Package: carbonr (via r-universe)

October 17, 2024

Title Calculate Carbon-Equivalent Emissions

Version 0.2.1

Description Provides a flexible tool for calculating carbon-equivalent emissions. Mostly using data from the UK Government's Greenhouse Gas Conversion Factors report <https://www.gov.uk/government/publications/greenhouse-gas-reporting-conversion-factors-2023>, it facilitates transparent emissions calculations for various sectors, including travel, accommodation, and clinical activities. The package is designed for easy integration into R workflows, with additional support for 'shiny' applications and community-driven extensions.

License LGPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Suggests knitr, purrr, rmarkdown, testthat (>= 2.0.0)

Config/testthat/edition 2

Imports airportr, checkmate, cowplot, dplyr, emojifont, ggplot2, ggpp, htmltools, lubridate, magrittr, readxl, rlang, shiny, shinydashboard, sp, stringr, tibble, tidyr, tidyselect

Depends R (>= 3.5.0)

LazyData true

VignetteBuilder knitr

NeedsCompilation no

Author Lily Clements [aut, cre] (<https://orcid.org/0000-0001-8864-0552>)

Maintainer Lily Clements <lily@idems.international>

Repository CRAN

Date/Publication 2024-10-16 18:10:09 UTC

Contents

add_inputs *Create multiple textInput functions in Shiny*

Description

For use in the shiny_emissions() function. Adding an unknown quantity of textInputs.

Usage

```
add_inputs(numeric_input, label, value)
```

Arguments

numeric_input	Name of numerical input that controls the number of items to add.
label	Label of new textInput.
value	Value of new textInput.

Value

Returns textInput for use in the shiny_emissions() function.

Examples

```
if(interactive()) {
ui <- shinydashboard::dashboardPage(header = shinydashboard::dashboardHeader(),
                                    sidebar = shinydashboard::dashboardSidebar(),
                                    shinydashboard::dashboardBody(
                                    shiny::fluidRow(
                                    shiny::column(12, align = "left",
                                    shiny::splitLayout(shinydashboard::box(width = NULL,
                                    shiny::numericInput("newbox_add",
                                                        "Number of new boxes:",
                                                        value = 0, min = 0),
                                    shiny::uiOutput("newbox_input")))))))
server <- function(input, output) {
  K_plane <- shiny::reactive({ input$newbox_add })
  output$newbox_input <- shiny::renderUI({ add_inputs(numeric_input = K_plane(),
                                                      label = "New Box:",
                                                      value = "textbox") })
}
shiny::shinyApp(ui, server)
}
```

`airplane_emissions` *Calculate CO2e emissions from an airplane journey*

Description

This function calculates the CO2e emissions between airports based on the provided parameters. The distances are calculated using the "airport_distance" function from the "airportr" package.

Usage

```
airplane_emissions(
  from,
  to,
  via = NULL,
  num_people = 1,
  radiative_force = TRUE,
  include_WTT = TRUE,
  round_trip = FALSE,
  class = c("Average passenger", "Economy class", "Business class",
    "Premium economy class", "First class")
)
```

Arguments

`from`	Three-letter IATA code corresponding to the departure airport. You can check the IATA code using the "airport_finder" function.
`to`	Three-letter IATA code corresponding to the destination airport. You can check the IATA code using the "airport_finder" function.
`via`	Optional. Vector of three-letter IATA codes corresponding to airports for any layovers or stops along the route.
`num_people`	Number of people taking the flight. Must be a single numerical value.
`radiative_force`	Logical. Determines whether radiative forcing should be taken into account. It is recommended to set this parameter as TRUE since emissions from airplanes at higher altitudes have a greater impact on climate change than those at ground level.
`include_WTT`	Logical. Determines whether emissions associated with extracting, refining, and transporting fuels should be included. It is recommended to set this parameter as TRUE.
`round_trip`	Logical. Determines if the flight is round trip (return) or one-way. Default is FALSE (one-way).
`class`	Class flown in. Options include "Average passenger", "Economy class", "Business class", "Premium economy class", and "First class".

Details

The distances are calculated using the "airport_distance" function from the "airportr" package. This means that the distances between locations uses the Haversine formula. This is calculated as the crow flies.

Value

Returns CO2e emissions in tonnes.

Examples

```
# Calculate emissions for a flight between Vancouver (YVR) and Toronto (YYZ)
airplane_emissions("YVR", "YYZ")
# Calculate emissions for a flight between London Heathrow (LHR)
# and Kisumu Airport (KIS), with layovers in Amsterdam (AMS) and Nairobi
# (NBO), flying in Economy class.
airplane_emissions("LHR", "KIS", via = c("AMS", "NBO"),
                   class = "Economy class")
```

---

airports *Table of airport detail data*

Description

This dataset is adapted from the `airportr` package. Full credit and acknowledgment go to the original authors of the `airportr` package for their contribution. A dataset containing names, codes, locations, altitude, and timezones for airports.

Usage

```
airports
```

Format

A data frame with 7698 rows and 14 variables:

OpenFlights ID OpenFlights database ID

Name Airport name, sometimes contains name of the city

City Name of the city served by the airport

IATA 3-letter IATA code

ICAO 4-letter ICAO code

Country Country name as in OpenFlights database. Note that country names may not be ISO 3166-1 standard.

Country Code ISO 3166-1 numeric country code

Country Code (Alpha-2) Two-letter ISO country code

Country Code (Alpha-3) Three-letter ISO country code

Latitude Latitude in decimal degrees

Longitude Longitude in decimal degrees

Altitude Altitude in feet

UTC Hours offset from UTC

DST Daylight Savings Time. One of E (Europe), A (US/Canada), S (South America), O (Australia), Z (New Zealand), N (None) or U (Unknown)

Timezone Timezone in Olson format

Type Type of airport (e.g., large airport, medium airport, small airport)

Source Source of data, generally sourced from OurAirports

Source

https://cran.r-project.org/package=airportr

`airport_finder` *Find the airport code for an airport*

Description

Find the name, city, country, and IATA code of an airport. For use in the `airplane_emissions` function.

Usage

```
airport_finder(
  name,
  city,
  country,
  IATA_code,
  distance = 0.1,
  ignore.case = FALSE
)
```

Arguments

`name`	Name of the airport.
`city`	City that the airport is in.
`country`	Country that the airport is in.
`IATA_code`	The IATA code.
`distance`	Maximum distance allowed for a match between the name/country/city given, and that of the value in the data set.
`ignore.case`	If `FALSE`, the check for is case-sensitive. If `TRUE`, case is ignored.

Value

Data frame containing the name, city, country, and IATA code of an airport.

Examples

```
# Can get the IATA code from the name of an airport. Gets similar matches.
airport_finder(name = "Bristo")

# Can get the IATA code from the name and city of an airport
airport_finder(name = "Bristo", country = "United Kingdom")

# Can find the name and city of an airport given the IATA code
airport_finder(IATA_code = "BRS")
```

---

anaesthetic_emissions *Anaesthetic Emissions*

Description

Estimates the CO2e emissions associated with different anaesthetic agents.

Usage

```
anaesthetic_emissions(
  desflurane = 0,
  sevoflurane = 0,
  isoflurane = 0,
  N2O = 0,
  methoxyflurane = 0,
  propofol = 0
)
```

Arguments

desflurane	Amount of desflurane used in KG (default: 0).
sevoflurane	Amount of sevoflurane used in KG (default: 0).
isoflurane	Amount of isoflurane used in KG (default: 0).
N2O	Amount of nitrous oxide (N2O) used in KG (default: 0).
methoxyflurane	Amount of methoxyflurane used in KG (default: 0).
propofol	Amount of propofol used in KG (default: 0).

Details

These estimates are based on available literature and may vary depending on factors such as specific anaesthetic agents, usage conditions, and waste gas management practices.

Value

The total CO2e emissions in tonnes.

References

- McGain F, Muret J, Lawson C, Sherman JD. Environmental sustainability in anaesthesia and critical care. Br J Anaesth. 2020 Nov;125(5):680-692. DOI: 10.1016/j.bja.2020.06.055. Epub 2020 Aug 12. PMID: 32798068; PMCID: PMC7421303.
- ACS Sustainable Chem. Eng. 2019, 7, 7, 6580–6591. Publication Date: January 20, 2019. Link
- Sherman, Jodi MD*; Le, Cathy; Lamers, Vanessa; Eckelman, Matthew PhD. Life Cycle Greenhouse Gas Emissions of Anesthetic Drugs. Anesthesia & Analgesia 114(5):p 1086-1090, May 2012. DOI: 10.1213/ANE.0b013e31824f6940. Link

Examples

```
anaesthetic_emissions(desflurane = 200, sevoflurane = 30, N2O = 5)
```

`building_emissions` *Building emissions*

Description

Further options are given in the `raw_fuels` function.

Usage

```
building_emissions(
  water_supply = 0,
  water_trt = TRUE,
  water_unit = c("cubic metres", "million litres"),
  electricity_kWh = 0,
  electricity_TD = TRUE,
  electricity_WTT = TRUE,
  heat_kWh = 0,
  heat_TD = TRUE,
  heat_WTT = TRUE
)
```

Arguments

`water_supply`	Amount of water used in the building.
`water_trt`	logical. Default `TRUE`. Whether to include emissions associated with water treatment for used water.

water_unit	Unit for water_supply variable. Options are "cubic metres" or "million litres".
electricity_kWh	Electricity used in kWh.
electricity_TD	logical. Default TRUE. Whether to include emissions associated with grid losses.
electricity_WTT	logical. Default TRUE. Whether to include emissions associated with extracting, refining, and transporting fuels.
heat_kWh	heat and steam used in kWh.
heat_TD	logical. Default TRUE. Whether to include emissions associated with grid losses.
heat_WTT	logical. Default TRUE. Whether to include emissions associated with extracting, refining, and transporting fuels.

Value

Returns CO2e emissions in tonnes.

References

Descriptions from 2022 UK Government Report: https://www.gov.uk/government/publications/greenhouse-gas-reporting-conversion-factors-2022

Examples

```
# specify emissions in an office
building_emissions(electricity_kWh = 200,
                heat_kWh = 100, water_supply = 100, water_trt = FALSE)
```

---

carbonr *carbonr: Calculate Carbon-Equivalent Emissions*

Description

The carbonr package provides a flexible tool for calculating carbon-equivalent emissions.

Author(s)

Maintainer: Lily Clements <lily@idems.international> (ORCID)

See Also

See the README on GitHub

carbon_price_credit *Calculate carbon price credit*

Description

This function calculates the carbon price credit for a given jurisdiction, year, period, and CO2e value. It uses CPI (Carbon Price Index) data to determine the carbon price for the specified jurisdiction and time period. The carbon price credit is calculated by multiplying the CO2e value by the corresponding carbon price.

Usage

```
carbon_price_credit(
  jurisdiction = NULL,
  year = NULL,
  period = 0,
  manual_price = NULL,
  co2e_val
)
```

Arguments

jurisdiction	A character string specifying the jurisdiction for which the carbon price credit should be calculated.
year	An optional numeric value specifying the year for which the carbon price credit should be calculated. If NULL, the most recent year available in the CPI data will be used.
period	An optional numeric value specifying the period within the specified year for which the carbon price credit should be calculated. If 1, the function will use the first period if it is available; if 2, the function will use the second period if it is available. If 0, the function will calculate the mean between the first and second period.
manual_price	An option to manually input a carbon price index to override the value in the World Bank Data. This should be a value of the carbon credit price per tCO2e.
co2e_val	A numeric value specifying the CO2e (carbon dioxide equivalent) value for which the carbon price credit should be calculated.

Value

The calculated carbon price credit in USD ($).

Examples

```
# Calculate carbon price credit for the United Kingdom in the year 2000,
# period 2, and CO2e value of 100
carbon_price_credit("United Kingdom", 2022, 2, co2e_val = 100)
```

```
# Or manually enter a value
carbon_price_credit(manual_price = 66.9, co2e_val = 100)
```

---

check_CPI *Check which jurisdictions are in the Carbon Credits data*

Description

Find jurisdictions available in the Carbon Credits data. If a jurisdiction is specified, find the years associated with that jurisdiction.

Usage

```
check_CPI(jurisdiction = NULL, period = FALSE)
```

Arguments

jurisdiction	(optional) A character string specifying the jurisdiction to filter the data by.
period	(logical) If TRUE, include the Period column in the output data frame.

Value

A vector or data frame containing the information.

Examples

```
which_jur <- check_CPI()
which_years <- check_CPI(jurisdiction = "Switzerland")
which_years_and_period <- check_CPI(jurisdiction = "Switzerland", period = TRUE)
```

---

clinical_theatre_data *Clinical Emissions: Data Frame and Plot*

Description

Get the clinical theatre emissions after a data frame is inputted into the function.

