

# Package ‘mcmsector’

April 15, 2026

**Type** Package

**Title** Estimating Subnational Public and Private Contraceptive Supply Shares Over Time

**Version** 1.0.2

**Author** Hannah Comiskey [aut, cre],  
David Fraizer [aut],  
Ole Maneesoonthorn [aut]

**Maintainer** Hannah Comiskey <hannahcomiskey68@gmail.com>

**Description** Engaging the private sector in contraceptive method supply is critical for equitable, sustainable, and accessible healthcare systems. This package implements Bayesian hierarchical models to estimate public and private contraceptive supply shares over time at national and subnational levels, using Demographic and Health Survey (DHS) data. Penalized splines are used to track supply shares over time, and spatial correlation structures link national and subnational estimates in data-sparse settings. For more details see Comiskey (2025) <[doi:10.48550/arXiv.2510.25153](https://doi.org/10.48550/arXiv.2510.25153)>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Depends** R (>= 4.0)

**LazyData** true

**Suggests** knitr, R2jags, rmarkdown, testthat (>= 3.0.0), mcmSupply

**Config/testthat/edition** 3

**Imports** haven, survey, stats, openxlsx, magrittr, rlang, dplyr, plyr, stringr, labelled, tibble, tidyr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2026-04-15 13:10:02 UTC

## Contents

add_index_variables . . . . .	2
build_jags_inputs . . . . .	3
build_spline_basis . . . . .	4
clean_fp_source_data . . . . .	5
compute_T_star . . . . .	6
Country_classification . . . . .	7
country_codes_DHS . . . . .	7
DEFT_DHS_database . . . . .	8
extract_mcmsector_params . . . . .	9
fix_region_names . . . . .	10
run_preprocessing_pipeline . . . . .	11
simulate_posterior_proportions . . . . .	12
standardize_country_names . . . . .	14
subnat_bivar_data . . . . .	14
summarise_posterior_proportions . . . . .	15
<b>Index</b>	<b>17</b>

---

add\_index\_variables    *Create index variables for subnational modelling*

---

### Description

Adds integer index columns for subnational area (`index_subnat`), country (`index_country`), method (`index_method`) and year (`index_year`) to facilitate modeling and JAGS data construction.

### Usage

```
add_index_variables(
  df,
  methods = c("Female Sterilization", "Implants", "Injectables", "IUD", "OC Pills"),
  all_years = seq(1990, 2030.5, by = 0.5)
)
```

### Arguments

<code>df</code>	A data frame that contains at least Country, Region, Method and average_year columns.
<code>methods</code>	Character vector of methods in the desired order (default examples given).
<code>all_years</code>	Numeric vector of years to index against (defaults to 1990–2030 by 0.5).

### Value

The input data frame augmented with index columns.

**Examples**

```
# Clean the subnational data
subnat_clean <- clean_fp_source_data(subnat_bivar_data)
# Add the region, country, method and time indices
indexed_subnat <- add_index_variables(subnat_clean)
```

---

build_jags_inputs	<i>Build JAGS metadata objects</i>
-------------------	------------------------------------

---

**Description**

From an indexed FP source data frame, build common metadata objects required for JAGS / modeling: lookups, counts of subnational units per country, match vectors used when indexing into parameter arrays, sequences of years, etc.

**Usage**

```
build_jags_inputs(df, all_years = seq(1990, 2030.5, by = 0.5))
```

**Arguments**

df	Data frame containing index columns produced by <code>add_index_variables()</code> .
all_years	Character or numeric vector of years used for indexing.

**Value**

A named list of metadata items used downstream.

**Examples**

```
# Clean the subnational data
subnat_clean <- clean_fp_source_data(subnat_bivar_data)

# Add the region, country, method and time indices
indexed_subnat <- add_index_variables(subnat_clean)

# Build JAGS inputs
meta <- build_jags_inputs(df = indexed_subnat,
  all_years = seq(1990, 2030.5, by = 0.5))
```

---

build\_spline\_basis      *Build a B-spline basis for each subnational area*

---

### Description

For each subnational area this builds a b-spline basis matrix across `all_years` using `bs_bbase_precise()` which is a small wrapper around `splines::bs()`. The returned arrays mimic the shape used in your original script.

