Package ‘migest’

October 28, 2022

Type Package
Title Methods for the Indirect Estimation of Bilateral Migration
Version 2.0.3
Maintainer Guy J. Abel <g.j.abel@gmail.com>
Description Tools for estimating, measuring and working with migration data.
URL http://guyabel.github.io/migest/
BugReports https://github.com/guyabel/migest/issues
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.2.1
Imports dplyr, purrr, tidyr, stringr, magrittr, stats, tibble, forcats, utils, matrixStats, migration.indices, circlize, graphics, grDevices, mipfp
Depends R (>= 2.10)
Suggests spelling, tidyverse, countrycode
Language en-US
NeedsCompilation no
Author Guy J. Abel [aut, cre] (<https://orcid.org/0000-0002-4893-5687>)
Repository CRAN
Date/Publication 2022-10-28 11:05:05 UTC

R topics documented:

migest-package ........................................... 3
alabama_1970 ........................................... 4
birth_mat ................................................. 5
block_matrix ............................................ 5
block_sum .............................................. 6
R topics documented:

bombay_1951 ......................................................... 7
cm2 ................................................................. 8
cm3 ................................................................. 10
cm_net ............................................................. 12
cm_net_tot ......................................................... 13
death_mat ......................................................... 15
dict_ims ............................................................ 16
ffs_demo ............................................................ 17
ffs_diff ............................................................. 20
ffs_rates ........................................................... 22
index_age .......................................................... 23
index_age_rc ....................................................... 25
index_connectivity ............................................... 26
index_distance .................................................... 28
index_impact ....................................................... 29
index_intensity ................................................... 31
indian_sub ........................................................ 33
ipf2 ................................................................. 34
ipf2_block ........................................................ 36
ipf2_stripe ........................................................ 37
ipf3 ................................................................. 39
ipf3_qi ............................................................. 41
ipf_seed ............................................................ 43
ipumsi_age ........................................................ 44
italy_area ......................................................... 45
korea_dist ........................................................ 46
korea_pop ........................................................ 46
korea_reg ........................................................ 47
manila_1970 ....................................................... 47
match_pob_tot ................................................... 48
mig_chord ......................................................... 49
mig_matrix ......................................................... 52
mig_tibble ........................................................ 53
multi_comp ......................................................... 54
multi_comp2 ....................................................... 55
nchars_wrap ....................................................... 56
net_sr .............................................................. 56
net_vs ............................................................ 58
new_england_1960 ............................................... 59
quadratic_eqn ..................................................... 60
rc_model_fund ..................................................... 61
rc_model_un ....................................................... 61
rescale_integer_sum ............................................. 62
rescale_nb ........................................................ 63
rescale_net ....................................................... 64
stripe_matrix ..................................................... 65
str_wrap_n ......................................................... 66
str_wrap_n_single ............................................... 67
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). Each of these routines are based on indirect estimation methods where marginal totals are known, and a Poisson regression (log-linear) model is assumed.

The ffs_diff, ffs_rates and ffs_demo functions provide different methods to estimate migration bilateral flows from changes in stocks, see Abel and Cohen (2019) for a review of different methods. The demo files, demo(cfplot_reg2), demo(cfplot_reg) and demo(cfplot_nat), produce circular migration flow plots for migration estimates from Abel(2018) and Abel and Sander (2014), which were derived using the ffs_demo function.

Github repo: https://github.com/guyabel/migest

Author(s)

Guy J. Abel
References


---

alabama_1970

*Alabama population totals in 1960 and 1970 by age, sex and race*

---

Description

Population data for Alabama by age, sex and race in 1960 and 1970

Usage

alabama_1970

Format

Data frame with 68 rows and 6 columns:

- **age_1970** Age group in 1970
- **sex** Sex from ‘male’ or ‘female’
- **race** Race from ‘white’ or ‘non-white’
- **pop_1960** Enumerated population in 1960. Number of births in first and second half of 1960s used for age groups ‘0-4’ and ‘5-9’.
- **pop_1970** Enumerated population in 1970
- **us_census_sr** Census survival ratio based on US population

Source

Data scraped from Figure 2.3 and Table 1-3A of Bogue, D. J., Hinze, K., & White, M. (1982). Techniques of Estimating Net Migration. Community and Family Study Center. University of Chicago.
birth_mat

*Calculate births for each element of place of birth - place of residence stock matrix*

**Description**

This function is predominantly intended to be used within the ffs routines in the migest package.

**Usage**

```
birth_mat(b_por = NULL, m2 = NULL, method = "native", non_negative = TRUE)
```

**Arguments**

- `b_por`: Vector of numeric values for births in each place of residence
- `m2`: 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 $t+1$.
- `method`: Character string of either "native" or "proportion" to choose method to distribute births. The "proportion" method assumes the rate of non-migration increase in each place of birth sub-group (native born and all foreign born stocks) is the same. The "native" method ensures that all births (non-migration increases) in stocks belong to the native born population (they do not move straight after birth).
- `non_negative`: Adjust birth matrix calculation to ensure all deductions from `m2` will result in positive population counts. On rare occasions when working with international stock data the number of births can exceed the increase in the number of native born population.

**Value**

Matrix of place of birth by place of residence for new-born’s

---

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 block 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`

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 over a selected block in a block matrix*

Description

Returns 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
```r
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)
```

bombay_1951

**Bombay population totals in 1941 and 1951 by age**

Description
Population data for Bombay by age in 1941 and 1951

Usage
```r
bombay_1951
```

Format
Data frame with 13 rows and 5 columns:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>age_1941</td>
<td>Age group in 1941</td>
</tr>
<tr>
<td>age_1951</td>
<td>Age group in 1951</td>
</tr>
<tr>
<td>pop_1941</td>
<td>Enumerated population in 1941</td>
</tr>
<tr>
<td>pop_1951</td>
<td>Enumerated population in 1951</td>
</tr>
<tr>
<td>sr</td>
<td>Census survival ratio derived from the United Nations model life table corresponding to a life expectancy at birth of 45 years for males. See Manual III: Methods for Population Projections by Sex and Age (United Nations publication, Sales No.: 56.XIII.3).</td>
</tr>
</tbody>
</table>

Source
**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 \( \alpha_i \) and \( \beta_j \) 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**

```r
cm2(row_tot = NULL, col_tot = NULL, m = matrix(data = 1, nrow = length(row_tot), ncol = length(col_tot)), tol = 1e-06, maxit = 500, verbose = TRUE, rtot = row_tot, ctot = col_tot)
```

**Arguments**

- `row_tot` Vector of origin totals to constrain the sum of the imputed cell rows.
- `col_tot` 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`.
- `rtot` Deprecated. Use `row_tot`
- `ctot` Deprecated. Use `col_tot`
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 (row_tot) and column (col_tot) arguments.

Returns a list object with

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

Author(s)

Guy J. Abel

References


See Also

ipf2

Examples

```r
## with Willekens (1999) data
r <- LETTERS[1:2]
y <- cm2(row_tot = c(18, 20), col_tot = c(16, 22),
        m = matrix(c(5, 1, 2, 7), ncol = 2, dimnames = list(orig = r, dest = r)))
y
## with all elements of offset equal (independence fit)
y <- cm2(row_tot = c(18, 20), col_tot = c(16, 22))
y
## with bigger matrix
r <- LETTERS[1:4]
y <- cm2(row_tot = c(250, 100, 140, 110), col_tot = c(150, 150, 180, 120),
        m = matrix(data = c(0, 100, 30, 70, 50, 0, 45, 40, 35, 0, 40, 20, 25, 20, 0),
                   nrow = 4, ncol = 4, dimnames = list(orig = r, dest = r), byrow = TRUE))

# display with row and col totals
round(addmargins(y$n))
```
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 \( \alpha_i \) and \( \beta_j \) 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

```r
cm3(
  row_tot = NULL,
  col_tot = NULL,
  m = NULL,
  tol = 1e-06,
  maxit = 500,
  verbose = TRUE
)
```

Arguments

- `row_tot`: Vector of origin totals to constrain the sum of the imputed cell rows.
- `col_tot`: 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**


**See Also**

cm2, ipf3

**Examples**

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

## over three tables
y <- cm3(row_tot = c(170, 120, 410), col_tot = 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 = r, dest = r, type = c("0--15", "15-60", ">60"))));
# display with row, col and table totals
y
```
cm_net

Conditional maximization routine for the indirect estimation of origin-destination-type migration flow tables with known net migration totals.

Description

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

\[ \log y_{ij} = \log \alpha_i + \log \alpha_i^{-1} + \log m_{ij} \]

Usage

```r
cm_net(
  net_tot = NULL,
  m = NULL,
  tol = 1e-06,
  maxit = 500,
  verbose = TRUE,
  alpha0 = rep(1, length(net_tot))
)
```

Arguments

- `net_tot` Vector of net migration totals to constrain the sum of the imputed cell row and 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.
- `alpha0` Vector of initial estimates for alpha

Value

Conditional maximisation routine set up using the partial likelihood derivatives. The argument `net_tot` takes the known net migration totals. The user must ensure that the net migration totals sum 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, Peter W. F. Smith

**Examples**

```r
m <- matrix(data = 1:16, nrow = 4)
# m[lower.tri(m)] <- t(m)[lower.tri(m)]
addmargins(m)
sum_net(m)

y <- cm_net(net_tot = c(30, 40, -15, -55), m = m)
addmargins(y$n)
sum_net(y$n)

m <- matrix(data = c(0, 100, 30, 70, 50, 0, 45, 5, 60, 35, 0, 40, 20, 25, 20, 0),
nrow = 4, ncol = 4, byrow = TRUE,
dimnames = list(orig = LETTERS[1:4], dest = LETTERS[1:4]))
addmargins(m)
sum_net(m)

y <- cm_net(net_tot = c(-100, 125, -75, 50), m = m)
addmargins(y$n)
sum_net(y$n)
```

---

**cm_net_tot**

Conditional maximization routine for the indirect estimation of origin-destination-type migration flow tables with known net migration and grand totals.

**Description**

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

\[
\log y_{ij} = \log \alpha_i + \log \alpha_i^{-1} + \log m_{ij}
\]

**Usage**