Usage

```
clinical_theatre_data(
  data,
  time,
  date_format = c("%d/%m/%Y"),
  name,
  wet_clinical_waste = 0,
  wet_clinical_waste_unit = c("tonnes", "kg"),
  desflurane = 0,
  sevoflurane = 0,
  isoflurane = 0,
  methoxyflurane = 0,
  N2O = 0,
  propofol = 0,
  water_supply = 0,
  water_trt = TRUE,
  water_unit = c("cubic metres", "million litres"),
  electricity_kWh = 0,
  electricity_TD = TRUE,
  electricity_WTT = TRUE,
  heat_kWh = 0,
  heat_TD = TRUE,
  heat_WTT = TRUE,
  glass = 0,
  board = 0,
  mixed = 0,
  paper = 0,
  average = 0,
  average_film = 0,
  average_rigid = 0,
  HDPE = 0,
  LDPE = 0,
  LLDPE = 0,
  PET = 0,
  PP = 0,
  PS = 0,
  PVC = 0,
  glass_WD = 0,
 glass_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
  board_WD = 0,
  mixed_WD = 0,
  paper_WD = 0,
 paper_waste_disposal = c("Closed-loop", "Combustion", "Composting", "Landfill"),
  average_WD = 0,
  average_film_WD = 0,
  average_rigid_WD = 0,
  HDPE_WD = 0,
  LDPE_WD = 0,
```

```
  LLDPE_WD = 0,
  PET_WD = 0,
  PP_WD = 0,
  PS_WD = 0,
  PVC_WD = 0,
 plastic_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
  fridges = 0,
  freezers = 0,
  electric_waste_disposal = c("Landfill", "Open-loop"),
  glass_units = c("kg", "tonnes"),
  paper_units = c("kg", "tonnes"),
  plastic_units = c("kg", "tonnes"),
  electrical_units = c("kg", "tonnes"),
  include_cpi = FALSE,
  jurisdiction = NULL,
  year = NULL,
  period = 0,
  manual_price = NULL,
  gti_by = c("default", "month", "year"),
  overall_by = c("default", "month", "year"),
  single_sheet = FALSE
)
```

Arguments

data	Data frame containing all data to be used in the emissions calculation
time	Variable in data that corresponds to the time.
date_format	The date format for the time variable (optional, default: c("%d/%m/%Y")).
name	Variable in data that corresponds to the theatre name.
wet_clinical_waste	Amount of (wet) clinical waste that is usually incinerated.
wet_clinical_waste_unit	Unit for wet_clinical_waste variable. Options are "tonnes" or "kg".
desflurane	Amount of desflurane used in KG (default: 0).
sevoflurane	Amount of sevoflurane used in KG (default: 0).
isoflurane	Amount of isoflurane used in KG (default: 0).
methoxyflurane	Amount of methoxyflurane used in KG (default: 0).
N2O	Amount of nitrous oxide (N2O) used in KG (default: 0).
propofol	Amount of propofol used in KG (default: 0).
water_supply	Amount of water used in the building.
water_trt	logical. Default TRUE. Whether to include emissions associated with water treatment for used water.
water_unit	Unit for water_supply variable. Options are "cubic metres" or "million litres".

`electricity_kWh`
: Electricity used in kWh.

`electricity_TD`
: logical. Default `TRUE`. Whether to include emissions associated with grid losses.

`electricity_WTT`
: logical. Default `TRUE`. Whether to include emissions associated with extracting, refining, and transporting fuels.

`heat_kWh`
: heat and steam used in kWh.

`heat_TD`
: logical. Default `TRUE`. Whether to include emissions associated with grid losses.

`heat_WTT`
: logical. Default `TRUE`. Whether to include emissions associated with extracting, refining, and transporting fuels.

`glass`
: Numeric value representing the amount of glass. Default is `0`.

`board`
: Numeric value indicating the weight of paperboard. Default is `0`.

`mixed`
: Numeric value indicating the weight of mixed paper. Default is `0`.

`paper`
: Numeric value indicating the weight of paper. Default is `0`.

`average`
: Numeric value indicating the weight of average plastic. Default is `0`.

`average_film`
: Numeric value indicating the weight of average film plastic. Default is `0`.

`average_rigid`
: Numeric value indicating the weight of average rigid plastic. Default is `0`.

`HDPE`
: Numeric value indicating the weight of HDPE plastic. Default is `0`.

`LDPE`
: Numeric value indicating the weight of LDPE plastic. Default is `0`.

`LLDPE`
: Numeric value indicating the weight of LLDPE plastic. Default is `0`.

`PET`
: Numeric value indicating the weight of PET plastic. Default is `0`.

`PP`
: Numeric value indicating the weight of PP plastic. Default is `0`.

`PS`
: Numeric value indicating the weight of PS plastic. Default is `0`.

`PVC`
: Numeric value indicating the weight of PVC plastic. Default is `0`.

`glass_WD`
: Numeric value representing the amount of glass waste with disposal. Default is `0`.

`glass_waste_disposal`
: Character vector specifying the waste disposal method to use for metal for calculating emissions. Possible values: `"Closed-loop"`, `"Combustion"`, `"Landfill"`, `"Open-loop"`. Default is "Closed-loop". See `details` for more information.

`board_WD`
: Numeric value indicating the weight of paperboard disposed of using waste disposal methods. Default is `0`.

`mixed_WD`
: Numeric value indicating the weight of mixed paper disposed of using waste disposal methods. Default is `0`.

`paper_WD`
: Numeric value indicating the weight of paper disposed of using waste disposal methods. Default is `0`.

`paper_waste_disposal`
: Character vector specifying the waste disposal method for paper to use for calculating emissions. Possible values: `"Closed-loop"`, `"Combustion"`, `"Composting"`, `"Landfill"`. Default is `"Closed-loop"`. See `details` for more information.

`average_WD`
: Numeric value indicating the weight of average plastic disposed of using waste disposal methods. Default is `0`.

`average_film_WD`
: Numeric value indicating the weight of average film plastic disposed of using waste disposal methods. Default is `0`.

`average_rigid_WD`
: Numeric value indicating the weight of average rigid plastic disposed of using waste disposal methods. Default is `0`.

`HDPE_WD`
: Numeric value indicating the weight of HDPE plastic disposed of using waste disposal methods. Default is `0`.

`LDPE_WD`
: Numeric value indicating the weight of LDPE plastic disposed of using waste disposal methods. Default is `0`.

`LLDPE_WD`
: Numeric value indicating the weight of LLDPE plastic disposed of using waste disposal methods. Default is `0`.

`PET_WD`
: Numeric value indicating the weight of PET plastic disposed of using waste disposal methods. Default is `0`.

`PP_WD`
: Numeric value indicating the weight of PP plastic disposed of using waste disposal methods. Default is `0`.

`PS_WD`
: Numeric value indicating the weight of PS plastic disposed of using waste disposal methods. Default is `0`.

`PVC_WD`
: Numeric value indicating the weight of PVC plastic disposed of using waste disposal methods. Default is `0`.

`plastic_waste_disposal`
: Character vector specifying the waste disposal method for plastic to use for calculating emissions. Possible values: `"Closed-loop"`, `"Combustion"`, `"Landfill"`, `"Open-loop"`. Default is `"Closed-loop"`. See `details` for more information.

`fridges`
: Numeric value indicating the weight of fridges. Default is `0`.

`freezers`
: Numeric value indicating the weight of freezers. Default is `0`.

`electric_waste_disposal`
: Character vector specifying the waste disposal method for electrical items to use for calculating emissions. Possible values: `"Landfill"`, `"Open-loop"`. Default is `"Landfill"`. See `details` for more information.

`glass_units`
: Character vector specifying the units of the emissions related to glass. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

`paper_units`
: Character vector specifying the units of the emissions related to paper. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

`plastic_units`
: Character vector specifying the units of the emissions related to plastic materials. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

`electrical_units`
: Character vector specifying the units of the emissions related to electrical materials. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

`include_cpi`
: Logical variable specifying whether to calculate carbon price credit as well as emissions.

`jurisdiction`
: A character string specifying the jurisdiction for which the carbon price credit should be calculated. Available jurisdictions can be found by `check_CPI()`.

year	An optional numeric value specifying the year for which the carbon price credit should be calculated. If NULL, the most recent year available in the CPI data will be used.
period	An optional numeric value specifying the period within the specified year for which the carbon price credit should be calculated. If 1, the function will use the first period if it is available; if 2, the function will use the second period if it is available. If 0, the function will calculate the mean between the first and second period.
manual_price	An option to manually input a carbon price index to override the value in the World Bank Data.
gti_by	The grouping type for calculating the GTI ("default", "month", "year").
overall_by	The grouping type for the total output plot ("default", "month", "year"). This is a plot of the emissions if include_cpi = FALSE, otherwise is the CPI value.
single_sheet	Options are NULL, TRUE or FALSE. This is whether to give the summaries in a single sheet display, or as a list containing the table and ggplot2 objects. If NULL then no graphical output is given.

Value

Returns list containing two objects. A table containing CO2e emissions for each row of data (and carbon price index in USD if include_cpi is TRUE), and a ggplot2 object plotting the CO2e emissions. The second object is a single sheet containing summaries if single_sheet = TRUE.

Examples

```
# Example with dummy data
df <- data.frame(time = c("10/04/2000", "10/04/2000", "11/04/2000",
                          "11/04/2000", "12/04/2000", "12/04/2000"),
                 theatre = rep(c("A", "B"), times = 3),
                 clinical_waste = c(80, 90, 80, 100, 120, 110),
                 electricity_kwh = c(100, 110, 90, 100, 100, 110),
                 general_waste = c(65, 55, 70, 50, 60, 30))

clinical_theatre_data(df, time = time, name = theatre,
                 wet_clinical_waste = clinical_waste,
                 wet_clinical_waste_unit = "kg",
                 desflurane = 10,
                 average = general_waste,
                 plastic_units = "kg",
                 electricity_kWh = electricity_kwh,
                 include_cpi = TRUE,
                 jurisdiction = "Australia",
                 year = 2023,
                 single_sheet = FALSE)
```

clinical_theatre_emissions
Clinical Emissions

Description

Calculate carbon-equivalent emissions following an operating theatre

Usage

```
clinical_theatre_emissions(
  wet_clinical_waste,
  wet_clinical_waste_unit = c("tonnes", "kg"),
  desflurane = 0,
  sevoflurane = 0,
  isoflurane = 0,
  methoxyflurane = 0,
  N2O = 0,
  propofol = 0,
  water_supply = 0,
  water_trt = TRUE,
  water_unit = c("cubic metres", "million litres"),
  electricity_kWh = 0,
  electricity_TD = TRUE,
  electricity_WTT = TRUE,
  heat_kWh = 0,
  heat_TD = TRUE,
  heat_WTT = TRUE,
  glass = 0,
  board = 0,
  mixed = 0,
  paper = 0,
  average = 0,
  average_film = 0,
  average_rigid = 0,
  HDPE = 0,
  LDPE = 0,
  LLDPE = 0,
  PET = 0,
  PP = 0,
  PS = 0,
  PVC = 0,
  glass_WD = 0,
 glass_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
  board_WD = 0,
  mixed_WD = 0,
  paper_WD = 0,
```

```
  fridges = 0,
  freezers = 0,
  electric_waste_disposal = c("Landfill", "Open-loop"),
  electrical_units = c("kg", "tonnes"),
 paper_waste_disposal = c("Closed-loop", "Combustion", "Composting", "Landfill"),
  average_WD = 0,
  average_film_WD = 0,
  average_rigid_WD = 0,
  HDPE_WD = 0,
  LDPE_WD = 0,
  LLDPE_WD = 0,
  PET_WD = 0,
  PP_WD = 0,
  PS_WD = 0,
  PVC_WD = 0,
 plastic_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
  glass_units = c("kg", "tonnes"),
  paper_units = c("kg", "tonnes"),
  plastic_units = c("kg", "tonnes")
)
```

Arguments

`wet_clinical_waste`	Amount of (wet) clinical waste that is usually incinerated.
`wet_clinical_waste_unit`	Unit for `wet_clinical_waste` variable. Options are `"tonnes"` or `"kg"`.
`desflurane`	Amount of desflurane used in KG (default: 0).
`sevoflurane`	Amount of sevoflurane used in KG (default: 0).
`isoflurane`	Amount of isoflurane used in KG (default: 0).
`methoxyflurane`	Amount of methoxyflurane used in KG (default: 0).
`N2O`	Amount of nitrous oxide (N2O) used in KG (default: 0).
`propofol`	Amount of propofol used in KG (default: 0).
`water_supply`	Amount of water used in the building.
`water_trt`	logical. Default `TRUE`. Whether to include emissions associated with water treatment for used water.
`water_unit`	Unit for `water_supply` variable. Options are `"cubic metres"` or `"million litres"`.
`electricity_kWh`	Electricity used in kWh.
`electricity_TD`	logical. Default `TRUE`. Whether to include emissions associated with grid losses.
`electricity_WTT`	logical. Default `TRUE`. Whether to include emissions associated with extracting, refining, and transporting fuels.
`heat_kWh`	heat and steam used in kWh.

heat_TD logical. Default TRUE. Whether to include emissions associated with grid losses.

heat_WTT logical. Default TRUE. Whether to include emissions associated with extracting, refining, and transporting fuels.

glass Numeric value representing the amount of glass. Default is 0.

board Numeric value indicating the weight of paperboard. Default is 0.

mixed Numeric value indicating the weight of mixed paper. Default is 0.

paper Numeric value indicating the weight of paper. Default is 0.

average Numeric value indicating the weight of average plastic. Default is 0.

average_film Numeric value indicating the weight of average film plastic. Default is 0.

average_rigid Numeric value indicating the weight of average rigid plastic. Default is 0.

