Package 'htaBIM'

May 8, 2026

Type Package

Title Budget Impact Modelling for Health Technology Assessment

Version 0.1.0

Description Implements a structured, reproducible framework for budget impact modelling (BIM) in health technology assessment (HTA), following the ISPOR Task Force guidelines (Sullivan et al. (2014) <doi:10.1016/j.jval.2013.08.2291> and Mauskopf et al. (2007) <doi:10.1111/j.1524-4733.2007.00187.x>). Provides functions for epidemiology-driven population estimation, market share modelling with flexible uptake dynamics, per-patient cost calculation across multiple cost categories, multi-year budget projections, payer perspective analysis, deterministic sensitivity analysis (DSA), and probabilistic sensitivity analysis (PSA) via Monte Carlo simulation. Produces submission-quality outputs including ISPOR-aligned summary tables, scenario comparison tables, per-patient cost breakdowns, tornado diagrams, PSA histograms, and text and HTML reports compatible with NICE, CADTH, and EU-HTA dossier formats. Ships with an interactive 'shiny' dashboard built on 'bslib' for point-and-click model building and exploration.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports stats, utils, graphics, rlang (>= 1.1.0)

Suggests dplyr (>= 1.1.0), tidyr (>= 1.3.0), ggplot2 (>= 3.4.0), cli (>= 3.6.0), checkmate (>= 2.3.0), glue (>= 1.7.0), purrr (>= 1.0.0), tibble (>= 3.2.0), scales (>= 1.3.0), flextable (>= 0.9.0), officer (>= 0.6.0), shiny (>= 1.8.0), bslib (>= 0.7.0), bsicons (>= 0.1.2), DT (>= 0.31), openxlsx (>= 4.2.0), knitr, rmarkdown, testthat (>= 3.0.0), spelling, covr, pkgdown

Config/Needs/website pkgdown

Config/Needs/check testthat

VignetteBuilder knitr

URL https://github.com/heorlytics/htaBIM,
https://heorlytics.github.io/htaBIM/

BugReports https://github.com/heorlytics/htaBIM/issues

Language en-GB

NeedsCompilation no

Author Shubhram Pandey [aut, cre] (ORCID:
<https://orcid.org/0009-0005-2303-1592>),
Heorlytics Ltd [cph]

Maintainer Shubhram Pandey <shubhram.pandey@heorlytics.com>

Repository CRAN

Date/Publication 2026-04-21 19:12:11 UTC

Contents

`bim_costs` *Build per-patient annual cost inputs for a budget impact model*

Description

Constructs a per-patient annual cost structure for each treatment and cost category (drug, administration, monitoring, adverse events, other). Supports optional inflation adjustment, discounting, and confidential rebates.

Usage

```
bim_costs(
  treatments,
  years = 1:5,
  drug_costs = NULL,
  admin_costs = NULL,
  monitoring_costs = NULL,
  ae_costs = NULL,
  other_costs = NULL,
  currency = "GBP",
  price_year = as.integer(format(Sys.Date(), "%Y")),
  inflation_rate = 0,
  rebates = NULL
)
```

Arguments

`treatments`	`character`. Vector of treatment names. Must match those in `bim_market_share()`.
`years`	`integer`. Projection years (default `1:5`).
`drug_costs`	Named `numeric` vector or `NULL`. Annual drug cost per patient by treatment.
`admin_costs`	Named `numeric` vector or `NULL`. Annual administration cost per patient (infusion, injection nurse, etc.).
`monitoring_costs`	Named `numeric` vector or `NULL`. Annual monitoring costs (lab tests, clinic visits, imaging).
`ae_costs`	Named `numeric` vector or `NULL`. Annual adverse event management costs per patient.

other_costs	Named numeric vector or NULL. Any other direct medical costs not captured above.
currency	character(1). ISO 4217 currency code (e.g. "GBP", "USD", "EUR", "CAD"). Default "GBP".
price_year	integer(1). Reference price year. Default is the current calendar year.
inflation_rate	numeric(1). Annual inflation rate applied to non-drug costs for years beyond Year 1. Default 0.0.
rebates	Named numeric vector or NULL. Confidential rebates as proportions (e.g. c(DrugA = 0.15) for 15% rebate). Applied to drug_costs only and kept internal (not printed by default).

Value

An object of class bim_costs, a list containing:

costs A data.frame with columns treatment, year, category, unit_cost, total_annual_cost.

total A data.frame with treatment, year, total_cost_per_patient.

params List of all input parameters (rebates stored but not printed).

meta List with currency, price_year, treatments.

See Also

bim_costs_drug(), bim_costs_ae(), bim_model()

Examples

```
costs <- bim_costs(
  treatments = c("RASi", "Sparsentan", "Sibeprenlimab"),
  currency   = "GBP",
  price_year = 2025L,
  drug_costs = c(
    RASi          = 200,
    Sparsentan    = 22000,
    Sibeprenlimab = 28500
  ),
  monitoring_costs = c(
    RASi          = 650,
    Sparsentan    = 1500,
    Sibeprenlimab = 1900
  )
)
print(costs)
```

`bim_costs_ae` *Calculate per-patient adverse event costs from AE rates and unit costs*

Description

Computes the expected annual cost of adverse event management per patient, as the sum of (AE rate ?? unit cost) across all adverse events.

Usage

```
bim_costs_ae(treatment, ae_table)
```

Arguments

`treatment`	`character(1)`. Treatment name.
`ae_table`	A `data.frame` with columns:
	`ae_name character`. Name of the adverse event.
	`rate numeric`. Probability of the AE per patient-year.
	`unit_cost numeric`. Cost per AE episode.

Value

A named `numeric` vector of length 1: expected annual AE cost per patient, suitable for use in `bim_costs()`.

Examples

```
ae_table <- data.frame(
  ae_name   = c("Injection site reaction", "Fatigue", "URTI"),
  rate      = c(0.07, 0.12, 0.09),
  unit_cost = c(180, 95, 65),
  stringsAsFactors = FALSE
)
bim_costs_ae("Sibeprenlimab", ae_table)
```

`bim_costs_drug` *Calculate per-patient drug cost from pack size and dosing schedule*

Description

Helper function to derive an annual drug cost per patient from list price, pack size, dose, and dosing frequency. Supports weight-based dosing.