```r
cm_net_tot(
  net_tot = NULL,
  tot = NULL,
  m = NULL,
  tol = 1e-06,
  maxit = 500,
  verbose = TRUE,
  alpha0 = rep(1, length(net_tot)),
  lambda0 = 1,
  alpha_constrained = TRUE
)
```
Arguments

net_tot  Vector of net migration totals to constrain the sum of the imputed cell row and columns. Elements must sum to zero.
tot    Numeric value of grand total to constrain sum of all imputed cells.
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.
alpha0    Vector of initial estimates for alpha
lambda0    Numeric value of initial estimates for lambda
alpha_constrained    Logical value to indicate if the first alpha should be constrain to unity. By default TRUE

Value

Conditional maximisation routine set up using the partial likelihood derivatives. The argument net_tot takes the known net migration totals. The user must ensure that the net migration totals sum 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, Peter W. F. Smith

Examples

```r
m <- matrix(data = 1:16, nrow = 4)
# m[lower.tri(m)] <- t(m)[lower.tri(m)]
addmargins(m)
sum_net(m)

y <- cm_net_tot(net_tot = c(30, 40, -15, -55), tot = 200, m = m)
addmargins(y$n)
sum_net(y$n)

m <- matrix(data = c(0, 100, 30, 70, 50, 0, 45, 5, 60, 35, 0, 40, 20, 25, 20, 0),
nrow = 4, ncol = 4, byrow = TRUE,
dimnames = list(orig = LETTERS[1:4], dest = LETTERS[1:4]))
```
death_mat

```r
addmargins(m)
sum_net(m)

y <- cm_net_tot(net_tot = c(-100, 125, -75, 50), tot = 600, m = m)
addmargins(y$n)
sum_net(y$n)
```

description

Calculate deaths for each element of place of birth - place of residence stock matrix

Description

This function is predominantly intended to be used within the ffs routines in the migest package.

Usage

```r
depth_mat(
d_por = NULL,
m1 = NULL,
method = "proportion",
m2 = NULL,
b_por = NULL
)
```

Arguments

- `d_por` Vector of numeric values for deaths in each place of residence.
- `m1` 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 \( t \). Used to distribute deaths proportionally to each migrant stock population.
- `method` Character string of either "proportion" or "accounting" to choose method to distribute deaths. The "proportion" method assumes the mortality rate in each place of birth sub-group (native born and all foreign born stocks) is the same. The "accounting" method ensures that the number of deaths by place of birth matches that implied by demographic accounting. Still needs to be explored fully.
- `m2` 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 \( t+1 \). Used to distribute deaths proportionally to each migrant stock population. For use when `method = "accounting"`
- `b_por` Vector of numeric values for births in each place of residence. For use when `method = "accounting"`

Value

Matrix of place of death by place of residence
dict_ims  

Dictionary to look up region geographies based on countries used in UN DESA International Migrant Stock.

Description

Intended for use as a custom dictionary with the countrycode package, where the existing UN region and area codes do not match those used by UN DESA in the WPP, see https://github.com/vincentarelbundock/countrycode/issues/253

Usage

dict_ims

Format

Data frame with 237 rows and 18 columns. One of first three columns intended as input for origin in countrycode.

- **name**  
Country name

- **iso3c**  
ISO numeric code

- **iso3n**  
ISO 3 letter code

Remaining columns intended as input for destination in countrycode.

- **name_short**  
Short country name

- **region**  
Geographic region of country (6)

- **region_sub**  
Geographic sub region of country (22). Filled using region if none given in original data

- **region_sdg**  
SDG region of country (8)

- **region_sdg_sub**  
Sub SDG region of country (9). Filled using region_sdg if none given in original data

- **un_develop**  
UN development group of country (3)

- **wb_income**  
World Bank income group of country (3)

- **wb_income_detail**  
Detailed World Bank income group of country (4)

- **lldc**  
Indicator variable for Land-Locked Developing Countries (32)

- **sids**  
Indicator variable for Small Island Developing States (58)

- **region_as2014**  
Region grouping used for global chord diagram plots by Abel and Sander (2014)

- **region_sub2014**  
Region grouping used for global chord diagram plots by Sander, Abel and Bauer (2014)

- **region_a2018**  
Region grouping used for global chord diagram plots by Abel (2018)

- **region_ac2022**  
Region grouping used for global chord diagram plots by Abel and Cohen (2022)

- **region_wb**  
World Bank region
Source


Examples

```r
## Not run:
library(tidyverse)
library(countrycode)
# download Abel and Cohen (2019) estimates
f

# use dictionary to get region to region flows
d <- f %>%
  mutate(
    orig = countrycode(
      sourcevar = orig, custom_dict = dict_ims,
      origin = "iso3c", destination = "region"),
    dest = countrycode(
      sourcevar = dest, custom_dict = dict_ims,
      origin = "iso3c", destination = "region")
  ) %>%
  group_by(year0, orig, dest) %>%
  summarise_all(sum)

d
## End(Not run)
```

**ffs_demo**

*Estimation of bilateral migrant flows from bilateral migrant stocks using demographic accounting approaches*

Description

Estimates migrant transitions flows between two sequential migrant stock tables. Replaces old ffs.

Usage

```r
ffs_demo(
  m1 = NULL,
  m2 = NULL,
  b_por = NULL,
  d_por = NULL,
  m = NULL,
  stayer_assumption = TRUE,
  match_global = "before-demo-adjust",
  match_pob_tot_method = "rescale",
)```
birth_method = "native",
birth_non_negative = TRUE,
death_method = "proportion",
verbose = FALSE,
return = "flow"
)

Arguments

m1 
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

m2 
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.

b_por 
Vector of the number of births between time t and t+1 in each region.

d_por 
Vector of the number of deaths between time t and t+1 in each region.

m 
Matrix of auxiliary data. By default set to 1 for all origin-destination combinations.

stayer_assumption 
Logical value to indicate whether to use a quasi-independent or independent IPFP to estimate flows. By default uses quasi-independent, i.e. is set to TRUE and estimates the minimum migration. When set to FALSE estimates flows under the independent model as used as part of Azose and Raftery (2019).

match_global 
Character string used to indicate whether to balance the change in stocks totals with the changes in births and deaths. Only applied when match_pob_tot_method is either rescale or rescale-adjust-zero-fb. By default uses after-demo-adjust rather than before-demo-adjust which I think minimises risk of negative values.

match_pob_tot_method 
Character string passed to method argument in match_pob_tot to ensure place of birth margins in stock tables match.

birth_method 
Character string passed to method argument in birth_mat.

birth_non_negative 
Logical value passed to non_negative argument in birth_mat.

death_method 
Character string passed to method argument in death_mat.

verbose 
Logical value to show progress of the estimation procedure. By default FALSE.

return 
Character string used to indicate whether to return the array of estimated flows when set to flow (default), array of demographic accounts when set to account or the demographic account, list of input settings and the origin-destination matrix when set to classic

Value

Estimates migrant transitions flows between two sequential migrant stock tables using various methods. See the example section for possible variations on estimation methods.

Detail of returned object varies depending on the setting used in the return argument.
Author(s)
Guy J. Abel

References
Abel and Cohen (2019) Bilateral international migration flow estimates for 200 countries Scientific Data 6 (1), 1-13
Azose & Raftery (2019) Estimation of emigration, return migration, and transit migration between all pairs of countries Proceedings of the National Academy of Sciences 116 (1) 116-122

See Also
ffs_diff, ffs_rates

Examples
##
## without births and deaths over period
##
# data as in demographic research and science paper papers
s1 <- matrix(data = c(1000, 100, 10, 0, 55, 555, 50, 5, 80, 40, 800, 40, 20, 25, 20, 200),
nrow = 4, ncol = 4, byrow = TRUE)
s2 <- matrix(data = c(950, 100, 60, 0, 80, 505, 75, 5, 90, 30, 800, 40, 40, 45, 0, 180),
nrow = 4, ncol = 4, byrow = TRUE)
b <- d <- rep(0, 4)
r <- LETTERS[1:4]
dimnames(s1) <- dimnames(s2) <- list(birth = r, dest = r)
names(b) <- names(d) <- r
addmargins(s1)
addmargins(s2)
b
d
# demographic research and science paper example
e0 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d)
e0
sum_od(e0)

# international migration review paper example
s1[,] <- c(100, 20, 10, 20, 10, 55, 40, 25, 10, 25, 140, 20, 0, 10, 65, 200)
s2[,] <- c(70, 25, 10, 40, 30, 60, 55, 45, 10, 10, 140, 0, 10, 15, 50, 180)
addmargins(s1)
addmargins(s2)
e1 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d)
sum_od(e1)

# international migration review supp. material example
# distance matrix
dd <- matrix(data = c(0, 5, 50, 500, 5, 0, 45, 495, 50, 45, 0, 450, 500, 495, 450, 0),
nrow = 4, ncol = 4, byrow = TRUE)
dimnames(dd) <- list(orig = r, dest = r)
dd
e2 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d, m = dd)
sum_od(e2)