### Usage

```
build_spline_basis(T_star, all_years = seq(1990, 2030.5, by = 0.5), nseg = 10)
```

### Arguments

<code>T_star</code>	Data frame returned by <code>compute_T_star()</code> with <code>average_year</code> and <code>index_subnat</code> .
<code>all_years</code>	Numeric vector of years for which the basis is evaluated.
<code>nseg</code>	Integer number of segments for interior knots (default 10).

### Value

A list with elements:

- `B_ik` A 3D array (`n_subnat` x `length(all_years)` x `K`) of spline basis values.
- `Kstar` Integer vector of length `n_subnat` (effective `K` for each area).
- `knots_all` Matrix of knot locations (`n_subnat` x `K`).
- `K` Number of basis columns.
- `H` `K - 1` (used for difference penalties in smoothing).

### Examples

```
# Example: Build B-spline basis
library(dplyr)

# Create a small T_star dataset (as returned by compute_T_star)
T_star <- data.frame(
  index_country = c(1, 1, 2),
  index_subnat  = c(1, 2, 3),
  average_year  = c(2005, 2010, 2008),
  index_year    = c(2, 3, 2)
)

# Define years to evaluate the spline basis
all_years <- seq(2000, 2015, by = 1)

# Build spline basis
splines_out <- build_spline_basis(
```

```

T_star    = T_star,
all_years = all_years,
nseg      = 5
)

```

---

clean\_fp\_source\_data *Clean Subnational Family Planning Source Data*

---

## Description

This function processes subnational family planning (FP) source data by filtering FP2030 countries, standardising region names, removing small-sample observations, fixing proportions, applying lemon-squeezer transformations, and cleaning standard errors (SEs) using DEFT-based imputation rules.

## Usage

```

clean_fp_source_data(
  subnat_data,
  deft_lookup = mcmsector::DEFT_DHS_database,
  fp2030_countries = c("Benin", "Burkina Faso", "Cameroon", "Cote d'Ivoire", "Ethiopia",
    "Ghana", "Guinea", "Kenya", "Liberia", "Madagascar", "Malawi", "Mali", "Mozambique",
    "Myanmar", "Nepal", "Niger", "Nigeria", "Pakistan", "Rwanda", "Senegal", "Tanzania",
    "Uganda", "Zimbabwe"),
  min_n = 20
)

```

## Arguments

subnat_data	A data frame containing raw subnational FP source data.
deft_lookup	A data frame containing DEFT values with a column named average_year.
fp2030_countries	A character vector of FP2030 countries. A default internal vector is provided.
min_n	Minimum sample size for keeping observations (default = 20).

## Details

Steps performed:

- Filters FP2030 countries.
- Removes regions "NA" and rows with insufficient sample size.
- Ensures proportions sum to 1 and replaces missing sectors with 0.
- Applies the lemon-squeezer transformation to avoid 0/1 boundaries.
- Cleans SE values
  - If one SE is 0, replaces with the mean of other sectors.
  - If all SEs missing or extremely small, uses DEFT-adjusted SE.
- Fixes region naming inconsistencies for specific countries.

**Value**

A cleaned data frame with harmonised regions, adjusted proportions, cleaned SE values, and restrictions applied.

**Examples**

```
cleaned <- clean_fp_source_data(subnat_bivar_data)
```

---

compute_T_star	<i>Compute T* (last observed year per subnational area)</i>
----------------	---

---

**Description**

For each country-region (subnational) pair, find the last observed index\_year and the associated average\_year.

**Usage**

```
compute_T_star(df)
```

**Arguments**

**df** Data frame containing at least Country, Region, index\_country, index\_subnat, average\_year and index\_year.

**Value**

A data frame with one row per subnational area containing its final observation information.

**Examples**

```
# Example: Compute T* (last observed year per subnational area)
library(dplyr)

# Create example dataset
df_idx <- data.frame(
  Country      = c("A", "A", "A", "A", "B", "B"),
  Region       = c("R1", "R1", "R2", "R2", "R1", "R1"),
  index_country = c(1, 1, 1, 1, 2, 2),
  index_subnat  = c(1, 1, 2, 2, 3, 3),
  average_year  = c(2000, 2005, 2001, 2003, 1999, 2004),
  index_year    = c(1, 2, 1, 2, 1, 2)
)

# Compute T*
T_star <- compute_T_star(df_idx)
```

---

Country\_classification

*The Country and area classification according to the United Nations Standard Statistical Division, Standard country or area codes for statistical use (M49). Adapted for use in FP2030 by the Track20 project. A subset of data from the United Nations country classifications*

