Package ‘vein’

May 8, 2017

Type Package

Title An R package for vehicular emissions inventories

Version 0.2.1-1

Author Sergio Ibarra Espinosa <`zergioibarra@gmail.com`>

Maintainer Sergio Ibarra Espinosa <`zergioibarra@gmail.com`>

Description Emissions inventories elaboration and visualization,
Consists the three stages, pre-processing activity data, processing
or estimating the emissions and post-processing of emissions in
maps and databases.

License MIT + file LICENSE

URL `https://github.com/ibarraespinosa/vein`

BugReports `https://github.com/ibarraespinosa/vein/issues/`

LazyData no

Depends sp, R (>= 2.10)

Imports units, graphics, raster, rgeos, rgdal, stats

Suggests maptools, ggplot2, RQGIS, knitr, rmarkdown, DiagrammeR,
RColorBrewer

RoxygenNote 6.0.1

NeedsCompilation no

VignetteBuilder knitr

R topics documented:

`age_hdv` *Returns amount of vehicles at each age*

Description

Returns amount of vehicles at each age

Usage

```
age_hdv(x, name, a = 0.2, b = 17, agemin = 1, agemax = 50, k = 1,
  bystreet = F)
```

Arguments

`x`	numerical vector of vehicles with length equal to lines features of raod network
`name`	of vehicle assigned to columns of dataframe
`a`	parameter of survival equation
`b`	parameter of survival equation
`agemin`	age of newest vehicles for that category
`agemax`	age of oldest vehicles for that category
`k`	multiplication factor
`bystreet`	when TRUE it is expecting that 'a' and 'b' are numeric vectors with length equal to x

Value

dataframe of age distrubution of vehicles

Examples

```
## Not run:
# Do not run
lt <- Vehicles(rnorm(100, 300, 10))
LT_B5 <- age_hdv(x = lt,name = "LT_B5")
plot(LT_B5)

## End(Not run)
```

age_ldv *Returns amount of vehicles at each age*

Description

Returns amount of vehicles at each age

Usage

```
age_ldv(x, name, a = 1.698, b = -0.2, agemin = 1, agemax = 50, k = 1,
  bystreet = F)
```

Arguments

x	numerical vector of vehicles
name	of vehicle assigned to columns of dataframe
a	parameter of survival equation
b	parameter of survival equation
agemin	age of newest vehicles for that category
agemax	age of oldest vehicles for that category
k	multiplication factor
bystreet	when TRUE it is expecting that 'a' and 'b' are numeric vectors with length equal to x

Value

dataframe of age distrubution of vehicles

Examples

```
## Not run:
# Do not run
pc <- rnorm(100, 300, 10)
PC_E25_1400 <- age_ldv(x = pc,name = "PC_E25_1400")
plot(PC_E25_1400)

## End(Not run)
```

age_moto *Returns amount of vehicles at each age*

Description

Returns amount of vehicles at each age

Usage

```
age_moto(x, name, a = 0.2, b = 17, agemin = 1, agemax = 50, k = 1,
  bystreet = F)
```

Arguments

x	numerical vector of vehicles
name	of vehicle assigned to columns of dataframe
a	parameter of survival equation
b	parameter of survival equation
agemin	age of newest vehicles for that category

agemax	age of oldest vehicles for that category
k	multiplication factor
bystreet	when TRUE it is expecting that 'a' and 'b' are numeric vectors with length equal to x

Value

dataframe of age distrubution of vehicles

Examples

```
## Not run:
# Do not run
m <- rnorm(100, 300, 10)
MOTO_E25_500 <- age_moto(x = m,name = "M_E25_500")
plot(MOTO_E25_500)

## End(Not run)
```

`ef_evap` *Evaporative emission factor*

Description

A lookup table with tier 2 evaporative emission factors from EMEP/EEA emisison guidelines

Usage

```
ef_evap(ef, v, cc, dt, ca, k = 1, show = FALSE)
```

Arguments

ef	Name of evaporative emission factor as *eshotc*: mean hot-soak with carburator, *eswarmc*: mean cold and warm-soak with carburator, eshotfi: mean hot-soak with fuel injection, *erhotc*: mean hot running losses with carburator, *erwarmc* mean cold and warm running losses, *erhotfi* mean hot running losses with fuel injection
v	Type of vehicles, "PC", "Motorcycles", "Motorcycles_2S" and "Moped"
cc	Size of engine in cc. PC "<=1400", "1400_2000" and "2000" Motorcycles_2S: "<=50". Motorcyces: ">50", "<250", "250_750" and ">750"
dt	Average daily temperature variation: "-5_10", "0_15", "10_25" and "20_35"
ca	Size of canister: "no" meaning no canister, "small", "medium" and "large"
k	multiplication factor
show	when TRUE shows row of table with respective emission factor.

Value

emission factors in g/trip or g/proced. The object has class (g) but it order to know it is g/trip or g/proceed the argument show must by T

References

Mellios G and Ntziachristos 2016. Gasoline evaporation. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2009

Examples

```
## Not run:
# Do not run
ef_evap(ef = "erhotc",v = "PC", cc = "<=1400", dt = "0_15", ca = "no",
show = T)

## End(Not run)
```

`ef_hdv_scaled` *Scaling constant with speed emission factors of Heavy Duty Vehicles*

Description

This function creates a list of scaled functions of emission factors. A scaled emission factor which at a speed of the dricing cycle (SDC) gives a desired value. This function needs a dataframe with local emission factors with a columns with the name "Euro_HDV" indicating the Euro equivalence standard, assuming that there are available local emission factors for several consecutive years.

Usage

```
ef_hdv_scaled(df, dfcol, SDC = 34.12, v, t, g, eu, gr, l, p)
```

Arguments

`df`	Dataframe with local emission factor
`dfcol`	Column of the dataframe with the local emission factors eg df$dfcol
`SDC`	Speed of the driving cycle
`v`	Category vehicle: "Coach", "Trucks" or "Ubus"
`t`	Sub-category of of vehicle: "3Axes", "Artic", "Midi", "RT, "Std" and "TT"
`g`	Gross weight of each category: "<=18", ">18", "<=15", ">15 & <=18", "<=7.5", ">7.5 & <=12", ">12 & <=14", ">14 & <=20", ">20 & <=26", ">26 & <=28", ">28 & <=32", ">32", ">20 & <=28", ">28 & <=34", ">34 & <=40", ">40 & <=50" or ">50 & <=60"
`eu`	Euro emission standard: "PRE", "I", "II", "III", "IV" and "V"
`gr`	Gradient or slope of road: -0.06, -0.04, -0.02, 0.00, 0.02. 0.04 or 0.06
`l`	Load of the vehicle: 0.0, 0.5 or 1.0
`p`	Pollutant: "CO", "FC", "NOx" or "HC"

Value

A list of scaled emission factors g/km

Note

The length of the list should be equal to the name of the age categories of a specific type of vehicle

Examples

```
## Not run:
# Do not run
data(fe2015)
co1 <- fe2015[fe2015$Pollutant=="CO",]
FE_LT_7_5_D_CO <- ef_hdv_scaled(co1, co1$LT, v = "Trucks", t = "RT",
g = "<=7.5", eu = co1$Euro_HDV, gr = 0, l = 0.5, p = "CO")
length(FE_LT_7_5_D_CO)

## End(Not run)
```

---

ef_hdv_speed — *Emissions factors for Heavy Duty Vehicles based on average speed*

Description

This function returns speed dependent emission factors. The emission factors comes from the guidelines EMEP/EEA air pollutant emission inventory guidebook http://www.eea.europa.eu/themes/air/emep-eea-air-pollutant-emission-inventory-guidebook

Usage

```
ef_hdv_speed(v, t, g, eu, gr, l, p, k = 1, show.equation = TRUE)
```

Arguments

`v`	Category vehicle: "Coach", "Trucks" or "Ubus"
`t`	Sub-category of of vehicle: "3Axes", "Artic", "Midi", "RT, "Std" and "TT"
`g`	Gross weight of each category: "<=18", ">18", "<=15", ">15 & <=18", "<=7.5", ">7.5 & <=12", ">12 & <=14", ">14 & <=20", ">20 & <=26", ">26 & <=28", ">28 & <=32", ">32", ">20 & <=28", ">28 & <=34", ">34 & <=40", ">40 & <=50" or ">50 & <=60"
`eu`	Euro emission standard: "PRE", "I", "II", "III", "IV" and "V"
`gr`	Gradient or slope of road: -0.06, -0.04, -0.02, 0.00, 0.02. 0.04 or 0.06
`l`	Load of the vehicle: 0.0, 0.5 or 1.0
`p`	Pollutant: "CO", "FC", "NOx" or "HC"
`k`	Multiplication factor
`show.equation`	Option to see or not the equation parameters

