

Package: CRIO (via r-universe)

October 9, 2026

Type Package

Title Risk and Gap Analysis for the CGIAR Research Investment Opportunities Report

Version 1.1.0

Description Reads IMPACT model scenario results (via GFSir) and builds the risk and gap analysis used in the Research Investment Opportunities (RIO) report: hunger, diet, nutrient adequacy and blue water indicators against targets; pathway analysis across socioeconomic and climate scenarios; maps, target trajectories and comparisons of Breeding For Tomorrow (B4T) scenarios against the baseline.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports collapse (>= 2.0.10), data.table (>= 1.14.8), dplyr (>= 1.1.4), forcats (>= 1.0.1), gamstransfer (>= 3.0.3), GFSir (>= 5.1.3), ggplot2 (>= 4.0.3), ggpubr (>= 0.6.0), maps (>= 3.4.2), openxlsx2 (>= 1.9), RColorBrewer (>= 1.1.3), rlang (>= 1.1.6), rpart, rpart.plot (>= 3.1.5), scales (>= 1.4.0), sf (>= 1.1.0), stats, stringr (>= 1.6.0), terra (>= 1.9.1), tibble (>= 3.2.1), tidyr (>= 1.3.1), tidyterra (>= 0.6.1)

Additional_repositories <https://ifpri.github.io/drat>

Repository <https://ifpri.r-universe.dev>

Date/Publication 2026-10-09 01:20:47 UTC

RemoteUrl <https://github.com/IFPRI/CRIO>

RemoteRef HEAD

RemoteSha cda2b478a7f3178e7ee46c0d24f69c7d60d85c3c

Contents

add_scenario_cols	3
aquatic_nutrient_per_kg	4
b4t_levels	4
b4t_regions	5
build_b4t_comparison	5
build_b4t_comparisons	6
build_pathway_analysis	6
cg_lookup	7
climate_counterpart	8
crio_baseline_id	8
crio_plot_all	9
default_pathways	11
exclude_cmdty	11
extract_rules	12
fat_kcal_share	12
filter_climate	13
food_groups	13
get_ader	14
get_calories	14
get_fat_share	15
get_foodavail	15
get_frtveg	16
get_hunger	16
get_mder	17
get_minkcal	17
get_nutrient_adequacy	18
get_sugar_share	19
get_water	19
indicator_specs	20
list_scenarios	20
load_crio_data	21
load_region_map	22
nutrient_labels	22
nutrient_quantity_specs	23
own_cg_region	23
plot_b4t_gap_closed	24
plot_b4t_gap_maps	25
plot_b4t_heatmap	26
plot_gap_overview	27
plot_indicator_maps	28
plot_pct_hits	29
plot_region_tree	29
plot_rules_heatmap	30
plot_spec_trajectories	31
plot_target_trajectories	31
read_scenarios	33

add_scenario_cols 3

relabel_scenarios	33
rni_lookup	34
row_countries	34
save_crio_data	34
scenario_id	35
select_scenarios	35
summarise_rules	36
world_outline	36

Index 37

<i>add_scenario_cols</i>	<i>Add scenario descriptor columns</i>
--------------------------	--

Description

GFSir parses the 4th id segment into co2, so B4T suffixes land there. Everything is derived from id, ssp, gcm and rcp instead.

Usage

```
add_scenario_cols(  
  df,  
  pathways = default_pathways,  
  low_warming_gcms = c("GFDL", "MPI", "MRI"),  
  lower_emission_rcps = "2.6"  
)
```

Arguments

<i>df</i>	Data with id, ssp, gcm and rcp columns.
<i>pathways</i>	Table of ssp, rcp and pathway names (default default_pathways).
<i>low_warming_gcms</i>	GCMs classed as "Low warming".
<i>lower_emission_rcps</i>	RCPs classed as "Lower emission".

Details

Adds:

b4t "None", the CG region where B4T is implemented, or "All CG regions".

ensemble "Climate" or "B4T".

pathway Name of the SSP x RCP combination from pathways (e.g. "Green world"), or "SSPx-RCPy" if not listed.

warming "Lower emission" for *lower_emission_rcps*; otherwise "Low warming" / "High warming" by GCM; "No climate change" for NoCC.

and resets *co2* / *co2_fert* (all runs are without CO2 fertilization).

Value

df with the added columns.

aquatic_nutrient_per_kg

Placeholder nutrient values per kg for aquatic foods

Description

Aquatic commodities are missing from `Nutrients_per_kg.gdx`. These are rough, globally generic approximations (same value for every country), informed by FAO/INFOODS uFiSh ranges and USDA food composition data for representative species in each IMPACT aquatic category.

Usage

aquatic_nutrient_per_kg

Format

A tibble with columns `cmdty`, `Nutrients` and `value`.

Details

Caveats: not country-specific; rough averages across highly variable species groups (small whole fish vs. large fillet fish differ by an order of magnitude for vitamin A, iron, zinc, folate). Treat as a placeholder and flag as a known limitation. Units match `Nutrients` codes, per kg edible portion, raw.

b4t_levels

Factor levels for the B4T scenario column

Description

The six CG regions plus "All CG regions" (the CG6 scenario).