---

### Description

The Country and area classification according to the United Nations Standard Statistical Division, Standard country or area codes for statistical use (M49). Adapted for use in FP2030 by the Track20 project. A subset of data from the United Nations country classifications

### Usage

Country\_classification

### Format

A data frame with 231 rows and 8 columns:

**Country or area** Country name

**ISO Code** 1, 2 & 3 number ISO country codes

**Major area** Continent

**Region** Sub-continent

**Developed region** Binary indicator for development status

**Least developed country** Binary indicator for least developed status

**Sub-Saharan Africa** Binary indicator for whether country is in Sub-Saharan Africa

**FP2020** Binary indicator for FP2020 participation status

### Source

<https://unstats.un.org/unsd/methodology/m49/>

---

country\_codes\_DHS

*Country Codes for DHS Surveys*

---

### Description

A dataset with ISO and DHS-specific country codes used in the mcmsector package.

### Usage

data(country\_codes\_DHS)

**Format**

A data frame with 91 rows and 4 variables:

**Country Name** DHS country name

**Code** 2-letter ISO country code

**India States** State codes of India

**State Name** State names of India

**Source**

"DHS Program"

---

DEFT_DHS_database	<i>DEFT_DHS_database A database of the design effects for some of the DHS surveys in the national and subnational datasets. Due to due to multistage and clustering of the DHS sample, the average standard error is increased by a design effect (DEFT) factor over that in an equivalent simple random sample.</i>
-------------------	--

---

**Description**

DEFT\_DHS\_database A database of the design effects for some of the DHS surveys in the national and subnational datasets. Due to due to multistage and clustering of the DHS sample, the average standard error is increased by a design effect (DEFT) factor over that in an equivalent simple random sample.

**Usage**

DEFT\_DHS\_database

**Format**

A dataframe of Country names, survey year and design effects for DHS surveys

**Source**

DHS final reports Appendix B, 'ESTIMATES OF SAMPLING ERRORS'.

---

 extract\_mcmsector\_params

*Extract alpha\_pms and beta.k MCMC samples from a JAGS model object*

---

## Description

Given a JAGS model object containing `BUGSoutput$sims.array`, this function extracts the posterior simulations for `alpha_pms` and `beta.k` into clean matrices suitable for downstream computation.

## Usage

```
extract_mcmsector_params(mod, n_method, n_subnat, n_beta = 13)
```

## Arguments

<code>mod</code>	A JAGS model object loaded with <code>readRDS()</code> , containing <code>mod\$BUGSoutput\$sims.array</code> .
<code>n_method</code>	Integer; number of contraceptive methods.
<code>n_subnat</code>	Integer; number of subnational regions.
<code>n_beta</code>	Integer; number of beta coefficients (typically 13).

## Value

A list with:

**alpha\_pms** A matrix of posterior samples for alpha parameters.

**beta\_k** A matrix of posterior samples for beta parameters.

## Examples

```
# Example: Extract parameters from mock JAGS output
set.seed(123)

# Create a fake sims.array
n_iter <- 10
n_chain <- 2
n_method <- 2
n_subnat <- 3
n_beta <- 2

var_names <- c(
  paste0("alpha_pms[", rep(1:n_method, each = n_subnat),
        ",", rep(1:n_subnat, times = n_method), "]"),
  paste0("beta.k[",
        rep(1:n_method, each = n_subnat * n_beta), ",",
        rep(rep(1:n_subnat, each = n_beta), times = n_method), ",",
        rep(1:n_beta, times = n_method * n_subnat), "]")
)
```

```
)  
  
sims_array <- array(  
  rnorm(n_iter * n_chain * length(var_names)),  
  dim = c(n_iter, n_chain, length(var_names)),  
  dimnames = list(NULL, NULL, var_names)  
)  
  
# Mock JAGS object  
mod <- list(BUGSoutput = list(sims.array = sims_array))  
  
# Run function  
params <- extract_mcmsector_params(  
  mod,  
  n_method = n_method,  
  n_subnat = n_subnat,  
  n_beta = n_beta  
)
```

---

fix\_region\_names

*Harmonise Region Names for Selected Countries*

---

## Description

This function standardises region names for Burkina Faso, Rwanda, Nigeria, and Cote d'Ivoire to ensure consistent naming across years.

## Usage

```
fix_region_names(df)
```

## Arguments

df                    A data frame containing Country and Region columns.

