Package 'migest'

February 5, 2018

Type Package

Title Methods for the Indirect Estimation of Bilateral Migration

Version 1.7.4

Author Guy J. Abel

Maintainer Guy J. Abel <g.j.abel@gmail.com>

Description Indirect methods for estimating bilateral migration flows in the presence of partial or missing data. Methods might be relevant to other categorical data situations on non-migration data, where for example, marginal totals are known and only auxiliary bilateral data is available.

URL https://github.com/gjabel/migest/

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-05 17:34:20 UTC

R topics documented:

`migest-package` *Methods for the Indirect Estimation of Bilateral Migration*

Description

The migest package contains a collection of R functions for indirect methods to estimate bilateral migration flows in the presence of partial or missing data. Methods might be relevant to other categorical data situations on non-migration data, where for example, marginal totals are known and only auxiliary bilateral data is available.

Details

Package:	migest
Type:	Package
License:	GPL-2

The estimation methods in this package can be grouped as 1) functions for origin-destination matrices (`cm2` and `ipf2`) and 2) functions for origin-destination matrices categorized by a further set of characteristics, such as ethnicity, employment or health status (`cm3`, `ipf3` and `ipf3_qi`). All these routines are based on indirect estimation methods where marginal totals are known, and a Poisson regression (log-linear) model is assumed.

The flow from stock function, `ffs` acts as a wrapper for a combination of some of these estimation routines with further adjustments for changes in foreign born stocks over a period. The demo files, `demo(cfplot_reg2)`, `demo(cfplot_reg)` and `demo(cfplot_nat)`, produce circular migration flow plots for migration estimates from Abel(2017) and Abel and Sander (2014), which were derived using the `ffs` function.

Blog posts with some additional details of the implementation of functions in the package can be found at http://gjabel.wordpress.com/category/r/migest/

Github repo: http://github.com/gjabel/migest

Author(s)

Guy J. Abel

References

Abel, G. J. (2017). Estimates of Global Bilateral Migration Flows by Gender between 1960 and 2015. *International Migration Review*.

Abel, G. J. (2013). Estimating Global Migration Flow Tables Using Place of Birth. *Demographic Research* 28, (18) 505-546

Abel, G. J. (2005) *The Indirect Estimation of Elderly Migrant Flows in England and Wales* (MS.c. Thesis). University of Southampton

Abel, G. J. and Sander, N. (2014). Quantifying Global International Migration Flows. *Science*, 343 (6178) 1520-1522

Raymer, J., G. J. Abel, and P. W. F. Smith (2007). Combining census and registration data to estimate detailed elderly migration flows in England and Wales. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 170 (4), 891–908.

Willekens, F. (1999). Modelling Approaches to the Indirect Estimation of Migration Flows: From Entropy to EM. *Mathematical Population Studies* 7 (3), 239–78.

`block_matrix` *Create a Block Matrix with Non-Uniform Block Sizes.*

Description

Creates a `matrix` with differing size blocks

Usage

```
block_matrix(x = NULL, b = NULL, byrow = FALSE, dimnames = NULL)
```

Arguments

`x`	Vector of numbers to identify each block.
`b`	Numeric value for the size of the blocks within the matrix ordered depending on `byrow`
`byrow`	Logical value. If `FALSE` (the default) the blocks are filled by columns, otherwise the blocks in the matrix are filled by rows.
`dimnames`	Character string of name attribute for the basis of the blcok matrix. If `NULL` a vector of the same length of `b` provides the basis of row and column names.#'

Value

Returns a `matrix` with block sizes determined by the `b` argument. Each block is filled with the same value taken from `x`.

Author(s)

Guy J. Abel

See Also

stripe_matrix, block_sum, ipf2_block

Examples

```
block_matrix(x = 1:16, b = c(2,3,4,2))
block_matrix(x = 1:25, b = c(2,3,4,2,1))
```

---

block_sum *Sum of Selected Block in a Block Matrix*

Description

Returns of a sum of a block within a matrix. This function is predominantly intended to be used within the ipf2_block routine.

Usage

```
block_sum(block = NULL, m = NULL, block_id = NULL)
```

Arguments

block	Numeric value of block to summed. To be matched against the matrix in block_id.
m	Matrix of all blocks combined.
block_id	Matrix of the same dimensions of m used to identify blocks.

Value

Returns a numeric value of the sum of a single block.

Author(s)

Guy J. Abel

See Also

block_matrix, stripe_matrix, ipf2_block

Examples

```
m <- matrix(data = 100:220, nrow = 11, ncol = 11)
b <- block_matrix(x = 1:16, b = c(2, 3, 4, 2))
block_sum(block = 1, m = m, block_id = b)
block_sum(block = 4, m = m, block_id = b)
block_sum(block = 16, m = m, block_id = b)
```

cm2 *Conditional Maximization Routine for the Indirect Estimation of Origin-Destination Migration Flow Table with Known Margins.*

Description

The cm2 function finds the maximum likelihood estimates for parameters in the log-linear model:

$$\log y_{ij} = \log \alpha_i + \log \beta_j + \log m_{ij}$$

as introduced by Willekens (1999). The α_i and β_j represent background information related to the characteristics of the origin and destinations respectively. The m_{ij} factor represents auxiliary information on migration flows, which imposes its interaction structure onto the estimated flow matrix.

Usage

```
cm2(rtot = NULL, ctot = NULL, m = matrix(1, length(rtot), length(ctot)),
  tol = 1e-05, maxit = 500, verbose = TRUE)
```

Arguments

rtot	Vector of origin totals to constrain the sum of the imputed cell rows.
ctot	Vector of destination totals to constrain the sum of the imputed cell columns.
m	Matrix of auxiliary data. By default set to 1 for all origin-destination combinations.
tol	Numeric value for the tolerance level used in the parameter estimation.
maxit	Numeric value for the maximum number of iterations used in the parameter estimation.
verbose	Logical value to indicate the print the parameter estimates at each iteration. By default FALSE.

