

Sergio Ibarra-Espinosa

Vehicular emissions inventory with VEIN



Contents

List of Tables	vii
List of Figures	ix
Preface	xi
1 Introduction	1
1.1 Definitions and sources of information	1
1.1.1 Approaches	2
1.2 Installation	2
1.3 Required R packages and dependencies	3
1.4 Data inside the VEIN model	4
1.4.1 Emission factors from CETESB	5
1.4.2 Mileage functions of Brazilian Fleet	5
1.4.3 Temporal factors for Passenger Cars	7
1.4.4 Profile for cold starts of Passenger Cars	8
1.5 Acknowledgements	9
2 Basics of R	11
2.1 Brief history	11
2.1.1 Installation	11
2.2 Using R	12
2.2.1 R objects	12
2.2.2 Factors	14
2.2.3 Missing Values	14
2.2.4 Matrices	15
2.2.5 Arrays	16
2.2.6 Lists	17
2.2.7 Data-Frames	18
2.2.8 Names	19

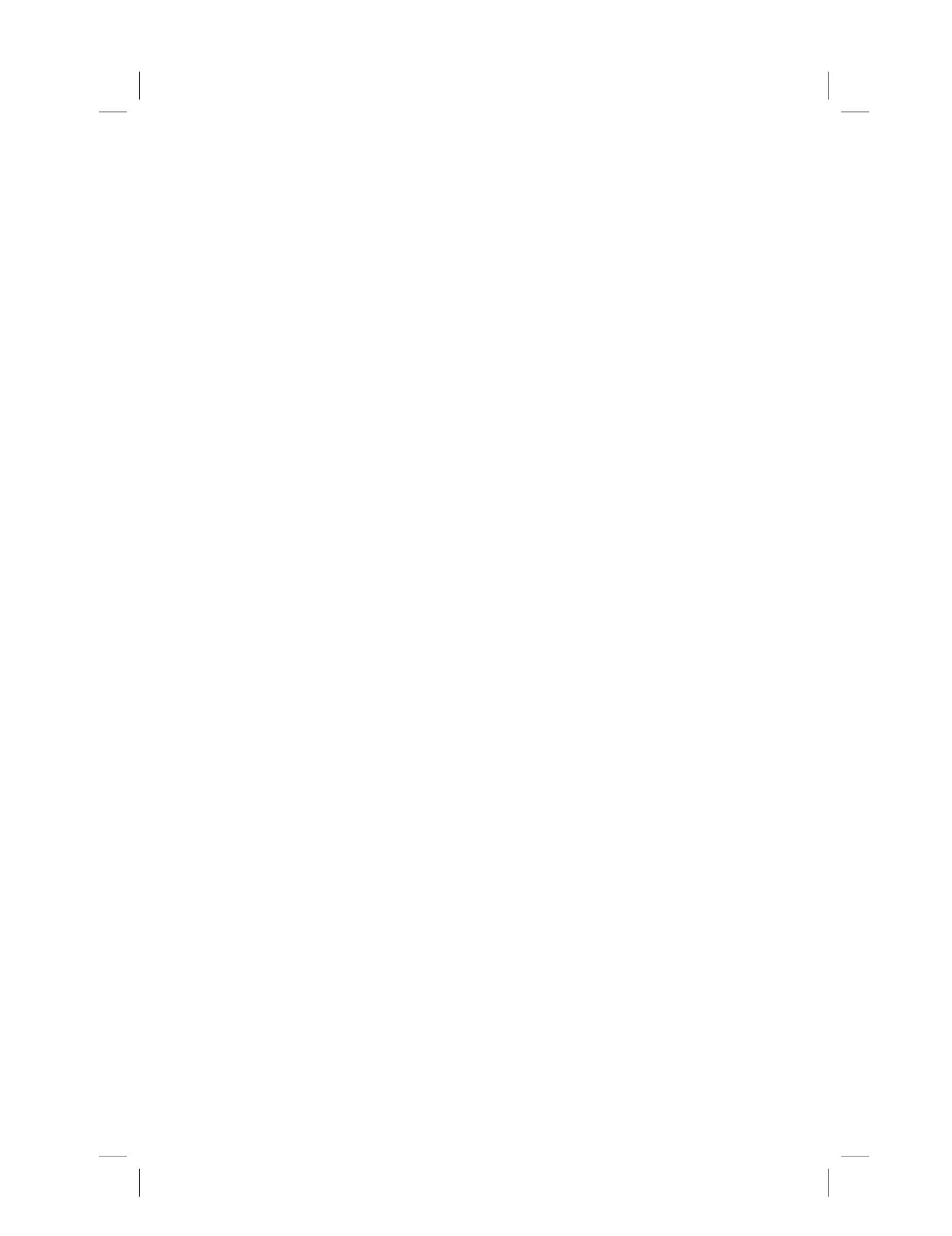
2.3	Inputting data into R	19
3	Structuring an Emissions Inventory	21
3.1	Overview of emissions inventorying using VEIN	21
3.2	The function	21
4	Traffic data	27
4.1	Sources of traffic data	28
4.1.1	Travel demand outputs	28
4.1.2	Inteprolation of traffic counts	29
4.1.3	Generating traffic flows from Origin-Destination-Surveys (ODS)	31
4.1.3.1	Google Distance Matrix API	33
4.1.3.2	pgRouting for postGIS	34
4.1.3.3	The R package googleway	34
4.1.3.4	The R package dodgr	36
4.1.4	Top-down approach	38
4.2	Main functions	39
4.2.1	Expanding traffic data with the function	40
4.2.2	Calculating speed at other hours with the function	41
4.2.3	Distribution of vehicles by age of use with the functions , and	42
4.2.4	The function	43
4.2.5	The class	43
4.2.6	Other traffic functions	44
4.3	Vehicular composition	45
4.4	Application	45
5	Emission Factors	47
5.1	Speed functions and	49
5.1.1	Emission factors of LDV depending on speed with the function	50
5.1.2	Emission factors of HDV depending on speed with the function	53
5.2	Local emission factors by age of use. The functions and	55
5.3	Incorporating speed variation with the functions and	58
5.4	Cold Starts	60

5.5	Deterioration	62
5.6	Evaporative emissions	63
5.7	Emission factors of wear	69
5.8	Emission factors from tunnel studies	70
5.9	Emission factors from International Vehicle Emissions (IVE)	71
6	Estimation of emissions	73
6.1	The function	74
6.2	The function	75
6.3	The function	76
6.4	The function	78
6.5	The function	80
7	Post estimation with emis_post	83
7.1	Emissions by street	85
7.2	Total emissions in data-frames	88
7.3	Merging emissions	91
7.4	Creating grids	94
7.5	Gridding emissions with	96
8	Speciation	101
8.1	Speed functions of VOC species	102
8.2	Speciate function	106
8.3	Black carbon and organic matter	107
8.4	Tyre wear, breaks wear and road abrasion	111
8.5	NO and NO_2	114
8.6	Volatile organic compounds:	116
8.7	The speciation	118
8.8	The speciation	123
9	Inputs for atmospheric models	125
9.1	WRFChem input with wrfinput and	126
9.1.1	Creating a WRF-Chem input file	126
9.1.1.1	o) Network	126
9.1.1.2	1) Vehicular composition	127
9.1.1.3	2) Emission factors	128
9.1.1.4	3) Estimation of emissions	128

9.1.1.5	4) Post-estimations	128
10	Quality check and errors	137
10.1	Guidance from EMEP/EEA air pollutant emission inventory guidebook	137
10.1.1	Key categories	137
10.1.2	Uncertainty	138
10.1.2.1	Default uncertainty ranges	139
10.1.3	Quality Assurance and Quality Check	141
10.1.4	Inventory management cycle	142
10.2	Avoiding errors with VEIN	143
10.2.1	Traffic flow	143
10.2.2	Vehicular composition	145
10.2.3	Units	146
10.2.4	Emission factors	146
10.2.5	Deterioration	147
10.2.6	Fuel evaluation	147
10.2.7	Emissions estimation	147
Appendix		147
Appendix A: Fast Example of VEIN		149
Appendix B: Detailed Example of VEIN		151
.1	Network	151
.2	Vehicular composition (CETESB, 2015)	151
.3	Traffic data	153
Appendix B		161

List ofTables

1.1	Emission factors from CETESB in VEIN model	5
4.1	Matrix OD for motorized individual trips between 06:30 and 08:30 MASP 2007	32
4.2	speeds for scheme = T	42
5.1	Most used combinations for emission factors for LDV . .	51
5.2	Emission factors from tunnels in São Paulo 2014	70
8.1	Black Carbon and Organic Matter [g/168h]	110
8.2	Ratio between break and tyre wear emissions	113
8.3	NO ₂ and NO speciation (t/y)	115
8.4	Speciation of PM _{2.5}	123
10.1	Precision indicators (Ntziachristos and Samaras 2016) .	140
.2	Vehicular composition of PC for applied in this book . .	154
.3	Vehicular composition of LCV for applied in this book .	155
.4	Vehicular composition of HGV for applied in this book .	156
.5	Vehicular composition of BUS for applied in this book .	157
.6	Vehicular composition of MC for applied in this book .	158



List of Figures

1.1	Mileage of PC using E25	6
1.2	Temporal Factors for PC	7
1.3	Cold-starts profile for PC	8
3.1	Structuring an emissions inventories with VEIN	22
4.1	LDV at 08:00-09:00 (LT) in west São Paulo	30
4.2	Zones OD for MASP 2007	32
4.3	Driving route between zones 'Luz' and 'Bom Retiro' in São Paulo	36
4.4	Daily trips of Light Duty Vehicles in São Paulo	37
4.5	Emissions of PM due traffic, 2017	39
4.6	Structuring an emissions inventories with VEIN	40
5.1	Structuring an emissions inventories with VEIN	49
5.2	Emission factor as a speed function	52
5.3	Emission factor of PC and LT by age of use	57
5.4	Emission factor of PC and LT	58
5.5	Cold Emission factor of PC CO (g/km)	62
5.6	Deteriorated (red) and not-deteriorated (blue) EF CO	64
5.7	Running losses evaporative emission actors (g/trip)	67
5.8	Hot Soak (blue), Diurnal (green) and Running Losses (red) EF (g/km)	69
6.1	Emissions estimation process with VEIN	74
7.1	Post-emissions	84
7.2	CO emissions of PC (g/h)	87
7.3	NOx emissions of LT (g/h)	88
7.4	NOx emissions of LT by hour	90
7.5	NOx emissions of LT by age	90

7.6	NOx emissions of LT	91
7.7	Grid with spacing of 500 mts	95
7.8	Spatial grid of HDV traffic 500 mts	98
7.9	NOx emissions for 00:00 [g/h]	99
8.1	Black Carbon [g/168h]	109
8.2	Organic Matter [g/168h]	110
8.3	Break and Tyre emissions (t/y)	113
8.4	http://www.lapat.iag.usp.br/aerossol/wrf9/index.php . .	119
8.5	Speciation for NMHC by type of fuel and process mol/100g (Ibarra-Espinosa, S., 2017)	121
9.1	Generation of wrfchem inputs using GriddedEmission- sArray	127
9.2	Gridded emissions (mol/h)	132
9.3	GriddedEmissionsArray (mol/h)	133
9.4	Wrfchem input file (NetCDF) (mol/h)	134

Preface

This book is about the process for estimating vehicular emissions. This process is complex and *can* be difficult. It is required a big amount of information related to traffic, emissions factors and then process the output for the desired purpose. In this book this process is addressed with the **VEIN** (Ibarra-Espinosa et al., 2018c) model, which is an R package available at [{}](#).

Purpose

I wrote this book to provide instructions to all possible users who want to estimate vehicular emissions using VEIN. VEIN provides several function which are related to each other reading different traffic sources and emission factors for different pollutants. However, incorporating all the functions with input/output (I/O) process can be difficult. Moreover, communicating the recommended practices and instructions to a increasingly broader audience can be more difficult. Therefore, the purpose of this book is to reach the maximum amount of interested people in a simple way. The language chosen is English because it is the global language.

Structure

Chapter 1 covers the introduction to this book including the installation of R packages and dependent libraries to several operative systems. Chapter 2 covers basics commands using R. Chapter 3 presents the function *inventory* to produce an structure of directories and scripts for running

vein. Chapter 4 covers the required traffic data for inputting into VEIN including different types and sources of information. Chapter 5 includes the emission factors that are included into the package and also, examples for inputting and creating new emission factors. Chapter 6 presents the estimations functions and tips. Chapter 7 includes the functions to do post-estimations to generate emission data-bases and also emissions at each street. Chapter 8 and 9 provides applications into air quality modeling, some studies with to investigate health effects and lastly, the use of VEIN as a tool for environmental planning.

About the author

Sergio Ibarra Espinosa is a Chilean Environmental and Loss Prevention Engineer from the Technological Metropolitan University (UTEM) where he studied the health effects of air pollution using R in year 2007. Then started working at the National Center for Environment (CENMA) in Chile focused on emissions from vehicles, airports, biomass burning and mining industry. Then obtained a MSc. in Environmental Planning and Management from the University of Chile, with a scholarship from CENMA. Then obtained a PhD. in Meteorology at the Institute of Astronomy, Geophysics and Atmospheric Sciences (IAG) from the University of São Paulo (USP) with scholarship from CAPES during the first year and Becas Chile for the last three years. During his PhD. he learned programming with R and developed the R package VEIN.

1

Introduction

1.1 Definitions and sources of information

Developing emissions inventories is a complex task. Therefore, before entering into the details of the functions, it is good to provide some initial definitions.

A good starting point is the book “*The Art of Emissions Inventorying*” (Pulles and Heslinga, 2010). This book provides general description of what is an emissions inventory. One of the early sources of information about emissions inventories and emissions factors is the Environmental Protection Agency (U.S.EPA) and its compilation of emission factors AP42 (). A third source of information are the European EMEP/EEA air pollutant emission inventory guidebook () .

An emissions inventory is the compilation of all mass emitted into the atmosphere by activities in a defined area during a lapse of time. The Eq. ((1.1) shows the general form (McGlade and Vidic, 2009):

$$Emission_{pollutant} = \sum_{activities} Activityrate \cdot Emission\ factor_{activity,pollutant} \quad (1.1)$$

There are two main types of inventories based on the application, one type is for policy application and the other for scientific application (Pulles and Heslinga, 2010).

- Inventories for policy application include the National Greenhouse Gas Inventory for the parties under the United Nations Framework on Climate Change (UNFCCC, 2017).
- Emissions inventories with scientific applications include the Emis-

sion Database for Global Atmospheric Research EDGAR (EJ-JRC/PBL, 2016).

Emissions inventories can be also multimedia inventories, such as the Pollutant Release and Transfer Registers (PRTR) which include pollutants released to air, water and soil and transferred off-site for treatment or disposal (). However, the type of inventory covered in this book cover only the emissions released into the atmosphere.

1.1.1 Approaches

It is necessary show some definitions of vehicular emissions inventories approaches, which in this case comes from the European Emissions Guidelines (Ntziachristos and Samaras, 2016):

- **Top-down** inventories are using input activity traffic data as fleet statistics, fuel consumption, representative speeds and country balances. The emissions factors are based on representative speeds. These inventories are also know as *static*.
- **Bottom-up** inventories are using input activity traffic data from traffic counts, traffic simulations, vehicle composition, speed recordings and length of roads. The emission factors are speed or/and acceleration functions. These inventories are also know as *dynamic*.
- **Reconciliation** Both approaches can be reconciled in urban emissions inventories with comparison of total mileage, cold start mileage or emission factors to ensure that the comparison with the total fuel consumption in the study area is satisfactory. This means that bottom-up and top-down vehicular emissions must match the fuel sales over a region of study.

1.2 Installation

The VEIN r-package can be installed from Comprehensive R Archive Network CRAN¹ directly:

```
install.packages
```

VEIN can be also installed from github:

```
library  
install_github
```

The github installation requires that the *devtools* package (Wickham and Chang, 2017) must be installed, including its dependencies.

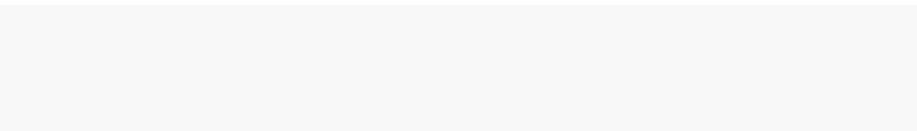
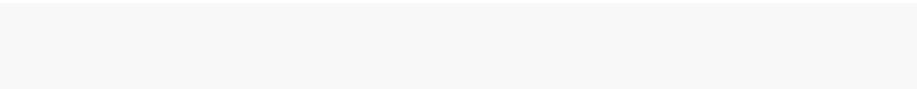
1.3 Required R packages and dependencies

VEIN imports functions from other R packages. Hence, these packages are necessary::

sf is a package that provides simple features access for R (Pebesma, 2017). It depends on the library GEOS () , ‘Geospatial’ Data Abstraction Library (‘GDAL’) () and PROJ (). **data.table** is a package that provides simple features access for R (Pebesma, 2017). It can be installed. - **sp**. This is a package that provides classes and methods for spatial data (Pebesma and Bivand, 2005) considering points, lines, polygons and grids. few function of sp are imported, most of them comes from sf. **eixport** is a package (Ibarra-Espinosa et al., 2017b) that creates inputs for atmospheric models (Ibarra-Espinosa et al., 2018b). - **units**. It is a package for measurement units in R (Pebesma et al., 2016). This package depends on the library udunits2 (). I

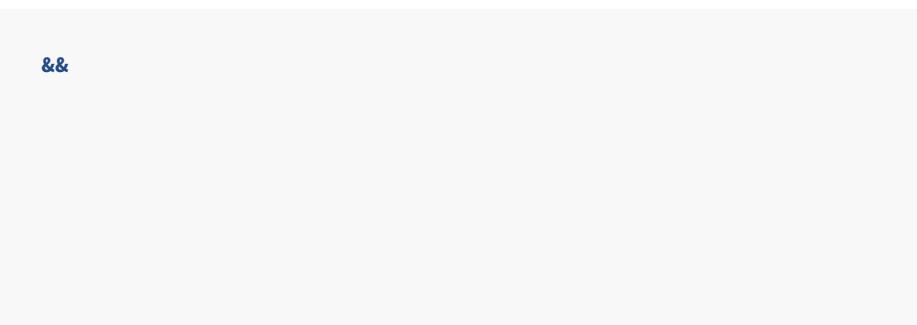
install.packagesc

In order to install **sf** package it is necessary to install the dependencies into each operational system.

Ubuntu**Fedora****Debian and other**

Dockerfiles from rocker geospatial².

Windows Via Rtools³.