##
## with births and deaths over period
##
# demographic research paper example (with births and deaths)
s1[,] <- c(1000, 55, 80, 20, 100, 555, 40, 25, 10, 50, 800, 20, 0, 5, 40, 200)
s2[,] <- c(1060, 45, 70, 30, 60, 540, 75, 30, 10, 70, 770, 20, 10, 0, 70, 230)
b[] <- c(80, 20, 40, 60)
d[] <- c(70, 30, 50, 10)
e3 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d, match_pob_tot_method = "open-dr")
sum_od(e3)
# makes more sense to use this method
e4 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d, match_pob_tot_method = "open")
sum_od(e4)

# science paper supp. material example
b[] <- c(80, 20, 60, 60)
e5 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d)
sum_od(e5)

# international migration review supp. material example (with births and deaths)
s1[,] <- c(100, 20, 10, 20, 10, 55, 40, 25, 10, 25, 140, 20, 0, 10, 65, 200)
s2[,] <- c(75, 20, 30, 30, 25, 45, 40, 30, 5, 30, 150, 20, 0, 15, 60, 230)
b[] <- c(10, 50, 25, 60)
d[] <- c(30, 10, 40, 10)
e6 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d)
sum_od(e6)

# scientific data 2019 paper
s1[,] <- c(100, 80, 30, 60, 10, 180, 10, 70, 10, 10, 140, 10, 0, 90, 40, 160)
s2[,] <- c(95, 75, 55, 35, 25, 225, 0, 25, 15, 5, 115, 25, 5, 55, 50, 215)
b[] <- c(0, 0, 0, 0)
d[] <- c(0, 0, 0, 0)
e7 <- ffs_demo(m1 = s1, m2 = s2, b_por = b, d_por = d)
sum_od(e7)

ffs_diff
Estimation of bilateral migrant flows from bilateral migrant stocks using stock differencing approaches
**Description**

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

**Usage**

```r
ffs_diff(m1, m2, decrease = "return", include_native_born = FALSE)
```

**Arguments**

- `m1` 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 \( t \).
- `m2` 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 estimates). See details for possible options. Default is `return`.
- `include_native_born` Logical value to indicate whether to include diagonal elements of `m1` and `m2`. Default of `FALSE` - not include.

**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

**References**


**See Also**

`ffs_demo, ffs_rates`

**Examples**

```r
s1 <- matrix(data = c(100, 10, 10, 0, 20, 55, 25, 10, 10, 40, 140, 65, 20, 25, 20, 200),
             nrow = 4, ncol = 4, byrow = TRUE)
s2 <- matrix(data = c(75, 25, 5, 15, 20, 45, 30, 15, 30, 40, 150, 35, 10, 50, 5, 200),
             nrow = 4, ncol = 4, byrow = TRUE)
r <- LETTERS[1:4]
```
dimnames(s1) <- dimnames(s2) <- list(pob = r, por = r)
s1; s2

ffs_diff(m1 = s1, m2 = s2, decrease = "zero")
ffs_diff(m1 = s1, m2 = s2, decrease = "return")

---

**ffs_rates**

*Estimation of bilateral migrant flows from bilateral migrant stocks using rates approaches*

### Description

Estimates migrant transitions flows between two sequential migrant stock tables using approached based on rates.

### Usage

```r
ffs_rates(m1 = NULL, m2 = NULL, M = NULL, method = "dennett")
```

### Arguments

- **m1**: 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*.
- **m2**: 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*.
- **M**: Numeric value for the global sum of migration flows, used for dennett approach.
- **method**: Method to estimate flows. Can take values dennett or rogers-von-rabenau. See details section for more information. Uses dennett as default.

### Value

Estimates migrant transitions flows based on migration rates.

When `method = "dennett"` migration are derived from the matrix supplied to `m1`. Dennett uses bilateral migrant stocks at beginning of period. Rates then multiplied by global migration flows supplied in `M`.

When `method = "rogers-von-rabenau"` a matrix of growth rates are derived from the changes in initial populations stock `m1` to obtain `m2`;

\[ P^{t+1} = gP^t \]

and then multiplied by the corresponding populations at risk in `m1`. Can result in negative flows.

### Author(s)

Guy J. Abel
index_age

References


See Also

ffs_demo, ffs_rates

Examples

```r
s1 <- matrix(data = c(100, 10, 10, 0, 20, 55, 25, 10, 10, 40, 140, 65, 20, 25, 20, 200),
nrow = 4, ncol = 4, byrow = TRUE)
s2 <- matrix(data = c(75, 25, 5, 15, 20, 45, 30, 15, 30, 40, 150, 35, 10, 50, 5, 200),
nrow = 4, ncol = 4, byrow = TRUE)
r <- LETTERS[1:4]
dimnames(s1) <- dimnames(s2) <- list(pob = r, por = r)
s1; s2

# calculate total migration flows for dennett approach
n <- colSums(s2) - colSums(s1)
ffs_rates(m1 = s1, M = sum(abs(n)), method = "dennett")
ffs_rates(m1 = s1, m2 = s2, method = "rogers-von-rabenau")
```

---

**index_age**  
Summary indices of migration age profile

**Description**

Summary measures of migration age profiles as proposed by Rogers (1975), Bell et. al. (2002), Bell and Muhidin (2009) and Bernard, Bell and Charles-Edwards (2014)

**Usage**

```r
index_age(
  d = NULL,
  age,
  mi,
  age_min = 5,
  age_max = 65,
  breadth = 5,
  age_col = "age",
  mi_col = "mi",
  long = TRUE
)
```
**Arguments**

d  Data frame of age specific migration intensities. If used, ensure the correct column names are passed to `age_col` and `mi_col`.

age  Numeric vector of ages. Used if `d` = `NULL`.

mi  Numeric vector of migration intensities corresponding to each value of `age`. Used if `d` = `NULL`.

age_min  Numeric value for minimum age for peak calculations. Taken as 5 by default.

age_max  Numeric value for maximum age for peak calculations. Taken as 65 by default.

breadth  Numeric value for number of age groups around peak to be used in `breadth_peak` measure. Default of 5.

age_col  Character string of the age column name (when `d` is provided)

mi_col  Character string of the migration intensities column name (when `d` is provided)

long  Logical to return a long data frame with index values all in one column

**Value**

A tibble with 8 summary measures where

- `gmr`  Gross migraproduction rate of Rogers (1975)
- `peak_mi`  Peak migration intensities, from Bell et. al. (2002)
- `peak_age`  Corresponding age of `peak_mi`, from Bell et. al. (2002)
- `peak_breadth`  Breadth of peak, from Bell and Muhidin (2009)
- `peak_share`  Percentage share of peak breadth of all migration, from Bell and Muhidin (2009)
- `murc`  Maximum upward rate of change of Bernard, Bell and Charles-Edwards (2014)
- `mdrc`  Maximum downward rate of change of Bernard, Bell and Charles-Edwards (2014)
- `asymmetry`  Asymmetry between the `murc` and `mdrc`, from Bernard, Bell and Charles-Edwards (2014)

**Source**


index_age_rc

Examples

```r
library(dplyr)
ipumsi_age %>%
  filter(sample == "BRA2000") %>%
  mutate(mi = migrants/population) %>%
  index_age()

ipumsi_age %>%
  group_by(sample) %>%
  mutate(mi = migrants/population) %>%
  index_age(long = FALSE)
```

index_age_rc

Summary indices of age migration profile based on parameters from a Rogers and Castro schedule

Description

Summary indices of age migration profile based on parameters from a Rogers and Castro schedule

Usage

```r
index_age_rc(pars = NULL, long = TRUE)
```

Arguments

- `pars` Named vector or parameters parameters from a Rogers and Castro schedule
- `long` Logical to return a long data frame with index values all in one column

Value

A tibble with at least five summary measures

Source


Examples

```r
library(dplyr)
library(tibble)
rc_model_fund %>%
deframe() %>%
index_age_rc()
```
### Summary indices of migration connectivity

#### Description

Summary indices of migration connectivity

#### Usage

```r
index_connectivity(
  m = NULL,
  gini_orig_all = FALSE,
  gini_dest_all = FALSE,
  gini_corrected = TRUE,
  orig_col = "orig",
  dest_col = "dest",
  flow_col = "flow",
  long = TRUE
)
```

#### Arguments

- **m** A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to `orig_col`, `dest_col` and `flow_col`.
- **gini_orig_all** Logical to include gini index values for all origin regions. Default FALSE.
- **gini_dest_all** Logical to include gini index values for all destination regions. Default FALSE.
- **gini_corrected** Logical to use corrected denominator in Gini index of Bell (2002) or original of David A. Plane and Mulligan (1997)
- **orig_col** Character string of the origin column name (when `m` is a data frame rather than a matrix)
- **dest_col** Character string of the destination column name (when `m` is a data frame rather than a matrix)
- **flow_col** Character string of the flow column name (when `m` is a data frame rather than a matrix)
- **long** Logical to return a long data frame with index values all in one column

#### Value

A tibble with 12 summary measures:

- **connectivity** \( I_{mc} \) of Bell et. al. (2002) for the share of non-zero flows. A value of 0 means no connections (all zero flows) and 1 shows that all regions are connected by migrants.
inequality_equal
I_\text{mi} of Bell et al. (2002) based on a distributions of flows compared to equal distributions of expected flows. A value of 0 shows complete equality in flows and 1 shows maximum inequality.

inequality_sim
I_\text{mi} of Bell et al. (2002) based on a distributions of flows compared to distributions of expected flows from a Poisson regression independence fit flow ~ orig + dest. A value of 0 shows complete equality in flows and 1 shows maximum inequality.

gini_total
Overall concentration of migration from Bell (2002), corrected from Plane and Mulligan (1997). A value of 0 means no spatial focusing and 1 shows that all migrants are found in one single flow. Calculated using migration.indices::migration.gini.total()

gini_orig_standardized
Relative extent to which the origin selections of out-migrations are spatially focused. A value of 0 means no spatial focusing and 1 shows maximum focusing. Adapted from migration.indices::migration.gini.row.standardized().

