Package 'cowfootR'

May 8, 2026

Type Package

Title Dairy Farm Carbon Footprint Assessment

Version 0.1.3

Description Calculates the carbon footprint of dairy farms based on methodologies of the International Dairy Federation and the Intergovernmental Panel on Climate Change. Includes tools for single-farm and batch analysis, report generation, and visualization. Methods follow International Dairy Federation (2022) ``The IDF global Carbon Footprint standard for the dairy sector'' (Bulletin of the IDF n° 520/2022) <doi:10.56169/FKRK7166> and IPCC (2019) ``2019 Refinement to the 2006 IPCC Guidelines for National Greenhouse Gas Inventories, Chapter 10: Emissions from Livestock and Manure Management'' <https://www.ipcc-nggip.iges.or.jp/public/2019rf/pdf/4_Volume4/19R_V4_Ch10_Livestock.pdf> guidelines.

License MIT + file LICENSE

URL https://github.com/juanmarcosmoreno-arch/cowfootR, https://juanmarcosmoreno-arch.github.io/cowfootR/

BugReports https://github.com/juanmarcosmoreno-arch/cowfootR/issues

Encoding UTF-8

Depends R (>= 4.1.0)

Imports writexl

Suggests testthat, knitr, readxl, rmarkdown, plotly, gt, dplyr, ggplot2, tidyr, withr

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author Juan Moreno [aut, cre]

Maintainer Juan Moreno <juanmarcosmoreno@gmail.com>

Repository CRAN

Date/Publication 2026-01-13 13:10:02 UTC

Contents

`benchmark_area_intensity` *Benchmark area intensity against regional data*

Description

Benchmark area intensity against regional data

Usage

```
benchmark_area_intensity(
  cf_area_intensity,
  region = NULL,
  benchmark_data = NULL
)
```

Arguments

`cf_area_intensity`	A cf_area_intensity object
`region`	Character. Region for comparison ("uruguay", "argentina", "brazil", "new_zealand", "ireland", "global")
`benchmark_data`	Named list. Custom benchmark data with mean and range

Value

Original object with added benchmarking information

Examples

```
res <- calc_intensity_area(total_emissions = 90000, area_total_ha = 150, area_productive_ha = 140)
out <- benchmark_area_intensity(res, region = "uruguay")
# str(out$benchmarking)
```

`calc_batch` *Batch carbon footprint calculation*

Description

Processes a data.frame of farms and computes annual emissions per farm, returning a summary plus per-farm details (optionally).

Usage

```
calc_batch(
  data,
  tier = 2,
  boundaries = set_system_boundaries("farm_gate"),
  benchmark_region = NULL,
  save_detailed_objects = FALSE
)
```

Arguments

`data`	A data.frame with one row per farm and annual activity data. At minimum, the following columns are required: • `FarmID`: Unique farm identifier. • `Milk_litres`: Annual milk production (litres/year). • `Cows_milking`: Number of milking cows. Additional columns are optional and enable more detailed Tier 2 calculations, including herd structure, feed intake, manure management, soil nitrogen inputs, energy use, and purchased inputs. When optional variables are not provided, default IPCC- or IDF-consistent values are used. All inputs are assumed to represent one accounting year.
`tier`	Integer; methodology tier (usually 1 or 2). Default = 2.
`boundaries`	System boundaries as returned by `set_system_boundaries()`.
`benchmark_region`	Optional character string specifying a geographic or regional benchmark (e.g., country or production region). When provided, emission intensity results are compared against region-specific reference values using internal benchmarking functions. This argument does not affect emission calculations and is only used for comparative performance assessment.
`save_detailed_objects`	Logical; if TRUE, returns detailed objects per farm.

Details

The input data frame is intentionally flexible to support heterogeneous data availability across farms. Each row represents one farm and all inputs are assumed to correspond to a single accounting year, unless explicitly stated otherwise by the column name (e.g., *_kg_day).

Column names follow cowfootR conventions. The complete and authoritative specification of supported input columns (including expected units and whether they are required or optional) is provided by the Excel template generated with `cf_download_template()`. This template represents the full set of columns that can be used by `calc_batch()`.

In addition, the vignettes "Get started" and "IPCC Methodology Tiers in cowfootR" describe how these columns are used conceptually and methodologically.