Usage

b4t_levels

Format

A character vector.

b4t_regions	<i>CGIAR regions where Breeding For Tomorrow (B4T) is implemented</i>
-------------	---

Description

CGIAR regions where Breeding For Tomorrow (B4T) is implemented

Usage

b4t_regions

Format

A character vector of CG region codes.

build_b4t_comparison	<i>Compare B4T scenarios with their climate counterfactual</i>
----------------------	--

Description

Each B4T run against the climate run with the same SSP, GCM and RCP ([climate_counterpart\(\)](#)), e.g. SSP3-MRI-370-WCA vs SSP3-MRI-370-379) in target_year, so B4T runs under any scenario are handled. gap = distance from target (0 if met); gap_closed = share of the counterfactual's gap closed by B4T (NA if it already meets the target).

Usage

```
build_b4t_comparison(
  df,
  value_col,
  threshold = NULL,
  threshold_col = NULL,
  direction = c("below", "above"),
  target_year = 2050
)
```

Arguments

df	Indicator data with scenario columns (climate + B4T runs).
value_col	Column with the indicator value.
threshold, threshold_col	Fixed or dynamic target.
direction	"below" or "above" the target is good.
target_year	Year compared.

Value

One row per region x B4T run with counterfactual (base_id, base_*) and B4T values, absolute and percent change, gaps, gap closed and target status change.

build_b4t_comparisons *B4T comparisons for a list of indicator specs*

Description

B4T comparisons for a list of indicator specs

Usage

```
build_b4t_comparisons(specs, ...)
```

Arguments

specs	Indicator specs (see indicator_specs()).
...	Further arguments to build_b4t_comparison() .

Value

All comparisons stacked, with an indicator column.

build_pathway_analysis
Pathway analysis: which scenarios hit a target

Description

For each region, the share of scenarios hitting the target in target_year, and a classification tree (rpart) on the scenario descriptors for regions where some but not all scenarios hit it. Run on the climate ensemble ([filter_climate\(\)](#)).

Usage

```
build_pathway_analysis(
  df,
  value_col,
  threshold = NULL,
  threshold_col = NULL,
  direction = c("below", "above"),
  target_year = 2050,
  predictors = c("pathway", "warming")
)
```

Arguments

df	Indicator data with scenario columns.
value_col	Column with the indicator value.
threshold	Fixed target value.
threshold_col	Column with a region/year-specific target (dynamic target, e.g. "minKCAL" or "ader"). Use instead of threshold.
direction	"below": hitting means value < target; "above": value > target.
target_year	Year evaluated.
predictors	Scenario columns used by the trees.

Value

A list with df_year, summary_tbl, partial_regions, tree_results and rules_tbl.

cg_lookup	<i>Country to CG region lookup</i>
-----------	------------------------------------

Description

Read from the regions sheet of a GFSir mapping file, which is long: one row per (cty, region) membership. Cached after the first call.

Usage

```
cg_lookup(mapping = "mappingCG6.xlsx")
```

Arguments

mapping	GFSir mapping file name.
---------	--------------------------

Value

A tibble with columns cty and cg_region.

climate_counterpart	<i>Climate run matching a scenario</i>
---------------------	--

Description

The climate (non-B4T) run with the same SSP, GCM and RCP, e.g. SSP3-MRI-370-WCA -> SSP3-MRI-370-379.
Used to compare each B4T run with its own counterfactual.

Usage

climate_counterpart(id)

Arguments

id Scenario id(s).

Value

Scenario id(s) of the matching climate run.

crio_baseline_id	<i>Default baseline scenario</i>
------------------	----------------------------------

Description

SSP3, MRI climate model, RCP 7.0, no CO2 fertilization.

Usage

crio_baseline_id

Format

A character string.

crio_plot_all	<i>Make all RIO figures</i>
---------------	-----------------------------

Description

Runs every figure type for a set of indicators and writes them to `plotdir`. Each figure type and each indicator can be switched on or off, and a failing figure is reported and skipped instead of stopping the run.

Usage

```
crio_plot_all(
  fulldata,
  plotdir,
  plots = c("maps", "pct_hits", "rules", "trajectories", "trajectories_countries",
    "quantities", "overview", "b4t_heatmap", "b4t_gap"),
  indicators = NULL,
  nutrients = TRUE,
  regions = c(b4t_regions, "WLD"),
  countries = NULL,
  ctymap = NULL,
  specs = NULL,
  nut_threshold = 1.5,
  water_threshold = 1,
  keep_above = -0.1,
  b4t_map_scenario = "All CG regions",
  b4t_show_regions = c("all", "own"),
  b4t_show_countries = c("own", "all"),
  baseline_id = crio_baseline_id,
  target_year = 2050,
  subdirs = TRUE,
  format = "png"
)
```

