# Package 'pifpaf'

September 28, 2017

**Title** Potential Impact Fraction and Population Attributable Fraction for Cross-Sectional Data

**Version** 1.0.1

**Description** Uses a generalized method to estimate the Potential Impact Fraction (PIF) and the Population Attributable Fraction (PAF) from cross-sectional data. It creates point-estimates, confidence intervals, and estimates of variance. In addition it generates plots for conducting sensitivity analysis. The estimation method corresponds to Zepeda-Tello, Camacho-García-Formentí, et al. 2017. 'Nonparametric Methods to Estimate the Potential Impact Fraction from Cross-sectional Data'. Unpublished manuscript. This package was developed under funding by Bloomberg Philanthropies.

**Depends** R (>= 3.4.1)

**License** GPL-3 | file LICENSE

**Encoding** UTF-8

**LazyData** true

**Type** Package

**Date** 2017-09-25

**Suggests** testthat, knitr, rmarkdown

**VignetteBuilder** knitr

**LazyLoad** yes

**Imports** ggplot2, matrixcalc, MASS, sfsmisc, stats, gridExtra, numDeriv

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Rodrigo Zepeda-Tello [aut, cre],
Dalia Camacho-García-Formentí [aut],
Tonatiuh Barrientos-Gutiérrez [ctb],
Ana Basto-Abreu [ctb],
Ariela Braverman-Bronstein [ctb],
Dèsirée Vidaña-Pérez [ctb],
Frederick Cudhea [ctb],
Instituto Nacional de Salud Pública [cph]

**Maintainer** Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

**Repository** CRAN

**Date/Publication** 2017-09-28 19:17:36 UTC

# R topics documented:



---

`counterfactual.plot` *Plot exposure's distribution under counterfactual scenario*

---

## Description

Generates a `ggplot2` plot of the current distribution of exposure `X` (continuous or discrete) as well as the distribution of `X` after a counterfactual function `cft(X)` is applied.

## Usage

```
counterfactual.plot(X, cft, weights = rep(1/nrow(as.matrix(X)),
  nrow(as.matrix(X))), adjust = 1, n = 512, ktype = "gaussian",
  bw = "SJ",
  title = "Exposure distribution under current and counterfactual scenarios",
  dnames = c("Current distribution", "Counterfactual distribution"),
  exposure.type = NA, legendtitle = "Scenario", xlab = "Exposure",
  ylab = "Density", colors = c("deepskyblue", "tomato3"),
  x_axis_order = unique(X[, 1]), fill_limits = c(-Inf, Inf), fill = TRUE,
  check_exposure = TRUE)
```

## Arguments

| | |
|---|---|
| X | Random sample (one-dimensional data.frame) of exposure. |
| cft | Function cft(X) for counterfactual. ****Optional**** |
| weights | Normalized survey weights for the sample X. |
| adjust | Adjust bandwith parameter (for "continuous" exposure) from density. |
| n | Number of equally spaced points at which the density (for "continuous" exposure) is to be estimated (see density). |
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "continuous" exposure) . Additional information on kernels in density. |
| bw | Smoothing bandwith parameter (for "continuous" exposure) from density. Default "SJ". |
| title | String with plot's title. |
| dnames | String vector indicating the labels for the plot's legend. |
| exposure.type | Either "continuous" if distribution is continuous or "discrete" if distribution is discrete. |
| legendtitle | String title for the plot's legend. |
| xlab | String label for the X-axis of the plot. |
| ylab | String label for the Y-axis of the plot. |
| colors | String vector specifying the fill-colors for the plotted distributions. |
| x_axis_order | String vector of names in X-axis for plot ("discrete" case). |
| fill_limits | Vector with lower and upper bounds of a subset of the exposure X such that only the Xs satisfying fill_limits[1] < X < fill_limits[2] are filled with color. |
| fill | Boolean that indicates whether there is interior colouring. Default TRUE. |
| check_exposure | Check exposure X is positive and numeric (if "continuous"). |

## Details

The function automatically tries to distinguish between "continuous" and "discrete" distribution inputs. By "continuous" we mean a vector of real numbers; by "discrete" a vector of strings or factor variables.

## Value

cft_plot ggplot object plotting the shift from observed to counterfactual distribution of exposure X under cft.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## References

Vander Hoorn, S., Ezzati, M., Rodgers, A., Lopez, A. D., & Murray, C. J. (2004). *Estimating attributable burden of disease from exposure and hazard data. Comparative quantification of health risks: global and regional burden of disease attributable to selected major risk factors*. Geneva: World Health Organization, 2129-40.

## See Also

`pif` for Potential Impact Fraction estimation, `pif.heatmap` for sensitivity analysis of the counterfactual, `pif.plot` for a plot of `pif` as a function of the relative risk's parameter `theta`.

## Examples

```
#Example 1: Bivariate exposure
#-------------------------------------------------------
set.seed(2783569)
X   <- data.frame(Exposure =
           sample(c("Exposed","Unexposed"), 100, replace = TRUE,
           prob = c(0.3, 0.7)))
cft <- function(X){

     #Find which indivuals are exposed
     exposed    <- which(X[,"Exposure"] == "Exposed")

     #Change 1/3 of exposed to unexposed
     reduced               <- sample(exposed, length(exposed)/3)
     X[reduced,"Exposure"] <- "Unexposed"

     return(X)
}
counterfactual.plot(X, cft)

## Not run:
#Example 2: Multivariate discrete
#-------------------------------------------------------
set.seed(2783569)
X   <- data.frame(Exposure =
         sample(c("Underweight","Normal","Overweight","Obese"), 1000,
                replace = TRUE, prob = c(0.05, 0.3, 0.25, 0.4)))

#Complex counterfactual of changing half of underweights to normal,
#1/2 of overweights to normal, 1/3 of obese to normal and
#1/3 of obese to overweight
cft <- function(X){

     #Classify the individuals
     underweights    <- which(X[,"Exposure"] == "Underweight")
     overweights     <- which(X[,"Exposure"] == "Overweight")
     obese           <- which(X[,"Exposure"] == "Obese")

     #Sample 1/2 underweights and overweights and 2/3 of obese
```

```
     changed_under    <- sample(underweights, length(underweights)/2)
     changed_over     <- sample(overweights,  length(overweights)/2)
     changed_obese    <- sample(obese,        2*length(obese)/3)

     #Assign those obese that go to normal and those that go to overweight
     obese_to_normal  <- sample(changed_obese, length(changed_obese)/2)
     obese_to_over    <- which(!(changed_obese %in% obese_to_normal))

     #Change the individuals to normal and overweight
     X[changed_under,"Exposure"]   <- "Normal"
     X[changed_over,"Exposure"]    <- "Normal"
     X[obese_to_normal,"Exposure"] <- "Normal"
     X[obese_to_over,"Exposure"]   <- "Overweight"

     return(X)
}

#Create plot of counterfactual distribution
cftplot <- counterfactual.plot(X, cft,
               x_axis_order = c("Underweight","Normal","Obese","Overweight"))
cftplot

#Objects returned are ggplot objects and you can play with them
#require(ggplot2)
#cftplot + coord_flip() + theme_grey()


#Example 3: Normal distribution and linear counterfactual
#---------------------------------------------------------
set.seed(2783569)
X   <- data.frame(Exposure = rnorm(1000, 150, 15))
cft <- function(X){0.35*X + 75}
counterfactual.plot(X, cft, xlab = "Usual SBP (mmHg)",
ylab = "Relative risk of ischemic heart disease",
dnames = c("Current distribution", "Theoretical Minimum Risk Distribution"),
title = paste0("Effect of a non-linear hazard function and choice",
               "\nof baseline on total population risk",
               "\n(Fig 25 from Vander Hoorn et al)"))

#Example 4: Counterfactual of BMI reduction only for those
#with excess-weight (BMI > 25)
#---------------------------------------------------------
set.seed(2783569)
X <- data.frame(Exposure = rlnorm(1000, 3, 0.2))
cft <- function(X){

     #Find individuals with excess weight
     excess_weight <- which(X["Exposure"] > 25)

     #Set those with excess weight to BMI of 25
     X[excess_weight, "Exposure"] <- 25

     return(X)
```

```
}

counterfactual.plot(X, cft, ktype = "epanechnikov")

#Change bandwidth method to reduce noise
counterfactual.plot(X, cft, ktype = "epanechnikov", bw = "nrd0")

## End(Not run)
```

---

paf *Population Attributable Fraction*

---

## Description

Function for estimating the Population Attributable Fraction `paf` from a cross-sectional sample of the exposure `X` with a known Relative Risk function `rr` with meta-analytical parameter `theta`, where the Population Attributable Fraction is given by:

$$PAF = \frac{E_X\left[rr(X;\theta)\right] - 1}{E_X\left[rr(X;\theta)\right]}.$$

## Usage

```
paf(X, thetahat, rr, method = "empirical",
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), Xvar = var(X),
  deriv.method.args = list(), deriv.method = "Richardson", adjust = 1,
  n = 512, ktype = "gaussian", bw = "SJ", check_exposure = TRUE,
  check_rr = TRUE, check_integrals = TRUE)
```

## Arguments

| | |
|---|---|
| `X` | Random sample (`data.frame`) which includes exposure and covariates or sample `mean` if `"approximate"` method is selected. |
| `thetahat` | Asymptotically consistent or Fisher consistent estimator (`vector`) of `theta` for the Relative Risk function `rr`. |
| `rr` | `function` for Relative Risk which uses parameter `theta`. The order of the parameters should be `rr(X, theta)`. |
| | ****Optional**** |
| `method` | Either `"empirical"` (default), `"kernel"` or `"approximate"`. For details on estimation methods see `pif`. |
| `weights` | Normalized survey `weights` for the sample `X`. |
| `Xvar` | Variance of exposure levels (for `"approximate"` method). |
| `deriv.method.args` | `method.args` for `hessian` (for `"approximate"` method). |

| | |
|---|---|
| deriv.method | method for hessian. Don't change this unless you know what you are doing (for "approximate" method). |
| adjust | Adjust bandwith parameter (for "kernel" method) from density. |
| n | Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density). |
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density. |
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| check_exposure | boolean Check that exposure X is positive and numeric. |
| check_rr | boolean Check that Relative Risk function rr equals 1 when evaluated at 0. |
| check_integrals | boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario. |

## Details

The Relative Risk function rr and counterfactual cft should consider X to be a data.frame. Each row of X is an "individual" and each column a variable (exposure or covariate) of said individual.

## Value

paf Estimate of Population Attributable Fraction.

## Note

"approximate" method should be the last choice. In practice "empirical" should be preferred as convergence is faster in simulations than "kernel". In addition, the scope of "kernel" is limited as it does not work with multivariate exposures X.

paf is a wrapper for pif with counterfactual of theoretical minimum risk exposure (rr = 1).

For more information on kernels see density.

Do not use the $ operator when using "approximate" method.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## References

Vander Hoorn, S., Ezzati, M., Rodgers, A., Lopez, A. D., & Murray, C. J. (2004). *Estimating attributable burden of disease from exposure and hazard data. Comparative quantification of health risks: global and regional burden of disease attributable to selected major risk factors*. Geneva: World Health Organization, 2129-40.

## See Also

paf.confidence for confidence interval estimation, pif for Potential Impact Fraction estimation.

See paf.exponential and paf.linear for fractions with ready-to-use exponential and linear Relative Risks respectively.

Sensitivity analysis plots can be done with paf.plot, and paf.sensitivity.

