) or the cases (`averted_cases()`/`attributable_cases()`) and transforms them into a `data.frame`.

Arguments

`x`	A `pif_class` or `cases_class` object.
`...`	Additional parameters (ignored)

Value

A `data.frame` containing the fraction or cases (`value`), as well as `standard_deviation`, and confidence interval bounds `ci_low` and `ci_up`. The `label` is included to differentiate among different fractions or cases.

Examples

```
#Transform one pif
my_pif <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1,
    var_beta = 0.2, label = "My pif")
as.data.frame(my_pif)

#Transform more than one pif
my_paf <- paf(p = 0.5, beta = 1.3, var_p = 0.1, var_beta = 0.2,
    label = "My paf")
```

```
as.data.frame(my_pif, my_paf)

#Transform averted cases
cases_1 <- averted_cases(16234, my_paf)
as.data.frame(cases_1)

#Transform multiple averted cases
cases_2 <- averted_cases(87980, my_pif)
as.data.frame(cases_1, cases_2)
```

`as.list` *Convert to list*

Description

Converts the object into a list.

Arguments

`x`	Either a `covariance_structure`, a `pif_atomic_class` or `pif_global_ensemble_class`
`...`	Additional parameters (ignored)

Value

The object as a list

Examples

```
#FOR POTENTIAL IMPACT FRACTIONS
#------------------------------------------------------------------------
#Potential impact fraction for women
paf_lead_women <- paf(0.27, 2.2, quiet = TRUE, var_p = 0.001,
    label = "Lead women")
paf_rad_women  <- paf(0.12, 1.2, quiet = TRUE, var_p = 0.001,
    label = "Radiation women")
paf_women      <- paf_ensemble(paf_lead_women, paf_rad_women,
    label = "Women")

as.list(paf_women)

#FOR COVARIANCE STRUCTURES
#------------------------------------------------------------------------
cov_str <- default_parameter_covariance_structure(paf_women)
as.list(cov_str)
```

as.matrix *Convert a* covariance_structure *to* matrix

Description

Transforms a covariance_structure into a matrix.

Arguments

x	A covariance_structure
default	How to fill the empty values (default = NA)
...	Additional parameters (ignored)

Details

Because each entry of a covariance structure class can be a matrix containing the covariances of different fractions, the as.matrix command flattens that matrix into a single object.

Value

A matrix with the flattened covariance among all the involved fractions

Examples

```
#Simple covariance structure to matrix
my_cov <- covariance_structure_class(
  list(
    "pif1" = list("pif1" = 0.21, "pif2" = 0.12, "pif3" = 0.31),
    "pif2" = list("pif1" = 0.12, "pif2" = 0.33, "pif3" = -0.01),
    "pif3" = list("pif1" = 0.31, "pif2" = -0.01, "pif3" = 0.80)
  )
)
as.matrix(my_cov)

#More complicated example: the covariance matrix is flattened
mat <- matrix(c(0.1, 0.21, 0.47, -0.3), ncol = 2)
vec <- c(0.22, -0.9, 0.01)

cov2 <- covariance_structure_class(
  list(
    "pif1" = list("pif1" = 0.21, "pif2" = mat, "pif3" = 0.31),
    "pif2" = list("pif1" = mat, "pif2" = 0.33, "pif3" = vec),
    "pif3" = list("pif1" = 0.31, "pif2" = vec, "pif3" = 0.80)
  )
)
as.matrix(cov2)
```

as_covstr *Transform into a covariance structure*

Description

Transforms either a matrix or a vector into a covariance structure.

Usage

```
as_covariance_structure(x, col_names = NULL, row_names = NULL, ...)
```

Arguments

x	A matrix or a vector
col_names	Names to assign to the columns
row_names	Names to assign to the rows
...	Additional arguments (currently ignored)

Value

A covariance_structure object representing the given matrix, number, vector or data.frame as a covariance structure for using with pif(), paf(), attributable_cases(), and averted_cases()

Examples

```
mat <- matrix(c(1,3,2,4), ncol = 2,
          dimnames = list(list("pif1", "pif2"), list("pif1", "pif2")))
as_covariance_structure(mat)

#Different colnames than dimnames
as_covariance_structure(mat, col_names = c("first", "second"))

#Also with a number
as_covariance_structure(2, col_names = "col", row_names = "row")

#Or with a vector
as_covariance_structure(seq(0.1, 0.2, length.out = 4),
    row_names = c("r1","r2","r3","r4"), col_names = "col")

#As well as a data.frame
data_mat <- as.data.frame(mat)
as_covariance_structure(data_mat, row_names = c("r1","r2"))
```

cancer_rr *Relative risks for cancer from Pandeya et al*

Description

Relative risks for cancer given different levels of alcohol consumption

Usage

```
cancer_rr
```

Format

A data frame with 24 rows and 5 columns:

cancer_type The type of cancer associated to the relative risk

dose Dose in grams/day for which the relative risk was estimated

RR Relative risk for cancer given the dose of alcohol

lower_CI, upper_CI Lower and upper bounds for the 95% confidence interval

References

Pandeya, Nirmala, et al. "Cancers in Australia in 2010 attributable to the consumption of alcohol." Australian and New Zealand journal of public health 39.5 (2015): 408-413.

See Also

dementiarisk for the relative risks and prevalence estimates

casecalc *Attributable cases*

Description

Calculates the number of attributable cases or the number of cases that would be averted under a counterfactual scenario for a given fraction (either `paf` or `pif`).

Usage

```
averted_cases(
  cases,
  pif,
  variance = 0,
  conf_level = 0.95,
  link = "identity",
  link_inv = NULL,
  link_deriv = NULL
)

attributable_cases(
  cases,
  paf,
  variance = 0,
  conf_level = 0.95,
  link = "identity",
  link_inv = NULL,
  link_deriv = NULL
)
```

Arguments

cases	The overall number of cases in the population.
pif	A potential impact fraction object created by pif, paf, pif_total, pif_ensemble, paf_total or paf_ensemble.
variance	The estimated variance for the cases (default = 0).
conf_level	Confidence level for the confidence interval (default 0.95).
link	Link function such that the case confidence intervals stay within the expected bounds (either log or identity).
link_inv	(Optional). If link is a function then yhe inverse of link. For example if link is logit this should be inv_logit.
link_deriv	Derivative of the link function. The function tries to build it automatically from link using Deriv::Deriv().
paf	A population attributable fraction object created by paf, paf_total or paf_ensemble.

Details

Negative cases are interpreted as cases that would be caused by the intervention.

Value

A cases_class object with the attributable cases.