Arguments

<code>fulldata</code>	Output of <code>load_crio_data()</code> .
<code>plotdir</code>	Output folder (created if needed).
<code>plots</code>	Figure types to make (see Details). Default: all.
<code>indicators</code>	Indicator names from <code>indicator_specs()</code> to include (e.g. <code>c("HNGR", "FRTVEG")</code>). NULL = all.
<code>nutrients</code>	Include the nutrient adequacy indicators when <code>indicators</code> is NULL.
<code>regions</code>	Region codes for regional figures.
<code>countries</code>	Country codes for country figures. NULL skips them.
<code>ctymap</code>	Region map from <code>load_region_map()</code> ; needed for maps.

specs	Indicator specs. Default <code>indicator_specs(fulldata, nut_threshold, water_threshold)</code> .
nut_threshold, water_threshold	Passed to <code>indicator_specs()</code> and <code>nutrient_quantity_specs()</code> .
keep_above	keep_above for the nutrient overviews (see <code>plot_gap_overview()</code>).
b4t_map_scenario	B4T run mapped in the b4t_gap maps.
b4t_show_regions, b4t_show_countries	B4T lines in trajectory figures for regions and countries: "all" = all 7 B4T runs; "own" = B4T in the panel's own CG region + all CG regions. Defaults: "all" for regions, "own" for countries.
baseline_id	Baseline scenario id.
target_year	Year evaluated for maps, pathway analysis and B4T comparisons.
subdirs	Write each figure type to its own subfolder of plotdir.
format	File extension for figures ("png", "pdf", "svg", ...).

Details

Figure types (plots):

maps 4-panel indicator maps (climate ensemble). Needs `ctymap`.

pct_hits Bar chart of percent of climate scenarios hitting the target.

rules Heatmap of decision-tree rules across regions.

trajectories Trajectories for regions (all B4T runs).

trajectories_countries Trajectories for countries (own-region B4T).

quantities Nutrient quantity trajectories (RNI and target lines), for regions and, if given, countries.

overview Distance-to-target overviews (main indicators and nutrients), for regions and, if given, countries.

b4t_heatmap Region x B4T heatmaps (absolute and percent change).

b4t_gap Gap closed by B4T (bars) and gap maps for `b4t_map_scenario`.

Value

Invisibly, a list with analyses (pathway analyses by indicator), b4t (all B4T comparisons) and failed (figures that errored, with the message).

default_pathways	<i>Default pathway names by SSP and RCP</i>
------------------	---

Description

Combinations not listed here are named automatically as "SSPx-RCPy". Pass your own table to [add_scenario_cols\(\)](#) (via [load_crio_data\(\)](#)) to rename or add pathways.

Usage

```
default_pathways
```

Format

A tibble with columns ssp, rcp (as parsed by GFSir, e.g. "2.6", "7.0") and pathway.

exclude_cmdty	<i>Commodities excluded from nutrient availability</i>
---------------	--

Description

Raw/intermediate commodities that would double count, plus palm kernel (food availability only in IDN and MYS, and small).

Usage

```
exclude_cmdty
```

Format

A character vector of IMPACT demand commodity codes.

extract_rules	<i>Extract rules from a classification tree</i>
---------------	---

Description

Extract rules from a classification tree

Usage

```
extract_rules(tree, region_name)
```

Arguments

tree	An rpart tree.
region_name	Region code to tag the rules with.

Value

A data frame with region, outcome ("hits"/"misses"), rule, cover.

fat_kcal_share	<i>Share of calories from fat by commodity</i>
----------------	--

Description

Fraction of a commodity's calories that come from fat (not per 100 g). Used to approximate the fat share of dietary energy.

Usage

```
fat_kcal_share
```

Format

A tibble with columns name, fat_kcal_share and notes.

filter_climate	<i>Keep the climate ensemble only</i>
----------------	---------------------------------------

Description

Drops B4T runs, leaving the SSP x GCM x RCP runs (including the baseline).

Usage

```
filter_climate(df)
```

Arguments

df Data with an ensemble column (see [add_scenario_cols\(\)](#)).

Value

Filtered df.

food_groups	<i>Food groups used for calorie and water totals</i>
-------------	--

Description

Food groups used for calorie and water totals

Usage

```
food_groups
```

Format

A character vector of GFSir food group codes.

get_ader	<i>Total calories against the average dietary energy requirement (ADER)</i>
----------	---

Description

ADER comes from FAOSTAT Food Security and is extended to future years with a linear trend per country.

Usage

```
get_ader(calories, faostat_csv, sets_xlsx, years = 2025:2050)
```

Arguments

calories	Output of get_calories() .
faostat_csv	Path to FAOSTAT Food_Security_Data_E_All_Data_(Normalized).csv.
sets_xlsx	Path to the IMPACT Sets.xlsx (FAO to IMPACT country codes).
years	Years to extrapolate ADER to.

Value

Total calories with ader and hits_target.

get_calories	<i>Per capita calories by commodity, with totals and shares</i>
--------------	---

Description

Adds a Total row over [food_groups](#) and each row's share of the total. GRL is dropped (no demand except fish).

Usage

```
get_calories(files, mapping = "mappingCG6.xlsx")
```

Arguments

files	Scenario GDX files (see list_scenarios()).
mapping	GFSir mapping file name.