MAC OS

The R package eixport imports function from **ncdf4** (Pierce, 2017), therefore the library NetCDF must be installed. For Ubuntu, do:

²

³

TABLE 1.1: Emission factors from CETESB in VEIN model

Age	Year	Pollutant	PC_G	LT
1	2015	CO	0.171	0.027
2	2014	CO	0.216	0.012
3	2013	CO	0.237	0.012
4	2012	CO	0.273	0.005
5	2011	CO	0.275	0.379
6	2010	CO	0.204	0.416

1.4 Data inside the VEIN model

There are several datasets available inside the model:

1.4.1 Emission factors from CETESB

- **fe2015:** Emission factors from the Environmental Agency of Sao Paulo (CETESB). Pollutants included: “CH4”, “CO”, “CO2”, “HC”, “N2O”, “NMHC”, “NOx” and “PM”. The type of vehicles included are Passenger Cars (PC).

usage:

```
library  
data
```

1.4.2 Mileage functions of Brazilian Fleet

- **fkm:** A list of functions of mileage in km for the Brazilian fleet. It includes mileage functions based on more than 2 million recordings of vehicles (Bruni and Bales, 2013). It consists a list of age functions. The type of vehicles are Passneger Cars (PC), Light Commercial Vehicles (LCV), Small Busses (SBUS), Trucks, Articulated Trucks (ATRUCKS), Motorcycles (MOTO) and Light vehicles (LDV). The fuels are Gasoline using 25% of ethanol (E25), Ethanol 100% (E100) and Diesel with 5% of biodiesel (B5). There are also vehicles with flex engines which can run either with E25, E100 or any mixture in between (Giroldo et al., 2005). The Fig. (1.1 shows the mileage of PC using fuel E25.

usage:

```
library  
data  
names
```

```
:  
:  
par c  
plot
```

- **net:** Road network of the west part of Sao Paulo city. It consistses in a SpatialLinesDataFrame (class of sp) with the attributes ldv (light duty vehicles, $1 \cdot h^{-1}$), hdv (heavy duty vehicles, $1 \cdot h^{-1}$), ps (peak speed, $km \cdot h^{-1}$), ffs (free flow speed, $km \cdot h^{-1}$), tstreet (type of street), lanes (number of lanes), capacity (capacity fo vehicles at each link, 1/h) and tmin (time for travelling each link, min). The Fig. (4 shows more details.

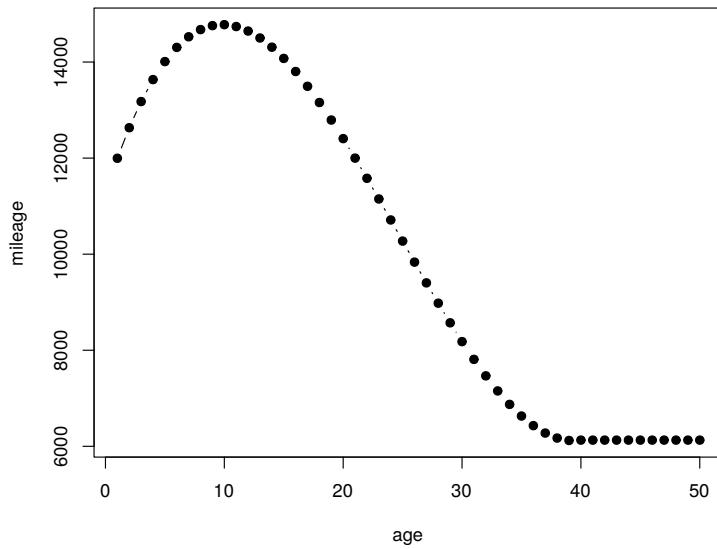


FIGURE 1.1: Mileage of PC using E25

1.4.3 Temporal factors for Passenger Cars

Temporal factors (TF) are matrix of hourly traffic data normalized for the hour of interest, normally the morning rush hour 08:00-09:00 in local time (LT) (Ibarra-Espinosa et al., 2018c). This dataset covers 168 hours of one week and it is based on traffic counts of toll stations near the center of the city of São Paulo, Brazil. The Fig. (1.2 shows the temporal factors for PC.

usage:

```
library  
data  
unlist  
:  
par c  
plot
```

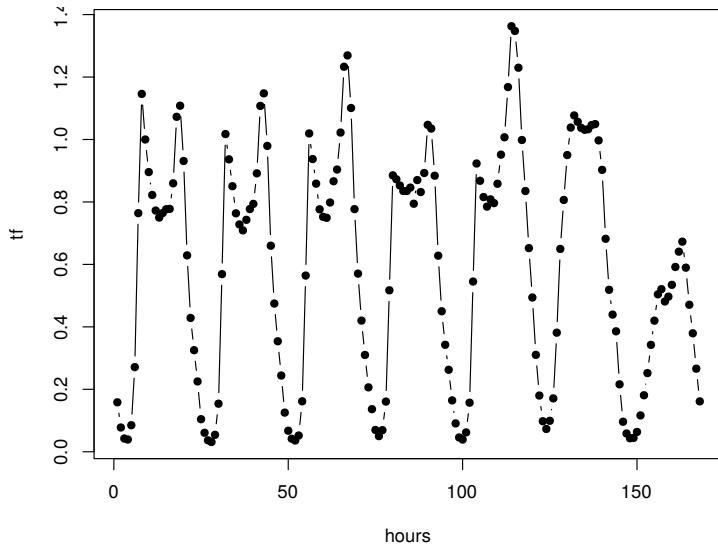


FIGURE 1.2: Temporal Factors for PC

1.4.4 Profile for cold starts of Passenger Cars

This is a profile with the hourly percentage of cold starts of Passenger Cars. This data covers 24 hours of a working day and it is based cold start recordings during the implementation of the International Vehicle Emissions (IVE) model (Davis et al., 2005) in São Paulo (Lents et al., 2004). The Fig. (1.3 shows the cold-start profile.

usage:

```
library
data
  unlist
  :
par c
plot
```

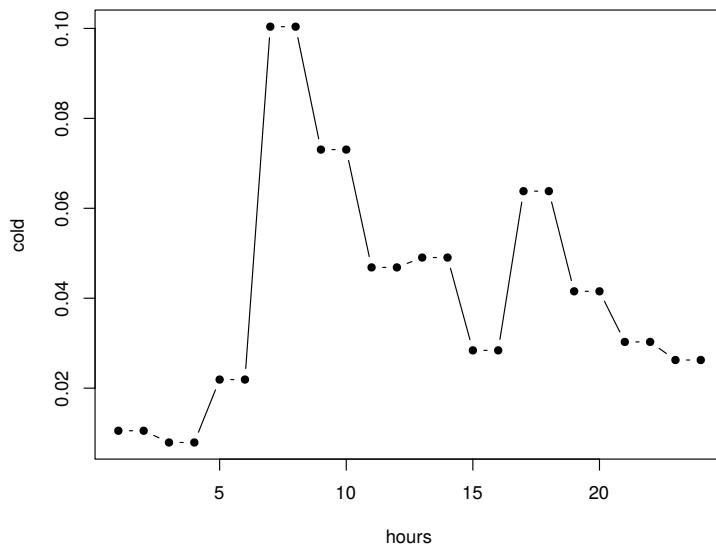


FIGURE 1.3: Cold-starts profile for PC

1.5 Acknowledgements

Martin Ramacher, PhD Student Chemistry Transport Modelling, Institute for Coastal Research, Helmholtz-Zentrum Geesthacht, Germany - Proofreading. Leila Droprinchinski Martins, Professor (Associate) Federal Technological University of Parana, Brazil - Comments.



2

Basics of R

2.1 Brief history

R is a programming language developed by Ross Ihaka and Robert Gentleman in 1993 (Ihaka and Gentleman, 1996). A good resource about R is the book “Programming for data science” (Peng, 2015). R can be explained as a free version of S-PLUS programming language (). It is free and recognized as GNU software () with GNU General Public License (GPLV2-GPLV3) license. This feature allowed that many developers started improving and adding code over the years.

R includes several statistical functions, therefore, users who just want to use the base capabilities of R are not forced to learn R-programming. However, the evolution from user to developer in R has been facilitated with numerous publications such as the book “R packages: organize, test, document, and share your code” (Wickham, 2015), or “The art of R programming: A tour of statistical software design” (Matloff, 2011).

2.1.1 Installation

R can be installed in different OS including Linux, Windows, Mac and Solaris. The webpage shows how to install R in any platform. A popular Integrated development environment (IDE) is **RStudio**, which contains many useful integrated options. However, R can be run on the terminal and any any text editor can be used to write scripts in R.

2.2 Using R

If we type at the R prompt

```
print
```

The integer `1` was *assigned* (`<-`) to `x`, then writing `x` or `print(x)` will show the value of `x`. Instead of numbers, other expressions such as strings, dates or other objects can be assigned. R includes several statistical functions. For instance, if we type ‘`pi`’ at the terminal, it will print the pi number.

2.2.1 R objects

There are five basic classes (or atomic classes).

- character.
- numeric.
- integer.
- complex.
- logical (TRUE/FALSE or just T or F).

Objects of the same class can be grouped in **vectors**. Vectors are created with writing `c()` with the elements of the vector inside the parenthesis, separated by colon. In fact, `c` is a built-in function to create vectors. In this case, the vector `v1` contains a sequence of three integers, from `1` to `3`. The resulting class is numeric.

```
:  
c  
identical
```

```
class
```

With the operator `[` we can get specific elements of your vectors. In the following code the function `length` is used additionally, which simply returns the length of an object.

```
length
```

If we create a vector with numbers and letter, the resulting vector will be “character”, automatically converting the numbers into characters.

```
c:
```

```
class
```

Objects can be converted to other classes with the `as.*` functions. For instance, if we convert the vector `v2` to numeric, it will recognize the numbers adding an NA into the position of the character.

```
as.numeric
```

2.2.2 Factors

Factors represents categorical data. A typical example is a character vector of days of the week. The function `factor` creates factors. To see the help section, type `?factor`. The following example will use this function with the arguments `levels` and `labels`.

```
factor c  
c
```

2.2.3 Missing Values

Missing values in data-sets are a typical source of headache for R users. Fortunately, R counts with several tools to avoid these headaches. These tools are the functions `is.na` to check for NA (not available) and `is.nan` (not a number). This function returns logical values.

```
c  
is.na
```

```
c  
is.na
```

```
is.nan
```

2.2.4 Matrices

Matrices are structure with rows and columns. They can be created using the function and also, using vectors. Remember, if you want to know more about any function you just have to type `?` and the function (e.g.) and the help documentation for the function will be opened.

Let's create a matrix using vectors:

```
:
```



```
matrix
```

We can check the dimensions of our matrix with , which are 3 and 4.

```
dim
```

In order to get the elements of matrix we can use the operator:

2.2.5 Arrays

Arrays are like matrices inside other matrices. In fact, a matrix is a 2-dimensional array. Let's create an array create an array using the same vector and same dimensions of , 3 and 4. has three arguments, , and . In the argument let's add the number 2, resulting in an array of two matrices identical to .

```
array      c
```

We can subset elements from array also with the operator.

```
dim
```

2.2.6 Lists

List are objects that can contain elements of different classes. I like to call lists as bags. In a bag you can put almost anything. Also, inside the bag, you can put other bags with different things. This means that you can have a list of lists of any R object. Lists can be created with the `list` function. Let's create an empty list. Then we will use the `vector` to create another list.

```
:  
list  
list  
length
```

```
as.list  
length
```

As mentioned, the list can have lists inside.

```
listlist
```

```
list  
length
```

2.2.7 Data-Frames

“Data-frames are used to store tabular data in R” (Peng, 2015). You can think of data-frames as spreadsheet-like objects. They are similar to matrices and you can have a matrix and a data-frame with the same dimensions. Basically, you have rows and columns, columns can have different classes, and the columns usually have a one-word name.

As this type of object is very used by scientists, there are R packages created to work with this type of objects. Below some of them:

- : This is not a package. It is a class and function from the library in R. With this function you can create data-frames.
- (Dowle and Srinivasan, 2017): objects has the class which inherits from the class , which means that they share common characteristics. However, the big difference with is the efficiency with memory. It includes functions such as for reading millions of observations in seconds.
- (Wickham and Henry, 2018): Sometimes your data is in long format and you need it in wide format, and viceversa. This package does the job.
- (Wickham and Bryan, 2017): Good package for importing Excel and LibreOffice files. I recommend this packages for newbies.
- (Pebesma, 2017): This package presents the class and introduces the list-column of spatial geometry.

Let's create a data-frame.

```
:  
:  
data.frame
```

```
class
```

2.2.8 Names

In R you can put names in almost everything. Below examples for , and .

```
names
```

```
names c  
names
```

2.3 Inputting data into R

Your data can be in different formats. In this section I will show you how you can input text and comma separated value files, which is a very common task. Let's say that you are using a spreadsheet program such as Microsoft Excel or LibreOffice. Then you might want to import your data into R.

- Your first check is, if the first row of your data has a name or not. If they do have names, this means that they have a header.
- Then click on 'Save as' and search text file with extension .txt or .csv.

- Then you edit your file with a notepad (e.g. bloc, gedit, vi or other tool) to check if the decimal character is point ‘.’ and the separator character is comma ‘,’ or semicolon ‘;’ or another character. This depends on the regional configuration of your spreadsheet software.
- If your file has the extension ‘.txt’, use `read.table` and if it is ‘.csv’, use `read.csv`.

Once you checked the format, you can import your ‘.txt’ or ‘.csv’ files in R:

```
read.csv  
read.table
```

3

Structuring an Emissions Inventory

The elaboration of vehicular emissions inventories in vein, consists of four stages:

1. pre-processing activity data,
2. preparing emissions factors,
3. estimating the emissions and
4. post-processing of emissions in maps and databases.

This means that vein provides functions to be used in each of the stages. However, in order to follow these stages, a structure of directories with scripts including vein functions is necessary. Therefore, the vein function was created to create this structure of directories.

3.1 Overview of emissions inventorying using VEIN

3.2 The inventory function

The inventory function creates creates a structure of directories and scripts to run vein. This structure is a suggestion and the user can use any other. The following code shows the arguments of the function. This is done using the base function with .

args

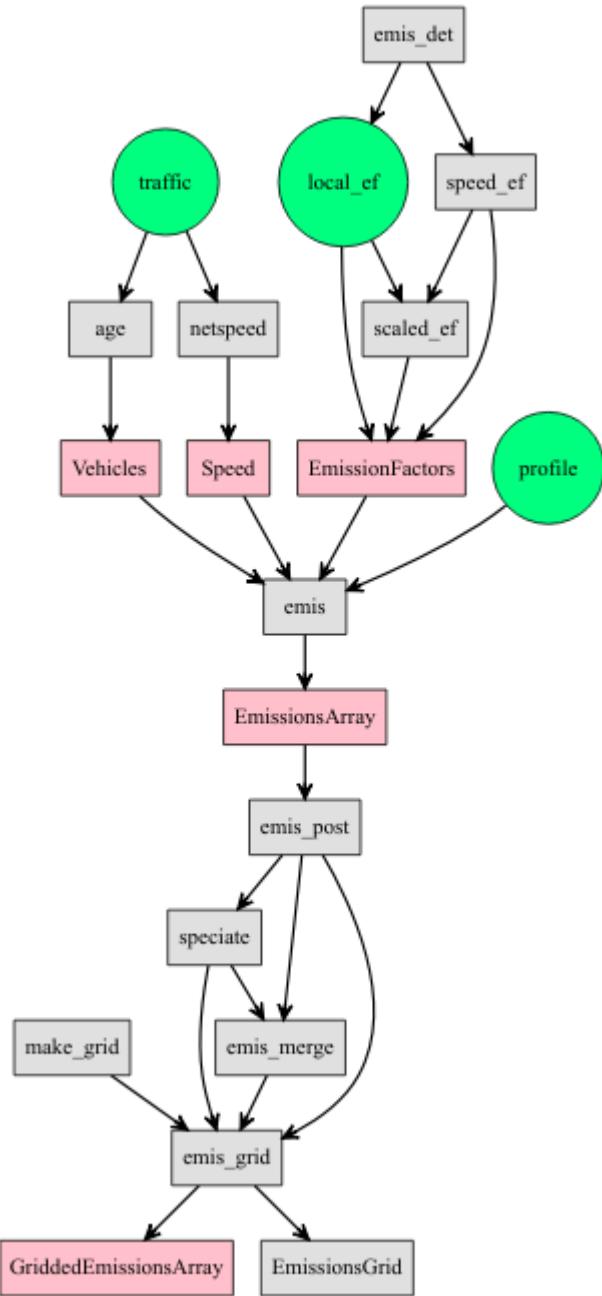


FIGURE 3.1: Structuring an emissions inventories with VEIN

The arguments are , , , and .

- The argument

The argument name is a word to be used as main directory where directories and scripts are stored in. It consists in *one* word with ascii characters. It is also recommended to be used without any special character or accents, for instance:

```
file.pathtempdir  
inventory
```

- The argument This argument is very important and comes from *Vehicular Composition*, which is the classification of the fleet by type of use, type of fuel, size of engine and gross weight, based on definitions of Corvalán et al. (2002). There is also the *technological composition* of the fleet which relates the technological modifications of the car in order to accomplish the emissions standards. However, vein uses a distribution by age of use as 4 shows.

The vehicular composition has 5 types of vehicles:

1. Passenger Cars (PC).
2. Light Commercial Vehicles (LCV).
3. Heavy Good Vehicles or trucks (HGV).
4. Buses (BUS).
5. Motorcycles (MC).

The default value of this argument is: , which means that there are 1 types of PC, 1 of LCV, 1 of trucks, 1 of buses and 1 types of motorcycles. This vehicular composition is only for illustrating that the user can change these values according to its own data. Appendix B shows the vehicular composition from the vehicular emissions inventory of the Environmental Agency of São Paulo, Brazil (CETESB, 2015). In Brazil, the fuel used in vehicles is blended with ethanol with and biodiesel. The user

can use **any** vehicular composition that represents its fleet with up-to 99 types of vehicles per category. For instance, if there are 4 types of PC in a fleet, , and each one has an age of use distribution.

- The argument

This is a logical argument to decide if the output of the function will return print the new directories or not. The use is:

```
file.pathtempdir  
inventory
```

creates the directory “YourCity” and the sub directories: daily, ef, emi, est, images, network and veh.

- **daily**: Directory for storing the profiles saved as .csv files. For instance, is a matrix that could be saved as .csv.
- **ef**: Directory for storing the emission factors data-frame, similar to

but including one column for each of the categories of the vehicular composition. For instance, if PC = 5, there should be 5 columns with emission factors in this file. If LCV = 5, another 5 columns should be present, and so on.

- **emi:** Directory with subdirectories matching the vehicular composition for saving the estimates. It is suggested to use .rds extension instead of .rda.
- **est:** Directory with subdirectories matching the vehicular composition for storing the scripts named .
- **images:** Directory for saving images.
- **network:** Directory for saving the road network with the required attributes. This file will include the vehicular flow per street to be used by functions , , or .
- **post:** Directory for storing the processed emissions. It includes the directories **df** for emissions by age of use, hour and other parameters, **streets** for storing the total emissions by streets and **grids** for storing total gridded emissions by pollutant.
- **veh:** Directory for storing the distribution by age of use of each category of the vehicular composition. Those are data-frames with number of columns with the age distribution and the number of rows as the number of streets. The class of these objects is “Vehicles”.
- The argument

This argument adds scripts into the directories. The default is TRUE. The type of scrpts create are:

- main.R: Adds the , , some comments and a loop to source all inputs. It is recommended not using the loop till it is certain that all scripts are correct.
- traffic.R: Includes two lines with exampmles of how to use the function and saving the output in the directory veh.
- input.R: Each file has the scripts for reading the network, vehicles, emission factors and estimating emissions.

- The argument

This argument prints the scripts created:

```
file.pathtempdir  
inventory
```

- The argument

This is a logical argument that deletes recursively when TRUE, or not when FALSE, the directory and creates another one. Default is TRUE.

The following chapters show the use of other vein functions inside an structure of directories and scripts with the name .

4

Traffic data

Traffic data is read in VEIN as spatial information. In other words, traffic data must be in any vectorial line spatial format with the drivers provided by the library **GDAL** (), which is called by the packages **rgdal** or **sf**. This means that traffic data must be in any GDAL spatial format. The Eq. (4.1) shows how traffic data is treated in VEIN.

$$F_{i,j,k}^* = Q_i \cdot VC_{i,j} \cdot Age_{j,k} \quad (4.1)$$

where $F_{i,j,k}^*$ is the vehicular flow at street link i for vehicle type j by age of use k . j defines the vehicular composition according to its type of use, type of fuel, size of engine and gross weight, based on definitions of (Corvalán et al., 2002). Q_i is the traffic flow at street link i . $VC_{i,j}$ is the fraction of vehicles varying according to the type of vehicles j in the vehicle fleet at street link i . $Age_{j,k}$ is the age distribution by vehicular composition j and age of use k .