HDPE Numeric value indicating the weight of HDPE plastic. Default is 0.

LDPE Numeric value indicating the weight of LDPE plastic. Default is 0.

LLDPE Numeric value indicating the weight of LLDPE plastic. Default is 0.

PET Numeric value indicating the weight of PET plastic. Default is 0.

PP Numeric value indicating the weight of PP plastic. Default is 0.

PS Numeric value indicating the weight of PS plastic. Default is 0.

PVC Numeric value indicating the weight of PVC plastic. Default is 0.

glass_WD Numeric value representing the amount of glass waste with disposal. Default is 0.

glass_waste_disposal
Character vector specifying the waste disposal method to use for metal for calculating emissions. Possible values: "Closed-loop", "Combustion", "Landfill", "Open-loop". Default is "Closed-loop". See details for more information.

board_WD Numeric value indicating the weight of paperboard disposed of using waste disposal methods. Default is 0.

mixed_WD Numeric value indicating the weight of mixed paper disposed of using waste disposal methods. Default is 0.

paper_WD Numeric value indicating the weight of paper disposed of using waste disposal methods. Default is 0.

fridges Numeric value indicating the weight of fridges. Default is 0.

freezers Numeric value indicating the weight of freezers. Default is 0.

electric_waste_disposal
Character vector specifying the waste disposal method for electrical items to use for calculating emissions. Possible values: "Landfill", "Open-loop". Default is "Landfill". See details for more information.

electrical_units
Character vector specifying the units of the emissions related to electrical materials. Possible values: "kg", "tonnes". Default is "kg".

paper_waste_disposal
Character vector specifying the waste disposal method for paper to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Composting", "Landfill". Default is "Closed-loop". See details for more information.

`average_WD` Numeric value indicating the weight of average plastic disposed of using waste disposal methods. Default is `0`.

`average_film_WD` Numeric value indicating the weight of average film plastic disposed of using waste disposal methods. Default is `0`.

`average_rigid_WD` Numeric value indicating the weight of average rigid plastic disposed of using waste disposal methods. Default is `0`.

`HDPE_WD` Numeric value indicating the weight of HDPE plastic disposed of using waste disposal methods. Default is `0`.

`LDPE_WD` Numeric value indicating the weight of LDPE plastic disposed of using waste disposal methods. Default is `0`.

`LLDPE_WD` Numeric value indicating the weight of LLDPE plastic disposed of using waste disposal methods. Default is `0`.

`PET_WD` Numeric value indicating the weight of PET plastic disposed of using waste disposal methods. Default is `0`.

`PP_WD` Numeric value indicating the weight of PP plastic disposed of using waste disposal methods. Default is `0`.

`PS_WD` Numeric value indicating the weight of PS plastic disposed of using waste disposal methods. Default is `0`.

`PVC_WD` Numeric value indicating the weight of PVC plastic disposed of using waste disposal methods. Default is `0`.

`plastic_waste_disposal` Character vector specifying the waste disposal method for plastic to use for calculating emissions. Possible values: `"Closed-loop"`, `"Combustion"`, `"Landfill"`, `"Open-loop"`. Default is `"Closed-loop"`. See `details` for more information.

`glass_units` Character vector specifying the units of the emissions related to glass. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

`paper_units` Character vector specifying the units of the emissions related to paper. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

`plastic_units` Character vector specifying the units of the emissions related to plastic materials. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

Value

Returns CO2e emissions in tonnes.

Examples

```
# clinical theatre emissions for the default options, with 100kg of wet clinical waste
clinical_theatre_emissions(wet_clinical_waste = 100, wet_clinical_waste_unit = "kg")
```

construction_emissions *Calculate emissions from construction*

Description

: This function calculates the construction emissions based on the input parameters.

Usage

```
construction_emissions(
  aggregates = 0,
  average = 0,
  asbestos = 0,
  asphalt = 0,
  bricks = 0,
  concrete = 0,
  insulation = 0,
  metals = 0,
  soils = 0,
  mineral_oil = 0,
  plasterboard = 0,
  tyres = 0,
  wood = 0,
  aggregates_WD = 0,
  average_WD = 0,
  asbestos_WD = 0,
  asphalt_WD = 0,
  bricks_WD = 0,
  concrete_WD = 0,
  insulation_WD = 0,
  metals_WD = 0,
  soils_WD = 0,
  mineral_oil_WD = 0,
  plasterboard_WD = 0,
  tyres_WD = 0,
  wood_WD = 0,
  units = c("kg", "tonnes"),
 waste_disposal = c("Closed-loop", "Combustion", "Composting", "Landfill", "Open-loop")
)
```

Arguments

aggregates	The weight of aggregates used in construction. Default is 0.
average	The weight of average material used in construction. Default is 0.
asbestos	The weight of asbestos used in construction. Default is 0.

asphalt	The weight of asphalt used in construction. Default is 0.
bricks	The weight of bricks used in construction. Default is 0.
concrete	The weight of concrete used in construction. Default is 0.
insulation	The weight of insulation material used in construction. Default is 0.
metals	The weight of metals used in construction. Default is 0.
soils	The weight of soils used in construction. Default is 0.
mineral_oil	The weight of mineral oil used in construction. Default is 0.
plasterboard	The weight of plasterboard used in construction. Default is 0.
tyres	The weight of tyres used in construction. Default is 0.
wood	The weight of wood used in construction. Default is 0.
aggregates_WD	The weight of aggregates disposed of as waste. Default is 0.
average_WD	The weight of average material disposed of as waste. Default is 0.
asbestos_WD	The weight of asbestos disposed of as waste. Default is 0.
asphalt_WD	The weight of asphalt disposed of as waste. Default is 0.
bricks_WD	The weight of bricks disposed of as waste. Default is 0.
concrete_WD	The weight of concrete disposed of as waste. Default is 0.
insulation_WD	The weight of insulation material disposed of as waste. Default is 0.
metals_WD	The weight of metals disposed of as waste. Default is 0.
soils_WD	The weight of soils disposed of as waste. Default is 0.
mineral_oil_WD	The weight of mineral oil disposed of as waste. Default is 0.
plasterboard_WD	The weight of plasterboard disposed of as waste. Default is 0.
tyres_WD	The weight of tyres disposed of as waste. Default is 0.
wood_WD	The weight of wood disposed of as waste. Default is 0.
units	The units in which the emissions should be returned ("kg" or "tonnes"). Default is 0.
waste_disposal	The method of waste disposal. Options are, "Closed-loop", "Combustion", "Composting", "Landfill", "Open-loop". Default is "Closed-loop". See details for more information on this.

Details

The function calculates the construction emissions based on the input quantities of different materials used in construction and the quantities of those materials disposed of as waste. The emissions values are obtained from a data source and are multiplied by the corresponding quantities to calculate the total emissions. The units of emissions can be specified as either kilograms (kg) or tonnes.

All assume Primary material production for the material used in construction, except soils which assumes Closed-loop

The waste disposal method can be selected from the options: "Closed-loop", "Combustion", "Composting", "Landfill", or "Open-loop". Note that: "Closed-loop" is valid for aggregates, average, asphalt, concrete, insulation, metal, soils, mineral oil, plasterboard, tyres, and wood.

"Combustion" is valid for average, mineral oil, and wood. "Composting" is valid for wood only. "Landfill" is valid for everything except average, mineral oil, and tyres. "Open-loop" is valid for aggregates, average, asphalt, bricks, concrete, If one of these is used for a value that does not provide it, then an "NA" is given.

Value

The calculated construction emissions as a numeric value in tonnes.

Examples

```
#Calculate construction emissions with default values
construction_emissions()

#Calculate construction emissions with specified quantities
construction_emissions(aggregates = 1000, concrete = 500, wood = 2000,
                       units = "kg", waste_disposal = "Landfill")
```

---

degree_conversion *Convert degrees to radians*

Description

Convert degrees to radians.

Usage

```
degree_conversion(deg)
```

Arguments

deg	Degree value to convert.

Value

Value in radians.

Examples

```
# Convert 90 degrees into radians
degree_conversion(90)
```

distance_calc *Distance calculator*

Description

Calculate distances between locations in miles using the Haversine formula. This is calculated as the crow flies.

Usage

```
distance_calc(lat1, lat2, long1, long2)
```

Arguments

lat1	Latitude of the first location.
lat2	Latitude of the second location.
long1	Longitude of the first location.
long2	Longitude of the second location.

Value

Distance between locations in miles

Examples

```
# Distance between the London Eye and the Eiffel Tower in miles
distance_calc(51.5033, 48.8584, 0.1196, 2.2945)
```

electrical_emissions *Calculate Electrical Emissions*

Description

This function calculates the emissions produced from different electrical items and their waste disposal based on the specified inputs. It considers emissions from primary material production and waste disposal of electrical items.

Usage

```
electrical_emissions(
  fridges = 0,
  freezers = 0,
  large = 0,
  IT = 0,
  small = 0,
  alkaline_batteries = 0,
  LiIon_batteries = 0,
  NiMh_batteries = 0,
  fridges_WD = 0,
  freezers_WD = 0,
  large_WD = 0,
  IT_WD = 0,
  small_WD = 0,
  alkaline_batteries_WD = 0,
  LiIon_batteries_WD = 0,
  NiMh_batteries_WD = 0,
  waste_disposal = c("Landfill", "Open-loop"),
  units = c("kg", "tonnes")
)
```

Arguments

`fridges`	Numeric value indicating the weight of fridges. Default is `0`.
`freezers`	Numeric value indicating the weight of freezers. Default is `0`.
`large`	Numeric value indicating the weight of large electrical items. Default is `0`.
`IT`	Numeric value indicating the weight of IT (Information Technology) equipment. Default is `0`.
`small`	Numeric value indicating the weight of small electrical items. Default is `0`.
`alkaline_batteries`	Numeric value indicating the weight of alkaline batteries. Default is `0`.
`LiIon_batteries`	Numeric value indicating the weight of Lithium-ion batteries. Default is `0`.
`NiMh_batteries`	Numeric value indicating the weight of Nickel Metal Hydride batteries. Default is `0`.
`fridges_WD`	Numeric value indicating the weight of fridges disposed of using waste disposal methods. Default is `0`.
`freezers_WD`	Numeric value indicating the weight of freezers disposed of using waste disposal methods. Default is `0`.
`large_WD`	Numeric value indicating the weight of large electrical items disposed of using waste disposal methods. Default is `0`.
`IT_WD`	Numeric value indicating the weight of IT equipment disposed of using waste disposal methods. Default is `0`.

`small_WD` Numeric value indicating the weight of small electrical items disposed of using waste disposal methods. Default is `0`.

`alkaline_batteries_WD` Numeric value indicating the weight of alkaline batteries disposed of using waste disposal methods. Default is `0`.

`LiIon_batteries_WD` Numeric value indicating the weight of Lithium-ion batteries disposed of using waste disposal methods. Default is `0`.

`NiMh_batteries_WD` Numeric value indicating the weight of Nickel Metal Hydride batteries disposed of using waste disposal methods. Default is `0`.

`waste_disposal` Character vector specifying the waste disposal method to use for calculating emissions. Possible values: `"Landfill"`, `"Open-loop"`. Default is `"Landfill"`. `"Open-loop"` is the process of recycling material into other products. `"Landfill"` the product goes to landfill after use.

`units` Character vector specifying the units of the emissions output. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

Value

The calculated electrical emissions as a numeric value in tonnes.

Examples

```
# Calculate electrical emissions using default values
electrical_emissions()

# Calculate electrical emissions with specific quantities and waste disposal
# method
electrical_emissions(fridges = 10, IT = 5, alkaline_batteries = 100,
                     waste_disposal = "Open-loop", units = "tonnes")
```

---

example_clinical_theatre

Clinical Theatre Data

Description

This dataset contains activities within a clinical theatre setting related to healthcare services or medical supplies. It is structured to facilitate analysis of healthcare operations, costs, and estimate CO2e emissions.

Usage

```
example_clinical_theatre
```

Format

A data frame with 4,386 rows and 6 columns, including:

time The date of transaction or activity in DDMMYYYY format.

prices The cost associated with each transaction or activity in AUD, related to services provided or medical supplies used.

quantities The amount or volume of a product or service provided, indicating the number of items used or procedures performed.

prodID A unique identifier for each type of product or service, such as medical supplies, medications, or surgical procedures.

retID A code identifying the department, supplier, or healthcare provider involved in the transaction, or the entity receiving the product or service.

description Categorises the clinical theatre activities into medical specialties or types of procedures, with 'ent' (Ear, Nose, and Throat), 'colorectal', 'hbt' (hematology or blood treatment), 'ortho' (orthopedics), 'paedsurg' (pediatric surgery), and 'gensurg' (general surgery).