Value

Calories (kcal/person/day).

get_fat_share	<i>Approximate share of calories from fat</i>
---------------	---

Description

Approximate share of calories from fat

Usage

```
get_fat_share(calories, fat_shares = fat_kcal_share)
```

Arguments

calories	Output of get_calories() .
fat_shares	Fat share of calories by commodity (default fat_kcal_share).

Value

One row per region x year x scenario with fat_energy_share.

get_foodavail	<i>Food availability (all commodities and food groups)</i>
---------------	--

Description

Food availability (all commodities and food groups)

Usage

```
get_foodavail(files, mapping = "mappingCG6.xlsx")
```

Arguments

files	Scenario GDX files (see list_scenarios()).
mapping	GFSir mapping file name.

Value

GFSir foodavail output for all scenarios (no scenario columns).

get_frtveg	<i>Fruit and vegetable availability (g/person/day)</i>
------------	--

Description

Fruit and vegetable availability (g/person/day)

Usage

```
get_frtveg(foodavail)
```

Arguments

foodavail Output of [get_foodavail\(\)](#).

Value

F&V availability.

get_hunger	<i>Population at risk of hunger</i>
------------	-------------------------------------

Description

Hunger is only solved in 5-year steps; GFSir's group5 merges it onto annual population, so the in-between years come back as 0. Only years with hunger results are kept.

Usage

```
get_hunger(files, mapping = "mappingCG6.xlsx")
```

Arguments

files Scenario GDX files (see [list_scenarios\(\)](#)).
mapping GFSir mapping file name.

Details

Note: the hunger module floors the share at risk at 1 percent, scaled to each country's FAO 2020 numbers, so countries can flatten at a country-specific minimum (e.g. ~3.3 percent for CIV).

Value

Hunger data with value (million people) and share.

get_mder	<i>Total calories against the minimum dietary energy requirement (MDER)</i>
----------	---

Description

Total calories against the minimum dietary energy requirement (MDER)

Usage

```
get_mder(calories, minkcal)
```

Arguments

calories	Output of <code>get_calories()</code> .
minkcal	Output of <code>get_minkcal()</code> .

Value

Total calories with minKCAL and hits_target.

get_minkcal	<i>Minimum calorie requirement by country</i>
-------------	---

Description

Minimum calorie requirement by country

Usage

```
get_minkcal(gdx)
```

Arguments

gdx	A scenario GDX file containing MinKCAL (e.g. the baseline).
-----	---

Value

A data frame with region and minKCAL.

get_nutrient_adequacy *Nutrient availability and adequacy ratios*

Description

Food availability (kg/person/day) x nutrient content per kg, summed over commodities, divided by the RNI. Nutrient contents come from the Hunger model's `Nutrients_per_kg.gdx` (country-specific) plus `aquatic_nutrient_per_kg` placeholders for aquatic foods.

Usage

```
get_nutrient_adequacy(
    foodavail,
    nutrients_gdx,
    mapping = "mappingCG6.xlsx",
    rni = rni_lookup,
    aquatic = aquatic_nutrient_per_kg,
    exclude = exclude_cmdty
)
```

Arguments

<code>foodavail</code>	Output of <code>get_foodavail()</code> .
<code>nutrients_gdx</code>	Path to <code>Nutrients_per_kg.gdx</code> .
<code>mapping</code>	GFSir mapping file name (commodity mapping from its crops sheet).
<code>rni</code>	RNI table (default <code>rni_lookup</code>).
<code>aquatic</code>	Aquatic nutrient values (default <code>aquatic_nutrient_per_kg</code>).
<code>exclude</code>	Commodities to exclude (default <code>exclude_cmdty</code>).

Details

This is nutrient supply, not intake.

Value

One row per region x year x scenario x nutrient with `total_nutrient` (per capita per day), `RNI` and `adequacy_ratio`.

get_sugar_share	<i>Share of calories from sugar</i>
-----------------	-------------------------------------

Description

Share of calories from sugar

Usage

```
get_sugar_share(calories, sugar_name = "O&S-Sugar (p)")
```

Arguments

calories	Output of get_calories() .
sugar_name	Commodity name for refined sugar.

Value

One row per region x year x scenario with sugar_energy_share.

get_water	<i>Blue water use, with totals, shares and ratio to 2025</i>
-----------	--

Description

Countries with total blue water below min_volume in the baseline in 2025 are dropped. ratio2025 is each value over its own 2025 value.

Usage

```
get_water(
  files,
  mapping = "mappingCG6.xlsx",
  baseline_id = crio_baseline_id,
  min_volume = 0.01
)
```

Arguments

files	Scenario GDX files (see list_scenarios()).
mapping	GFSir mapping file name.
baseline_id	Baseline scenario id.
min_volume	Minimum 2025 baseline total (cubic km) to keep a country.

Value

Blue water data (cubic km).

indicator_specs	<i>Indicator specifications</i>
-----------------	---------------------------------

Description

One entry per indicator with its data, target and labels. Used to drive trajectory figures, distance-to-target overviews and B4T comparisons in a loop.