Tier 2–relevant optional columns: When `tier = 2`, `calc_batch()` uses additional farm-specific information if available. The most relevant optional columns include:

- **Enteric fermentation:** `Milk_yield_kg_cow_year`, `Body_weight_cows_kg`, `MS_intake_cows_milking_kg_day`, `Ym_percent`.
- **Young stock (optional refinement):** `Body_weight_heifers_kg`, `Body_weight_calves_kg`, `Body_weight_bulls_kg`, `MS_intake_heifers_kg_day`, `MS_intake_calves_kg_day`, `MS_intake_bulls_kg_day`.
- **Manure management:** `Manure_system`, `Diet_digestibility`, `Protein_intake_kg_day`, `Retention_days`, `System_temperature`, `Climate_zone`.

If any Tier 2–relevant column is missing, the function automatically falls back to Tier 1–consistent default assumptions following IPCC and IDF guidance. Missing optional inputs therefore do not cause errors.

Value

A list with `$summary` and `$farm_results`; class `cf_batch_complete`. Absolute emissions returned in `$farm_results` (e.g., `emissions_total`, `emissions_enteric`, `emissions_manure`, etc.) are annual emissions expressed as kg CO2-equivalent per year (kg CO2eq yr-1) at the farm (system) level, within the defined system boundaries. Intensity metrics are reported as kg CO2eq per kg FPCM and kg CO2eq per ha (based on annual milk production and managed area).

Examples

```
farms <- data.frame(
  FarmID = c("A", "B"),
  Milk_litres = c(5e5, 7e5),
  Cows_milking = c(100, 140)
)
res <- calc_batch(
  data = farms,
  tier = 2,
  boundaries = set_system_boundaries("farm_gate"),
  benchmark_region = "uruguay",
  save_detailed_objects = FALSE
)
str(res$summary)
```

`calc_emissions_energy` *Calculate energy-related emissions*

Description

Estimates CO2 emissions from fossil fuel use and electricity consumption on dairy farms following IDF/IPCC methodology.

Usage

```
calc_emissions_energy(
  diesel_l = 0,
  petrol_l = 0,
  lpg_kg = 0,
  natural_gas_m3 = 0,
  electricity_kwh = 0,
  country = "UY",
  ef_diesel = 2.67,
  ef_petrol = 2.31,
  ef_lpg = 3,
  ef_natural_gas = 2,
  ef_electricity = NULL,
  include_upstream = FALSE,
  energy_breakdown = NULL,
  boundaries = NULL
)
```

Arguments

`diesel_l`	Numeric. Diesel consumption (liters/year). Default = 0.
`petrol_l`	Numeric. Petrol/gasoline consumption (liters/year). Default = 0.
`lpg_kg`	Numeric. LPG/propane consumption (kg/year). Default = 0.
`natural_gas_m3`	Numeric. Natural gas consumption (m³/year). Default = 0.
`electricity_kwh`	Numeric. Electricity consumption (kWh/year). Default = 0.
`country`	Character. Country code for electricity grid factors. Default = "UY" (Uruguay). Options include "UY", "AR", "BR", "NZ", "US", etc.
`ef_diesel`	Numeric. Emission factor for diesel (kg CO2/liter). Default = 2.67 (IPCC 2019, combustion).
`ef_petrol`	Numeric. Emission factor for petrol (kg CO2/liter). Default = 2.31 (IPCC 2019).
`ef_lpg`	Numeric. Emission factor for LPG (kg CO2/kg). Default = 3.0 (IPCC 2019).
`ef_natural_gas`	Numeric. Emission factor for natural gas (kg CO2/m³). Default = 2.0 (IPCC 2019).

`ef_electricity` Numeric. Emission factor for electricity (kg CO2/kWh). If NULL, uses country-specific grid factors.

`include_upstream` Logical. Include upstream emissions from fuel production? Default = FALSE (combustion only).

`energy_breakdown` Optional. Detailed breakdown by equipment/use (list or data.frame). If list, each element can include diesel_l, petrol_l, lpg_kg, natural_gas_m3, electricity_kwh.

`boundaries` Optional. An object from `set_system_boundaries()`. If "energy" is not included, returns an excluded record.

Value

A list with detailed emissions by fuel type, total (co2eq_kg), metadata, and (if provided) breakdown by use. Compatible with `calc_total_emissions()`.

Examples

```
# Minimal, fast example (<<1s)
res <- calc_emissions_energy(
  diesel_l = 10,
  electricity_kwh = 100,
  country = "UY"
)
print(res$co2eq_kg)
```

---

calc_emissions_enteric

Calculate enteric methane emissions

Description

Estimates enteric methane (CH4) emissions from cattle using IPCC Tier 1 or Tier 2 approaches with practical defaults for dairy systems.