This Equation shows that VC splits the total vehicular flow Q to identify the vehicular fraction, which varies according to the type of fuel, size of motor and gross weight. For example, if Q is light duty vehicles (LDV) and it is known that 5% of the Q are passenger cars (PC), with engines lesser than 1400 cc, VC is 0.05. This characterization of the fleet depends on the amount and quality of the available information. VEIN then multiplies the traffic with Age to obtain the amount of each type of vehicle by age of use.

4.1 Sources of traffic data

4.1.1 Travel demand outputs

VEIN was developed according to the available traffic data in São Paulo, Brazil. In this case, the available data was a 4-stage macroscopic travel model for morning rush hours and hourly traffic counts for morning and evening rush hours. The travel model is based on data from an Origin-Destination-Survey (ODS) (Metro, 2017) which started in the decade of 1950 in São Paulo. A classic reference of a 4-stage modeling transport is (Ortuzar and Willumsen, 2002). The 4 stages of the traffic modeling include characterization of trip attractions and productions by zone in some regions, distribution of these trips, the preferred mode of transport for traveling and finally the allocation of the trips at each mode; in this case into the road network. The ODS is made every 10 years by Metro (), which is the underground company, and they perform a smaller update of ODS after 5 years. The information gathered in the ODS is massive due to participation of thousands of commuters. It helps to identify characteristics of the trips inside MASP. CET uses the information from ODS and performs the traffic simulation. In this case, it is a macro or strategic traffic simulation which represents the equilibrium between offer and supply of transportation at maximum load of the road network, that is, at the rush hour, which is from 08:00 to 09:00 Local Time (LT).

VEIN incorporates an extract of a traffic model simulation for the west part of São Paulo named *net*. The Fig. (4.1 shows the traffic of Light Duty Vehicles (LDV). This data covers the surrounding area of the University of São Paulo. The information obtained from CET consists in:

- *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).

The following scripts show how to load this data in R. This data has the class “SpatialLinesDataFrame” from the package (Pebesma and Bivand, 2005). This data was converted into an spatial feature “sf” object (Pebesma, 2017) because it consists of a data-frame with a list-column of geometry with the spatial attributes and it is easier to handle than an sp object class. As mentioned, future versions of VEIN will migrate to sf.

The user must call the library VEIN first, then load the data net. The object net has the class “SpatialLinesDataFrame”. This data can be loaded and shown in R like this:

```
library  
library  
data  
class
```

The Fig. (4.1) shows that traffic is concentrated in main streets such as the motorway Marginal Pinheiros, located in the north east part of the area.

```
library  
data  
$ as.numeric$  
spplot list  
revbpy.colors
```

4.1.2 Interpolation of traffic counts

Travel model simulations are not always available and the user interested in estimating vehicular emissions with a bottom-up approach would have to look for alternatives to obtain traffic data at street level. These alterna-

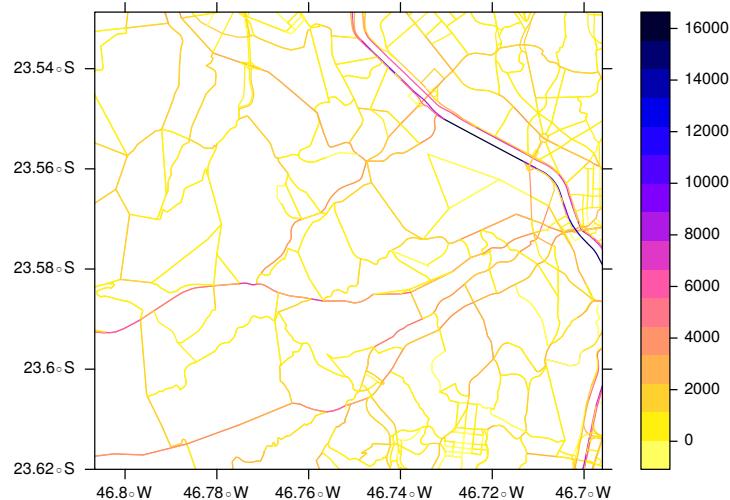


FIGURE 4.1: LDV at 08:00-09:00 (LT) in west São Paulo

tives cover interpolating traffic data for different temporal resolutions. For instance, Réquia Jr et al. (2015) developed a vehicular bottom-up inventory using traffic counts and a geostatistic method called Global Moran's I test (Anselin, 1995). It measures the spatial autocorrelation. However, this technique requires that the observed values are normal, which is not the case for count data.

Other authors were interested in predicting Annual Average Daily Traffic (AADT) or only ADT. Lowry (2014) presented a new method for predicting AADT based on the concept of origin-destination centrality. The idea is to obtain predictor variables directly from the road network. He identified origin and destination zones and added multiplication factors based on the land-use characteristics.

Zhao and Chung (2001) tested several multiple stepwise regressions incorporating land-use characteristics. They considered several variables: number of lanes, classification of road type, employment numbers and access to expressway with correlations between 0.66 and 0.82. Lam and Xu (2000) compared neural networks and regressions for a dataset of 13 locations. Kriging methods were used in AADT interpolation by Eom et al. (2006), who predicted AADT for non-motorway streets. Also, Selby

and Kockelman (2013) compared Kriging and geographically weighted regressions in the prediction of AADT.

Another approach is to perform a spatial regression based on the distribution on the observed data. As the data is counts of traffic, a poisson, quasi-poisson or negative binomial regressions can be used (Zeileis et al., 2008). Recently, Ibarra-Espinosa et al. (2017a) compared quasi-poisson and negative binomial regressions predicting hourly traffic data, with better results for a quasi-poisson approach (correlation of 0.72).

Newer approaches involve GPS data from smartphones and cars.

4.1.3 Generating traffic flows from Origin-Destination-Surveys (ODS)

The origin destination survey (ODS) is an important tool which quantifies the amount of trips generated in the study area. ODS studies started decades ago. The oldest ODS paper on Google Scholar is entitled “FUNCTIONS OF A HIGHWAY-PLANNING SURVEY DEPARTMENT” by W. J. Sapp in 1938 at the TWENTY-FOURTH ANNUAL ROAD SCHOOL (Sapp, 1938). In this paper the importance of information for the traffic engineer is discussed: “The engineer must have the supporting data available to substantiate his decisions.”

Another study of 1942 shows the importance of traffic counts, discussing the elaboration of an origin destination survey where 8000 Boy Scouts of America participated in counting traffic. The method for analyzing the data consisted in counting the vehicles identifying the licence plate in order to determine the origin (the first time the vehicle was recorded), the route of vehicles, the destination and the approximate time for the trip.

After those pioneer studies many other studies went deeper into the subject. And after that, with the irruption of new technologies, new software, use of smartphones with GPS, satellites and big technological centers owned by companies such as Google (), new ways for characterizing trips were developed. In this section i will briefly mention the Google Distance Matrix (), the pgRouting library (Patrushev, 2007) for postGIS and will expand with two R packages, googleway (Cooley, 2018) and dodgr (Padgham and Peutschning, 2018).

TABLE 4.1: Matrix OD for motorized individual trips between 06:30 and 08:30 MASP 2007

	V101	V102	V103	V104	V105	V106	V107	V108	V109	V110
101	42	0	0	83	0	84	42	0	0	318
102	49	0	10	0	0	20	0	10	0	0
103	0	10	0	19	0	81	17	0	0	54
104	0	35	86	40	0	170	40	35	35	0
105	0	0	0	13	0	0	0	0	0	0
106	94	0	72	87	24	196	15	0	0	27
107	0	0	13	64	13	28	44	0	0	0
108	0	0	0	0	0	0	0	3525	885	0
109	0	0	0	0	0	206	0	3916	975	0
110	954	0	18	954	0	0	0	652	939	2928

The ODS will provide us the matrix of pairs of zones origin and destinations by mode of transport. Then, we can use any of the mentioned softwares to find the shortest path between each zone. The table 4.1 shows an extraction of the OD matrix for the Metropolitan Area of São Paulo (MASP) in 2007 for motorized individual trips. The fig @ref(fig.zod) shows a map with the location of the zones OD.

Now, as we know the trips between each pair of OD zones, we can find the routes that connects them. Algorithms are used to find the shortest path. It connects two nodes minimizing the total travel time. There are several algorithms, including Dijkstra (1959).

4.1.3.1 Google Distance Matrix API

This service allows to “retrieve duration and distance values based on the recommended route between start and end points.” It returns durations and distance and it is possible to choose modes of transport as well as to choose between current or historical times. The first step is to get a KEY.

If you browse

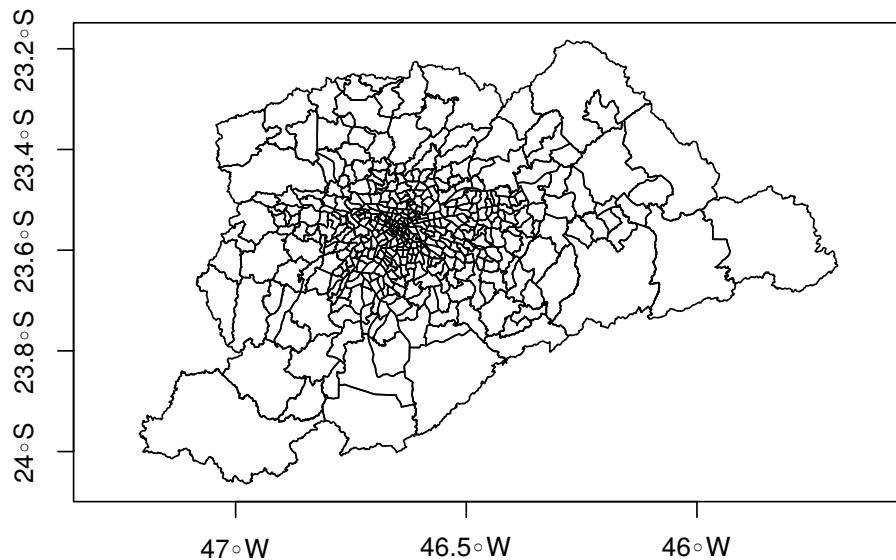


FIGURE 4.2: Zones OD for MASP 2007

&&&

and replace YOUR_API_KEY you will see (my brower is in portuguese):

{

{

{

}

```
    }  
  
}
```

The modes of transport covered are **driving** using the road network, **walking** via pedestrian paths & sidewalks, **bicycling** via bicycle paths and **transit** via public transit routes. For more information read the documentation.

4.1.3.2 pgRouting for postGIS

pgRouting () is an open source routing library for postGIS. It provides many routing algorithms and it can be run via QGIS (). This library is very extensive. A good resource is the book “pgRouting: A Practical Guide” () .

4.1.3.3 The R package googleway

googleway (Cooley, 2018) R package allows to access Goolge Maps API . The API functions are:

- Directions -
- Distance Matrix -
- Elevation -
- Geocoding -
- Reverse Geocoding -
- Places -
- Place Details -
- Time zone -
- Roads - and

This package allows to plot over Google Maps.

```
library
```

```
set_key  
google_keys
```

From zone Luz 7, coordinates long -46.63461 and lat -23.53137 to zone 8 Bom Retiro coordinates -46.64482 -23.52204 there are 133 LDV trips between 6:30 and 08:30. We first create the data-frame. This data is based on OD from São Paulo.

```
data.frame
```

Then we use the package to create the points between origin and destination and maptools to transform points to lines.

```
library
library
google_directions unlist :
unlist :

decode_pl$$$
::st_as_sf c
list $ $
map2SpatialLines
plot

:::include_graphics
```

Route of 133 LDV trips

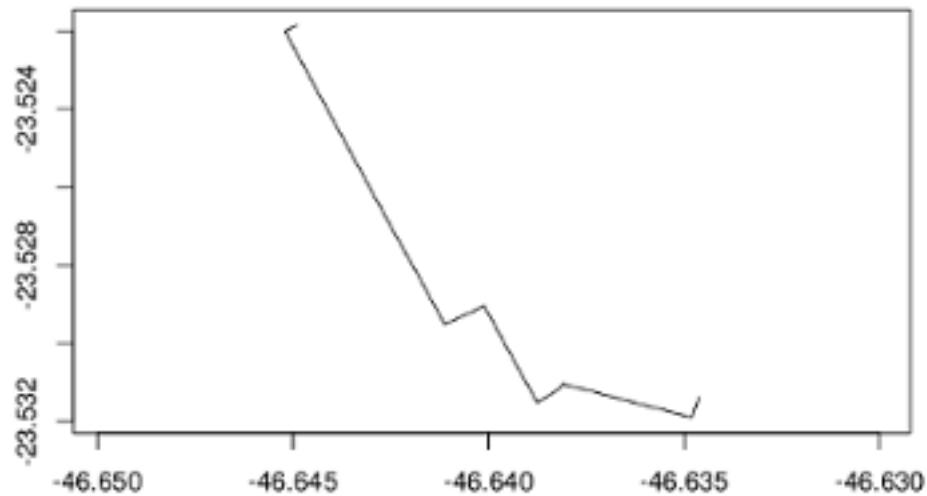


FIGURE 4.3: Driving route between zones 'Luz' and 'Bom Retiro' in São Paulo

4.1.3.4 The R package dodgr

dodgr(Padgham and Peutschning, 2018) is a new R package with great capabilities. It calculates the distance on dual-weight graphs using priority-queue shortest paths. In order to calculate the traffic flows it is necessary to have the matrix OD for the desired mode of transport and the coordinates of centroids of the OD zones.

The function uses the osmdata r package to download street network data from Open Street Map (contributors, 2017) for the points of the centroids of zones OD. Depending on the spatial extent of the data, the resulting data can be large. The function weight the lines from OSM road network according to a specific profile of traffic: “psv” for Public Service Vehicle, “bicycle”, and others. The weights are profile values for the OSM type of roads based on this webpage: . The function extract the vertices pf the graph including the coordinates. Finally, the function reads the graph and plots the flows. Below is an example for São Paulo using the data of ODS (Metro, 2017).

4.1.4 Top-down approach

VEIN was designed with traffic flow at street leve on mind, however, it is possible that user might want or need to use a top-down approach. Here are some possible causes:

1. Bottom-up approach can demand more computatonal resources for a country or a continent.
2. Another possibility is that the emissions inventory is going to be used solely for air quality modelling purposes, where the proportion of grid spacing and street is such that spatial detail would be lose, for instance with resolution of 10 km.
3. Also, it is possible that there is no way of obtaining traffic information at street level.
4. Limited resources, time, funding, human resources, etc.
5. Lastly, due to simplism. It is also possible that the objective simply is to estimate an emissions inventory using a top-down approach.

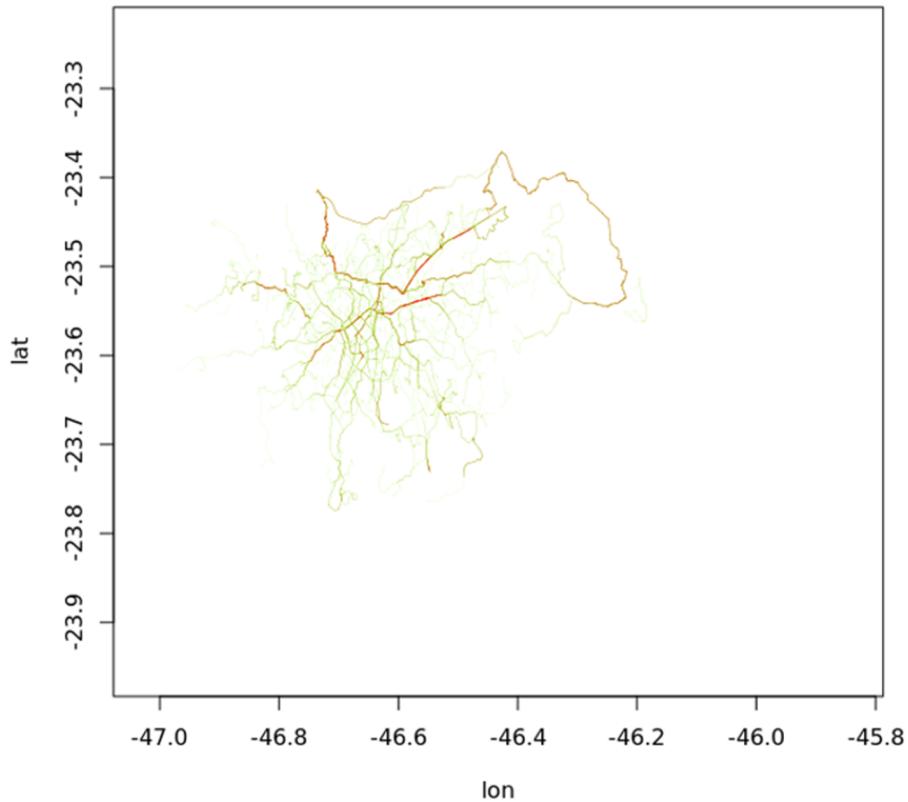


FIGURE 4.4: Daily trips of Light Duty Vehicles in São Paulo

Under these circumstances a top-down approach would be better suited for some users. As VEIN is a toolkit for estimating emissions, it is reasonable to use all VEIN resources with a top-down approach.

In this case, the user must follow some considerations:

- Use functions in the same way as shown on section 3.
- Create a network, but instead of using a road network with spatial lines, use spatial polygons. The Spatial polygon might represent some area where the amount of vehicles is known.
- The functions shown in following sections, show that it is possible to apply age distributions to each street, in this case, to each area.

The Fig. 4.5 shows the emissions of PM for each state Brazil and pre-

pared for a congress using VEIN as a top-down tool (Ibarra-Espinosa et al., 2018a).

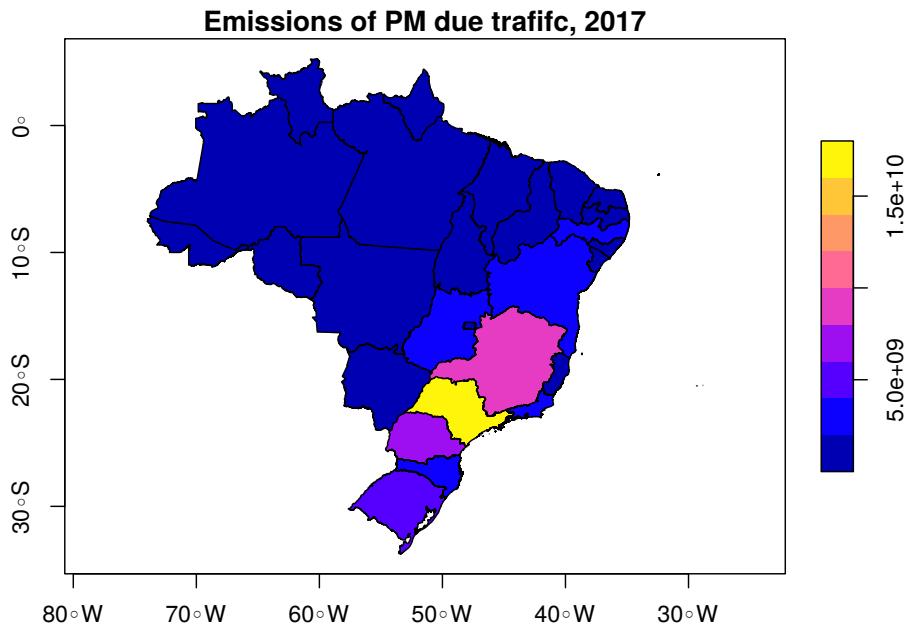


FIGURE 4.5: Emissions of PM due to traffic, 2017

4.2 Main functions

The Fig.(3.1) shows a complete diagram with the steps for elaborating an emissions inventory. Also, the function shown on sub-section (3.1) shows the functions to structure an inventory. This and the following subsection of application shows respective the part of the diagram Fig.(4.6) and how to run VEIN as well as storing the results in the directories shown in Fig. (3).