Value

Parameter estimates are obtained using the EM algorithm outlined in Willekens (1999). This is equivalent to a conditional maximization of the likelihood, as discussed by Raymer et. al. (2007). It also provides identical indirect estimates to those obtained from the ipf2 routine.

The user must ensure that the row and column totals are equal in sum. Care must also be taken to allow the dimension of the auxiliary matrix (m) to equal those provided in the row (rtot) and column (ctot) arguments.

Returns a list object with

N	Origin-Destination matrix of indirect estimates
theta	Collection of parameter estimates

Author(s)

Guy J. Abel

References

Raymer, J., G. J. Abel, and P. W. F. Smith (2007). Combining census and registration data to estimate detailed elderly migration flows in England and Wales. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 170 (4), 891–908.

Willekens, F. (1999). Modelling Approaches to the Indirect Estimation of Migration Flows: From Entropy to EM. *Mathematical Population Studies* 7 (3), 239–78.

See Also

ipf2

Examples

```
## with Willekens (1999) data
dn <- LETTERS[1:2]
y <- cm2(rtot = c(18, 20), ctot = c(16, 22),
         m = matrix(c(5, 1, 2, 7), ncol = 2, dimnames = list(orig = dn, dest = dn)))
y

## with all elements of offset equal (independence fit)
y <- cm2(rtot = c(18, 20), ctot = c(16, 22))
y

## with bigger matrix
dn <- LETTERS[1:3]
y <- cm2(rtot = c(170, 120, 410), ctot = c(500, 140, 60),
         m = matrix(c(50, 10, 220, 120, 120, 30, 545, 0, 10),
                    ncol = 3,
                    dimnames = list(orig = dn, dest = dn)))

# display with row and col totals
round(addmargins(y$N))
```

---

cm3 — *Conditional Maximization Routine for the Indirect Estimation of Origin-Destination-Migrant Type Migration Flow Tables with Known Origin and Destination Margins.*

Description

The cm3 function finds the maximum likelihood estimates for parameters in the log-linear model:

$$\log y_{ijk} = \log \alpha_i + \log \beta_j + \log m_{ijk}$$

as introduced by Abel (2005). The α_i and β_j represent background information related to the characteristics of the origin and destinations respectively. The m_{ijk} factor represents auxiliary information on origin-destination migration flows by a migrant characteristic (such as age, sex, disability, household type, economic status, etc.). This method is useful for combining data from detailed data collection processes (such as a Census) with more up-to-date information on migration inflows and outflows (where details on movements by migrant characteristics are not known).

Usage

```
cm3(rtot = NULL, ctot = NULL, m, tol = 1e-05, maxit = 500,
  verbose = FALSE)
```

Arguments

`rtot`	Vector of origin totals to constrain the sum of the imputed cell rows.
`ctot`	Vector of destination totals to constrain the sum of the imputed cell columns.
`m`	Array of auxiliary data. By default set to 1 for all origin-destination-migrant typology combinations.
`tol`	Numeric value for the tolerance level used in the parameter estimation.
`maxit`	Numeric value for the maximum number of iterations used in the parameter estimation.
`verbose`	Logical value to indicate the print the parameter estimates at each iteration. By default `FALSE`.

Value

Parameter estimates were obtained using the conditional maximization of the likelihood, as discussed by Abel (2005) and Raymer et. al. (2007).

The user must ensure that the row and column totals are equal in sum. Care must also be taken to allow the row and column dimension of the auxiliary matrix (`m`) to equal those provided in the row and column totals.

Returns a `list` object with

`N`	Origin-Destination matrix of indirect estimates
`theta`	Collection of parameter estimates

Author(s)

Guy J. Abel

References

Abel, G. J. (2005) *The Indirect Estimation of Elderly Migrant Flows in England and Wales* (MS.c. Thesis). University of Southampton

Raymer, J., G. J. Abel, and P. W. F. Smith (2007). Combining census and registration data to estimate detailed elderly migration flows in England and Wales. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 170 (4), 891–908.

See Also

cm2, ipf3

Examples

```
## over two tables
dn <- LETTERS[1:2]
y <- cm3(rtot = c(18, 20) * 2, ctot = c(16, 22) * 2,
         m = array(c(5, 1, 2, 7, 4, 2, 5, 9), dim = c(2, 2, 2),
                   dimnames = list(orig = dn, dest = dn, type = c("ILL", "HEALTHY"))))
# display with row, col and table totals
y

## over three tables
y <- cm3(rtot = c(170, 120, 410), ctot = c(500, 140, 60),
         m = array(c(5, 1, 2, 7, 4, 2, 5, 9, 5, 4, 3, 1), dim = c(2, 2, 3),
                dimnames = list(orig = dn, dest = dn, type = c("0--15", "15-60", ">60"))),
         verbose = FALSE)
# display with row, col and table totals
y
```

---

ffs *Estimation of Bilateral Migrant Flows from Bilteral Migrant Stocks*

Description

Estimates migrant transitions flows between two sequential migrant stock tables.