gini_dest_standardized
Relative extent to which the destination selections of in-migrations are spatially focused. A value of 0 means no spatial focusing and 1 shows maximum focusing. Adapted from migration.indices::migration.gini.col.standardized().

mwg_orig
Origin spatial focusing, from Bell et al. (2002). Calculated using migration.indices::migration.weighted.gini.out()

mwg_dest
Destination spatial focusing, from Bell et al. (2002). Calculated using migration.indices::migration.weighted.gini.in()

mwg_mean
Mean spatial focusing, from Bell et al. (2002). Average of the origin and destination migration weighted Gini indices (mwg_orig and mwg_dest). A value of 0 means no spatial focusing and 1 shows that all migrants are found in one region. Calculated using migration.indices::migration.weighted.gini.mean()

cv
Coefficient of variation from Rogers and Raymer (1998).

acv
Aggregated system-wide coefficient of variation from Rogers and Sweeney (1998), using migration.indices::migration.acv()

Source


Examples

library(dplyr)
korea_reg %>%
  filter(year == 2020) %>%
  index_connectivity()

index_distance

Summary indices of migration distance

Description

Summary indices of migration distance

Usage

index_distance(
  m = NULL,
  d = NULL,
  orig_col = "orig",
  dest_col = "dest",
  flow_col = "flow",
  dist_col = "dist",
  long = TRUE
)

Arguments

m A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to orig_col, dest_col and flow_col.

d A matrix or data frame of origin-destination distances. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to orig_col, dest_col and dist_col. Region names should match those in m.

orig_col Character string of the origin column name (when m is a data frame rather than a matrix)

dest_col Character string of the destination column name (when m is a data frame rather than a matrix)

flow_col Character string of the flow column name (when m is a data frame rather than a matrix)

dist_col Character string of the distance column name (when dist is a data frame rather than a matrix)

long Logical to return a long data frame with index values all in one column
**Value**

A tibble with 3 summary measures where

- **mean**: Mean migration distance from Bell et. al. (2002) - not discussed in text but given in Table 6
- **median**: Mean migration distance from Bell et. al. (2002)
- **decay**: Distance decay parameter obtained from a Poisson regression model \( \text{flow} \sim \text{orig} + \text{dest} + \log(\text{dist}) \)

**Source**


**Examples**

```r
# single year
index_distance(
  m = subset(korea_reg, year == 2020),
  d = korea_dist
)

library(dplyr)
library(tidyr)
library(purrr)

# multiple years
korea_reg %>%
  nest(m = c(orig, dest, flow)) %>%
  mutate(d = list(korea_dist)) %>%
  mutate(i = map2(.x = m, .y = d,
    .f = ~index_distance(m = .x, d = .y, long = FALSE))) %>%
  select(-m, -d) %>%
  unnest(i)
```

<table>
<thead>
<tr>
<th>index_impact</th>
<th>Summary indices of migration impact</th>
</tr>
</thead>
</table>

**Description**

Summary indices of migration impact
Usage

```r
index_impact(
  m,
  p,
  pop_col = "pop",
  reg_col = "region",
  orig_col = "orig",
  dest_col = "dest",
  flow_col = "flow",
  long = TRUE
)
```

Arguments

- **m**: A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to `orig_col`, `dest_col` and `flow_col`.
- **p**: A data frame or named vector for the total population. When data frame, column of populations labelled using `pop_col` and region names labelled `reg_col`.
- **pop_col**: Character string of the population column name
- **reg_col**: Character string of the region column name. Must match dimension names or values in origin and destination columns of `m`.
- **orig_col**: Character string of the origin column name (when `m` is a data frame rather than a matrix)
- **dest_col**: Character string of the destination column name (when `m` is a data frame rather than a matrix)
- **flow_col**: Character string of the flow column name (when `m` is a data frame rather than a matrix)
- **long**: Logical to return a long data frame with index values all in one column

Value

A tibble with 4 summary measures where

- **effectiveness**: Migration effectiveness index (MEI) from Shryock et al. (1975). Values range between 0 and 100. High values indicate migration is an efficient mechanism of population redistribution, generating a large net migration. Conversely, low values denote that migration is closely balanced, leading to comparatively little redistribution.
- **anmr**: Aggregate net migration rate from Bell et al. (2002). The population weighted version of mei.
- **perference**: Index of preference, given in UN DESA (1983). From Bachi (1957) and Shryock et al. (1975) - measures size of migration compared to expected flows based on uniform migration. Can go from 0 to infinity
velocity  Index of velocity, given in UN DESA (1983). From Bogue, Shryock, Jr. & Hoermann (1957) - measures size of migration compared to expected flows based on population size alone. Can go from 0 to infinity

Source


Examples

# single year
index_intensity(m = subset(korea_reg, year == 2020),
p = subset(korea_pop, year == 2020),
  pop_col = "population")

# multiple years
library(dplyr)
library(tidyr)
library(purrr)
korea_reg %>%
  nest(m = c(orig, dest, flow)) %>%
  left_join(korea_pop) %>%
  nest(p = c(region, population)) %>%
  mutate(i = map2(.x = m, .y = p,
                .f = ~ index_intensity(m = .x, p = .y, pop_col = "population", long = FALSE))) %>%
  select(-m, -p) %>%
  unnest(i)

index_intensity  Summary indices of migration intensity

Description

Summary indices of migration intensity
Usage

index_intensity(mig_total = NULL, pop_total = NULL, n = NULL, long = TRUE)

Arguments

mig_total Numeric value for the total number of migrations.
pop_total Numeric value for the total population.
n Numeric value for the number of regions used in the definition of migration for 
mig_total.
long Logical to return a long data frame with index values all in one column

Value

A tibble with 2 summary measures where

cmp Crude migration probability from Bell et. al. (2002), sometimes known as crude
migration intensity, e.g. Bernard (2017)
courgeau_k Intensity measure of Courgeau (1973)

Source

Bell, M., Blake, M., Boyle, P., Duke-Williams, O., Rees, P. H., Stillwell, J., & Hugo, G. J. (2002).
Cross-national comparison of internal migration: issues and measures. Journal of the Royal Sta-
tical Society: Series A (Statistics in Society), 165(3), 435–464. https://doi.org/10.1111/1467-
985X.00247

internal migration across the countries of Latin America: A multidimensional approach. Plos One,
12(3), e0173895. https://doi.org/10.1371/journal.pone.0173895

Examples

# single year
library(dplyr)
m <- korea_reg %>%
 filter(year == 2020, 
 orig != dest) %>%
 pull(flow) %>%
 sum()
m
p <- korea_pop %>%
 filter(year == 2020) %>%
 pull(population) %>%
 sum()
p
index_intensity(mig_total = m, pop_total = p, n = n_distinct(korea_pop$region))

# multiple years
mm <- korea_reg %>%
```r
mm <- korea_pop %>%
  group_by(year) %>%
  summarise(m = sum(flow))

mm

pp <- korea_pop %>%
  group_by(year) %>%
  summarise(p = sum(population))

library(purrr)
library(tidyr)

mm %>%
  left_join(pp) %>%
  mutate(i = map2(.x = m, .y = p,
                 .f = ~index_intensity(mig_total = .x,
                                       pop_total = .y,
                                       n = n_distinct(korea_pop$region),
                                       long = FALSE))) %>%
  unnest(cols = i)
```

---

**indian_sub**

Lifetime migration totals for states and zones in the Indian 1901 to 1931

---

**Description**

Lifetime migration (stock) totals from India

**Usage**

`indian_sub`

**Format**

Data frame with 164 rows and 7 columns:

- **zone**: Zone of state. In some cases the state and zone are the same entity
- **state**: Indian state
- **sex**: Migrant sex
- **in_migrants**: In-migrant total based on birthplace
- **out_migrants**: Out-migrant total based on birthplace
- **net_migrants**: Net migrant total based on birthplace

**Source**


Scraped from [https://archive.org/details/in.ernet.dli.2015.130424/page/n73/mode/2up](https://archive.org/details/in.ernet.dli.2015.130424/page/n73/mode/2up)
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.

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 $\alpha_i$ term if ($row_{tot} = NULL$) or without the $\beta_j$ term if ($col_{tot} = NULL$).