The first element that the user must have is the traffic at street level. This is shown as a green circle with the word 'traffic' on Fig. (4.6). The user must use any function which produces objects of class . The function produces an object of class . This objects are required by the function.

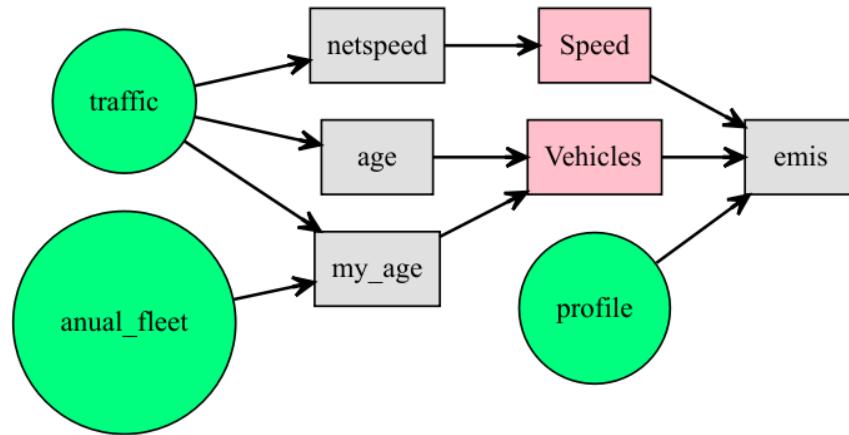


FIGURE 4.6: Structuring an emissions inventories with VEIN

4.2.1 Expanding traffic data with the function `temp_fact`

Traffic data must be temporally extrapolated because it is usually available only for the morning rush hour. Traffic data can be estimated from short period traffic count datasets, then expanded to represent longer timespan, such as Annual Average Daily Traffic (AADT; (Wang and Kockelman, 2009, lam2000estimation)). The next step is to extrapolate the vehicular flow at street link i , vehicle type j , and age of use k , to obtain the vehicular flow for hour of the week l ($F_{i,j,k,l}$; see equation (4.2).

$$F_{i,j,k,l} = F_{i,j,k}^* \cdot TF_{j,l} \quad (4.2)$$

where $TF_{j,l}$ are the temporal factors varying according to each hour of l and type of vehicle j . For instance, TF is defined as a matrix with 24 lines and numbers of columns to each day considered, from Monday to Sunday. In order to expand traffic to other hours, TF matrices must be normalized to the hour that represents the traffic data. It means that TF values at morning peak hour must be 1 and the respective proportion must be assigned to the other hours. For example, TF values can be obtained from automatic traffic count stations.

The function `return $F_{i,j,k,l}$` as an object with class . The arguments are:

- is a data-frame of traffic flow to each hour (veh/h) at each street.
- s a matrix or data-frame to extrapolate .

4.2.2 Calculating speed at other hours with the function `netspeed`

The average speed of traffic flow is very important and it must be determined for each link and hour. Once the vehicular flow is identified for each hour, the average speed is then identified for each hour. This was accomplished by employing curves from the Bureau of Public Roads (BPR; (Bureau of Public Roads, 1964)), as shown in Eq. (4.3). The process involves calculating speed by dividing the length of road by the time. The time is calculated using the total traffic expanded to each street link i and hour l .

$$T_{i,l} = To_i \cdot \left(1 + \alpha \cdot \left(\frac{Q_{i,l}}{C_i} \right)^\beta \right) \quad (4.3)$$

The function do this calculations. The arguments of this function are:

args

allows basically creates two types of speeds data-frames which depends on the availability of data by the user. If the user has , , and the user can use the argument , or when the user only has and the user can use the argument .

- is a data-frame of traffic flow to each hour (veh/h) at each street.
 - is the Peak speed (km/h) at each street.
 - Free flow speed (km/h) at each street.
 - Capacity of link (veh/h) at each street.
 - Distance of link (km) of each street.
 - Parameter of Bureau of Public Roads (1964) curves.
 - Parameter of Bureau of Public Roads (1964) curves.
 - Logical to create a Speed data-frame with 24 hours and a default profile.
- It needs and at each street:

TABLE 4.2: speeds for scheme = T

Period	Speed
00:00-06:00	ffs (Free flow speed)
06:00-07:00	average between ffs and ps
07:00-10:00	ps (Peakspeed)
10:00-17:00	average between ffs and ps
17:00-20:00	ps (Peak speed)
20:00-22:00	average between ffs and ps
22:00-00:00	ffs (Free flow speed)

4.2.3 Distribution of vehicles by age of use with the functions `age_ldv`, `age_hdv` and `age_moto`

These functions reads traffic data at each street and return an object of class , which is a data-frame with the number of vehicles at each street and the number of columns represent the amount of vehicles at each age. The functions and are identical. These functions apply survival equations into the fleet from Ministerio do Meio Ambiente (2011). The arguments are:

- numerical vector of vehicles at each street.
- word of vehicle assigned to columns of dataframe.
- parameter of survival equation. The default value for is 1.698 and for and is 0.2.
- parameter of survival equation. The default value for is -0.2 and for and is 17.
- age of newest vehicles for that category. The default value is 1, however it can be bigger than one when it is a vehicle that is not circulating more then a year ago.
- age of oldest vehicles for that category. The default value is 50 assuming that the oldest vehicle in circulation has 50 years of use. However, new type of vehicles will be newer.
- multiplication factor. This factor helps to split the traffic in the vehicular composition.

- when TRUE it is expecting that `and` are numeric vectors with length equal to `. In other words, the values of and varies in each street.`
- message with average age and total number of vehicles.

4.2.4 The function `my_age`

These functions also reads traffic data at each street and returns an object of class `, which is a data-frame with the number of vehicles at each street and the columns represent the amount of vehicles at each age. However, it is based on data the user has and not by parameters. Therefore, using this function with own data should produce more representative results.`

The arguments are:

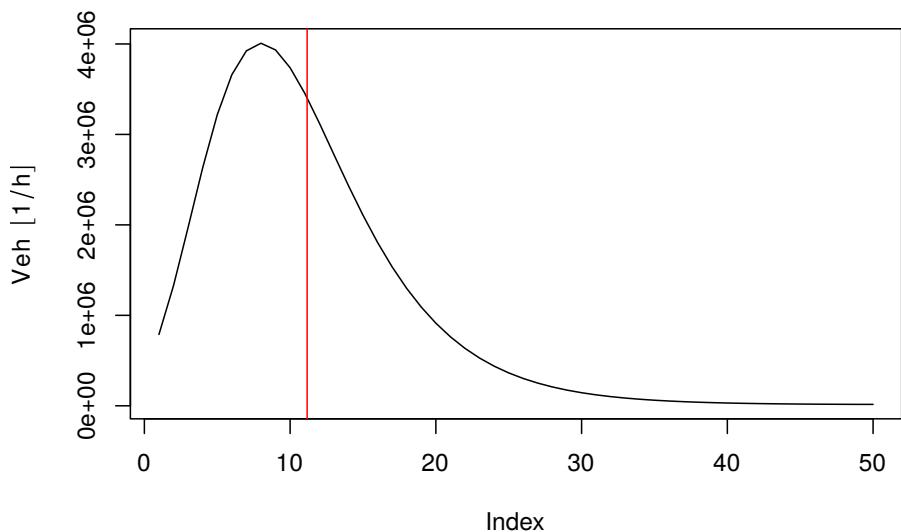
args

- numerical vector of vehicles at each street.
- Age distribution of vehicles. This parameter can be annual sales or annual registry for the category of vehicle.
- of vehicle assigned to columns of dataframe.
- multiplication factor. This factor helps to split the traffic in the vehicular composition.
- message with average age and total number of vehicles.

4.2.5 The class `Vehicles`

is a class in VEIN, shown in on Fig. (4.6). This class includes the methods `, and . This means that, objects presents customized versions of , and in order to make easier to the user to use VEIN. The figure @ref(fig.plotpc) shows a simple plot of a object, followed by the of this object. The plot shows the sum of each type of vehicle by age of use, with a vertical red line indicating the average age, 11.17 years of use.`

```
library  
data  
  age_ldv $ /*  
plot
```



```
head :
```

4.2.6 Other traffic functions

Another function is which calculates average daily traffic (ADT) from hourly traffic data. This function reads numeric vectors with the amount of vehicles at each street and expands the traffic with temporal factors for each type of vehicle. The arguments are:

args

- numeric vector for passenger cars
- numeric vector for light commercial vehicles
- numeric vector for heavy good vehicles or trucks
- numeric vector for bus
- numeric vector for motorcycles
- data-frame profile for passenger cars
- data-frame profile for light commercial vehicles
- data-frame profile for heavy good vehicles or trucks
- data-frame profile for bus
- data-frame profile for motorcycles

4.3 Vehicular composition

4.4 Application

As mentioned above, the application consists in using the travel demand model for west part of São Paulo city, present in VEIN. This script reads traffic data and expands it applying an age function. It uses the function with the default vehicular composition, shown on section (@red(st)).

Please, use the scripts in the extra material of this book. The base year is 2015 and the extended example is available in appendix 1.

```
library  
inventoryfile.pathtempdir  
setwd  
data  
data  
data  
temp_fact$+$  
netspeed $ $ $ $  
saveRDS  
saveRDS
```

5

Emission Factors

An emission factor is the amount of mass of pollutant generated by activities (Pulses and Heslinga, 2010). In the case of vehicles, the emission factors are generally known as the mass of pollutant per traveled distance $g \cdot km^{-1}$ in metric units. The distance unit could be different, like miles for instance. The type of units depends on measurement used when developing the emission factors.

There are hot and cold exhaust emissions. *Hot* exhaust emissions are produced by a vehicle when its engine and exhaust after-treatment system are at their normal operational temperatures and *Cold* during the vehicle warm-up phase (P et al., 2009).

The emission factors come from measurements of pollutants emitted by the sources. In the case of vehicles, the dynamometer measurements of the emission factors are the mass of pollutant collected during the distance traveled by a vehicle following a specific driving cycle. The Federal Test Procedure (FTP) is a test to certify the vehicle emissions under the driving cycles such as FTP-75. This driving cycle has a duration of 1874 s for a distance of 17783 m and an average speed of $34.12 \text{ km} \cdot h^{-1}$ (Barlow, 2009).

In the literature it is possible to find different type of emission factors. For instance, in Europe, during the development of the project Assessment of Reliability of Transport Emission Models and Inventory Systems (ARTEMIS), the *traffic situation model* was developed. The model concept consists in the combination of parameters that affects vehicle kinematics and engine operation: type of road, sinuosity and gradient, traffic conditions or level of service (free-flow, heavy, saturated and stop and go) and average speed by speed limit of the road (Boulter and Barlow, 2009). Measurements of vehicular emissions under each traffic situation lead to emission factors by traffic situation, used in the Handbook of Emission Factors (HBEFA, www.hbefa.net).

There is another approach based on 1 Hz measurement of emissions. Here, the pollutants are measured on real live traffic driving. Emission factors can be derived here using mainly two approaches. One approach is applying the Vehicular Specific Power (VSP) (Jimenez et al., 1999). This method is used in the Motor Vehicle Emission Simulator (MOVES) (Koupal et al., 2003) and the International Vehicle Emissions model (IVE) (Davis et al., 2005). Another approach is to apply the Passenger car and Heavy-duty Emission Model (PHEM), which is based on instantaneous engine power demand and engine speed developed during the ARTEMIS project (Barlow, 2009, Boulter and Barlow (2009)).

Perhaps the most used approach to obtain emission factors are speed functions. As emissions tests on driving cycles are valid for only one average speed, measuring emissions of different driving cycles at different average speeds, allows to obtain point clouds where a regression is applied, obtaining an emission factor as a speed function. The literature shows experiments in different parts of the world, such as in Chile (Corvalán et al., 2002) and Europe (Ntziachristos and Samaras, 2000). However, probably the most used source of emission factors as speed functions are the European emission guidelines developed by the European Monitoring and Evaluation Programme (EMEP,) and published by the European Environment Agency (EEA,) (Ntziachristos and Samaras, 2016). These emission factors are included in the VEIN model.

Another aspect important in vehicular emissions is the degradation of the emissions of aged vehicles. Over the time, different part of the vehicles suffer deterioration, increasing the emissions. Therefore, approaches have been developed to account for this effect in inventories. One approach consists in determining functions that increment the emissions by accumulated mileage, such as studies of Corvalán and Vargas (2003).

In Brazil, the emission factors are reported by the Environmental Protection Agency of São Paulo CETESB (2015), in their vehicular emissions inventory report. Light vehicles emission factors come from average measurements of the FTP75 driving cycle, motorcycle emission factors come from average measurements of World Motorcycle Test Cycle (WMTC) and HDV from the European Stationary Cycle (ETC).

As the VEIN model was developed in Brazil, but users of other parts of

the world may be interested in using the model, the approach in VEIN has to allow flexibility to the user. Therefore, the emission factors approach in VEIN consists in three options, as shown on Fig. (5.1). The first approach consists in local emissions factors by age of use, such as the one from CETESB (2015), denoted as `local_ef` in the green circle. The second approach consists in selecting speed functions from Ntziachristos and Samaras (2016), available in VEIN and denoted as the grey box as `speed_ef`. The third approach consists in incorporating speed variation from the speed function into the local emission factors by age of use. In any case, VEIN uses a function to account for the degradation of emissions due to accumulated mileage (denoted in grey box as `emis_det`). These emissions are entered into the emission estimation function.

The following sections show how to do these calculations in VEIN, step by step.

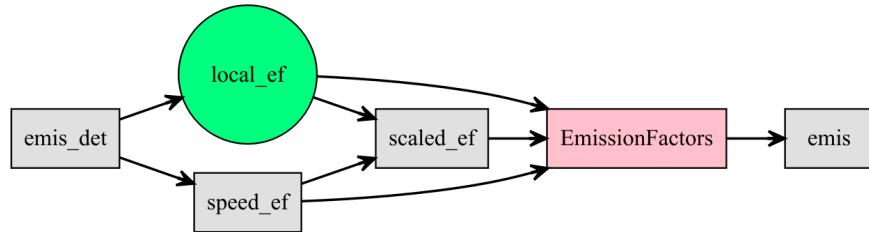


FIGURE 5.1: Structuring an emissions inventories with VEIN

5.1 Speed functions `ef_ldv_speed` and `ef_hdv_speed`

The estimation of emissions in VEIN is shown in the equation (5.1):

$$EH_{i,j,k,l,m} = F_{i,j,k,l} \cdot L_i \cdot EF(V_{i,l})_{j,k,m} \cdot DF_{j,k} \quad (5.1)$$

where $EH_{i,j,k,l,m}$ is the emission for each street link i , vehicle category from composition k , hour l and pollutant m , $F_{i,j,k,l}$ is the vehicular flow

calculated in Eq. 1. L_i is the length of the street link i . $EF(V_{i,l})_{j,k,m}$ is the emission factor of each pollutant m . $DF_{j,k}$ is the deterioration factor for vehicle of type j and age of use k , explained in detail in section 5.5.

These functions return speed functions as $f(V)$. The functions come from the EMEP/EEA emissions guidelines (Ntziachristos and Samaras, 2016) and are available in PDF reports. The equations and their parameters were translated to spreadsheets and loaded internally in VEIN. Therefore, the user only must enter the required parameters obtaining the desired function. The functions respect the assumptions and speed limits.

5.1.1 Emission factors of LDV depending on speed with the function `ef_ldv_speed`

The arguments for are:

- : Category of vehicle: “PC”, “LCV”, “Motorcycle” or “Moped”.
- : Sub-category of vehicle: PC: “ECE_1501”, “ECE_1502”, “ECE_1503”, “ECE_1504”, “IMPROVED_CONVENTIONAL”, “OPEN_LOOP”, “ALL”, “2S” or “4S”. LCV: “4S”, Motorcycle: “2S” or “4S”. Moped: “2S” or “4S”.
- : Size of engine in cc: PC: “<=1400”, “>1400”, “1400_2000”, “>2000”, “<=800”, “<=2000”. Motorcycle: “>=50” (for “2S”), “<=250”, “250_750”, “>=750”. Moped: “<=50”. LCV : “<3.5” for gross weight.
- : Type of fuel: “G”, “D”, “LPG” or “FH” (Full Hybrid: starts by electric motor).
- : Euro standard: “PRE”, “I”, “II”, “III”, “III+DPF”, “IV”, “V”, “VI” or “VIc”.
- : Pollutants: **Criteria**: “CO”, “NOx”, “HC”, “PM”, “CH4”, “NMHC”, “CO2”, “SO2”, “Pb”, “FC” (Fuel Consumption).- **PAH and POP**: “indeno(1,2,3-cd)pyrene”, “benzo(k)fluoranthene”, “benzo(b)fluoranthene”, “benzo(ghi)perylene”, “fluoranthene”, “benzo(a)pyrene”, “pyrene”, “perylene”, “anthanthrene”, “benzo(b)fluorene”, “benzo(e)pyrene”, “triphenylene”, “benzo(j)fluoranthene”, “dibenzo(a,j)anthacene”, “dibenzo(a,l)pyrene”, “3,6-dimethyl-phenanthrene”, “benzo(a)anthracene”, “acenaphthylene”, “acenaphthene”, “fluorene”, “chrysene”, “phenanthrene”, “naphthalene”, “anthracene”, “coronene”, “dibenzo(ah)anthracene” **Dioxins and Furans**: “PCDD”, “PCDF”, “PCB”. **Metals**: “As”, “Cd”, “Cr”, “Cu”, “Hg”, “Ni”, “Pb”, “Se”, “Zn”. **NMHC**: - **ALKANES**: “ethane”, “propane”, “butane”, “isobutane”, “pentane”, “isopentane”, “hexane”,

TABLE 5.1: Most used combinations for emission factors for LDV

Argument	Value
v	PC LCV Motorcycles
t	4S
cc	<=1400 1400_2000 >2000
f	G D
eu	PRE I II III IV V VI
p	CO NOx HC FC PM

“heptane”, “octane”, “TWO_methylhexane”, “nonane”, “TWO_methylheptane”, “THREE_methylhexane”, “decane”, “THREE_methylheptane”, “alcanes_C10_C12”, “alkanes_C13”. CYCLOALKANES: “cycloalcanes”. ALKENES: “ethylene”, “propylene”, “propadiene”, “ONE_butene”, “isobutene”, “TWO_butene”, “ONE_3_butadiene”, “ONE_pentene”, “TWO_pentene”, “ONE_hexene”, “dimethylhexene”. ALKYNES: “ONE_butine”, “propyne”, “acetylene”. ALDEHYDES: “formaldehyde”, “acetaldehyde”, “acrolein”, “benzaldehyde”, “crotonaldehyde”, “methacrolein”, “butyraldehyde”, “isobutanaldehyde”, “propionaldehyde”, “hexanal”, “i_valeraldehyde”, “valeraldehyde”, “o_tolualdehyde”, “m_tolualdehyde”, “p_tolualdehyde”. KETONES: “acetone”, “methylethylketone”. AROMATICS: “toluene”, “ethylbenzene”, “m_p_xylene”, “o_xylene”, “ONE_2_3_trimethylbenzene”, “ONE_2_4_trimethylbenzene”, “ONE_3_5_trimethylbenzene”, “styrene”, “benzene”, “C9”, “C10”, “C13”.