Usage

```
ffs(P1, P2, d, b, m = NULL, method = "stocks", b.mat = NULL,
  d.mat = NULL, b.deduct = "native.gt0", ...)
```

Arguments

P1	Matrix of migrant stock totals at time *t*. Rows in the matrix correspond to place of birth and columns to place of residence at time *t*
P2	Matrix of migrant stock totals at time *t*+1. Rows in the matrix correspond to place of birth and columns to place of residence at time *t*+1.
d	Vector of the number of deaths between time *t* and *t*+1 in each region.
b	Vector of the number of births between time *t* and *t*+1 in each region.
m	Matrix of auxiliary data. By default set to 1 for all origin-destination combinations.
method	Method used to adjust row margin totals of P1 and P2 to equal. By default method="stocks". Can also take values method="outside" or method="deaths". See details for explanation on each method.

b.mat	Matrix containing the number of births during the period in each birthplace (rows) by place of residence (columns) combination created by the user. By default, this argument is NULL, and hence within the function a b.mat is formed as a diagonal matrix of b (i.e. all births happen in their respective place of residence at the end of the period, there are no infant migrants.)
d.mat	Matrix containing the number of deaths during the period in each birthplace (rows) by place of residence (columns) combination created by the user. By default, this argument is NULL, and hence within the function a d.mat is formed as a proportional allocation of d over all populations (i.e. the mortality rate in each place of birth sub-group (native born and all foreign born stocks) is the same.)
b.deduct	Method used to deduct births. By default b.deduct="native.gt0" deducts births from diagonals of stock table (i.e. the native born populations) with the exception of regions where this would lead to a negative adjusted population. In these select regions, births are spread over all population stocks (both native and foreign) thus avoiding potential negative flows. Can also take b.deduct="native.only" in which all births are deducted from the diagonals of stock table (i.e. native born populations only).
...	Additional arguments passes to ipf3_qi.

Value

Estimates migrant transitions flows between two sequential migrant stock tables as shown in Abel (2013), when method="outside". The length of b and d must equal the number of rows in P1 and number of columns in P2.

Setting method="stocks" estimates migration flows using an alternative demographic accounting method to adjusted stock tables to match the row totals of the stock table after demographic accounting. Setting method="deaths" uses the calculation of the deaths by place of residence table to match the row totals of the stock table after demographic accounting. Note, when b and d are equal, the same estimated flows from stocks are obtained regardless of the method argument. Both of these options maintain the net migration flow implied by the population, birth and death data. I still need to write up these methods.

Setting b.deduct="native.gt0" allows estimates to correct for cases where the number of births far exceeds the change in the native born population. Such cases potentially occur where place of birth stock data imply a change in native born population which directly conflict with changes in demographic data unless there is mass migration of all new borns.

Returns a list object with:

mu	Array of indirect estimates of origin-destination matrices by place of birth.
it	Iteration count.
tol	Tolerance level at final iteration.
y	Array of indirect estimates of origin-destination matrices by place of birth with additional rows and columns for births, deaths and moves to other regions.

Author(s)

Guy J. Abel

References

Abel, G. J. (2016). Estimates of Global Bilateral Migration Flows by Gender between 1960 and 2015. *Vienna Institute of Demography Working Papers* 2/2016.

Abel, G. J. and Sander, N. (2014). Quantifying Global International Migration Flows. *Science*, 343 (6178) 1520-1522

Abel, G. J. (2013). Estimating Global Migration Flow Tables Using Place of Birth. *Demographic Research* 28, (18) 505-546

See Also

ipf3_qi, od_sum, ffs_diff

Examples

```
## create P1 and P2 stock tables
dn <- LETTERS[1:4]
P1 <- matrix(data = c(1000, 100, 10, 0, 55, 555, 50, 5, 80, 40, 800, 40, 20, 25, 20, 200),
             nrow = 4, ncol = 4, dimnames = list(pob = dn, por = dn), byrow = TRUE)
P2 <- matrix(data = c(950, 100, 60, 0, 80, 505, 75, 5, 90, 30, 800, 40, 40, 45, 0, 180),
             nrow = 4, ncol = 4, dimnames = list(pob = dn, por = dn), byrow = TRUE)
# display with row and col totals
addmargins(A = P1)
addmargins(A = P2)

# no births and deaths
b <- rep(x = 0, 4)
d <- rep(x = 0, 4)

y <- ffs(P1 = P1, P2 = P2, d = d, b = b)
# display with row, col and table totals
round(x = addmargins(A = y$mu), digits = 1)
# display with row and col totals
round(x = od_sum(y = y$mu), digits = 1)

## alternative offset term
dis <- matrix(data = c(1, 2, 3, 4, 2, 1, 5, 6, 3, 4, 1, 7, 4, 6, 7, 1), nrow = 4, ncol = 4)
y <- ffs(P1 = P1, P2 = P2, d = d, b = b, m = dis)
# display with row, col and table totals
round(x = addmargins(A = y$mu), digits = 1)
# display with row and col totals
round(x = od_sum(y = y$mu), digits = 1)

## alternative P2 and changes in population from natural increase
P2 <- matrix(data = c(1060, 60, 10, 10, 45, 540, 40, 0, 70, 75, 770, 70, 30, 30, 20, 230),
             nrow = 4, ncol = 4, dimnames = list(pob = dn, por = dn), byrow = TRUE)
# display with row and col totals
addmargins(A = P2)
b <- c(80, 20, 40, 60)
d <- c(70, 30, 50, 10)

y <- ffs(P1 = P1, P2 = P2, d = d, b = b, method = "outside")
```

```
# display with row, col and table totals
round(x = addmargins(A = y$mu), digits = 1)
# display with row and col totals
round(x = od_sum(y = y$mu), digits = 1)
```

`ffs_diff` *Estimation of Bilateral Migrant Flows from Bilteral Migrant Stocks Using Stock Differencing*

Description

Estimates migrant transitions flows between two sequential migrant stock tables using differencing approaches communly used by economists.

Usage

```
ffs_diff(P1, P2, decrease = "return")
```

Arguments

`P1`	Matrix of migrant stock totals at time *t*. Rows in the matrix correspond to place of birth and columns to place of residence at time *t*
`P2`	Matrix of migrant stock totals at time *t*+1. Rows in the matrix correspond to place of birth and columns to place of residence at time *t*+1.
`decrease`	How to treat decreases in bilateral stocks over the *t* to *t*+1 period (so as to avoid a negative bilateral flow estiamtes). See details for possible options.

Value

Estimates migrant transitions flows between two sequential migrant stock tables.