## Examples

```
#Example 1: Exponential Relative Risk
#-------------------------------------------
set.seed(18427)
X        <- data.frame(Exposure = rnorm(100,3,1))
thetahat <- 0.12
rr       <- function(X, theta){exp(theta*X)}

#Using the empirical method
paf(X, thetahat, rr)

#Same example with kernel method
paf(X, thetahat, rr, method = "kernel")

#Same example with approximate method
Xmean <- data.frame(Exposure = mean(X[,"Exposure"]))
Xvar  <- var(X[,"Exposure"])
paf(Xmean, thetahat, rr, method = "approximate", Xvar = Xvar)

#Additional options for approximate:
paf(Xmean, thetahat, rr, method = "approximate", Xvar = Xvar,
   deriv.method = "Richardson",  deriv.method.args = list(eps=1e-3, d=0.1))

#Example 2: Linear Relative Risk with weighted sample
#-------------------------------------------
set.seed(18427)
X                   <- data.frame(Exposure = rbeta(100,3,1))
weights             <- runif(100)
normalized_weights  <- weights/sum(weights)
thetahat            <- 0.12
rr                  <- function(X, theta){theta*X^2 + 1}
paf(X, thetahat, rr, weights = normalized_weights)


#Additional options for kernel:
paf(X, thetahat, rr, weights = normalized_weights,
     method = "kernel", ktype = "cosine", bw = "nrd0")


#Example 3: Multivariate Linear Relative Risk
#-------------------------------------------
set.seed(18427)
X1       <- rnorm(100,4,1)
```

```
X2       <- rnorm(100,2,0.4)
X        <- data.frame(Exposure = X1, Covariate = X2)
thetahat <- c(0.12, 0.03)

#When creating relative risks avoid using the $ operator
#as it doesn't work under approximate method
rr_not    <- function(X, theta){
               exp(theta[1]*X$Exposure + theta[2]*X$Covariate)
             }
rr_better <- function(X, theta){
               exp(theta[1]*X[,"Exposure"] + theta[2]*X[,"Covariate"])
             }

#For the empirical method it makes no difference:
paf(X, thetahat, rr_better)
paf(X, thetahat, rr_not)


#But the approximate method crashes due to operator
Xmean <- data.frame(Exposure = mean(X[,"Exposure"]),
                    Covariate = mean(X[,"Covariate"]))
Xvar  <- var(X)

paf(Xmean, thetahat, rr_better, method = "approximate", Xvar = Xvar)
## Not run:
#Error: $ operator in rr definitions don't work in approximate
paf(Xmean, thetahat, rr_not, method = "approximate", Xvar = Xvar)

## End(Not run)


## Not run:
#Error: Multivariate cases cannot be evaluated with kernel method
paf(X, thetahat, rr, method = "kernel")

## End(Not run)

#Example 4: Categorical Relative Risk & Exposure
#-----------------------------------------
set.seed(18427)
mysample  <- sample(c("Normal","Overweight","Obese"), 100,
                    replace = TRUE, prob = c(0.4, 0.1, 0.5))
X        <- data.frame(Exposure = mysample)

thetahat <- c(1, 1.2, 1.5)

#Categorical relative risk function
rr <- function(X, theta){

   #Create return vector with default risk of 1
   r_risk <- rep(1, nrow(X))

   #Assign categorical relative risk
```

```
   r_risk[which(X[,"Exposure"] == "Normal")]      <- thetahat[1]
   r_risk[which(X[,"Exposure"] == "Overweight")]  <- thetahat[2]
   r_risk[which(X[,"Exposure"] == "Obese")]       <- thetahat[3]

   return(r_risk)
}

paf(X, thetahat, rr, check_rr = FALSE)

#Example 5: Continuous Exposure and Categorical Relative Risk
#-------------------------------------------------------------------
set.seed(18427)

#Assume we have BMI from a sample
BMI          <- data.frame(Exposure = rlnorm(100, 3.1, sdlog = 0.1))

#Theoretical minimum risk exposure is at 20kg/m^2 in borderline "Normal" category
BMI_adjusted <- BMI - 20

thetahat <- c(Malnourished = 2.2, Normal = 1, Overweight = 1.8,
              Obese = 2.5)
rr       <- function(X, theta){

     #Create return vector with default risk of 1
     r_risk <- rep(1, nrow(X))

     #Assign categorical relative risk
     r_risk[which(X[,"Exposure"] < 0)]             <- theta[1] #Malnourished
     r_risk[intersect(which(X[,"Exposure"] >= 0),
                      which(X[,"Exposure"] < 5))]  <- theta[2] #Normal
     r_risk[intersect(which(X[,"Exposure"] >= 5),
                      which(X[,"Exposure"] < 10))] <- theta[3] #Overweight
     r_risk[which(X[,"Exposure"] >= 10)]           <- theta[4] #Obese

   return(r_risk)
}

paf(BMI_adjusted, thetahat, rr, check_exposure = FALSE)

#Example 6: Bivariate exposure and rr ("classical PAF")
#-------------------------------------------------------------------
set.seed(18427)
mysample  <- sample(c("Exposed","Unexposed"), 1000,
                replace = TRUE, prob = c(0.1, 0.9))
X         <- data.frame(Exposure = mysample)
theta     <- c("Exposed" = 2.5, "Unexposed" = 1.2)
rr        <- function(X, theta){

   #Create relative risk function
   r_risk <- rep(1, nrow(X))

   #Assign values of relative risk
   r_risk[which(X[,"Exposure"] == "Unexposed")] <- theta["Unexposed"]
```

```
   r_risk[which(X[,"Exposure"] == "Exposed")]   <- theta["Exposed"]

   return(r_risk)
}

paf(X, theta, rr)

#Example 7: Continuous exposure, several covariates
#-----------------------------------------------------------------
X <- data.frame(Exposure = rbeta(100, 2, 3),
                Age      = runif(100, 20, 100),
                Sex      = sample(c("M","F"), 100, replace = TRUE),
                BMI      = rlnorm(100, 3.2, 0.2))
thetahat <- c(-0.1, 0.05, 0.2, -0.4, 0.3, 0.1)

rr <- function(X, theta){

     #Create risk vector
     Risk    <- rep(1, nrow(X))

     #Identify subpopulations
     males   <- which(X[,"Sex"] == "M")
     females <- which(X[,"Sex"] == "F")

     #Calculate population specific rr
     Risk[males] <- theta[1]*X[males,"Exposure"] +
                                     theta[2]*X[males,"Age"]^2 +
                                     theta[3]*X[males,"BMI"]/2

     Risk[females] <- theta[4]*X[females,"Exposure"] +
                                     theta[5]*X[females,"Age"]^2 +
                                     theta[6]*X[females,"BMI"]/2

    return(Risk)
}

paf(X, thetahat, rr)
```

---

`paf.combine` *Combine point estimates of PAF from different subpopulations*

---

## Description

Function for fast-computing an overall `paf` from subpopulation `paf`s.

## Usage

```
paf.combine(paf_vector, proportions)
```

## Arguments

| | |
|---|---|
| paf_vector | Vector containing pafs of each specific subpopulation. |
| proportions | Vector establishing the proportion of individuals in each subpopulation. |

## Details

The subpopulations considered should not contain common elements.

## Value

overall_paf An overall point-estimate of paf combining all subpopulations.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

paf for Population Attributable Fraction estimation, pif for Potential Impact Fraction estimation, and pif.combine for combining several PIF.

## Examples

```
#Example 1
#-------------

#Estimate PAF for each subpopulation
pafmen   <- paf(X = data.frame(2.7), thetahat = 0.12,
                rr = function(X, theta){X*theta + 1},
                Xvar = 0.11, method = "approximate")
pafwomen <- paf(X = data.frame(3.1), thetahat = 0.12,
                rr = function(X, theta){exp(X*theta/3)},
                Xvar = 0.17, method = "approximate")

#Combine estimates
paf.combine(c(pafmen, pafwomen), c(0.45, 0.55))
```

paf.confidence *Confidence Intervals for the Population Attributable Fraction*

## Description

Function that estimates confidence intervals for the Population Attributable Fraction paf from a cross-sectional sample of the exposure X with a known Relative Risk function rr with meta-analytical parameter theta, where the Population Attributable Fraction is given by:

$$PAF = \frac{E_X\left[rr(X;\theta)\right] - 1}{E_X\left[rr(X;\theta)\right]}.$$

## Usage

```
paf.confidence(X, thetahat, rr, thetavar = NA, thetalow = NA,
  thetaup = NA, method = "empirical", confidence_method = "bootstrap",
  confidence = 95, confidence_theta = 99, nsim = 1000,
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), Xvar = var(X),
  deriv.method.args = list(), deriv.method = "Richardson", adjust = 1,
  n = 512, ktype = "gaussian", bw = "SJ", check_exposure = TRUE,
  check_cft = TRUE, check_rr = TRUE, check_xvar = TRUE,
  check_integrals = TRUE, check_thetas = TRUE, force.min = FALSE)
```

## Arguments

| | |
|---|---|
| X | Random sample (data.frame) which includes exposure and covariates or sample mean if "approximate" method is selected. |
| thetahat | Asymptotically consistent of Fisher consistent estimator (vector) of theta for the Relative Risk function. thetahat should be asymptotically normal with mean theta and variance var_of_theta. |
| rr | function for Relative Risk which uses parameter theta. The order of the parameters should be rr(X, theta). |
| | ****Optional**** |
| thetavar | Estimator of variance var_of_theta of asymptotic normality of thetahat. |
| thetalow | (vector) lower bound of the confidence interval of theta. |
| thetaup | (vector) upper bound of the confidence interval of theta. |
| method | Either "empirical" (default), "kernel" or "approximate". For details on estimation methods see pif. |
| confidence_method | Either bootstrap (default) inverse, one2one, linear, loglinear. See details for additional explanation. |
| confidence | Confidence level % (default 95). If confidence_method "one2one" is selected, confidence should be at most the one from theta's confidence interval (confidence_theta%). |

confidence_theta
: Confidence level % of theta corresponding to the interval [thetalow, thetaup] (default: 99%).

nsim
: Number of simulations for estimation of variance.

weights
: Normalized survey weights for the sample X.

Xvar
: Variance of exposure levels (for "approximate" method).

deriv.method.args
: method.args for hessian (for "approximate" method).

deriv.method
: method for hessian. Don't change this unless you know what you are doing (for "approximate" method).

adjust
: Adjust bandwith parameter (for "kernel" method) from density.

n
: Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density).

ktype
: kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density.

bw
: Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ".

check_exposure
: boolean Check that exposure X is positive and numeric.

check_cft
: boolean Check that counterfactual function cft reduces exposure.

check_rr
: boolean Check that Relative Risk function rr equals 1 when evaluated at 0.

check_xvar
: boolean Check Xvar is a covariance matrix.

check_integrals
: boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario.

check_thetas
: boolean Check that theta associated parameters are correctly inputed for the model.

force.min
: Boolean indicating whether to force the rr to have a minimum value of 1 instead of 0 (not recommended). This works only for confidence_method "inverse".

## Details

The confidence_method estimates confidence intervals with different methods. A bootstrap approximation is conducted by "bootstrap". The Delta Method is applied to paf or log(paf) when choosing "linear" and "loglinear" respectively. The "inverse" method estimates confidence intervals for the Relative Risk function rr and applies the transformation 1 - 1/rr. Finally, "one2one" works with functions for which the expected value over X of the relative risk is injective in theta.

Additional information on confidence method estimations can be found in the package's vignette: browseVignettes("pifpaf").

## Value

pafvec Vector with lower ("Lower_CI"), and upper ("Upper_CI") confidence bounds for the paf as well as point estimate "Point_Estimate" and estimated variance or variance of log(paf) (if confidence_method is "loglinear").