Returns a list object with

---

**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**

```R
ipf2(
    row_tot = NULL,
    col_tot = NULL,
    m = matrix(1, length(row_tot), length(col_tot)),
    tol = 1e-05,
    maxit = 500,
    verbose = FALSE
)
```

**Arguments**

- `row_tot` Vector of origin totals to constrain the sum of the imputed cell rows.
- `col_tot` 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.
ipf2

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

Author(s)

Guy J. Abel

References


See Also

cm2, ipf3

Examples

```r
## with Willekens (1999) data
dn <- LETTERS[1:2]
y <- ipf2(row_tot = c(18, 20), col_tot = 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(row_tot = c(18, 20), col_tot = c(16, 22))
round(addmargins(y$mu),2)

## with bigger matrix
dn <- LETTERS[1:3]
y <- ipf2(row_tot = c(170, 120, 410), col_tot = 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(row_tot = c(18, 20), col_tot = 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

```r
ipf2_block(
  row_tot = NULL,
  col_tot = NULL,
  block_tot = NULL,
  block = NULL,
  m = NULL,
  tol = 1e-05,
  maxit = 500,
  verbose = TRUE,
  ...
)
```

#### Arguments

- **row_tot**: Vector of origin totals to constrain the sum of the imputed cell rows.
- **col_tot**: Vector of destination totals to constrain the sum of the imputed cell columns.
- **block_tot**: Matrix of block totals to constrain the sum of the imputed cell blocks.
- **block**: Matrix of block structure corresponding to `block_tot`.
- **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 row_tot and col_tot take the row-table and column-table specific known margins. The block_tot 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

Examples

```r
y <- ipf2_stripe(row_tot= c(30,20,30,10,20,5,0,10,5,5,5,10),
col_tot = c(45,10,10,5,10,50,5,10,0,0,0,0),
block_tot = 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)
```
Usage

    ipf2_stripe(
      row_tot = NULL,
      col_tot = NULL,
      stripe_tot = NULL,
      stripe = NULL,
      m = NULL,
      tol = 1e-05,
      maxit = 500,
      verbose = TRUE,
      ...
    )

Arguments

    row_tot    Vector of origin totals to constrain the sum of the imputed cell rows.
    col_tot    Vector of destination totals to constrain the sum of the imputed cell columns.
    stripe_tot Matrix of stripe totals to constrain the sum of the imputed cell blocks.
    stripe    Matrix of stripe structure corresponding to stripe_tot.
    m          Matrix of auxiliary data. By default set to 1 for all origin-destination combina-
                tions.
    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 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 row_tot and col_tot take the row-table and column-table specific known margins. The stripe_tot 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 character-
                istic
    it        Iteration count
    tol       Tolerance level at final iteration

Author(s)

Guy J. Abel
**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**

```r
ipf3(
  row_tot = NULL,
  col_tot = NULL,
  m = NULL,
  tol = 1e-05,
  maxit = 500,
  verbose = TRUE
)
```

**Arguments**

- `row_tot` Vector of origin totals to constrain the sum of the imputed cell rows.
- `col_tot` 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 row_tot and col_tot 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 and Cohen (2019) Bilateral international migration flow estimates for 200 countries Scientific Data 6 (1), 1-13
Azose & Raftery (2019) Estimation of emigration, return migration, and transit migration between all pairs of countries Proceedings of the National Academy of Sciences 116 (1) 116-122

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,
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**

```r
ipf3_qi(
  row_tot = NULL,
  col_tot = NULL,
  diag_count = NULL,
  m = NULL,
  speed = TRUE,
  tol = 1e-05,
  maxit = 500,
  verbose = TRUE
)
```
Arguments

- **row_tot**: Vector of origin totals to constrain the sum of the imputed cell rows.
- **col_tot**: Vector of destination totals to constrain the sum of the imputed cell columns.
- **diag_count**: 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 `row_tot` and `col_tot` 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
# ipf_seed

Quickly create IPF seed

## References


## See Also

ipf3, ffs_demo

## Examples

```r
## 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_qi(row_tot = t(P1), col_tot = P2)
# display with row, col and table totals
# round(addmargins(y$mu), 1)
# origin-destination flow table
# round(sum_od(y$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_qi(row_tot = t(P1), col_tot = P2, m = dis)
# display with row, col and table totals
# round(addmargins(y$mu), 1)
# origin-destination flow table
# round(sum_od(y$mu), 1)
```
Description
This function is predominantly intended to be used within the ipf routines in the migest package.

Usage
ipf_seed(m = NULL, R = NULL, n_dim = NULL, dn = NULL)

Arguments
m Matrix, Array or NULL to build seed. If NULL seed will be 1 for all elements.
R Number of rows, columns and possibly n_dimensions for seed matrix or array.
n_dim Numeric integer for the number of n_dimensions - 2 for matrix, 3 or more for an array
dn Vector of character strings for n_dimension names

Value
An array or matrix

Author(s)
Guy J. Abel

Description
Age specific migration and population counts from two IPUMSI samples from Brazil 2000 and France 2006 IPUMS International samples. Attempt to recreate the unsmoothed data used in the appendix of Bernard, Bell and Charles-Edwards (2014)

Usage
ipumsi_age

Format
Data frame with 202 rows and 4 columns:
sample IPUMS International sample - either BRA2000 or FRA2006
age Age on census data
migrants Number of migrants, defined by those who had changed usual place of residence to a different minor administrative region compared to usual place of residence five years prior to the census. Obtained by summing person weights for ‘migrate5’ variable equal to any of code 12, 20 or 30.
population Population of each age group, obtained by summing person weights ‘perwt’ variable.
Source


---

**italy_area**

Single year age-specific origin destination migration flows between Italian NUTS1 areas

---

Description

Origin-destination migration flows from 7 years between 1970 and 2000 by five-year age groups

Usage

italy_area

Format

Data frame with 3500 rows and 5 columns:

- **orig** Origin area (NUTS1 region)
- **dest** Destination area (NUTS1 region)
- **year** Year of flow
- **age_grp** Five-year age group
- **flow** Migration flow

Source


### korea_dist

**Description**

Distance matrix of kilometers between population weighted centroids in 2020 of first level administrative regions of South Korea.

**Usage**

`korea_dist`

**Format**

An object of class `matrix` (inherits from `array`) with 17 rows and 17 columns.

**Source**


### korea_pop

**Description**

Annual resident population totals between 2012 and 2020 based on first level administrative regions.

**Usage**

`korea_pop`

**Format**

Data frame with 2,601 rows and 4 columns:

- **region** Administrative region
- **year** Year of flow
- **population** Resident Population

**Source**

**korea_reg**

Annual origin destination migration flows between Korean regions

**Description**

Origin-destination migration flows between 2012 and 2020 based on first level administrative regions.

**Usage**

korea_reg

**Format**

Data frame with 2,601 rows and 4 columns:

- **orig** Origin region
- **dest** Destination region
- **year** Year of flow
- **flow** Migration flow

**Source**


**manila_1970**

Manila female population 1970 by age

**Description**

Population data for Manila by age in 1960 and 1970

**Usage**

manila_1970

**Format**

Data frame with 13 rows and 5 columns:

- **age_1970** Age group in 1970
- **pop_1960** Enumerated population in 1960
- **pop_1970** Enumerated population in 1970
- **phl_census_sr** Census survival ratio derived from the national data.
match_pob_tot

Adjust migrant stock tables to have matching place of birth totals

Description

This function is predominantly intended to be used within the ffs routines in the migest package.

Usage

match_pob_tot(m1, m2, method = "rescale", verbose = FALSE)

Arguments

m1 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+1.

m2 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.

method Character string matching either rescale, rescale-adjust-zero-fb, open or open-dr. See details.

verbose Logical value to indicate the print the parameter estimates at each iteration of the rescale, as used in ipf2. By default FALSE.

Details

The rescale and rescale-adjust-zero-fb method ensure flow estimates closely match the net migration totals implied by the changes in population totals, births and deaths - as introduced in the Science paper. The rescale-adjust-zero-fb can adjust for rare cases when row total margins that are smaller than native born totals in countries where there are no foreign born populations (e.g. South Sudan 1990-1995). The open-dr method allows for moves in and out of the global system - as introduced in the Demographic Research paper. The open method is a slight improvement over open-dr - the calculation of the moves and in and out using more sensible weights.
**Value**

Returns a list object with:

- `m1_adj`: Matrix of adjusted `m1` where rows (place of births) match `m2_adj`.
- `m2_adj`: Matrix of adjusted `m2` where rows (place of births) match `m1_adj`.
- `in_mat`: Matrix of estimated inflows into the system.
- `out_mat`: Matrix of estimated outflows from the system.

**Author(s)**

Guy J. Abel

**References**


Azose & Raftery (2019) Estimation of emigration, return migration, and transit migration between all pairs of countries *Proceedings of the National Academy of Sciences* 116 (1) 116-122


**See Also**

- `ipf3_qi`, `ffs_diff`

---

### mig_chord

**Chord diagram for directional origin-destination data**

**Description**

Adaption of `circlize::chordDiagramFromDataFrame()` with defaults set to allow for more effective visualisation of directional origin-destination data

**Usage**

```r
mig_chord(
  x,
  lab = NULL,
  lab_bend1 = NULL,
  lab_bend2 = NULL,
  label_size = 1,
  label_nudge = 0,
  label_squeeze = 0,
)```
axis_size = 0.8,
axis_breaks = NULL,
...,  
no_labels = FALSE,
no_axis = FALSE,
clear_circos_par = TRUE,
zero_margin = TRUE,
start.degree = 90,
gap.degree = 4,
track.margin = c(-0.1, 0.1),
points.overflow.warning = FALSE
)

Arguments

x  Data frame with origin in first column, destination in second column and bilateral measure in third column
lab  Named vector of labels for plot. If NULL will use names from d
lab_bend1  Named vector of bending labels for plot. Note line breaks do not work with facing = "bending" in circlize.
lab_bend2  Named vector of second row of bending labels for plot.
label_size  Font size of label text.
label_nudge  Numeric value to nudge labels towards (negative number) or away (positive number) the sector axis.
label_squeeze  Numeric value to nudge lab_bend1 and lab_bend2 labels apart (negative number) or together (positive number).
axis_size  Font size on axis labels.
axis_breaks  Numeric value for how often to add axis label breaks. Default not activated, uses default from circlize::circos.axis()
...  Arguments for circlize::chordDiagramFromDataFrame().
no_labels  Logical to indicate if to include plot labels. Set to FALSE by default.
no_axis  Logical to indicate if to include plot axis. Set to FALSE by default.
clear_circos_par  Logical to run circlize::circos.clear(). Set to TRUE by default. Set to FALSE if you wish to add further to the plot.
zero_margin  Set margins of the plotting graphics device to zero. Set to TRUE by default.
start.degree  Argument for circlize::circos.par().
gap.degree  Argument for circlize::chordDiagramFromDataFrame().
track.margin  Argument for circlize::chordDiagramFromDataFrame().
points.overflow.warning  Argument for circlize::chordDiagramFromDataFrame().
mig_chord

Value

Chord diagram based on first three columns of \texttt{x}. The function tweaks the defaults of \texttt{circlize::chordDiagramFromDataFrame()} for easier plotting of directional origin-destination data. Users can override these defaults and pass additional tweaks using any of the \texttt{circlize::chordDiagramFromDataFrame()} arguments.