Value

an emission factor function which depends of the average speed V g/km

Examples

```
## Not run:
# Do not run
V <- 0:130
ef1 <- ef_hdv_speed(v = "Trucks",t = "RT", g = "<=7.5", e = "I", gr = 0,l = 0.5, p = "CO")
plot(1:130, ef1(1:130))
ef2 <- ef_hdv_speed(v = "Trucks",t = "RT", g = "<=7.5", e = "II", gr = 0,l = 0.5, p = "THC")
plot(1:130, ef2(1:130))
ef3 <- ef_hdv_speed(v = "Trucks",t = "RT", g = "<=7.5", e = "II", gr = 0,l = 0.5, p = "PM")
plot(1:130, ef3(1:130))

## End(Not run)
```

ef_ldv_cold *Cold-Start Emissions factors for Light Duty Vehicles*

Description

This function returns speed functions which depends on ambient temperature average speed. The emission factors comes from the guidelines EMEP/EEA air pollutant emission inventory guidebook http://www.eea.europa.eu/themes/air/emep-eea-air-pollutant-emission-inventory-guidebook

Usage

```
ef_ldv_cold(v = "LDV", ta, cc, f, eu, p, k = 1, show.equation = FALSE)
```

Arguments

v	Category vehicle: "LDV"
ta	Ambient temperature. Monthly men can be used
cc	Size of engine in cc: "<=1400", "1400_2000" or ">2000"
f	Type of fuel: "G", "D" or "LPG"
eu	Euro standard: "PRE", "I", "II", "III", "IV", "V", "VI" or "VIc"
p	Pollutant: "CO", "FC", "NOx", "HC" or "PM"
k	Multiplication factor
show.equation	Option to see or not the equation parameters

Value

an emission factor function which depends of the average speed V and ambient temperature. g/km

Examples

```
## Not run:
# Do not run
V <- 0:150
ef1 <- ef_ldv_cold(ta = 15, cc = "<=1400", f ="G", eu = "I",
p = "CO")
ef1(10)

## End(Not run)
```

ef_ldv_cold_list *List of cold start emission factors of Light Duty Vehicles*

Description

This function creates a list of functions of cold start emission factors considering different euro emission standard to the elements of the list.

Usage

```
ef_ldv_cold_list(df, v = "LDV", ta, cc, f, eu, p)
```

Arguments

df	Dataframe with local emission factor
v	Category vehicle: "LDV"
ta	ambient temperature. Montly average van be used
cc	Size of engine in cc: <=1400", "1400_2000" and ">2000"
f	Type of fuel: "G" or "D"
eu	character vector of euro standards: "PRE", "I", "II", "III", "IV", "V", "VI" or "VIc".
p	Pollutant: "CO", "FC", "NOx", "HC" or "PM"

Value

A list of cold start emission factors g/km

Note

The length of the list should be equal to the name of the age categories of a specific type of vehicle

Examples

```
## Not run:
# Do not run
df <- data.frame(age1 = c(1,1), age2 = c(2,2))
eu = c("I", "PRE")
l <- ef_ldv_cold(t = 17, cc = "<=1400", f = "G",
eu = "I", p = "CO")
l_cold <- ef_ldv_cold_list(df, t = 17, cc = "<=1400", f = "G",
eu = eu, p = "CO")
length(l_cold)

## End(Not run)
```

ef_ldv_scaled *Scaling constant with speed emission factors of Light Duty Vehicles*

Description

This function creates a list of scaled functions of emission factors. A scaled emission factor which at a speed of the driving cycle (SDC) gives a desired value.

Usage

```
ef_ldv_scaled(df, dfcol, SDC = 34.12, v, t, cc, f, eu, p)
```

Arguments

df	Dataframe with local emission factor
dfcol	Column of the dataframe with the local emission factors eg df$dfcol
SDC	Speed of the driving cycle
v	Category vehicle: "PC", "LCV", "Motorcycle" or "Moped
t	Sub-category of of vehicle: "PRE_ECE", "ECE_1501", "ECE_1502", "ECE_1503", "ECE_1504" , "IMPROVED_CONVENTIONAL", "OPEN_LOOP", "ALL", "2S" or "4S"
cc	Size of engine in cc: "ALL", "<=1400", ">1400", "1400_2000", ">2000", "<=800", "800_1400", "<=2000", "2S", "<=50", ">=50", "<=250", "250_750", ">=750", or ">50"
f	Type of fuel: "G", "D", "LPG" or "FH" (Full Hybrid: starts by electric motor)
eu	Euro standard: "PRE", "I", "II", "III", "III+DPF", "IV", "V", "VI", "VIc" or "ALL"
p	Pollutant: "CO", "FC", "NOx", "HC" or "PM"

Details

This function calls "ef_ldv_speed" and calculate the specific k value, dividing the local emission factor by the respective speed emissions factor at the speed representative of the local emission factor, e.g. If the local emission factors were tested with the FTP-75 test procedure, SDC = 34.12 km/h.

Value

A list of scaled emission factors g/km

Note

The length of the list should be equal to the name of the age categories of a specific type of vehicle. Thanks to Glauber Camponogara by the help.

See Also

ef_ldv_seed

Examples

```
## Not run:
# Do not run
data(fe2015)
co1 <- fe2015[fe2015$Pollutant=="CO" & fe2015$Age<25, ] #24 obs!!!
l1 <- ef_ldv_scaled(co1, co1$PC_G, v = "PC", t = "ALL", cc = "ALL", f = "G",
eu = co1$Euro_LDV, p = "CO")
co2 <- fe2015[fe2015$Pollutant=="CO" & fe2015$Age>24,] #22 obs!!!
l2 <- ef_ldv_scaled(co2$PC_G, v = "PC", t = "PRE_ECE", cc = "ALL", f = "G",
eu = co2$Euro_LDV, p = "CO")
FE_PC_E25_1400_CO <- c(l1,l2,l2[[12]],l2[[12]],l2[[12]],l2[[12]])
FE_PC_E25_1400_CO[[1]](34.12) #first element
length(FE_PC_E25_1400_CO)

## End(Not run)
```

`ef_ldv_speed` *Emissions factors for Light Duty Vehicles and Motorcycles*

Description

This function returns speed dependent emission factors. The emission factors comes from the guidelines EMEP/EEA air pollutant emission inventory guidebook http://www.eea.europa.eu/themes/air/emep-eea-air-pollutant-emission-inventory-guidebook

Usage

```
ef_ldv_speed(v, t, cc, f, eu, p, k = 1, show.equation = TRUE)
```

Arguments

`v`	Category vehicle: "PC", "LCV", "Motorcycle" or "Moped
`t`	Sub-category of of vehicle: "PRE_ECE", "ECE_1501", "ECE_1502", "ECE_1503", "ECE_1504" , "IMPROVED_CONVENTIONAL", "OPEN_LOOP", "ALL", "2S" or "4S"
`cc`	Size of engine in cc: "ALL", "<=1400", ">1400", "1400_2000", ">2000", "<=800", "800_1400", "<=2000", "2S", "<=50", ">=50", "<=250", "250_750", ">=750", or ">50"
`f`	Type of fuel: "G", "D", "LPG" or "FH" (Full Hybrid: starts by electric motor)
`eu`	Euro standard: "PRE", "I", "II", "III", "III+DPF", "IV", "V", "VI", "VIc" or "ALL"
`p`	Pollutant: "CO", "FC", "NOx", "HC" or "PM"
`k`	Multiplication factor
`show.equation`	Option to see or not the equation parameters

Value

an emission factor function which depends of the average speed V g/km

Examples

```
## Not run:
# Do not run
V <- 0:150
ef1 <- ef_ldv_speed(v = "PC",t = "PRE_ECE", cc = "ALL", f = "G", eu = "PRE", p = "CO")
plot(1:150, ef1(1:150))

## End(Not run)
```