## Note

paf.confidence is a wrapper for pif.confidence with counterfactual of theoretical minimum risk exposure (rr = 1) .

For more information on kernels see density.

Do not use the $ operator when using "approximate" method.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

pif.confidence for confidence interval estimation of pif, and paf for only point estimates.

Sensitivity analysis plots can be done with paf.plot, and paf.sensitivity.

## Examples

```
#Example 1: Exponential Relative Risk
#------------------------------------------
set.seed(18427)
X        <- data.frame(Exposure = rnorm(100,3,1))
thetahat <- 0.32
thetavar <- 0.02
rr       <- function(X, theta){exp(theta*X)}

#Using bootstrap method
paf.confidence(X, thetahat, rr, thetavar)

## Not run:
#Same example with loglinear method
paf.confidence(X, thetahat, rr, thetavar, confidence_method = "loglinear")

#Same example with linear method (usually the widest and least precise)
paf.confidence(X, thetahat, rr, thetavar, confidence_method = "linear")

#Same example with inverse method
paf.confidence(X, thetahat, rr, thetavar, confidence_method = "inverse")

#Same example with one2one method
#assume 99% ci of theta is [0.27, 0.35]
paf.confidence(X, thetahat, rr, thetalow = 0.27, thetaup = 0.35,
confidence_method = "one2one", confidence_theta = 99)

#Example 2: Linear Relative Risk with weighted sample
#------------------------------------------
set.seed(18427)
X                  <- data.frame(Exposure = rbeta(100,3,1))
weights            <- runif(100)
```

```
normalized_weights  <- weights/sum(weights)
thetahat            <- 0.17
thetavar            <- 0.01
rr                  <- function(X, theta){theta*X^2 + 1}
paf.confidence(X, thetahat, rr, thetavar, weights = normalized_weights)

#Change the confidence level and paf method
paf.confidence(X, thetahat, rr,  thetavar, weights = normalized_weights,
     method = "kernel", confidence = 90)


#Example 3: Multivariate Linear Relative Risk
#-----------------------------------------
set.seed(18427)
X1       <- rnorm(100,4,1)
X2       <- rnorm(100,2,0.4)
thetahat <- c(0.12, 0.03)
thetavar <- diag(c(0.01, 0.02))

#But the approximate method crashes due to operator
Xmean <- data.frame(Exposure = mean(X1),
                    Covariate = mean(X2))
Xvar  <- var(cbind(X1, X2))

#When creating relative risks avoid using the $ operator
#as it doesn't work under approximate method of PAF
rr_not    <- function(X, theta){
               exp(theta[1]*X$Exposure + theta[2]*X$Covariate)
             }
rr_better <- function(X, theta){
               exp(theta[1]*X[,"Exposure"] + theta[2]*X[,"Covariate"])
             }

paf.confidence(Xmean, thetahat, rr_better, thetavar,
               method = "approximate", Xvar = Xvar)

## End(Not run)
## Not run:
#Warning: $ operator in rr definitions don't work in approximate
paf.confidence(Xmean, thetahat, rr_not, thetavar,
               method = "approximate", Xvar = Xvar)

## End(Not run)

## Not run:
#Example 4: Categorical Relative Risk & Exposure
#-----------------------------------------
set.seed(18427)
mysample  <- sample(c("Normal","Overweight","Obese"), 100,
                    replace = TRUE, prob = c(0.4, 0.1, 0.5))
X        <- data.frame(Exposure = mysample)

thetahat <- c(1, 1.2, 1.5)
```

```
thetavar <- diag(c(0.1, 0.2, 0.3))

#Categorical relative risk function
rr <- function(X, theta){

   #Create return vector with default risk of 1
   r_risk <- rep(1, nrow(X))

   #Assign categorical relative risk
   r_risk[which(X[,"Exposure"] == "Normal")]      <- thetahat[1]
   r_risk[which(X[,"Exposure"] == "Overweight")]  <- thetahat[2]
   r_risk[which(X[,"Exposure"] == "Obese")]       <- thetahat[3]

   return(r_risk)
}

paf.confidence(X, thetahat, rr, thetavar, check_rr = FALSE)


#Example 5: Continuous Exposure and Categorical Relative Risk
#-----------------------------------------------------------------
set.seed(18427)

#Assume we have BMI from a sample
BMI          <- data.frame(Exposure = rlnorm(100, 3.1, sdlog = 0.1))

#Theoretical minimum risk exposure is at 20kg/m^2 in borderline "Normal" category
BMI_adjusted <- BMI - 20

thetahat <- c(Malnourished = 2.2, Normal = 1, Overweight = 1.8,
              Obese = 2.5)
thetavar <- diag(c(0.1, 0.2, 0.2, 0.1))
rr       <- function(X, theta){

     #Create return vector with default risk of 1
     r_risk <- rep(1, nrow(X))

     #Assign categorical relative risk
     r_risk[which(X[,"Exposure"] < 0)]             <- theta[1] #Malnourished
     r_risk[intersect(which(X[,"Exposure"] >= 0),
                      which(X[,"Exposure"] < 5))]  <- theta[2] #Normal
     r_risk[intersect(which(X[,"Exposure"] >= 5),
                      which(X[,"Exposure"] < 10))] <- theta[3] #Overweight
     r_risk[which(X[,"Exposure"] >= 10)]           <- theta[4] #Obese

   return(r_risk)
}

paf.confidence(BMI_adjusted, thetahat, rr, thetavar, check_exposure = FALSE)

#Example 6: Bivariate exposure and rr ("classical PAF")
#-----------------------------------------------------------------
set.seed(18427)
```

```
mysample  <- sample(c("Exposed","Unexposed"), 1000,
                replace = TRUE, prob = c(0.1, 0.9))
X         <- data.frame(Exposure = mysample)
theta     <- c("Exposed" = 2.5, "Unexposed" = 1.2)
thetavar  <- matrix(c(0.04, 0.02, 0.02, 0.03), ncol = 2)
rr        <- function(X, theta){

   #Create relative risk function
   r_risk <- rep(1, nrow(X))

   #Assign values of relative risk
   r_risk[which(X[,"Exposure"] == "Unexposed")] <- theta["Unexposed"]
   r_risk[which(X[,"Exposure"] == "Exposed")]   <- theta["Exposed"]

   return(r_risk)
}

paf.confidence(X, theta, rr, thetavar)

#Example 7: Continuous exposure, several covariates
#-------------------------------------------------------------------
X <- data.frame(Exposure = rbeta(100, 2, 3),
                Age      = runif(100, 20, 100),
                Sex      = sample(c("M","F"), 100, replace = TRUE),
                BMI      = rlnorm(100, 3.2, 0.2))
thetahat <- c(-0.1, 0.05, 0.2, -0.4, 0.3, 0.1)

#Create variance of theta
almostvar <- matrix(runif(6^2), ncol = 6)
thetavar  <- t(almostvar) %*% almostvar
rr <- function(X, theta){
     #Create risk vector
     Risk    <- rep(1, nrow(X))

     #Identify subpopulations
     males   <- which(X[,"Sex"] == "M")
     females <- which(X[,"Sex"] == "F")

     #Calculate population specific rr
     Risk[males] <- theta[1]*X[males,"Exposure"] +
                                     theta[2]*X[males,"Age"]^2 +
                                     theta[3]*X[males,"BMI"]/2

     Risk[females] <- theta[4]*X[females,"Exposure"] +
                                     theta[5]*X[females,"Age"]^2 +
                                     theta[6]*X[females,"BMI"]/2

    return(Risk)
}

paf.confidence(X, thetahat, rr, thetavar)

## End(Not run)
```

paf.exponential *Population Attributable Fraction with Exponential Relative Risk Function*

## Description

Function that estimates the Population Attributable Fraction paf with exponential relative risk function rr given by

$$rr(X;\theta) = \exp\Big(\sum_{i=1}^{n} \theta_i X_i\Big).$$

## Usage

```
paf.exponential(X, thetahat, method = "empirical",
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), Xvar = var(X),
  deriv.method.args = list(), deriv.method = c("Richardson", "complex"),
  adjust = 1, n = 512, ktype = "gaussian", bw = "SJ",
  check_exposure = TRUE, check_rr = TRUE, check_integrals = TRUE)
```

## Arguments

| | |
|---|---|
| X | Random sample (data.frame) which includes exposure and covariates or sample mean if "approximate" method is selected. |
| thetahat | Asymptotically consistent or Fisher consistent estimator (vector) of theta for the Relative Risk function rr. ****Optional**** |
| method | Either "empirical" (default), "kernel" or "approximate". For details on estimation methods see pif. |
| weights | Normalized survey weights for the sample X. |
| Xvar | Variance of exposure levels (for "approximate" method). |
| deriv.method.args | method.args for hessian (for "approximate" method). |
| deriv.method | method for hessian. Don't change this unless you know what you are doing (for "approximate" method). |
| adjust | Adjust bandwith parameter (for "kernel" method) from density. |
| n | Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density). |
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density. |
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| check_exposure | boolean Check that exposure X is positive and numeric. |

check_rr boolean Check that Relative Risk function rr equals 1 when evaluated at 0.

check_integrals boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario.

## Value

paf Estimate of Population Attributable Fraction with exponential relative risk.

## Note

paf.exponential is a wrapper for paf with exponential relative risk.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## References

Vander Hoorn, S., Ezzati, M., Rodgers, A., Lopez, A. D., & Murray, C. J. (2004). *Estimating attributable burden of disease from exposure and hazard data. Comparative quantification of health risks: global and regional burden of disease attributable to selected major risk factors*. Geneva: World Health Organization, 2129-40.

## See Also

See paf for Population Attributable Fraction (with arbitrary relative risk), and pif for Potential Impact Fraction estimation.

See paf.linear for PAF with ready-to-use linear relative risk function.

For more information on kernels see density.

## Examples

```
#Example 1: Univariate relative risk
#---------------------------------------
set.seed(18427)
X <- data.frame(Exposure = rnorm(100, 3, .5))
thetahat <- 0.12
paf.exponential(X, thetahat) #Exponential risk given exp(0.12*X)

#This is the same as doing:
paf(X, thetahat, rr = function(X, theta){exp(X*theta)})

#Same example with kernel method
paf.exponential(X, thetahat,  method = "kernel")

#Same example with approximate method
Xmean <- data.frame(mean(X[,"Exposure"]))
```

```
Xvar  <- var(X)
paf.exponential(Xmean, thetahat, method = "approximate", Xvar = Xvar)

#Example 2: Multivariate relative risk
#---------------------------------------
X     <- data.frame(Exposure = rnorm(100,2,.7), Covariate = rnorm(100,4,1))
theta <- c(0.3,0.1)
paf.exponential(X,theta) #Exponential risk given exp(0.3*X1 + 0.1*X2)
```

---

paf.linear *Population Attributable Fraction with Linear Relative Risk Function*

---

## Description

Function that calculates the Population Attributable Fraction paf with linear Relative Risk function rr given by

$$rr(X;\theta) = \theta_1 + \sum_{i=1}^{n} \theta_{i+1} X_i.$$

## Usage

```
paf.linear(X, thetahat, method = "empirical",
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), Xvar = var(X),
  deriv.method.args = list(), deriv.method = c("Richardson", "complex"),
  adjust = 1, n = 512, ktype = "gaussian", bw = "SJ",
  check_exposure = TRUE, check_rr = TRUE, check_integrals = TRUE)
```

## Arguments

| | |
|---|---|
| X | Random sample (data.frame) which includes exposure and covariates or sample mean if "approximate" method is selected. |
| thetahat | Asymptotically consistent or Fisher consistent estimator (vector) of theta for the Relative Risk function rr. ****Optional**** |
| method | Either "empirical" (default), "kernel" or "approximate". For details on estimation methods see pif. |
| weights | Normalized survey weights for the sample X. |
| Xvar | Variance of exposure levels (for "approximate" method). |
| deriv.method.args | method.args for hessian (for "approximate" method). |
| deriv.method | method for hessian. Don't change this unless you know what you are doing (for "approximate" method). |
| adjust | Adjust bandwith parameter (for "kernel" method) from density. |

| | |
|---|---|
| n | Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density). |
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density. |
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| check_exposure | boolean Check that exposure X is positive and numeric. |
| check_rr | boolean Check that Relative Risk function rr equals 1 when evaluated at 0. |
| check_integrals | boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario. |

## Value

paf Estimate of Population Attributable Fraction with linear relative risk.