Usage

```
indicator_specs(fulldata, nut_threshold = 1.5, water_threshold = 1)
```

Arguments

fulldata	Output of <code>load_crio_data()</code> .
nut_threshold	Nutrient adequacy target as a multiple of the RNI.
water_threshold	Target for blue water use relative to 2025.

Details

Each entry has: df, value_col, threshold or threshold_col, direction ("below" or "above" the target is good), label, optional short_label, labeller (axis labels) and delta_labeller (absolute change vs baseline; shares in percentage points).

Value

A named list of specs. The names of the non-nutrient indicators are stored in `attr(, "main")`.

list_scenarios	<i>List scenario GDX files</i>
----------------	--------------------------------

Description

Climate runs are named SSP-GCM-RCP-379 (no CO2 fertilization), e.g. SSP1-GFDL-126-379 or SSP3-MRI-370-379. B4T runs replace the last segment with the CG region where B4T is implemented (CWANA, ESA, ..., CG6 = all CG regions). Any SSP (1-5), GCM and RCP is picked up; you can also pass your own vector of files to the readers instead.

Usage

```
list_scenarios(scendir, include_nocc = FALSE)
```

Arguments

- scendir Folder with scenario GDX files.
- include_nocc Keep NoCC (no climate change) runs. Default FALSE.

Value

Character vector of full file paths.

load_crio_data	<i>Build all RIO indicator datasets</i>
----------------	---

Description

Build all RIO indicator datasets

Usage

```
load_crio_data(  
  files,  
  nutrients_gdx,  
  faostat_csv,  
  sets_xlsx,  
  mapping = "mappingCG6.xlsx",  
  baseline_id = crio_baseline_id  
)
```

Arguments

- files Scenario GDX files (see [list_scenarios\(\)](#)).
- nutrients_gdx Path to Nutrients_per_kg.gdx.
- faostat_csv Path to the FAOSTAT Food Security csv (for ADER).
- sets_xlsx Path to IMPACT Sets.xlsx (for ADER).
- mapping GFSir mapping file name.
- baseline_id Baseline scenario id; its GDX supplies MinKCAL.

Value

A named list: hunger, fruits_and_vegetables, calories, fats, sugar, MDER, ADER, RNI, WATER, minkcal.

load_region_map	<i>Load the IMPACT region shapefile</i>
-----------------	---

Description

Load the IMPACT region shapefile

Usage

```
load_region_map(shp)
```

Arguments

shp	Path to the IMPACT regions shapefile (with a NEW_REGION column).
-----	--

Value

A SpatVector.

nutrient_labels	<i>Display names for nutrient codes</i>
-----------------	---

Description

Display names for nutrient codes

Usage

```
nutrient_labels
```

Format

A named character vector, names are Nutrients codes.

`nutrient_quantity_specs`*Nutrient quantity specifications*

Description

Nutrient availability in physical units (per capita per day) instead of adequacy ratios. The target is `nut_threshold` x RNI; the RNI itself is added as a reference line (`ref_value`).

Usage

```
nutrient_quantity_specs(fulldata, nut_threshold = 1.5)
```

Arguments

`fulldata` Output of `load_crio_data()`.
`nut_threshold` Nutrient adequacy target as a multiple of the RNI.

Value

A named list of specs (names are nutrient codes), with `title`, `ref_value` and `ref_label` set.

`own_cg_region`*CG region a region code belongs to*

Description

CG region a region code belongs to

Usage

```
own_cg_region(r, mapping = "mappingCG6.xlsx")
```

Arguments

`r` Region codes (countries and/or CG regions).
`mapping` GFSir mapping file name.

Value

The code itself for CG regions, the CG region for countries, NA for anything else (WLD, DVD/DVG, countries outside CG regions).

plot_b4t_gap_closed	<i>Share of the baseline gap closed by each B4T run</i>
---------------------	---

Description

Bar chart, one panel per region.

Usage

```
plot_b4t_gap_closed(
  cmp,
  regions = c(b4t_regions, "WLD"),
  indicator_label = "target",
  baseline_id = crio_baseline_id,
  title = NULL,
  save = FALSE,
  outfile = NULL,
  plotdir = ".",
  width = 10,
  height = 6
)
```

Arguments

cmp	Output of build_b4t_comparison() .
regions	Region codes (panels).
indicator_label	Indicator name for the y axis and title.
baseline_id	Climate run whose B4T runs are shown.
title	Plot title.
save, outfile, plotdir, width, height	Saving options.

Value

A ggplot.

plot_b4t_gap_maps	<i>Maps of the target gap: baseline vs one B4T run</i>
-------------------	--

Description

(A) baseline gap; (B) gap under the B4T run; (C) change in value vs baseline; (D) share of the baseline gap closed.