`ef_nitro` *Emissions factors of N2O and NH3*

Description

This function returns emission factors as a functions of accumulated mileage. The emission factors comes from the guidelines EMEP/EEA air pollutant emission inventory guidebook http://www.eea.europa.eu/themes/air/eme
eea-air-pollutant-emission-inventory-guidebook

Usage

```
ef_nitro(v, t, cc, f, eu, p, S, k = 1, show.equation = TRUE)
```

Arguments

`v`	Category vehicle: "PC", "LCV", "Motorcycle" or "Moped
`t`	Type: "PC", "LCV", "LDV", "Motorcycles", "Trucks", "HDV", "HDV-A", "BUS" and "Coach"
`cc`	"Cold", "Hot", "<50", ">=50", ">3.5", "7.5_12", "12_28", "28_34", ">34", "ALL".
`f`	Type of fuel: "G", "D" or "LPG"
`eu`	Euro standard: "PRE", "I", "II", "III", "III+DPF", "IV", "V", "VI", "VIc", "2S", 4S" and "ALL"
`p`	Pollutant: "N2O", "NH3"
`S`	Sulphur (ppm)
`k`	Multiplication factor
`show.equation`	Option to see or not the equation parameters

Value

an emission factor function which depends of the average speed V g/km

Examples

```
## Not run:
# Do not run
V <- 0:150
ef1 <- ef_ldv_speed(v = "PC",t = "PRE_ECE", cc = "ALL", f = "G", eu = "PRE", p = "CO")
plot(1:150, ef1(1:150))

## End(Not run)
```

`ef_wear` *Emissions factors from tyre, break and road surface wear*

Description

Estimation of wear emissions. The sources are tyres, breaks and road surface.

Usage

```
ef_wear(wear, type, pol = "TSP", speed, load = 0.5, axle = 2)
```

Arguments

`wear`	Type of wear: "tyre", "break" and "road"
`type`	TYpe of vehicle: "2W", "PC", "LCV", 'HDV"
`pol`	Pollutant: "TSP", "PM10", "PM2.5", "PM1" and "PM0.1"
`speed`	List of speeds
`load`	Load of the HDV
`axle`	Number of axle of the HDV

Value

emission factors grams/km

References

Ntziachristos and Boulter 2016. Automobile tyre and break wear and road abrasion. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2016

Examples

```
## Not run:
# Do not run

## End(Not run)
```

`emis` *Emissions estimation hourly for the of the week*

Description

The vehicular emissions are estimated as the product of the vehicles on a road, length of the road, emission factor avaliated at the respective speed. $E = VEH * LENGTH * EF(speed)$

Usage

```
emis(veh, lkm, ef, speed, agemax, profile, hour = 1, day = 1, array = F)
```

Arguments

`veh`	Numeric vector with length of elements equals to number of streets
`lkm`	Length of each link
`ef`	List of functions of emission factors
`speed`	List of speeds
`agemax`	Age of oldest vehicles for that category
`profile`	Numerical or dataframe with nrows equal to 24 and ncol 7 day of the week
`hour`	Number of considered hours in estimation
`day`	Number of considered days in estimation
`array`	When FALSE produces a dataframe of the estimation. When TRUE expects a profile as a dataframe producing an array with dimensions (streets x columns x hours x days)

Value

emission estimation g/h

Examples

```
## Not run:
# Do not run
data(net)
data(pc_profile)
data(fe2015)
data(fkm)
PC_G <- c(33491,22340,24818,31808,46458,28574,24856,28972,37818,49050,87923,
          133833,138441,142682,171029,151048,115228,98664,126444,101027,
          84771,55864,36306,21079,20138,17439, 7854,2215,656,1262,476,512,
          1181, 4991, 3711, 5653, 7039, 5839, 4257,3824, 3068)
veh <- data.frame(PC_G = PC_G)
pc1 <- my_age(x = net$ldv, y = PC_G, name = "PC")
pcw <- temp_fact(net$ldv+net$hdv, pc_profile)
speed <- netspeed(pcw, net$ps, net$ffs, net$capacity, net$lkm, alpha = 1,
isList = T)
pckm <- fkm[[1]](1:24); pckma <- cumsum(pckm)
cod1 <- emis_det(po = "CO", cc = 1000, eu = "III", km = pckma[1:11])
cod2 <- emis_det(po = "CO", cc = 1000, eu = "I", km = pckma[12:24])
#vehicles newer than pre-euro
co1 <- fe2015[fe2015$Pollutant=="CO", ] #24 obs!!!
cod <- c(co1$PC_G[1:24]*c(cod1,cod2),co1$PC_G[25:nrow(co1)])
lef <- ef_ldv_scaled(co1, cod, v = "PC", t = "ALL", cc = "ALL",
                     f = "G",p = "CO", eu=co1$Euro_LDV)
lef <- c(lef,lef[length(lef)],lef[length(lef)],lef[length(lef)],
         lef[length(lef)],lef[length(lef)])
E_CO <- emis(veh = pc1,lkm = net$lkm, ef = lef, speed = speed, agemax = 41,
             profile = pc_profile, hour = 24, day = 7, array = T)
class(E_CO)

## End(Not run)
```

EmissionFactors *Construction function for class "EmissionFactors"*

Description

Returns a tranformed object with class "EmissionFactors" and units g/km.

Usage

```
EmissionFactors(x, ...)

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

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

```
## S3 method for class 'EmissionFactors'
plot(x, ...)
```

Arguments

`x`	Object with class "data.frame", "matrix" or "numeric"
`...`	ignored
`object`	Object with class "EmissionFactors"

Value

Objects of class "EmissionFactors" or "units"

Examples

```
## Not run:
data(fe2015)
names(fe2015)
class(fe2015)
df <- fe2015[fe2015$Pollutant=="CO", c(ncol(fe2015)-1,ncol(fe2015))]
ef1 <- EmissionFactors(df)
class(ef1)
summary(ef1)
plot(ef1)

## End(Not run)
```

`EmissionFactorsList` *Construction function for class "EmissionFactorsList"*

Description

Returns a tranformed object with class"EmissionsFactorsList".

Usage

```
EmissionFactorsList(x, ...)

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

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

## S3 method for class 'EmissionFactorsList'
plot(x, ...)
```

Arguments

`x`	Object with class "list"
`...`	ignored
`object`	Object with class "EmissionFactorsList"

Value

Objects of class "EmissionFactorsList"

Examples

```
## Not run:
data(fe2015)
names(fe2015)
class(fe2015)
df <- fe2015[fe2015$Pollutant=="CO", c(ncol(fe2015)-1,ncol(fe2015))]
ef1 <- EmissionFactorsList(df)
class(ef1)
length(ef1)
length(ef1[[1]])
summary(ef1)
ef1

## End(Not run)
```

`Emissions` *Construction function for class "Emissions"*

Description

Returns a tranformed object with class "Emissions". The type of objects supported are of classes "matrix", "data.frame" and "numeric". If the class of the object is "matrix" this function returns a dataframe.

Usage

```
Emissions(x, ...)

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

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

## S3 method for class 'Emissions'
plot(x, ...)
```

Arguments

`x`	Object with class "data.frame", "matrix" or "numeric"
`...`	ignored
`object`	object with class "Emissions"

Value

Objects of class "Emissions" or "units"

Examples

```
## Not run:
data(net)
data(pc_profile)
data(fe2015)
data(fkm)
PC_G <- c(33491,22340,24818,31808,46458,28574,24856,28972,37818,49050,87923,
          133833,138441,142682,171029,151048,115228,98664,126444,101027,
          84771,55864,36306,21079,20138,17439, 7854,2215,656,1262,476,512,
          1181, 4991, 3711, 5653, 7039, 5839, 4257,3824, 3068)
veh <- data.frame(PC_G = PC_G)
pc1 <- my_age(x = net$ldv, y = PC_G, name = "PC")
pcw <- temp_fact(net$ldv+net$hdv, pc_profile)
speed <- netspeed(pcw, net$ps, net$ffs, net$capacity, net$lkm, alpha = 1,
isList = T)
pckm <- fkm[[1]](1:24); pckma <- cumsum(pckm)
cod1 <- emis_det(po = "CO", cc = 1000, eu = "III", km = pckma[1:11])
cod2 <- emis_det(po = "CO", cc = 1000, eu = "I", km = pckma[12:24])
#vehicles newer than pre-euro
co1 <- fe2015[fe2015$Pollutant=="CO", ] #24 obs!!!
cod <- c(co1$PC_G[1:24]*c(cod1,cod2),co1$PC_G[25:nrow(co1)])
lef <- ef_ldv_scaled(co1, cod, v = "PC", t = "ALL", cc = "ALL",
                     f = "G",p = "CO", eu=co1$Euro_LDV)
lef <- c(lef,lef[length(lef)],lef[length(lef)],lef[length(lef)],
         lef[length(lef)],lef[length(lef)])
E_CO <- emis(veh = pc1,lkm = net$lkm, ef = lef, speed = speed, agemax = 41,
             profile = pc_profile, hour = 24, day = 7, array = T)
dim(E_CO) # streets x vehicle categories x hours x days
class(E_CO[ , , 1, 1])
df <- Emissions(E_CO[ , , 1, 1]) # Firt hour x First day
class(df)
summary(df)
head(df)
plot(df)

## End(Not run)
```

EmissionsArray *Construction function for class "EmissionsArray"*

Description

Returns a tranformed object with class "EmissionsArray".

Usage

```
EmissionsArray(x, ...)