When `decrease = "zero"` all decreases in migrant stocks over there period are set to zero, following the approach of Bertoli and Fernandez-Huertas Moraga (2015)

When `decrease = "return"` all decreases in migrant stocks are assumed to correspond to return flows back to their place of birth, following the approach of Beine and Parsons (2015)

Author(s)

Guy J. Abel

See Also

`ffs`

Examples

```
dn <- LETTERS[1:4]
P1 <- matrix(c(0, 100, 10, 0, 55, 0, 50, 5, 80, 40, 0, 40, 20, 25, 20, 0), 4, 4,
             dimnames = list(pob = dn, por = dn), byrow = TRUE)
P2 <- matrix(c(0, 100, 60, 0, 80, 0, 75, 5, 90, 30, 0, 40, 40, 45, 0, 0), 4, 4,
             dimnames = list(pob = dn, por = dn), byrow = TRUE)
P1; P2

ffs_diff(P1, P2, decrease = "zero")
ffs_diff(P1, P2, decrease = "return")
```

---

ipf2 *Iterative Proportional Fitting Routine for the Indirect Estimation of Origin-Destination Migration Flow Table with Known Margins.*

Description

The `ipf2` function finds the maximum likelihood estimates for fitted values in the log-linear model:

$$\log y_{ij} = \log \alpha_i + \log \beta_j + \log m_{ij}$$

where m_{ij} is a set of prior estimates for y_{ij} and itself is no more complex than the one being fitted.

Usage

```
ipf2(rtot = NULL, ctot = NULL, m = matrix(1, length(rtot), length(ctot)),
  tol = 1e-05, maxit = 500, verbose = FALSE)
```

Arguments

`rtot`	Vector of origin totals to constrain the sum of the imputed cell rows.
`ctot`	Vector of destination totals to constrain the sum of the imputed cell columns.
`m`	Matrix of auxiliary data. By default set to 1 for all origin-destination combinations.
`tol`	Numeric value for the tolerance level used in the parameter estimation.
`maxit`	Numeric value for the maximum number of iterations used in the parameter estimation.
`verbose`	Logical value to indicate the print the parameter estimates at each iteration. By default `FALSE`.

Value

Iterative Proportional Fitting routine set up in a similar manner to Agresti (2002, p.343). This is equivalent to a conditional maximization of the likelihood, as discussed by Willekens (1999), and hence provides identical indirect estimates to those obtained from the `cm2` routine.

The user must ensure that the row and column totals are equal in sum. Care must also be taken to allow the dimension of the auxiliary matrix (`m`) to equal those provided in the row and column totals.

If only one of the margins is known, the function can still be run. The indirect estimates will correspond to the log-linear model without the α_i term if (`rtot = NULL`) or without the β_j term if (`ctot = NULL`)

Returns a `list` object with

`mu`	Origin-Destination matrix of indirect estimates
`it`	Iteration count
`tol`	Tolerance level at final iteration

Author(s)

Guy J. Abel

References

Agresti, A. (2002). *Categorical Data Analysis* 2nd edition. Wiley.

Willekens, F. (1999). Modelling Approaches to the Indirect Estimation of Migration Flows: From Entropy to EM. *Mathematical Population Studies* 7 (3), 239–78.

See Also

`cm2`, `ipf3`

Examples

```
## with Willekens (1999) data
dn <- LETTERS[1:2]
y <- ipf2(rtot = c(18, 20), ctot = c(16, 22),
          m = matrix(c(5, 1, 2, 7), ncol = 2,
                     dimnames = list(orig = dn, dest = dn)))
round(addmargins(y$mu),2)

## with all elements of offset equal
y <- ipf2(rtot = c(18, 20), ctot = c(16, 22))
round(addmargins(y$mu),2)

## with bigger matrix
dn <- LETTERS[1:3]
y <- ipf2(rtot = c(170, 120, 410), ctot = c(500, 140, 60),
          m = matrix(c(50, 10, 220, 120, 120, 30, 545, 0, 10), ncol = 3,
                     dimnames = list(orig = dn, dest = dn)))
# display with row and col totals
round(addmargins(y$mu))

## only one margin known
dn <- LETTERS[1:2]
y <- ipf2(rtot = c(18, 20), ctot = NULL,
```

```
          m = matrix(c(5, 1, 2, 7), ncol = 2,
                     dimnames = list(orig = dn, dest = dn)))
round(addmargins(y$mu))
```

`ipf2_block` *Iterative Proportional Fitting Routine for the Indirect Estimation of Origin-Destination-Type Migration Flow Tables with Known Origin and Destination Margins and Block Diagonal Elements.*

Description

The `ipf2.b` function finds the maximum likelihood estimates for fitted values in the log-linear model:

$$\log y_{pq} = \log \alpha_p + \log \beta_q + \log \lambda_{ij} I(p \in i, q \in j) + \log m_{pq}$$

where m_{pq} is a prior estimate for y_{pq} and is no more complex than the matrices being fitted. The $\lambda_{ij} I(p \in i, q \in j)$ term ensures a saturated fit on the block the (i, j) block.

Usage

```
ipf2_block(rtot = NULL, ctot = NULL, btot = NULL, block = NULL,
  m = NULL, tol = 1e-05, maxit = 500, verbose = TRUE, ...)
```

Arguments

`rtot`	Vector of origin totals to constrain the sum of the imputed cell rows.
`ctot`	Vector of destination totals to constrain the sum of the imputed cell columns.
`btot`	Matrix of block totals to constrain the sum of the imputed cell blocks.
`block`	Matrix of block structure corresponding to `btot`.
`m`	Matrix of auxiliary data. By default set to 1 for all origin-destination combinations.
`tol`	Numeric value for the tolerance level used in the parameter estimation.
`maxit`	Numeric value for the maximum number of iterations used in the parameter estimation.
`verbose`	Logical value to indicate the print the parameter estimates at each iteration. By default `FALSE`.
`...`	Additional arguments passes to `block_matrix`.