## Note

The "approximate" method should be the last choice. In practice "empirical" should be preferred as convergence is faster than "kernel" for most functions. In addition, the scope of "kernel" is limited as it does not work with multivariate exposure data X.

paf.linear is a wrapper for paf with linear relative risk.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## References

Vander Hoorn, S., Ezzati, M., Rodgers, A., Lopez, A. D., & Murray, C. J. (2004). *Estimating attributable burden of disease from exposure and hazard data. Comparative quantification of health risks: global and regional burden of disease attributable to selected major risk factors*. Geneva: World Health Organization, 2129-40.

## See Also

See paf for Population Attributable Fraction (with arbitrary relative risk), and pif for Potential Impact Fraction estimation.

See paf.exponential for PAF with ready-to-use exponential relative risk function.

For more information on kernels see density.

## Examples

```
#Example 1: Univariate relative risk
#---------------------------------------
set.seed(18427)
X <- data.frame(Exposure = rnorm(100,3,.5))
thetahat <- c(1, 0.12)   #Linear risk given by 1 + 0.12*X
paf.linear(X, thetahat)

#This is the same as doing:
paf(X, thetahat, rr = function(X, theta){X*theta[2] + theta[1]})

#Same example with kernel method
paf.linear(X, thetahat,  method = "kernel")

#Same example with approximate method
Xmean <- data.frame(mean(X[,"Exposure"]))
Xvar  <- var(X)
paf.linear(Xmean, thetahat, method = "approximate", Xvar = Xvar)

#Example 2: Multivariate relative risk
#---------------------------------------
X     <- data.frame(Exposure = rnorm(100,2,.7), Covariate = rnorm(100,4,1))
theta <- c(1, 0.3,0.1)
paf.linear(X, theta)   #Linear risk given by 1 + 0.3*X1 + 0.1*X2

#Example 3: Polynomial relative risk
#---------------------------------------
X     <- runif(100)
X2    <- X^2
X3    <- X^3
matX  <- data.frame(X,X2,X3)
theta <- c(1, 0.3,0.1, 0.4)
paf.linear(matX,theta) #Polynomial risk: 1 + 0.3*X + 0.1*X^2 + 0.4*X^3
```

---

`paf.plot` *Plot of Population Attributable Fraction under different values of Relative Risk's parameter theta*

---

## Description

Function that plots the `paf` under different values of a univariate parameter `theta` of the Relative Risk function `rr` which depends on the exposure `X` and a `theta` parameter (`rr(X, theta)`)

## Usage

```
paf.plot(X, thetalow, thetaup, rr, method = "empirical",
  confidence_method = "bootstrap", confidence = 95, nsim = 100,
```

```
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), mpoints = 100,
  adjust = 1, n = 512, Xvar = var(X), deriv.method.args = list(),
  deriv.method = "Richardson", ktype = "gaussian", bw = "SJ",
  colors = c("deepskyblue", "gray25"), xlab = "Theta", ylab = "PAF",
 title = "Population Attributable Fraction (PAF) under different values of theta",
  check_exposure = TRUE, check_rr = TRUE, check_integrals = TRUE)
```

## Arguments

| | |
|---|---|
| `X` | Random sample (`data.frame`) which includes exposure and covariates or sample `mean` if `"approximate"` method is selected. |
| `thetalow` | Minimum of `theta` (parameter of relative risk `rr`) for plot. |
| `thetaup` | Maximum of `theta` (parameter of relative risk `rr`) for plot. |
| `rr` | `function` for Relative Risk which uses parameter `theta`. The order of the parameters should be `rr(X, theta)`. |
| | **Optional** |
| `method` | Either `"empirical"` (default), `"kernel"` or `"approximate"`. For details on estimation methods see `pif`. |
| `confidence_method` | Either `bootstrap` (default) `inverse`, `one2one`, `linear`, `loglinear`. See `paf` details for additional information. |
| `confidence` | Confidence level % (default `95`). |
| `nsim` | Number of simulations to generate confidence intervals. |
| `weights` | Normalized survey `weights` for the sample `X`. |
| `mpoints` | Number of points in plot. |
| `adjust` | Adjust bandwith parameter (for `"kernel"` method) from `density`. |
| `n` | Number of equally spaced points at which the density (for `"kernel"` method) is to be estimated (see `density`). |
| `Xvar` | Variance of exposure levels (for `"approximate"` method). |
| `deriv.method.args` | `method.args` for `hessian` (for `"approximate"` method). |
| `deriv.method` | `method` for `hessian`. Don't change this unless you know what you are doing (for `"approximate"` method). |
| `ktype` | `kernel` type: `"gaussian"`, `"epanechnikov"`, `"rectangular"`, `"triangular"`, `"biweight"`, `"cosine"`, `"optcosine"` (for `"kernel"` method). Additional information on kernels in `density`. |
| `bw` | Smoothing bandwith parameter (for `"kernel"` method) from `density`. Default `"SJ"`. |
| `colors` | `vector` Colors of plot. |
| `xlab` | `string` Label of x-axis in plot. |
| `ylab` | `string` Label of y-axis in plot. |
| `title` | `string` Title of plot. |

check_exposure boolean Check that exposure X is positive and numeric.

check_rr boolean Check that Relative Risk function rr equals 1 when evaluated at 0.

check_integrals boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario.

## Value

paf.plot ggplot object with plot of Population Attributable Fraction as function of theta.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

See paf for Population Attributable Fraction estimation with confidence intervals paf.confidence.

See pif.plot for same plot with Potential Impact Fraction pif.

## Examples

```
## Not run:
#Example 1: Exponential Relative Risk empirical method
#-----------------------------------------------------
set.seed(18427)
X <- data.frame(Exposure = rbeta(25, 4.2, 10))
paf.plot(X, thetalow = 0, thetaup = 2, function(X, theta){exp(theta*X)})


#Same example with kernel method
paf.plot(X, 0, 2, function(X, theta){exp(theta*X)}, method = "kernel",
title = "Kernel method example")

#Same example for approximate method
Xmean <- data.frame(mean(X[,"Exposure"]))
Xvar  <- var(X)
paf.plot(Xmean, 0, 2, function(X, theta){exp(theta*X)},
method = "approximate", Xvar = Xvar, title = "Approximate method example")

## End(Not run)
```

paf.sensitivity *Population Attributable Fraction Sensitivity Analysis Plot*

## Description

Function that plots a sensitivity analysis for the Population Attributable Fraction paf by checking how estimates vary when reducing the exposure's sample X.

## Usage

```
paf.sensitivity(X, thetahat, rr, method = "empirical",
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), nsim = 50,
  mremove = min(nrow(as.matrix(X))/2, 100), adjust = 1, n = 512,
  ktype = "gaussian", bw = "SJ", ylab = "PAF",
  xlab = "Number of randomly deleted observations for X",
  legendtitle = "Sensitivity Analysis",
  title = paste0("Population Attributable Fraction (PAF) ",
  "Sensitivity Analysis"), colors = c("red", "deepskyblue", "gray75",
  "gray25"), check_exposure = TRUE, check_rr = TRUE,
  check_integrals = TRUE)
```

## Arguments

| | |
|---|---|
| X | Random sample (data.frame) which includes exposure and covariates or sample mean if "approximate" method is selected. |
| thetahat | Asymptotically consistent or Fisher consistent estimator (vector) of theta for the Relative Risk function. |
| rr | function for Relative Risk which uses parameter theta. The order of the parameters should be rr(X, theta). |
| | ****Optional**** |
| method | Either "empirical" (default), "kernel" or "approximate". For details on estimation methods see pif. |
| weights | Normalized survey weights for the sample X. |
| nsim | Integer with number of samples to include (for each removal) in order to conduct sensitivity analysis. See details for additional information. |
| mremove | Limit number of measurements of X to remove when resampling. See details for additional information. |
| adjust | Adjust bandwith parameter (for "kernel" method) from density. |
| n | Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density). |
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density. |

| | |
|---|---|
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| ylab | string label for the Y-axis of the plot. |
| xlab | string label for the X-axis of the plot. |
| legendtitle | String title for the legend of plot. |
| title | string Title of plot. |
| colors | String vector with colors for the plot. |
| check_exposure | boolean Check that exposure X is positive and numeric. |
| check_rr | boolean Check that Relative Risk function rr equals 1 when evaluated at 0. |
| check_integrals | boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario. |

## Details

paf.sensitivity conducts a sensitivity analysis of the paf estimate by removing mremove elements nsim times and re-estimating paf with the reduced sample.

## Value

plotpaf ggplot object plotting a sensitivity analysis of paf.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

paf for Population Attributable Fraction estimation, paf.plot for a plot of Population Attributable Fraction as a function of the relative risk's parameter theta.

## Examples

```
## Not run:
#Example 1
#------------------------------------------------------------------
set.seed(3284)
X  <- data.frame(rnorm(250,3))            #Sample
rr <- function(X,theta){exp(X*theta)}     #Relative risk
theta <- 0.1                              #Estimate of theta
paf.sensitivity(X, thetahat = theta, rr = rr)


#Save file
#require(ggplot2)
#ggsave("My Population Attributable Fraction Sensitivity Analysis.pdf")
```

```
#Example 2
#------------------------------------------------------------
set.seed(3284)
X     <- data.frame(rbeta(1000, 1, 0.2))
theta <- c(0.12, 1)
rr    <- function(X, theta){X*theta[1] + theta[2]}


#Using empirical method
paf.sensitivity(X, thetahat = theta, rr = rr,
                mremove = 100, nsim = 50,
                title = "My Sensitivity Analysis for example 1")

#Same example with kernel
paf.sensitivity(X, theta, rr = rr,
                 mremove = 100, nsim = 50, method = "kernel",
                 title = "Sensitivity Analysis for example 1 using kernel")


#Example 3: Plot counterfactual with categorical risks
#------------------------------------------------------------------
set.seed(18427)
X        <- data.frame(Exposure =
              sample(c("Normal","Overweight","Obese"), 1000,
                     replace = TRUE, prob = c(0.4, 0.1, 0.5)))
thetahat <- c(1, 1.7, 2)

#Categorical relative risk function
rr <- function(X, theta){

   #Create return vector with default risk of 1
   r_risk <- rep(1, nrow(X))

   #Assign categorical relative risk
   r_risk[which(X[,"Exposure"] == "Normal")]      <- thetahat[1]
   r_risk[which(X[,"Exposure"] == "Overweight")]  <- thetahat[2]
   r_risk[which(X[,"Exposure"] == "Obese")]       <- thetahat[3]

   return(r_risk)
}


pafplot <- paf.sensitivity(X, thetahat = thetahat, rr = rr,
                           title = "Sensitivity analysis of PAF for excess-weight",
                           colors = rainbow(4),
                           legendtitle = "Values",
                           check_exposure = FALSE, check_rr = FALSE)
pafplot

#You can edit pafplot as it is a ggplot object
#require(ggplot2)
#pafplot + theme_classic()
```

```
## End(Not run)
```

