

# Package ‘bpbounds’

May 24, 2026

**Title** Nonparametric Bounds for the Average Causal Effect Due to Balke and Pearl and Extensions

**Version** 0.1.7

**Description** Implementation of the nonparametric bounds for the average causal effect under an instrumental variable model by Balke and Pearl (Bounds on Treatment Effects from Studies with Imperfect Compliance, JASA, 1997, 92, 439, 1171-1176, <[doi:10.1080/01621459.1997.10474074](https://doi.org/10.1080/01621459.1997.10474074)>). The package can calculate bounds for a binary outcome, a binary treatment/phenotype, and an instrument with either 2 or 3 categories. The package implements bounds for situations where these 3 variables are measured in the same dataset (trivariate data) or where the outcome and instrument are measured in one study and the treatment/phenotype and instrument are measured in another study (bivariate data).

**License** GPL-3

**URL** <https://github.com/remlapmot/bpbounds>,  
<https://remlapmot.github.io/bpbounds/>,  
<https://mrcieu.r-universe.dev/bpbounds>

**BugReports** <https://github.com/remlapmot/bpbounds/issues>

**Depends** R (>= 4.1)

**Suggests** knitr, rmarkdown, shiny, testthat, tidyr

**VignetteBuilder** knitr

**Config/roxygen2/markdown** TRUE

**Config/roxygen2/version** 8.0.0

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Tom Palmer [aut, cre] (ORCID: <<https://orcid.org/0000-0003-4655-4511>>),  
Roland Ramsahai [aut] (ORCID: <<https://orcid.org/0000-0002-7349-1977>>),  
Vanessa Didelez [aut] (ORCID: <<https://orcid.org/0000-0001-8587-7706>>),  
Nuala Sheehan [aut] (ORCID: <<https://orcid.org/0000-0001-9094-2605>>)

**Maintainer** Tom Palmer <remlapmot@hotmail.com>

**Repository** CRAN

**Date/Publication** 2026-05-24 15:50:02 UTC

## Contents

bpbounds	2
runExample	4

<b>Index</b>	<b>6</b>
--------------	----------

---

bpbounds	<i>Nonparametric Bounds for the Average Causal Effect due to Balke and Pearl.</i>
----------	---

---

## Description

Nonparametric Bounds for the Average Causal Effect due to Balke and Pearl.

## Usage

```
bpbounds(p, t = NULL, fmt = "trivariate")
```

## Arguments

**p** Object of class "table" containing either cell counts or conditional probabilities. For trivariate data these are for the phenotype/treatment-outcome association given Z, i.e.  $P(X, Y | Z)$ .

Cell counts could be generated from `xtabs(~ x + y + z, data = data)`. And then conditional probabilities obtained by calling `prop.table(..., margins = 3)` on your object from `xtabs()`.

If you only know the conditional probabilities you can enter these, e.g. for the Balke and Pearl Vitamin A example:

```
cp <- c(.0064, 0, .9936, 0, .0028, .001, .1972, .799)
tabp <- as.table(array(
  cp,
  dim = c(2, 2, 2),
  dimnames = list(
    x = c(0, 1),
    y = c(0, 1),
    z = c(0, 1)
  )
))
```

And then call `bpbounds()` using this object.

For bivariate data this object contains cell conditional probabilities for the outcome-instrument (Y|Z) association.

t	Specified for bivariate data. Object with treatment/phenotype-instrument cell counts or conditional probabilities, i.e. (X Z).
fmt	A character string which should be either "bivariate" (i.e. X, Z in one dataset and Y, Z in another dataset) or "trivariate" (X, Y, Z in the same dataset).

### Value

List with the following elements:

**fmt** whether the data is bivariate or trivariate  
**nzcats** 2 or 3, the number of instrument categories  
**inequality** Logical, indicating whether the IV inequality is satisfied  
**bp1b** Lower bound of ACE  
**bp1b** Upper bound of ACE  
**bp1lower** Vector of lower bound probabilities  
**bp1upper** Vector of upper bound probabilities  
**p11low** Lower bound of  $P(Y=1|do(X=1))$   
**p11upp** Upper bound of  $P(Y=1|do(X=1))$   
**p10low** Lower bound of  $P(Y=1|do(X=0))$   
**p10upp** Upper bound of  $P(Y=1|do(X=0))$   
**p11lower** Vector of probabilities for lower bound of  $P(Y=1|do(X=1))$   
**p11upper** Vector of probabilities for upper bound of  $P(Y=1|do(X=1))$   
**p10lower** Vector of probabilities for lower bound of  $P(Y=1|do(X=0))$   
**p10upper** Vector of probabilities for upper bound of  $P(Y=1|do(X=0))$   
**crr1b** Lower bound of CRR  
**crr1b** Upper bound of CRR  
**monoinequality** Logical, indicating whether the monotonicity inequality is satisfied  
**monobp1b** Lower bound of ACE assuming monotonicity  
**monobp1b** Upper bound of ACE assuming monotonicity  
**monobp1lower** Vector of probabilities for lower bound of ACE assuming monotonicity  
**monobp1upper** Vector of probabilities for upper bound of ACE assuming monotonicity  
**monop11low** Lower bound of  $P(Y=1|do(X=1))$  assuming monotonicity  
**monop11upp** Upper bound of  $P(Y=1|do(X=1))$  assuming monotonicity  
**monop10low** Lower bound of  $P(Y=1|do(X=0))$  assuming monotonicity  
**monop10upp** Upper bound of  $P(Y=1|do(X=0))$  assuming monotonicity  
**monop11lower** Vector for corresponding bound above  
**monop11upper** Vector for corresponding bound above  
**monop10lower** Vector for corresponding bound above  
**monop10upper** Vector for corresponding bound above  
**monocrr1b** Lower bound of CRR assuming monotonicity  
**monocrr1b** Upper bound of CRR assuming monotonicity

## Examples

```
# Vitamin A example, using cell counts

require(tidyr)
require(bpbounds)

tab1dat <- data.frame(
  z = c(0, 0, 1, 1, 1, 1, 0, 0),
  x = c(0, 0, 0, 0, 1, 1, 1, 1),
  y = c(0, 1, 0, 1, 0, 1, 0, 1),
  freq = c(74, 11514, 34, 2385, 12, 9663, 0, 0)
)
tablinddat <- uncount(tab1dat, freq)
xt <- xtabs(~ x + y + z, data = tablinddat)
p <- prop.table(xt, margin = 3)
bpres <- bpbounds(p)
sbpres <- summary(bpres)
print(sbpres)

# Vitamin A example, using conditional probabilities

require(bpbounds)
cp = c(.0064, 0, .9936, 0, .0028, .001, .1972, .799)
tabp = as.table(array(
  cp,
  dim = c(2, 2, 2),
  dimnames = list(
    x = c(0, 1),
    y = c(0, 1),
    z = c(0, 1)
  )
))
bpbounds(tabp)
```

---

runExample

*Run Shiny App demonstrating the package*

---

## Description

Run Shiny App demonstrating the package

## Usage

```
runExample(...)
```

**Arguments**

... passed to `shiny::runApp()`, e.g. `port`, `launch.browser`

**Examples**

```
if (interactive() && requireNamespace("shiny", quietly = TRUE)) {  
  bpbounds::runExample()  
}
```

# Index

`bpbounds`, [2](#)

`runExample`, [4](#)

`shiny::runApp()`, [5](#)