Usage

```
bim_costs_drug(
  treatment,
  list_price_per_pack,
  dose_per_admin,
  admin_per_year,
  units_per_pack = 1,
  wastage_factor = 1,
  body_weight_kg = NULL
)
```

Arguments

`treatment`	`character(1)`. Treatment name.
`list_price_per_pack`	`numeric(1)`. List price per pack or vial.
`dose_per_admin`	`numeric(1)`. Dose per administration (in the units consistent with pack size).
`admin_per_year`	`numeric(1)`. Number of administrations per year.
`units_per_pack`	`numeric(1)`. Number of dose units per pack. Default `1`.
`wastage_factor`	`numeric(1)`. Factor for vial/pack wastage (e.g. `1.0` for no wastage, `1.15` for 15% wastage). Default `1.0`.
`body_weight_kg`	`numeric(1)` or `NULL`. Mean patient body weight (kg), if dosing is weight-based. Default `NULL`.

Value

A named `numeric` vector of length 1: annual drug cost per patient, suitable for use in `bim_costs()`.

Examples

```
sib_cost <- bim_costs_drug(
  treatment       = "Sibeprenlimab",
  list_price_per_pack = 2375,
  dose_per_admin  = 1,
  admin_per_year  = 12,
  units_per_pack  = 1
)
sib_cost
```

`bim_cost_breakdown` *Per-patient cost breakdown by component and treatment*

Description

Extracts and formats the per-patient annual cost decomposed by cost category (drug, admin, monitoring, adverse events, other) for each treatment in the model. This supports transparency and helps reviewers understand the drivers of differential costs between treatments.

The table is suitable for direct inclusion in HTA dossier appendices.

Usage

```
bim_cost_breakdown(model, year = NULL, currency_millions = FALSE, digits = 0L)
```

Arguments

`model`	A `bim_model` object.
`year`	`integer(1)`. Price year to extract costs for. Defaults to `model$costs$meta$price_year` (base price year, before inflation).
`currency_millions`	`logical(1)`. Express values in millions. Default `FALSE` (per-patient costs are typically in whole currency units).
`digits`	`integer(1)`. Decimal places. Default `0L`.

Value

A `data.frame` with rows = cost categories and columns = treatments, plus a **Total** row. Values are formatted character strings. Carries a `"caption"` attribute.

See Also

`bim_costs()`, `bim_costs_drug()`, `bim_costs_ae()`

Examples

```
pop <- bim_population(
  indication  = "Disease X", country = "custom",
  years = 1:5, prevalence = 0.003, n_total_pop = 42e6,
  diagnosed_rate = 0.60, treated_rate = 0.45, eligible_rate = 0.30
)
ms <- bim_market_share(
  population     = pop,
  treatments     = c("Drug C (SoC)", "Drug A (new)"),
  new_drug       = "Drug A (new)",
  shares_current = c("Drug C (SoC)" = 1.0, "Drug A (new)" = 0.0),
  shares_new     = c("Drug C (SoC)" = 0.8, "Drug A (new)" = 0.2)
)
```

```
costs <- bim_costs(
  treatments       = c("Drug C (SoC)", "Drug A (new)"),
  drug_costs       = c("Drug C (SoC)" = 500,  "Drug A (new)" = 25000),
  monitoring_costs = c("Drug C (SoC)" = 200,  "Drug A (new)" = 1500),
  ae_costs         = c("Drug C (SoC)" = 50,   "Drug A (new)" = 300)
)
model <- bim_model(pop, ms, costs)
bim_cost_breakdown(model)
```

`bim_example` *Example budget impact model inputs: Disease X*

Description

A named list containing example inputs for a hypothetical budget impact model for a new treatment (Drug A) in a chronic condition (Disease X), for use in vignettes, examples, and testing. All values are illustrative only and do not represent any real drug, price, or epidemiological estimate.

Usage

```
bim_example
```

Format

A named `list` with three elements:

`population_params` A `list` of arguments for `bim_population()`.

`market_share_params` A `list` of arguments for `bim_market_share()` (excluding `population`).

`cost_params` A `list` of arguments for `bim_costs()`.

Source

Illustrative values only. Not based on any real submission data.

Examples

```
data(bim_example)
str(bim_example)

# Reconstruct the full model
pop   <- do.call(bim_population, bim_example$population_params)
ms    <- do.call(bim_market_share,
                 c(list(population = pop), bim_example$market_share_params))
costs <- do.call(bim_costs, bim_example$cost_params)
model <- bim_model(pop, ms, costs)
summary(model)
```

`bim_export_population` *Export a population funnel to a formatted Excel workbook*

Description

Writes a formatted `.xlsx` workbook with three sheets:

1. **Annual Funnel** – the full year-by-year patient count table, as a styled Excel table.
2. **Parameters** – every model input documented in a two-column table.
3. **Funnel Snapshot** – a single-year view with percentage retained at each stage and percentage of total population (highlighted Eligible row).

Requires the **openxlsx** package.

Usage

```
bim_export_population(
  pop,
  file = "population_funnel.xlsx",
  snapshot_year = 1L,
  overwrite = FALSE
)
```

Arguments

`pop`	A `bim_population` object from `bim_population()`.
`file`	`character(1)`. Output file path including the `.xlsx` extension. The parent directory must already exist.
`snapshot_year`	`integer(1)` or `NULL`. Year used for the `"Funnel Snapshot"` sheet. Pass `NULL` to omit that sheet. Default `1L`.
`overwrite`	`logical(1)`. If `FALSE` (default) and `file` already exists, an error is raised. Set `TRUE` to silently overwrite.

Value

The file path `file`, invisibly. A message is printed on success.

See Also

`bim_population()`, `bim_plot_funnel()`

Examples

```
pop <- bim_population(
  indication     = "Disease X",
  country        = "GB",
  years          = 1:5,
  prevalence     = 0.003,
  n_total_pop    = 42e6,
  diagnosed_rate = 0.60,
  treated_rate   = 0.45,
  eligible_rate  = 0.30,
  extra_filters  = list(second_line_plus = 0.55, biomarker_positive = 0.40)
)
bim_export_population(pop, file = file.path(tempdir(), "population_funnel.xlsx"))
```

---

bim_extract *Extract tidy results from a budget impact model*

Description

Returns a tidy `data.frame` of budget impact results from a `bim_model()` object, optionally filtered to a specific level of aggregation.