---

pif *Potential Impact Fraction*

---

## Description

Function for estimating the Potential Impact Fraction `pif` from a cross-sectional sample of the exposure `X` with known Relative Risk function `rr` with parameter `theta`, where the Potential Impact Fraction is given by:

$$PIF = \frac{E_X\left[rr(X;\theta)\right] - E_X\left[rr\big(\text{cft}(X);\theta\big)\right]}{E_X\left[rr(X;\theta)\right]}.$$

## Usage

```
pif(X, thetahat, rr, cft = NA, method = "empirical",
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), Xvar = var(X),
  deriv.method.args = list(), deriv.method = "Richardson", adjust = 1,
  n = 512, ktype = "gaussian", bw = "SJ", check_exposure = TRUE,
  check_integrals = TRUE, check_rr = TRUE, is_paf = FALSE)
```

## Arguments

| | |
|---|---|
| `X` | Random sample (`data.frame`) which includes exposure and covariates or sample `mean` if `"approximate"` method is selected. |
| `thetahat` | Asymptotically consistent or Fisher consistent estimator (`vector`) of `theta` for the Relative Risk function. |
| `rr` | `function` for Relative Risk which uses parameter `theta`. The order of the parameters should be `rr(X, theta)`. ****Optional**** |
| `cft` | Function `cft(X)` for counterfactual. Leave empty for the Population Attributable Fraction `paf` where counterfactual is that of a theoretical minimum risk exposure `X0` such that `rr(X0,theta) = 1`. |
| `method` | Either `"empirical"` (default), `"kernel"` or `"approximate"`. |
| `weights` | Normalized survey `weights` for the sample `X`. |
| `Xvar` | Variance of exposure levels (for `"approximate"` method). |
| `deriv.method.args` | `method.args` for `hessian` (for `"approximate"` method). |
| `deriv.method` | `method` for `hessian`. Don't change this unless you know what you are doing (for `"approximate"` method). |
| `adjust` | Adjust bandwith parameter (for `"kernel"` method) from `density`. |
| `n` | Number of equally spaced points at which the density (for `"kernel"` method) is to be estimated (see `density`). |

| | |
|---|---|
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density. |
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| check_exposure | Check exposure X is positive and numeric. |
| check_integrals | Check that counterfactual of theoretical minimum risk exposure and relative risk's expected values are well defined for this scenario. |
| check_rr | Check that Relative Risk function rr equals 1 when evaluated at 0. |
| is_paf | Boolean forcing evaluation of paf. This forces the pif function ignore the inputed counterfactual and set the relative risk to the theoretical minimum risk value of 1. |

## Details

The "empirical" method estimates the pif by

$$PIF = 1 - \frac{\sum_{i=1}^{n} w_i rr\big(cft(X_i);\theta\big)}{\sum_{i=1}^{n} w_i rr(X_i;\theta)}.$$

The "kernel" method approximates the density of the exposure X and estimates its expected value from that approximation:

$$PIF = 1 - \frac{\int_{-\infty}^{\infty} rr\big(cft(X);\theta\big)\hat{f}(x)dx}{\int_{-\infty}^{\infty} rr\big(cft(X);\theta\big)\hat{f}(x)dx}.$$

The "approximate" method conducts a Laplace approximation of the pif. Additional information on the methods is dicussed in the package's vignette: browseVignettes("pifpaf").

In practice "approximate" method should be the last choice. Simulations have shown that "empirical"'s convergence is faster than "kernel" for most functions. In addition, the scope of "kernel" is limited as it does not work with multivariate exposure data X.

## Value

pif Estimate of Potential Impact Fraction.

## Note

For more information on kernels see density.

Do not use the $ operator when using "approximate" method.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## References

Vander Hoorn, S., Ezzati, M., Rodgers, A., Lopez, A. D., & Murray, C. J. (2004). *Estimating attributable burden of disease from exposure and hazard data. Comparative quantification of health risks: global and regional burden of disease attributable to selected major risk factors*. Geneva: World Health Organization, 2129-40.

## See Also

See `pif.confidence` for confidence interval estimation, and `paf` for Population Attributable Fraction estimation.

Sensitivity analysis plots can be done with `pif.plot`, `pif.sensitivity`, and `pif.heatmap`.

## Examples

```
#Example 1: Exponential Relative Risk
#-----------------------------------------
set.seed(18427)
X        <- data.frame(Exposure = rnorm(100,3,1))
thetahat <- 0.12
rr       <- function(X, theta){exp(theta*X)}

#Without specifying counterfactual pif matches paf
pif(X, thetahat, rr)
paf(X, thetahat, rr)

#Same example with kernel method
pif(X, thetahat, rr, method = "kernel")

#Same example with approximate method
Xmean <- data.frame(Exposure = mean(X[,"Exposure"]))
Xvar  <- var(X[,"Exposure"])
pif(Xmean, thetahat, rr, method = "approximate", Xvar = Xvar)

#Same example considering counterfactual of halving exposure
cft   <- function(X){ 0.5*X }
pif(X, thetahat, rr, cft, method = "empirical")

#Example 2: Linear Relative Risk
#-----------------------------------------
set.seed(18427)
X        <- data.frame(Exposure = rbeta(100,3,1))
thetahat <- 0.12
rr       <- function(X, theta){theta*X + 1}
cft      <-  function(X){ 0.5*X }
weights             <- runif(100)
```

```
normalized_weights  <- weights/sum(weights)
pif(X, thetahat, rr, cft, weights = normalized_weights)

#Same example with more complex counterfactual that reduces
#only the values > 0.75 are halved
cft       <- function(X){

   #Indentify the ones with "a lot" of exposure:
   where_excess_exposure    <- which(X[,"Exposure"] > 0.75)

   #Halve their exposure
   X[where_excess_exposure, "Exposure"] <-
            X[where_excess_exposure, "Exposure"]/2
   return(X)
}
pif(X, thetahat, rr, cft, weights = normalized_weights)


#Example 3: Multivariate Linear Relative Risk
#-------------------------------------------
set.seed(18427)
X1       <- rnorm(100,4,1)
X2       <- rnorm(100,2,0.4)
X        <- data.frame(Exposure = X1, Covariate = X2)
thetahat <- c(0.12, 0.03)

#When creating relative risks and counterfactuals avoid using $ operator
#as it doesn't work under approximate method
rr_not    <- function(X, theta){
               exp(theta[1]*X$Exposure + theta[2]*X$Covariate)
             }
rr_better <- function(X, theta){
               exp(theta[1]*X[,"Exposure"] + theta[2]*X[,"Covariate"])
             }

#Creating a counterfactual.
cft  <- function(X){
   Y               <- X
   Y[,"Exposure"]  <- 0.5*X[,"Exposure"]
   Y[,"Covariate"] <- 1.1*X[,"Covariate"] + 1
   return(Y)
}
pif(X, thetahat, rr_better, cft)

#Same multivariate example for approximate method calculating
#mean and variance
Xmean <- data.frame(Exposure = mean(X$Exposure),
                    Covariate = mean(X$Covariate))
Xvar  <- var(X)
pif(Xmean, thetahat, rr_better, method = "approximate", Xvar = Xvar)

## Not run:
#The one with $ operators doesn't work:
```

```
pif(Xmean, thetahat, rr_not, method = "approximate", Xvar = Xvar)

## End(Not run)
## Not run:
#Warning: Multivariate cases cannot be evaluated with kernel method
pif(X, thetahat, rr_better, method = "kernel")

## End(Not run)

#Example 4: Categorical Relative Risk & Exposure
#------------------------------------------
set.seed(18427)
mysample  <- sample(c("Normal","Overweight","Obese"), 100,
                    replace = TRUE, prob = c(0.4, 0.1, 0.5))
X        <- data.frame(Exposure = mysample)

thetahat <- c(1, 1.2, 1.5)

#Categorical relative risk function
rr <- function(X, theta){

   #Create return vector with default risk of 1
   r_risk <- rep(1, nrow(X))

   #Assign categorical relative risk
   r_risk[which(X[,"Exposure"] == "Normal")]      <- thetahat[1]
   r_risk[which(X[,"Exposure"] == "Overweight")]  <- thetahat[2]
   r_risk[which(X[,"Exposure"] == "Obese")]       <- thetahat[3]

   return(r_risk)
}

pif(X, thetahat, rr, check_rr = FALSE)

#Counterfactual of reducing all obesity to normality
cft <- function(X){
   X[which(X[,"Exposure"] == "Obese"),] <- "Normal"
   return(X)
}

pif(X, thetahat, rr, cft, check_rr = FALSE)

#Example 5: Categorical Relative Risk & continuous exposure
#----------------------------------------------------------
set.seed(18427)
BMI      <- data.frame(Exposure = rlnorm(100, 3.1, sdlog = 0.1))

#Theoretical minimum risk exposure is at 20kg/m^2 in borderline "Normal" category
BMI_adjusted <- BMI - 20

thetahat <- c(Malnourished = 2.2, Normal = 1, Overweight = 1.8,
              Obese = 2.5)
```

```
rr       <- function(X, theta){

     #Create return vector with default risk of 1
     r_risk <- rep(1, nrow(X))

     #Assign categorical relative risk
     r_risk[which(X[,"Exposure"] < 0)]             <- theta[1] #Malnourished
     r_risk[intersect(which(X[,"Exposure"] >= 0),
                      which(X[,"Exposure"] < 5))]  <- theta[2] #Normal
     r_risk[intersect(which(X[,"Exposure"] >= 5),
                      which(X[,"Exposure"] < 10))] <- theta[3] #Overweight
     r_risk[which(X[,"Exposure"] >= 10)]           <- theta[4] #Obese

   return(r_risk)
}

#Counterfactual of everyone in normal range
cft <- function(bmi){
     bmi           <- data.frame(rep(2.5, nrow(bmi)), ncol = 1)
     colnames(bmi) <- c("Exposure")
     return(bmi)
}

pif(BMI_adjusted, thetahat, rr, cft,
    check_exposure = FALSE, method = "empirical")


#Example 6: Bivariate exposure and rr ("classical PAF")
#------------------------------------------------------------------
set.seed(18427)
mysample  <- sample(c("Exposed","Unexposed"), 1000,
                replace = TRUE, prob = c(0.1, 0.9))
X         <- data.frame(Exposure = mysample)
theta     <- c("Exposed" = 2.5, "Unexposed" = 1.2)
rr        <- function(X, theta){

   #Create relative risk function
   r_risk <- rep(1, nrow(X))

   #Assign values of relative risk
   r_risk[which(X[,"Exposure"] == "Unexposed")] <- theta["Unexposed"]
   r_risk[which(X[,"Exposure"] == "Exposed")]   <- theta["Exposed"]

   return(r_risk)
}

#Counterfactual of reducing the exposure in half of the individuals
cft <- function(X){

   #Find out which ones are exposed
   Xexp  <- which(X[,"Exposure"] == "Exposed")

   #Use only half of the exposed randomly
```

```
   reduc <- sample(Xexp, length(Xexp)/2)

   #Unexpose those individuals
   X[reduc, "Exposure"] <- "Unexposed"

   return(X)
}

pif(X, theta, rr, cft)

#Example 7: Continuous exposure, several covariates
#-----------------------------------------------------------------
X <- data.frame(Exposure = rbeta(100, 2, 3),
                Age      = runif(100, 20, 100),
                Sex      = sample(c("M","F"), 100, replace = TRUE),
                BMI      = rlnorm(100, 3.2, 0.2))
thetahat <- c(-0.1, 0.05, 0.2, -0.4, 0.3, 0.1)

rr <- function(X, theta){
     #Create risk vector
     Risk    <- rep(1, nrow(X))

     #Identify subpopulations
     males   <- which(X[,"Sex"] == "M")
     females <- which(X[,"Sex"] == "F")

     #Calculate population specific rr
     Risk[males] <- theta[1]*X[males,"Exposure"] +
                                     theta[2]*X[males,"Age"]^2 +
                                     theta[3]*X[males,"BMI"]/2

     Risk[females] <- theta[4]*X[females,"Exposure"] +
                                     theta[5]*X[females,"Age"]^2 +
                                     theta[6]*X[females,"BMI"]/2

    return(Risk)
}

#Counterfactual of reducing BMI
cft <- function(X){
    excess_bmi           <- which(X[,"BMI"] > 25)
    X[excess_bmi,"BMI"]  <- 25
    return(X)
}

pif(X, thetahat, rr, cft)
```

---

`pif.combine` *Combine point estimates of PIF from different subpopulations*

---

## Description

Function for fast-computing an overall pif from subpopulation pifs and pafs.