Formulas

The attributable cases are calculated as:

$$\text{Attributable cases} = \text{PAF} \times \text{Cases}$$

and the averted cases are respectively:

$$\text{Averted cases} = \text{PIF} \times \text{Cases}$$

The variance is estimated using the product-variance formula:

$$\text{Var}[\text{Averted cases}] = \sigma^2_{\text{Cases}} \cdot \big(\text{PIF}\big)^2 + \sigma^2_{\text{PIF}} \cdot \big(\text{Cases}\big)^2 + \sigma^2_{\text{PIF}} \cdot \sigma^2_{\text{Cases}}$$

See Also

`pif()`, `paf()`

Examples

```
frac <- paf(p = 0.499, beta = log(3.6), var_p = 0.002, var_beta = FALSE)
attributable_cases(100, paf = frac)

frac <- pif(p = 0.499, beta = log(3.6), p_cft = 0.1, var_p = 0.002, var_beta = FALSE)
averted_cases(100, pif = frac)
```

---

change_link *Change the link*

Description

Change the link function for the potential impact fraction or population attributable fraction to a different link.

Usage

```
change_link(x, link = "identity", link_inv = NULL, link_deriv = NULL)
```

Arguments

`x`	A `pif_class` object
`link`	Link function such that the `pif` confidence intervals stays within the expected bounds.
`link_inv`	(Optional). If `link` is a function then yhe inverse of `link`. For example if `link` is `logit` this should be `inv_logit`.
`link_deriv`	Derivative of the `link` function. The function tries to build it automatically from `link` using `Deriv::Deriv()`.

Value

A pif_class object with a different link.

Examples

```
#A potential impact fraction
pif1 <- pif(p = 0.2, p_cft = 0.1, beta = 1.2, var_p = 0.01,
  var_beta = 0.2)
pif1

#Now change the pif to logit to control the negatives
pif1_logit <- change_link(pif1, link = "logit")
pif1_logit
```

---

children *Get the children labels from a* pif_class

Description

Gets the labels of the pif elements that make up a pif_global_ensemble_class.

Usage

```
children(x, ...)
```

Arguments

x	A pif_global_ensemble_class
...	Additional arguments (currently ignored)

Value

A character vector with the names of the fractions that make up x

Note

The result is NULL if x is a pif_atomic_class

coef *Extract coefficients of a pif object*

Description

Gets the potential impact fraction value

Arguments

object	A pif_class object.
...	Additional parameters to pass to coef (ignored)

Value

A double containing point-estimate of the object.

Examples

```
my_pif <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1, var_beta = 0.2)
coef(my_pif)
```

confint *Extract confidence intervals*

Description

Gets the confidence interval for any fraction or cases.

Arguments

object	A pif_class or a cases_class.
level	Level of confidence desired.
...	Additional parameters to pass to confint (ignored)

Value

A vector containing the lower and upper bounds of the confidence interval for the object.

Examples

```
my_pif <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1, var_beta = 0.2)
#Default 95% CI
confint(my_pif)

#Custom 90% ci:
confint(my_pif, level = 0.90)
```

`covariance_structure_class` *Covariance structure class*

Description

The covariance structure class represents a collection of matrices and 0's such that `cov[[i]][[j]]` contains the covariance of elements of the i-th potential impact fraction and the j-th potential impact fraction.

Usage

```
covariance_structure_class(cov_list = list())
```

Arguments

`cov_list`	Named list such that `list[[i]][[j]]` represents the covariance between elements of the i-th potential impact fraction and the j-th potential impact fraction.

Value

An `S7` `covariance_structure_class` where entry in row `i` and column `j` represents the covariance between fraction `i` and `j`.

See Also

`as_covariance_structure()` to transform matrices to covariance structures and `covariance_structures()` for default covariance structures

Examples

```
#A simple covariance structure
cov <- covariance_structure_class(
  list(
    "pif1" = list("pif1" = 0.21, "pif2" = 0.12, "pif3" = 0.31),
    "pif2" = list("pif1" = 0.12, "pif2" = 0.33, "pif3" = -0.01),
    "pif3" = list("pif1" = 0.31, "pif2" = -0.01, "pif3" = 0.80)
  )
)
cov

#Values can be extracted as in matrices
cov[[1]][[3]]
cov[["pif1"]][["pif1"]]

#And assignment also works
cov[[1]][[3]] <- 100
cov[["pif1"]][["pif1"]] <- 500
cov
```

```
#Covariance structures are designed to contain the covariance between
#numbers or vectors (or between numbers and vectors). Hence they can
#contain matrices or vectors too:
mat <- matrix(c(0.1, 0.21, 0.47, -0.3), ncol = 2)
vec <- c(0.22, -0.9, 0.01)

cov2 <- covariance_structure_class(
  list(
    "pif1" = list("pif1" = 0.21, "pif2" = mat, "pif3" = 0.31),
    "pif2" = list("pif1" = mat, "pif2" = 0.33, "pif3" = vec),
    "pif3" = list("pif1" = 0.31, "pif2" = vec, "pif3" = 0.80)
  )
)
cov2
cov2[["pif1"]][["pif2"]]
```

`covariance_structures` *Default covariance structures*

Description

These are multidimensional arrays of lists where the entry `list[[i]][[j]]` exists represents the covariance between elements of the i-th potential impact fraction and the j-th potential impact fraction. This is particularly useful when handling ensembles and totals.

Usage

```
covariance_structure(pif, is_variance = FALSE, warning = TRUE)

covariance_structure2(pif1, pif2, warning = TRUE)

default_weight_covariance_structure(pif, is_variance = FALSE, warning = TRUE)

default_weight_covariance_structure2(pif1, pif2, warning = TRUE)

default_parameter_covariance_structure(
  pif,
  parameter = "p",
  is_variance = FALSE,
  warning = TRUE
)

default_parameter_covariance_structure2(
  pif1,
  pif2,
```

```
  parameter = "p",
  warning = TRUE
)

default_pif_covariance_structure(pif, is_variance = FALSE, warning = TRUE)

default_pif_covariance_structure2(pif1, pif2, warning = TRUE)

default_weight_pif_covariance_structure(
  pif,
  is_variance = FALSE,
  warning = TRUE
)

default_weight_pif_covariance_structure2(pif1, pif2, warning = TRUE)
```

Arguments

pif	A potential impact fraction
is_variance	Whether the covariance structure corresponds to a variance (i.e. when pif1 and pif2 are identical)
warning	Whether to throw a warning if pif1 or pif2 have common labels
pif1	A potential impact fraction to obtain a covariance structure with pif2.
pif2	A potential impact fraction to obtain a covariance structure with pif1,
parameter	Either beta or p. Indicating which parameter we are calculating covariance for.

Value

A nested list of lists with the entry [[i]][[j]] representing the covariance between elements i and j.