Usage

```
bim_extract(model, level = c("annual", "cumulative"), scenario = "all")
```

Arguments

`model`	A `bim_model` object.
`level`	`character(1)`. Level of aggregation: • `"annual"` – annual budget impact by year and scenario (default). • `"cumulative"` – cumulative totals by scenario.
`scenario`	`character` or `"all"`. Scenarios to include. Default `"all"`.

Value

A `data.frame`.

Examples

```
pop <- bim_population(
  indication    = "Example",
  country       = "GB",
  years         = 1:3,
  prevalence    = 0.003,
  n_total_pop   = 42e6,
```

```
  eligible_rate = 0.30
)
ms <- bim_market_share(
  population     = pop,
  treatments     = c("RASi", "NewDrug"),
  new_drug       = "NewDrug",
  shares_current = c(RASi = 1.0, NewDrug = 0.0),
  shares_new     = c(RASi = 0.8, NewDrug = 0.2)
)
costs <- bim_costs(
  treatments = c("RASi", "NewDrug"),
  drug_costs = c(RASi = 500, NewDrug = 25000)
)
model <- bim_model(pop, ms, costs)
bim_extract(model, level = "annual")
bim_extract(model, level = "cumulative")
```

`bim_market_share` *Specify treatment market shares for a budget impact model*

Description

Defines how treatment market shares evolve over time, both in the current scenario (without the new drug) and in the new scenario (with the new drug introduced). Supports constant, linear ramp, logistic S-curve, and step uptake dynamics.

Usage

```
bim_market_share(
  population,
  treatments,
  new_drug,
  shares_current,
  shares_new,
  dynamics = c("constant", "linear", "logistic", "step"),
  uptake_params = NULL,
  scenarios = NULL
)
```

Arguments

`population`	A `bim_population` object from `bim_population()`.
`treatments`	`character`. Vector of all treatment names, including the new drug.
`new_drug`	`character(1)`. Name of the new intervention. Must be an element of `treatments`.
`shares_current`	Named `numeric` vector. Market shares in the current scenario (without the new drug). Values must sum to 1 and all be in `[0, 1]`. Names must match `treatments`.

shares_new	Named numeric vector. Market shares in the new scenario (with the new drug at full uptake). Values must sum to 1. Names must match treatments.
dynamics	character(1). How the new drug's uptake evolves: • "constant" – shares_new apply uniformly in all years (default). • "linear" – new drug ramps linearly from 0 to target share over uptake_params$ramp_years years. • "logistic" – S-curve uptake. Requires uptake_params$year_50pct and optionally uptake_params$steepness. • "step" – shares_new is a named list with one vector per year.
uptake_params	list or NULL. Parameters controlling uptake dynamics: • For "linear": list(ramp_years = 3). • For "logistic": list(year_50pct = 2, steepness = 2).
scenarios	Named list or NULL. Alternative market share vectors (named numerics, same structure as shares_new) for scenario analysis. E.g. list(conservative = c(...), optimistic = c(...)).

Value

An object of class bim_market_share, a list containing:

shares A data.frame with columns year, treatment, scenario, share, n_patients.

params List of input parameters.

meta List with treatments, new_drug, dynamics.

See Also

bim_population(), bim_model()

Examples

```
pop <- bim_population(
  indication  = "Disease X",
  country     = "GB",
  years       = 1:5,
  prevalence  = 0.003,
  n_total_pop = 42e6,
  eligible_rate = 0.30
)

ms <- bim_market_share(
  population     = pop,
  treatments     = c("RASi", "Sparsentan", "Sibeprenlimab"),
  new_drug       = "Sibeprenlimab",
  shares_current = c(RASi = 0.75, Sparsentan = 0.25, Sibeprenlimab = 0.00),
  shares_new     = c(RASi = 0.60, Sparsentan = 0.20, Sibeprenlimab = 0.20),
  dynamics       = "linear",
  uptake_params  = list(ramp_years = 3)
)
print(ms)
```

`bim_model` *Assemble and run a budget impact model*

Description

Combines a `bim_population()`, `bim_market_share()`, and `bim_costs()` object into a complete budget impact model and computes the annual and cumulative budget impact across all scenarios.

The budget impact for year *t* is defined as:

$$BI_t = \sum_i N_t \cdot s_i^{new}(t) \cdot c_i(t) - \sum_i N_t \cdot s_i^{current}(t) \cdot c_i(t)$$

where N_t is the number of eligible patients, $s_i(t)$ is the market share of treatment i, and $c_i(t)$ is the cost per patient for treatment i.

Usage

```
bim_model(
  population,
  market_share,
  costs,
  payer = bim_payer_default(),
  discount_rate = 0,
  label = NULL
)
```

Arguments

`population`	A `bim_population` object from `bim_population()`.
`market_share`	A `bim_market_share` object from `bim_market_share()`.
`costs`	A `bim_costs` object from `bim_costs()`.
`payer`	A `bim_payer` object from `bim_payer()` or one of the pre-built payer functions. Default is `bim_payer_default()`.
`discount_rate`	`numeric(1)`. Annual discount rate applied to Year 2+ costs. Per ISPOR guidelines, the base case should be undiscounted (`0`). Default `0`.
`label`	`character(1)` or `NULL`. Optional model label for reporting.

Value

An object of class `bim_model`, a list containing:

`population` The input `bim_population` object.

`market_share` The input `bim_market_share` object.

`costs` The input `bim_costs` object.

`payer` The input `bim_payer` object.

`results` A list with `annual` and `cumulative` data frames.

`meta` A list with model metadata.

See Also

bim_population(), bim_market_share(), bim_costs(), bim_payer(), plot.bim_model(), bim_table()

Examples

```
pop <- bim_population(
  indication    = "Disease X",
  country       = "GB",
  years         = 1:3,
  prevalence    = 0.003,
  n_total_pop   = 42e6,
  eligible_rate = 0.30
)

ms <- bim_market_share(
  population     = pop,
  treatments     = c("RASi", "Sparsentan", "Sibeprenlimab"),
  new_drug       = "Sibeprenlimab",
  shares_current = c(RASi = 0.75, Sparsentan = 0.25, Sibeprenlimab = 0.00),
  shares_new     = c(RASi = 0.60, Sparsentan = 0.20, Sibeprenlimab = 0.20)
)

costs <- bim_costs(
  treatments = c("RASi", "Sparsentan", "Sibeprenlimab"),
  drug_costs = c(RASi = 200, Sparsentan = 22000, Sibeprenlimab = 28500)
)

model <- bim_model(pop, ms, costs, label = "IgAN BIM")
summary(model)
```

---

bim_payer *Define a payer perspective for a budget impact model*

Description

Specifies which costs are borne by the budget holder and the coverage fraction applied to drug costs. Pre-built payer functions cover the most common HTA settings.