Usage

```
plot_b4t_gap_maps(
  cmp,
  ctymap,
  b4t_scenario = "All CG regions",
  baseline_id = crio_baseline_id,
  direction = c("below", "above"),
  gap_labeller = scales::label_number(),
  delta_labeller = scales::label_number(),
  indicator_label = "indicator",
  target_year = 2050,
  save = FALSE,
  outfile = NULL,
  plotdir = ".",
  width = 10,
  height = 6
)
```

Arguments

cmp	Output of build_b4t_comparison() .
ctymap	Region map from load_region_map() .
b4t_scenario	B4T run to map (a CG region or "All CG regions").
baseline_id	Climate run whose B4T runs are shown.
direction	"below" or "above" the target is good (sets colours of C).
gap_labeller	Label function for gaps.
delta_labeller	Label function for the change in value.
indicator_label	Indicator name for titles.
target_year	Year (titles).
save, outfile, plotdir, width, height	Saving options.

Value

A ggarrange object.

plot_b4t_heatmap	<i>Heatmap of B4T effects: region x where B4T is implemented</i>
------------------	--

Description

Fill = percent change vs baseline; tile text = absolute change (top) and percent change (bottom).
Own-region effects sit on the diagonal.

Usage

```
plot_b4t_heatmap(
  cmp,
  regions = c(b4t_regions, "WLD"),
  direction = c("below", "above"),
  delta_labeller = scales::label_number(accuracy = 0.01, style_positive = "plus"),
  pct_labeller = scales::percent_format(accuracy = 0.01, style_positive = "plus"),
  baseline_id = crio_baseline_id,
  title = NULL,
  save = FALSE,
  outfile = NULL,
  plotdir = ".",
  width = 8,
  height = 5.5
)
```

Arguments

cmp	Output of <code>build_b4t_comparison()</code> .
regions	Region codes (rows).
direction	"below" or "above" the target is good (sets colours).
delta_labeller, pct_labeller	Label functions for absolute and percent change.
baseline_id	Climate run whose B4T runs are shown (B4T runs are compared with this run).
title	Plot title.
save, outfile, plotdir, width, height	Saving options.

Value

A ggplot.

plot_gap_overview	<i>Distance-to-target overview across indicators</i>
-------------------	--

Description

Puts all indicators on one scale: $\text{gap} = (\text{value} - \text{target}) / \text{target}$ for "below" targets and $(\text{target} - \text{value}) / \text{target}$ for "above" targets, so a positive gap is short of target and zero or negative is target met. Lines are the baseline; markers are B4T in the panel's own CG region (open circle) and in all CG regions (cross). WLD and countries outside CG regions have no own-region run.

Usage

```
plot_gap_overview(
  specs,
  regions = c(b4t_regions, "WLD"),
  keep_above = NULL,
  target_year = 2050,
  baseline_id = crio_baseline_id,
  mapping = "mappingCG6.xlsx",
  title = "Distance to target",
  save = FALSE,
  outfile = NULL,
  plotdir = ".",
  width = 12,
  height = 8
)
```

Arguments

specs	Indicator specs (see indicator_specs()).
regions	Region codes: CG regions and/or countries.
keep_above	If set, keep only region x indicator lines whose baseline gap in target_year is above this value (e.g. -0.1 = short of target or within 10 percent of it). Drops indicators far above target that squash the axis.
target_year	Year used by keep_above.
baseline_id	Baseline scenario id.
mapping	GFSir mapping file (for country -> CG region).
title	Plot title.
save, outfile, plotdir, width, height	Saving options.

Value

A ggplot.

plot_indicator_maps *Four-panel indicator maps*

Description

(A) baseline value in baseline_year; (B) mean across scenarios in target_year; (C) baseline distance from target in target_year; (D) percent of scenarios hitting the target. Rest-of-world countries ([row_countries](#)) are greyed out.

Usage

```
plot_indicator_maps(
  df,
  summary_tbl,
  ctymap,
  value_col,
  baseline_id = crio_baseline_id,
  threshold = NULL,
  threshold_col = NULL,
  direction = c("below", "above"),
  indicator_label,
  value_labeller = scales::percent_format(accuracy = 1),
  target_year = 2050,
  baseline_year = 2025,
  save = FALSE,
  outfile = NULL,
  plotdir = ".",
  width = 10,
  height = 6
)
```

Arguments

df	Indicator data (normally the climate ensemble, filter_climate()).
summary_tbl	summary_tbl from build_pathway_analysis() .
ctymap	Region map from load_region_map() .
value_col	Column with the indicator value.
baseline_id	Baseline scenario id.
threshold, threshold_col	Fixed or dynamic target (see build_pathway_analysis()).
direction	"below" or "above" the target is good.
indicator_label	Label for titles and legends.
value_labeller	Label function for values.

target_year, baseline_year
Years for panels.
save, outfile, plotdir, width, height
Saving options.

Details

With a fixed threshold the value scales diverge at the target; with a dynamic threshold_col they use a simple gradient.

Value

A ggarrange object.

plot_pct_hits	Bar chart of the share of scenarios hitting a target
---------------	--

Description

Regions where some but not all scenarios hit the target.

Usage

plot_pct_hits(analysis, title = "% scenarios hitting target")

Arguments

analysis Output of [build_pathway_analysis\(\)](#).
title Plot title.