- : Multiplication factor.
- : Option to see (or not) the equation parameters.

It is very important that the user enters the right values into the arguments because the final category must match. If the user enters values that do not match, the resulting speed function will be wrong. There are 146 matchs. The probably most used combinations are:

The returning speed function represents how a type of vehicle emits pollutants on average. Fig. (5.2) shows the emissions of CO by Passenger Cars with technologies Pre Euro and Euro I from Ntziachristos and Samaras (2016). Higher emissions occure at low average speeds and older

vehicles with technologies pre Euro emit more than vehicles with Euro I. Even more, the average emissions of Pre Euro are $29.31 \text{ g} \cdot \text{km}^{-1}$ and Euro I $2.61 \text{ g} \cdot \text{km}^{-1}$, meaning that, on average, a vehicle Pre euro is 14 times higher than a vehicle with euro I.

The Fig. (5.2) shows the plot using the package ggplot2 (Wickham, 2009). This package requires that the data must be in long format.

```
library
library
  ef_ldv_speed

  ef_ldv_speed

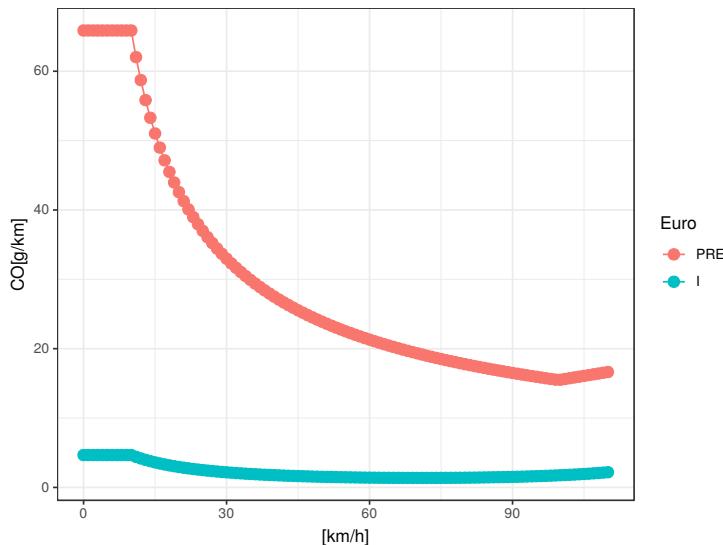
  :
  data.frame cef1 ef2
$ 
$ factorcrep rep
  c
ggplot aes   + geom_point  +
  geom_line + theme_bw+
  labs
```

5.1.2 Emission factors of HDV depending on speed with the function ef_hdv_speed

The arguments for are:

```
args
```

- Category of vehicle: “Coach”, “Trucks” or “Ubus”
- Sub-category of vehicle: “3Axes”, “Artic”, “Midi”, “RT,”Std” and “TT”
- Gross weight of each category: “ ≤ 18 ”, “ > 18 ”, “ ≤ 15 ”, “ $> 15 \& \leq 18$ ”, “ ≤ 7.5 ”, - “ $> 7.5 \& \leq 12$ ”, “ $> 12 \& \leq 14$ ”, “ $> 14 \& \leq 20$ ”, “ $> 20 \& \leq 26$ ”, “ $> 26 \& \leq 28$ ”, “ > 28

**FIGURE 5.2:** Emission factor as a speed function

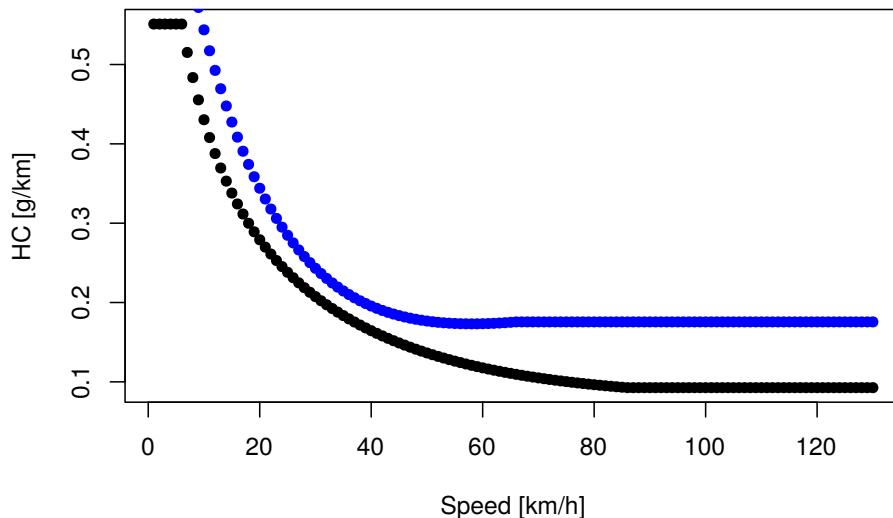
& <=32”, “>32”, “>20 & <=28”, “>28 & <=34”, “>34 & <=40”, “>40 & <=50” or
“>50 & <=60”

- ‘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
- Load of the vehicle: 0.0, 0.5 or 1.0
- Pollutant: **Criteria:** “CO”, “NOx”, “HC”, “PM”, “CH4”, “NMHC”, “CO2”, “SO2”, “Pb”, “FC” (Fuel Consumption).- **PAH and POP:** “indeno(1,2,3-cd)pyrene”, “benzo(k)fluoranthene”, “benzo(b)fluoranthene”, “benzo(ghi)perylene”, “fluoranthene”, “benzo(a)pyrene”, “pyrene”, “perylene”, “anthanthrene”, “benzo(b)fluorene”, “benzo(e)pyrene”, “triphenylene”, “benzo(j)fluoranthene”, “dibenzo(a,j)anthacene”, “dibenzo(a,l)pyrene”, “3,6-dimethyl-phenanthrene”, “benzo(a)anthracene”, “acenaphthylene”, “acenaphthene”, “fluorene”, “chrysene”, “phenanthrene”, “naphthalene”, “anthracene”, “coronene”, “dibenzo(ah)anthracene” **Dioxins and Furans:** “PCDD”, “PCDF”, “PCB”. **Metals:** “As”, “Cd”, “Cr”, “Cu”, “Hg”, “Ni”, “Pb”, “Se”, “Zn”. **NMHC:** - ALKANES: “ethane”, “propane”, “butane”, “isobutane”, “pentane”, “isopentane”, “hexane”, “heptane”, “octane”, “TWO_methylhexane”, “nonane”, “TWO_methylheptane”, “THREE_methylhexane”, “decane”, “THREE_methylheptane”, “alcanes_C10_C12”, “alkanes_C13”. CYCLOALKANES:

"cycloalcanes". ALKENES: "ethylene", "propylene", "propadiene", "ONE_butene", "isobutene", "TWO_butene", "ONE_3_butadiene", "ONE_pentene", "TWO_pentene", "ONE_hexene", "dimethylhexene". ALKYNES: "ONE_butine", "propyne", "acetylene". ALDEHYDES: "formaldehyde", "acetaldehyde", "acrolein", "benzaldehyde", "crotonaldehyde", "methacrolein", "butyraldehyde", "isobutanaldehyde", "propionaldehyde", "hexanal", "i_valeraldehyde", "valeraldehyde", "o_tolualdehyde", "m_tolualdehyde", "p_tolualdehyde". KETONES: "acetone", "methylethylketone". AROMATICS: "toluene", "ethylbenzene", "m_p_xylene", "o_xylene", "ONE_2_3_trimethylbenzene", "ONE_2_4_trimethylbenzene", "ONE_3_5_trimethylbenzene", "styrene", "benzene", "C9", "C10", "C13".

- : Multiplication factor.
- Multiplication factor
- Option to see or not the equation parameters

```
:  
ef_hdv_speed  
  
ef_hdv_speed  
  
plot: ef1:  
  
lines: ef2:
```



5.2 Local emission factors by age of use. The functions `EmissionFactors` and `EmissionFactorsList`

The functions and create the classes with same names. The class is also a data-frame with numeric columns with units of $g \cdot km^{-1}$. This class is not used in critical steps of VEIN, and it has a purpose of creating a data-frame with numeric columns and units. The function is applied to a numeric vector and it just converts it to adding units to it.

In the case of , this class is a list of functions $f(V)$. This is usefull because the function, seen in detail on chapter 6, requires that the emission factors are functions of velocity $f(V)$, regardless $f(V)$ if it is constant or not. When this function is applied to a data-frame or to a matrix, it returns a list and each column of the data-frame or matrix is another list. Also, each row of the original data-frame is converted to a $f(V)$.

The arguments or and are:

-: Object with class “data.frame”, “matrix” or “numeric”

The following lines of code show examples using . Firstly, it is loaded the dataset of emission factors fe2015 for CO and the vehicles PC using gasoline

PC_G and Light Trucks LT. The class of fe2015 is data-frame. The subset of the data is named DF_CO. The first 6 elemens of the data-frame are numbers without units.

```
library  
data  
$== c  
head
```

The object EF_CO is of class and “data-frame” and each observation has the units $g \cdot km^{-1}$.

```
EmissionFactors  
head
```

```
class
```

The plot of an object of class shows maximum 9 emission factors. Fig. (5.3)

shows the emission factors of both categories, where newer vehicles emit less and older vehicles emit more.

plot

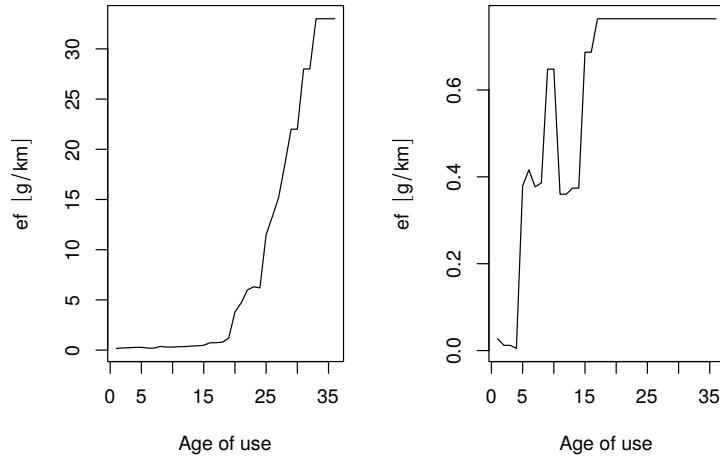


FIGURE 5.3: Emission factor of PC and LT by age of use

Regarding the class , the and methods are very simple, only showing a description of the object. The example of the following lines of code also uses the same set of Emission Factors from CETESB (2015). The resulting object, EF_COL, is of class and . The print of the object shows only a description of the first and last member of the list object.

EmissionFactorsList
class

Indeed, if we want to see the content of the we have to extract the elements

of the list. EFL_CO has two lists; one for PC and one for LT. Then, each list has 36 functions, the command shows the object is a function $f(V)$.

Also, as the speed functions are constant, there is no need to add values of V.

Finally, after extracting the value, Fig. (5.4) shows the same figure as Fig. (5.3)

```
sapply: function  
sapply: function  
EmissionFactors$bind  
plot
```

5.3 Incorporating speed variation with the functions `ef_ldv_scaled` and `ef_hdv_scaled`

The functions in section 5.1 and 5.2 showed the speed and local emission factors respectively. However, the user might have local and constant emission factors depending only on the age of use, but also, outputs from a traffic demand model which includes speed. Under those circumstances,

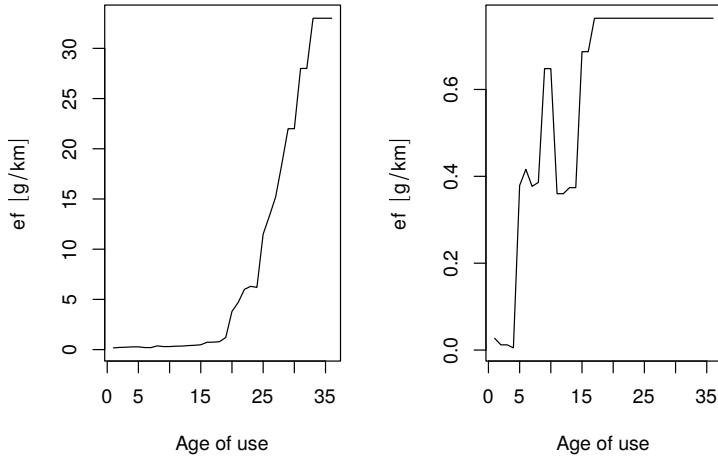


FIGURE 5.4: Emission factor of PC and LT

it would be interesting to investigate the effects of the average speed of traffic flow. VEIN includes functions to transform a constant emission factor by age of use into a speed dependent function.

These functions are called `and`. The concept is explained in detail in my Ph.D thesis (Ibarra-Espinosa, 2017). The idea comes from revisiting the nature of the local emission factors. The Brazilian emission factors consist of average measurements for emissions certification tests which use the driving cycles Federal Test Procedure FTP-75 for LDV, World Motorcycle Test Cycle (WMTC) for motorcycles, and also the European Stationary Cycle (ESC) for HDV (CETESB, 2015). The average speed of FTP-75 and WMTC are known, being $34.2 \text{ km} \cdot \text{h}^{-1}$ and $54.7 \text{ km} \cdot \text{h}^{-1}$ based on the reference report on driving cycles from the Transport and Research Laboratory (TRL) (Barlow, 2009). In the case of ESC, the user might search on literature a way to relate this cycle with an average speed.

Coming back to the CETESB (2015) emission factors, these are annual average values of emissions for the different technologies to accomplish Brazilian emission standards, which are valid only for a specific average speed. If two vehicles have similar technology but different emission factors, one constant and other speed function, it would be possible to relate them. These functions use the speed to relate both emission factors. The condition to transform a constant emission factor into a speed dependent

function is that, the new speed dependent function must be of the same value at the speed which is representative for the local factor, and the values at other speeds are the incorporation of the speed variation. The equation (5.2) shows the procedure:

$$EF_{scaled}(V_{i,l})_{j,k,m} = EF(V_{i,l})_{j,k,m} \cdot \frac{EF_{local,j,k,m}}{EF(Vdc_{i,l})_{j,k,m}} \quad (5.2)$$

Where $EF_{scaled}(V_{i,l})_{j,k,m}$ is the scaled emission factor and $EF(V_{i,l})_{j,k,m}$ is the Copert emission factor for each street link i , vehicle from composition k , hour l and pollutant m . $EF_{local,j,k,m}$ represents the constant emission factor (not as speed functions). $EF(Vdc_{i,l})_{j,k,m}$ are Copert emission factors with average speed value of the respective driving cycle for the vehicular category j .

The functions and return scaled emissions factors automatically. The arguments are:

- : Data-frame with emission factors (Deprecated in 0.3.10).
- : Numeric vector with the emission factors constant by age of use. These emission factors are the target to incorporate speed variation.
- : A number with the speed of the driving cycle of the emission factors from the argument . The resulting speed functions will return the same values of dfcol at the speed . The default is 34.12 “km/h”.
- . Category of vehicle: “PC”, “LCV”, “Motorcycle” or “Moped”.
- . Character; sub-category of vehicle: PC: “ECE_1501”, “ECE_1502”, “ECE_1503”, “ECE_1504” , “IMPROVED_CONVENTIONAL”, “OPEN_LOOP”, “ALL”, “2S” or “4S”. LCV: “4S”, Motorcycle: “2S” or “4S”. Moped: “2S” or “4S”. The default is .
- . Character; size of engine in cc: PC: “<=1400”, “>1400”, “1400_2000”, “>2000”, “<=800”, “<=2000”. Motorcycle: “>=50” (for “2S”), “<=250”, “250_750”, “>=750”. Moped: “<=50”. LCV : “<3.5” for gross weight.
- : Character; type of fuel: “G”, “D”, “LPG” or “FH” (Full Hybrid: starts by electric motor).
- : Character; euro standard: “PRE”, “I”, “II”, “III”, “III+DPF”, “IV”, “V”, “VI” or “VIc”.
- : Character; pollutant: “CO”, “FC”, “NOx”, “HC” or “PM”.

And the argument of are: !! something is missing here!!

5.4 Cold Starts

Cold start emissions are produced during engine startup, when the engine and/or catalytic converter system has not reached its normal operational temperature. Several studies have shown the significant impact for these types of emissions (Chen et al., 2011, Weilenmann et al. (2009)). Under this condition emissions will be higher, and if the atmospheric temperature decreases, cold start emissions will increase regardless of whether the catalyst has reached its optimum temperature for functioning (Boulter and Barlow, 2009). VEIN also considers cold start emissions using the approach outlined in Copert (Ntziachristos and Samaras, 2016), as shown in equation (5.3).

$$EC_{i,j,k,l,m} = \beta_j \cdot F_{i,j,k,l} \cdot L_i \cdot EF(V_{i,l})_{j,k,m} \cdot DF_{j,k} \cdot (EF_{\text{cold}}(ta_n, V_{i,l})_{j,k,m} - 1) \quad (5.3)$$

This approach adds two terms to Eq. 5.1. The first term $EF_{\text{cold}}(ta_n, V_{i,l})_{j,k,m} - 1$ is the emission factors for cold start conditions at each street link i , vehicle category from composition k , hour l and pollutant m and monthly average temperature n . (Ntziachristos and Samaras, 2016) suggest using monthly average temperature. This is an important aspect that will be reviewed in future versions of VEIN.

The second term β_j is defined as the fraction of mileage driven with a cold engine/catalyst (Ntziachristos and Samaras, 2016). The VEIN model incorporates a dataset of cold starts recorded during the implementation of the International Vehicle Emissions (IVE) model (Davis et al., 2005) in São Paulo (Lents et al., 2004), which provides the hourly mileage driven with cold start conditions.

The function for cold start emission factors is and its arguments are:

```
args
```

where:

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

```
ef_ldv_cold
ef_ldv_cold
ef_ldv_cold
ef_ldv_cold
```

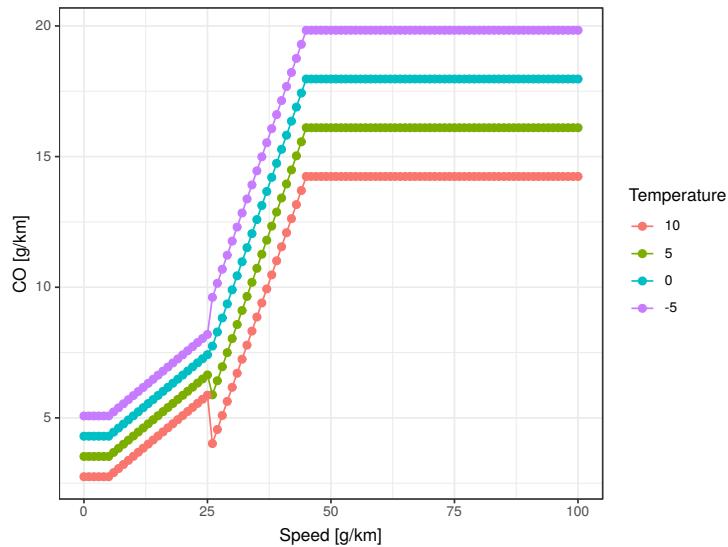


FIGURE 5.5: Cold Emission factor of PC CO (g/km)

There is another function called which will be deprecated soon.

5.5 Deterioration

By default, VEIN includes deterioration factors from Copert (Ntziachristos and Samaras, 2016). However, it is possible to include other sources, such as from (Corvalán and Vargas, 2003) or include deteriorated local emission factors. This aspect is faced in VEIN with the function `.det`. This function returns deteriorated factors, as numeric vectors, which must be multiplied with the emission factors of vehicles with `o` mileage.