Usage

```
bim_payer(
  name,
  perspective = c("healthcare_system", "payer", "societal"),
  cost_coverage = 1,
  description = NULL
)
```

Arguments

name	character(1). Descriptive payer name (e.g. "NHS England").
perspective	character(1). One of "healthcare_system", "payer", or "societal". Informational; affects reporting only.
cost_coverage	numeric(1). Proportion of costs covered by this payer. Must be in [0, 1]. Default 1.0 (100%).
description	character(1) or NULL. Optional free-text description appended to outputs.

Value

An object of class bim_payer.

See Also

bim_payer_nhs(), bim_payer_default(), bim_model()

Examples

```
p <- bim_payer(
  name         = "NHS England",
  perspective  = "healthcare_system",
  cost_coverage = 1.0
)
print(p)
```

---

bim_payer_cadth *CADTH Canadian public payer perspective*

Description

Returns a bim_payer() representing the Canadian Drug Review (CDR) public payer perspective used in CADTH submissions.

Usage

```
bim_payer_cadth()
```

Value

A bim_payer object.

bim_payer_default *Default payer perspective (healthcare system, 100% coverage)*

Description

Returns a `bim_payer()` representing a generic healthcare system perspective with full cost coverage. Used as the default in `bim_model()`.

Usage

```
bim_payer_default()
```

Value

A `bim_payer` object.

bim_payer_nhs *NHS England payer perspective*

Description

Returns a `bim_payer()` representing the NHS England perspective, appropriate for NICE Technology Appraisal submissions.

Usage

```
bim_payer_nhs()
```

Value

A `bim_payer` object.

bim_payer_us_commercial *US commercial payer perspective*

Description

Returns a `bim_payer()` representing a US commercial insurer perspective.

Usage

```
bim_payer_us_commercial()
```

Value

A `bim_payer` object.

bim_plot_bar *Grouped bar chart of annual budget impact by year*

Description

Displays the annual budget impact as grouped bars, with one group per year and one bar per scenario.

Usage

```
bim_plot_bar(model, currency_millions = TRUE, colours = NULL, title = NULL)
```

Arguments

model	A bim_model object.
currency_millions	logical(1). Default TRUE.
colours	character or NULL. Bar colours per scenario.
title	character(1) or NULL.

Value

Called for side effects. Returns invisibly.

bim_plot_cost_breakdown *Plot per-patient cost breakdown as a stacked bar chart*

Description

Draws a stacked horizontal bar chart of per-patient annual costs by cost component, with one bar per treatment. Useful for visually comparing the cost structure across treatments.

Usage

```
bim_plot_cost_breakdown(model, year = NULL, colours = NULL, title = NULL)
```

Arguments

model	A bim_model object.
year	integer(1) or NULL. Year to plot. Defaults to first available year in costs.
colours	Named character vector of colours per cost category. Defaults use the htaBIM colour palette.
title	character(1) or NULL. Plot title.

Value

Called for side effects. Returns invisibly.

`bim_plot_funnel` *Plot the epidemiological patient funnel*

Description

Draws a publication-quality visualisation of the sequential patient-filtering steps from total population down to the eligible patients, using a `bim_population` object. Two display modes are available:

- `"funnel"` – classic centred bars that narrow at each stage, giving the traditional funnel appearance used in HTA dossiers.
- `"flowchart"` – labelled boxes connected by downward arrows, useful for slide decks and regulatory submissions.

Percentage-retained annotations can be toggled on or off.

Usage

```
bim_plot_funnel(
  pop,
  year = 1L,
  type = c("funnel", "flowchart"),
  show_pct = TRUE,
  title = NULL,
  palette = c("#2166AC", "#AEC6E8")
)
```

Arguments

`pop`	A `bim_population` object from `bim_population()`.
`year`	`integer(1)`. The projection year to display. Must be present in `pop$annual$year`. Default `1L`.
`type`	`character(1)`. `"funnel"` (default) or `"flowchart"`.
`show_pct`	`logical(1)`. If `TRUE` (default), annotates each transition arrow / gap with the percentage of the previous stage retained.
`title`	`character(1)` or `NULL`. Plot title. A sensible default is constructed from the indication and year when `NULL`.
`palette`	`character(2)`. Two hex colours used for the fill gradient: the first is applied to the widest (Total Population) bar / box, and the second to the narrowest (Eligible) stage. Default `c("#2166AC", "#AEC6E8")`.

Value

A `ggplot2` object, returned invisibly. The plot is printed as a side effect.

See Also

bim_population(), bim_export_population()

Examples

```
pop <- bim_population(
  indication     = "Disease X",
  country        = "GB",
  years          = 1:5,
  prevalence     = 0.003,
  n_total_pop    = 42e6,
  diagnosed_rate = 0.60,
  treated_rate   = 0.45,
  eligible_rate  = 0.30,
  extra_filters  = list(second_line_plus = 0.55, biomarker_positive = 0.40)
)
bim_plot_funnel(pop)
bim_plot_funnel(pop, type = "flowchart")
bim_plot_funnel(pop, show_pct = FALSE, palette = c("#1B7837", "#A6DBA0"))
```

---

bim_plot_line *Line plot of annual budget impact over time*

Description

Plots the annual budget impact (or cumulative budget impact) over the projection horizon, with one line per scenario.

Usage

```
bim_plot_line(
  model,
  cumulative = FALSE,
  scenario = "base",
  currency_millions = TRUE,
  colours = NULL,
  title = NULL
)
```

Arguments

model	A bim_model object.
cumulative	logical(1). If TRUE, plot cumulative impact. Default FALSE.
scenario	character. Scenarios to plot. Default "base".
currency_millions	logical(1). Express values in millions. Default TRUE.

colours	Named character vector. Line colours by scenario name. Defaults use the htaBIM colour palette.
title	character(1) or NULL. Plot title.

Value

Called for side effects (plot). Returns invisibly.