Details

The dataset provides an overview of clinical theatre operations.

`ferry_emissions` *Calculate CO2e emissions from ferry journeys*

Description

A function that calculates CO2e emissions between ferry ports.

Usage

```
ferry_emissions(
  from,
  to,
  via = NULL,
  type = c("Foot", "Car", "Average"),
  num_people = 1,
  times_journey = 1,
  include_WTT = TRUE,
  round_trip = FALSE
)
```

Arguments

from	Port code for the port departing from. Use seaport_finder to find port code.
to	Port code for the port arriving from. Use seaport_finder to find port code.
via	Optional. Takes a vector containing the port code that the ferry travels through. Use seaport_finder to find port code.
type	Whether the journey is taken on foot or by car. Options are "Foot", "Car", "Average".
num_people	Number of people taking the journey. Takes a single numerical value.
times_journey	Number of times the journey is taken.
include_WTT	logical. Recommended TRUE. Whether to include emissions associated with extracting, refining, and transporting fuels.
round_trip	Whether the journey is one-way or return.

Details

The distances are calculated using the Haversine formula. This is calculated as the crow flies.

Value

Returns CO2e emissions in tonnes for the ferry journey.

Examples

```
# Emissions for a ferry journey between Belfast and New York City
seaport_finder(city = "Belfast")
seaport_finder(city = "New York")
ferry_emissions(from = "BEL", to = "BOY")
```

---

gg_value_box *Create a Value Box for Reports*

Description

This function creates a value box for use in reports.

Usage

```
gg_value_box(values, information, icons)
```

Arguments

values	A vector of numeric values to be displayed in the value box.
information	A vector of strings providing information or labels for the values.
icons	A vector of Font Awesome unicode symbols to be displayed as icons.

Details

This function creates a value box with customizable values, information, and icons. The function takes inputs for the values, information, icons, and color of the value box. The values and information are provided as vectors, while the icons are specified using Font Awesome unicode symbols. The color of the value box can be customized using a factor variable. The resulting value box is a ggplot2 object that can be further customized or combined with other plots or elements in a report.

Value

A ggplot2 object with a value box for report use.

References

Modified from Stack Overflow post: https://stackoverflow.com/questions/47105282/valuebox-like-function-for-static-reports

Examples

```
# Create a value box with custom values and icons
gg_value_box(
  values = c(100, 500, 1000),
  information = c("Sales", "Revenue", "Customers"),
  icons = c("\U0000f155", "\U0000f155", "\U0000f0f7")
)
```

---

hotel_emissions *Calculate CO2e emissions from a hotel stay*

Description

Indirect emissions from a stay at a hotel. Values to calculate emissions are from UK government 2022 report.

Usage

```
hotel_emissions(location = "UK", nights = 1, rooms = 1)
```

Arguments

location	Location of the hotel stay. Current accepted locations are "UK", "UK (London)" "Argentina", "Australia", "Austria", "Belgium", "Brazil", "Canada", "Chile", "China", "Colombia", "Costa Rica", "Czechia", "Egypt", "Fiji", "France", "Germany", "Greece", "Hong Kong, China", "India", "Indonesia", "Ireland", "Israel", "Italy", "Japan", "Jordan", "Korea", "Macau", "Malaysia", "Maldives", "Mexico", "Netherlands", "New Zealand", "Oman", "Panama", "Peru", "Philippines", "Poland", "Portugal", "Qatar", "Romania", "Russia", "Saudi Arabia", "Singapore", "Slovakia", "South Africa", "Spain", "Switzerland", "Taiwan", "Thailand", "Turkey", "United Arab Emirates", "United States", "Vietnam".

nights	Number of nights stayed in the hotel.
rooms	Number of rooms used in the hotel.

Value

Tonnes of CO2e emissions for a stay in a hotel.

Examples

```
# Emissions for a two night stay in Fiji.
hotel_emissions(location = "Fiji", nights = 2)
```

---

household_emissions *Calculate household material emissions*

Description

Calculate household material emissions

Usage

```
household_emissions(
  glass = 0,
  clothing = 0,
  food = 0,
  drink = 0,
  compost_from_garden = 0,
  compost_from_food_and_garden = 0,
  board = 0,
  mixed = 0,
  paper = 0,
  fridges = 0,
  freezers = 0,
  large_electrical = 0,
  IT = 0,
  small_electrical = 0,
  alkaline_batteries = 0,
  LiIon_batteries = 0,
  NiMh_batteries = 0,
  aluminuim_cans = 0,
  aluminuim_foil = 0,
  mixed_cans = 0,
  scrap = 0,
  steel_cans = 0,
  average = 0,
  average_film = 0,
  average_rigid = 0,
```

```
 HDPE = 0,
 LDPE = 0,
 LLDPE = 0,
 PET = 0,
 PP = 0,
 PS = 0,
 PVC = 0,
 glass_WD = 0,
 books_WD = 0,
 clothing_WD = 0,
 gcb_waste_disposal = c("Closed-loop", "Combustion", "Landfill"),
 household_residual_waste = 0,
 hh_waste_disposal = c("Combustion", "Landfill"),
 food_WD = 0,
 drink_WD = 0,
 compost_from_garden_WD = 0,
 compost_from_food_and_garden_WD = 0,
 compost_waste_disposal = c("Anaerobic digestion", "Combustion", "Composting",
   "Landfill"),
 aluminuim_cans_WD = 0,
 aluminuim_foil_WD = 0,
 mixed_cans_WD = 0,
 scrap_WD = 0,
 steel_cans_WD = 0,
metal_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
 board_WD = 0,
 mixed_WD = 0,
 paper_WD = 0,
paper_waste_disposal = c("Closed-loop", "Combustion", "Composting", "Landfill"),
 average_WD = 0,
 average_film_WD = 0,
 average_rigid_WD = 0,
 HDPE_WD = 0,
 LDPE_WD = 0,
 LLDPE_WD = 0,
 PET_WD = 0,
 PP_WD = 0,
 PS_WD = 0,
 PVC_WD = 0,
plastic_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
 fridges_WD = 0,
 freezers_WD = 0,
 large_electrical_WD = 0,
 IT_WD = 0,
 small_electrical_WD = 0,
 alkaline_batteries_WD = 0,
 LiIon_batteries_WD = 0,
 NiMh_batteries_WD = 0,
```

```
  electric_waste_disposal = c("Landfill", "Open-loop"),
  units = c("kg", "tonnes")
)
```

Arguments

glass	Numeric value representing the amount of glass. Default is 0.
clothing	Numeric value representing the amount of clothing. Default is 0.
food	Numeric value representing the amount of food. Default is 0.
drink	Numeric value representing the amount of drink. Default is 0.
compost_from_garden	Numeric value representing the amount of compost from garden waste. Default is 0.
compost_from_food_and_garden	Numeric value representing the amount of compost from food and garden waste. Default is 0.
board	Numeric value indicating the weight of paperboard. Default is 0.
mixed	Numeric value indicating the weight of mixed paper. Default is 0.
paper	Numeric value indicating the weight of paper. Default is 0.
fridges	Numeric value indicating the weight of fridges. Default is 0.
freezers	Numeric value indicating the weight of freezers. Default is 0.
large_electrical	Numeric value indicating the weight of large electrical items. Default is 0.
IT	Numeric value indicating the weight of IT (Information Technology) equipment. Default is 0.
small_electrical	Numeric value indicating the weight of small electrical items. Default is 0.
alkaline_batteries	Numeric value indicating the weight of alkaline batteries. Default is 0.
LiIon_batteries	Numeric value indicating the weight of Lithium-ion batteries. Default is 0.
NiMh_batteries	Numeric value indicating the weight of Nickel Metal Hydride batteries. Default is 0.
aluminuim_cans	Numeric value indicating the weight of aluminum cans. Default is 0.
aluminuim_foil	Numeric value indicating the weight of aluminum foil. Default is 0.
mixed_cans	Numeric value indicating the weight of mixed metal cans. Default is 0.
scrap	Numeric value indicating the weight of metal scrap. Default is 0.
steel_cans	Numeric value indicating the weight of steel cans. Default is 0.
average	Numeric value indicating the weight of average plastic. Default is 0.
average_film	Numeric value indicating the weight of average film plastic. Default is 0.
average_rigid	Numeric value indicating the weight of average rigid plastic. Default is 0.
HDPE	Numeric value indicating the weight of HDPE plastic. Default is 0.

LDPE Numeric value indicating the weight of LDPE plastic. Default is 0.

LLDPE Numeric value indicating the weight of LLDPE plastic. Default is 0.

PET Numeric value indicating the weight of PET plastic. Default is 0.

PP Numeric value indicating the weight of PP plastic. Default is 0.

PS Numeric value indicating the weight of PS plastic. Default is 0.

PVC Numeric value indicating the weight of PVC plastic. Default is 0.

glass_WD Numeric value representing the amount of glass waste with disposal. Default is 0.

books_WD Numeric value representing the amount of books waste with disposal. Default is 0.

clothing_WD Numeric value representing the amount of clothing waste with disposal. Default is 0.

gcb_waste_disposal
Character value specifying the waste disposal method for glass, clothing, and books waste (options: "Closed-loop", "Combustion", "Landfill"). Default is "Closed-loop". See details for more information.

household_residual_waste
Numeric value representing the amount of household residual waste. Default is 0.

hh_waste_disposal
Character value specifying the waste disposal method for waste (options: "Combustion", "Landfill"). Default is "Combustion". See details for more information.

food_WD Numeric value indicating the weight of food disposed of using waste disposal methods. Default is 0.

drink_WD Numeric value indicating the weight of drink disposed of using waste disposal methods. Default is 0.

compost_from_garden_WD
Numeric value indicating the weight of compost from garden waste disposed of using waste disposal methods. Default is 0.

compost_from_food_and_garden_WD
Numeric value indicating the weight of compost from garden and food waste disposed of using waste disposal methods. Default is 0.

compost_waste_disposal
Character value specifying the waste disposal method for compost waste (options: "Anaerobic digestion", "Combustion", "Composting", "Landfill"). Default is "Anaerobic digestion". See details for more information.

aluminuim_cans_WD
Numeric value indicating the weight of aluminum cans disposed of using waste disposal methods. Default is 0.

aluminuim_foil_WD
Numeric value indicating the weight of aluminum foil disposed of using waste disposal methods. Default is 0.

mixed_cans_WD Numeric value indicating the weight of mixed metal cans disposed of using waste disposal methods. Default is 0.

scrap_WD Numeric value indicating the weight of metal scrap disposed of using waste disposal methods. Default is 0.

steel_cans_WD Numeric value indicating the weight of steel cans disposed of using waste disposal methods. Default is 0.

metal_waste_disposal
Character vector specifying the waste disposal method to use for metal for calculating emissions. Possible values: "Closed-loop", "Combustion", "Landfill", "Open-loop". Default is "Closed-loop". See details for more information.

board_WD Numeric value indicating the weight of paperboard disposed of using waste disposal methods. Default is 0.

mixed_WD Numeric value indicating the weight of mixed paper disposed of using waste disposal methods. Default is 0.

paper_WD Numeric value indicating the weight of paper disposed of using waste disposal methods. Default is 0.

paper_waste_disposal
Character vector specifying the waste disposal method for paper to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Composting", "Landfill". Default is "Closed-loop". See details for more information.

average_WD Numeric value indicating the weight of average plastic disposed of using waste disposal methods. Default is 0.

average_film_WD
Numeric value indicating the weight of average film plastic disposed of using waste disposal methods. Default is 0.

average_rigid_WD
Numeric value indicating the weight of average rigid plastic disposed of using waste disposal methods. Default is 0.

HDPE_WD Numeric value indicating the weight of HDPE plastic disposed of using waste disposal methods. Default is 0.

LDPE_WD Numeric value indicating the weight of LDPE plastic disposed of using waste disposal methods. Default is 0.

LLDPE_WD Numeric value indicating the weight of LLDPE plastic disposed of using waste disposal methods. Default is 0.

PET_WD Numeric value indicating the weight of PET plastic disposed of using waste disposal methods. Default is 0.

PP_WD Numeric value indicating the weight of PP plastic disposed of using waste disposal methods. Default is 0.

PS_WD Numeric value indicating the weight of PS plastic disposed of using waste disposal methods. Default is 0.

PVC_WD Numeric value indicating the weight of PVC plastic disposed of using waste disposal methods. Default is 0.

plastic_waste_disposal
Character vector specifying the waste disposal method for plastic to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Landfill", "Open-loop". Default is "Closed-loop". See details for more information.

`fridges_WD` Numeric value indicating the weight of fridges disposed of using waste disposal methods. Default is `0`.

`freezers_WD` Numeric value indicating the weight of freezers disposed of using waste disposal methods. Default is `0`.

`large_electrical_WD` Numeric value indicating the weight of large electrical items disposed of using waste disposal methods. Default is `0`.

`IT_WD` Numeric value indicating the weight of IT equipment disposed of using waste disposal methods. Default is `0`.

`small_electrical_WD` Numeric value indicating the weight of small electrical items disposed of using waste disposal methods. Default is `0`.