The arguments of this function are:

```
args
```

where:

- : Pollutant.
- : Size of engine in cc.
- : Euro standard: “PRE”, “I”, “II”, “III”, “III”, “IV”, “V”, “VI”.
- : mileage in km. VEIN includes a dataset of mileage function named “fkm”.

```
data
data
cumsum$KM_PC_E25:
$ ==
subset $ %in% c  $
subset $ %in% c $ 
emis_det
:length
emis_det
length + :
$ 
$ * c  rep
```

```
plot :
```

```
lines :
```

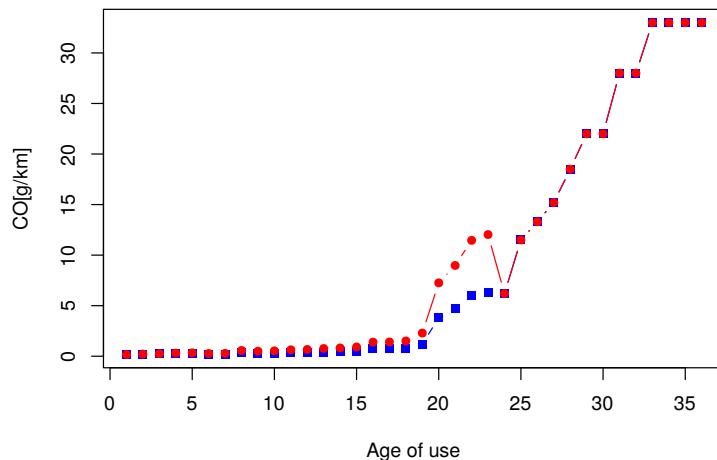


FIGURE 5.6: Deteriorated (red) and not-deteriorated (blue) EF CO

5.6 Evaporative emissions ef_evap

Evaporative emissions are important sources of hydrocarbons and these emissions are produced by vaporization of fuel due to variations in ambient temperatures (Mellios and Ntziachristos, 2016, Andrade et al. (2017)). There are three types of emissions: **diurnal emissions**, due to increases in atmospheric temperature, which lead to thermal expansion of vapor fuel inside the tank; **running losses**, when the fuel evaporates inside the tank due to normal operation of the vehicle; and **hot soak** emissions, which occur when the hot engine is turned off. These methods implemented in VEIN were sourced from the evaporative emissions methods of Copert Tier 2 (Mellios and Ntziachristos, 2016). In VEIN it is also possible to use local emission factors. Here I'm showing both approaches:

a) Tier 2 copert approach

$$EV_{j,k} = \sum_s D_s \cdot \sum_j F_j \cdot (HS_{j,k} + de_{j,k} + RL_{j,k}) \quad (5.4)$$

where EV_j are the volatile organic compounds (VOC) evaporative emissions due to each type of vehicle j . D_s is the “seasonal days” or number of days when the mean monthly temperature is within a determined range: $[-5^\circ, 10^\circ\text{C}]$, $[0^\circ, 15^\circ\text{C}]$, $[10^\circ, 25^\circ\text{C}]$ and $[20^\circ, 35^\circ\text{C}]$. $F_{j,k}$ is the number of vehicles according to the same type j and age of use k . $HS_{j,k}$, $de_{j,k}$ and $RL_{j,k}$ are average hot/warm soak, diurnal and running losses evaporative emissions ($\text{g} \cdot \text{day}^{-1}$), respectively, according to the vehicle type j and age of use k . $HS_{j,k}$ and $RL_{j,k}$ are obtained using equations also sourced from (Mellios and Ntziachristos, 2016):

$$HS_{j,k} = x_{j,k} \cdot (c \cdot (p \cdot e_{shc} + (1 - p) \cdot e_{swc}) + (1 - c) \cdot e_{shfi}) \quad (5.5)$$

where: x is the number of trips per day for the vehicular type j and age of use k . c is the fraction of vehicles with fuel return systems. p is the fraction of trips finished with hot engine, for example, an engine that has reached its normal operating temperature and the catalyst has reached its light-off temperature (Ntziachristos and Samaras, 2016). The light-off temperature is the temperature at the point when catalytic reactions occur inside a catalytic converter. e_{shc} and e_{swc} are average hot-soak and warm-soak emission factors for gasoline vehicles with carburettor or fuel return systems ($\text{g} \cdot \text{parking}^{-1}$). e_{shfi} is the average hot-soak emission factors for gasoline vehicles equipped with fuel injection and non-return fuel systems ($\text{g} \cdot \text{parking}^{-1}$).

The Eq. (5.6) shows the procedure.

$$RL_{j,k} = x_{j,k} \cdot (c \cdot (p \cdot e_{rhc} + (1 - p) \cdot e_{rwc}) + (1 - c) \cdot e_{rhfi}) \quad (5.6)$$

where: x and p have the same meanings of Eq. (5.5). e_{rhc} and e_{rwc} are average hot and warm running losses emission factors for gasoline vehicles with carburettor or fuel return systems ($\text{g} \cdot \text{trip}^{-1}$) e_{rhfi} are average hot

running losses emission factors for gasoline vehicles equipped with fuel injection and non-return fuel systems ($\text{g} \cdot \text{trip}^{-1}$).

It is recommended to estimate the number of trips per day (Mellios and Ntziachristos, 2016), x , as the division between the mileage and 365 times the length of trip: $x = \frac{\text{mileage}_j}{(365 \cdot \text{ltrip})}$.

However, the mileage of a vehicle is not constant throughout the years. Therefore, VEIN incorporates a dataset of equations to estimate mileage of different types of vehicles by age of use (Bruni and Bales, 2013).

The function for evaporative emission factors is and its arguments are :

args

where: - : Name of evaporative emission factor as $eshotc$ ($\text{g} \cdot \text{proced}^{-1}$): mean hot-soak with carburetor, $eswarmc$ ($\text{g} \cdot \text{proced}^{-1}$): mean cold and warm-soak with carburetor, $eshotfi$: mean hot-soak with fuel injection, $erhotc$ ($\text{g} \cdot \text{trip}^{-1}$): mean hot running losses with carburetor, $erwarmc$ ($\text{g} \cdot \text{trip}^{-1}$) mean cold and warm running losses, $erhotfi$ ($\text{g} \cdot \text{trip}^{-1}$) mean hot running losses with fuel injection. - : Type of vehicles, “PC”, “Motorcycles”, “Motorcycles_2S” and “Moped” - : Size of engine in cc. PC “ ≤ 1400 ”, “1400_2000” and “2000” Motorcycles_2S: “ ≤ 50 ”. Motorcycles: “ > 50 ”, “ < 250 ”, “ 250_750 ” and “ > 750 ” - : Average daily temperature variation: “ -5_{-10} ”, “ 0_{-15} ”, “ 10_{-25} ” and “ 20_{-35} ” - : Size of canister: “no” meaning no canister, “small”, “medium” and “large” - : multiplication factor - : when TRUE shows row of table with respective emission factor.

Example:

```
library
library
library
c
sapply: function
ef_evap
```

```
data.frame      c
source
ggplot aes      +
  geom_bar     cpt      +
  labs         +
  theme_black
```

b) Alternative approach

The evaporative emission factors mentioned before are suitable for top-down estimations. That approach also requires knowledge of the ‘procedure’ and ‘trips’. In this section it is introduced an alternative which consists in transform the emission factors into $g \cdot km^{-1}$ and then do a bottom-up estimation.

In order to do that, we need to follow some assumptions. First, we consider that the number of procedures is equal to trips. Then we estimate how many trips per day are produced each day and then we estimate how many kilometers are driven each day and convert these to emission factors as shown on Eq. (5.7).

$$EF_{ev}(\frac{g}{km}) = EF_{(ev)}(\frac{g}{trip}) \cdot A(\frac{trips}{day}) \cdot B(\frac{km}{day})^{-1} \quad (5.7)$$

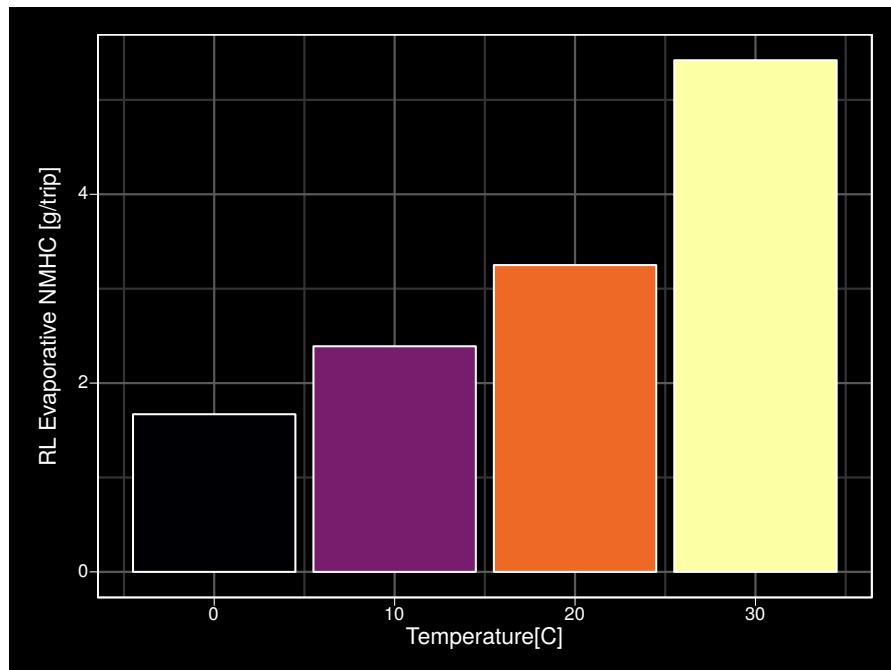


FIGURE 5.7: Running losses evaporative emission actors (g/trip)

Clearly, we need to know $A\left(\frac{\text{trips}}{\text{day}}\right)$ and $B\left(\frac{\text{km}}{\text{day}}\right)^{-1}$. Diurnal evaporative emissions factors are expressed as $\left(\frac{\text{g}}{\text{day}}\right)$. To convert to $\left(\frac{\text{g}}{\text{km}}\right)$ we only need the kilometers driven each day $\frac{\text{km}}{\text{day}}^{-1}$.

Now, it will be shown an example with the evaporative emission factors from CETESB (2015). These emission factors are based on the methodology Tier 2 from -@Mellios and Ntziachristos (2016). If we consider evaporative emission factors of Passenger Cars using Gasoline (blended with 25% of Ethanol) (CETESB, 2015) and 4.6 $\left(\frac{\text{trips}}{\text{day}}\right)$ (Ibarra-Espinosa, 2017).

The variable ed_20_35 are diurnal evaporative emission factors in $g \cdot day^{-1}$ and eshot_20_35 and erhot_20_35 $g \cdot trip^{-1}$.

```
library
data
$ $ / $KM_PC_E25:/ 
$ $ * / $KM_PC_E25:/ 
$ $ * / $KM_PC_E25:/ 
plot : $ 

lines : $ 

lines : $
```

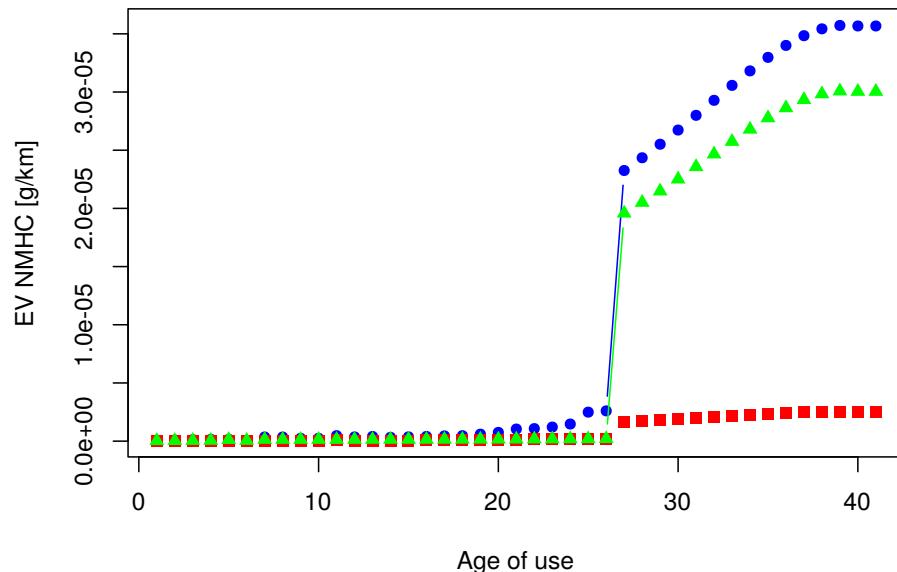


FIGURE 5.8: Hot Soak (blue), Diurnal (green) and Running Losses (red) EF (g/km)

5.7 Emission factors of wear `ef_wear`

Wear emissions are very important because they contribute with an important fraction of PM from tyre and break wear. VEIN includes these functions from Ntziachristos and Boulter (2009) which include wear emission factors form tyre, break and road abrasion.

The arguments are:

args

- : Character; type of wear: “tyre”, “break” and “road”
- : Character; type of vehicle: “2W”, “PC”, “LCV”, ’HDV”
- : Character; pollutant: “TSP”, “PM₁₀”, “PM_{2.5}”, “PM₁” and “PM_{0.1}”
- : List of speeds
- : Load of the HDV
- : Number of axle of the HDV

library

5.8 Emission factors from tunnel studies

With VEIN you can choose three type of emissions factors, speed functions, local emission factors or scaled factors. In this section I will show how to expand the use of local emission factors considering tunnel emission factors (Pérez-Martínez et al., 2014), from the International Vehicle Emissions (IVE) Model (Davis et al., 2005) applied in Colombia (González et al., 2017).

TABLE 5.2: Emission factors from tunnels in São Paulo 2014

	CO	NOx
LDV	5.8 [g/km]	0.3 [g/km]
HDV	3.6 [g/km]	9.2 [g/km]

Pérez-Martinez et al. (2014) measured air pollutant concentrations in two tunnels in São Paulo city, one with predominant use of LDV vehicles and another with more HDV. The resulting emission factors for the fleet are shown on Table 5.2.

If we want to compare these with the CETESB (2015) emission factors, we would have to weight these factors with the fleet. The weight mean emission factor of CO and Passenger Cars (PC) is $1.40 \text{ g} \cdot \text{km}^{-1}$ and for Light Trucks (LT) is $0.60 \text{ g} \cdot \text{km}^{-1}$. In the case of NOx, Passenger Cars (PC) is $0.16 \text{ g} \cdot \text{km}^{-1}$ and for Light Trucks (LT) is $3.24 \text{ g} \cdot \text{km}^{-1}$. The emission factors from the example are for vehicles with 0 kilometers without deterioration. Anyways, tunnel emission factors are higher than the weighted emission factors from CETESB.

```
library
data
$ ==
$ ==
$ ==
$ ==
age_ldv
age_hdv
weighted.mean
```

```
weighted.mean
```

weighted.mean

weighted.mean

If the user wants to use tunnel emission factors from Pérez-Martinez et al. (2014) in VEIN, the user would have to use the same value for all respective categories of the vehicular composition. For instance, after running , include tunnel emission factors in each .

5.9 Emission factors from International Vehicle Emissions (IVE)

The IVE models applies the vehicle specific power (VSP) methodology to predict emissions (Jimenez et al., 1999). The formula is shown on Eq. (5.8).

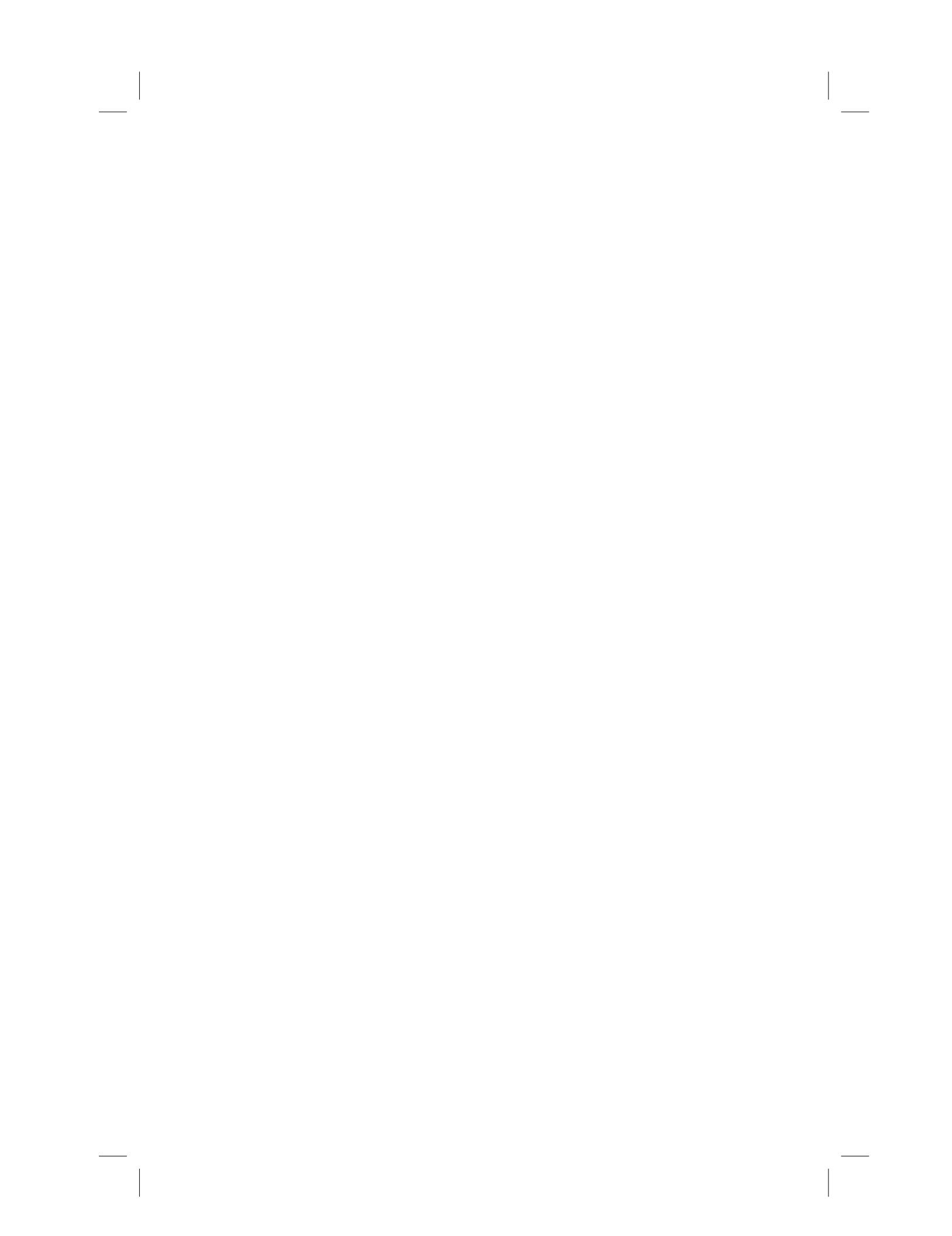
$$VSP = v \cdot (1.1 \cdot a + 9.81(\text{atan}(\sin(\text{grade}))) + 0.132) + 0.000302v^3 \quad (5.8)$$

where: v is velocity ($m \cdot s^{-1}$), a is the second by second acceleration ($m \cdot s^{-2}$) and grade the second by second change in height defined as $\frac{(h_{t=0} - h_{t=-1})}{v}$; being h as altitude m . This method requires that the user measures the activity Global Position System (GPS) recordings second by second and/or emissions measurements with Portable Emissions Measurement Systems (PEMS).