`bim_plot_psa` *Plot PSA results*

Description

Produces a histogram of simulated budget impacts with the base-case value and 95 \

Usage

```
bim_plot_psa(
  psa,
  currency_millions = TRUE,
  title = NULL,
  col_bar = "#AEC6E8",
  col_base = "#1a3a5c",
  col_ci = "#D6604D"
)
```

Arguments

psa	A bim_psa object from bim_run_psa().
currency_millions	logical(1). Express values in millions. Default TRUE.
title	character(1) or NULL. Plot title.
col_bar	character(1). Histogram bar fill colour. Default light blue.
col_base	character(1). Colour for base-case line. Default dark blue.
col_ci	character(1). Colour for credible interval lines. Default orange-red.

Value

Called for side effects. Returns invisibly.

bim_plot_shares *Market share stacked bar chart*

Description

Displays market shares as stacked bars – one panel for the current scenario (without new drug) and one for the new drug scenario – across years.

Usage

```
bim_plot_shares(model, scenario = "base", colours = NULL, title = NULL)
```

Arguments

model	A bim_model object.
scenario	character(1). New drug scenario. Default "base".
colours	character or NULL. Named vector of colours by treatment.
title	character(1) or NULL. Plot title.

Value

Called for side effects. Returns invisibly.

bim_plot_tornado *Tornado diagram for DSA results*

Description

Draws a horizontal tornado plot from the output of bim_run_dsa(), showing the range of budget impact for each parameter varied.

Usage

```
bim_plot_tornado(
  dsa,
  top_n = 10L,
  currency = "GBP",
  currency_millions = TRUE,
  title = NULL,
  col_low = "#2171B5",
  col_high = "#CB181D"
)
```

Arguments

dsa	A bim_dsa data frame from bim_run_dsa().
top_n	integer(1). Maximum number of parameters to show. Default 10L.
currency	character(1). Currency label for x-axis. Default "GBP".
currency_millions	logical(1). Default TRUE.
title	character(1) or NULL. Plot title.
col_low	character(1). Bar colour for low values. Default blue.
col_high	character(1). Bar colour for high values. Default red.

Value

Called for side effects. Returns invisibly.

---

bim_population — *Estimate the annual eligible patient population for a budget impact model*

Description

Builds a year-by-year estimate of the number of patients eligible for a new treatment, using an epidemiology-driven funnel approach aligned with ISPOR Task Force guidelines (Sullivan et al., 2014). Supports prevalent, incident, or combined population approaches.

Usage

```
bim_population(
  indication,
  country = "GB",
  years = 1:5,
  prevalence = NULL,
  incidence = NULL,
  n_total_pop = NULL,
  diagnosed_rate = 1,
  treated_rate = 1,
  eligible_rate = 1,
  growth_rate = 0,
  approach = c("prevalent", "incident", "both"),
  extra_filters = list(),
  eligible_fn = NULL,
  data_source = NULL
)
```

Arguments

indication	character(1). Name of the disease or indication. Used in outputs and reports.
country	character(1). ISO 3166-1 alpha-2 country code (e.g. "GB", "US", "CA", "DE"). Used to look up built-in population data if n_total_pop is NULL. Use "custom" to rely solely on n_total_pop.
years	integer. Vector of projection years (e.g. 1:5). Default is 1:5.
prevalence	numeric(1) or NULL. Point prevalence as a proportion (e.g. 0.002 for 0.2%). Required when approach is "prevalent" or "both".
incidence	numeric(1) or NULL. Annual incidence rate per 100,000. Required when approach is "incident" or "both".
n_total_pop	numeric(1) or NULL. Total reference population size. If NULL and country is recognised, uses built-in population data.
diagnosed_rate	numeric(1). Proportion of prevalent/incident cases that are diagnosed. Must be in [0, 1]. Default 1.0.
treated_rate	numeric(1). Proportion of diagnosed patients receiving any systemic treatment. Must be in [0, 1]. Default 1.0.
eligible_rate	numeric(1). Proportion of treated patients eligible for the new drug (e.g. meeting label criteria). Must be in [0, 1]. Default 1.0.
growth_rate	numeric(1). Annual growth rate applied to the total population (e.g. 0.005 for 0.5% per year). Default 0.0.
approach	character(1). Population approach: "prevalent" (stock population), "incident" (new cases per year), or "both" (sum of prevalent and incident). Default "prevalent".
extra_filters	list or NULL. Optional named list of additional filtering proportions applied sequentially **after** eligible_rate. Each element must be a single numeric in [0, 1]. The names become column labels in the output (e.g. list(prior_therapy_failure = 0.6, biomarker_positive = 0.4)). Ignored if eligible_fn is supplied.
eligible_fn	function or NULL. Optional custom function that **replaces** the entire treated_rate * eligible_rate * extra_filters calculation. Must accept three arguments: n_diagnosed (numeric vector, one value per year), n_treated (numeric vector), and params (the list of all scalar inputs). Must return a numeric vector of the same length. Use this when your eligibility logic is not simple sequential multiplication (e.g. additive components, minimum/cap rules, lookup tables). See the examples below.
data_source	character(1) or NULL. Citation for the epidemiology data, appended to outputs. Optional.

Value

An object of class bim_population, which is a list containing:

annual A data.frame with columns year, n_total_pop, n_prevalent_or_incident, n_diagnosed, n_treated, and n_eligible. When extra_filters are supplied, one additional column per filter step is inserted before n_eligible, showing the intermediate patient count at each stage.

params A list of all input parameters.

meta A list with indication, country, approach, data_source.