## Usage

```
pif.combine(pif_vector, paf_vector, proportions)
```

## Arguments

| | |
|---|---|
| pif_vector | Vector containing pifs for each specific subpopulation. |
| paf_vector | Vector containing pafs for each specific subpopulation. |
| proportions | Vector establishing the proportion of individuals in each subpopulation. |

## Details

The subpopulations considered should not contain common elements.

## Value

overall_pif An overall point-estimate of pif combining all subpopulations.

## Note

To combine pifs both pifs and pafs are required.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

See paf for Population Attributable Fraction estimation, pif for Population Impact Fraction estimation, and paf.combine for combining several PAF.

## Examples

```
#Example 1
#-------------

#Estimate PAF for each subpopulation
pafmen   <- paf(X = data.frame(2.7), thetahat = 0.12, rr = function(X, theta){X*theta + 1},
                Xvar = 0.11, method = "approximate")
pafwomen <- paf(X = data.frame(3.1), thetahat = 0.12, rr = function(X, theta){exp(X*theta/3)},
                Xvar = 0.17, method = "approximate")

#Estimate PIF for each subpopulation
pifmen   <- pif(X = data.frame(2.7), thetahat = 0.12, rr = function(X, theta){X*theta + 1},
                cft = function(X){X/2}, Xvar = 0.11, method = "approximate")
```

```
pifwomen <- pif(X = data.frame(3.1), thetahat = 0.12, rr = function(X, theta){exp(X*theta/3)},
                cft = function(X){X/2}, Xvar = 0.17, method = "approximate")

#Combine estimates
pif.combine(c(pifmen, pifwomen), c(pafmen, pafwomen), c(0.45, 0.55))
```

---

`pif.confidence` *Confidence Intervals for the Potential Impact Fraction*

---

## Description

Function that estimates confidence intervals for the Potential Impact Fraction `pif` from a cross-sectional sample of the exposure `X` with a known Relative Risk function `rr` with parameter `theta`, where the Potential Impact Fraction is given by:

$$PIF = \frac{E_X\left[rr(X;\theta)\right] - E_X\left[rr\big(\text{cft}(X);\theta\big)\right]}{E_X\left[rr(X;\theta)\right]}.$$

## Usage

```
pif.confidence(X, thetahat, rr, thetavar, cft = NA, method = "empirical",
  confidence_method = "bootstrap", confidence = 95, nsim = 1000,
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), Xvar = var(X),
  deriv.method.args = list(), deriv.method = "Richardson", adjust = 1,
  n = 512, ktype = "gaussian", bw = "SJ", check_exposure = TRUE,
  check_cft = TRUE, check_rr = TRUE, check_xvar = TRUE,
  check_integrals = TRUE, check_thetas = TRUE, is_paf = FALSE)
```

## Arguments

| | |
|---|---|
| `X` | Random sample (`data.frame`) which includes exposure and covariates or sample `mean` if `"approximate"` method is selected. |
| `thetahat` | Asymptotically consistent or Fisher consistent estimator (`vector`) of `theta` for the Relative Risk function. `thetahat` should be asymptotically normal (`N(theta, var_of_theta)`) with mean `theta` and estimated variance `var_of_theta`. |
| `rr` | `function` for Relative Risk which uses parameter `theta`. The order of the parameters shound be `rr(X, theta)`. |
| `thetavar` | Estimator of variance `var_of_theta`.<br>****Optional**** |
| `cft` | Function `cft(X)` for counterfactual. Leave empty for the Population Attributable Fraction `paf` where counterfactual is that of the theoretical minimum risk exposure `X0` such that `rr(X0,theta) = 1`. |
| `method` | Either `"empirical"` (default), `"kernel"` or `"approximate"`. For details on estimation methods see `pif`. |

confidence_method
: Either bootstrap (default), linear, loglinear. See paf details for additional information.

confidence
: Confidence level % (default 95).

nsim
: Number of simulations for estimation of variance.

weights
: Normalized survey weights for the sample X.

Xvar
: Variance of exposure levels (for "approximate" method).

deriv.method.args
: method.args for hessian (for "approximate" method).

deriv.method
: method for hessian. Don't change this unless you know what you are doing (for "approximate" method).

adjust
: Adjust bandwith parameter (for "kernel" method) from density.

n
: Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density).

ktype
: kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density.

bw
: Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ".

check_exposure
: boolean Check that exposure X is positive and numeric.

check_cft
: boolean Check that counterfactual function cft reduces exposure.

check_rr
: boolean Check that Relative Risk function rr equals 1 when evaluated at 0.

check_xvar
: boolean Check Xvar is covariance matrix.

check_integrals
: boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario.

check_thetas
: boolean Check that theta associated parameters are correctly inputed for the model.

is_paf
: Boolean forcing evaluation of paf. This forces the pif function to ignore the inputed counterfactual and set it to the theoretical minimum risk value of rr = 1.

## Value

pifvec Vector with lower ("Lower_CI"), and upper ("Upper_CI") confidence bounds for the pif as well as point estimate "Point_Estimate" and estimated variance of log(pif) (if confidence_method is "loglinear").

## Note

For more information on kernels see density.

Do not use the $ operator when using "approximate" method.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

See paf.confidence for confidence interval estimation of paf, and pif for only point estimate.

Sensitivity analysis plots can be done with paf.plot, and paf.sensitivity

## Examples

```
#Example 1: Exponential Relative Risk
#-----------------------------------------
set.seed(18427)
X        <- data.frame(Exposure = rnorm(100,3,1))
thetahat <- 0.12
thetavar <- 0.02
rr       <- function(X, theta){exp(theta*X)}


#Counterfactual of halving exposure
cft   <- function(X){ 0.5*X }

#Using bootstrap method
pif.confidence(X, thetahat, rr, thetavar, cft)

## Not run:
#Same example with loglinear method
pif.confidence(X, thetahat, rr, thetavar, cft, confidence_method = "loglinear")

#Same example with linear method (usually the widest and least precise)
pif.confidence(X, thetahat, rr, thetavar, cft, confidence_method = "linear")


#Example 2: Linear Relative Risk
#-----------------------------------------
set.seed(18427)
X        <- data.frame(Exposure = rbeta(100,3,1))
thetahat <- 0.17
thetavar <- 0.01
rr       <- function(X, theta){theta*X + 1}
cft      <- function(X){ 0.5*X }
weights             <- runif(100)
normalized_weights  <- weights/sum(weights)
pif.confidence(X, thetahat, rr, thetavar, cft, weights = normalized_weights)

#Same example with more complex counterfactual that reduces
#only the values > 0.75 are halved
cft       <- function(X){

   #Indentify the ones with "a lot" of exposure:
```

```
   where_excess_exposure    <- which(X[,"Exposure"] > 0.75)

   #Halve their exposure
   X[where_excess_exposure, "Exposure"] <-
            X[where_excess_exposure, "Exposure"]/2
   return(X)
}
pif.confidence(X, thetahat, rr, thetavar, cft, weights = normalized_weights)


#Example 3: Multivariate Linear Relative Risk
#-----------------------------------------
set.seed(18427)
X1       <- rnorm(100,4,1)
X2       <- rnorm(100,2,0.4)
thetahat <- c(0.12, 0.03)
thetavar <- diag(c(0.01, 0.02))

#But the approximate method crashes due to operator
Xmean <- data.frame(Exposure = mean(X1),
                    Covariate = mean(X2))
Xvar  <- var(cbind(X1, X2))

#When creating relative risks avoid using the $ operator
#as it doesn't work under approximate method of PAF
rr_not    <- function(X, theta){
               exp(theta[1]*X$Exposure + theta[2]*X$Covariate)
             }
rr_better <- function(X, theta){
               exp(theta[1]*X[,"Exposure"] + theta[2]*X[,"Covariate"])
             }

#Creating a counterfactual.
cft  <- function(X){
   Y              <- X
   Y[,"Exposure"]  <- 0.5*X[,"Exposure"]
   Y[,"Covariate"] <- 1.1*X[,"Covariate"] + 1
   return(Y)
}

pif.confidence(Xmean, thetahat, rr_better, thetavar, cft,
method = "approximate", Xvar = Xvar)

## End(Not run)

## Not run:
#Error: $ operator in rr definitions don't work in approximate
pif.confidence(Xmean, thetahat, rr_not, thetavar, cft, method = "approximate", Xvar = Xvar)

## End(Not run)

## Not run:
#Example 4: Categorical Relative Risk & Exposure
```

```
#-----------------------------------------
set.seed(18427)
mysample  <- sample(c("Normal","Overweight","Obese"), 100,
                    replace = TRUE, prob = c(0.4, 0.1, 0.5))
X        <- data.frame(Exposure = mysample)

thetahat <- c(1, 1.2, 1.5)
thetavar <- diag(c(0.1, 0.2, 0.3))

#Categorical relative risk function
rr <- function(X, theta){

   #Create return vector with default risk of 1
   r_risk <- rep(1, nrow(X))

   #Assign categorical relative risk
   r_risk[which(X[,"Exposure"] == "Normal")]      <- thetahat[1]
   r_risk[which(X[,"Exposure"] == "Overweight")]  <- thetahat[2]
   r_risk[which(X[,"Exposure"] == "Obese")]       <- thetahat[3]

   return(r_risk)
}


#Counterfactual of reducing all obesity to normality
cft <- function(X){
   X[which(X[,"Exposure"] == "Obese"),] <- "Normal"
   return(X)
}

pif.confidence(X, thetahat, rr, thetavar, cft, check_rr = FALSE)


#Example 5: Categorical Relative Risk & continuous exposure
#-----------------------------------------------------------
set.seed(18427)
BMI      <- data.frame(Exposure = rlnorm(100, 3.1, sdlog = 0.1))

#Theoretical minimum risk exposure is 20kg/m^2 in borderline "Normal" category
BMI_adjusted <- BMI - 20

thetahat <- c(Malnourished = 2.2, Normal = 1, Overweight = 1.8,
              Obese = 2.5)
thetavar <- diag(c(0.1, 0.2, 0.2, 0.1))

rr       <- function(X, theta){

     #Create return vector with default risk of 1
     r_risk <- rep(1, nrow(X))

     #Assign categorical relative risk
     r_risk[which(X[,"Exposure"] < 0)]             <- theta[1] #Malnourished
     r_risk[intersect(which(X[,"Exposure"] >= 0),
```

```
                    which(X[,"Exposure"] < 5))]  <- theta[2] #Normal
     r_risk[intersect(which(X[,"Exposure"] >= 5),
                    which(X[,"Exposure"] < 10))] <- theta[3] #Overweight
     r_risk[which(X[,"Exposure"] >= 10)]          <- theta[4] #Obese

   return(r_risk)
}

#Counterfactual of everyone in normal range
cft <- function(bmi){
     bmi           <- data.frame(rep(2.5, nrow(bmi)), ncol = 1)
     colnames(bmi) <- c("Exposure")
     return(bmi)
}

pif.confidence(BMI_adjusted, thetahat, rr, thetavar, cft,
                check_exposure = FALSE, method = "empirical")


#Example 6: Bivariate exposure and rr ("classical PAF")
#------------------------------------------------------------------
set.seed(18427)
mysample  <- sample(c("Exposed","Unexposed"), 1000,
                replace = TRUE, prob = c(0.1, 0.9))
X         <- data.frame(Exposure = mysample)
theta     <- c("Exposed" = 2.5, "Unexposed" = 1.2)
thetavar  <- matrix(c(0.04, 0.02, 0.02, 0.03), ncol = 2)
rr        <- function(X, theta){

   #Create relative risk function
   r_risk <- rep(1, nrow(X))

   #Assign values of relative risk
   r_risk[which(X[,"Exposure"] == "Unexposed")] <- theta["Unexposed"]
   r_risk[which(X[,"Exposure"] == "Exposed")]   <- theta["Exposed"]

   return(r_risk)
}

#Counterfactual of reducing the exposure in half of the individuals
cft <- function(X){

   #Find out which ones are exposed
   Xexp  <- which(X[,"Exposure"] == "Exposed")

   #Use only half of the exposed randomly
   reduc <- sample(Xexp, length(Xexp)/2)

   #Unexpose those individuals
   X[reduc, "Exposure"] <- "Unexposed"

   return(X)
}
```

```
pif.confidence(X, theta, rr, thetavar, cft)

#Example 7: Continuous exposure, several covariates
#-----------------------------------------------------------------------
X <- data.frame(Exposure = rbeta(100, 2, 3),
                Age      = runif(100, 20, 100),
                Sex      = sample(c("M","F"), 100, replace = TRUE),
                BMI      = rlnorm(100, 3.2, 0.2))
thetahat <- c(-0.1, 0.05, 0.2, -0.4, 0.3, 0.1)

#Create variance of theta
almostvar <- matrix(runif(6^2), ncol = 6)

thetavar <- t(almostvar) %*% almostvar

rr <- function(X, theta){
     #Create risk vector
     Risk    <- rep(1, nrow(X))

     #Identify subpopulations
     males   <- which(X[,"Sex"] == "M")
     females <- which(X[,"Sex"] == "F")

     #Calculate population specific rr
     Risk[males] <- theta[1]*X[males,"Exposure"] +
                                     theta[2]*X[males,"Age"]^2 +
                                     theta[3]*X[males,"BMI"]/2

     Risk[females] <- theta[4]*X[females,"Exposure"] +
                                     theta[5]*X[females,"Age"]^2 +
                                     theta[6]*X[females,"BMI"]/2

    return(Risk)
}

#Counterfactual of reducing BMI
cft <- function(X){
    excess_bmi           <- which(X[,"BMI"] > 25)
    X[excess_bmi,"BMI"]  <- 25
    return(X)
}

pif.confidence(X, thetahat, rr, thetavar, cft)

## End(Not run)
```