The layout of the plots are designed to specifically work on plotting images into PDF devices with widths and heights of 7 inches (the default dimension when using the \texttt{pdf} function). See the end of the examples for converting PDF to PNG images in R.

Fitting the sector labels on the page is usually the most time consuming task. Use the different label options, including line breaks, \texttt{label_nudge}, track height in \texttt{preAllocateTracks} and font sizes in \texttt{label_size} and \texttt{axis_size} to find the best fit. If none of the label options produce desirable results, plot your own using \texttt{circlize::circos.text} having set \texttt{no_labels = TRUE} and \texttt{clear_circos_par = FALSE}.

Examples

```r
## Not run:
library(tidyverse)
library(countrycode)
# download Abel and Cohen (2019) estimates
f

# use dictionary to get region to region flows
d <- f %>%
  mutate(
    orig = countrycode(sourcevar = orig, custom_dict = dict_ims,
      origin = "iso3c", destination = "region"),
    dest = countrycode(sourcevar = dest, custom_dict = dict_ims,
      origin = "iso3c", destination = "region")
) %>%
  group_by(year0, orig, dest) %>%
  summarise_all(sum) %>%
  ungroup()
d

# 2015-2020 pseudo-Bayesian estimates for plotting
pb <- d %>%
  filter(year0 == 2015) %>%
  mutate(flow = da_pb_closed/1e6) %>%
  select(orig, dest, flow)
pb

# pdf(file = "chord.pdf")
mig_chord(x = pb)
# dev.off()
# file.show("chord.pdf")

# pass arguments to circlize::chordDiagramFromDataFrame
# pdf(file = "chord.pdf")
mig_chord(x = pb,
  # order of regions
```
mig_matrix

Helper function to format migration input
mig_tibble

Description
Helper function to format migration input

Usage
mig_matrix(
m,  
array = TRUE,  
orig_col = "orig",  
dest_col = "dest",  
flow_col = "flow"
)

Arguments

m A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to orig_col, dest_col and flow_col.
array Logical on return of array of all dimensions or origin-destination matrix (summed over all other dimensions)
orig_col Character string of the origin column name (when m is a data frame rather than a matrix)
dest_col Character string of the destination column name (when m is a data frame rather than a matrix)
flow_col Character string of the flow column name (when m is a data frame rather than a matrix)

Value
Formatted matrix

mig_tibble Helper function to format migration input

Description
Helper function to format migration input

Usage
mig_tibble(m, orig_col = "orig", dest_col = "dest", flow_col = "flow")
**Arguments**

- **m**: A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to `orig_col`, `dest_col` and `flow_col`.

- **orig_col**: Character string of the origin column name (when `m` is a data frame rather than a matrix).

- **dest_col**: Character string of the destination column name (when `m` is a data frame rather than a matrix).

- **flow_col**: Character string of the flow column name (when `m` is a data frame rather than a matrix).

**Value**

Formatted tibble

```r
multi_comp

  # A tibble: n x 3
  #      orig     dest     flow
  #     ---     ---     ---
  # 1     A       B       C
```

**Description**

Multiplicative component descriptions of *n*-dimension flow tables based on total reference coding system.

**Usage**

```r
multi_comp(m)
```

**Arguments**

- **m**: matrix or array of migration flows

**Value**

matrix or array of multiplicative components of `m`. When output is an array the total for each table of origin-destination flows is used.

**References**


Examples

```r
r <- LETTERS[1:4]
m0 <- matrix(data = c(0, 100, 30, 70, 50, 0, 45, 5, 60, 35, 0, 40, 20, 25, 20, 0),
nrow = 4, ncol = 4, byrow = TRUE, dimnames = list(orig = r, dest = r))
addmargins(m0)
multi_comp(m = m0)
```

# data frame
library(dplyr)
italy_area %>%
  filter(year == 2000) %>%
multi_comp() %>%
  round(digits = 3)
```

**multi_comp2**

Multiplicative component descriptions of origin-destination flow tables based on total reference coding system.

**Description**

Multiplicative component descriptions of origin-destination flow tables based on total reference coding system.

**Usage**

```r
multi_comp2(m)
```

**Arguments**

- `m` matrix of migration flows

**Value**

matrix of multiplicative components of `m`. When output is an array the total for each table of origin-destination flows is used.

**References**


Examples

```r
r <- LETTERS[1:2]
m0 <- array(c(5, 1, 2, 7, 4, 2, 5, 9), dim = c(2, 2, 2),
            dimnames = list(orig = r, dest = r, type = c("ILL", "HEALTHY")))
addmargins(m0)
multi_comp2(m = m0)
```

---

**nchars_wrap**

*Count the number of characters per line*

### Description

Count the number of characters per line

### Usage

```r
nchars_wrap(b, w)
```

### Arguments

- **b**
  - Numeric vector for the position of line breaks between the words in `w`
- **w**
  - Character string vector of words

### Value

List with vectors for number of characters per line and the number of words per line

---

**net_sr**

*Estimate net migration from survival ratios applied to lifetime migration data*

### Description

Using survival ratios to estimate net migration from lifetime migration data

### Usage

```r
net_sr(
    .data,
    pop0_col = "pop0",
    pop1_col = "pop1",
    survival_ratio_col = "sr",
    net_children = FALSE,
    maternal_exposure = c(0.25, 0.75),
    maternal_age_id = 4:9,
    maternal_col = pop1_col
)
```
Arguments

.data A data frame with two rows with the total number of lifetime in- and out-
migrants in separate columns. The first row contains totals at the first time point
and second row at the second time point.

pop0_col Character string name of column containing name of initial populations. Default
"pop0".

pop1_col Character string name of column containing name of end populations. Default
"pop1".

survival_ratio_col Character string name of column containing survivor ratios. Default "sr".

net_children Logical to indicate if to estimate net migration when no survival ratio exists.
Default `FALSE`.

maternal_exposure Vector for maternal exposures to interval to be used to estimate net migration
for each of the unknown children age groups. Length should correspond to the
number of children age groups where net migration estimates are required.

maternal_age_id Row numbers to indicate which rows correspond to maternal age groups at the
end of the period.

maternal_col Name of maternal population column, required for the estimation of net migra-
tion of children.

Value

Data frame with estimates of net migration

References

and Family Study Center. University of Chicago.

Examples

# results to match un manual 1984 (table 24)
net_sr(bombay_1951, pop0_col = "pop_1941", pop1_col = "pop_1951")

# results to match Bogue, Hinze and White (1982)
library(dplyr)
alabama_1970 %>%
  filter(race == "white", sex == "male") %>%
  select(-race, -sex) %>%
  group_by(age_1970) %>%
  net_sr(pop0_col = "pop_1960", pop1_col = "pop_1970",
         survival_ratio_col = "us_census_sr")

# results to match UN manual 1992 (table 6)
net_sr(manila_1970, pop0_col = "pop_1960", pop1_col = "pop_1970",
       survival_ratio_col = "phl_census_sr")
# with children net migration estimate
net_sr(manila_1970, pop0_col = "pop_1960", pop1_col = "pop_1970",
       survival_ratio_col = "phl_census_sr", net_children = TRUE)

---

net_vs

## net_mig

### Description

Estimate net migration from vital statistics

### Usage

```r
net_vs(
  .data,
  pop0_col = NULL,
  pop1_col = NULL,
  births_col = "births",
  deaths_col = "deaths"
)
```

### Arguments

- **.data**
  - A data frame with two rows with the total number of lifetime in- and out-migrants in separate columns. The first row contains totals at the first time point and second row at the second time point.

- **pop0_col**
  - Character string name of column containing name of initial populations. Default "pop0".

- **pop1_col**
  - Character string name of column containing name of end populations. Default "pop1".

- **births_col**
  - Character string name of column containing name of births over the period. Default "births".

- **deaths_col**
  - Character string name of column containing name of deaths over the period. Default "deaths".

### Value

A tibble with additional columns for the population change ("pop_change"), the natural population increase ("natural_inc") and the net migration ("net") over the period.

### References

library(dplyr)
d <- alabama_1970 %>%
  group_by(race, sex) %>%
  summarise(births = sum(pop_1960[1:2]),
             pop_1960 = sum(pop_1960) - births,
             pop_1970 = sum(pop_1970)) %>%
  ungroup()
d
d %>%
  mutate(deaths = c(51449, 58845, 86880, 123220)) %>%
  net_vs(pop0_col = "pop_1960", pop1_col = "pop_1970")

---

New England male white-native population totals in 1950 and 1960
by place of birth and age

Description

New England population data for by place of birth and age in 1950 and 1960 for male white native born.

Usage

new_england_1960

Format

Data frame with 72 rows and 4 columns:

- birthplace: Place of birth (US Census area)
- year: Year
- age_1960: Age group in 1960
- pop_1950: Enumerated population in 1950
- pop_1960: Enumerated population in 1960

Source

United States Bureau of the Census, United States Census of Population: 1960..Subject Reports."State of birth" (Washington, D.C.), table 25, pp. 61-62. Persons with place of birth not reported were distributed pro rata among those with place of birth reported.

quadratic_eqn

Solutions from the 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 corresponding to the roots for the quadratic equation.

Author(s)

Guy J. Abel

Source

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

Examples

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

**Description**

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

**Usage**

`rc_model_fund`

**Format**

A tibble with two columns and seven rows:

- **param** Character string for the seven parameters
- **value** Parameter values

**Source**


---

rc_model_un

**Description**

Sets of parameters for the Rogers-Castro migration age schedule proposed by UN DESA

**Usage**

`rc_model_un`

**Format**

A tibble with five columns and 84 rows:

- **schedule** Character string for full name of schedule
- **value** Character string for abbreviated name of schedule
- **param** Character string for sex of schedule
- **param** Character string for the seven parameters
- **value** Parameter values
rescale_integer_sum

Rescale integer vector to a set sum

Description

For when you want to rescale a set of numbers to sum to a given value and do not want all rescaled values to be integers.

Usage

rescale_integer_sum(x, tot)

Arguments

x Vector of numeric values
tot Numeric integer value to rescale sum to.

Value

Vector or integer values that sum to tot

Author(s)

Guy J. Abel

See Also

ipf3_qi, ffs_diff

Examples