References

Sullivan SD, Mauskopf JA, Augustovski F et al. (2014). Budget impact analysis–principles of good practice: report of the ISPOR 2012 Budget Impact Analysis Good Practice II Task Force. *Value Health*, 17(1):5-14. doi:10.1016/j.jval.2013.08.2291

See Also

`bim_market_share()`, `bim_costs()`, `bim_model()`

Examples

```
# Standard funnel
pop <- bim_population(
  indication     = "Disease X",
  country        = "GB",
  years          = 1:5,
  prevalence     = 0.003,
  n_total_pop    = 42e6,
  diagnosed_rate = 0.60,
  treated_rate   = 0.45,
  eligible_rate  = 0.30
)
print(pop)
summary(pop)

# Extra filters: add line-of-therapy and biomarker criteria after eligible_rate
pop2 <- bim_population(
  indication     = "Disease X",
  country        = "GB",
  years          = 1:5,
  prevalence     = 0.003,
  n_total_pop    = 42e6,
  diagnosed_rate = 0.60,
  treated_rate   = 0.45,
  eligible_rate  = 0.30,
  extra_filters  = list(second_line_plus = 0.55, biomarker_positive = 0.40)
)
pop2$annual

# Custom formula: eligible = max of two additive subgroups, capped at treated
pop3 <- bim_population(
  indication     = "Disease X",
  country        = "GB",
  years          = 1:5,
  prevalence     = 0.003,
  n_total_pop    = 42e6,
  diagnosed_rate = 0.60,
  treated_rate   = 0.45,
  eligible_fn    = function(n_diagnosed, n_treated, params) {
    subgroup_a <- n_diagnosed * 0.20
    subgroup_b <- n_diagnosed * 0.15
    pmin(subgroup_a + subgroup_b, n_treated)
```

```
  }
)
```

`bim_report` *Generate a budget impact model report*

Description

Produces a structured text summary report of a budget impact model, written to a file or returned as a character vector. For Word (.docx) or HTML output, install the suggested packages `officer` and `rmarkdown`.

Usage

```
bim_report(
  model,
  output_file = NULL,
  format = c("text", "html", "docx"),
  title = NULL,
  author = NULL,
  date = Sys.Date(),
  scenario = "base"
)
```

Arguments

`model`	A `bim_model` object.
`output_file`	`character(1)` or `NULL`. File path for the output report. If `NULL`, returns the report as a character vector (default).
`format`	`character(1)`. Output format: `"text"` (default), `"html"`, or `"docx"`. For `"html"` and `"docx"`, the `rmarkdown` and `officer` packages must be installed.
`title`	`character(1)` or `NULL`. Report title. Defaults to the model label.
`author`	`character(1)` or `NULL`. Author name for the report header.
`date`	`Date` or `character(1)`. Report date. Default `Sys.Date()`.
`scenario`	`character(1)`. Scenario to report. Default `"base"`.

Value

If `output_file` is `NULL`, a character vector of report lines. Otherwise, the path to the written file, invisibly.

Examples

```
pop <- bim_population(
  indication    = "Disease X",
  country       = "GB",
  years         = 1:3,
  prevalence    = 0.003,
  n_total_pop   = 42e6,
  eligible_rate = 0.30
)
ms <- bim_market_share(
  population     = pop,
  treatments     = c("RASi", "NewDrug"),
  new_drug       = "NewDrug",
  shares_current = c(RASi = 1.0, NewDrug = 0.0),
  shares_new     = c(RASi = 0.8, NewDrug = 0.2)
)
costs <- bim_costs(
  treatments = c("RASi", "NewDrug"),
  drug_costs = c(RASi = 500, NewDrug = 25000)
)
model <- bim_model(pop, ms, costs)

# Return as character vector
rpt <- bim_report(model)
cat(rpt, sep = "\n")
```

`bim_run_dsa` *Run a deterministic sensitivity analysis on a budget impact model*

Description

Executes a one-way deterministic sensitivity analysis (DSA) by varying each parameter in a `bim_sensitivity_spec()` individually across its low/high range while holding all others at their base values.

Usage

```
bim_run_dsa(model, sensitivity, year = NULL, scenario = "base")
```

Arguments

`model`	A `bim_model` object.
`sensitivity`	A `bim_sensitivity_spec` object from `bim_sensitivity_spec()`.
`year`	`integer(1)`. The projection year on which DSA results are evaluated. Default is the final year in the model.
`scenario`	`character(1)`. Which scenario to use as base case. Default `"base"`.

Value

A data.frame with columns parameter, label, low_value, high_value, bi_low, bi_base, bi_high, range, sorted by range descending (largest impact first). Can be passed directly to bim_plot_tornado().

See Also

bim_sensitivity_spec(), bim_plot_tornado()

Examples

```
pop <- bim_population(
  indication    = "Example",
  country       = "GB",
  years         = 1:3,
  prevalence    = 0.003,
  n_total_pop   = 42e6,
  eligible_rate = 0.30
)
ms <- bim_market_share(
  population     = pop,
  treatments     = c("RASi", "NewDrug"),
  new_drug       = "NewDrug",
  shares_current = c(RASi = 1.0, NewDrug = 0.0),
  shares_new     = c(RASi = 0.8, NewDrug = 0.2)
)
costs <- bim_costs(
  treatments = c("RASi", "NewDrug"),
  drug_costs = c(RASi = 500, NewDrug = 25000)
)
model <- bim_model(pop, ms, costs)

sens <- bim_sensitivity_spec(
  prevalence_range        = c(0.002, 0.005),
  eligible_rate_range     = c(0.20, 0.45),
  drug_cost_multiplier_range = c(0.85, 1.15)
)
dsa <- bim_run_dsa(model, sens, year = 3L)
print(dsa)
```

---

bim_run_psa *Run a probabilistic sensitivity analysis (PSA)*

Description

Performs a Monte Carlo PSA by repeatedly sampling uncertain parameters from their assumed statistical distributions and re-running the budget impact model for each draw. This produces a distribution of budget impact outcomes that reflects joint parameter uncertainty.

Distributional assumptions

- Prevalence, diagnosed rate, treated rate, eligible rate – **Beta** distribution parameterised from the base-case value and a standard error.
- Drug cost – **LogNormal** distribution parameterised from the base-case value and a coefficient of variation (CV).

Usage

```
bim_run_psa(
  model,
  n_sim = 1000L,
  prevalence_se = NULL,
  diagnosed_rate_se = NULL,
  treated_rate_se = NULL,
  eligible_rate_se = NULL,
  cost_cv = NULL,
  year = NULL,
  scenario = "base",
  seed = NULL
)

## S3 method for class 'bim_psa'
print(x, ...)
```

Arguments

`model`	A `bim_model` object (base case).
`n_sim`	`integer(1)`. Number of Monte Carlo simulations. Default `1000L`.
`prevalence_se`	`numeric(1)`. Standard error for prevalence. If `NULL` (default), prevalence is held fixed.
`diagnosed_rate_se`	`numeric(1)` or `NULL`. SE for diagnosed rate.
`treated_rate_se`	`numeric(1)` or `NULL`. SE for treated rate.
`eligible_rate_se`	`numeric(1)` or `NULL`. SE for eligible rate.
`cost_cv`	`numeric(1)` or `NULL`. Coefficient of variation applied to all drug costs simultaneously. If `NULL`, costs are held fixed.
`year`	`integer(1)`. Budget impact year to summarise. Defaults to the last year in the model.
`scenario`	`character(1)`. Scenario to use. Default `"base"`.
`seed`	`integer(1)` or `NULL`. Random seed for reproducibility.
`x`	A `bim_psa` object returned by `bim_run_psa()`.
`...`	Further arguments (ignored).