Value

A ggplot.

plot_region_tree	Plot the classification tree for one region
------------------	---

Description

Plot the classification tree for one region

Usage

plot_region_tree(analysis, region_code, title = NULL)

Arguments

analysis	Output of <code>build_pathway_analysis()</code> .
region_code	Region code.
title	Plot title.

Value

Called for its side effect (base graphics plot).

plot_rules_heatmap	<i>Heatmap of tree rules across regions</i>
--------------------	---

Description

Heatmap of tree rules across regions

Usage

```
plot_rules_heatmap(
  analysis,
  outcome_filter = "hits",
  title = "Pathways by region",
  wrap_width = 30,
  base_size = 11,
  save = FALSE,
  outfile = NULL,
  plotdir = ".",
  width = 18,
  height = 14
)
```

Arguments

analysis	Output of <code>build_pathway_analysis()</code> .
outcome_filter	"hits" or "misses".
title	Plot title.
wrap_width	Wrap width for rule labels.
base_size	Base font size.
save, outfile, plotdir, width, height	Saving options.

Value

A ggplot.

plot_spec_trajectories

Trajectories for one indicator spec

Description

Convenience wrapper: passes a spec from `indicator_specs()` or `nutrient_quantity_specs()` to `plot_target_trajectories()`.

Usage

```
plot_spec_trajectories(spec, ...)
```

Arguments

spec	One spec entry.
...	Further arguments to <code>plot_target_trajectories()</code> (e.g. regions, b4t_show, save, outfile, plotdir).

Value

A ggplot.

plot_target_trajectories

Indicator trajectories against a target

Description

Line figure over time, one panel per region. Bands show the range across GCMs for each pathway (SSP x RCP, climate ensemble); the black line is the baseline; coloured lines are the B4T runs built on the baseline (same SSP, GCM and RCP); the dashed line is the target.

Usage

```
plot_target_trajectories(
  df,
  value_col,
  threshold = NULL,
  threshold_col = NULL,
  regions = c(b4t_regions, "WLD"),
  b4t_show = c("all", "own"),
  ref_value = NULL,
  ref_label = "reference",
  baseline_id = crio_baseline_id,
```

```

mapping = "mappingCG6.xlsx",
indicator_label,
value_labeller = scales::label_number(),
title = NULL,
save = FALSE,
outfile = NULL,
plotdir = ".",
width = 10,
height = 7
)

```

Arguments

<code>df</code>	Indicator data with scenario columns (climate + B4T ensembles).
<code>value_col</code>	Column with the indicator value.
<code>threshold, threshold_col</code>	Fixed or dynamic target. A dynamic target follows the baseline's values over time.
<code>regions</code>	Region codes: CG regions and/or countries (e.g. <code>c("NGA", "IND")</code>).
<code>b4t_show</code>	"all": all 7 B4T runs; "own": B4T in the panel's own CG region + B4T in all CG regions.
<code>ref_value</code>	Optional extra horizontal reference line (dotted), e.g. the RNI when the target is a multiple of it.
<code>ref_label</code>	Name of the reference line in the subtitle.
<code>baseline_id</code>	Baseline scenario id.
<code>mapping</code>	GFSir mapping file (for country -> CG region).
<code>indicator_label</code>	Y axis label.
<code>value_labeller</code>	Label function for the y axis.
<code>title</code>	Plot title.
<code>save, outfile, plotdir, width, height</code>	Saving options.

Details

Note: if all GCMs give the same value (e.g. hunger at the model's floor), a band has zero width and is not visible.

Value

A ggplot.

read_scenarios	<i>Read one GFSir indicator for many scenarios</i>
----------------	--

Description

Calls a GFSir reader (e.g. `GFSir::group5()`) on each GDX file, adds the scenario id and binds the results.

Usage

```
read_scenarios(files, fun, ...)
```

Arguments

files	GDX file paths.
fun	GFSir reader function taking gdx as first argument.
...	Further arguments passed to fun (e.g. mapping, indicator).

Value

A data.table with all scenarios stacked.

relabel_scenarios	<i>Relabel scenarios in all datasets</i>
-------------------	--

Description

Re-applies `add_scenario_cols()` to every element of fulldata, e.g. to rename pathways or change the GCM classification without re-reading GDXs.

Usage

```
relabel_scenarios(fulldata, ...)
```

Arguments

fulldata	Output of <code>load_crio_data()</code> .
...	Arguments to <code>add_scenario_cols()</code> : pathways, low_warming_gcms, lower_emission_rcps.

Value

fulldata with updated scenario columns.

rni_lookup	<i>Recommended nutrient intakes (RNI)</i>
------------	---

Description

Adult RNIs per capita per day, from Sherwin's GTAP paper. Units are in the nutrient code suffix (g, mg, mcg).

Usage

```
rni_lookup
```

Format

A tibble with columns Nutrients and RNI.

row_countries	<i>Rest-of-world countries greyed out on maps</i>
---------------	---

Description

Rest-of-world countries greyed out on maps

Usage

```
row_countries
```

Format

A character vector of IMPACT region codes.

save_crio_data	<i>Save indicator datasets as RDS and/or CSV</i>
----------------	--

Description

One file per element of fulldata. CSVs keep only food groups, their commodities and totals where a name column exists.