Note

The covariance_structures ending in 2 are meant to obtain the default covariance structure between two fractions pif1 and pif2 while the ones that don't end in 2 are meant to obtain the covariance structure of a fraction with itself.

Examples

```
pif_lead_women <- paf(0.27, 2.2, quiet = TRUE, var_p = 0.001, var_beta = 0.015,
                      label = "Women lead")
pif_rad_women  <- paf(0.12, 1.2, quiet = TRUE, var_p = 0.001, var_beta = 0.022,
                      label = "Women radiation")
pif_women      <- pif_ensemble(pif_lead_women, pif_rad_women, label = "Women",
                               weights = c(0.8, 0.72),
                               var_weights = matrix(c(0.3, 0.1, 0.1, 0.4), ncol = 2))

pif_lead_men   <- paf(0.30, 2.2, quiet = TRUE, var_p = 0.001, var_beta = 0.015,
                       label = "Men lead")
```

```
pif_rad_men    <- paf(0.10, 1.2, quiet = TRUE, var_p = 0.001, var_beta = 0.022,
                      label = "Men radiation")
pif_men        <- pif_ensemble(pif_lead_men, pif_rad_men, label = "Men",
                      weights = c(0.65, 0.68),
                      var_weights = matrix(c(0.1, -0.2, -0.2, 0.5), ncol = 2))
pif_tot        <- pif_total(pif_men, pif_women,
                      weights = c(0.49, 0.51), label = "Population",
                      var_weights = matrix(c(0.22, 0.4, 0.4, 0.8), ncol = 2))

#This is the default constructor of a covariance. Use it for custom covariances
covariance_structure(pif_lead_women)
covariance_structure2(pif_lead_women, pif_lead_men)
default_weight_covariance_structure2(pif_men, pif_men)
default_weight_covariance_structure(pif_tot)
default_weight_covariance_structure2(pif_men, pif_women)
default_parameter_covariance_structure(pif_tot, parameter = "beta")
default_parameter_covariance_structure2(pif_lead_women, pif_lead_men, parameter = "beta")
```

---

covcor *Covariance matrix, correlation matrix, variance and standard deviation for potential impact fractions*

Description

Computes the covariance (`covariance`) or correlation (`correlation`) for multiple potential impact fractions and the variance `variance` and standard deviation `standard_deviation`for a potential impact fractions.

Usage

```
covariance(
  x,
  ...,
  var_p = NULL,
  var_beta = NULL,
  var_weights = NULL,
  var_pif_weights = NULL,
  var_pifs = NULL,
  warning = FALSE
)

variance(x, ...)

standard_deviation(x, ...)

correlation(x, ..., var_p = NULL, var_beta = NULL)
```

Arguments

x	A potential impact fraction
...	Multiple additional potential impact fraction objects separated by commas.
var_p	Estimate of the link_covariance matrix of p where the entry var_p[i,j] represents the link_covariance between p[i] and p[j].
var_beta	Estimate of the link_covariance matrix of beta where the entry var_beta[i,j] represents the link_covariance between beta[i] and beta[j].
var_weights	covariance matrix for the weights when calculating the total PIF (respectively PAF) in pif_total.
var_pif_weights	Covariance vector between the weights in pif_ensemble and the pif_atomic. This refers to the term $\mathrm{Cov}\left(\hat{q}_i, \widehat{\mathrm{PIF}}_{B,j}\right)$ in the equation below. If set to NULL its automatically calculated.
var_pifs	Covariance vector between the potential impact fractions in pif_ensemble and pif_atomic. This refers to the term $\mathrm{Cov}\left(\mathrm{PIF}_{A,i}, \widehat{\mathrm{PIF}}_{B,j}\right)$ in the equation below. If set to NULL its automatically calculated.
warning	Boolean indicating whether to throw a warning if the labels on the fractions involved are not unique.

Value

A variance, covariance or correlation matrix.

Examples

```
# Get the approximate link_variance of a pif object
my_pif <- pif(p = 0.5, p_cft = 0.25, beta = 1.3,
              var_p = 0.1, var_beta = 0.2)
variance(my_pif)

# This is the same as link_covariance with just 1 pIF
covariance(my_pif)

# Calculate the link_covariance between 3 fractions with shared relative risk
beta <- 0.3
var_beta <- 0.1
pif1 <- pif(0.5, 0.2, beta, var_p = 0.5 * (1 - 0.5) / 100, var_beta = var_beta)
pif2 <- pif(0.3, 0.1, beta, var_p = 0.3 * (1 - 0.3) / 100, var_beta = var_beta)
pif3 <- pif(0.7, 0.3, beta, var_p = 0.7 * (1 - 0.7) / 100, var_beta = var_beta)
covariance(pif1, pif2, pif3)

# The link_covariance between a pif and itself only has the link_variance as entries
covariance(pif1, pif1)

# Or if there is a link_covariance structure between different betas you can specify with
# var_beta in the link_covariance
betas <- c(1.3, 1.2, 1.27)
```

```
# link_covariance among all betas
var_beta <- matrix(c(
  1.0000000, -0.12123053, 0.35429369,
  -0.1212305, 1.00000000, -0.04266409,
  0.3542937, -0.04266409, 1.00000000
), byrow = TRUE, ncol = 3)
pif1 <- pif(0.5, 0.2, betas[1], var_p = 0.5 * (1 - 0.5) / 100, var_beta = var_beta[1, 1])
pif2 <- pif(0.3, 0.1, betas[2], var_p = 0.3 * (1 - 0.3) / 100, var_beta = var_beta[2, 2])
pif3 <- pif(0.3, 0.1, betas[3], var_p = 0.3 * (1 - 0.3) / 100, var_beta = var_beta[3, 3])
covariance(pif1, pif2, pif3, var_beta = var_beta)

# Compute the correlation
correlation(pif1, pif2, pif3, var_beta = var_beta)
```

`dementiacov` *Relative risk covariance from Lee et al*

Description

Covariances between the relative risks of Lee et al

Usage

```
dementiacov
```

Format

A covariance matrix with 12 rows and 12 columns. Each entry represents the covariance between them.

References

Lee, Mark, et al. "Variation in population attributable fraction of dementia associated with potentially modifiable risk factors by race and ethnicity in the US." JAMA network open 5.7 (2022): e2219672-e2219672.