Usage

```
calc_emissions_enteric(
  n_animals,
  cattle_category = "dairy_cows",
  production_system = "mixed",
  avg_milk_yield = 6000,
  avg_body_weight = NULL,
  dry_matter_intake = NULL,
  feed_inputs = NULL,
  ym_percent = 6.5,
  emission_factor_ch4 = NULL,
```

```
  tier = 1L,
  gwp_ch4 = 27.2,
  boundaries = NULL
)
```

Arguments

n_animals	Numeric scalar > 0. Number of animals.
cattle_category	Character. One of "dairy_cows", "heifers", "calves", "bulls". Default = "dairy_cows".
production_system	Character. One of "intensive", "extensive", "mixed". Default = "mixed".
avg_milk_yield	Numeric >= 0. Average annual milk yield per cow (kg/year). Default = 6000. Used in Tier 2 fallback for dairy cows.
avg_body_weight	Numeric > 0. Average live weight (kg). If NULL, a category-specific default is used (e.g. 550 kg for dairy cows).
dry_matter_intake	Numeric > 0. Dry matter intake (kg/animal/day). If provided (Tier 2), overrides body-weight/energy-based estimation.
feed_inputs	Named numeric vector/list with feed DM amounts in kg/year per herd (e.g., grain_dry, grain_wet, byproducts, proteins). Optional. If given and `dry_matter_intake` is NULL, DMI is inferred as `sum(feed_inputs)/(n_animals*365)`.
ym_percent	Numeric in (0, 100]. Methane conversion factor Ym (% of GE to CH4). Default = 6.5.
emission_factor_ch4	Numeric > 0. If provided, CH4 EF (kg CH4/head/year) is used directly; otherwise it is calculated (Tier 1 or Tier 2).
tier	Integer 1 or 2. Default = 1.
gwp_ch4	Numeric. GWP for CH4 (100-yr, AR6). Default = 27.2.
boundaries	Optional list from `set_system_boundaries()`.

Value

A list with annual CH4 emissions and annual CO2-equivalent emissions.

- `ch4_kg`: annual CH4 emissions (kg CH4 yr-1)
- `co2eq_kg`: annual CO2-equivalent emissions (kg CO2eq yr-1), for compatibility with `calc_total_emissions()`
- `units_ch4` and `units_co2eq`: explicit unit strings

Examples

```
# Tier 1, mixed dairy cows
calc_emissions_enteric(n_animals = 100)

# Tier 2 with explicit DMI
calc_emissions_enteric(
```

```
  n_animals = 120, tier = 2, avg_milk_yield = 7500, dry_matter_intake = 18
)

# Boundary exclusion: enteric not included
b <- list(include = c("manure", "energy"))
calc_emissions_enteric(100, boundaries = b)$co2eq_kg # NULL → excluded
```

`calc_emissions_inputs` *Calculate indirect emissions from purchased inputs*

Description

Estimates CO2e emissions from purchased inputs such as feeds, fertilizers, and plastics using regional factors, with optional uncertainty analysis.

Usage

```
calc_emissions_inputs(
  conc_kg = 0,
  fert_n_kg = 0,
  plastic_kg = 0,
  feed_grain_dry_kg = 0,
  feed_grain_wet_kg = 0,
  feed_ration_kg = 0,
  feed_byproducts_kg = 0,
  feed_proteins_kg = 0,
  feed_corn_kg = 0,
  feed_soy_kg = 0,
  feed_wheat_kg = 0,
  region = "global",
  fert_type = "mixed",
  plastic_type = "mixed",
  include_uncertainty = FALSE,
  transport_km = NULL,
  ef_conc = NULL,
  ef_fert = NULL,
  ef_plastic = NULL,
  boundaries = NULL
)
```

Arguments

`conc_kg`	Numeric. Purchased concentrate feed (kg/year). Default = 0.
`fert_n_kg`	Numeric. Purchased nitrogen fertilizer (kg N/year). Default = 0.
`plastic_kg`	Numeric. Agricultural plastics used (kg/year). Default = 0.

`feed_grain_dry_kg`
: Numeric. Grain dry (kg/year, DM). Default = 0.

`feed_grain_wet_kg`
: Numeric. Grain wet (kg/year, DM). Default = 0.