González et al. (2017) measure the vehicular activity in the city of Manizales, Colombia. In this case, there were no PEMS measurements, but IVE allows to produce emission factors from the GPS recordings based on Mobile EPA emissions model(Arbor, 2003, Davis et al. (2005)).

The resulting emission factors vary for each type of vehicle and includes

several pollutants. If the user wants to use IVE emission factors in VEIN, he/she would have to add them in each , after running .



6

Estimation of emissions

The emissions estimation process was shown on Eq. (5.1) on chapter 5. In this equation, it was shown that how the emissions are obtained by multiplying the traffic flow with the distance that each car travels (length of the road if it is a road network), and emissions and deterioration factors.

In this chapter I will show how VEIN interprets this and other emission equations to estimate emissions. At the end of each section The Fig. 6.1 shows a diagram with the estimation process. The pink boxes shows VEIN classes, the circle data and the grey boxes functions. The object is a class consisting in a matrix of vehicles with units $1 \cdot h^{-1}$, as shown on chapter 4, however, *the units in is not a requirement for estimating emissions*. The object of class is a matrix of speeds with the number of columns as the calculated speeds at different hours at each row. This object can be present or not, depending on the type of emission factor considered, for instant, emission factors not depending on speed. The object class of are list of functions depending on speed as $f(V)$. Each function can depend on speed or not as shown on chapter 5, section 5.2. Lastly, the object lkm is the length of the road expressed in lkm .

The function reads the inputs and estimates the emissions producing an object with class which has 4 dimensions, number of streets, ages of use of vehicle, number of hours and number of days.

The estimation process is done with the emissions functions. The emissions estimations functions are:

- for hot emissions,
- for cold start emissions,
- for evaporative emissions,
- for resuspension emissions on paved roads, and
- for wear of brakes, tyres and roads.

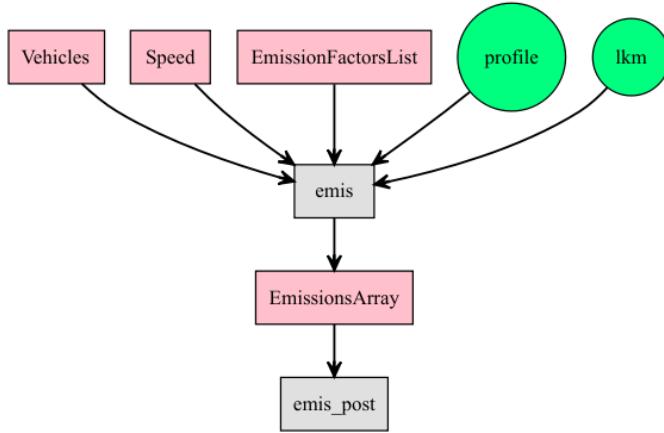


FIGURE 6.1: Emissions estimation process with VEIN

These functions perform internal checks for ensuring that the length lkm has the unit of *km*. This means that, if has units different than *km*, the function will not run. These function runs internally which are faster written in C. Despite that these functions are fast, i will try to make them faster with some RCPP implementation in future.

6.1 The `emis` function

The arguments of are:

`args::`

- : “Vehicles” data-frame or list of “Vehicles” data-frame. Each data-frame

has number of columns matching the age distribution of that type of vehicle. The number of rows is equal to the number of streets link.

- : Length of each link that must in km . As consequence, emission factors must be in $g \cdot km^{-1}$.
- : “EmissionFactorsList”. A list of emission factors as speed functions. Each element has the form $f(V)$. The implicit unit is $g \cdot km^{-1}$.
- : “Speed” object. A Speed data-frame with number of columns as hours.
- : Age of oldest vehicles of the vehicles veh. The information of this argument can be obtained from the argument Therefore, this argument will be deprecated.
- : Numerical or dataframe or matrix with nrow equal to the hours and cols to each day. This is traffic data normalized to the hour of the input traffic data.
- : Number of considered hours in estimation. As this information can be derived form the argument , the default value is the number of rows of the matrix profile.
- : Number of considered days in estimation. As this information can be derived form the argument , the default value is the number of coluns of the matrix profile.
- : 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)

6.2 The `emis_cold` function

The arguments of are:

```
args::
```

The arguments are similar with `emis_cold_start` with the difference that it is added two arguments:

- `fun`: List of functions of cold start emission factors of vehicular categories. Technically, each function is also of the form $f(V)$.
- `dist`: Dataframe with the hourly cold-start distribution to each day of the period. Number of rows are hours and columns are days. It represents the fraction of mileage driven under cold-start conditions.

The equation for estimating cold-start emissions is on Eq. (5.3).

6.3 The `emis_evap` function

The estimation of evaporative emissions applies Tier 2 from Mellios and Ntziachristos (2016). The approach followed in VEIN was very simply, consisting in only adding elements of a data-frame.

args::

- `vehs`: this are the number of vehicles at each age of use and for each street. It is a `tbl_df` object.
- `name`: Character indicating the name of the vehicle.
- `size`: Character indicating the size of the vehicle.
- `fuel`: Character indicating the fuel of the vehicle.
- `age`: Numeric vector with the age distribution, for instance, 1:40 for vehicles between 1 and 40 years of use.
- `days`: Number of days of your peridiof of study with monthly average temper-

ature between 20 and 35 celcius degrees. If the period of time is a year, this is the annual number of days under that condition.

- : Number of days of your perdiol of study with monthly average temperature between 10 and 25 celcius degrees.
- : Number of days of your perdiol of study with monthly average temperature between 0 and 15 celcius degrees.
- : Number of days of your perdiol of study with monthly average temperature between -5 and 10 celcius degrees.
- : Number of average daily hot-soak evaporative emissions for days with temperature between 20 and 35 celcius degrees
- : Number of average daily hot-soak evaporative emissions for days with temperature between 10 and 25 celcius degrees
- : Number of average daily hot-soak evaporative emissions for days with temperature between 0 and 15 celcius degrees
- : Number of average daily hot-soak evaporative emissions for days with temperature between -5 and 10 celcius degrees
- : Number of average daily running losses evaporative emissions for days with temperature between 20 and 35 celcius degrees
- : Number of average daily running losses evaporative emissions for days with temperature between 10 and 25 celcius degrees
- : Number of average daily running losses evaporative emissions for days with temperature between 0 and 15 celcius degrees
- : Number of average daily running losses evaporative emissions for days with temperature between -5 and 10 celcius degrees
- : Number of average daily diurnal evaporative emissions for days with temperature between 20 and 35 celcius degrees
- : Number of average daily diurnal evaporative emissions for days with temperature between 10 and 25 celcius degrees
- : Number of average daily diurnal evaporative emissions for days with temperature between 0 and 15 celcius degrees
- : Number of average daily diurnal evaporative emissions for days with temperature between -5 and 10 celcius degrees

6.4 The `emis_paved` function

Evaporative emissions are emissions of resuspended dust due to traffic circulating over paved or non-paved roads. These emissions can be an important source of mass particulate matter, and several cities programs that aims to improve air quality by diminishing these emissions (Amato et al., 2010). However, these emissions are not considered by in the European emissions guidelines, where their focus is on primary particles and not those resulting from the resuspension of previously deposited material (Ntziachristos and Boulter, 2009).

The method adopted in VEIN comes from the USEPA (USA-EPA, 2016) which presents equations for annual, daily and hourly estimations:

$$EF_{paved} = k \cdot sL^{0.91} \cdot W^{1.02} \quad (6.1)$$

Where

- EF_{paved} is the emission factor of particulate matter.
- k particle size splitter.
- sL road surface silt loading $g \cdot m^{-2}$.
- W average weight.

Equation (6.1) can be extrapolated to consider natural mitigation of rainy periods of time with daily basis:

$$EF_{annual} = EF_{paved} \cdot \left(1 - \frac{P}{4 \cdot N}\right) \quad (6.2)$$

Where

- EF_{annual} annual or long term emission factor.
- P number of days with accumulated rain above 0.254 mm.
- N total number of days.

or hourly

$$EF_{hourly} = EF_{paved} \cdot \left(1 - \frac{1.2 \cdot P}{N}\right) \quad (6.3)$$

Where

- EF_{annual} annual or long term emission factor.
- P number of hours with accumulated rain above 0.254 mm.
- N total number of hours. For instance, 8760 for 1 year.

The function has the arguments:

args::

Where,

- : It is an array with dimensions number of streets x hours of day x days of week. - : Length of each link. - : $K_{PM30} = 3.23$, $K_{PM15} = 0.77$, $K_{PM10} = 0.62$ and $K_{PM2.5} = 0.15$. - : Silt loading (g/m²) for roads with ADT <= 500.
- : Silt loading (g/m²) for roads with ADT > 500 and <= 5000. - : Silt loading (g/m²) for roads with ADT > 5000 and <= 1000. - : Silt loading (g/m²) for roads with ADT > 10000. - : array of dimensions of veh. It consists in the hourly averaged weight of traffic fleet in each road.

For instance, Lets create an array of vehicles. The funcion calculates the amount pf vehicles at all hours. Lets use the asumming no buses. Let's calculate the total traffic for only 24 hours with the same profile. For simplicity, let's assume that all vehicle are expressed in *vehicle equivalent* and it is a dry period.

```
library
data
data
  vkm$* matrix$
  vkm$* matrix$
  vkm$* matrix$
  vkm$ matrix$
  +
dim
```

is the average hourly fleet weight. It has the same dimensions of . Let's assume an individual weight of PC is 1, LCV 1.5, HGV = 10 and MC 0.5. This example is a simplification, should be calculated with full characterization of the fleet.

```
vkm$* $ *
vkm$* $ *
vkm$* $ *
vkm$ $ *
+ + + /
is.na
dim
```

and now we estimate the emissions with the default values from PM10.

```
emis_paved $ 
:::
```

6.5 The `emis_wear` function

Wear emissions can be very important, in fact, these are second most important source of particulate matter in Europe (Lükewille et al., 2017). These methods in VEIN comes from Ntziachristos and Boulter (2009) which include wear of “tyre”, “break” and “road”.

The arguments of the function are:

```
args::
```

Where,

- : Object of class “Vehicles”.
- : Length of the road.
- : list of emission factor functions class “EmissionFactorsList”, length equals to hours.
- : Age of oldest vehicles for that category
- : Dataframe or matrix with nrows equal to hours and ncol as days of the week
- : Number of considered hours in estimation, with default value as number of rows of profile.
- : Number of considered days in estimation, with default value as number of columns of profile.

Let's estimate wear emissions!

First, we get the emission factor with the function:

```
library  
data  
data  
$  
temp_fact$+$  
netspeed $ $ $ $  
ef_wear
```

Then we estimate emissions

```
emis_wear age_ldv$  
$
```

```
emis_post  
:::
```

7

Post estimation with

Once the emissions are estimated we obtained objects, which as mentioned before, they are arrays with the dimensions streets x age distribution of vehicles x hours x days.

Now the function reads the and convert it to data-frames that are easier to manage.

The arguments of are:

args::

Where,

- : obtained with the functions. It is an array of emissions 4d: streets x category of vehicles x hours x days or 3d: streets x category of vehicles x hours
- : Character indicating the type of vehicle , for instance: “PC”.
- : Character indicating the size or weight, for instance: “<1400”, “1400<cc<2000”, “GLP”, “Diesel”.
- : Character indicating fuel, for instance: “Gasoline”, “E_25”, “GLP”, “Diesel”.
- : Character indicating the pollutant, for instance “CO”, “NOx”, etc.
- : Character indicating type of output, “veh” for total vehicular category , “streets_narrow” or “streets_wide”. This is the most important argument of this function. “streets_wide” returns a datafram 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

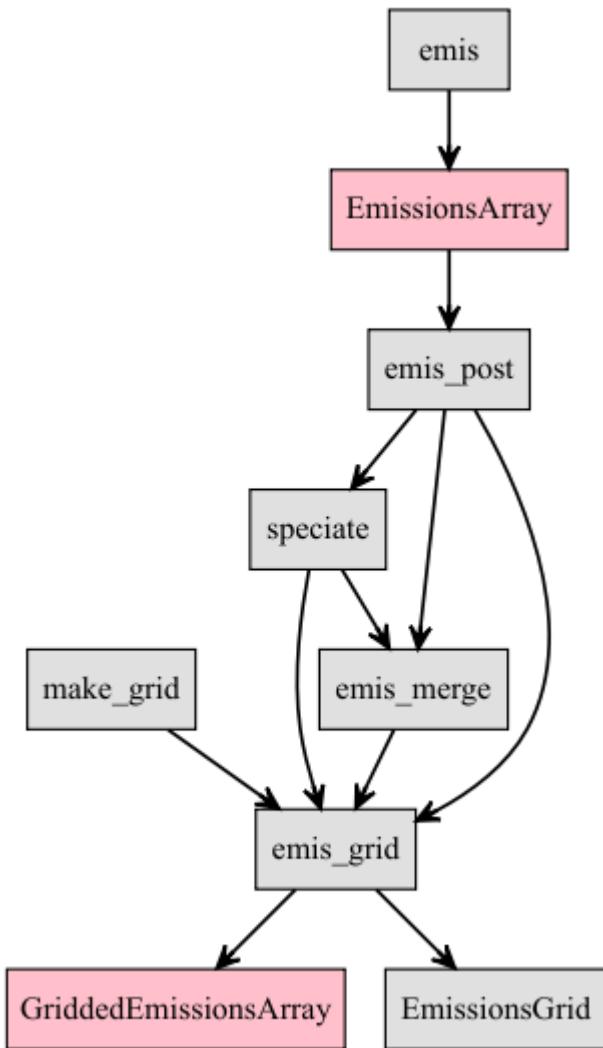


FIGURE 7.1: Post-emissions

Lets create some objects

7.1 Emissions by street

From previous chapters we worked with , and estimate emissions generating . Now let's create an object.

```
library  
library  
library  
library  
data  
data  
data  
age_ldv $
```

```
temp_fact$ $  
  
temp_fact$ $  
netspeed + $$$$  
EmissionFactorsList$==  
emis $  
$
```

Now that we have our E_CO, we can process it with the function . The emissions by street are obtained with the argument = “streets_wide”:

```
emis_post  
:::
```

In the version 0.3.17 it was added the capability of returning an spatial object class ‘sf’ with ‘LINESTRING’:

```
emis_post  
class  
  
ggplot + geom_sf aes as.numeric +  
scale_color_gradientn revcpt + theme_bw +  
ggtitle
```

Now, if we estimate the NOx emissions of Light Trucks we will a slight different pattern of emissions:

```
age_hdv $
```

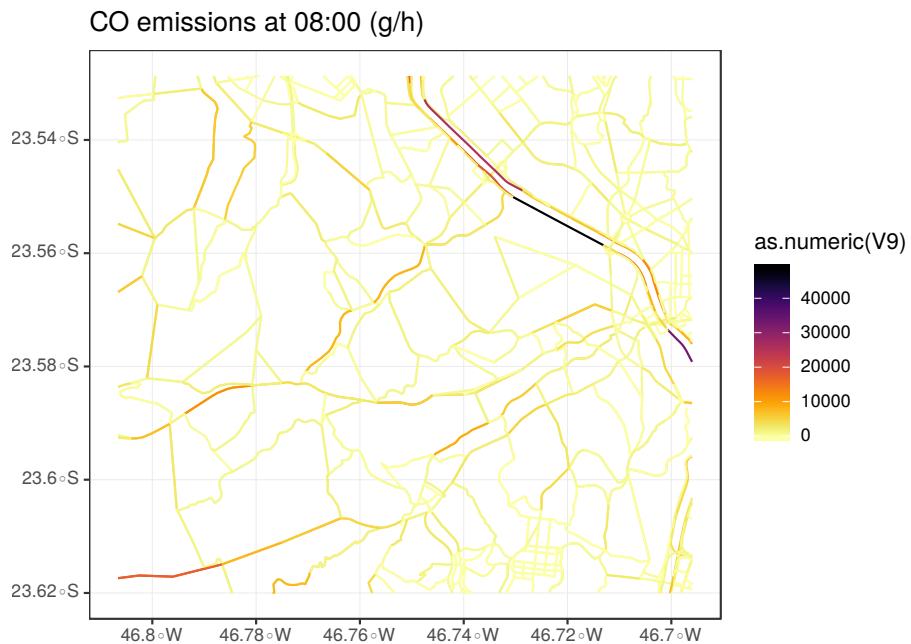


FIGURE 7.2: CO emisions of PC (g/h)

```
EmissionFactorsList$==  
emis      $  
          $
```

```
emis_post  
ggplot + geom_sf aes as.numeric +  
scale_color_gradientn revcpt + theme_bw+  
ggtitle
```

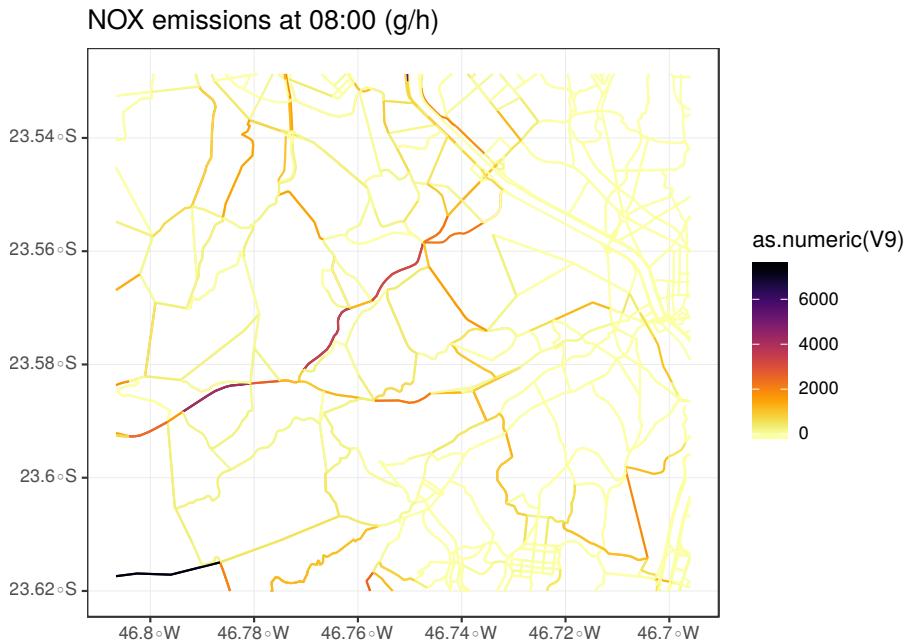


FIGURE 7.3: NOx emisions of LT (g/h)

7.2 Total emissions in data-frames

In order to obtain the total emissions in data-frame, the same function is used, but in this case, with the argument = “veh”. We also need to add arguments of for the type of vehicle, the size or gross weight, and . The argument is not required in this case.

```
emis_post
head
```

The resulting object is a data-frame with the name of the input array, then it has the g/h. Also, the name of the vehicle, the size, fuel, pollutant, age and hour. This is interesting because we can now what are the most pollutant vehicle and at which hour this emissions occurs.

```
$ == max$
```

The highest emissions by hour are of Light Trucks of 17 years of use at hour 11 on local time. However, it is possible to get more interesting insights of the data. The data can be aggregated by age, hour or plot all together as shown on Fig. 7.6. The highest emissions occur between 17 and 27 years of use. This figure also shows the effect of emissions standards introduced in different years.

```
aggregate$ list$  
plot
```

```
library  
library  
aggregate$ list$  
names c  
ggplot aes as.numeric as.numeric +  
geom_bar +  
scale_fill_gradientn cpt + theme_black +  
labs
```

```
library  
library  
ggplot aes as.numeric +  
geom_bar +
```