Value

An object of class `bim_psa`: a list with elements:

`simulations` `data.frame` with one row per simulation: `sim`, `budget_impact`, and the sampled parameter values.

`summary` `data.frame` with mean, SD, median, and 95 \ interval of budget impact.

`year` The year summarised.

`scenario` The scenario used.

`n_sim` Number of simulations run.

`base_bi` Base-case budget impact for reference.

`print.bim_psa` invisibly returns `x`. Called for its side effect of printing a formatted PSA summary (mean, median, SD, 95% credible interval) to the console.

See Also

`bim_plot_psa()`, `bim_run_dsa()`

Examples

```
pop <- bim_population(
  indication  = "Disease X", country = "custom",
  years = 1:5, prevalence = 0.003, n_total_pop = 42e6,
  diagnosed_rate = 0.60, treated_rate = 0.45, eligible_rate = 0.30
)
ms <- bim_market_share(
  population     = pop,
  treatments     = c("Drug C (SoC)", "Drug A (new)"),
  new_drug       = "Drug A (new)",
  shares_current = c("Drug C (SoC)" = 1.0, "Drug A (new)" = 0.0),
  shares_new     = c("Drug C (SoC)" = 0.8, "Drug A (new)" = 0.2)
)
costs <- bim_costs(
  treatments = c("Drug C (SoC)", "Drug A (new)"),
  drug_costs = c("Drug C (SoC)" = 500, "Drug A (new)" = 25000)
)
model <- bim_model(pop, ms, costs)

set.seed(1)
psa <- bim_run_psa(model, n_sim = 200L, prevalence_se = 0.0005,
                   eligible_rate_se = 0.05, cost_cv = 0.10)
print(psa)
```

`bim_scenario_table` *Cross-scenario budget impact comparison table*

Description

Produces a side-by-side summary table of budget impact results across all scenarios in a model, showing Year 1, mid-point, final year, and cumulative totals. This is the standard tabular format for dossier submissions following ISPOR Task Force guidelines.

Usage

```
bim_scenario_table(model, years = NULL, currency_millions = TRUE, digits = 2L)
```

Arguments

`model`	A `bim_model` object.
`years`	`integer` vector. Years to include as columns. If `NULL` (default), uses Year 1, the middle year, and the last year.
`currency_millions`	`logical(1)`. Express values in millions. Default `TRUE`.
`digits`	`integer(1)`. Decimal places for formatted values. Default `2L`.

Value

A `data.frame` with one row per scenario and columns for each selected year plus cumulative total, formatted as character strings. The `data.frame` carries a `"caption"` attribute suitable for passing to `knitr::kable()`.

See Also

`bim_table()`, `bim_extract()`

Examples

```
pop <- bim_population(
  indication  = "Disease X", country = "custom",
  years = 1:5, prevalence = 0.003, n_total_pop = 42e6,
  diagnosed_rate = 0.60, treated_rate = 0.45, eligible_rate = 0.30
)
ms <- bim_market_share(
  population     = pop,
  treatments     = c("Drug C (SoC)", "Drug A (new)"),
  new_drug       = "Drug A (new)",
  shares_current = c("Drug C (SoC)" = 1.0, "Drug A (new)" = 0.0),
  shares_new     = c("Drug C (SoC)" = 0.8, "Drug A (new)" = 0.2),
  scenarios      = list(
    conservative = c("Drug C (SoC)" = 0.9, "Drug A (new)" = 0.1),
    optimistic   = c("Drug C (SoC)" = 0.7, "Drug A (new)" = 0.3)
```

```
  )
)
costs <- bim_costs(
  treatments = c("Drug C (SoC)", "Drug A (new)"),
  drug_costs = c("Drug C (SoC)" = 500, "Drug A (new)" = 25000)
)
model <- bim_model(pop, ms, costs)
st <- bim_scenario_table(model)
print(st)
```

`bim_sensitivity_spec` *Specify a deterministic sensitivity analysis for a budget impact model*

Description

Defines parameter ranges for a deterministic sensitivity analysis (DSA) on a `bim_model()`. Each parameter is varied individually from its low to high value while all others are held at their base case value.

Usage

```
bim_sensitivity_spec(
  prevalence_range = NULL,
  diagnosed_rate_range = NULL,
  treated_rate_range = NULL,
  eligible_rate_range = NULL,
  new_drug_share_range = NULL,
  drug_cost_multiplier_range = c(0.85, 1.15),
  extra_params = NULL
)
```

Arguments

`prevalence_range`
: `numeric(2)` or `NULL`. Low and high values for disease prevalence (proportion).

`diagnosed_rate_range`
: `numeric(2)` or `NULL`. Low and high values for diagnosed rate.

`treated_rate_range`
: `numeric(2)` or `NULL`. Low and high values for treated rate.

`eligible_rate_range`
: `numeric(2)` or `NULL`. Low and high values for eligible rate.

`new_drug_share_range`
: `numeric(2)` or `NULL`. Low and high values for new drug market share (applied uniformly across years).

`drug_cost_multiplier_range`
: `numeric(2)` or `NULL`. Low and high multipliers applied to the new drug cost (e.g. `c(0.85, 1.15)` for plus/minus 15%). Default `c(0.85, 1.15)`.

extra_params	Named list or NULL. Additional parameter ranges as named elements, each a list(label, base, low, high).

Value

An object of class bim_sensitivity_spec.

See Also

bim_run_dsa(), bim_model()

Examples

```
sens <- bim_sensitivity_spec(
  prevalence_range        = c(0.002, 0.005),
  eligible_rate_range     = c(0.20, 0.45),
  new_drug_share_range    = c(0.10, 0.30),
  drug_cost_multiplier_range = c(0.85, 1.15)
)
```

---

bim_table *Generate a formatted budget impact summary table*

Description

Produces a formatted HTML or plain-text summary table of annual and/or cumulative budget impact, suitable for inclusion in RMarkdown reports or HTA dossiers.