Value

Iterative Proportional Fitting routine set up using the partial likelihood derivatives. The arguments `rtot` and `ctot` take the row-table and column-table specific known margins. The `btot` take the totals over the blocks in the matrix defined with `b`. Diagonal values can be added by the user, but care must be taken to ensure resulting diagonals are feasible given the set of margins.

The user must ensure that the row and column totals in each table sum to the same value. Care must also be taken to allow the dimension of the auxiliary matrix (`m`) equal those provided in the row and column totals.

Returns a `list` object with

`mu`	Array of indirect estimates of origin-destination matrices by migrant characteristic
`it`	Iteration count
`tol`	Tolerance level at final iteration

Author(s)

Guy J. Abel

See Also

`block_matrix`, `stripe_matrix`, `block_sum`

Examples

```
y <- ipf2_block(rtot= c(30,20,30,10,20,5,0,10,5,5,5,10),
                ctot = c(45,10,10,5,5,10,50,5,10,0,0,0),
                btot = matrix(data = c(0,0 ,50,0, 35,0,25,0, 10,10,0,0, 10,10,0,0),
                              nrow = 4, byrow = TRUE),
                block = block_matrix(x = 1:16, b = c(2,3,4,3)))
addmargins(y$mu)
```

---

ipf2_stripe *Iterative Proportional Fitting Routine for the Indirect Estimation of Origin-Destination-Type Migration Flow Tables with Known Origin and Destination Margins and Stripe Elements.*

Description

The `ipf2.b` function finds the maximum likelihood estimates for fitted values in the log-linear model:

$$\log y_{pq} = \log \alpha_p + \log \beta_q + \log \lambda_{ij} I(p \in i, q \in j) + \log m_{pq}$$

where m_{pq} is a prior estimate for y_{pq} and is no more complex than the matrices being fitted. The $\lambda_{ij} I(p \in i, q \in j)$ term ensures a saturated fit on the block the (i, j) block.

Usage

```
ipf2_stripe(rtot = NULL, ctot = NULL, stot = NULL, stripe = NULL,
  m = NULL, tol = 1e-05, maxit = 500, verbose = TRUE, ...)
```

Arguments

rtot	Vector of origin totals to constrain the sum of the imputed cell rows.
ctot	Vector of destination totals to constrain the sum of the imputed cell columns.
stot	Matrix of stripe totals to constrain the sum of the imputed cell blocks.
stripe	Matrix of stripe stucture corresponding to stot.
m	Matrix of auxiliary data. By default set to 1 for all origin-destination combinations.
tol	Numeric value for the tolerance level used in the parameter estimation.
maxit	Numeric value for the maximum number of iterations used in the parameter estimation.
verbose	Logical value to indicate the print the parameter estimates at each iteration. By default FALSE.
...	Additional arguments passes to stripe_matrix.

Value

Iterative Proportional Fitting routine set up using the partial likelihood derivatives. The arguments rtot and ctot take the row-table and column-table specific known margins. The stot take the totals over the stripes in the matrix defined with b. Diagonal values can be added by the user, but care must be taken to ensure resulting diagonals are feasible given the set of margins. The user must ensure that the row and column totals in each table sum to the same value. Care must also be taken to allow the dimension of the auxiliary matrix (m) equal those provided in the row and column totals. Returns a list object with

mu	Array of indirect estimates of origin-destination matrices by migrant characteristic
it	Iteration count
tol	Tolerance level at final iteration

Author(s)

Guy J. Abel

See Also

stripe_matrix, block_matrix, block_sum

Examples

```
y <- ipf2_stripe(rtot = c(85, 70, 35, 30, 60, 55, 65),
 stot = matrix(c(15,20,50,
                 35,10,25,
                 5 ,0 ,30,
                 10,10,10,
                 30,30,0,
                 15,30,10,
                 35,25,5 ), ncol = 3, byrow = TRUE),
```

```
stripe = stripe_matrix(x = 1:21, s = c(2,2,3), byrow = TRUE))
addmargins(y$mu)
```

`ipf3` *Iterative Proportional Fitting Routine for the Indirect Estimation of Origin-Destination-Migrant Type Migration Flow Tables with Known Origin and Destination Margins.*

Description

The `ipf3` function finds the maximum likelihood estimates for fitted values in the log-linear model:

$$\log y_{ijk} = \log \alpha_i + \log \beta_j + \log \lambda_k + \log \gamma_{ik} + \log \kappa_{jk} + \log m_{ijk}$$

where m_{ijk} is a set of prior estimates for y_{ijk} and is no more complex than the matrices being fitted.

Usage

```
ipf3(rtot = NULL, ctot = NULL, m = NULL, tol = 1e-05, maxit = 500,
  verbose = TRUE)
```

Arguments

`rtot`	Vector of origin totals to constrain the sum of the imputed cell rows.
`ctot`	Vector of destination totals to constrain the sum of the imputed cell columns.
`m`	Array of auxiliary data. By default set to 1 for all origin-destination-migrant typologies combinations.
`tol`	Numeric value for the tolerance level used in the parameter estimation.
`maxit`	Numeric value for the maximum number of iterations used in the parameter estimation.
`verbose`	Logical value to indicate the print the parameter estimates at each iteration. By default `FALSE`.

Value

Iterative Proportional Fitting routine set up in a similar manner to Agresti (2002, p.343). The arguments `rtot` and `ctot` take the row-table and column-table specific known margins.

The user must ensure that the row and column totals in each table sum to the same value. Care must also be taken to allow the dimension of the auxiliary matrix (`m`) to equal those provided in the row and column totals.

Returns a `list` object with

`mu`	Array of indirect estimates of origin-destination matrices by migrant characteristic
`it`	Iteration count
`tol`	Tolerance level at final iteration

Author(s)

Guy J. Abel

References

Abel, G. J. (2013). Estimating Global Migration Flow Tables Using Place of Birth. *Demographic Research* 28, (18) 505-546

Agresti, A. (2002). *Categorical Data Analysis* 2nd edition. Wiley.