## Value

A data frame with modified region names.

## Examples

```
clean <- fix_region_names(mcmsector::subnat_bivar_data)
```

---

```
run_preprocessing_pipeline
```

*High-level preprocessing pipeline*

---

### Description

Master wrapper that runs the full preprocessing pipeline:

- standardise country names & join area classification
- add index variables
- build JAGS metadata
- compute T\_star
- build spline bases

This returns a list containing cleaned data, metadata and spline objects.

### Usage

```
run_preprocessing_pipeline(
  raw_df,
  area_classification,
  deflt_lookup = NULL,
  methods = c("Female Sterilization", "Implants", "Injectables", "IUD", "OC Pills"),
  all_years = seq(1990, 2030.5, by = 0.5),
  nseg = 10
)
```

### Arguments

raw_df	Raw FP source data frame (unindexed) (e.g. mcmsector::subnat_bivar_data) .
area_classification	A country/area classification table (e.g. mcmsector::Country_classification).
deflt_lookup	Optional DEFT lookup (passed through to SE cleaning if used earlier).
methods	Character vector of methods.
all_years	Numeric vector of years to index across.
nseg	Number of spline segments.

### Value

A named list with elements:

- data A cleaned + indexed data frame
- meta A jags metadata list
- T\_star A final-observation table
- splines A spline basis object

**Examples**

```
out <- run_preprocessing_pipeline(raw_df = subnat_bivar_data ,
                                area_classification = Country_classification)
```

---

```
simulate_posterior_proportions
```

*Simulate posterior public/private proportions from alpha and beta parameters*

---

**Description**

Computes posterior draws of the logit-scale proportions and the corresponding public/private probabilities for all methods, regions, and years.

**Usage**

```
simulate_posterior_proportions(
  alpha_pms,
  beta_k,
  B_ik,
  n_method,
  n_subnat,
  all_years,
  n_samps = 4000
)
```

**Arguments**

alpha_pms	Matrix of alpha posterior samples.
beta_k	Matrix of beta posterior samples.
B_ik	A 3D array of basis functions with dimensions: (subnational, year, n_beta).
n_method	Number of methods.
n_subnat	Number of subnational units.
all_years	Vector of years included in the model.
n_samps	Number of posterior samples to compute (default = 4000).

**Value**

A list containing:

**P** A 5D array of public/private probabilities.

**Examples**

```

set.seed(123)

# Dimensions
n_samps <- 100
n_method <- 2
n_subnat <- 3
n_years <- 5
n_beta <- 4

all_years <- 2000:(2000 + n_years - 1)

# Simulate alpha posterior samples
alpha_pms <- matrix(rnorm(n_samps * n_method * n_subnat),
                    nrow = n_samps)

colnames(alpha_pms) <- as.vector(
  outer(
    paste0("alpha_pms[", 1:n_method, ","),
    paste0(1:n_subnat, "]"),
    paste0
  )
)

# Simulate beta posterior samples
beta_k <- matrix(rnorm(n_samps * n_method * n_subnat * n_beta),
                 nrow = n_samps)

colnames(beta_k) <- unlist(
  lapply(1:n_method, function(m) {
    lapply(1:n_subnat, function(p) {
      paste0("beta.k[", m, ", ", p, ", ", 1:n_beta, "]")
    })
  })
)

# Create basis function array
B_ik <- array(runif(n_subnat * n_years * n_beta),
              dim = c(n_subnat, n_years, n_beta))

# Run simulation
P <- simulate_posterior_proportions(
  alpha_pms = alpha_pms,
  beta_k     = beta_k,
  B_ik       = B_ik,
  n_method   = n_method,
  n_subnat   = n_subnat,
  all_years  = all_years,
  n_samps    = n_samps
)

```

---

 standardize\_country\_names

*Standardise country names and attach super-region classification*


---

### Description

Harmonises a few common country name variants and joins a country-area classification table to produce a Super\_region column.

### Usage

```
standardize_country_names(
  df,
  area_classification,
  country_name_fixes = c(`Bolivia (Plurinational State of)` = "Bolivia",
    `Republic of Moldova` = "Moldova", `Viet Nam` = "Vietnam", Kyrgyzstan =
    "Kyrgyz Republic")
)
```

### Arguments

`df` A data frame with a Country column.

`area_classification` A data frame with columns Country or area and Region (e.g., Country\_classification).

`country_name_fixes` Named character vector of replacements (optional).