---

`pif.heatmap` *Graphical Sensitivity Analysis of Potential Impact Fraction's Counterfactual*

---

## Description

Provides a graphical sensitivity analysis for `pif` by varying the parameters of a bivariate counterfactual function `cft`. By default it evaluates the counterfactual:

$$\text{cft}(X) = aX + b.$$

## Usage

```
pif.heatmap(X, thetahat, rr, cft = function(X, a, b) {     a * X + b },
  method = "empirical", weights = rep(1/nrow(as.matrix(X)),
  nrow(as.matrix(X))), Xvar = var(X), deriv.method.args = list(),
  deriv.method = "Richardson", adjust = 1, n = 512, ktype = "gaussian",
  bw = "SJ", legendtitle = "PIF", mina = 0, maxa = 1, minb = -1,
  maxb = 0, nmesh = 10,
  title = paste0("Potential Impact Fraction (PIF) with counterfactual",
  "\nf(X)= aX+b"), xlab = "a", ylab = "b",
  colors = rev(heat.colors(nmesh)), check_exposure = TRUE,
  check_rr = TRUE, check_integrals = TRUE)
```

## Arguments

| | |
|---|---|
| `X` | Random sample (`data.frame`) which includes exposure and covariates or sample `mean` if `"approximate"` method is selected. |
| `thetahat` | Asymptotically consistent or Fisher consistent estimator (`vector`) of `theta` for the Relative Risk function. |
| `rr` | `function` for Relative Risk which uses parameter `theta`. The order of the parameters shound be `rr(X, theta)`. |
| | ****Optional**** |
| `cft` | `function cft(X, a, b)` for counterfactual dependent on one dimensional parameters `a` and `b`. Default counterfactual is affine: `aX + b`. |
| `method` | Either `"empirical"` (default), `"kernel"` or `"approximate"`. For details on estimation methods see `pif`. |
| `weights` | Normalized survey `weights` for the sample `X`. |
| `Xvar` | Variance of exposure levels (for `"approximate"` method). |
| `deriv.method.args` | `method.args` for `hessian` (for `"approximate"` method). |
| `deriv.method` | `method` for `hessian`. Don't change this unless you know what you are doing (for `"approximate"` method). |
| `adjust` | Adjust bandwith parameter (for `"kernel"` method) from `density`. |
| `n` | Number of equally spaced points at which the density (for `"kernel"` method) is to be estimated (see `density`). |
| `ktype` | `kernel` type: `"gaussian"`, `"epanechnikov"`, `"rectangular"`, `"triangular"`, `"biweight"`, `"cosine"`, `"optcosine"` (for `"kernel"` method). Additional information on kernels in `density`. |

| | |
|---|---|
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| legendtitle | string title for the legend of plot. |
| mina | Minimum for parameter a for the counterfactual (default 0). |
| maxa | Maximum for parameter a for the counterfactual (default 1). |
| minb | Minimum for parameter b for the counterfactual (default -1). |
| maxb | Maximum for parameter b for the counterfactual (default 0). |
| nmesh | Number of tiles in plot (default 10). |
| title | string title for the plot. |
| xlab | string label for the X-axis of the plot (corresponding to "a"). |
| ylab | string label for the Y-axis of the plot (corresponding to "b"). |
| colors | vector of colours for the heatmap. |
| check_exposure | boolean Check that exposure X is positive and numeric. |
| check_rr | boolean Check that Relative Risk function rr equals 1 when evaluated at 0. |
| check_integrals | boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario. |

## Value

plotpif ggplot object plotting a heatmap with sensitivity analysis of the counterfactual.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

See pif for Potential Impact Fraction estimation, pif.sensitivity for sensitivity analysis of the convergence process, pif.plot for a plot of Potential Impact Fraction as a function of the relative risk's parameter theta.

## Examples

```
## Not run:
#Example 1
#-------------------------------------------------------------------
X  <- data.frame(rnorm(25,3))           #Sample
rr <- function(X,theta){exp(X*theta)}   #Relative risk
theta <- 0.01                           #Estimate of theta
pif.heatmap(X, theta = theta, rr = rr)

#Save file using ggplot2
#require(ggplot2)
#ggsave("My Potential Impact Fraction Heatmap Analysis.pdf")
```

```
#Change pif estimation method to kernel

pif.heatmap(X, theta = theta, rr = rr, method = "kernel")

#Example 2
#-----------------------------------------------------------------
X     <- data.frame(Exposure = rbeta(100, 1, 0.2))
theta <- c(0.12, 1)
rr    <- function(X,theta){X*theta[1] + theta[2]}
cft   <- function(X, a, b){sin(a*X)*b}
pif.heatmap(X, theta = theta, rr = rr, cft = cft,
     nmesh = 15, colors = rainbow(30), method = "empirical",
     title = "PIF with counterfactual cft(X) = sin(a*X)*b")

#Change estimation method to approximate
Xmean <- data.frame(mean(X["Exposure"]))
Xvar  <- var(X)
pif.heatmap(Xmean, Xvar = Xvar, theta = theta, rr = rr, cft = cft,
     nmesh = 15, colors = rainbow(30), method = "approximate",
     title = "PIF with counterfactual cft(X) = sin(a*X)*b")


#Example 3: Plot univariate counterfactuals
#-----------------------------------------------------------------
X       <- data.frame(rgamma(100, 1, 0.2))
theta   <- c(0.12, 1)
rr      <- function(X,theta){X*theta[1] + theta[2]}
cft     <- function(X, a, b){sqrt(a*X)}   #Leave two variables in it
pifplot <- pif.heatmap(X, theta = theta, rr = rr,
 cft = cft, mina = 0, maxa = 1, minb = 0, maxb = 0,
 legendtitle = "Potential Impact Fraction",
 title ="Univariate counterfactual", ylab = "", colors = topo.colors(10))
pifplot

#You can also add additional ggplot objects
#require(ggplot2)
#pifplot + annotate("text", x = 0.25, y = 0.4, label = "10yr scenario") +
#geom_vline(aes(xintercept = 0.5), linetype = "dashed") +
#geom_segment(aes(x = 0.25, y = 0.38, xend = 0.5, yend = 0.3),
#arrow = arrow(length = unit(0.25, "cm")))

#Example 4: Plot counterfactual with categorical risks
#-----------------------------------------------------------------
set.seed(18427)
X        <- data.frame(Exposure =
             sample(c("Normal","Overweight","Obese"), 100,
                    replace = TRUE, prob = c(0.4, 0.1, 0.5)))
thetahat <- c(1, 1.7, 2)

#Categorical relative risk function
rr <- function(X, theta){
```

```
   #Create return vector with default risk of 1
   r_risk <- rep(1, nrow(X))

   #Assign categorical relative risk
   r_risk[which(X[,"Exposure"] == "Normal")]      <- thetahat[1]
   r_risk[which(X[,"Exposure"] == "Overweight")]  <- thetahat[2]
   r_risk[which(X[,"Exposure"] == "Obese")]       <- thetahat[3]

   return(r_risk)
}

#Counterfactual of reducing a certain percent of obesity and overweight cases
#to normality
cft <- function(X, per.over, per.obese){

   #Find the overweight and obese individuals
   which_obese <- which(X[,"Exposure"] == "Obese")
   which_over  <- which(X[,"Exposure"] == "Overweight")

   #Reduce per.over % of overweight and per.obese % of obese
   X[sample(which_obese, length(which_obese)*per.obese),
       "Exposure"] <- "Normal"
   X[sample(which_over,  length(which_over)*per.over),
       "Exposure"] <- "Normal"

   return(X)
}

pif.heatmap(X, thetahat = thetahat, rr = rr, cft = cft, mina = 0, minb = 0, maxa = 1,
            maxb = 1, title = "PIF of excess-weight reduction",
            xlab = "% Overweight cases", ylab = "% Obese cases",
            check_exposure = FALSE, check_rr = FALSE)

## End(Not run)
```

---

`pif.plot` *Plot of Potential Impact Fraction under different values of Relative Risk's parameter theta*

---

## Description

Function that plots the `pif` under different values of a univariate parameter `theta` of the relative risk function `rr` which depends on the exposure `X` and a parameter `theta` (`rr(X, theta)`)