`alkaline_batteries_WD` Numeric value indicating the weight of alkaline batteries disposed of using waste disposal methods. Default is `0`.

`LiIon_batteries_WD` Numeric value indicating the weight of Lithium-ion batteries disposed of using waste disposal methods. Default is `0`.

`NiMh_batteries_WD` Numeric value indicating the weight of Nickel Metal Hydride batteries disposed of using waste disposal methods. Default is `0`.

`electric_waste_disposal` Character vector specifying the waste disposal method for electrical items to use for calculating emissions. Possible values: `"Landfill"`, `"Open-loop"`. Default is `"Landfill"`. See `details` for more information.

`units` Character vector specifying the units of the emissions output. Possible values: `"kg"`, `"tonnes"`. Default is `"kg"`.

Details

`*_waste_disposal` methods: `"Open-loop"` is the process of recycling material into other products. `"Closed-loop"` is the process of recycling material back into the same product. `"Combustion"` energy is recovered from the waste through incineration and subsequent generation of electricity. `"Compost"` CO2e emitted as a result of composting a waste stream. `"Landfill"` the product goes to landfill after use. `"Anaerobic digestion"` energy is recovered from the waste through anaerobic digestion.

Value

The calculated household emissions as a numeric value in tonnes.

Examples

```
household_emissions(glass = 100, clothing = 10, glass_WD = 10, units = "kg")
```

import_CPI *Import CPI data from an Excel file*

Description

This function uses the downloaded data from the World Bank Carbon Pricing Dashboard (https://carbonpricingdashboard.wor
It imports data from an Excel file containing CPI (Carbon Price Index) data. It filters the data for ETS (Emissions Trading Scheme) instruments, performs necessary transformations, and returns a processed data frame.

Usage

```
import_CPI(path, sheet = "Data_Price", skip = 2)
```

Arguments

path	A character string specifying the file path of the Excel file.
sheet	A character string specifying the name of the sheet in the Excel file to read data from.
skip	An integer specifying the number of rows to skip while reading the Excel sheet.

Value

A processed data frame containing CPI data.

land_emissions *Calculate CO2e emissions from land-travel journeys*

Description

A function that calculates CO2e emissions on a journey on land.

Usage

```
land_emissions(
  distance,
  units = c("miles", "km"),
  num = 1,
 vehicle = c("Cars", "Motorbike", "Taxis", "Bus", "National rail", "International rail",
    "Light rail and tram", "London Underground", "Coach"),
  fuel = c("Petrol", "Diesel", "Unknown", "Battery Electric Vehicle",
    "Plug-in Hybrid Electric Vehicle"),
  car_type = c("Average car", "Small car", "Medium car", "Large car", "Mini",
   "Supermini", "Lower medium", "Upper medium", "Executive", "Luxury", "Sports",
    "Dual purpose 4X4", "MPV"),
```

```
  bike_type = c("Average", "Small", "Medium", "Large"),
  bus_type = c("Local bus (not London)", "Local London bus", "Average local bus"),
  taxi_type = c("Regular taxi", "Black cab"),
  TD = TRUE,
  include_WTT = TRUE,
  include_electricity = TRUE,
  owned_by_org = TRUE
)
```

Arguments

distance	Distance in km or miles of the journey made (this can be calculated with other tools, such as google maps.).
units	Units for the distance travelled. Options are "km" or "miles".
num	Number of passengers if vehicle is one of coach, tram, or tube. Otherwise, number of vehicles used.
vehicle	Vehicle used for the journey. Options are "Cars", "Motorbike", "Taxis", "Bus", "National rail", "International rail, "Coach", "Light rail and tram", "London Underground". Note: car, taxi, motorbike is per vehicle.
fuel	Fuel type used for the journey. For car, "Petrol", "Diesel", "Unknown", "Battery Electric Vehicle", "Plug-in Hybrid Electric Vehicle" are options. ##' "hybrid electric" and "battery electric" account for electricity kWh emissions.
car_type	Size/type of vehicle for car. Options are c("Average car", "Small car", "Medium car", "Large car" "Mini", "Supermini", "Lower medium", "Upper medium", "Executive", "Luxury", "Sports", "Dual purpose 4X4", "MPV"),. Small denotes up to a 1.4L engine, unless diesel which is up to 1.7L engine. Medium denotes 1.4-2.0L for petrol cars, 1.7-2.0L for diesel cars. Large denotes 2.0L+ engine.
bike_type	Size of vehicle for motorbike. Options are "Small", "Medium", "Large", or "Average". Sizes denote upto 125cc, 125cc-500cc, 500cc+ respectively.
bus_type	Options are "local_nL", "local_L", "local", or "average". These denote whether the bus is local but outside of London, local in London, local, or average.
taxi_type	Whether a taxi is regular or black cab. Options are "Regular taxi", "Black cab".
TD	logical.Whether to account for transmission and distribution (TD) for electric vehicles (only car and van)
include_WTT	logical. Well-to-tank (include_WTT) - whether to account for emissions associated with extraction, refining and transportation of the fuels (for non-electric vehicles).
include_electricity	logical. Whether to account for ... for electric vehicles (car and van).
owned_by_org	logical. Whether the vehicle used is owned by the organisation or not (only for car, motorbike).

Value

Tonnes of CO2e emissions per mile travelled.

Examples

```
# Emissions for a 100 mile car journey
land_emissions(distance = 100)

# Emissions for a 100 mile motorbike journey where the motorbike is 500+cc
land_emissions(distance = 100, vehicle = "Motorbike", bike_type = "Large")
```

---

material_emissions *Material (and waste) emissions*

Description

Material (and waste) emissions

Usage

```
material_emissions(
  glass = 0,
  board = 0,
  mixed = 0,
  paper = 0,
  fridges = 0,
  freezers = 0,
  large_electrical = 0,
  IT = 0,
  small_electrical = 0,
  alkaline_batteries = 0,
  LiIon_batteries = 0,
  NiMh_batteries = 0,
  aluminuim_cans = 0,
  aluminuim_foil = 0,
  mixed_cans = 0,
  scrap = 0,
  steel_cans = 0,
  average = 0,
  average_film = 0,
  average_rigid = 0,
  HDPE = 0,
  LDPE = 0,
  LLDPE = 0,
  PET = 0,
  PP = 0,
  PS = 0,
```

```
 PVC = 0,
 glass_WD = 0,
glass_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
 industrial_waste = 0,
 industrial_waste_disposal = c("Combustion", "Landfill"),
 aluminuim_cans_WD = 0,
 aluminuim_foil_WD = 0,
 mixed_cans_WD = 0,
 scrap_WD = 0,
 steel_cans_WD = 0,
metal_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
 board_WD = 0,
 mixed_WD = 0,
 paper_WD = 0,
paper_waste_disposal = c("Closed-loop", "Combustion", "Composting", "Landfill"),
 average_WD = 0,
 average_film_WD = 0,
 average_rigid_WD = 0,
 HDPE_WD = 0,
 LDPE_WD = 0,
 LLDPE_WD = 0,
 PET_WD = 0,
 PP_WD = 0,
 PS_WD = 0,
 PVC_WD = 0,
plastic_waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
 fridges_WD = 0,
 freezers_WD = 0,
 large_electrical_WD = 0,
 IT_WD = 0,
 small_electrical_WD = 0,
 alkaline_batteries_WD = 0,
 LiIon_batteries_WD = 0,
 NiMh_batteries_WD = 0,
 electric_waste_disposal = c("Landfill", "Open-loop"),
 aggregates = 0,
 construction_average = 0,
 asbestos = 0,
 asphalt = 0,
 bricks = 0,
 concrete = 0,
 insulation = 0,
 metals = 0,
 soils = 0,
 mineral_oil = 0,
 plasterboard = 0,
 tyres = 0,
 wood = 0,
```

```
  aggregates_WD = 0,
  construction_average_WD = 0,
  asbestos_WD = 0,
  asphalt_WD = 0,
  bricks_WD = 0,
  concrete_WD = 0,
  insulation_WD = 0,
  metals_WD = 0,
  soils_WD = 0,
  mineral_oil_WD = 0,
  plasterboard_WD = 0,
  tyres_WD = 0,
  wood_WD = 0,
 construction_waste_disposal = c("Closed-loop", "Combustion", "Composting", "Landfill",
    "Open-loop"),
  construction_units = c("kg", "tonnes"),
  metal_units = c("kg", "tonnes"),
  glass_units = c("kg", "tonnes"),
  paper_units = c("kg", "tonnes"),
  plastic_units = c("kg", "tonnes"),
  electrical_units = c("kg", "tonnes")
)
```

Arguments

glass	Numeric value representing the amount of glass. Default is 0.
board	Numeric value indicating the weight of paperboard. Default is 0.
mixed	Numeric value indicating the weight of mixed paper. Default is 0.
paper	Numeric value indicating the weight of paper. Default is 0.
fridges	Numeric value indicating the weight of fridges. Default is 0.
freezers	Numeric value indicating the weight of freezers. Default is 0.
large_electrical	Numeric value indicating the weight of large electrical items. Default is 0.
IT	Numeric value indicating the weight of IT (Information Technology) equipment. Default is 0.
small_electrical	Numeric value indicating the weight of small electrical items. Default is 0.
alkaline_batteries	Numeric value indicating the weight of alkaline batteries. Default is 0.
LiIon_batteries	Numeric value indicating the weight of Lithium-ion batteries. Default is 0.
NiMh_batteries	Numeric value indicating the weight of Nickel Metal Hydride batteries. Default is 0.
aluminuim_cans	Numeric value indicating the weight of aluminum cans. Default is 0.
aluminuim_foil	Numeric value indicating the weight of aluminum foil. Default is 0.

`mixed_cans` Numeric value indicating the weight of mixed metal cans. Default is `0`.

`scrap` Numeric value indicating the weight of metal scrap. Default is `0`.

`steel_cans` Numeric value indicating the weight of steel cans. Default is `0`.

`average` Numeric value indicating the weight of average plastic. Default is `0`.

`average_film` Numeric value indicating the weight of average film plastic. Default is `0`.

`average_rigid` Numeric value indicating the weight of average rigid plastic. Default is `0`.

`HDPE` Numeric value indicating the weight of HDPE plastic. Default is `0`.

`LDPE` Numeric value indicating the weight of LDPE plastic. Default is `0`.

`LLDPE` Numeric value indicating the weight of LLDPE plastic. Default is `0`.

`PET` Numeric value indicating the weight of PET plastic. Default is `0`.

`PP` Numeric value indicating the weight of PP plastic. Default is `0`.

`PS` Numeric value indicating the weight of PS plastic. Default is `0`.

`PVC` Numeric value indicating the weight of PVC plastic. Default is `0`.

`glass_WD` Numeric value representing the amount of glass waste with disposal. Default is `0`.

`glass_waste_disposal`
Character vector specifying the waste disposal method to use for metal for calculating emissions. Possible values: `"Closed-loop"`, `"Combustion"`, `"Landfill"`, `"Open-loop"`. Default is "Closed-loop". See `details` for more information.

`industrial_waste`
Numeric value representing the amount of household residual waste. Default is `0`.

`industrial_waste_disposal`
Character vector specifying the waste disposal method to use for metal for calculating emissions. Possible values: `"Combustion"`, `"Landfill"`. Default is `"Combustion"`. See `details` for more information.

`aluminuim_cans_WD`
Numeric value indicating the weight of aluminum cans disposed of using waste disposal methods. Default is `0`.

`aluminuim_foil_WD`
Numeric value indicating the weight of aluminum foil disposed of using waste disposal methods. Default is `0`.

`mixed_cans_WD` Numeric value indicating the weight of mixed metal cans disposed of using waste disposal methods. Default is `0`.

`scrap_WD` Numeric value indicating the weight of metal scrap disposed of using waste disposal methods. Default is `0`.

`steel_cans_WD` Numeric value indicating the weight of steel cans disposed of using waste disposal methods. Default is `0`.

`metal_waste_disposal`
Character vector specifying the waste disposal method to use for metal for calculating emissions. Possible values: `"Closed-loop"`, `"Combustion"`, `"Landfill"`, `"Open-loop"`. Default is "Closed-loop". See `details` for more information.

board_WD Numeric value indicating the weight of paperboard disposed of using waste disposal methods. Default is 0.

mixed_WD Numeric value indicating the weight of mixed paper disposed of using waste disposal methods. Default is 0.

paper_WD Numeric value indicating the weight of paper disposed of using waste disposal methods. Default is 0.

paper_waste_disposal
Character vector specifying the waste disposal method for paper to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Composting", "Landfill". Default is "Closed-loop". See details for more information.

average_WD Numeric value indicating the weight of average plastic disposed of using waste disposal methods. Default is 0.

average_film_WD
Numeric value indicating the weight of average film plastic disposed of using waste disposal methods. Default is 0.

average_rigid_WD
Numeric value indicating the weight of average rigid plastic disposed of using waste disposal methods. Default is 0.

HDPE_WD Numeric value indicating the weight of HDPE plastic disposed of using waste disposal methods. Default is 0.