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

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

## S3 method for class 'EmissionsArray'
plot(x, ...)
```

Arguments

x	Object with class "data.frame", "matrix" or "numeric"
...	ignored
object	object with class "EmissionsArray'

Value

Objects of class "EmissionsArray"

Examples

```
## Not run:
data(net)
data(pc_profile)
data(fe2015)
data(fkm)
PC_G <- c(33491,22340,24818,31808,46458,28574,24856,28972,37818,49050,87923,
          133833,138441,142682,171029,151048,115228,98664,126444,101027,
          84771,55864,36306,21079,20138,17439, 7854,2215,656,1262,476,512,
          1181, 4991, 3711, 5653, 7039, 5839, 4257,3824, 3068)
veh <- data.frame(PC_G = PC_G)
pc1 <- my_age(x = net$ldv, y = PC_G, name = "PC")
pcw <- temp_fact(net$ldv+net$hdv, pc_profile)
speed <- netspeed(pcw, net$ps, net$ffs, net$capacity, net$lkm, alpha = 1,
isList = T)
pckm <- fkm[[1]](1:24); pckma <- cumsum(pckm)
cod1 <- emis_det(po = "CO", cc = 1000, eu = "III", km = pckma[1:11])
```

```
cod2 <- emis_det(po = "CO", cc = 1000, eu = "I", km = pckma[12:24])
#vehicles newer than pre-euro
co1 <- fe2015[fe2015$Pollutant=="CO", ] #24 obs!!!
cod <- c(co1$PC_G[1:24]*c(cod1,cod2),co1$PC_G[25:nrow(co1)])
lef <- ef_ldv_scaled(co1, cod, v = "PC", t = "ALL", cc = "ALL",
                     f = "G",p = "CO", eu=co1$Euro_LDV)
lef <- c(lef,lef[length(lef)],lef[length(lef)],lef[length(lef)],
         lef[length(lef)],lef[length(lef)])
E_CO <- emis(veh = pc1,lkm = net$lkm, ef = lef, speed = speed, agemax = 41,
             profile = pc_profile, hour = 24, day = 7, array = T)
class(E_CO)
summary(E_CO)
E_CO
plot(E_CO)

## End(Not run)
```

EmissionsList *Construction function for class "EmissionsList"*

Description

Returns a tranformed object with class "EmissionsList".

Usage

```
EmissionsList(x, ...)

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

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

## S3 method for class 'EmissionsList'
plot(x, ...)
```

Arguments

`x`	object with class "EmissionList"
`...`	ignored
`object`	object with class "EmissionList"

Value

Objects of class "EmissionsList" and numeric elements as "units"

Examples

```
## Not run:
data(net)
data(pc_profile)
data(fe2015)
data(fkm)
PC_G <- c(33491,22340,24818,31808,46458,28574,24856,28972,37818,49050,87923,
          133833,138441,142682,171029,151048,115228,98664,126444,101027,
          84771,55864,36306,21079,20138,17439, 7854,2215,656,1262,476,512,
          1181, 4991, 3711, 5653, 7039, 5839, 4257,3824, 3068)
veh <- data.frame(PC_G = PC_G)
pc1 <- my_age(x = net$ldv, y = PC_G, name = "PC")
pcw <- temp_fact(net$ldv+net$hdv, pc_profile)
speed <- netspeed(pcw, net$ps, net$ffs, net$capacity, net$lkm, alpha = 1,
isList = T)
pckm <- fkm[[1]](1:24); pckma <- cumsum(pckm)
cod1 <- emis_det(po = "CO", cc = 1000, eu = "III", km = pckma[1:11])
cod2 <- emis_det(po = "CO", cc = 1000, eu = "I", km = pckma[12:24])
#vehicles newer than pre-euro
co1 <- fe2015[fe2015$Pollutant=="CO", ] #24 obs!!!
cod <- c(co1$PC_G[1:24]*c(cod1,cod2),co1$PC_G[25:nrow(co1)])
lef <- ef_ldv_scaled(co1, cod, v = "PC", t = "ALL", cc = "ALL",
                     f = "G",p = "CO", eu=co1$Euro_LDV)
lef <- c(lef,lef[length(lef)],lef[length(lef)],lef[length(lef)],
         lef[length(lef)],lef[length(lef)])
E_CO <- emis(veh = pc1,lkm = net$lkm, ef = lef, speed = speed, agemax = 41,
             profile = pc_profile, hour = 24, day = 7, array = F)
class(E_CO)

## End(Not run)
```

---

emis_cold — *Estimation of cold start emissions hourly for the of the week*

Description

The vehicular emissions are estimated as the product of the vehicles on a road, length of the road, emission factor avaliated at the respective speed. The estimation considers beta parameter, the fraction of mileage driven

Usage

```
emis_cold(veh, lkm, ef, efcold, beta, speed, agemax, profile, hour = 1,
  day = 1, array = F)
```

Arguments

veh	Numeric vector with length of elements equals to number of streets
lkm	Length of each link

`ef`	List of functions of emission factors
`efcold`	List of functions of cold start emission factors
`beta`	Datraframe with the hourly cold-start distribution to each day of the period. Number of rows are hours and columns are days
`speed`	List of speeds
`agemax`	Age of oldest vehicles for that category
`profile`	Numerical or dataframe with nrows equal to 24 and ncol 7 day of the week
`hour`	Number of considered hours in estimation
`day`	Number of considered days in estimation
`array`	When FALSE produces a dataframe of the estimation. When TRUE expects a profile as a dataframe producing an array with dimensions (streets x columns x hours x days)

Value

EmissionsArray g/h

Note

Actually dcold is not necessary, it would be enough to multiply an existing cold-start distribution with the daily profile, but it was added because it is important to clarify both, the data and the concepts

Examples

```
## Not run:
# Do not run
data(net)
data(pc_profile)
data(fe2015)
data(fkm)
data(pc_cold)
pcf <- as.data.frame(cbind(pc_cold,pc_cold,pc_cold,pc_cold,pc_cold,pc_cold,
pc_cold))
PC_G <- c(33491,22340,24818,31808,46458,28574,24856,28972,37818,49050,87923,
          133833,138441,142682,171029,151048,115228,98664,126444,101027,
          84771,55864,36306,21079,20138,17439, 7854,2215,656,1262,476,512,
          1181, 4991, 3711, 5653, 7039, 5839, 4257,3824, 3068)
veh <- data.frame(PC_G = PC_G)
pc1 <- my_age(x = net$ldv, y = PC_G, name = "PC")
pcw <- temp_fact(net$ldv+net$hdv, pc_profile)
speed <- netspeed(pcw, net$ps, net$ffs, net$capacity, net$lkm, alpha = 1,
isList = T)
pckm <- fkm[[1]](1:24); pckma <- cumsum(pckm)
cod1 <- emis_det(po = "CO", cc = 1000, eu = "III", km = pckma[1:11])
cod2 <- emis_det(po = "CO", cc = 1000, eu = "I", km = pckma[12:24])
#vehicles newer than pre-euro
co1 <- fe2015[fe2015$Pollutant=="CO", ] #24 obs!!!
cod <- c(co1$PC_G[1:24]*c(cod1,cod2),co1$PC_G[25:nrow(co1)])
```