`feed_ration_kg`
: Numeric. Ration (total mixed ration) (kg/year, DM). Default = 0.

`feed_byproducts_kg`
: Numeric. Byproducts (kg/year, DM). Default = 0.

`feed_proteins_kg`
: Numeric. Protein feeds (kg/year, DM). Default = 0.

`feed_corn_kg`
: Numeric. Corn (kg/year, DM). Default = 0.

`feed_soy_kg`
: Numeric. Soybean meal (kg/year, DM). Default = 0.

`feed_wheat_kg`
: Numeric. Wheat (kg/year, DM). Default = 0.

`region`
: Character. "EU","US","Brazil","Argentina","Australia","global". Default "global".

`fert_type`
: Character. "urea","ammonium_nitrate","mixed","organic". Default "mixed".

`plastic_type`
: Character. "LDPE","HDPE","PP","mixed". Default "mixed".

`include_uncertainty`
: Logical. Include uncertainty ranges? Default FALSE.

`transport_km`
: Numeric. Average feed transport distance (km). Optional.

`ef_conc`, `ef_fert`, `ef_plastic`
: Numeric overrides for emission factors (kg CO2e per unit).

`boundaries`
: Optional. Object from `set_system_boundaries()`.

Details

Notes:

- When system boundaries exclude "inputs", this function MUST return a list with `source = "inputs"` and a numeric `co2eq_kg = 0` to satisfy partial-boundaries integration.
- The primary total field is `co2eq_kg` (for compatibility with `calc_total_emissions()`); `total_co2eq_kg` is included as a duplicate for convenience.

Value

A list with fields:

- source = "inputs"
- emissions_breakdown (named values per input)
- co2eq_kg (numeric total)
- total_co2eq_kg (duplicate of co2eq_kg)
- emission_factors_used, inputs_summary, contribution_analysis, uncertainty (if requested)
- metadata (methodology, standards, date)

Examples

```
# Quick example (runs fast)
calc_emissions_inputs(conc_kg = 1000, fert_n_kg = 200, region = "EU")

# With uncertainty analysis
calc_emissions_inputs(feed_corn_kg = 2000, region = "US", include_uncertainty = TRUE)
```

---

calc_emissions_manure *Calculate manure management emissions (Tier 1 & Tier 2)*

Description

Estimates CH4 and N2O emissions from manure management using IPCC Tier 1 or Tier 2 methodology with practical settings for dairy systems.

Usage

```
calc_emissions_manure(
  n_cows,
  manure_system = "pasture",
  tier = 1L,
  ef_ch4 = NULL,
  n_excreted = 100,
  ef_n2o_direct = 0.02,
  include_indirect = FALSE,
  climate = "temperate",
  avg_body_weight = 600,
  diet_digestibility = 0.65,
  protein_intake_kg = NULL,
  retention_days = NULL,
  system_temperature = NULL,
  gwp_ch4 = 27.2,
  gwp_n2o = 273,
  boundaries = NULL
)
```

Arguments

n_cows	Numeric scalar > 0. Number of dairy cows.
manure_system	Character. One of "pasture", "solid_storage", "liquid_storage", "anaerobic_digester". Default = "pasture".
tier	Integer. IPCC tier (1 or 2). Default = 1.
ef_ch4	Numeric. CH4 EF (kg CH4/cow/year). If NULL, system-specific defaults are used (Tier 1 only).
n_excreted	Numeric. N excreted per cow per year (kg N). Default = 100. In Tier 2 it may be recalculated if protein intake is provided.

`ef_n2o_direct` Numeric. Direct N2O-N EF (kg N2O-N per kg N). Default = 0.02.

`include_indirect`
: Logical. Include indirect N2O (volatilization + leaching)? Default = FALSE.

`climate` Character. One of "cold", "temperate", "warm". Default = "temperate" (Tier 2).

`avg_body_weight`
: Numeric. Average live weight (kg). Default = 600 (Tier 2).

`diet_digestibility`
: Numeric in (0, 1]. Apparent digestibility. Default = 0.65 (Tier 2).

`protein_intake_kg`
: Numeric. Daily protein intake (kg/day). If provided, Tier 2 can refine N excretion.

`retention_days` Numeric. Days manure remains in system (Tier 2 adjustment).

`system_temperature`
: Numeric. Average system temperature (Tier 2 adjustment).

`gwp_ch4` Numeric. GWP for CH4 (AR6). Default = 27.2.