```r
x <- rnorm(n = 10, mean = 5, sd = 20)
y <- rescale_integer_sum(x, tot = 10)
y
sum(y)
```

```r
for(i in 1:10){
  y <- rescale_integer_sum(x = rpois(n = 10, lambda = 10), tot = 1000)
  print(sum(y))
}
```
rescale_nb

Rescale native born populations to match global differences in births and deaths over period

Description

This function is predominantly intended to be used within the ffs routines in the migest package. Adjustment to ensure that global differences in stocks match the global demographic changes from births and deaths.

Usage

rescale_nb(m1, m2, b, d, verbose = FALSE)

Arguments

m1 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

m2 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.

b Vector of the number of births between time t and t+1 in each region.

d Vector of the number of deaths between time t and t+1 in each region.

verbose Logical value to indicate the print the parameter estimates at each iteration. By default FALSE.

Value

List with adjusted m1 and m2.

Author(s)

Guy J. Abel

See Also

ipf3_qi, ffs_diff

Examples

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)

# births and deaths
b <- rep(x = 10, 4)
d <- rep(x = 5, 4)
# no change in stocks, but 20 more births than deaths...
sum(P2 - P1) + sum(b - d)

# rescale
# y <- rescale_nb(m1 = P1, m2 = P2, b = b, d = d)
# y
# sum(y$m1_adj - y$m2_adj) + sum(b - d)

# check for when extra is positive and odd
d[1] <- 31
d
sum(P2 - P1) - sum(b - d)

# rescale
# y <- rescale_nb(m1 = P1, m2 = P2, b = b, d = d)
# y
# sum(y$m1_adj - y$m2_adj) - sum(b - d)

---

**rescale_net**

Rescale net migration total to a global zero sum

### Description

Modify a set of net migration (or any numbers) so that they sum to zero.

### Usage

```r
rescale_net(
  x,
  method = "no-switches",
  w = rep(1, length(x)),
  integer_result = TRUE
)
```

### Arguments

- **x**
  - Vector of net migration values
- **method**
  - Method used to adjust net migration values of `x` to obtain a global zero sum. By default method="no-switches". Can also take values method="switches". See details for explanation on each method.
- **w**
  - Weights used in rescaling method
- **integer_result**
  - Logical operator to indicate if output should be integers, default is TRUE.
**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)`

---

**Value**

Rescales net migration for a number of regions in vector `x` to sum to zero. When `method="no-switches"` rescaling of values are done for the positive and negative values separately, to ensure the final global sum is zero. When `method="switches"` the mean of the unscaled net migration is subtracted from each value.

**Author(s)**

Guy J. Abel

**References**


**Examples**

```r
# net migration in regions countries (does not add up to zero)
x <- c(-200, -30, -5, 0, 10, 20, 60, 80)
x
sum(x)
# rescale
y1 <- rescale_net(x)
y1
sum(y1)
# rescale without integer restriction
y2 <- rescale_net(x, integer_result = FALSE)
y2
sum(y2)
# rescale allowing switching of signs (small negative value becomes positive)
y3 <- rescale_net(x, method = "switches")
y3
sum(y3)
```
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`, `ipf2_stripe`

Examples

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

---

### str_wrap_n

**Wrap character string to fit a target number of lines**

Description

Inserts line breaks for spaces, where the position of the line breaks are chosen to provide the most balanced length of each line.

Usage

```r
str_wrap_n(string = NULL, n = 2)
```

Arguments

- **string**: Character string to be broken up
- **n**: Number of lines to break the string over
Details

Function is intended for a small number of line breaks. The n argument is not allowed to be greater than 8 as all combinations of possible line breaks are explored.

When there a number of possible solutions that provide equally balanced number of characters in each line, the function returns the character string where the number of spaces are distributed most evenly.

Value

The original string with line breaks inserted at optimal positions.

Examples

```r
str_wrap_n(string = "a bb ccc dddd eeee fffffff", n = 2)
str_wrap_n(string = "a bb ccc dddd eeee fffffff", n = 4)
str_wrap_n(string = "a bb ccc dddd eeee fffffff", n = 8)
str_wrap_n(string = c("a bb", "a bb ccc"), n = 2)
```

---

**str_wrap_n_single**  
**Single line wrap for string**

Description

Single line wrap for string

Usage

```r
str_wrap_n_single(string = NULL, n = 2)
```

Arguments

- **string**: string from `str_wrap_n`
- **n**: n from from `str_wrap_n`

Value

String with line breaks
sum_bilat

Summary of bilateral flows, counter-flow and net migration flow

Description
Summary of bilateral flows, counter-flow and net migration flow

Usage
sum_bilat(
  m,
  label = "flow",
  orig_col = "orig",
  dest_col = "dest",
  flow_col = "flow"
)

Arguments
m
A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to orig_col, dest_col and flow_col.

label
Character string for the prefix of the calculated columns. Can take values flow or stream

orig_col
Character string of the origin column name (when m is a data frame rather than a matrix)

dest_col
Character string of the destination column name (when m is a data frame rather than a matrix)

flow_col
Character string of the flow column name (when m is a data frame rather than a matrix)

Value
A tibble with columns for orig, destination, corridor, flow, counter-flow and net flow in each bilateral pair.

Examples
# matrix
r <- LETTERS[1:4]
m <- matrix(data = c(0, 100, 30, 70, 50, 0, 45, 5, 60, 35, 0, 40, 20, 25, 20, 0),
nrow = 4, ncol = 4, dimnames = list(orig = r, dest = r), byrow = TRUE)
m
sum_bilat(m)

# data frame
library(dplyr)
library(tidyr)

d <- expand_grid(orig = r, dest = r, sex = c("female", "male")) %>%
  mutate(flow = sample(x = 1:100, size = 32))

d
# use group_by to distinguish od tables

d %>%
  group_by(sex) %>%
  sum_bilat()

---

**sum_expand**

*Sum bilateral data to include aggregate bilateral totals for origin and destination meta areas*

---

**Description**

Expand matrix of data frame of migration data to include aggregate sums for corresponding origin and destination meta regions.

**Usage**

```r
sum_expand(
  m,
  return_matrix = FALSE,
  guess_order = TRUE,
  area_first = TRUE,
  orig_col = "orig",
  dest_col = "dest",
  flow_col = "flow",
  orig_area_col = "orig_area",
  dest_area_col = "dest_area",
  orig_area = NULL,
  dest_area = NULL
)
```

**Arguments**

- **m**: A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to `orig_col`, `dest_col` and `flow_col`.
- **return_matrix**: Logical to return a matrix. Default `FALSE`.
- **guess_order**: Logical to return a matrix or data frame ordered by origin and destination with area names at the end of each block. Default `TRUE`. If `FALSE` returns matrix or data frame based on alphabetical order of origin and destinations.
- **area_first**: Order area sums to be placed before the origin and destination values. Default `TRUE`. 
orig_col Character string of the origin column name (when \( m \) is a data frame rather than a matrix)

dest_col Character string of the destination column name (when \( m \) is a data frame rather than a matrix)

flow_col Character string of the flow column name (when \( m \) is a data frame rather than a matrix)

orig_area_col Character string of the origin area column name (when \( m \) is a data frame rather than a matrix)

dest_area_col Character string of the destination area column name (when \( m \) is a data frame rather than a matrix)

orig_area Vector of labels for the origin areas of each row of \( m \).

dest_area Vector of labels for the destination areas of each row of \( m \).

Value

A tibble or matrix with additional row and columns (for matrices) for aggregate sums for origin and destination meta-regions

Examples

```r
## from matrix
##
m <- block_matrix(x = 1:16, b = c(2,3,4,2))
m

# requires a vector of origin and destination areas
a <- rep(LETTERS[1:4], times = c(2,3,4,2))
a
sum_expand(m = m, orig_area = a, dest_area = a)

# place area sums after regions
sum_expand(m = m, orig_area = a, dest_area = a, area_first = FALSE)

## from large data frame
##
## Not run:
library(tidyverse)
library(countrycode)

# download Abel and Cohen (2019) estimates
f

# 1990-1995 flow estimates
f %>%
  filter(year0 == 1990) %>%
  mutate(
    orig_area = countrycode(sourcevar = orig, custom_dict = dict_ims,
sum_lump

origin = "iso3c", destination = "region"),
dest_area = countrycode(sourcevar = dest, custom_dict = dict_ims,
origin = "iso3c", destination = "region")
) %>%
sum_expand(flow_col = "da_pb_closed", return_matrix = FALSE)

# by group (period)
f %>%
mutate(
  orig_area = countrycode(sourcevar = orig, custom_dict = dict_ims,
  origin = "iso3c", destination = "region"),
  dest_area = countrycode(sourcevar = dest, custom_dict = dict_ims,
  origin = "iso3c", destination = "region")
) %>%
group_by(year0) %>%
sum_expand(flow_col = "da_pb_closed", return_matrix = FALSE)

## End(Not run)

---

**sum_lump**

_Sum and lump together small flows into a “other” category_

---

**Description**

Lump together regions/countries if their flows are below a given threshold.