```
lef <- ef_ldv_scaled(co1, cod, v = "PC", t = "ALL", cc = "ALL",
                     f = "G",p = "CO", eu=co1$Euro_LDV)
lef <- c(lef,lef[length(lef)],lef[length(lef)],lef[length(lef)],
         lef[length(lef)],lef[length(lef)])
# Mohtly average temperature 18 Celcius degrees
lefc <- ef_ldv_cold_list(df = co1, ta = 18, cc = "<=1400", f = "G",
                         eu = co1$Euro_LDV, p = "CO" )
lefec <- c(lefc,lefc[[length(lefc)]],lefc[[length(lefc)]],
            lefc[[length(lefc)]],lefc[[length(lefc)]],lef[[length(lef)]])
class(lefec)
PC_CO_COLD <- emis_cold(veh = pc1, lkm = net$lkm, ef = lef, efcold = lefec,
beta = pcf, speed = speed, agemax = 41, profile = pc_profile, hour = 24,
day = 7, array = T)
class(PC_CO_COLD)
plot(PC_CO_COLD)

## End(Not run)
```

---

emis_det   *Determine deterioration factors for urban conditions*

Description

This function returns deterioration factors. The emission factors comes from the guidelines for developing emission factors of the EMEP/EEA air pollutant emission inventory guidebook http://www.eea.europa.eu/themes/air eea-air-pollutant-emission-inventory-guidebook This function subset an internal database of emission factors with each argument

Usage

```
emis_det(po, cc, eu, km)
```

Arguments

`po`	Pollutant
`cc`	Size of engine in cc
`eu`	Euro standard: "PRE", "I", "II", "III", "III", "IV", "V", "VI"
`km`	mileage in km

Value

It returns a numeric vector without "units"

Examples

```
## Not run:
# Do not run

## End(Not run)
```

emis_evap *Estimation of evaporative emissions*

Description

Estimation of evaporative emissions from EMEP/EEA emisison guidelines

Usage

```
emis_evap(veh, name, size, fuel, aged, nd4, nd3, nd2, nd1, hs_nd4, hs_nd3,
  hs_nd2, hs_nd1, rl_nd4, rl_nd3, rl_nd2, rl_nd1, d_nd4, d_nd3, d_nd2, d_nd1)
```

Arguments

veh	Total number of vehicles by age of use
name	Character of type of vehicle
size	Character of size of vehicle
fuel	Character of fuel of vehicle
aged	Age distribution vector. E.g.: 1:40
nd4	Number of days with temperature between 20 and 35 celcius degrees
nd3	Number of days with temperature between 10 and 25 celcius degrees
nd2	Number of days with temperature between 0 and 15 celcius degrees
nd1	Number of days with temperature between -5 and 10 celcius degrees
hs_nd4	average daily hot-soak evaporative emissions for days with temperature between 20 and 35 celcius degrees
hs_nd3	average daily hot-soak evaporative emissions for days with temperature between 10 and 25 celcius degrees
hs_nd2	average daily hot-soak evaporative emissions for days with temperature between 0 and 15 celcius degrees
hs_nd1	average daily hot-soak evaporative emissions for days with temperature between -5 and 10 celcius degrees
rl_nd4	average daily running losses evaporative emissions for days with temperature between 20 and 35 celcius degrees
rl_nd3	average daily running losses evaporative emissions for days with temperature between 10 and 25 celcius degrees
rl_nd2	average daily running losses evaporative emissions for days with temperature between 0 and 15 celcius degrees
rl_nd1	average daily running losses evaporative emissions for days with temperature between -5 and 10 celcius degrees
d_nd4	average daily diurnal evaporative emissions for days with temperature between 20 and 35 celcius degrees

d_nd3	average daily diurnal evaporative emissions for days with temperature between 10 and 25 celcius degrees
d_nd2	average daily diurnal evaporative emissions for days with temperature between 0 and 15 celcius degrees
d_nd1	average daily diurnal evaporative emissions for days with temperature between -5 and 10 celcius degrees

Value

dataframe of emission estimation in grams/days

References

Mellios G and Ntziachristos 2016. Gasoline evaporation. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2009

Examples

```
## Not run:
# Do not run
ef1 <- ef_evap(ef = "erhotc",v = "PC", cc = "<=1400", dt = "0_15", ca = "no")
dfe <- emis_evap(rep(50,3),"PC","<=1400","G", 1:3,
                    10,4,2,1,
                    ef1*1:3, ef1*1:3, ef1*1:3, ef1*1:3,
                    ef1*1:3, ef1*1:3, ef1*1:3, ef1*1:3,
                    ef1*1:3, ef1*1:3, ef1*1:3, ef1*1:3)

## End(Not run)
```

---

emis_grid *Allocate emissions into a grid*

Description

The allocation is proportionally to each grid cell. The process is performed by intersection between geometries and the grid. Geometries suported, so, far are lines with raster::intersect and points with sp::over. The allocation of lines is by interaction, then update the pollutant values according the new length of road inside each grid cell. It means that requires "sr" according with your location for the projection. It is assumed that soobj is a spatial*DataFrame with the pollutant in data. Also, it is required that, when is a SpatialLinesDataFrame, there is a field called lkm, with the length of the road, in this case, in km.

Usage

```
emis_grid(spobj, g, sr, type = "lines")
```

Arguments

spobj	A spatial dataframe of class sp
g	A grid with class SpatialPolygonsDataFrame
sr	Spatial reference, default is "+init=epsg:4326"
type	type of geometry: "lines" or "points"

Details

This function accepts data with "units" but they are converted internally to numeric and then return SpatialPolygonsDataFrame with numeric data.frame

Examples

```
## Not run:
data(net)
data(pc_profile)
data(fe2015)
data(fkm)
PC_G <- c(33491,22340,24818,31808,46458,28574,24856,28972,37818,49050,87923,
          133833,138441,142682,171029,151048,115228,98664,126444,101027,
          84771,55864,36306,21079,20138,17439, 7854,2215,656,1262,476,512,
          1181, 4991, 3711, 5653, 7039, 5839, 4257,3824, 3068)
veh <- data.frame(PC_G = PC_G)
pc1 <- my_age(x = net$ldv, y = PC_G, name = "PC")
pcw <- temp_fact(net$ldv+net$hdv, pc_profile)
speed <- netspeed(pcw, net$ps, net$ffs, net$capacity, net$lkm, alpha = 1,
isList = T)
pckm <- fkm[[1]](1:24); pckma <- cumsum(pckm)
cod1 <- emis_det(po = "CO", cc = 1000, eu = "III", km = pckma[1:11])
cod2 <- emis_det(po = "CO", cc = 1000, eu = "I", km = pckma[12:24])
#vehicles newer than pre-euro
co1 <- fe2015[fe2015$Pollutant=="CO", ] #24 obs!!!
cod <- c(co1$PC_G[1:24]*c(cod1,cod2),co1$PC_G[25:nrow(co1)])
lef <- ef_ldv_scaled(co1, cod, v = "PC", t = "ALL", cc = "ALL",
                     f = "G",p = "CO", eu=co1$Euro_LDV)
lef <- c(lef,lef[length(lef)],lef[length(lef)],lef[length(lef)],
         lef[length(lef)],lef[length(lef)])
E_CO <- emis(veh = pc1,lkm = net$lkm, ef = lef, speed = speed, agemax = 41,
             profile = pc_profile, hour = 24, day = 7, array = T)
# arguments required: arra, pollutant ad by
E_CO_STREETS <- emis_post(arra = E_CO, pollutant = "CO", by = "streets_wide")
net@data <- cbind(net@data, E_CO_STREETS)
head(net@data)
g <- make_grid(net, 1/102.47/2, 1/102.47/2, polygon = T) #500m in degrees
net@data <- net@data[,- c(1:9)]
names(net)
E_CO_g <- emis_grid(spobj = net, g = g, sr= "+init=epsg:31983")
head(E_CO_g@data)
library(RColorBrewer)
spplot(E_CO_g, "V138", scales=list(draw=T),cuts=8,
colorkey = list(space = "bottom", height = 1),
col.regions=brewer.pal(9, "Blues"),
sp.layout = list("sp.lines", net, pch = 16, cex = 2, col = "black"))