See Also

ipf3_qi, ipf2

Examples

```
## create row-table and column-table specific known margins.
dn <- LETTERS[1:4]
P1 <- matrix(c(1000, 100,  10,   0,
               55,   555,  50,   5,
               80,    40, 800 , 40,
               20,    25,  20, 200),
             nrow = 4, ncol = 4, byrow = TRUE,
             dimnames = list(pob = dn, por = dn))
P2 <- matrix(c(950, 100,  60,   0,
                80, 505,  75,   5,
                90,  30, 800,  40,
                40,  45,   0, 180),
             nrow = 4, ncol = 4, byrow = TRUE,
             dimnames = list(pob = dn, por = dn))
# display with row and col totals
addmargins(P1)
addmargins(P2)

# run ipf
y <- ipf3(rtot = t(P1), ctot = P2)
# display with row, col and table totals
round(addmargins(y$mu), 1)
# origin-destination flow table
round(od_sum(y$mu), 1)

## with alternative offset term
dis <- array(c(1, 2, 3, 4, 2, 1, 5, 6, 3, 4, 1, 7, 4, 6, 7, 1), c(4, 4, 4))
y <- ipf3(rtot = t(P1), ctot = P2, m = dis)
# display with row, col and table totals
round(addmargins(y$mu), 1)
# origin-destination flow table
round(od_sum(y$mu), 1)
```

ipf3_diag *Iterative Proportional Fitting Routine for the Indirect Estimation of Origin-Destination-Migrant Type Migration Flow Tables with Known Origin and Destination Margins and Diagonal Elements.*

Description

This function is predominantly intended to be used within the `ffs` routine.

Usage

```
ipf3_diag(rtot = NULL, ctot = NULL, dtot = NULL, m = NULL,
  speed = TRUE, tol = 1e-05, maxit = 500, verbose = TRUE)
```

Arguments

`rtot`	Vector of origin totals to constrain the sum of the imputed cell rows.
`ctot`	Vector of destination totals to constrain the sum of the imputed cell columns.
`dtot`	Array with counts on diagonal to constrain diagonal elements of the indirect estimates too. By default these are taken as their maximum possible values given the relevant margins totals in each table. If user specifies their own array of diagonal totals, values on the non-diagonals in the array can take any positive number (they are ultimately ignored).
`m`	Array of auxiliary data. By default set to 1 for all origin-destination-migrant typologies combinations.
`speed`	Speeds up the IPF algorithm by minimizing sufficient statistics.
`tol`	Numeric value for the tolerance level used in the parameter estimation.
`maxit`	Numeric value for the maximum number of iterations used in the parameter estimation.
`verbose`	Logical value to indicate the print the parameter estimates at each iteration. By default `FALSE`.

Details

The `ipf3` function finds the maximum likelihood estimates for fitted values in the log-linear model:

$$\log y_{ijk} = \log \alpha_i + \log \beta_j + \log \lambda_k + \log \gamma_{ik} + \log \kappa_{jk} + \log \delta_{ijk} I(i = j) + \log m_{ijk}$$

where m_{ijk} is a set of prior estimates for y_{ijk} and is no more complex than the matrices being fitted. The $\delta_{ijk} I(i = j)$ term ensures a saturated fit on the diagonal elements of each (i, j) matrix.

Value

Iterative Proportional Fitting routine set up using the partial likelihood derivatives illustrated in Abel (2013). The arguments `rtot` and `ctot` take the row-table and column-table specific known margins. By default the diagonal values are taken as their maximum possible values given the relevant margins totals in each table. Diagonal values can be added by the user, but care must be taken to ensure resulting diagonals are feasible given the set of margins.

The user must ensure that the row and column totals in each table sum to the same value. Care must also be taken to allow the dimension of the auxiliary matrix (`m`) equal those provided in the row and column totals.

Returns a `list` object with

`mu`	Array of indirect estimates of origin-destination matrices by migrant characteristic
`it`	Iteration count
`tol`	Tolerance level at final iteration

Author(s)

Guy J. Abel

References

Abel, G. J. (2013). Estimating Global Migration Flow Tables Using Place of Birth. *Demographic Research* 28, (18) 505-546

See Also

`ipf3`, `ffs`, `od_sum`

Examples

```
## create row-table and column-table specific known margins.
dn <- LETTERS[1:4]
P1 <- matrix(c(1000, 100,  10,   0,
               55,   555,  50,   5,
               80,    40, 800 , 40,
               20,    25,  20, 200),
             nrow = 4, ncol = 4, byrow = TRUE,
             dimnames = list(pob = dn, por = dn))
P2 <- matrix(c(950, 100,  60,   0,
                80, 505,  75,   5,
                90,  30, 800,  40,
                40,  45,   0, 180),
             nrow = 4, ncol = 4, byrow = TRUE,
             dimnames = list(pob = dn, por = dn))
# display with row and col totals
addmargins(P1)
addmargins(P2)
```

```
# run ipf

## with alternative offset term
dis <- array(c(1, 2, 3, 4, 2, 1, 5, 6, 3, 4, 1, 7, 4, 6, 7, 1), c(4, 4, 4))
y <- ipf3_diag(rtot = t(P1), ctot = P2, m = dis)
# display with row, col and table totals
round(addmargins(y$mu), 1)
# origin-destination flow table
round(od_sum(y$mu), 1)
```

---

ipf3_qi *Iterative Proportional Fitting Routine for the Indirect Estimation of Origin-Destination-Migrant Type Migration Flow Tables with Known Origin and Destination Margins and Diagonal Elements.*

Description

This function is predominantly intended to be used within the `ffs` routine.