See Also

dementiarisk for the relative risks and prevalence estimates

`dementiarisk` *Exposure and Relative Risk data from Lee et Al*

Description

Relative risk and exposure data for dementia risk-factors

Usage

```
dementiarisk
```

Format

A data frame with 12 rows and 11 columns:

risk_factor The risk factor for dementia

RR The relative risk of dementia associated to the risk factor

lower_CI, upper_CI Lower and upper bounds for the 95% confidence interval

logrr The logarithm of the relative risk `log(RR)`

sdlog The variance of the logarithm of the relative risk `variance(log(RR))`

total The proportion of individuals exposed in the overall population

hispanic The proportion of hispanic individuals exposed

asian The proportion of non-hispanic asian individuals exposed

black The proportion of non-hispanic black individuals exposed

white The proportion of non-hispanic white individuals exposed

References

Lee, Mark, et al. "Variation in population attributable fraction of dementia associated with potentially modifiable risk factors by race and ethnicity in the US." JAMA network open 5.7 (2022): e2219672-e2219672.

See Also

dementiacov for the covariance between the risk factors

`derivatives` *Partial derivatives of PIF*

Description

Calculates the partial derivatives of a potential impact fraction with respect to the parameters `p` or `beta`.

Usage

```
deriv_pif_p(p, p_cft, rr, mu_obs = NULL, mu_cft = NULL)

deriv_pif_beta(p, p_cft, rr, rr_link_deriv_vals, mu_obs = NULL, mu_cft = NULL)
```

Arguments

`p`	Prevalence (proportion) of the exposed individuals for each of the `N` exposure levels.
`p_cft`	Counterfactual prevalence (proportion) of the exposed individuals for each of the `N` exposure levels.
`rr`	The relative risk for each of the exposure levels.
`mu_obs`	The average value of the relative risk in the observed population.
`mu_cft`	The average value of the counterfactual relative risk in the population.
`rr_link_deriv_vals`	The derivative of the relative risk function `g` with respect to the parameter `beta` evaluated at `beta`.

Value

The partial derivative (usually a vector)

Formulas

The partial derivative of `PIF` with respect to `p` is:

$$\frac{\partial \text{PIF}}{\partial p} = \frac{\mu^{\text{cft}}}{\left(\mu^{\text{obs}}\right)^2} \cdot \left(\text{RR}(\beta) - 1\right)$$

The partial derivative of `PIF` with respect to `beta` is:

$$\frac{\partial \text{PIF}}{\partial \beta} = \left(\frac{\mu^{\text{cft}} \cdot p - \mu^{\text{obs}} \cdot p_*}{\left(\mu^{\text{obs}}\right)^2} \right) \odot \text{RR}'(\beta)$$

with $\odot$ representing the Hadamard (elementwise) product.

Note

As `p` and `beta` are usually vectors these are vector-valued derivatives.

flatten_names *Names of a PIF's components*

Description

Returns a character vector of the names of all of the fractions that make up a `pif` object. This is particularly useful for `pif_total` and `pif_ensemble` to obtain the names that build them up.

Usage

```
flatten_names(pif)
```

Arguments

`pif`	A potential impact fraction of either `pif_atomic_class` or `pif_global_ensemble_class`

Value

A character vector with the names of all the fractions that make up the `pif`.

Examples

```
paf_lead_women <- paf(0.27, 2.2, quiet = TRUE, var_p = 0.001,
        label = "Women lead")
paf_rad_women  <- paf(0.12, 1.2, quiet = TRUE, var_p = 0.001,
        label = "Women radiation")
paf_women      <- paf_ensemble(paf_lead_women, paf_rad_women,
        label = "Women")
paf_lead_men   <- paf(0.30, 2.2, quiet = TRUE, var_p = 0.001,
        label = "Men lead")
paf_rad_men    <- paf(0.10, 1.2, quiet = TRUE, var_p = 0.001,
        label = "Men radiation")
paf_men        <- paf_ensemble(paf_lead_men, paf_rad_men,
        label = "Men")
paf_tot        <- paf_total(paf_men, paf_women, weights = c(0.49, 0.51),
        label = "Population")

#For a single PIF return the names
flatten_names(paf_lead_women)

#For an ensemble return the ones that make them up
flatten_names(paf_women)

#For totals return the ones that make them up
flatten_names(paf_tot)
```

fraction_type *Get the type of the fraction*

Description

Obtain whether a fraction is a potential impact fraction (PIF) or a population attributable fraction (PAF)

Usage

```
fraction_type(x)
```

Arguments

x	A pif_class object

Value

A character either PIF or PAF depending on the object

Examples

```
#A potential impact fraction
pif1 <- pif(p = 0.2, p_cft = 0.1, beta = 1.2, quiet = TRUE)
fraction_type(pif1)

#A population attributable fraction
paf1 <- paf(p = 0.2, beta = 1.2, quiet = TRUE)
fraction_type(paf1)
```

length *Length of an object*

Description

Gets the length of a covariance structure or a pif_class

Arguments

x	A pif_class or a covariance_structure

Value

The number of entries in a covariance_structure_class or in a pif_class (for example in an ensemble composed of multiple fractions)

Examples

```
#Calculate the length of a single fraction
pif_lead_women <- paf(0.27, 2.2, quiet = TRUE, var_p = 0.001,
  var_beta = 0.015, label = "Women lead")
length(pif_lead_women)

#Calculate the length of an ensemble
pif_rad_women  <- paf(0.12, 1.2, quiet = TRUE, var_p = 0.001,
  var_beta = 0.022, label = "Women radiation")

pif_women      <- pif_ensemble(pif_lead_women, pif_rad_women,
  label = "Women", weights = c(0.8, 0.72),
  var_weights = matrix(c(0.3, 0.1, 0.1, 0.4), ncol = 2))

length(pif_women) #= 2 as it contains 2 fractions

#Calculate the length of a covariance structure (= # cols)
length(covariance_structure2(pif_lead_women, pif_rad_women))
```

`names` *Get the label of a PIF or PAF*

Description

Gets the label of a potential impact fraction or a population attributable fraction

Arguments

`x`	A `pif_class` object.
`...`	Additional parameters (ignored)

Value

The labels of a fraction or of the fractions that make it

Note

In the case of fractions, `names` cannot be used to assign names as in `names(pif_tot) <- c("a","b")`

Examples

```
#A simple example
my_pif <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1,
                var_beta = 0.2, label = "Test")
names(my_pif)

#A pif composed of others
my_pif1 <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1,
```

```
                var_beta = 0.2, label = "Test 1")
my_pif2 <- pif(p = 0.4, p_cft = 0.1, beta = 1.3, var_p = 0.1,
                var_beta = 0.2, label = "Test 2")
pif_tot <- pif_total(my_pif1, my_pif2, weights = c(0.2, 0.8),
                label = "Parent")
names(pif_tot)
```

---

pifpaf *Potential Impact fraction and Population Attributable Fraction*

Description

Calculates the potential impact fraction `pif` or the population attributable fraction `paf` for a categorical exposure considering an observed prevalence of `p` and a relative risk (or relative risk parameter) of `beta`.