Usage

```
save_crio_data(fulldata, rdsdir = NULL, csvdir = NULL, suffix = "")
```

Arguments

- fulldata Output of `load_crio_data()`.
- rdsdir Folder for RDS files (NULL to skip).
- csvdir Folder for CSV files (NULL to skip).
- suffix Suffix added to CSV file names, e.g. "-[v3]".

Value

fulldata, invisibly.

scenario_id	<i>Scenario id from a GDX file path</i>
-------------	---

Description

Scenario id from a GDX file path

Usage

scenario_id(gdx)

Arguments

- gdx GDX file path(s).

Value

File name without folder and .gdx extension.

select_scenarios	<i>Select scenarios in all datasets</i>
------------------	---

Description

Filters every element of fulldata that has scenario columns, so all later figures and tables use only the chosen runs. NULL keeps everything for that dimension.

Usage

```
select_scenarios(  
  fulldata,  
  ssp = NULL,  
  rcp = NULL,  
  gcm = NULL,  
  b4t = NULL,  
  ids = NULL  
)
```

Arguments

fulldata	Output of <code>load_crio_data()</code> .
ssp	SSPs to keep, e.g. <code>c("SSP2", "SSP3")</code> .
rcp	RCPs to keep, as parsed by GFSir, e.g. <code>"7.0"</code> .
gcm	GCMs to keep, e.g. <code>c("MRI", "UKESM")</code> .
b4t	B4T values to keep: <code>"None"</code> for climate runs, CG regions, <code>"All CG regions"</code> .
ids	Scenario ids to keep.

Value

fulldata with filtered elements.

summarise_rules	<i>Summarise tree rules by region and outcome</i>
-----------------	---

Description

Summarise tree rules by region and outcome

Usage

```
summarise_rules(rules_tbl)
```

Arguments

rules_tbl	rules_tbl from <code>build_pathway_analysis()</code> .
-----------	--

Value

One row per region with total cover, number of rules and the rules for "hits" and "misses".

world_outline	<i>World country outline for map borders</i>
---------------	--

Description

Built from the maps package, without Antarctica, dissolved into one outline. Cached after the first call.

Usage

```
world_outline()
```

Value

A `SpatVector`.

Index

* datasets

- aquatic_nutrient_per_kg, 4
- b4t_levels, 4
- b4t_regions, 5
- crio_baseline_id, 8
- default_pathways, 11
- exclude_cmdty, 11
- fat_kcal_share, 12
- food_groups, 13
- nutrient_labels, 22
- rni_lookup, 34
- row_countries, 34

add_scenario_cols, 3

add_scenario_cols(), 11, 13, 33

aquatic_nutrient_per_kg, 4, 18

b4t_levels, 4

b4t_regions, 5

build_b4t_comparison, 5

build_b4t_comparison(), 6, 24–26

build_b4t_comparisons, 6

build_pathway_analysis, 6

build_pathway_analysis(), 28–30, 36

cg_lookup, 7

climate_counterpart, 8

climate_counterpart(), 5

crio_baseline_id, 8

crio_plot_all, 9

default_pathways, 3, 11

exclude_cmdty, 11, 18

extract_rules, 12

fat_kcal_share, 12, 15

filter_climate, 13

filter_climate(), 6, 28

food_groups, 13, 14

get_ader, 14

get_calories, 14

get_calories(), 14, 15, 17, 19

get_fat_share, 15

get_foodavail, 15

get_foodavail(), 16, 18

get_frtveg, 16

get_hunger, 16

get_mder, 17

get_minkcal, 17

get_minkcal(), 17

get_nutrient_adequacy, 18

get_sugar_share, 19

get_water, 19

GFSir::group5(), 33

indicator_specs, 20

indicator_specs(), 6, 9, 10, 27, 31

list_scenarios, 20

list_scenarios(), 14–16, 19, 21

load_crio_data, 21

load_crio_data(), 9, 11, 20, 23, 33, 35, 36

load_region_map, 22

load_region_map(), 9, 25, 28

nutrient_labels, 22

nutrient_quantity_specs, 23

nutrient_quantity_specs(), 10, 31

own_cg_region, 23

plot_b4t_gap_closed, 24

plot_b4t_gap_maps, 25

plot_b4t_heatmap, 26

plot_gap_overview, 27

plot_gap_overview(), 10

plot_indicator_maps, 28

plot_pct_hits, 29

plot_region_tree, 29

plot_rules_heatmap, 30

`plot_spec_trajectories`, [31](#)
`plot_target_trajectories`, [31](#)
`plot_target_trajectories()`, [31](#)

`read_scenarios`, [33](#)
`relabel_scenarios`, [33](#)
`rni_lookup`, [18](#), [34](#)
`row_countries`, [28](#), [34](#)

`save_crio_data`, [34](#)
`scenario_id`, [35](#)
`select_scenarios`, [35](#)
`summarise_rules`, [36](#)

`world_outline`, [36](#)