## Usage

```
pif.plot(X, thetalow, thetaup, rr, cft = NA, method = "empirical",
  confidence_method = "bootstrap", confidence = 95, nsim = 100,
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), mpoints = 100,
  adjust = 1, n = 512, Xvar = var(X), deriv.method.args = list(),
```

```
deriv.method = "Richardson", ktype = "gaussian", bw = "SJ",
colors = c("deepskyblue", "gray25"), xlab = "Theta", ylab = "PIF",
title = "Potential Impact Fraction (PIF) under different values of theta",
check_exposure = TRUE, check_rr = TRUE, check_integrals = TRUE,
check_cft = TRUE, check_thetas = TRUE, check_xvar = TRUE,
is_paf = FALSE)
```

## Arguments

| | |
|---|---|
| X | Random sample (data.frame) which includes exposure and covariates or sample mean if "approximate" method is selected. |
| thetalow | Minimum of theta (parameter of relative risk rr) for plot. |
| thetaup | Maximum of theta (parameter of relative risk rr) for plot. |
| rr | function for Relative Risk which uses parameter theta. The order of the parameters should be rr(X, theta).<br>**Optional** |
| cft | Function cft(X) for counterfactual. Leave empty for the Population Attributable Fraction paf where counterfactual is that of the theoretical minimum risk exposure X0 such that rr(X0,theta) = 1. |
| method | Either "empirical" (default), "kernel" or "approximate". For details on estimation methods see pif. |
| confidence_method | Either bootstrap (default), linear, loglinear. See paf details for additional information. |
| confidence | Confidence level % (default 95). |
| nsim | Number of simulations to generate confidence intervals. |
| weights | Normalized survey weights for the sample X. |
| mpoints | Number of points in plot. |
| adjust | Adjust bandwith parameter (for "kernel" method) from density. |
| n | Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density). |
| Xvar | Variance of exposure levels (for "approximate" method). |
| deriv.method.args | method.args for hessian (for "approximate" method). |
| deriv.method | method for hessian. Don't change this unless you know what you are doing (for "approximate" method). |
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density. |
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| colors | vector Colours of plot. |
| xlab | string Label of x-axis in plot. |

| | |
|---|---|
| ylab | string Label of y-axis in plot. |
| title | string Title of plot. |
| check_exposure | boolean Check that exposure X is positive and numeric. |
| check_rr | boolean Check that Relative Risk function rr equals 1 when evaluated at 0. |
| check_integrals | boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario. |
| check_cft | boolean Check that counterfactual function cft reduces exposure. |
| check_thetas | boolean Check that theta associated parameters are correctly inputed for the model. |
| check_xvar | boolean Check Xvar is covariance matrix. |
| is_paf | Boolean forcing evaluation of paf. This forces the pif function ignore the inputed counterfactual and set it to the theoretical minimum risk value of 1. |

## Value

pif.plot ggplot object with plot of Potential Impact Fraction as function of theta.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

See pif for Potential Impact Fraction estimation with confidence intervals pif.confidence.

See paf.plot for same plot with Population Attributable Fraction paf.

## Examples

```
#Example 1: Exponential Relative Risk empirical method
#-----------------------------------------------------
## Not run:
set.seed(18427)
X <- data.frame(Exposure = rbeta(25, 4.2, 10))
pif.plot(X, thetalow = 0, thetaup = 10, rr =  function(X, theta){exp(theta*X)})

#Same example with kernel method
pif.plot(X, thetalow = 0, thetaup = 10, rr =  function(X, theta){exp(theta*X)},
method = "kernel", title = "Kernel method example")

#Same example for approximate method. Notice that approximate method has
#more uncertainty
Xmean <- data.frame(mean(X[,"Exposure"]))
Xvar  <- var(X)
pif.plot(Xmean, thetalow = 0, thetaup = 10, rr =  function(X, theta){exp(theta*X)},
method = "approximate", Xvar = Xvar, title = "Approximate method example")
```

```
#Example with counterfactual
pif.plot(X, thetalow = -10, thetaup = -5, rr = function(X, theta){exp(theta*X)},
cft = function(X){sqrt(X)})

#Example for approximate method with square root counterfactual
#Notice how the approximate represents information loss and thus the interval
#loses precision.
pif.plot(Xmean, thetalow = -10, thetaup = -5, rr = function(X, theta){exp(theta*X)},
cft = function(X){sqrt(X)}, method = "approximate", Xvar = Xvar)

## End(Not run)
```

---

`pif.sensitivity` *Potential Impact Fraction Sensitivity Analysis plot*

---

## Description

Function that plots a sensitivity analysis for the Potential Impact Fraction `pif` by checking how estimates vary when reducing the exposure's sample `X`.

## Usage

```
pif.sensitivity(X, thetahat, rr, cft = NA, method = "empirical",
  weights = rep(1/nrow(as.matrix(X)), nrow(as.matrix(X))), nsim = 50,
  mremove = min(nrow(as.matrix(X))/2, 100), adjust = 1, n = 512,
  ktype = "gaussian", bw = "SJ", ylab = "PIF",
  xlab = "Number of randomly deleted observations for X",
  legendtitle = "Sensitivity Analysis",
  title = "Potential Impact Fraction (PIF) Sensitivity Analysis",
  colors = c("red", "deepskyblue", "gray75", "gray25"),
  check_exposure = TRUE, check_rr = TRUE, check_integrals = TRUE,
  is_paf = FALSE)
```

## Arguments

| | |
|---|---|
| `X` | Random sample (`data.frame`) which includes exposure and covariates or sample `mean` if `"approximate"` method is selected. |
| `thetahat` | Asymptotically consistent or Fisher consistent estimator (`vector`) of `theta` for the Relative Risk function. |
| `rr` | `function` for Relative Risk which uses parameter `theta`. The order of the parameters should be `rr(X, theta)`. |
| | ****Optional**** |
| `cft` | `function cft(X)` for counterfactual. Leave empty for the Population Attributable Fraction `paf` where counterfactual is that of a theoretical minimum risk exposure `X0` such that `rr(X0,theta) = 1`. |

| | |
|---|---|
| method | Either "empirical" (default), "kernel" or "approximate". For details on estimation methods see pif. |
| weights | Normalized survey weights for the sample X. |
| nsim | Integer with number of samples to include (for each removal) in order to conduct sensitivity analysis. See details for additional information. |
| mremove | Limit number of measurements of X to remove when resampling. See details for additional information. |
| adjust | Adjust bandwith parameter (for "kernel" method) from density. |
| n | Number of equally spaced points at which the density (for "kernel" method) is to be estimated (see density). |
| ktype | kernel type: "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine" (for "kernel" method). Additional information on kernels in density. |
| bw | Smoothing bandwith parameter (for "kernel" method) from density. Default "SJ". |
| ylab | string label for the Y-axis of the plot. |
| xlab | string label for the X-axis of the plot. |
| legendtitle | string title for the legend of plot. |
| title | string title of plot. |
| colors | string vector with colours for plots. |
| check_exposure | boolean Check that exposure X is positive and numeric. |
| check_rr | boolean Check that Relative Risk function rr equals 1 when evaluated at 0. |
| check_integrals | boolean Check that counterfactual cft and relative risk's rr expected values are well defined for this scenario. |
| is_paf | Boolean forcing evaluation of paf. This forces the pif function ignore the inputed counterfactual and set it to the theoretical minimum risk value of 1. |

## Details

pif.sensitivity conducts a sensitivity analysis of the pif estimate by removing mremove elements nsim times and re-estimating pif with the reduced sample.

## Value

plotpif ggplot object plotting a sensitivity analysis of pif.

## Author(s)

Rodrigo Zepeda-Tello <rzepeda17@gmail.com>

Dalia Camacho-García-Formentí <daliaf172@gmail.com>

## See Also

See pif for Potential Impact Fraction estimation, pif.heatmap for sensitivity analysis of counterfactual, pif.plot for a plot of Potential Impact Fraction as a function of the relative risk's parameter theta.

## Examples

```
## Not run:
#Example 1
#-------------------------------------------------------------------
set.seed(3284)
X  <- data.frame(Exposure = rnorm(250,3)) #Sample
rr <- function(X,theta){exp(X*theta)}     #Relative risk
theta <- 0.1                              #Estimate of theta

pif.sensitivity(X, thetahat = theta, rr = rr)


#Save file
#require(ggplot2)
#ggsave("My Potential Impact Fraction Sensitivity Analysis.pdf")

#Example 2
#-----------------------------------------------------------------
set.seed(3284)
X     <- data.frame(Exposure = rbeta(1000, 1, 0.2))
theta <- c(0.12, 1)
rr    <- function(X, theta){X*theta[1] + theta[2]}
cft   <- function(X){X/2}


#Using empirical method
pif.sensitivity(X, thetahat = theta, rr = rr, cft = cft,
                mremove = 100, nsim = 50,
                title = "My Sensitivity Analysis for example 1")

#Same example with kernel
pif.sensitivity(X, theta, rr = rr, cft = cft,
                 mremove = 100, nsim = 50, method = "kernel",
                 title = "Sensitivity Analysis for example 1 using kernel")


#Example 3: Plot counterfactual with categorical risks
#-------------------------------------------------------------------
set.seed(18427)
X        <- data.frame(Exposure =
               sample(c("Normal","Overweight","Obese"), 1000,
                      replace = TRUE, prob = c(0.4, 0.1, 0.5)))
thetahat <- c(1, 1.7, 2)

#Categorical relative risk function
rr <- function(X, theta){
```

```
   #Create return vector with default risk of 1
   r_risk <- rep(1, length(X))

   #Assign categorical relative risk
   r_risk[which(X == "Normal")]      <- thetahat[1]
   r_risk[which(X == "Overweight")]  <- thetahat[2]
   r_risk[which(X == "Obese")]       <- thetahat[3]

   return(r_risk)
}


#Counterfactual of halving the percent of obesity and overweight cases
#to normality
cft <- function(X){

   #Find the overweight and obese individuals
   which_obese <- which(X == "Obese")
   which_over  <- which(X == "Overweight")

   #Reduce per.over % of overweight and per.obese % of obese
   X[sample(which_obese, floor(length(which_obese)*0.5)),1] <- "Normal"
   X[sample(which_over,  floor(length(which_over)*0.5)),1]  <- "Normal"

   return(X)
}


pifplot <- pif.sensitivity(X, thetahat = thetahat, rr = rr, cft = cft,
                           title = "Sensitivity analysis of PIF for excess-weight",
                           colors = rainbow(4),
                           legendtitle = "Values",
                           check_exposure = FALSE, check_rr = FALSE)
pifplot

#You can edit pifplot as it is a ggplot object
#require(ggplot2)
#pifplot + theme_classic()

## End(Not run)
```

---

pifpaf *pifpaf: A package for estimating the Potential Impact Fraction and the Population Attributable Fraction from cross-sectional data*

---

## Description

Uses a generalized method to estimate the Potential Impact Fraction, (PIF) and the Population Attributable Fraction (PAF) from cross-sectional data. It creates point-estimates (pif, paf), con-

fidence intervals (pif.confidence, paf.confidence),and estimates of variance. In addition it generates plots for conducting sensitivity analysis (pif.sensitivity, pif.heatmap, pif.plot). The estimation method corresponds to Zepeda-Tello, Camacho-García-Formentí, et al, 'Nonparametric Methods to Estimate the Potential Impact Fraction from Cross-sectional Data', Unpublished manuscript. 2017. This package was developed by the National Institute of Public Health of Mexico under funding by Bloomberg Philanthropies. Please see the package's vignette for more information: `browseVignettes("pifpaf")`.

# Index