Usage

```
paf(
  p,
  beta,
  var_p = NULL,
  var_beta = NULL,
  rr_link = "exponential",
  rr_link_deriv = NULL,
  link = "log-complement",
  link_inv = NULL,
  link_deriv = NULL,
  conf_level = 0.95,
  quiet = FALSE,
  label = NULL
)

pif(
  p,
  p_cft = rep(0, length(p)),
  beta,
  var_p = NULL,
  var_beta = NULL,
  rr_link = "exponential",
  rr_link_deriv = NULL,
  link = "log-complement",
  link_inv = NULL,
  link_deriv = NULL,
  conf_level = 0.95,
  type = "PIF",
```

```
  quiet = FALSE,
  label = NULL
)
```

Arguments

p	Prevalence (proportion) of the exposed individuals for each of the N exposure levels.
beta	Relative risk parameter for which standard deviation is available (usually its either the relative risk directly or the log of the relative risk as most RRs, ORs and HRs come from exponential models).
var_p	Estimate of the link_covariance matrix of p where the entry var_p[i,j] represents the link_covariance between p[i] and p[j].
var_beta	Estimate of the link_covariance matrix of beta where the entry var_beta[i,j] represents the link_covariance between beta[i] and beta[j].
rr_link	Link function such that the relative risk is given by rr_link(beta).
rr_link_deriv	Derivative of the link function for the relative risk. The constructor tries to build it automatically from rr_link using Deriv::Deriv().
link	Link function such that the pif confidence intervals stays within the expected bounds.
link_inv	(Optional). If link is a function then yhe inverse of link. For example if link is logit this should be inv_logit.
link_deriv	Derivative of the link function. The function tries to build it automatically from link using Deriv::Deriv().
conf_level	Confidence level for the confidence interval (default 0.95).
quiet	Whether to show messages.
label	Character identifier for the impact fraction. This is for
p_cft	Counterfactual prevalence (proportion) of the exposed individuals for each of the N exposure levels.
type	Character either Potential Impact Fraction (PIF) or Population Attributable Fraction (PAF)

Value

A pif_class object with the estimate of the potential impact fraction (pif()) or the population attributable fraction (paf()).

Formulas

This function computed the potential impact fraction and its confidence intervals using Walter's formula:

$$\text{PIF} = \frac{\sum_{i=1}^{N} p_i \text{RR}_i - \sum_{i=1}^{N} p_i^{\text{cft}} \text{RR}_i}{\sum_{i=1}^{N} p_i \text{RR}_i}, \quad \text{and} \quad \text{PAF} = \frac{\sum_{i=1}^{N} p_i \text{RR}_i - 1}{\sum_{i=1}^{N} p_i \text{RR}_i}$$

in the case of `N` exposure categories which is equivalent to Levine's formula when there is only `1` exposure category:

$$\text{PIF} = \frac{p(\text{RR}-1) - p^{\text{cft}}(\text{RR}-1)}{1 + p(\text{RR}-1)} \quad \text{and} \quad \text{PAF} = \frac{p(\text{RR}-1)}{1 + p(\text{RR}-1)}$$

The construction of confidence intervals is done via a link function. Its variance is given by:

$$\sigma_f^2 = \text{Var}\Big[f(\text{PIF})\Big] \approx \Big(f'(\text{PIF})\Big)^2 \text{Var}\Big[\text{PIF}\Big]$$

and the intervals are constructed as:

$$\text{CI} = f^{-1}\Big(f(\text{PIF}) \pm Z_{\alpha/2} \cdot \sigma_f\Big)$$

the function f is called the link function for the PIF, f^{-1} represents its inverse, and f' its derivative.

Link functions for the PIF

By default the `pif` and `paf` calculations use the `log-complement` link which guarantees the impact fractions' intervals have valid values (between `-Inf` and `1`).

Depending on the application the following link functions are also implemented:

log-complement To achieve fractions between `-Inf` and `1`. This is the function `f(x) = ln(1 - x)` with inverse `finv(x) = 1 - exp(x)`

logit To achieve strictly positive fractions between `0` and `1`. This is the function `f(x) = ln(x / (1 - x))` with inverse `finv(x) = 1 / (1 + exp(-x))`

identity An approximation for fractions that does not guarantee confidence intervals in valid regions. This is the function `f(x) = x`.with inverse `finv(x) = x`

hawkins Hawkins' fraction for controlling the variance. This is the function `f(x) = ln(x + sqrt(x^2 + 1))` with inverse `finv(x) = 0.5 * exp(-x) * (exp(2 * x) - 1)`

custom User specified (see below).

In general, `logit` should be preferred if it is known and certain that the fractions can only be positive (i.e. when all relative risks (including CIs) are > 1 and prevalence > 0 and there is an epidemiological / biological justification).

Custom link functions can be implemented as long as they are invertible in the range of interest by providing:

link The function

link_inv The inverse function of `link`

link_deriv The derivative of `link`

If no derivative is provided the package attempts to estimate it symbolically using `Deriv::Deriv()` however there is no guarantee that this will work for non-standard functions (i.e. not logarithm / trigonometric / exponential)

As an example considering implementing a square root custom link:

```
# The link is square root
link      <- sqrt

# Inverse of sqrt(x) is x^2
link_inv  <- function(x) x^2

# The derivative of sqrt(x) is 1/(2 * sqrt(x))
link_deriv <- function(pif) 1 / (2 * sqrt(pif))

# Then the pif can be calculated as:
pif(p = 0.499, beta = 1.6, p_cft = 0.499/2, var_p = 0.1, var_beta = 0.2,
     link = link, link_inv = link_inv, link_deriv = link_deriv)
```

Link functions for beta

By default the `pif` and `paf` use the `exponential` link which means that the values for `beta` are the log-relative risks and the variance `var_beta` corresponds to the log-relative risk's variance. Depending on the relative risk's source the following options are available:

exponential For when the relative risks correspond to the exponential of another parameter, usually called `beta`. This is the exponential function `f(beta) = exp(beta)` with inverse `finv(rr) = log(rr) = beta`

identity For when the relative risk and their variance are reported directly. This is the function `f(beta) = beta` with inverse `finv(rr) = rr = beta`

custom User specified (see below).

Note that in most cases including contingency tables, Poisson and Logistic regressions, and Cox Proportional Hazards, the relative risks are estimated by exponentiating a parameter.

As in the previous section, custom link functions for beta can be implemented as long as they are invertible in the range of interest by providing the function `rr_link` and its derivative `rr_link_deriv`. If no derivative is provided the package does an attempt to estimate it symbolically using `Deriv::Deriv()` however there is no guarantee that this will work non-standard functions (i.e. not logarithm / trigonometric / exponential)

Population Attributable Fraction

The population attributable fraction corresponds to the potential impact fraction at the theoretical minimum risk level. It is assumed that the theoretical minimum risk level is a relative risk of 1. If no counterfactual prevalence `p_cft` is specified, the model computes the population attributable fraction.