## End(Not run)
```

---

emis_paved *Estimation of resuspension emissions from paved roads*

Description

The vehicular emissions are estimated as the product of the vehicles on a road, length of the road, emission factor from AP42 13.2.1 Paved roads. It is assumed dry hours and anual aggregation should consider moisture factor. It depends on Average Daily Traffic (ADT)

Usage

```
emis_paved(veh, lkm, k, sL1, sL2, sL3, sL4, W)
```

Arguments

veh	Numeric vector with length of elements equals to number of streets It is an array with dimenssions number of streets x hours of day x days of week
lkm	Length of each link
k	K_PM30 = 3.23, K_PM15 = 0.77, K_PM10 = 0.62 and K_PM2.5 = 0.15
sL1	Silt loading (g/m2) for roads with ADT <= 500
sL2	Silt loading (g/m2) for roads with ADT > 500 and <= 5000
sL3	Silt loading (g/m2) for roads with ADT > 5000 and <= 1000
sL4	Silt loading (g/m2) for roads with ADT > 10000
W	array of dimensions of veh. It consists in the hourly averaged weight of traffic fleet in each road

Value

emission estimation g/h

References

EPA, 2016. Emission factor documentation for AP-42. Section 13.2.1, Paved Roads. https://www3.epa.gov/ttn/chief/ap42/ch

Examples

```
## Not run:
# Do not run
veh <- array(pnorm(q=c(1:100), mean=500, sd = 100),
             c(100,24,7))
W <- veh*1e+05
lkm <-  rnorm(n = 100, mean = 10, sd = 1)
sL1 <- 0.6
emi  <- emis_paved(veh = veh, lkm = lkm, k = 0.65,
                       sL1 = sL1, sL2 = sL1/4, sL3 = sL1/16, sL4 = sL1/32,
                       W = W)

## End(Not run)
```

---

emis_post *Post emissions*

Description

Simplify emissions estimated as total per type category of vehicle or by street. It reads array of emissions

Usage

```
emis_post(arra, veh, size, fuel, pollutant, by = "veh")
```

Arguments

arra	Array of emissions (streets x category of vehicles x hours x days)
veh	Type of vehicle
size	Size or weight
fuel	Fuel
pollutant	Pollutant
by	Type of output, "veh" for total vehicular category , "streets_narrow" or "streets_wide". "streets_wide" returns a dataframe with rows as number of streets and columns the hours as days*hours considered, e.g. 168 columns as the hours of a whole week and "streets_wide repeats the row number of streets by hour and day of the week

Examples

```
## Not run:
# Do not run
data(net)
data(pc_profile)
data(fe2015)
data(fkm)
```

```
PC_G <- c(33491,22340,24818,31808,46458,28574,24856,28972,37818,49050,87923,
          133833,138441,142682,171029,151048,115228,98664,126444,101027,
          84771,55864,36306,21079,20138,17439, 7854,2215,656,1262,476,512,
          1181, 4991, 3711, 5653, 7039, 5839, 4257,3824, 3068)
veh <- data.frame(PC_G = PC_G)
pc1 <- my_age(x = net$ldv, y = PC_G, name = "PC")
pcw <- temp_fact(net$ldv+net$hdv, pc_profile)
speed <- netspeed(pcw, net$ps, net$ffs, net$capacity, net$lkm, alpha = 1,
isList = T)
pckm <- fkm[[1]](1:24); pckma <- cumsum(pckm)
cod1 <- emis_det(po = "CO", cc = 1000, eu = "III", km = pckma[1:11])
cod2 <- emis_det(po = "CO", cc = 1000, eu = "I", km = pckma[12:24])
#vehicles newer than pre-euro
co1 <- fe2015[fe2015$Pollutant=="CO", ] #24 obs!!!
cod <- c(co1$PC_G[1:24]*c(cod1,cod2),co1$PC_G[25:nrow(co1)])
lef <- ef_ldv_scaled(co1, cod, v = "PC", t = "ALL", cc = "ALL",
                     f = "G",p = "CO", eu=co1$Euro_LDV)
lef <- c(lef,lef[length(lef)],lef[length(lef)],lef[length(lef)],
         lef[length(lef)],lef[length(lef)])
E_CO <- emis(veh = pc1,lkm = net$lkm, ef = lef, speed = speed, agemax = 41,
             profile = pc_profile, hour = 24, day = 7, array = T)
# arguments required: arra, pollutant ad by
E_CO_STREETS <- emis_post(arra = E_CO, pollutant = "CO", by = "streets_wide")
summary(E_CO_STREETS)
# arguments required: arra, veh, size, fuel, pollutant ad by
E_CO_DF <- emis_post(arra = E_CO,  veh = "PC", size = "<1400", fuel = "G",
pollutant = "CO", by = "veh")
head(E_CO_DF)

## End(Not run)
```

`emis_wear` *Emission estimation from tyre, break and road surface wear*

Description

Estimation of wear emissions. The sources are tyres, breaks and road surface.

Usage

```
emis_wear(veh, lkm, ef, agemax, profile, hour = 1, day = 1)
```

Arguments

veh	Object of class "Vehicles"
lkm	Length of the road
ef	list of emission factor functions class "EmissionFactorsList", length equals to hours.
agemax	Age of oldest vehicles for that category

`profile`	Numerical or dataframe with nrows equal to 24 and ncol 7 day of the week
`hour`	Number of considered hours in estimation
`day`	Number of considered days in estimation

Value

emission estimation g/h

References

Ntziachristos and Boulter 2016. Automobile tyre and break wear and road abrasion. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2016

Examples

```
## Not run:
# Do not run

## End(Not run)
```

`emis_wrf` *Generates emissions dataframe to generate WRF-Chem inputs*

Description

It returns a dataframes with columns lat, long, id, pollutants, local time and GMT time. This dataframe has the proper format to be used with WRF assimilation system: "Another Asimilation System 4 WRF (AAS4WRF)" as published by Vera-Vala et al (2016)

Usage

```
emis_wrf(sdf, nr, dmyhm, tz, utc, islist)
```

Arguments

`sdf`	Grid emissions, which can be a SpatialPolygonsDataFrame, or a list of SpatialPolygonsDataFrame
`nr`	Number of repetitions of the emissions period
`dmyhm`	String indicating Day Month Year Hour and Minute in the format "d-m-Y H:M" e.g.: "01-05-2014 00:00" It represents the time of the first hour of emissions in Local Time
`tz`	Time zone as required in for function `as.POSIXct`
`utc`	interger indicating the difference between local and GMT time
`islist`	logical value to indicate if sdf is a list or not

Value

data-frame of gridded emissions g/h

Note

The reference of the emissions assimilation system is Vara-Vela, A., Andrade, M. F., Kumar, P., Ynoue, R. Y., and Munoz, A. G.: Impact of vehicular emissions on the formation of fine particles in the Sao Paulo Metropolitan Area: a numerical study with the WRF-Chem model, Atmos. Chem. Phys., 16, 777-797, doi:10.5194/acp-16-777-2016, 2016. A good website with timezones is http://www.timezoneconverter.com/cgi-bin/tzc

Examples

```
## Not run:
# Do not run

## End(Not run)
```

`Evaporative` *Construction function for class "Evaporative"*

Description

Returns a tranformed object with class "Evaporative" and units g/day. This class represents the daily emissions presented by Mellios G and Ntziachristos (2016) Gasoline evaporation, Tier 2. Eventually it will be incorporated the techniques of Tier 3.

Usage

```
Evaporative(x, ...)