`gwp_n2o` Numeric. GWP for N2O (AR6). Default = 273.

`boundaries` Optional list from `set_system_boundaries()`.

Value

A list with CH4 (kg), N2O (kg), CO2eq (kg), metadata, and per-cow metrics. The returned object includes a `co2eq_kg` field compatible with `calc_total_emissions()`. Absolute emissions are annual farm-level emissions (kg CO2eq yr-1) within the defined system boundaries.

Examples

```
# Tier 1, liquid storage
calc_emissions_manure(n_cows = 120, manure_system = "liquid_storage")

# Tier 1 with indirect N2O
calc_emissions_manure(n_cows = 120, manure_system = "solid_storage", include_indirect = TRUE)

# Tier 2 (VS_B0_MCF approach) with refinements
calc_emissions_manure(
  n_cows = 100, manure_system = "liquid_storage", tier = 2,
  avg_body_weight = 580, diet_digestibility = 0.68, climate = "temperate",
  protein_intake_kg = 3.2, include_indirect = TRUE
)
```

`calc_emissions_soil` *Calculate soil N2O emissions*

Description

Estimates direct and indirect N2O emissions from soils due to fertilisation, excreta deposition and crop residues, following a Tier 1-style IPCC approach.

Usage

```
calc_emissions_soil(
  n_fertilizer_synthetic = 0,
  n_fertilizer_organic = 0,
  n_excreta_pasture = 0,
  n_crop_residues = 0,
  area_ha = NULL,
  soil_type = "well_drained",
  climate = "temperate",
  ef_direct = NULL,
  include_indirect = TRUE,
  gwp_n2o = 273,
  boundaries = NULL
)
```

Arguments

`n_fertilizer_synthetic`	Numeric. Synthetic N fertiliser applied (kg N/year). Default = 0.
`n_fertilizer_organic`	Numeric. Organic N fertiliser applied (kg N/year). Default = 0.
`n_excreta_pasture`	Numeric. N excreted directly on pasture (kg N/year). Default = 0.
`n_crop_residues`	Numeric. N in crop residues returned to soil (kg N/year). Default = 0.
`area_ha`	Numeric. Total farm area (ha). Optional, for per-hectare metrics.
`soil_type`	Character. "well_drained" or "poorly_drained". Default = "well_drained".
`climate`	Character. "temperate" or "tropical". Default = "temperate".
`ef_direct`	Numeric. Direct EF for N2O-N (kg N2O-N per kg N input). If NULL, uses IPCC-style values by soil/climate.
`include_indirect`	Logical. Include indirect N2O (volatilisation + leaching)? Default = TRUE.
`gwp_n2o`	Numeric. GWP of N2O. Default = 273 (IPCC AR6).
`boundaries`	Optional. Object from `set_system_boundaries()`. If soil is excluded, returns `co2eq_kg = 0`.

Details

IMPORTANT: When system boundaries exclude soil, this function must return a list with `source = "soil"` and `co2eq_kg = 0` (numeric zero) to match partial-boundaries integration tests.

Value

A list with at least `source="soil"` and `co2eq_kg` (numeric), plus detailed breakdown metadata when included by boundaries. Absolute emissions are annual farm-level emissions (kg CO2eq yr-1) within the defined system boundaries.

Examples

```
# Direct + indirect (default), temperate, well-drained
calc_emissions_soil(
  n_fertilizer_synthetic = 2500,
  n_fertilizer_organic   = 500,
  n_excreta_pasture      = 1200,
  n_crop_residues        = 300,
  area_ha                = 150
)

# Direct-only
calc_emissions_soil(n_fertilizer_synthetic = 2000, include_indirect = FALSE)

# Boundary exclusion example
b <- list(include = c("energy", "manure")) # soil not included
calc_emissions_soil(n_fertilizer_synthetic = 1000, boundaries = b)$co2eq_kg # 0
```

---

calc_intensity_area *Calculate carbon footprint intensity per hectare*

Description

Computes emissions intensity per unit of land area for dairy farm analysis.