**Usage**

```r
define_func("sum_lump", 
  m, 
  threshold = 1, 
  lump = "flow", 
  other_level = "other", 
  complete = FALSE, 
  fill = 0, 
  return_matrix = TRUE, 
  orig_col = "orig", 
  dest_col = "dest", 
  flow_col = "flow"
)
```

**Arguments**

- **m**: A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to `orig_col`, `dest_col` and `flow_col`.
- **threshold**: Numeric value used to determine small flows, origins or destinations that will be grouped (lumped) together.
lump  Character string to indicate where to apply the threshold. Choose from the flow values, in migration region and/or out migration region.
other_level  Character string for the origin and/or destination label for the lumped values below the threshold. Default "other".
complete  Logical value to return a tibble with complete the origin-destination combinations
fill  Numeric value for to fill small cells below the threshold when complete = TRUE. Default of zero.
return_matrix  Logical to return a matrix. Default FALSE.
orig_col  Character string of the origin column name (when m is a data frame rather than a matrix)
dest_col  Character string of the destination column name (when m is a data frame rather than a matrix)
flow_col  Character string of the flow column name (when m is a data frame rather than a matrix)

Details

The lump argument can take values flow or bilat to apply the threshold to the data values for between region migration, in or imm to apply the threshold to the incoming region region and out or emi to apply the threshold to outgoing region region.

Value

A tibble with an additional other origins and/or destinations region based on the grouping together of small values below the threshold argument and the lump argument to indicate on where to apply the threshold.

Examples

```r
r <- LETTERS[1:4]
m <- matrix(data = c(0, 100, 30, 10, 50, 0, 50, 5, 10, 40, 0, 40, 20, 25, 20, 0),
nrow = 4, ncol = 4, dimnames = list(orig = r, dest = r), byrow = TRUE)
m

# threshold on in and out region
sum_lump(m, threshold = 100, lump = c("in", "out"))

# threshold on flows (default)
sum_lump(m, threshold = 40)

# return a matrix (only possible when input is a matrix and # complete = TRUE) with small values replaced by zeros
sum_lump(m, threshold = 50, complete = TRUE)

# return a data frame with small values replaced with zero
sum_lump(m, threshold = 80, complete = TRUE, return_matrix = FALSE)
```

## Not run:
# data frame (tidy) format
library(tidyverse)

# download Abel and Cohen (2019) estimates
f

# large 1990-1995 flow estimates
f %>%
  filter(year0 == 1990) %>%
  sum_lump(flow_col = "da_pb_closed", threshold = 1e5)

# large flow estimates for each year
f %>%
  group_by(year0) %>%
  sum_lump(flow_col = "da_pb_closed", threshold = 1e5)

## End(Not run)

### sum_net

Calculate net migration from an origin-destination migration flow matrix.

**Description**

Sums each regions flows to obtain net migration sums.

**Usage**

```r
sum_net(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
Examples

```r
r <- LETTERS[1:4]
m <- matrix(data = 1:16, nrow = 4, ncol = 4,
            dimnames = list(orig = r, dest = r))
m
sum_net(m)
```

---

**sum_od**

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`). Primarily intended to work with output from `ffs_demo`.

Usage

```r
sum_od(x = NULL, zero_diag = TRUE, add_margins = TRUE)
```

Arguments

- `x`: 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).
- `zero_diag`: Logical to indicate if to set diagonal terms to zero. Default `TRUE`.
- `add_margins`: Logical to indicate if to add row and column for immigration and emigration totals. Default `TRUE`.

Value

Matrix from summing over the first and second dimension. Set diagonals to zero.

Returns a `matrix` object of origin-destination flows.

See Also

`ffs_demo`
Summary of regional in-, out-, turnover and net-migration totals from an origin-destination migration flow matrix or data frame.

Description

Summary of regional in-, out-, turnover and net-migration totals from an origin-destination migration flow matrix or data frame.

Usage

```r
sum_region(
  m,
  drop_diagonal = TRUE,
  orig_col = "orig",
  dest_col = "dest",
  flow_col = "flow",
  international = FALSE,
  include_net = TRUE
)
```

```r
sum_country(
  m,
  drop_diagonal = TRUE,
  orig_col = "orig",
  dest_col = "dest",
  flow_col = "flow",
  include_net = TRUE,
  international = TRUE
)
```

Arguments

- `m`: A matrix or data frame of origin-destination flows. For matrix the first and second dimensions correspond to origin and destination respectively. For a data frame ensure the correct column names are passed to `orig_col`, `dest_col` and `flow_col`.
- `drop_diagonal`: Logical to indicate dropping of diagonal terms, where the origin and destination are the same, in the calculation of totals. Default TRUE.
- `orig_col`: Character string of the origin column name (when `m` is a data frame rather than a matrix)
- `dest_col`: Character string of the destination column name (when `m` is a data frame rather than a matrix)
- `flow_col`: Character string of the flow column name (when `m` is a data frame rather than a matrix)
international Logical to indicate if flows are international.
include_net Logical to indicate inclusion of a net migration total column for each region, in addition to the total in- and out-flows. Default TRUE.

Value
A tibble with total in-, out- and turnover of flows for each region.

Examples

# matrix
r <- LETTERS[1:4]
m <- matrix(data = c(0, 100, 30, 70, 0, 45, 5, 60, 0, 40, 20, 25, 20, 0),
nrow = 4, ncol = 4, dimnames = list(orig = r, dest = r), byrow = TRUE)
m
sum_region(m)

## Not run:
# data frame (tidy) format
library(tidyverse)

# download Abel and Cohen (2019) estimates
f

# single period
f %>%
  filter(year0 == 1990) %>%
  sum_country(flow_col = "da_pb_closed")

# all periods using group_by
f %>%
  group_by(year0) %>%
  sum_country(flow_col = "da_pb_closed")

## End(Not run)

uar_1960

Lifetime migration data for Governorates of United Arab Republic in 1960

Description
Lifetime migration (stock) bilateral data from Governorates of the United Arab Republic

Usage
uar_1960
**umbrella**

**Format**
Matrix with 11 rows and columns

- **orig** Governorate of birth
- **carat** Governorate of enumeration

**Source**


---

**umbrella**

*Umbrella colour scheme*

**Description**
Vector of hexadecimal codes for a umbrella rainbow colour scheme

**Usage**

`umbrella`

**Format**
An object of class character of length 9.

---

**usa_1960**

*US population totals in 1950 and 1960 by place of birth, age, sex and race*

**Description**
Population data by place of birth, age, sex and race in 1950 and 1960

**Usage**

`usa_1960`
Format

Data frame with 288 rows and 7 columns:

**birthplace**  Place of birth (US Census area)
**race**  Race from ‘white’ or ‘non-white’
**sex**  Sex from ‘male’ or ‘female’
**age_1950**  Age group in 1950
**age_1960**  Age group in 1960
**pop_1950**  Enumerated population in 1950
**pop_1960**  Enumerated population in 1960

Source

Index

* datasets
  alabama_1970, 4
  bombay_1951, 7
  dict_ims, 16
  indian_sub, 33
  ipumsi_age, 44
  italy_area, 45
  korea_dist, 46
  korea_pop, 46
  korea_reg, 47
  manila_1970, 47
  new_england_1960, 59
  rc_model_fund, 61
  rc_model_un, 61
  uar_1960, 76
  umbrella, 77
  usa_1960, 77
  alabama_1970, 4
  birth_mat, 5
  block_matrix, 5, 7, 36, 37, 39, 66
  block_sum, 6
  bombay_1951, 7
  cm2, 3, 8, 11, 34, 35
  cm3, 3, 10
  cm_net, 12
  cm_net_tot, 13
  death_mat, 15
  dict_ims, 16
  ffs, 41
  ffs(ffs_demo), 17
  ffs_demo, 3, 17, 21, 23, 43, 74
  ffs_diff, 3, 19, 20, 49, 62, 63
  ffs_rates, 3, 19, 21, 22, 23
  index_age, 23
  index_age_rc, 25
  index_connectivity, 26
  index_distance, 28
  index_impact, 29
  index_intensity, 31
  indian_sub, 33
  ipf2, 3, 9, 34, 40
  ipf2_block, 6, 7, 36
  ipf2_stripe, 37, 66
  ipf3, 3, 11, 35, 39, 43
  ipf3_qi, 3, 40, 41, 49, 62, 63
  ipf_seed, 43
  ipumsi_age, 44
  italy_area, 45
  korea_dist, 46
  korea_pop, 46
  korea_reg, 47
  manila_1970, 47
  match_pob_tot, 48
  mig_chord, 49
  mig_matrix, 52
  mig_tibble, 53
  migest (migest-package), 3
  migest-package, 3
  multi_comp, 54
  multi_comp2, 55
  nchars_wrap, 56
  net_sr, 56
  net_vs, 58
  new_england_1960, 59
  quadratic_eqn, 60
  rc_model_fund, 61
  rc_model_un, 61
  rescale_integer_sum, 62
  rescale_nb, 63
  rescale_net, 64
str_wrap_n, 66
str_wrap_n_single, 67
stripe_matrix, 6, 7, 37–39, 65
sum_bilat, 68
sum_country (sum_region), 75
sum_expand, 69
sum_lump, 71
sum_net, 73
sum_od, 74
sum_region, 75

uar_1960, 76
umbrella, 77
usa_1960, 77