LDPE_WD Numeric value indicating the weight of LDPE plastic disposed of using waste disposal methods. Default is 0.

LLDPE_WD Numeric value indicating the weight of LLDPE plastic disposed of using waste disposal methods. Default is 0.

PET_WD Numeric value indicating the weight of PET plastic disposed of using waste disposal methods. Default is 0.

PP_WD Numeric value indicating the weight of PP plastic disposed of using waste disposal methods. Default is 0.

PS_WD Numeric value indicating the weight of PS plastic disposed of using waste disposal methods. Default is 0.

PVC_WD Numeric value indicating the weight of PVC plastic disposed of using waste disposal methods. Default is 0.

plastic_waste_disposal
Character vector specifying the waste disposal method for plastic to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Landfill", "Open-loop". Default is "Closed-loop". See details for more information.

fridges_WD Numeric value indicating the weight of fridges disposed of using waste disposal methods. Default is 0.

freezers_WD Numeric value indicating the weight of freezers disposed of using waste disposal methods. Default is 0.

large_electrical_WD
Numeric value indicating the weight of large electrical items disposed of using waste disposal methods. Default is 0.

IT_WD Numeric value indicating the weight of IT equipment disposed of using waste disposal methods. Default is 0.

small_electrical_WD
Numeric value indicating the weight of small electrical items disposed of using waste disposal methods. Default is 0.

alkaline_batteries_WD
Numeric value indicating the weight of alkaline batteries disposed of using waste disposal methods. Default is 0.

LiIon_batteries_WD
Numeric value indicating the weight of Lithium-ion batteries disposed of using waste disposal methods. Default is 0.

NiMh_batteries_WD
Numeric value indicating the weight of Nickel Metal Hydride batteries disposed of using waste disposal methods. Default is 0.

electric_waste_disposal
Character vector specifying the waste disposal method for electrical items to use for calculating emissions. Possible values: "Landfill", "Open-loop". Default is "Landfill". See details for more information.

aggregates The weight of aggregates used in construction. Default is 0.

construction_average
The weight of average material used in construction. Default is 0.

asbestos The weight of asbestos used in construction. Default is 0.

asphalt The weight of asphalt used in construction. Default is 0.

bricks The weight of bricks used in construction. Default is 0.

concrete The weight of concrete used in construction. Default is 0.

insulation The weight of insulation material used in construction. Default is 0.

metals The weight of metals used in construction. Default is 0.

soils The weight of soils used in construction. Default is 0.

mineral_oil The weight of mineral oil used in construction. Default is 0.

plasterboard The weight of plasterboard used in construction. Default is 0.

tyres The weight of tyres used in construction. Default is 0.

wood The weight of wood used in construction. Default is 0.

aggregates_WD The weight of aggregates disposed of as waste. Default is 0.

construction_average_WD
The weight of average material disposed of as waste. Default is 0.

asbestos_WD The weight of asbestos disposed of as waste. Default is 0.

asphalt_WD The weight of asphalt disposed of as waste. Default is 0.

bricks_WD The weight of bricks disposed of as waste. Default is 0.

concrete_WD The weight of concrete disposed of as waste. Default is 0.

insulation_WD The weight of insulation material disposed of as waste. Default is 0.

metals_WD The weight of metals disposed of as waste. Default is 0.

soils_WD The weight of soils disposed of as waste. Default is 0.

mineral_oil_WD The weight of mineral oil disposed of as waste. Default is 0.

plasterboard_WD
The weight of plasterboard disposed of as waste. Default is 0.

tyres_WD The weight of tyres disposed of as waste. Default is 0.

wood_WD The weight of wood disposed of as waste. Default is 0.

construction_waste_disposal
Character vector specifying the waste disposal method for electrical items to use for calculating emissions. Options are, "Closed-loop", "Combustion", "Composting", "Landfill", "Open-loop". Default is "Closed-loop". Note that: "Closed-loop" is valid for aggregates, average, asphalt, concrete, insulation, metal, soils, mineral oil, plasterboard, tyres, and wood. "Combustion" is valid for average, mineral oil, and wood. "Composting" is valid for wood only. "Landfill" is valid for everything except average, mineral oil, and tyres. "Open-loop" is valid for aggregates, average, asphalt, bricks, concrete, If one of these is used for a value that does not provide it, then an "NA" is given.

construction_units
Character vector specifying the units of the emissions related to construction. Possible values: "kg", "tonnes". Default is "kg".

metal_units Character vector specifying the units of the emissions related to metal. Possible values: "kg", "tonnes". Default is "kg".

glass_units Character vector specifying the units of the emissions related to glass. Possible values: "kg", "tonnes". Default is "kg".

paper_units Character vector specifying the units of the emissions related to paper. Possible values: "kg", "tonnes". Default is "kg".

plastic_units Character vector specifying the units of the emissions related to plastic materials. Possible values: "kg", "tonnes". Default is "kg".

electrical_units
Character vector specifying the units of the emissions related to electrical materials. Possible values: "kg", "tonnes". Default is "kg".

Details

*_waste_disposal methods: "Open-loop" is the process of recycling material into other products. "Closed-loop" is the process of recycling material back into the same product. "Combustion" energy is recovered from the waste through incineration and subsequent generation of electricity. "Compost" CO2e emitted as a result of composting a waste stream. "Landfill" the product goes to landfill after use. "Anaerobic digestion" energy is recovered from the waste through anaerobic digestion.

Value

The calculated household emissions as a numeric value in tonnes.

Examples

```
material_emissions(glass = 100, glass_WD = 10, glass_units = "kg")
```

metal_emissions *Metal Emissions*

Description

This function calculates the emissions produced from different metal sources based on the specified inputs. It considers emissions from primary material production and waste disposal of various metal types.

Usage

```
metal_emissions(
  aluminuim_cans = 0,
  aluminuim_foil = 0,
  mixed_cans = 0,
  scrap = 0,
  steel_cans = 0,
  aluminuim_cans_WD = 0,
  aluminuim_foil_WD = 0,
  mixed_cans_WD = 0,
  scrap_WD = 0,
  steel_cans_WD = 0,
  waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
  units = c("kg", "tonnes")
)
```

Arguments

aluminuim_cans	Numeric value indicating the weight of aluminum cans. Default is 0.
aluminuim_foil	Numeric value indicating the weight of aluminum foil. Default is 0.
mixed_cans	Numeric value indicating the weight of mixed metal cans. Default is 0.
scrap	Numeric value indicating the weight of metal scrap. Default is 0.
steel_cans	Numeric value indicating the weight of steel cans. Default is 0.
aluminuim_cans_WD	Numeric value indicating the weight of aluminum cans disposed of using waste disposal methods. Default is 0.
aluminuim_foil_WD	Numeric value indicating the weight of aluminum foil disposed of using waste disposal methods. Default is 0.
mixed_cans_WD	Numeric value indicating the weight of mixed metal cans disposed of using waste disposal methods. Default is 0.
scrap_WD	Numeric value indicating the weight of metal scrap disposed of using waste disposal methods. Default is 0.
steel_cans_WD	Numeric value indicating the weight of steel cans disposed of using waste disposal methods. Default is 0.

waste_disposal Character vector specifying the waste disposal method to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Landfill", "Open-loop". Default is "Closed-loop". "Open-loop" is the process of recycling material into other products. "Closed-loop" is the process of recycling material back into the same product. "Combustion" energy is recovered from the waste through incineration and subsequent generation of electricity. "Landfill" the product goes to landfill after use.

units Character vector specifying the units of the emissions output. Possible values: "kg", "tonnes". Default is "kg".

Value

The function returns the calculated CO2-equivalent emissions as a numeric value in tonnes.

Examples

```
metal_emissions(aluminuim_cans = 1.4, units = "tonnes")
```

---

office_emissions *Office emissions*

Description

Office emissions

Usage

```
office_emissions(
  specify = FALSE,
  office_num = 1,
  WFH_num = 0,
  WFH_hours = 0,
  WFH_type = c("Office Equipment", "Heating"),
  water_supply = 0,
  water_trt = TRUE,
  water_unit = c("cubic metres", "million litres"),
  electricity_kWh = 0,
  electricity_TD = TRUE,
  electricity_WTT = TRUE,
  heat_kWh = 0,
  heat_TD = TRUE,
  heat_WTT = TRUE
)
```

Arguments

specify	logical. Whether an average should be used, or if the amount of energy used will be specified.
office_num	If specify = FALSE, the number of individuals in the office.
WFH_num	Number of people working from home.
WFH_hours	Number of hours working from home (per individual).
WFH_type	Whether to account for "Office Equipment" and/or "Heating". Default is both.
water_supply	Amount of water used in the office.
water_trt	logical. Default TRUE. Whether to include emissions associated with water treatment for used water.
water_unit	Unit for water_supply variable. Options are "cubic metres" or "million litres".
electricity_kWh	Electricity used in kWh.
electricity_TD	logical. Default TRUE. Whether to include emissions associated with grid losses.
electricity_WTT	logical. Default TRUE. Whether to include emissions associated with extracting, refining, and transporting fuels.
heat_kWh	heat and steam used in kWh.
heat_TD	logical. Default TRUE. Whether to include emissions associated with grid losses.
heat_WTT	logical. Default TRUE. Whether to include emissions associated with extracting, refining, and transporting fuels.

Value

Returns CO2e emissions in tonnes.

References

Descriptions from 2022 UK Government Report: https://www.gov.uk/government/publications/greenhouse-gas-reporting-conversion-factors-2022

Examples

```
# specify emissions in an office
office_emissions(specify = TRUE, electricity_kWh = 200,
                 heat_kWh = 100, water_supply = 100, water_trt = FALSE)
```

output_display *Display a grid of plots and tables*

Description

This function generates a grid of plots and tables, including a value box, data table, relative GPI plot, and total output plot.

Usage

```
output_display(
  data = x$data,
  time = time,
  date_format = c("%d/%m/%Y"),
  name = theatre,
  relative_gpi_val = emissions,
  gti_by = c("default", "month", "year"),
  plot_val = carbon_price_credit,
  plot_by = "default",
  pdf = TRUE
)
```

Arguments

data	The data frame containing the data.
time	The variable representing the time dimension.
date_format	The date format for the time variable (optional, default: "%d/%m/%Y").
name	The variable representing the grouping variable.
relative_gpi_val	The variable representing the relative GPI (Growth to Previous Index) value.
gti_by	The grouping type for calculating the GTI ("default", "month", "year").
plot_val	The variable to plot in the total output plot.
plot_by	The grouping type for the total output plot ("default", "month", "year").
pdf	Whether to export the plots to a PDF file (default: TRUE).

Details

The function utilises other auxiliary functions such as relative_gti() and total_output().

Value

A grid of plots and tables showing the value box, data table, relative GPI plot, and total output plot.

paper_emissions *Paper Emissions*

Description

This function calculates the emissions produced from different paper sources based on the specified inputs. It considers emissions from primary material production and waste disposal of paper materials.

Usage

```
paper_emissions(
  board = 0,
  mixed = 0,
  paper = 0,
  board_WD = 0,
  mixed_WD = 0,
  paper_WD = 0,
  waste_disposal = c("Closed-loop", "Combustion", "Composting", "Landfill"),
  units = c("kg", "tonnes")
)
```

Arguments

board	Numeric value indicating the weight of paperboard. Default is 0.
mixed	Numeric value indicating the weight of mixed paper. Default is 0.
paper	Numeric value indicating the weight of paper. Default is 0.
board_WD	Numeric value indicating the weight of paperboard disposed of using waste disposal methods. Default is 0.
mixed_WD	Numeric value indicating the weight of mixed paper disposed of using waste disposal methods. Default is 0.
paper_WD	Numeric value indicating the weight of paper disposed of using waste disposal methods. Default is 0.
waste_disposal	Character vector specifying the waste disposal method to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Composting", "Landfill". Default is "Closed-loop". "Closed-loop" is the process of recycling material back into the same product. "Combustion" energy is recovered from the waste through incineration and subsequent generation of electricity. "Compost" CO2e emitted as a result of composting a waste stream. "Landfill" the product goes to landfill after use.
units	Character vector specifying the units of the emissions output. Possible values: "kg", "tonnes". Default is "kg".

Value

The function returns the calculated paper emissions as a numeric value in tonnes.

Examples

```
paper_emissions(board = 10, board_WD = 10, paper = 100, paper_WD = 100, units = "kg")
```

`plastic_emissions` *Calculate Plastic Emissions*

Description

This function calculates the emissions produced from different plastic sources based on the specified inputs. It considers emissions from primary material production and waste disposal of plastic materials.

Usage

```
plastic_emissions(
  average = 0,
  average_film = 0,
  average_rigid = 0,
  HDPE = 0,
  LDPE = 0,
  LLDPE = 0,
  PET = 0,
  PP = 0,
  PS = 0,
  PVC = 0,
  average_WD = 0,
  average_film_WD = 0,
  average_rigid_WD = 0,
  HDPE_WD = 0,
  LDPE_WD = 0,
  LLDPE_WD = 0,
  PET_WD = 0,
  PP_WD = 0,
  PS_WD = 0,
  PVC_WD = 0,
  waste_disposal = c("Closed-loop", "Combustion", "Landfill", "Open-loop"),
  units = c("kg", "tonnes")
)
```

Arguments

`average`	Numeric value indicating the weight of average plastic. Default is `0`.
`average_film`	Numeric value indicating the weight of average film plastic. Default is `0`.
`average_rigid`	Numeric value indicating the weight of average rigid plastic. Default is `0`.
`HDPE`	Numeric value indicating the weight of HDPE plastic. Default is `0`.
`LDPE`	Numeric value indicating the weight of LDPE plastic. Default is `0`.