Note

This function assumes `p` and `beta` have been pre-computed from the data and the individual-level data are not accessible to the researchers. If either the data for the individual-level prevalence of exposure `p` or the data for the individual-level risk estimate `beta` can be accessed by the researcher other methods (such as the `pifpaf` package should be preferred).

See Also

pif_total(), pif_ensemble(), paf_total(), paf_ensemble(), weighted_adjusted_paf(), weighted_adjusted_pif(), averted_cases(), attributable_cases().

Examples

```
#EXAMPLE 1: ONE EXPOSURE CATEGORY (CLASSIC LEVIN)
#------------------------------------------------------------------------
# This example comes from Levin 1953
# Relative risk of lung cancer given smoking was 3.6
# Proportion of individuals smoking where 49.9%
# Calculates PAF (i.e. counterfactual is no smoking)
paf(p = 0.499, beta = log(3.6))

# Assuming that beta and p had a link_variance
paf(p = 0.499, beta = log(3.6), var_p = 0.001, var_beta = 0.1)

# If the link_variance was to high a logistic transform would be required
# Generates incorrect values for the interval:
paf(p = 0.499, beta = log(3.6), var_p = 0.1, var_beta = 0.3)

# Logit fixes it
paf(p = 0.499, beta = log(3.6), var_p = 0.1, var_beta = 0.3,
    link = "logit", quiet = TRUE)

# If the counterfactual was reducing the smoking population by 1/2
pif(p = 0.499, beta = log(3.6), p_cft = 0.499/2, var_p = 0.001,
    var_beta = 0.1, link = "logit", quiet = TRUE)

#EXAMPLE 2: MULTIPLE EXPOSURE CATEGORIES
#------------------------------------------------------------------------
#In "Excess Deaths Associated With Underweight, Overweight, and Obesity"
#the PAF of mortality associated to different BMI levels is calculated

#These are the relative risks for age-group 25-59 for each BMI level:
rr <- c("<18.5" = 1.38, "25 to <30" = 0.83 ,
        "30 to <35" = 1.20, ">=35" = 1.83)

#While the prevalences are:
p <- c("<18.5" = 1.9, "25 to <30" = 34.8,
        "30 to <35" = 17.3, ">=35" = 13.3) / 100

#The variance of the relative risk is obtained from the CIs:
var_log_rr <- c("<18.5" = 0.2653156, "25 to <30" =  0.1247604,
                "30 to <35" = 0.1828293, ">=35" = 0.1847374)^2

#Note that we are omitting the group "18.5 to < 25" as it is the
#reference (RR = 1)
paf(p = p, beta = rr, var_beta = var_log_rr, var_p = 0, quiet = TRUE,
    link = "logit")

#We can compute a potential impact fraction of, for example, reducing the
```

```
#amount of people over 35 to 0 and having them in the "30 to <35":
p_cft <- c("<18.5" = 1.9, "25 to <30" = 34.8,
            "30 to <35" = 17.3 + 13.3, ">=35" = 0) / 100

#The potential impact fraction is as follows:
pif(p = p, p_cft = p_cft, beta = rr, link = "logit",
  var_beta = var_log_rr, var_p = 0, quiet = TRUE)
```

`print` *Print or show a potential impact fraction*

Description

Function to print or show a potential impact fraction object

Function to print an object

Arguments

`accuracy`	The accuracy of the printed value
`x`	A `covariance_structure_class`, a `pif_class` or a `cases_class`
`...`	Additional arguments to pass to `print`

Value

NULL. Called for its side-effects.

Examples

```
my_pif <- pif(p = 0.2, beta = 1.3, var_beta = 0.1)
print(my_pif)

# Change the ammount of digits to show just 1
print(my_pif, accuracy = 0.1)
pif_lead_women <- paf(0.27, 2.2, quiet = TRUE, var_p = 0.001, var_beta = 0.015,
                      label = "Women lead")
pif_rad_women  <- paf(0.12, 1.2, quiet = TRUE, var_p = 0.001, var_beta = 0.022,
                      label = "Women radiation")
pif_women      <- pif_ensemble(pif_lead_women, pif_rad_women, label = "Women",
                               weights = c(0.8, 0.72),
                               var_weights = matrix(c(0.3, 0.1, 0.1, 0.4), ncol = 2))

pif_lead_men   <- paf(0.30, 2.2, quiet = TRUE, var_p = 0.001, var_beta = 0.015,
                       label = "Men lead")
pif_rad_men    <- paf(0.10, 1.2, quiet = TRUE, var_p = 0.001, var_beta = 0.022,
                        label = "Men radiation")
pif_men        <- pif_ensemble(pif_lead_men, pif_rad_men, label = "Men",
                        weights = c(0.65, 0.68),
                        var_weights = matrix(c(0.1, -0.2, -0.2, 0.5), ncol = 2))
```

```
pif_tot        <- pif_total(pif_men, pif_women,
                        weights = c(0.49, 0.51), label = "Population",
                        var_weights = matrix(c(0.22, 0.4, 0.4, 0.8), ncol = 2))

print(covariance_structure(pif_lead_women))
print(covariance_structure2(pif_lead_women, pif_lead_men))
print(default_weight_covariance_structure2(pif_men, pif_women))
print(default_parameter_covariance_structure(pif_tot, parameter = "beta"))
```

`rowcol` *Row and Column names*

Description

Retrieve the row and column names of a `covariance_structure`

Usage

```
row_names(x)

col_names(x)
```

Arguments

`x`	A `covariance_structure`

Value

The names of the rows or columns of a `covariance_structure`

Examples

```
#' #A pif composed of others
my_pif1 <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1,
                var_beta = 0.2, label = "pif 1")
my_pif2 <- pif(p = 0.4, p_cft = 0.1, beta = 1.3, var_p = 0.1,
                var_beta = 0.2, label = "pif 2")
covst <- covariance_structure2(my_pif1, my_pif2)
row_names(covst)
col_names(covst)
```

subset *Subset a* covariance_structure

Description

Obtains a smaller covariance_structure with the entries given by the select option as a vector.

Arguments

x	A covariance_structure
select	A vector of covariate names to keep in the covariance_structure.
cols	A vector of covariate column names to keep in the covariance_structure.
rows	A vector of covariate row names to keep in the covariance_structure.
negate	If TRUE subsets the variables that have not been specified
...	Additional parameters (ignored)

Value

A covariance_structure with only the elements specified for the subset.