Usage

```
calc_intensity_area(
  total_emissions,
  area_total_ha,
  area_productive_ha = NULL,
  area_breakdown = NULL,
  validate_area_sum = TRUE
)
```

Arguments

total_emissions	Numeric or cf_total object. Total emissions in kg CO2eq (from `calc_total_emissions()`) or the object itself.
area_total_ha	Numeric. Total farm area in hectares.
area_productive_ha	Numeric. Productive/utilized area in hectares. If NULL, uses total area. Default = NULL.
area_breakdown	Named list or named numeric vector. Optional detailed area breakdown by land use type. Names should be descriptive (e.g., "pasture_permanent", "crops_feed").
validate_area_sum	Logical. Check if area breakdown sums to total? Default = TRUE.

Details

The area_breakdown parameter allows detailed tracking by land use:

```
area_breakdown = list(
  pasture_permanent = 80,
  pasture_temporary = 20,
  crops_feed = 15,
  crops_cash = 5,
  infrastructure = 2,
  woodland = 8
)
```

Value

A list of class "cf_area_intensity" with intensity metrics and area analysis.

Examples

```
# Basic calculation
calc_intensity_area(total_emissions = 85000, area_total_ha = 120)

# With productive area distinction
calc_intensity_area(
  total_emissions = 95000,
  area_total_ha = 150,
  area_productive_ha = 135
)

# With area breakdown
area_detail <- list(
  pasture_permanent = 80,
  pasture_temporary = 25,
  crops_feed = 20,
  infrastructure = 3,
  woodland = 7
```

```
)
calc_intensity_area(
  total_emissions = 88000,
  area_total_ha = 135,
  area_breakdown = area_detail
)

# Using with calc_total_emissions output
#

# b <- set_system_boundaries("farm_gate")
# e1 <- calc_emissions_enteric(100, boundaries = b)
# e2 <- calc_emissions_manure(100, boundaries = b)
# tot <- calc_total_emissions(e1, e2)
# calc_intensity_area(tot, area_total_ha = 120)
#
```

`calc_intensity_litre` *Calculate carbon footprint intensity per kg of milk*

Description

Computes emissions intensity as kg CO2eq per kg of fat- and protein-corrected milk (FPCM).

Usage

```
calc_intensity_litre(
  total_emissions,
  milk_litres,
  fat = 4,
  protein = 3.3,
  milk_density = 1.03
)
```

Arguments

`total_emissions`	Numeric or cf_total object. Total emissions in kg CO2eq (from `calc_total_emissions()`) or the object itself.
`milk_litres`	Numeric. Annual milk production in litres.
`fat`	Numeric. Average fat percentage of milk (0-100). Default = 4.
`protein`	Numeric. Average protein percentage of milk (0-100). Default = 3.3.
`milk_density`	Numeric. Milk density in kg/L. Default = 1.03.

Details

The correction to FPCM (fat- and protein-corrected milk) follows the IDF formula:

$$FPCM = milk_kg * (0.1226 * fat_pct + 0.0776 * protein_pct + 0.2534)$$

Where milk_kg = milk_litres * milk_density

Value

A list of class "cf_intensity" with intensity (kg CO2eq/kg FPCM), FPCM production, and calculation details.

Examples

```
# Using numeric total emissions directly
calc_intensity_litre(total_emissions = 85000, milk_litres = 750000)

# If you have a cf_total object 'tot' (e.g., from calc_total_emissions):
# calc_intensity_litre(tot, milk_litres = 750000)
```

---

calc_total_emissions *Calculate total emissions (robust and boundary-aware)*

Description

Aggregates results from different sources (enteric, manure, soil, energy, inputs) even if they don't use exactly the same field name for the total. IMPORTANT: If a source explicitly reports `co2eq_kg = NULL` (e.g. excluded by system boundaries), it is treated as zero and no fallback summation is attempted.

Usage

```
calc_total_emissions(...)
```

Arguments

`...`	Results from `calc_emissions_*()` functions (lists).

Value

Object of class `cf_total` including:

- `total_co2eq`: total absolute emissions (kg CO2eq yr-1)
- `breakdown`: named numeric vector by source (kg CO2eq yr-1)
- `by_source`: data.frame with `source`, `co2eq_kg`, and `units`
- `units_total`, `units_by_source`: unit strings

`export_hdc_report` *Export cowfootR batch results to Excel*

Description

Exports results from `calc_batch()` into an Excel file with summary and farm-level sheets. Emission columns are exported as annual emissions (kg CO2eq yr-1). Intensity columns are exported as kg CO2eq per kg FPCM and kg CO2eq per ha.