LLDPE	Numeric value indicating the weight of LLDPE plastic. Default is 0.
PET	Numeric value indicating the weight of PET plastic. Default is 0.
PP	Numeric value indicating the weight of PP plastic. Default is 0.
PS	Numeric value indicating the weight of PS plastic. Default is 0.
PVC	Numeric value indicating the weight of PVC plastic. Default is 0.
average_WD	Numeric value indicating the weight of average plastic disposed of using waste disposal methods. Default is 0.
average_film_WD	Numeric value indicating the weight of average film plastic disposed of using waste disposal methods. Default is 0.
average_rigid_WD	Numeric value indicating the weight of average rigid plastic disposed of using waste disposal methods. Default is 0.
HDPE_WD	Numeric value indicating the weight of HDPE plastic disposed of using waste disposal methods. Default is 0.
LDPE_WD	Numeric value indicating the weight of LDPE plastic disposed of using waste disposal methods. Default is 0.
LLDPE_WD	Numeric value indicating the weight of LLDPE plastic disposed of using waste disposal methods. Default is 0.
PET_WD	Numeric value indicating the weight of PET plastic disposed of using waste disposal methods. Default is 0.
PP_WD	Numeric value indicating the weight of PP plastic disposed of using waste disposal methods. Default is 0.
PS_WD	Numeric value indicating the weight of PS plastic disposed of using waste disposal methods. Default is 0.
PVC_WD	Numeric value indicating the weight of PVC plastic disposed of using waste disposal methods. Default is 0.
waste_disposal	Character vector specifying the waste disposal method to use for calculating emissions. Possible values: "Closed-loop", "Combustion", "Landfill", "Open-loop". Default is "Closed-loop". "Open-loop" is the process of recycling material into other products. "Closed-loop" is the process of recycling material back into the same product. "Combustion" energy is recovered from the waste through incineration and subsequent generation of electricity. "Landfill" the product goes to landfill after use.
units	Character vector specifying the units of the emissions output. Possible values: "kg", "tonnes". Default is "kg".

Value

The calculated plastic emissions as a numeric value in tonnes.

Examples

```
# Calculate plastic emissions using default values
plastic_emissions()

# Calculate plastic emissions with specific quantities and waste disposal
# method
plastic_emissions(average = 100, HDPE = 50, PET = 25,
                  waste_disposal = "Combustion", units = "tonnes")
```

`rail_emissions` *Calculate CO2e emissions from a train journey*

Description

A function that calculates CO2e emissions between train stations in the UK.

Usage

```
rail_emissions(
  from,
  to,
  via = NULL,
  num_people = 1,
  times_journey = 1,
  include_WTT = TRUE,
  round_trip = FALSE
)
```

Arguments

`from`	Station departing from.
`to`	Station arriving to
`via`	Optional. Takes a vector containing the stations the train travels via.
`num_people`	Number of people taking the journey. Takes a single numerical value.
`times_journey`	Number of times the journey is taken.
`include_WTT`	logical. Recommended `TRUE`. Whether to include emissions associated with extracting, refining, and transporting fuels.
`round_trip`	Whether the journey is one-way or return.

Details

The distances are calculated using the Haversine formula. This is calculated as the crow flies. As a result, inputting the "via" journeys will make for a more reliable function.

Value

Returns CO2e emissions in tonnes for the train journey.

Examples

```
# Emissions for a train journey between Southampton Central and
# Manchester Piccadilly Station
rail_emissions("Southampton Central", "Manchester Piccadilly")

# Emissions for a train journey between Bristol Temple Meads and
# London Paddington via Bath, Swindon, and Reading
# Use the \code{rail_finder} function to find the name of London Paddington
rail_finder(region = "London")
# Then calculate emissions
rail_emissions("Bristol Temple Meads", "Paddington", via = c("Bath Spa",
"Swindon", "Reading"))
```

---

rail_finder *Find the station code for a train station*

Description

Find the name, area, and code of a train station in the UK. For use in the `rail_emissions` function.

Usage

```
rail_finder(
  station,
  region,
  county,
  district,
  station_code,
  distance = 0.1,
  ignore.case = FALSE
)
```

Arguments

`station`	Name of train station.
`region`	Region the train station is in. One of `c("London", "Scotland", "Wales - Cymru", "North West", "West Midlands", "North East", "East", "South East", "East Midlands", "Yorkshire And The Humber", "South West", NA)`.
`county`	County the train station is in.
`district`	District the train station is in.
`station_code`	Code of the train station.
`distance`	Maximum distance allowed for a match between the name/country/city given, and that of the value in the data set.
`ignore.case`	If `FALSE`, the check for is case-sensitive. If `TRUE`, case is ignored.

Value

Data frame containing the station code, station name, region, county, district, latitude, and longitude of a train station in the UK.

Examples

```
# Can get the station code from the station. Gets similar matches.
rail_finder(station = "Bristo")

# Can get the code from the station and city.
rail_finder(station = "Bristo", county = "Bristol")

# Can find the name and district of a train station given the IATA code
rail_finder(station_code = "BRI")
```

---

raw_fuels *Raw Fuels Emissions*

Description

Raw Fuels Emissions

Usage

```
raw_fuels(
  num_people = 1,
  butane = 0,
  CNG = 0,
  LPG = 0,
  LNG = 0,
  natural_gas = 0,
  natural_gas_mineral = 0,
  other_petroleum_gas = 0,
  propane = 0,
  aviation = 0,
  aviation_fuel = 0,
  burning_oil = 0,
  diesel = 0,
  diesel_mineral = 0,
  fuel_oil = 0,
  gas_oil = 0,
  lubricants = 0,
  naptha = 0,
  petrol_biofuel = 0,
  petrol_mineral = 0,
  residual_oil = 0,
  distillate = 0,
```

```
refinery_miscellaneous = 0,
waste_oils = 0,
marine_gas = 0,
marine_fuel = 0,
coal_industrial = 0,
coal_electricity_gen = 0,
coal_domestic = 0,
coking_coal = 0,
petroleum_coke = 0,
coal_home_produced_gen = 0,
bioethanol = 0,
biodiesel = 0,
biomethane = 0,
biodiesel_cooking_oil = 0,
biodiesel_tallow = 0,
biodiesel_HVO = 0,
biopropane = 0,
bio_petrol = 0,
renewable_petrol = 0,
wood_log = 0,
wood_chips = 0,
wood_pellets = 0,
grass = 0,
biogas = 0,
landfill_gas = 0,
butane_units = c("kwh", "litres", "tonnes"),
CNG_units = c("kwh", "litres", "tonnes"),
LPG_units = c("kwh", "litres", "tonnes"),
LNG_units = c("kwh", "litres", "tonnes"),
natural_gas_units = c("kwh", "cubic metres", "tonnes"),
natural_gas_mineral_units = c("kwh", "cubic metres", "tonnes"),
other_petroleum_gas_units = c("kwh", "litres", "tonnes"),
propane_units = c("kwh", "litres", "tonnes"),
aviation_units = c("kwh", "litres", "tonnes"),
aviation_fuel_units = c("kwh", "litres", "tonnes"),
burning_oil_units = c("kwh", "litres", "tonnes"),
diesel_units = c("kwh", "litres", "tonnes"),
diesel_mineral_units = c("kwh", "litres", "tonnes"),
fuel_oil_units = c("kwh", "litres", "tonnes"),
gas_oil_units = c("kwh", "litres", "tonnes"),
lubricants_units = c("kwh", "litres", "tonnes"),
naptha_units = c("kwh", "litres", "tonnes"),
petrol_biofuel_units = c("kwh", "litres", "tonnes"),
petrol_mineral_units = c("kwh", "litres", "tonnes"),
residual_oil_units = c("kwh", "litres", "tonnes"),
distillate_units = c("kwh", "litres", "tonnes"),
refinery_miscellaneous_units = c("kwh", "litres", "tonnes"),
waste_oils_units = c("kwh", "tonnes"),
```

```
  marine_gas_units = c("kwh", "tonnes"),
  marine_fuel_units = c("kwh", "tonnes"),
  coal_industrial_units = c("kwh", "tonnes"),
  coal_electricity_gen_units = c("kwh", "tonnes"),
  coal_domestic_units = c("kwh", "tonnes"),
  coking_coal_units = c("kwh", "tonnes"),
  petroleum_coke_units = c("kwh", "tonnes"),
  coal_home_produced_gen_units = c("kwh", "tonnes"),
  bioethanol_units = c("litres", "GJ", "kg"),
  biodiesel_units = c("litres", "GJ", "kg"),
  biomethane_units = c("litres", "GJ", "kg"),
  biodiesel_cooking_oil_units = c("litres", "GJ", "kg"),
  biodiesel_tallow_units = c("litres", "GJ", "kg"),
  biodiesel_HVO_units = c("litres", "GJ", "kg"),
  biopropane_units = c("litres", "GJ", "kg"),
  bio_petrol_units = c("litres", "GJ", "kg"),
  renewable_petrol_units = c("litres", "GJ", "kg"),
  wood_log_units = c("kwh", "tonnes"),
  wood_chips_units = c("kwh", "tonnes"),
  wood_pellets_units = c("kwh", "tonnes"),
  grass_units = c("kwh", "tonnes"),
  biogas_units = c("kwh", "tonnes"),
  landfill_gas_units = c("kwh", "tonnes")
)
```

Arguments

num_people	Number of people to account for.
butane	amount of Butane used.
CNG	amount used. Compressed natural gas (CNG). A compressed version of the natural gas used in homes. An alternative transport fuel.
LPG	amount used. Liquid petroleum gas. Used to power cooking stoves or heaters off-grid and fuel some vehicles (e.g. fork-lift trucks and vans).
LNG	amount used. Liquefied natural gas. An alternative transport fuel.
natural_gas	amount used. Standard natural gas received through the gas mains grid network in the UK.
natural_gas_mineral	amount used. Natural gas (100% mineral blend) factor is natural gas not obtained through the grid and therefore does not contain any biogas content. It can be used for calculating bespoke fuel mixtures.
other_petroleum_gas	amount used. Consists mainly of ethane, plus other hydrocarbons, (excludes butane and propane).
propane	amount used.
aviation	amount used. Fuel for piston-engined aircraft - a high octane petrol (aka AVGAS).

`aviation_fuel` amount used. Fuel for turbo-prop aircraft and jets (aka jet fuel). Similar to kerosene used as a heating fuel, but refined to a higher quality.

`burning_oil` amount used. Main purpose is for heating/lighting on a domestic scale (also known as kerosene).

`diesel` amount used. Standard diesel bought from any local filling station (across the board forecourt fuel typically contains biofuel content).

`diesel_mineral` amount used. Diesel that has not been blended with biofuel (non-forecourt diesel).

`fuel_oil` amount used. Heavy oil used as fuel in furnaces and boilers of power stations, in industry, for industrial heating and in ships.

`gas_oil` amount used. Medium oil used in diesel engines and heating systems (also known as red diesel).

`lubricants` amount used. Waste petroleum-based lubricating oils recovered for use as fuels

`naptha` amount used. A product of crude oil refining - often used as a solvent.

`petrol_biofuel` amount used. Standard petrol bought from any local filling station (across the board forecourt fuel typically contains biofuel content).

`petrol_mineral` amount used. Petrol that has not been blended with biofuel (non forecourt petrol).

`residual_oil` amount used. Waste oils meeting the 'residual' oil definition contained in the 'Processed Fuel Oil Quality Protocol'.

`distillate` amount used. Waste oils meeting the 'distillate' oil definition contained in the 'Processed Fuel Oil Quality Protocol'.

`refinery_miscellaneous` amount used. Includes aromatic extracts, defoament solvents and other minor miscellaneous products

`waste_oils` amount used. Recycled oils outside of the 'Processed Fuel Oil Quality Protocol' definitions.

`marine_gas` amount used. Distillate fuels are commonly called "Marine gas oil". Distillate fuel is composed of petroleum fractions of crude oil that are separated in a refinery by a boiling or "distillation" process.

`marine_fuel` amount used. Residual fuels are called "Marine fuel oil". Residual fuel or "residuum" is the fraction that did not boil, sometimes referred to as "tar" or "petroleum pitch".

`coal_industrial` amount used. Coal used in sources other than power stations and domestic use.

`coal_electricity_gen` amount used. Coal used in power stations to generate electricity.

`coal_domestic` amount used. Coal used domestically.

`coking_coal` amount used. Coke may be used as a heating fuel and as a reducing agent in a blast furnace.