Usage

```
ipf3_qi(rtot = NULL, ctot = NULL, dtot = NULL, m = NULL, speed = TRUE,
  tol = 1e-05, maxit = 500, verbose = TRUE)
```

Arguments

`rtot`	Vector of origin totals to constrain the sum of the imputed cell rows.
`ctot`	Vector of destination totals to constrain the sum of the imputed cell columns.
`dtot`	Array with counts on diagonal to constrain diagonal elements of the indirect estimates too. By default these are taken as their maximum possible values given the relevant margins totals in each table. If user specifies their own array of diagonal totals, values on the non-diagonals in the array can take any positive number (they are ultimately ignored).
`m`	Array of auxiliary data. By default set to 1 for all origin-destination-migrant typologies combinations.
`speed`	Speeds up the IPF algorithm by minimizing sufficient statistics.
`tol`	Numeric value for the tolerance level used in the parameter estimation.
`maxit`	Numeric value for the maximum number of iterations used in the parameter estimation.
`verbose`	Logical value to indicate the print the parameter estimates at each iteration. By default `FALSE`.

Details

The `ipf3` function finds the maximum likelihood estimates for fitted values in the log-linear model:

$$\log y_{ijk} = \log \alpha_i + \log \beta_j + \log \lambda_k + \log \gamma_{ik} + \log \kappa_{jk} + \log \delta_{ijk} I(i = j) + \log m_{ijk}$$

where m_{ijk} is a set of prior estimates for y_{ijk} and is no more complex than the matrices being fitted. The $\delta_{ijk} I(i = j)$ term ensures a saturated fit on the diagonal elements of each (i, j) matrix.

Value

Iterative Proportional Fitting routine set up using the partial likelihood derivatives illustrated in Abel (2013). The arguments `rtot` and `ctot` take the row-table and column-table specific known margins. By default the diagonal values are taken as their maximum possible values given the relevant margins totals in each table. Diagonal values can be added by the user, but care must be taken to ensure resulting diagonals are feasible given the set of margins.

The user must ensure that the row and column totals in each table sum to the same value. Care must also be taken to allow the dimension of the auxiliary matrix (`m`) equal those provided in the row and column totals.

Returns a `list` object with

`mu`	Array of indirect estimates of origin-destination matrices by migrant characteristic
`it`	Iteration count
`tol`	Tolerance level at final iteration

Author(s)

Guy J. Abel

References

Abel, G. J. (2013). Estimating Global Migration Flow Tables Using Place of Birth. *Demographic Research* 28, (18) 505-546

See Also

`ipf3`, `ffs`, `od_sum`

Examples

```
## create row-table and column-table specific known margins.
dn <- LETTERS[1:4]
P1 <- matrix(c(1000, 100,  10,   0,
               55,   555,  50,   5,
               80,    40, 800 , 40,
               20,    25,  20, 200),
             nrow = 4, ncol = 4, byrow = TRUE,
             dimnames = list(pob = dn, por = dn))
P2 <- matrix(c(950, 100,  60,   0,
```

```
                80, 505,  75,   5,
                90,  30, 800,  40,
                40,  45,   0, 180),
              nrow = 4, ncol = 4, byrow = TRUE,
              dimnames = list(pob = dn, por = dn))
# display with row and col totals
addmargins(P1)
addmargins(P2)

# run ipf

## with alternative offset term
dis <- array(c(1, 2, 3, 4, 2, 1, 5, 6, 3, 4, 1, 7, 4, 6, 7, 1), c(4, 4, 4))
y <- ipf3_qi(rtot = t(P1), ctot = P2, m = dis)
# display with row, col and table totals
round(addmargins(y$mu), 1)
# origin-destination flow table
round(od_sum(y$mu), 1)
```

`ipf_net` *Iterative Proportional Fitting Routine for the Indirect Estimation of Origin-Destination-Type Migration Flow Tables with Known Net Migration Totals.*

Description

The `ipf_net` function finds the maximum likelihood estimates for fitted values in the log-linear model:

???

Usage

```
ipf_net(ntot = NULL, m = NULL, tol = 1e-05, maxit = 500,
  verbose = TRUE)
```

Arguments

`ntot`	Vector of net migraiton totals to constrain the sum of the imputed cell columns. Elements must sum to zero.
`m`	Array of auxiliary data. By default set to 1 for all origin-destination-migrant typologies combinations.
`tol`	Numeric value for the tolerance level used in the parameter estimation.
`maxit`	Numeric value for the maximum number of iterations used in the parameter estimation.
`verbose`	Logical value to indicate the print the parameter estimates at each iteration. By default `FALSE`.

Value

Iterative Proportional Fitting routine set up using the partial likelihood derivatives. The argument `ntot` takes the known net migration totals. The user must ensure that the net migraiton totals sum to globally to zero.

Returns a `list` object with

`mu`	Array of indirect estimates of origin-destination matrices by migrant characteristic
`it`	Iteration count
`tol`	Tolerance level at final iteration

Author(s)

Guy J. Abel

Examples

```
m <- matrix(data = 1:16, nrow = 4)
y <- ipf_net(ntot = c(30, 40, -15, -55), m = m)
addmargins(m)
net_sum(y$mu)
```

---

net_param *Estimate Parameters for Net Migration Scaling.*

Description

This function is predominantly intended to be used within the `ipf_net` routine.

Usage

```
net_param(m, region, ntot)
```

Arguments

`m`	Matrix of origin-destination flows, where the first and second dimensions correspond to origin and destination respectively.
`region`	Integer value corresponding to the region that the net migration sum is desired. Will return sums for all regions by default.
`ntot`	Vector of net migraiton totals to constrain the sum of the imputed cell columns. Elements must sum to zero.

Value

Vector of two values corresonding to the roots fo the quatratic equation.

Author(s)

Guy J. Abel

`net_scale` *Scale Migration Flows in Origin-Destination*

Description

This function is predominantly intended to be used within the `ipf_net` routine.