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

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

## S3 method for class 'Evaporative'
plot(x, ...)
```

Arguments

`x`	Object with class "numeric"
`...`	ignored
`object`	Object with class "Evaporative"

Value

Objects of class "Evaporative" or "units"

Examples

```
## Not run:
ef1 <- ef_evap(ef = "erhotc",v = "PC", cc = "<=1400", dt = "0_15", ca = "no")
dfe <- emis_evap(rep(50,3),"PC","<=1400","G", 1:3,
                     10,4,2,1,
                     ef1*1:3, ef1*1:3, ef1*1:3, ef1*1:3,
                     ef1*1:3, ef1*1:3, ef1*1:3, ef1*1:3,
                     ef1*1:3, ef1*1:3, ef1*1:3, ef1*1:3)

## End(Not run)
```

`fe2015` *Emission factors from Environmental Agency of Sao Paulo CETESB*

Description

A dataset containing emission factors from CETESB and its equivalency with EURO

Usage

```
data(fe2015)
```

Format

A data frame with 288 rows and 12 variables:

Age Age of use

Year Year of emission factor

Pollutant Pollutants included: "CH4", "CO", "CO2", "HC", "N2O", "NMHC", "NOx", and "PM"

Proconve_LDV Proconve emission standard: "PP", "L1", "L2", "L3", "L4", "L5", "L6"

t_Euro_LDV Euro emission standard equivalence: "PRE_ECE", "I", "II", "III","IV", "V"

Euro_LDV Euro emission standard equivalence: "PRE_ECE", "I", "II", "III","IV", "V"

Proconve_HDV Proconve emission standard: "PP", "P1", "P2", "P3", "P4", "P5", "P7"

Euro_HDV Euro emission standard equivalence: "PRE", "I", "II", "III", "V"

Promot Promot emission standard: "PP", "M1", "M2", "M3"

Euro_moto Euro emission standard equivalence: "PRE", "I", "II", "III"

PC_G CETESB emission standard for Passenger Cars with Gasoline (g/km)

LT CETESB emission standard for Light Trucks with Diesel (g/km)

Source

http://veicular.cetesb.sp.gov.br/relatorios-e-publicacoes/

`fkm` *List of functions of mileage in km fro Brazilian fleet*

Description

Functions from CETESB: Antonio de Castro Bruni and Marcelo Pereira Bales. 2013. Curvas de intensidade de uso por tipo de veiculo automotor da frota da cidade de Sao Paulo This functions depends on the age of use of the vehicle

Usage

```
data(fkm)
```

Format

A data frame with 288 rows and 12 variables:

KM_PC_E25 Mileage in km of Passenger Cars using Gasoline with 25% Ethanol

KM_PC_E100 Mileage in km of Passenger Cars using Ethanol 100%

KM_PC_FLEX Mileage in km of Passenger Cars using Flex engines

KM_LCV_E25 Mileage in km of Light Commercial Vehicles using Gasoline with 25% Ethanol

KM_LCV_FLEX Mileage in km of Light Commercial Vehicles using Flex

KM_PC_B5 Mileage in km of Passenger Cars using Diesel with 5% biodiesel

KM_TRUCKS_B5 Mileage in km of Trucks using Diesel with 5% biodiesel

KM_BUS_B5 Mileage in km of Bus using Diesel with 5% biodiesel

KM_LCV_B5 Mileage in km of Light Commercial Vehicles using Diesel with 5% biodiesel

KM_SBUS_B5 Mileage in km of Small Bus using Diesel with 5% biodiesel

KM_ATRUCKS_B5 Mileage in km of Articulated Trucks using Diesel with 5% biodiesel

KM_MOTO_E25 Mileage in km of Motorcycles using Gasoline with 25% Ethanol

KM_LDV_GNV Mileage in km of Light Duty Vehicles using Natural Gas

Source

http://veicular.cetesb.sp.gov.br/relatorios-e-publicacoes/

hot_soak *Estimation of average running hot-soak evaporative emissions*

Description

Estimation of evaporative emissions from EMEP/EEA emisison guidelines

Usage

```
hot_soak(x, carb, p, eshotc, eswarmc, eshotfi)
```

Arguments

`x`	Mean number of trips per vehicle per day
`carb`	fraction of gasoline vehicles with carburator or fuel return system
`p`	Fraction of trips finished with hot engine
`eshotc`	average daily hot-soak evaporative factor for vehicles with carburator or fuel return system
`eswarmc`	average daily cold-warm-soak evaporative factor for vehicles with carburator or fuel return system
`eshotfi`	average daily hot-soak evaporative factor for vehicles with fuel injection and returnless fuel systems

Value

numeric vector of emission estimation in grams

References

Mellios G and Ntziachristos 2016. Gasoline evaporation. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2009

Examples

```
## Not run:
# Do not run
ev <- hot_soak(x = 1:10, carb = 0, p = 1, eshot = 1, eswarmc =1,
eshotfi = 1)

## End(Not run)
```

make_grid *Creates rectangular grid for emission allocation*

Description

It is created a SpatialGridDataFrame.The spatial reference is taken from the spatial object.

Usage

```
make_grid(spobj, width, height, polygon = T)
```

Arguments

spobj	A spatial object of class sp
width	Width of grid cell, units according sr
height	Height of grid cell, units according sr
polygon	whe TRUE return a polygon grid, when FALSE a SpatialGridDataFrame

Value

grid

Examples

```
## Not run:
#do not run
data(net)
grid <- make_grid(net, width = 0.5/102.47, height = 0.5/102.47) #500 mts
spplot(net, scales=list(draw=T),
sp.layout = list("sp.polygons", grid, pch = 16, cex = 2, col = "black"))

## End(Not run)
```

my_age *Returns amount of vehicles at each age*

Description

Returns amount of vehicles at each age using a numeric vector

Usage

```
my_age(x, y, name, k = 1)
```

Arguments

x	numerical vector of vehicles
y	Age dustribution of vehicles
name	of vehicle assigned to columns of dataframe
k	multiplication factor

Value

dataframe of age distrubution of vehicles

Examples

```
## Not run:
# Do not run
pc <- rnorm(100, 300, 10)
dpc <- rnorm(10, 100, 1)
PC_E25_1400 <- my_age(x = pc,y = dpc, name = "PC_E25_1400")

## End(Not run)
```

---

net *Road network of the west part of Sao Paulo city*

Description

This dataset is a SpatialLineDataFrame of sp package with roads from a traffic simulations made by CET Sao Paulo, Brazil

Usage

```
data(net)
```

Format

A data frame with 1796 rows and 1 variables:

ldv Light Duty Vehicles (1/h)

hdv Heavy Duty Vehicles (1/h)

lkm Length of the link (km)

ps Peak Speed (km/h)

ffs Free Flow Speed (km/h)

tstreet Type of street

lanes Number of lanes per link

capacity Capacity of vehicles in each link (1/h)

tmin Time for travelling each link (min)

Source

http://www.cetsp.com.br/

`netspeed` *Calculate speeds of traffic network*

Description

Creates a dataframe of speeds fir diferent hours and each link based on morning rush traffic data

Usage

```
netspeed(q, ps, ffs, cap, lkm, alpha = 0.15, beta = 4, isList = FALSE,
  distance = "km", time = "h")
```

Arguments

`q`	Data-frame of traffic flow to each hour (veh/h)
`ps`	Peak speed (km/h)
`ffs`	Free flow speed (km/h)
`cap`	Capacity of link (veh/h)
`lkm`	Distance of link (km)
`alpha`	Parameter of BPR curves
`beta`	Parameter of BPR curves
`isList`	Logical to specify type of return, list or data-frame
`distance`	Character specifying the units for distance. Default is "km"
`time`	Character specifying the units for time Default is "h"

Value

dataframe or list of speeds with units

Examples

```
## Not run:
# Do not run
data(net)
data(pc_profile)
pc_week <- temp_fact(net$ldv+net$hdv, pc_profile)
df <- netspeed(pc_week, net$ps, net$ffs, net$capacity, net$lkm)
class(df)
plot(df) #plot of the average speed at each hour, +- sd

## End(Not run)
```

pc_cold *Profile of Vehicle start patterns*

Description

This dataset is a dataframe with percetage of hourly starts with a lapse of 6 hours with engine turned off. Data source is: Lents J., Davis N., Nikkila N., Osses M. 2004. Sao Paulo vehicle activity study. ISSRC. www.issrc.org

Usage

```
data(pc_cold)
```

Format

A data frame with 24 rows and 1 variables:

V1 24 hours profile vehicle starts for Monday

pc_profile *Profile of traffic data 24 hours 7 n days of the week*

Description

This dataset is a dataframe with traffic activity normalized monday 08:00-09:00. This data is normalized at 08:00-09:00. It comes from data of toll stations near Sao Paulo City. The source is ARTESP (www.artesp.com.br)

Usage

```
data(pc_profile)
```

Format

A data frame with 24 rows and 7 variables:

V1 24 hours profile for Monday

V2 24 hours profile for Tuesday

V3 24 hours profile for Wednesday

V4 24 hours profile for Thursday

V5 24 hours profile for Friday

V6 24 hours profile for Saturday

V7 24 hours profile for Sunday

`running_losses` *Estimation of average running losses evaporative emissions*

Description

Estimation of evaporative emissions from EMEP/EEA emisison guidelines

Usage