Usage

```
export_hdc_report(
  batch_results,
  file = "cowfootR_report.xlsx",
  include_details = FALSE
)
```

Arguments

`batch_results`	A `cf_batch_complete` object returned by `calc_batch()`.
`file`	Path to the Excel file to save. Default = "cowfootR_report.xlsx".
`include_details`	Logical. If TRUE, includes extra sheets with detailed objects (if available).

Value

Invisibly returns the file path. The Excel output includes unit-consistent columns for annual emissions (kg CO2eq yr-1) and emission intensities.

Examples

```
# Minimal dummy object (como el devuelto por calc_batch)
br <- list(
  summary = list(
    n_farms_processed = 1L,
    n_farms_successful = 1L,
    n_farms_with_errors = 0L,
    boundaries_used = list(scope = "farm_gate"),
    benchmark_region = NA_character_,
    processing_date = Sys.Date()
  ),
  farm_results = list(list(
    success = TRUE,
    farm_id = "Farm_A",
    year = format(Sys.Date(), "%Y"),
    emissions_enteric = 100, emissions_manure = 50, emissions_soil = 20,
    emissions_energy = 10, emissions_inputs = 5, emissions_total = 185,
    intensity_milk_kg_co2eq_per_kg_fpcm = 1.2,
```

```
    intensity_area_kg_co2eq_per_ha_total = 800,
    intensity_area_kg_co2eq_per_ha_productive = 1000,
    fpcm_production_kg = 150000, milk_production_kg = 154500,
    milk_production_litres = 150000,
    land_use_efficiency = 3000,
    total_animals = 200, dairy_cows = 120,
    benchmark_region = NA_character_, benchmark_performance = NA_character_,
    processing_date = Sys.Date(), boundaries_used = "farm_gate",
    tier_used = "tier_2", detailed_objects = NULL
  ))
)
class(br) <- "cf_batch_complete"

f <- tempfile(fileext = ".xlsx")
export_hdc_report(br, file = f)
file.exists(f)
```

`print.cf_area_intensity` *Print method for cf_area_intensity objects*

Description

Print method for cf_area_intensity objects

Usage

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

Arguments

`x`	A cf_area_intensity object
`...`	Additional arguments (ignored)

Value

No return value, called for side effects. Prints formatted area intensity information to the console and invisibly returns the input object.

The input object `x`, invisibly.

Examples

```
x <- list(
  intensity_per_total_ha = 900,
  intensity_per_productive_ha = 1100,
  land_use_efficiency = 0.92,
  total_emissions_co2eq = 108000,
```

```
  area_total_ha = 120,
  area_productive_ha = 110,
  date = Sys.Date()
)
class(x) <- "cf_area_intensity"
print(x)
```

`print.cf_intensity` *Print method for cf_intensity objects*

Description

Print method for cf_intensity objects

Usage

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

Arguments

`x`	A cf_intensity object
`...`	Additional arguments (ignored)

Value

No return value, called for side effects. Prints formatted carbon footprint intensity information to the console and invisibly returns the input object.

The input object `x`, invisibly.

Examples

```
x <- list(
  intensity_co2eq_per_kg_fpcm = 0.9,
  total_emissions_co2eq = 85000,
  milk_production_litres = 750000,
  milk_production_kg = 750000 * 1.03,
  fpcm_production_kg = 750000 * 1.03 * (0.1226 * 4 + 0.0776 * 3.3 + 0.2534),
  fat_percent = 4, protein_percent = 3.3, milk_density_kg_per_l = 1.03,
  date = Sys.Date()
)
class(x) <- "cf_intensity"
# print(x)
```

print.cf_total *Print method for cf_total objects*

Description

Print method for cf_total objects

Usage

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

Arguments

x	A cf_total object
...	Additional arguments passed to print methods (currently ignored)

Value

The input object x, invisibly.

set_system_boundaries *Define system boundaries for carbon footprint calculation*

Description

Define system boundaries for carbon footprint calculation

Usage

```
set_system_boundaries(scope = "farm_gate", include = NULL)
```

Arguments

scope	Character. Options: • "farm_gate" (default): includes enteric, manure, soil, energy, inputs • "cradle_to_farm_gate": includes feed production + farm emissions • "partial": user-specified
include	Character vector of processes to include (optional).

Value

A list with $scope and $include

Examples

```
b1 <- set_system_boundaries("farm_gate")
b2 <- set_system_boundaries(include = c("enteric", "manure", "soil"))
b3 <- set_system_boundaries(include = c("enteric", "manure"))
b1$scope
b2$include
b3$include
```

Index