

# Package ‘ukghg’

March 30, 2020

**Title** Greenhouse Gas Fluxes from the UK

**Version** 0.7.1

**Description** Spatio-temporal predictions of UK GHG emissions.

**Depends** R (>= 3.2.0),  
raster

**Imports** mgcv,  
shiny

**Suggests** ncdf4,  
testthat,  
knitr,  
covr

**License** MIT + file LICENSE

**BugReports** <https://github.com/NERC-CEH/ukghg/issues>

**LazyData** true

**VignetteBuilder** knitr

**RoxygenNote** 7.1.0

**ByteCompile** true

## R topics documented:

ukghg-package . . . . .	1
calcFlux . . . . .	2
runShinyApp . . . . .	3

<b>Index</b>	<b>4</b>
--------------	----------

---

ukghg-package	<i>Generate maps of GHG fluxes for the UK.</i>
---------------	--

---

## Description

ukghg allows you to produce maps of GHG fluxes for the UK and write these to netCDF files.

## Details

The only function you're likely to need from **ukghg** is **calcFlux**. Refer to the vignettes for details of how to use it - use `vignette()`.

## Author(s)

**Maintainer:** Peter Levy <plevy@ceh.ac.uk> ([ORCID](#)) [copyright holder]

## See Also

Useful links:

- Report bugs at <https://github.com/NERC-CEH/ukghg/issues>

---

calcFlux

*A high-level function for calculating a sequence of maps of GHG flux*

---

## Description

This function calculates greenhouse gas fluxes from the UK, based on a spatio-temporal model and the national GHG inventory data.

## Usage

```
calcFlux(
  ghgName = c("ch4", "co2", "n2o", "c2h6", "voc"),
  datect = datect,
  proj = c("OSGB", "LonLat"),
  res = c("1", "20", "100"),
  unitType = c("mol", "g"),
  unitSIprefix = c("peta", "tera", "giga", "mega", "kilo", "none", "milli", "micro",
    "nano", "pico"),
  writeNetCDF = FALSE,
  sectorList = 1:10,
  includeBio = TRUE,
  timeScales = c(TRUE, TRUE, TRUE, TRUE),
  beta_df = data.frame(sector = 1:10, beta_year = rep(1, 10), beta_yday = rep(1, 10),
    beta_wday = rep(1, 10), beta_hour = rep(1, 10))
)
```

## Arguments

ghgName	Greenhouse gas: one of "ch4", "co2", "n2o", "c2h6" or "voc". Defaults to "ch4".
datect	A vector of timestamps in POSIXct format.
proj	Geographic projection for the gridded data, either "OSGB" or "LonLat". Defaults to OSGB.
res	Resolution for the gridded data, either 1, 20 or 100 km. Defaults to "1km". Not yet implemented for LonLat.
unitType	Either molar ("mol") or mass-based ("g").
unitSIprefix	Any standard SI prefix for the output units, from "peta" to "pico".

writeNetCDF	Write NetCDF output files. Defaults to FALSE.
sectorList	A vector of sector numbers for which alpha values should be returned, e.g. c(1,3,7). Defaults to all.
includeBio	A logical for whether biogenic fluxes should be calculated as well as anthropogenic sectors 1-10. Defaults to TRUE.
timeScales	A vector of logicals for including variation at inter-annual, seasonal, intra-weekly, and diurnal time scales (i.e. the POSIXlt variables year, yday, wday, and hour. Defaults to TRUE for all four.
beta_df	A data frame of beta parameters, used in calibration of the model. Defaults to a dataframe with beta = 1 for all parameters.

### Value

total A vector of total flux

s\_ghgTotal A RasterStack of total flux

ls\_ghgByTimeBySector A list of RasterStacks of ghg fluxes where the z dimension corresponds to sector, one per timestep

ls\_ghgBySectorByTime A list of RasterStacks of ghg fluxes where the z dimension corresponds to timestep, one per sector

### Examples

```
startDate <- as.POSIXct(strptime("01/06/2006", "%d/%m/%Y"), tz = "UTC")
endDate   <- as.POSIXct(strptime("02/06/2006", "%d/%m/%Y"), tz = "UTC")
nTimes <- 2
# create a sequence of timestamps
datect <- seq(startDate, endDate, length = nTimes)
# calculate fluxes for these times
myFlux <- calcFlux("ch4", datect, proj = "OSGB", res = "100" , "mol", "nano")
```

---

runShinyApp

*Launches the shiny app for the ukghg package*


---

### Description

This function provides a web browser interface to calculate greenhouse gas fluxes from the UK, based on a spatio-temporal model and the national GHG inventory data.

### Usage

```
runShinyApp()
```

### Value

shiny application object

# Index

- \*Topic **app**
  - runShinyApp, [3](#)
- \*Topic **shiny**
  - runShinyApp, [3](#)
- \*Topic **units**
  - calcFlux, [2](#)
- calcFlux, [2](#), [2](#)
- runShinyApp, [3](#)
- ukghg (ukghg-package), [1](#)
- ukghg-package, [1](#)