`petroleum_coke` amount used. Normally used in cement manufacture and power plants.

coal_home_produced_gen
: amount used. Coal used in power stations to generate electricity (only for coal produced in the UK).

bioethanol
: amount used. Renewable fuel derived from common crops (such as sugar cane and sugar beet).

biodiesel
: amount used. Renewable fuel almost exclusively derived from common natural oils (for example, vegetable oils).

biomethane
: amount used. The methane constituent of biogas. Biogas comes from anaerobic digestion of organic matter.

biodiesel_cooking_oil
: amount used. Renewable fuel almost exclusively derived from common natural oils (such as vegetable oils).

biodiesel_tallow
: amount used. Renewable fuel almost exclusively derived from common natural oils (such as vegetable oils).

biodiesel_HVO
: amount used.

biopropane
: amount used.

bio_petrol
: amount used.

renewable_petrol
: amount used.

wood_log
: amount used.

wood_chips
: amount used.

wood_pellets
: amount used. Compressed low quality wood (such as sawdust and shavings) made into pellet form

grass
: amount used.

biogas
: amount used. A naturally occurring gas from the anaerobic digestion of organic materials (such as sewage and food waste), or produced intentionally as a fuel from the anaerobic digestion of biogenic substances (such as energy crops and agricultural residues).

landfill_gas
: amount used. Gas collected from a landfill site. This may be used for electricity generation, collected and purified for use as a transport fuel, or be flared off

butane_units
: units that the gas is given in. Options are "tonnes", "litres", "kwh".

CNG_units
: units that the gas is given in. Options are "tonnes", "litres", "kwh".

LPG_units
: units that the gas is given in. Options are "tonnes", "litres", "kwh".

LNG_units
: units that the gas is given in. Options are "tonnes", "litres", "kwh".

natural_gas_units
: units that the gas is given in. Options are "tonnes", "cubic metres", "kwh".

natural_gas_mineral_units
: units that the gas is given in. Options are "tonnes", "cubic metres", "kwh".

other_petroleum_gas_units
: units that the gas is given in. Options are "tonnes", "litres", "kwh".

propane_units
: units that the gas is given in. Options are "tonnes", "litres", "kwh".

aviation_units units that the fuel is given in. Options are "tonnes", "litres", "kwh".

aviation_fuel_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

burning_oil_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

diesel_units units that the fuel is given in. Options are "tonnes", "litres", "kwh".

diesel_mineral_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

fuel_oil_units units that the fuel is given in. Options are "tonnes", "litres", "kwh".

gas_oil_units units that the fuel is given in. Options are "tonnes", "litres", "kwh".

lubricants_units
: units that the fuel is given in. Options are "tonnes", "kwh".

naptha_units units that the fuel is given in. Options are "tonnes", "litres", "kwh".

petrol_biofuel_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

petrol_mineral_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

residual_oil_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

distillate_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

refinery_miscellaneous_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

waste_oils_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

marine_gas_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

marine_fuel_units
: units that the fuel is given in. Options are "tonnes", "litres", "kwh".

coal_industrial_units
: units that the fuel is given in. Options are "kwh", "tonnes".

coal_electricity_gen_units
: units that the fuel is given in. Options are "kwh", "tonnes".

coal_domestic_units
: units that the fuel is given in. Options are "kwh", "tonnes".

coking_coal_units
: units that the fuel is given in. Options are "kwh", "tonnes".

petroleum_coke_units
: units that the fuel is given in. Options are "kwh", "tonnes".

coal_home_produced_gen_units
: units that the fuel is given in. Options are "kwh", "tonnes".

bioethanol_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

biodiesel_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

biomethane_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

biodiesel_cooking_oil_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

biodiesel_tallow_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

biodiesel_HVO_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

biopropane_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

bio_petrol_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

renewable_petrol_units
: units that the biofuel is given in. Options are "litres", "GJ", "kg".

wood_log_units
: units that the biomass is given in. Options are "tonnes", "kwh".

wood_chips_units
: units that the biomass is given in. Options are "tonnes", "kwh".

wood_pellets_units
: units that the biomass is given in. Options are "tonnes", "kwh".

grass_units
: units that the biomass is given in. Options are "tonnes", "kwh".

biogas_units
: units that the biogas is given in. Options are "tonnes", "kwh".

landfill_gas_units
: units that the biogas is given in. Options are "tonnes", "kwh".

Details

This function calculates CO2e emissions from a wide variety of fuels, considering different unit conversions for each type of fuel. It supports the calculation of emissions from commonly used fuels such as diesel, petrol, natural gas, and biodiesel, as well as more specific fuels like aviation fuel, marine fuel, and landfill gas.

Unit conversions are done internally based on the specified units for each type of fuel (e.g., kWh, litres, tonnes). The function is useful for assessing the carbon footprint associated with different fuel sources over a specified time period.

Value

A data frame with calculated emissions in tonnes of CO2e for each type of fuel input.

References

- DEFRA Conversion Factors for Greenhouse Gas (GHG) Reporting: https://www.gov.uk/government/collections/govern conversion-factors-for-company-reporting
- Descriptions from 2021 UK Government Report: https://www.gov.uk/government/publications/greenhouse-gas-reporting-conversion-factors-2021

Examples

```
# Calculate emissions for 100 litres of diesel and 500 kWh of natural gas:
raw_fuels(
  diesel = 100, diesel_units = "litres",
  natural_gas = 500, natural_gas_units = "kwh",
)

# Calculate emissions for 10 tonnes of aviation fuel:
raw_fuels(
  aviation_fuel = 10, aviation_fuel_units = "tonnes",
)
```

---

relative_gti *Relative GTI Plot*

Description

Calculate and plot the relative growth to index (GTI) over time

Usage

```
relative_gti(
  data = data,
  time = time,
  date_format = c("%d/%m/%Y"),
  name = theatre,
  val = emissions,
  gti_by = c("default", "month", "year")
)
```

Arguments

data	The data frame containing the data.
time	The variable representing the time dimension.
date_format	The date format for the time variable (optional, default: c("%d/%m/%Y")).
name	The variable representing the grouping variable.
val	The variable representing the value.
gti_by	The grouping type for calculating the GTI ("default", "month", "year").

Value

A ggplot2 object showing the relative GTI (Growth to Index) over time.

Examples

```
# Example dataset
emission_data <- data.frame(
  theatre = c("Theatre A", "Theatre A", "Theatre B", "Theatre B", "Theatre A", "Theatre B"),
   emissions = c(200, 250, 150, 180, 300, 220),
  date = c("01/01/2023", "01/02/2023", "01/01/2023", "01/02/2023", "01/03/2023", "01/03/2023")
   )

# Using the relative_gti function
relative_gti_plot <- relative_gti(
  data = emission_data,
  time = date,
  date_format = "%d/%m/%Y",  # Date format used in the dataset
  name = theatre,
  val = emissions,
  gti_by = "default"  # Calculating based on default time period
)

# Plot the relative GTI
print(relative_gti_plot)
```

---

seaports *Dataset of different seaports*

Description

A dataset containing the country, country code, city, city code, and coordinates of seaports

Usage

```
data(seaports)
```

Format

A data frame with 8736 rows and 7 variables:

country Name of the country with the port

city Name of the city with the port

country_code Code of the country name

port_code Code of the city port

latitude Latitude of the port

longitude Longitude of the port

Source

https://www.kaggle.com/therohk/world-seaport-airport-dataset-and-codes/script

seaport_finder *Check the code or name of a seaport*

Description

Find the name and/or code of a seaport. For use in the `ferry_emissions` function.

Usage

```
seaport_finder(city, country, port_code, distance = 0.1, ignore.case = FALSE)
```

Arguments

`city`	Name of the city.
`country`	Name of the country.
`port_code`	Name of the port.
`distance`	Maximum distance allowed for a match between the country/city given, and that of the value in the data set.
`ignore.case`	If `FALSE`, the check is case-sensitive. If `TRUE`, case is ignored.

Value

Data frame containing the country, city, country code, port code, latitude, and longitude of a seaport.

Examples

```
# Look up the city of Aberdeen to find the port_code for it
seaport_finder(city = "Aberdeen")

# Search for a country and city and it finds matches
seaport_finder(country = "United", city = "borunemouth", ignore.case = TRUE)
```

shiny_emissions *Run 'shiny' App to Calculate Carbon Emissions*

Description

Runs a GUI to the functions in the 'carbonr' package to calculate carbon-equivalent emissions.

Usage

```
shiny_emissions()
```

Value

'shiny' app to calculate carbon-equivalent emissions

Examples

```
if(interactive()){shiny_emissions()}
```

`stations` *Dataset of UK train stations*

Description

A dataset containing the city, station code, and coordinates of seaports

Usage

```
data(stations)
```

Format

A data frame with 2608 rows and 4 variables:

station Name of the station

station_code Code of the station

region Region of the station. One of `"London"`, `"Scotland"`, `"Wales - Cymru"`, `"North West"`, `"West Midlands"`, `"North East"`, `"East"`, `"South East"`, `"East Midlands"`, `"Yorkshire And The Humber"`, `"South West"`, `NA`

county County of the station

district District of the station

latitude Latitude of the station

longitude Longitude of the station

Source

https://www.data.gov.uk/dataset/ff93ffc1-6656-47d8-9155-85ea0b8f2251/national-public-transport-acce

https://www.theguardian.com/news/datablog/2011/may/19/train-stations-listed-rail#data

`total_output` *Calculate Total Output and Generate Plot*

Description

This function calculates the total output and generates a plot based on the specified parameters.

Usage

```
total_output(
  data = x$data,
  time = time,
  date_format = c("%d/%m/%Y"),
  name = theatre,
  val = carbon_price_credit,
  plot_by = c("default", "month", "year")
)
```

Arguments

`data`	The data frame containing the data.
`time`	The variable representing the time dimension.
`date_format`	The date format for the time variable (optional, default: c("%d/%m/%Y")).
`name`	The variable representing the grouping variable.
`val`	The variable to calculate the total output for (default: carbon_price_credit).
`plot_by`	The grouping type for the total output plot ("default", "month", "year").

Details

This function calculates the total output by grouping the data based on the specified parameters (grouping variable and time dimension). It then summarises the specified variable (CPI or emissions) using the sum function. The resulting data is used to create a line plot showing the total output over time, with each group represented by a different color. The plot can be grouped by the default grouping, month, or year, based on the plot_by parameter.

Value

A ggplot object showing the total output plot.

vehicle_emissions *Calculate CO2e emissions from land-travel journeys*

Description

A function that calculates CO2e emissions on a journey on land.

Usage

```
vehicle_emissions(
  distance,
  units = c("miles", "km"),
  num = 1,
  vehicle = c("Cars", "Motorbike"),
  fuel = c("Petrol", "Diesel", "Unknown", "Battery Electric Vehicle",
    "Plug-in Hybrid Electric Vehicle"),
  car_type = c("Average car", "Small car", "Medium car", "Large car", "Mini",
   "Supermini", "Lower medium", "Upper medium", "Executive", "Luxury", "Sports",
    "Dual purpose 4X4", "MPV"),
  bike_type = c("Average", "Small", "Medium", "Large"),
  TD = TRUE,
  include_WTT = TRUE,
  include_electricity = TRUE,
  owned_by_org = TRUE
)
```

Arguments

distance	Distance in km or miles of the journey made (this can be calculated with other tools, such as google maps.).
units	Units for the distance travelled. Options are "km" or "miles".
num	Number of vehicles used.
vehicle	Vehicle used for the journey. Options are "Cars", "Motorbike".
fuel	Fuel type used for the journey. For car, "Petrol", "Diesel", "Unknown", "Battery Electric Vehicle", "Plug-in Hybrid Electric Vehicle" are options. ##' "hybrid electric" and "battery electric" account for electricity kWh emissions.
car_type	Size/type of vehicle for car. Options are c("Average car", "Small car", "Medium car", "Large car" "Mini", "Supermini", "Lower medium", "Upper medium", "Executive", "Luxury", "Sports", "Dual purpose 4X4", "MPV"),. Small denotes up to a 1.4L engine, unless diesel which is up to 1.7L engine. Medium denotes 1.4-2.0L for petrol cars, 1.7-2.0L for diesel cars. Large denotes 2.0L+ engine.
bike_type	Size of vehicle for motorbike. Options are "Small", "Medium", "Large", or "Average". Sizes denote upto 125cc, 125cc-500cc, 500cc+ respectively.

TD	logical.Whether to account for transmission and distribution (TD) for electric vehicles (only car and van)
include_WTT	logical. Well-to-tank (include_WTT) - whether to account for emissions associated with extraction, refining and transportation of the fuels (for non-electric vehicles).
include_electricity	logical. Whether to account for ... for electric vehicles (car and van).
owned_by_org	logical. Whether the vehicle used is owned by the organisation or not (only for car, motorbike).

Value

Tonnes of CO2e emissions per mile travelled.

Examples

```
# Emissions for a 100 mile car journey
 vehicle_emissions(distance = 100)

# Emissions for a 100 mile motorbike journey where the motorbike is 500+cc
 vehicle_emissions(distance = 100, vehicle = "Motorbike", bike_type = "Large")
```

Index