```
running_losses(x, carb, p, erhotc, erwarmc, erhotfi)
```

Arguments

`x`	Mean number of trips per vehicle per day
`carb`	fraction of gasoline vehicles with carburator or fuel return system
`p`	Fraction of trips finished with hot engine
`erhotc`	average daily running losses evaporative factor for vehicles with carburator or fuel return system
`erwarmc`	average daily cold and warm running losses evaporative factor for vehicles with carburator or fuel return system
`erhotfi`	average daily hot running losses evaporative factor for vehicles with fuel injection and returnless fuel systems

Value

numeric vector of emission estimation in grams

References

Mellios G and Ntziachristos 2016. Gasoline evaporation. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2009

Examples

```
## Not run:
# Do not run
ev <- running_losses(x = 1:10, carb = 0, p = 1, erhot = 1, erwarmc =1,
erhotfi = 1)

## End(Not run)
```

`speciate` *Speciation of emissions*

Description

The separation of emissions in different compunds. It includes black carbon and organic matter from particulate matter. Soon it will be added more speciations

Usage

```
speciate(x, spec = "bcom", veh, fuel, eu, show = FALSE, list = FALSE)
```

Arguments

`x`	Emissions estimation
`spec`	type of speciation, e.g.: "bcom" stands for black carbon and organic matter. The speciations are: "bcom", tyre", "break", "road","iag" and "nox".
`veh`	Type of vehicle. When spec is "bcom" or "nox" veh can be "PC", "LCV", HDV" or "Motorcycle". When spec is "iag" veh is only "veh". Not required for "tyre", "break" or "road"
`fuel`	Fuel. When spec is "bcom" fuel can be "G" or "D". When spec is "iag" fuel can be "G", "E" or "D". When spec is "nox" fuel can be "G", "D", "LPG", "E85" or "CNG". Not required for "tyre", "break" or "road"
`eu`	Euro emission standard: "PRE", "ECE_1501", "ECE_1502", "ECE_1503", "I", "II", "III", "IV", "V", "III-CDFP","IV-CDFP","V-CDFP", "III-ADFP", "IV-ADFP","V-ADFP" and "OPEN_LOOP". When spec is "iag" accept the values "Exhaust" "Evaporative" and "Liquid". When spec is "nox" eu can be "PRE", "I", "II", "III", "IV", "V", "VI", "VIc", "III-DPF" or "III+CRT". Not required for "tyre", "break" or "road"
`show`	when TRUE shows row of table with respective speciation
`list`	when TRUE returns a list with number of elements of the list as the number species of pollutants

Value

dataframe of speciation in grams

Note

when spec = "iag", veh is only "VEH", STANDARD is "Evaporative", "Liquid" or "Exhaust", FUEL is "G" for gasoline (blended with 25% ethanol), "E" for Ethanol and "D" for diesel (blended with 5% of biodiesel). When spec = "bcom", veh can be "PC", "LCV", "Motorcycle" or "HDV" VEH", STANDARD is "Evaporative", "Liquid" or "Exhaust", FUEL is "G" for gasoline (blended with 25% ethanol), "E" for Ethanol and "D" for diesel (blended with 5% of biodiesel).

References

"bcom": Ntziachristos and Zamaras. 2016. Passneger cars, light commercial trucks, heavy-duty vehicles including buses and motor cycles. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2016

"tyre", "break" and "road": Ntziachristos and Boulter 2016. Automobile tyre and break wear and road abrasion. In: EEA, EMEP. EEA air pollutant emission inventory guidebook-2009. European Environment Agency, Copenhagen, 2016

"iag": RAFEE, S.A.A. Estudo numerico do impacto das emissoes veiculares e fixas da cidade de Manaus nas concentracoes de poluentes atmosfericos da regiao amazonica. 2015. 109 f. Dissertacao (Mestrado). Programa de Pos-Graduacao em Engenharia Ambiental (PPGEA) - Universidade Tecnologica Federal do Parana. Londrina, 2015. http://repositorio.utfpr.edu.br/jspui/bitstream/1/1675/1/LD_PPGEA_M_Ra

"iag": Vela, A. L. V. Avaliacao do impacto da mudanca dos fatores de emissao veicular na formacao de ozonio troposferico na Regiao Metropolitana de Sao Paulo. 2013. Dissertacao de Mestrado. Instituto de Astronomia, Geofisica e Ciencias Atmosfericas, Universidade de Sao Paulo, Sao Paulo. http://www.iag.usp.br/pos/sites/default/files/d_angel_l_v_vela_corrigida_0.pdf

Examples

```
## Not run:
# Do not run
pm <- rnorm(n = 100, mean = 400, sd = 2)
df <- speciate(pm, veh="PC", fuel="G", eu="I")

## End(Not run)
```

Speed *Construction function for class "Speed"*

Description

Returns a tranformed object with class "Speed" and units km/h. This functions includes two arguments, distance and time. Therefore, it is posibel to change the units of the speed to "m" to "s" for example. This function returns a dataframe with units for speed. When this function is applied to numeric vectors it add class "units".

Usage

```
Speed(x, ...)

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

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

## S3 method for class 'Speed'
plot(x, ...)
```

Arguments

`x`	Object with class "data.frame", "matrix" or "numeric"
`...`	ignored
`object`	Object with class "Speed"

Value

Constructor for class "Speed" or "units"

See Also

units

Examples

```
## Not run:
data(net)
data(pc_profile)
speed <- Speed(net$ps)
class(speed)
plot(speed, type = "l")
pc_week <- temp_fact(net$ldv+net$hdv, pc_profile)
df <- netspeed(pc_week, net$ps, net$ffs, net$capacity, net$lkm)
summary(df)

## End(Not run)
```

`temp_fact` *Expansion of hourly traffic data*

Description

Matrix multiplication between traffic and hourly expansion data-frames to obtain a data-frame of traffic at each link to every hour

Usage

```
temp_fact(q, pro)
```

Arguments

`q`	traffic data per each link
`pro`	expansion factors data-frames

Value

data-frames of expanded traffic

Examples

```
## Not run:
# Do not run
data(net)
data(pc_profile)
pc_week <- temp_fact(net$ldv+net$hdv, pc_profile)
plot(pc_week)

## End(Not run)
```

Vehicles *Construction function for class "Vehicles"*

Description

Returns a tranformed object with class "Vehicles" and units 1/h. The type of objects supported are of classes "matrix", "data.frame", "numeric" and "array". If the object is a matrix it is converted to data.frame. If the object is "numeric" it is converted to class "units". The function `emis_paved` needs veh to be an array, therefore in this case, veh must be an array in the total fleet at each street and dimensions total fleet, hours and days

Usage

```
Vehicles(x, ...)

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

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

## S3 method for class 'Vehicles'
plot(x, ...)
```

Arguments

`x`	Object with class "Vehicles"
`...`	ignored
`object`	Object with class "Vehicles"

Value

Objects of class "Vehicles" or "units"

Examples

```
## Not run:
lt <- rnorm(100, 300, 10)
class(lt)
vlt <- Vehicles(lt)
class(vlt)
plot(vlt)
LT_B5 <- age_hdv(x = lt,name = "LT_B5")
print(LT_B5)
summary(LT_B5)
plot(LT_B5)

## End(Not run)
```

---

vkm *Estimation of VKM*

Description

VKM consists in the product of the number of vehicles and the distance driven by these vehicles in km. This function reads hourly vehciles and then extrapolates the vehicles

Usage

```
vkm(veh, lkm, profile, hour = 1, day = 1, array = F)
```

Arguments

`veh`	Numeric vector with number of vehicles per street
`lkm`	Length of each link (km)
`profile`	Numerical or dataframe with nrows equal to 24 and ncol 7 day of the week
`hour`	Number of considered hours in estimation
`day`	Number of considered days in estimation
`array`	When FALSE produces a dataframe of the estimation. When TRUE expects a profile as a dataframe producing an array with dimensions (streets x hours x days)

Value

emission estimation of vkm

Examples

```
## Not run:
# Do not run
pc <- lkm <- abs(rnorm(10,1,1))*100
pro <- matrix(abs(rnorm(24*7,0.5,1)), ncol=7, nrow=24)
vkms  <- vkm(veh = pc, lkm = lkm, profile = pro, hour = 24, day = 7, array = T)

## End(Not run)
```

Index