Usage

```
bim_table(
  model,
  format = c("both", "annual", "cumulative"),
  scenario = "base",
  digits = 0L,
  caption = NULL,
  footnote = NULL
)
```

Arguments

model	A bim_model object.
format	character(1). Table format: "annual", "cumulative", or "both". Default "both".
scenario	character(1). Scenario to display. Default "base".
digits	integer(1). Rounding digits. Default 0.
caption	character(1) or NULL. Table caption.
footnote	character(1) or NULL. Table footnote.

Value

A `data.frame` formatted for display.

Examples

```
pop <- bim_population(
  indication    = "Example",
  country       = "GB",
  years         = 1:3,
  prevalence    = 0.003,
  n_total_pop   = 42e6,
  eligible_rate = 0.30
)
ms <- bim_market_share(
  population     = pop,
  treatments     = c("RASi", "NewDrug"),
  new_drug       = "NewDrug",
  shares_current = c(RASi = 1.0, NewDrug = 0.0),
  shares_new     = c(RASi = 0.8, NewDrug = 0.2)
)
costs <- bim_costs(
  treatments = c("RASi", "NewDrug"),
  drug_costs = c(RASi = 500, NewDrug = 25000)
)
model <- bim_model(pop, ms, costs)
tab <- bim_table(model)
print(tab)
```

---

launch_shiny *Launch the htaBIM interactive Shiny dashboard*

Description

Opens the `htaBIM` interactive budget impact modelling dashboard in the default web browser. Requires the `shiny` package to be installed.

Usage

```
launch_shiny(...)
```

Arguments

`...`	Additional arguments passed to `shiny::runApp()`.

Value

Called for its side effect (launches a Shiny app). Returns invisibly.

Examples

```
if (interactive() && requireNamespace("shiny", quietly = TRUE)) {
  launch_shiny()
}
```

`plot.bim_model` *Plot a budget impact model*

Description

Dispatcher for the various `htaBIM` plot types. Calls the appropriate plotting function based on the `type` argument.

Usage

```
## S3 method for class 'bim_model'
plot(x, type = c("line", "bar", "tornado", "shares"), ...)
```

Arguments

`x`	A `bim_model` object.
`type`	`character(1)`. Plot type: • `"line"` – annual budget impact over time (default). • `"bar"` – grouped bar chart by year and scenario. • `"tornado"` – DSA tornado diagram (requires `dsa` argument). • `"shares"` – market share stacked bar chart.
`...`	Additional arguments passed to the specific plot function.

Value

Called for side effects (plot). Returns `x` invisibly.

See Also

`bim_plot_line()`, `bim_plot_bar()`, `bim_plot_tornado()`, `bim_plot_shares()`

Examples

```
pop <- bim_population(
  indication    = "Disease X",
  country       = "GB",
  years         = 1:5,
  prevalence    = 0.003,
  n_total_pop   = 42e6,
  eligible_rate = 0.30
)
```

```
ms <- bim_market_share(
  population     = pop,
  treatments     = c("RASi", "Sparsentan", "Sibeprenlimab"),
  new_drug       = "Sibeprenlimab",
  shares_current = c(RASi = 0.75, Sparsentan = 0.25, Sibeprenlimab = 0.00),
  shares_new     = c(RASi = 0.60, Sparsentan = 0.20, Sibeprenlimab = 0.20)
)
costs <- bim_costs(
  treatments = c("RASi", "Sparsentan", "Sibeprenlimab"),
  drug_costs = c(RASi = 200, Sparsentan = 22000, Sibeprenlimab = 28500)
)
model <- bim_model(pop, ms, costs)
plot(model, type = "line")
plot(model, type = "bar")
plot(model, type = "shares")
```

`print.bim_costs` *Print method for bim_costs*

Description

Print method for bim_costs

Usage

```
## S3 method for class 'bim_costs'
print(x, ...)
```

Arguments

`x`	A `bim_costs` object.
`...`	Further arguments (ignored).

Value

Invisibly returns `x`. Called for its side effect of printing a formatted summary of the cost inputs to the console.

print.bim_dsa *Print method for bim_dsa*

Description

Print method for bim_dsa

Usage

```
## S3 method for class 'bim_dsa'
print(x, ...)
```

Arguments

x	A bim_dsa object.
...	Further arguments (ignored).

Value

Invisibly returns x. Called for its side effect of printing a formatted summary of the deterministic sensitivity analysis results to the console.

print.bim_market_share *Print method for bim_market_share*

Description

Print method for bim_market_share

Usage

```
## S3 method for class 'bim_market_share'
print(x, ...)
```

Arguments

x	A bim_market_share object.
...	Further arguments (ignored).

Value

Invisibly returns x. Called for its side effect of printing a formatted summary of the market share inputs to the console.

print.bim_model *Print method for bim_model*

Description

Print method for bim_model

Usage

```
## S3 method for class 'bim_model'
print(x, ...)
```

Arguments

x	A bim_model object.
...	Further arguments (ignored).

Value

Invisibly returns x. Called for its side effect of printing a formatted summary of the budget impact model results to the console.

print.bim_payer *Print method for bim_payer*

Description

Print method for bim_payer

Usage

```
## S3 method for class 'bim_payer'
print(x, ...)
```

Arguments

x	A bim_payer object.
...	Further arguments (ignored).

Value

Invisibly returns x. Called for its side effect of printing a formatted summary of the payer perspective settings to the console.

print.bim_population *Print method for bim_population*

Description

Print method for bim_population

Usage

```
## S3 method for class 'bim_population'
print(x, ...)
```

Arguments

x	A bim_population object.
...	Further arguments (ignored).

Value

Invisibly returns x. Called for its side effect of printing a formatted summary of the patient population inputs to the console.

summary.bim_model *Summary method for bim_model*

Description

Summary method for bim_model

Usage

```
## S3 method for class 'bim_model'
summary(object, digits = 0L, ...)
```

Arguments

object	A bim_model object.
digits	integer(1). Decimal digits for currency amounts. Default 0.
...	Further arguments (ignored).

Value

The bim_model object, invisibly.

summary.bim_population *Summary method for bim_population*

Description

Summary method for bim_population

Usage

```
## S3 method for class 'bim_population'
summary(object, ...)
```

Arguments

object	A bim_population object.
...	Further arguments (ignored).

Value

The bim_population object, invisibly.

Index