Examples

```
pif_lead_women <- paf(0.27, 2.2, quiet = TRUE, var_p = 0.001, var_beta = 0.015,
                      label = "Women lead")
pif_rad_women  <- paf(0.12, 1.2, quiet = TRUE, var_p = 0.001, var_beta = 0.022,
                      label = "Women radiation")
covstr <- default_parameter_covariance_structure2(pif_lead_women, pif_rad_women, parameter = "beta")
subset(covstr, "Women lead")
subset(covstr, "Women lead", negate = TRUE)
subset(covstr, c("Women radiation", "Women lead"))
subset(covstr, 2)
subset(covstr, 1:2)
```

summary *Summary of an object*

Description

Gets the point-estimate and confidence interval of an object

Arguments

object	A pif_class or cases_class object.
...	Additional parameters to pass to summary (ignored)

Value

A named vector with the point-estimate, confidence interval and standard deviation of a `pif_class` or a `cases_class` estimate.

Examples

```
my_pif <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1, var_beta = 0.2)
summary(my_pif)
```

---

totalpifpaf *Combine Potential Impact Fractions and Population Attributable Fractions*

Description

Combine potential impact fractions or the population attributable fractions to either generate the total fraction from the fractions of subpopulations (`pif_total/paf_total`) or the ensemble fraction of a population from different (independent) exposures.

Usage

```
paf_total(
  paf1,
  ...,
  weights,
  var_weights = 0,
  var_pif_weights = NULL,
  conf_level = 0.95,
  link = "log-complement",
  link_inv = NULL,
  link_deriv = NULL,
  quiet = FALSE,
  label = NULL
)

pif_total(
  pif1,
  ...,
  weights,
  var_weights = 0,
  var_pif_weights = NULL,
  conf_level = 0.95,
  link = "log-complement",
  link_inv = NULL,
  link_deriv = NULL,
  quiet = FALSE,
  label = NULL,
```

```
  is_paf = FALSE,
  weights_sum_to_1 = TRUE
)

paf_ensemble(
  paf1,
  ...,
  weights = NULL,
  var_weights = 0,
  var_pif_weights = NULL,
  link = "identity",
  link_inv = NULL,
  link_deriv = NULL,
  conf_level = 0.95,
  label = NULL,
  quiet = FALSE
)

pif_ensemble(
  pif1,
  ...,
  weights = NULL,
  var_weights = 0,
  var_pif_weights = NULL,
  link = "identity",
  link_inv = NULL,
  link_deriv = NULL,
  conf_level = 0.95,
  quiet = FALSE,
  label = NULL,
  is_paf = FALSE
)
```

Arguments

paf1	A population attributable fraction (class pif_class)
...	The remaining potential impact fractions or (respectively) population attributable fractions.
weights	A vector containing the proportion of the population for each of the categories (for each of the pifs given).
var_weights	link_covariance structure for the weights. Can be 0 (default) if the weights are not random, a vector if only the link_variances of the weights are available or a link_covariance matrix.
var_pif_weights	covariance matrix with row i and column j representing the covariance between the i-th potential impact fraction of the list and the j-th weight
conf_level	Confidence level for the confidence interval (default 0.95).

`link`	Link function such that the `pif` confidence intervals stays within the expected bounds.
`link_inv`	(Optional). If `link` is a function then yhe inverse of `link`. For example if `link` is `logit` this should be `inv_logit`.
`link_deriv`	Derivative of the `link` function. The function tries to build it automatically from `link` using `Deriv::Deriv()`.
`quiet`	Whether to show messages.
`label`	Character identifier for the impact fraction. This is for
`pif1`	A potential impact fraction (class `pif_class`)
`is_paf`	Whether the computed quantity is a population attributable fraction or not
`weights_sum_to_1`	Boolean flag indicating if the weights sum to 1 (normalized weights) or if they are not (unnormalized).

Value

A `pif_class` object containing the estimate for the total fraction or the ensemble fraction that aggregates several fractions (either potential impact or population attributable).

Total potential impact fraction

Assuming the overall population can be subdivided into N distinct subpopulations each of them with a different potential impact fraction (or population attributable fraction) we can estimate the total population attributable fraction or potential impact fraction of the whole population as:

$$\text{PIF}_{\text{Total}} = \sum_{i=1}^{N} \pi_i \cdot \text{PIF}_i$$

where each PIF_i corresponds to the potential impact fraction of the i-th subpopulation and π_i correspond to the proportion of the total population occupied by PIF_i. The weights are such that $\sum_{i=1}^{N} \pi_i = 1$.

Ensemble potential impact fraction

If a population is exposed to K different independent risk factors then the ensemble impact fraction of the combination of those factors can be written as:

$$\text{PIF}_{\text{Ensemble}} = 1 - \prod_{\ell=1}^{K} \left(1 - \pi_\ell \cdot \text{PIF}_\ell\right)$$

where each PIF_ℓ corresponds to the potential impact fraction of the ℓ-th risk factor for the same population.

Examples

```
#Potential impact fraction for women
pif_women <- pif(0.32, 0.1, 1.2, quiet = TRUE, var_p = 0.1)

#Potential impact fraction for men
pif_men <- pif(0.27, 0.1, 1.3, quiet = TRUE, var_p = 0.1)

#Population potential impact fraction with 49% men and 51% women
pif_total(pif_men, pif_women, weights = c(0.49, 0.51), link = "logit")

#Population attributable  fraction for women
paf_women <- paf(0.32, 1.3, quiet = TRUE, var_p = 0.1)

#Population attributable  fraction for men
paf_men <- paf(0.27, 1.3, quiet = TRUE, var_p = 0.1)
paf_total(paf_men, paf_women, weights = c(0.49, 0.51), link = "logit")

# Calculate the ensemble from lead and radiation exposure
paf_lead <- paf(0.2, 2.2, quiet = TRUE, var_p = 0.001)
paf_rad  <- paf(0.1, 1.2, quiet = TRUE, var_p = 0.0001)
pif_ensemble(paf_lead, paf_rad)

# Totals and ensembles can be combined
pif_lead_women <- paf(0.27, 2.2, quiet = TRUE, var_p = 0.001)
pif_rad_women  <- paf(0.12, 1.2, quiet = TRUE, var_p = 0.001)
pif_women      <- pif_ensemble(pif_lead_women, pif_rad_women)
pif_lead_men   <- paf(0.30, 2.2, quiet = TRUE, var_p = 0.001)
pif_rad_men    <- paf(0.10, 1.2, quiet = TRUE, var_p = 0.001)
pif_men        <- pif_ensemble(pif_lead_men, pif_rad_men)

pif_total(pif_men, pif_women, weights = c(0.49, 0.51))
```

`weighted_adjusted` *Weighted Adjusted Fractions*

Description

Calculates the weighted adjusted potential impact fractions (or population attributable fractions). Each individual fraction $\widehat{\text{PIF}}_i$ is rescaled proportionally so that together they are consistent with the ensemble fraction.