### Value

The input data frame with Super\_region added and country names fixed.

### Examples

```
standardize_country_names(subnat_bivar_data, Country_classification)
```

---

 subnat\_bivar\_data

*DHS survey observations for the proportion of modern contraceptives supplied by the public and private sectors at the subnational level*


---

### Description

DHS survey observations for the proportion of modern contraceptives supplied by the public and private sectors at the subnational level

**Usage**

```
subnat_bivar_data
```

**Format**

```
subnat_bivar_data:
```

A data frame with 1737 rows and 12 columns:

**Country** Country names

**Region** Regional names

**Method** Contraceptive method name

**average\_year** Average year of the survey

**year** Year of the survey

**country\_code** ISO country codes

**Public** Proportion supplied by the Public sector

**se.Public** Standard error of proportion supplied by the Public sector

**Public\_n** Sample size used to calculate proportion supplied by the Public sector

**Private** Proportion supplied by the private sector

**se.Private** Standard error of proportion supplied by the private sector

**Private\_n** Sample size used to calculate proportion supplied by the private sector

**check\_sum** Check to see if proportions sum to 1

**Source**

On request from DHS microdata - using the womens individuals recode file. Contact details found at [https://dhsprogram.com/data/dataset\\_admin/login\\_main.cfm](https://dhsprogram.com/data/dataset_admin/login_main.cfm)

---

```
summarise_posterior_proportions
```

*Summarise posterior probability samples into mean and quantiles*

---

**Description**

Converts a 5D posterior draws array P into a tidy long data frame containing posterior means and 95% credible intervals for each sector, method, region, and year.

**Usage**

```
summarise_posterior_proportions(  
  P,  
  method_index_table,  
  sector_index_table,  
  subnat_index_table,  
  year_index_table  
)
```

**Arguments**

**P** A 5D array of posterior samples with dimensions: (sample, sector, method, subnat, year).

**method\_index\_table** Tibble mapping method indices to method names.

**sector\_index\_table** Tibble mapping sector indices to sector names.

**subnat\_index\_table** Mapping subnational indices to regions/countries.

**year\_index\_table** Mapping year index to calendar year.

**Value**

A tibble with posterior mean, lower 95%, and upper 95% intervals.

**Examples**

```
# Example: Summarise posterior probabilities
library(dplyr)
library(tidyr)

# Simulate a small 5D posterior array:
# Dimensions: sample x sector x method x subnat x year
n_samps <- 10
n_sector <- 2
n_method <- 2
n_subnat <- 3
n_year <- 4

set.seed(123)
P <- array(runif(n_samps * n_sector * n_method * n_subnat * n_year),
           dim = c(n_samps, n_sector, n_method, n_subnat, n_year))

# Create minimal index tables
method_tbl <- tibble(index_method = 1:n_method, Method = c("MethodA", "MethodB"))
sector_tbl <- tibble(index_sector = 1:n_sector, SectorName = c("Public", "Private"))
subnat_tbl <- tibble(index_subnat = 1:n_subnat, Region = paste0("Region", 1:n_subnat))
year_tbl <- tibble(index_year = 1:n_year, Year = 2000 + 0:(n_year-1))

# Summarise posterior
df_summary <- summarise_posterior_proportions(
  P,
  method_index_table = method_tbl,
  sector_index_table = sector_tbl,
  subnat_index_table = subnat_tbl,
  year_index_table = year_tbl
)
```

# Index

- \* **Country\_classification**
  - Country\_classification, [7](#)
- \* **DEFT\_DHS\_database**
  - DEFT\_DHS\_database, [8](#)
- \* **datasets**
  - country\_codes\_DHS, [7](#)
  - subnat\_bivar\_data, [14](#)
- add\_index\_variables, [2](#)
- build\_jags\_inputs, [3](#)
- build\_spline\_basis, [4](#)
- clean\_fp\_source\_data, [5](#)
- compute\_T\_star, [6](#)
- Country\_classification, [7](#)
- country\_codes\_DHS, [7](#)
- DEFT\_DHS\_database, [8](#)
- extract\_mcmsector\_params, [9](#)
- fix\_region\_names, [10](#)
- run\_preprocessing\_pipeline, [11](#)
- simulate\_posterior\_proportions, [12](#)
- standardize\_country\_names, [14](#)
- subnat\_bivar\_data, [14](#)
- summarise\_posterior\_proportions, [15](#)