Usage

```
net_scale(m, region = NULL, alpha)
```

Arguments

`m`	Matrix of origin-destination flows, where the first and second dimensions correspond to origin and destination respectively.
`region`	Integer corresponding to row (column) in a square matrix for the region where scaling is to be applied
`alpha`	Numeric value of the scaling factor

Value

Matrix scaled in region(s) by value of alpha, where `alpha` applied to destination flows and inverse of `alpha` applied to origin flows

Author(s)

Guy J. Abel

`net_sum` *Extract Net Migration from an Origin-Destination Migration Flow Matrix.*

Description

Sums each regions flows (from origin rows and destination columns) to obtain net migration sums.

Usage

```
net_sum(m, region = 1:dim(m)[1])
```

Arguments

m	Matrix of origin-destination flows, where the first and second dimensions correspond to origin and destination respectively.
region	Integer value corresponding to the region that the net migration sum is desired. Will return sums for all regions by default.

Value

Returns a numeric value of the sum of a single block.

Author(s)

Guy J. Abel

See Also

block_sum, od_sum

Examples

```
m <- matrix(data = 1:16, nrow = 4, ncol = 4)
net_sum(m)
```

---

od_sum *Extract a Classic Origin-Destination Migration Flow Matrix.*

Description

Extract a classic origin-destination migration flow matrix from a more detailed dis-aggregation of flows stored in an (array) object.

Usage

```
od_sum(y)
```

Arguments

y	Array of origin-destination matrices, where the first and second dimensions correspond to origin and destination respectively. Higher dimension(s) refer to additional migrant characteristic(s).

Value

Sums over the first and second dimension and removes counts on diagonals.

Returns a matrix object of origin-destination flows

Author(s)

Guy J. Abel

Examples

```
dn <- LETTERS[1:2]
y <- cm3(rtot = c(18, 20) * 2, ctot = c(16, 22) * 2,
         m = array(c(5, 1, 2, 7, 4, 2, 5, 9), dim = c(2, 2, 2),
                   dimnames = list(orig = dn, dest = dn, type = c("ILL", "HEALTHY"))))
round(addmargins(y$N))
round(addmargins(od_sum(y$N)))
```

---

quadratic_eqn *Solve Quadratic Equation*

Description

General function to solve classic quadratic equation:

$$ax^2 + bx + c = 0$$

Usage

```
quadratic_eqn(a, b, c)
```

Arguments

a	Numeric value for quadratic term of x.
b	Numeric value for multiplicative term of x.
c	Numeric value for constant term.

Value

Vector of two values corresonding to the roots fo the quatratic equation.

Author(s)

Guy J. Abel

Source

Adapted from https://rpubs.com/kikihatzistavrou/80124

Examples

```
quadratic_eqn(a = 2, b = 4, c = -6)
```

rc9 *Rogers-Castro Migration Schedule*

Description

Provides the Rogers-Castro schedule,

$$M(x) = a_1 \exp[-\alpha_1 x] + a_2 \exp[\alpha_2(x - \mu_2) - \exp[\lambda_2(x - \mu_2)]] + c$$

for a given set of parameters and ages.

Usage

```
rc9(x, param = NULL, scaled = TRUE)
```

Arguments

x	Vector of numbers
param	List with names matching the parameters in the age schedule
scaled	Scale estimates to sum to one across all ages, x.

Value

Returns the M(x) values from the Rogers-Castro schedule of age specific migration rate. The age range for the calculation can take any sequence of positive numbers, such as ages in single or 5-year intervals. The `param` argument must be a list with correct names for each parameter. See for example the `rc9.fund` object for an example of the naming convention.

Author(s)

Guy J. Abel

References

Rogers, A., and L. J. Castro. (1981). Model Migration Schedules. *IIASA Research Report 81* RR-81-30

See Also

`rc9.fund`

Examples

```
# single age groups
x <- 1:100
m <- rc9(x, param = rc9.fund)
plot(x, m, type="l")

# 5 year age groups
m <- rc9(x, param = rc9.fund)
plot(x, m, type="l")
```

`rc9.fund` *Fundamental Parameters for Rogers-Castro Migration Schedule*

Description

Set of fundamental parameters for the Rogers-Castro migration age schedule, as suggested in Rogers and Castro (1981).

Usage

```
rc9.fund
```

Format

A `list` of the parameters required by the `rc9` function:

$$a_1 = 0.02$$

$$\alpha_1 = 0.1$$

$$a_2 = 0.06$$

$$\alpha_2 = 0.1$$

$$\mu_2 = 20$$

$$\lambda_2 = 0.4$$

$$c = 0.003$$

Source

Rogers, A., and L. J. Castro. (1981). Model Migration Schedules. *IIASA Research Report 81* RR-81-30

Examples

```
# check format
str(rc9.fund)

# single age groups
x <- 1:100
m <- rc9(x, param = rc9.fund)
plot(x, m, type="l")

# alter to see the effect of mu2
p1 <- rc9.fund
p1$mu2 <- 30
m1 <- rc9(x, param = p1)
plot(x, m, type="l")
lines(x, m1, lty=2)
```

`stripe_matrix` *Create a Stripped Matrix with Non-Uniform Block Sizes.*

Description

Create a Stripped Matrix with Non-Uniform Block Sizes.

Usage

```
stripe_matrix(x = NULL, s = NULL, byrow = FALSE, dimnames = NULL)
```

Arguments

`x`	Vector of numbers to identify each stripe.
`s`	Vector of values for the size of the stripes, order depending on `byrow`
`byrow`	Logical value. If `FALSE` (the default) the stripes are filled by columns, otherwise the stripes in the matrix are filled by rows.
`dimnames`	Character string of name attribute for the basis of the stripped matrix. If `NULL` a vector of the same length of `s` provides the basis of row and column names.

Value

Returns a `matrix` with stripe sizes determined by the `s` argument. Each stripe is filled with the same value taken from `x`.

Author(s)

Guy J. Abel

See Also

`block_matrix`, `block_sum`, `ipf2_stripe`

Examples

```
stripe_matrix(x = 1:44, s = c(2,3,4,2), dimnames = LETTERS[1:4], byrow = TRUE)
```

Index