Usage

```
weighted_adjusted_paf(
  paf1,
  ...,
  weights = NULL,
  var_weights = 0,
```

```
  var_pif_weights = NULL,
  var_p = NULL,
  var_beta = NULL,
  conf_level = 0.95,
  label_ensemble = NULL,
  label_sum = NULL,
  quiet = FALSE
)

weighted_adjusted_pif(
  pif1,
  ...,
  weights = NULL,
  var_weights = 0,
  var_pif_weights = NULL,
  var_p = NULL,
  var_beta = NULL,
  pif_total_link = "log-complement",
  pif_total_link_inv = NULL,
  pif_total_link_deriv = NULL,
  pif_ensemble_link = "identity",
  pif_ensemble_link_inv = NULL,
  pif_ensemble_link_deriv = NULL,
  conf_level = 0.95,
  label_ensemble = NULL,
  label_sum = NULL,
  quiet = FALSE
)
```

Arguments

paf1	A population attributable fraction (class pif_class). This is the first of the individual fractions to be adjusted.
...	The remaining potential impact fractions (class pif_class). All fractions must be of the same type (all PIF or all PAF).
weights	Weights for the ensemble (pif_ensemble). Passed directly to pif_ensemble() / paf_ensemble(). Defaults to NULL (equal weights of 1 for each fraction).
var_weights	Covariance structure for weights. Passed directly to pif_ensemble() / paf_ensemble(). Defaults to 0.
var_pif_weights	Covariance matrix between individual fractions and weights. Passed to pif_ensemble() / paf_ensemble(). Defaults to NULL.
var_p	Covariance matrix for the prevalence parameters p across all fractions. Passed to covariance() when computing cross-covariances. Defaults to NULL.
var_beta	Covariance matrix for the relative-risk parameters beta across all fractions. Passed to covariance() when computing cross-covariances. Defaults to NULL.
conf_level	Confidence level for the confidence interval (default 0.95).

`label_ensemble` Character label for the internally constructed ensemble fraction. Defaults to `NULL` (auto-generated).

`label_sum` Character label for the internally constructed sum (total) fraction. Defaults to `NULL` (auto-generated).

`quiet` Whether to show messages.

`pif1` A potential impact fraction (class `pif_class`). This is the first of the individual fractions to be adjusted.

`pif_total_link` Link to pass to `pif_total` in the denominator of the adjustment

`pif_total_link_inv` Inverse of the link to pass to `pif_total` in the denominator of the adjustment

`pif_total_link_deriv` Derivative of the link to pass to `pif_total` in the denominator of the adjustment

`pif_ensemble_link` Link to pass to `pif_ensemble` in the numerator of the adjustment

`pif_ensemble_link_inv` Inverse of the link to pass to `pif_ensemble` in the numerator of the adjustment

`pif_ensemble_link_deriv` Derivative of the link to pass to `pif_ensemble` in the numerator of the adjustment

Value

A named list of `pif_class` objects, one per input fraction, each being the weighted adjusted PAF (or PIF, respectively).

Formula

The weighted adjusted fraction for the i-th exposure is:

$$\widehat{\mathrm{PIF}}_i^{\mathrm{adj}} = \frac{\widehat{\mathrm{PIF}}_i}{\sum_{j=1}^{n} \widehat{\mathrm{PIF}}_j} \cdot \widehat{\mathrm{PIF}}_{\mathrm{Ensemble}}$$

Using the log-transform, the variance is:

$$\begin{aligned}\mathrm{Var}\Big[\ln \widehat{\mathrm{PIF}}_i^{\mathrm{adj}}\Big] = {} & \mathrm{Var}\Big[\ln \widehat{\mathrm{PIF}}_i\Big] + \mathrm{Var}\Big[\ln \textstyle\sum_j \widehat{\mathrm{PIF}}_j\Big] + \mathrm{Var}\Big[\ln \widehat{\mathrm{PIF}}_{\mathrm{Ensemble}}\Big] \\ & + 2\left[\mathrm{Cov}\Big(\ln \widehat{\mathrm{PIF}}_i, \ln \widehat{\mathrm{PIF}}_{\mathrm{Ensemble}}\Big) - \mathrm{Cov}\Big(\ln \widehat{\mathrm{PIF}}_i, \ln \textstyle\sum_j \widehat{\mathrm{PIF}}_j\Big) - \mathrm{Cov}\Big(\ln \widehat{\mathrm{PIF}}_{\mathrm{Ensemble}}, \ln \textstyle\sum_j \widehat{\mathrm{PIF}}_j\Big)\right]\end{aligned}$$

where each log-covariance is approximated via the delta method:

$$\mathrm{Cov}\big(\ln X, \ln Y\big) \approx \frac{\mathrm{Cov}(X, Y)}{X \cdot Y}$$

The covariances $\mathrm{Cov}(\widehat{\mathrm{PIF}}_i, \cdot)$ are computed by the internal function `cov_total_pif()` applied to the individual fraction and the internally constructed sum/ensemble objects, which automatically propagates the full uncertainty structure already embedded in those objects.

See Also

pif(), paf(), pif_ensemble(), paf_ensemble()

Examples

```
paf_lead <- paf(0.2, 2.2, quiet = TRUE, var_p = 0.001, label = "Lead")
paf_rad  <- paf(0.1, 1.2, quiet = TRUE, var_p = 0.0001, label = "Radiation")

weighted_adjusted_paf(paf_lead, paf_rad)

pif_lead <- pif(0.2, p_cft = 0.1, beta = log(2.2), quiet = TRUE,
                var_p = 0.001, label = "Lead")
pif_rad  <- pif(0.1, p_cft = 0.05, beta = log(1.2), quiet = TRUE,
                var_p = 0.0001, label = "Radiation")

weighted_adjusted_pif(pif_lead, pif_rad)
```

---

weights *Extract weights of a pif_global_ensemble*

Description

Gets the weights of a pif_global_ensemble object

Arguments

object	A pif_global_ensemble object.
...	Additional parameters to pass to weights (ignored)

Value

The weights used to construct a pif_total or pif_ensemble.

Examples

```
my_pif1 <- pif(p = 0.5, p_cft = 0.25, beta = 1.3, var_p = 0.1, var_beta = 0.2)
my_pif2 <- pif(p = 0.3, p_cft = 0.1, beta = 1.5, var_p = 0.1, var_beta = 0.2)
my_pif  <- pif_total(my_pif1, my_pif2, weights = c(0.8, 0.2))
weights(my_pif)
```

Index