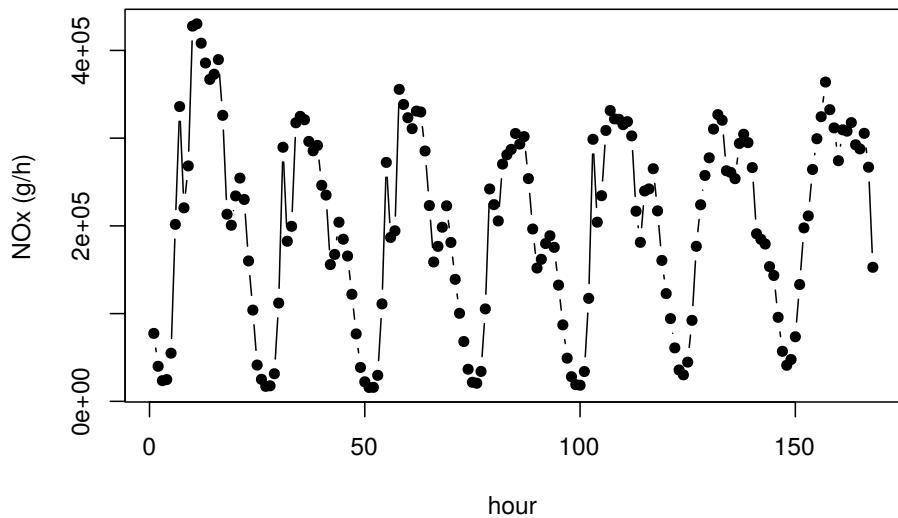


FIGURE 7.4: NOx emissions of LT by hour

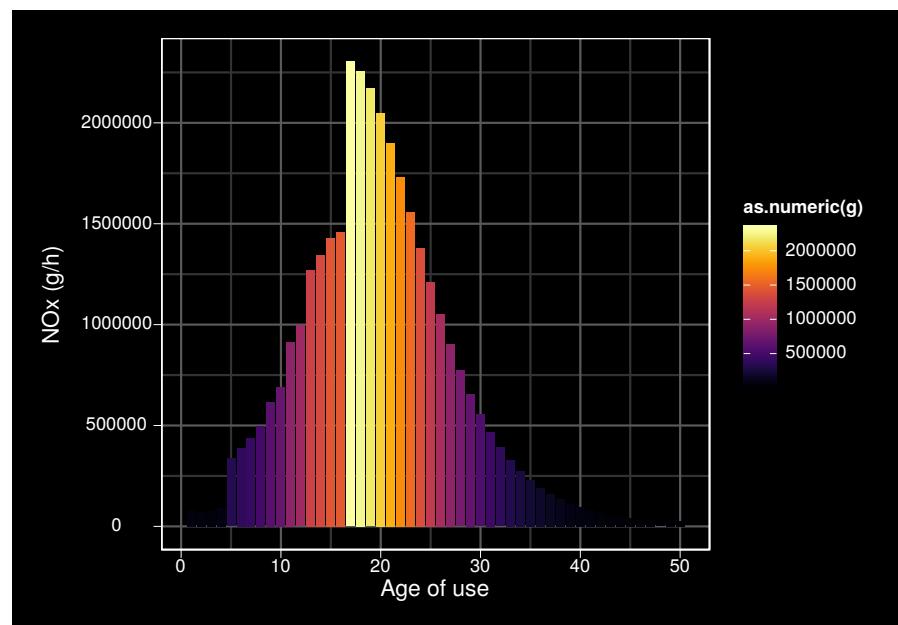


FIGURE 7.5: NOx emissions of LT by age

```
scale_fill_gradientn cpt + theme_black +  
  labs
```

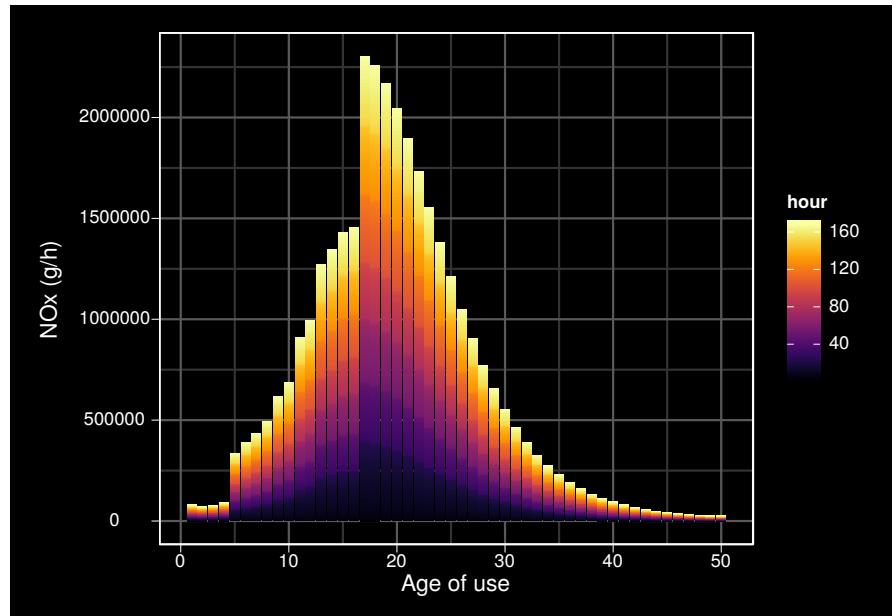


FIGURE 7.6: NOx emissions of LT

7.3 Merging emissions

The function was designed to store the emissions of each type of vehicle in each directory. Hence, it was necessary to create a function that read and merge the emissions that are in each directory, returning a data-frame or a spatial feature data-frame. The function is `. As it can be seen on Fig. 7.1, comes after or .`

```
args::
```

Where,

- : Character. Pollutant.
- : Character. Word to search the emissions names, “STREETS”, “DF” or whatever name. It is important to include the extension ‘.rds’
- : Logical. If true, emis_merge will read the street emissions created with emis_post by “streets_wide”, returning an object with class ‘sf’. If false, it will read the emissions data-frame and rbind them.
- : ‘Spatial feature’ or ‘SpatialLinesDataFrame’ with the streets. It is expected #‘that the number of rows is equal to the number of rows of street emissions. If #’ not, the function will stop.
- : Character. Path where emissions are located
- : coordinate reference system in numeric format from to transform/project spatial data using sf::st_transform

In order to see how this functions works, let’s run the example in the function. First, lets create a project, or in other words a directory named “YourCity”, into the temporary directory. In this part you can place your project wherever you want with the actual name of your city. Then run with the default configuration.

```
file.pathtempdir  
:::inventory
```

In my case, the files are in directory /tmp/RtmpUEeCGB/YourCity. Now, if you open the file file main.R, you will see:

```
junk@junk-pc:~/RtmpUEeCGB$ Rscript main.R
1 setwd('/tmp/RtmpUEeCGB/YourCity')
2 library(vetin)
3 # 1) Network #####
4 data(net)
5 net <- sf::st_as_sf(net)
6 net <- sf::st_transform(net, 31983)
7 saveRDS(net, 'network/net.rds')
8 ## Are you going to need Speed?
9 data(pc.profile)
10 pc_week <- temp_fact(net$ldv+net$hdv, pc.profile)
11 speed <- netspeed(pc_week, net$ps, net$ffs, net$capacity, net$lk, alpha = 1)
12 saveRDS(speed, 'network/speed.rds')
13 # 2) Traffic #####
14 source('traffic.R') # Edit traffic.R
15 # 3) Estimation #####
16 inputs <- list.files(path = 'est', pattern = 'input.R',
17                      recursive = TRUE, full.names = TRUE)
18 for (i in 1:length(inputs)){
19   print(inputs[i])
20   source(inputs[i])
21 }
22 # 4) Post-estimation #####
23 g <- make_grid(net, 1000)
24 source('post.R')
```

This script starts with `function`, Then the first section is the network, where user reads and prepares the traffic information. Then the second section runs a script which runs the age functions for the default vehicular composition of the `function`. The third section estimates the emissions by running each `input.R` file inside the directory `est`. The fourth section, creates a grid and run the script `post.R` is shown below:

```
1 # streets #####
2 CO <- emis_merge('CO', net = net)
3 saveRDS(CO, 'post/streets/CO.rds')
4
5 # grids #####
6 gCO <- emis_grid(CO, g)
7 print(plot(gCO['V1'], axes = TRUE)) # only an example
8
9 saveRDS(gCO, 'post/grids/gCO.rds')
10
11 # df #####
12 dfCO <- emis_merge('CO', what = 'DF.rds', FALSE)
13 saveRDS(dfCO, 'post/df/dfCO.rds')
14 aggregate(dfCO[, by = list(dfCO$veh), sum, na.rm = TRUE] # Only an example
```

The function shown in line 2 is designed to read all the files with names `CO` and the default argument is “`STREETS.rds`”, for instance, `PC_GASO-`

LINE_CO_STREETS.rds. Hence, this function reads all the files under that condition. The other argument is with default value of , meaning that the function now knows that the resulting object must be an spatial network, and the argument is already calling the net object part of the VEIN data, the function will merge all the objects summing the emissions of all vehicle street by street. The resulting object is a spatial “LINESTRING” class of . This function assumes that emissions files are data.frames and non tests have been made to see if this function would read work objects.

On the other hand, the line 12 also runs all the emissions files whose names includes the word CO, but also including the words “DF.rds”, which are the files from with argument equals to “veh”. In this case, it reads and merge the data-frames and prints the sum emissions by pollutant.

If the user source he file main.R, it will calculate the emissions and produce some plots.

```
sourcepaste0
```

7.4 Creating grids

Once we have processed our emissions array with and/or we have spatial emissions, we can then create grids and allocate our emissions into the grids. Lets check the function first:

```
args::
```

This function originally used an approach. However, now uses an approach for creating the grid. Actually, the arguments are pretty much the same with the exception that is focused in returning polygons

with coordinates at centroids of each cell. Therefore, the arguments height and polygon are deprecated. Lets create grid for our net.

```
library  
library  
data  
st_transformst_as_sf  
make_grid
```

```
class
```

```
plot$  
plot$
```

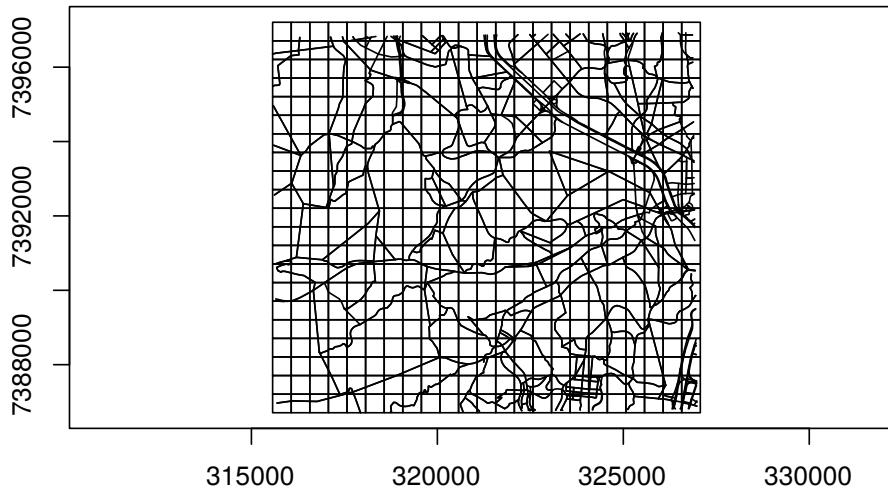


FIGURE 7.7: Grid with spacing of 500 mts

The class of is , which is converted to . Then, as the coordinate reference system of net is WGS84 with epsg 4326, we are transforming to UTM to

create a grid with grid spacing of 500 mts. The class of the grid object, is “sf data.frame”.

7.5 Gridding emissions with `emis_grid`

The function allocates emissions proportionally to each grid cell. The process is performed by intersection between geometries and the grid. It means that requires “sr” according with your location for the projection. It is assumed that spobj is a spatial* DataFrame or an “sf” with the pollutants in data. This function return an object class “sf”.

Before era, i tried to use which imports and took **ages**. Then I started to use QGIS Development Team (2017) (), and later the R package Muenchow et al. (2018) which takes 10 minutes for allocating emissions of the mega-city of São Paulo and a grid spacing of 1 km. *10 min is not bad*. However, when appeared and also, included the Spatial Indexes (), it changed the game. A new era started. It took **5.5 seconds**. I could not believe it. But there also some i could accelerate this process importing some aggregation functions. Now can aggregate big spatial extensions. I havent tried yet at continental scale, but I will try. Anyways, let's go see the arguments:

args::

Where,

- : A spatial dataframe of class “sp” or “sf”. When class is “sp” it is transformed to “sf”.
- : A grid with class “SpatialPolygonsDataFrame” or “sf”.
- : Spatial reference e.g: 31983. It is required if spobj and g are not projected. Please, see .
- : type of geometry: “lines” or “points”.

```
data  
@ data.frame  
make_grid //
```

```
emis_grid
```

```
head
```

```
plot
```

The example in this case, showed the allocation of traffic. The resulting object is a `grid` with the sum of the attributes inside each grid cell. This means that, when the input are street emissions at different hours, it will return gridded emissions at each hour:

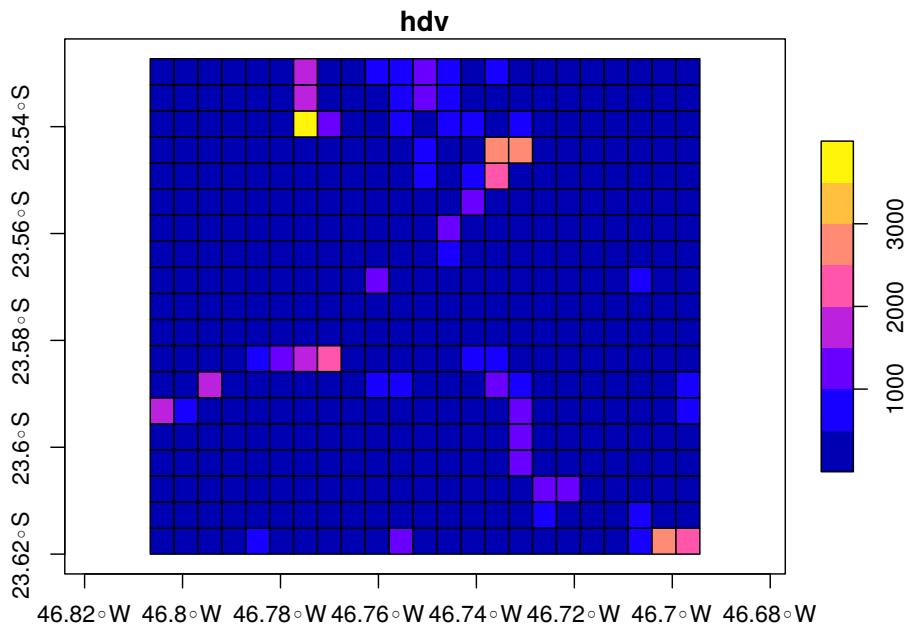


FIGURE 7.8: Spatial grid of HDV traffic 500 mts

```
emis_post  
emis_grid
```

```
plot
```

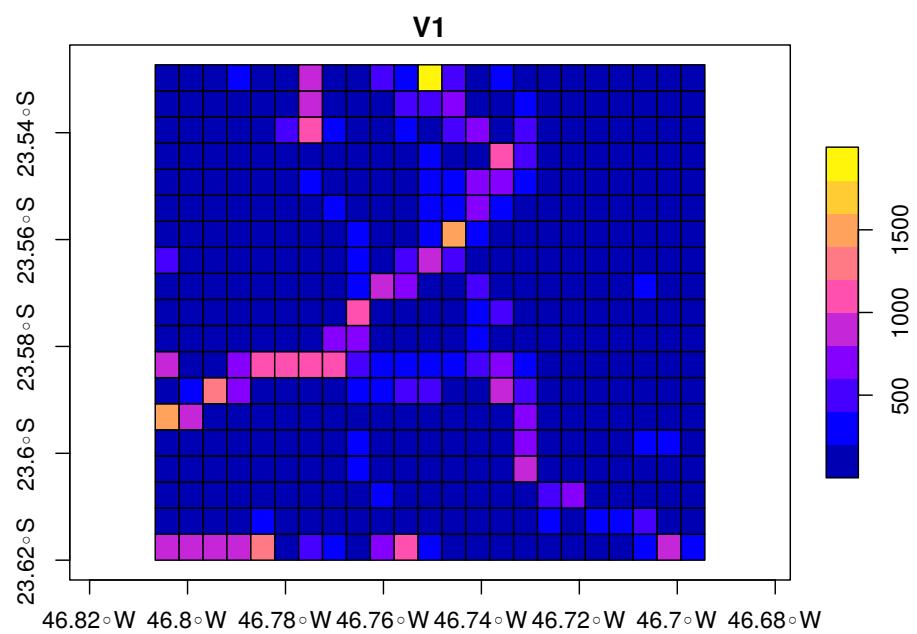


FIGURE 7.9: NOx emissions for 00:00 [g/h]



8

Speciation

The speciation for emissions is a very important part that deserves a only chapter on this book. VEIN initially covered speed functions emission factors of CO, HC, NOx, PM and Fuel Consumption (FC). But currently, the the pollutants covered are:

- **Criteria:** “CO”, “NOx”, “HC”, “PM”, “CH4”, “NMHC”, “CO2”, “SO2”, “Pb”, “FC” (Fuel Consumption).
- **Polycyclic Aromatic Hydrocarbons (PAH) and Persisten Organic Pollutants (POP):** “indeno(1,2,3-cd)pyrene”, “benzo(k)fluoranthene”, “benzo(b)fluoranthene”, “benzo(ghi)perylene”, “fluoranthene”, “benzo(a)pyrene”, “pyrene”, “perylene”, “anthanthrene”, “benzo(b)fluorene”, “benzo(e)pyrene”, “triphenylene”, “benzo(j)fluoranthene”, “dibenzo(a,j)anthacene”, “dibenzo(a,l)pyrene”, “3,6-dimethylphenanthrene”, “benzo(a)anthracene”, “acenaphthylene”, “acenaphthene”, “fluorene”, “chrysene”, “phenanthrene”, “naphthalene”, “anthracene”, “coronene”, “dibenzo(ah)anthracene”
- **Dioxins and Furans:** “PCDD”, “PCDF”, “PCB”.
- **Metals:** “As”, “Cd”, “Cr”, “Cu”, “Hg”, “Ni”, “Pb”, “Se”, “Zn”.
- **NMHC:**
- **ALKANES:** “ethane”, “propane”, “butane”, “isobutane”, “pentane”, “isopentane”, “hexane”, “heptane”, “octane”, “TWO_methylhexane”, “nonane”, “TWO_methylheptane”, “THREE_methylhexane”, “decane”, “THREE_methylheptane”, “alkanes_C10_C12”, “alkanes_C13”.
- **CYCLOALKANES:** “cycloalcanes”.
- **ALKENES:** “ethylene”, “propylene”, “propadiene”, “ONE_butene”, “isobutene”, “TWO_butene”, “ONE_3_butadiene”, “ONE_pentene”, “TWO_pentene”, “ONE_hexene”, “dimethylhexene”.
- **ALKYNES:** “ONE_butine”, “propyne”, “acetylene”.
- **ALDEHYDES:** “formaldehyde”, “acetaldehyde”, “acrolein”, “benzaldehyde”, “crotonaldehyde”, “methacrolein”, “butyraldehyde”, “isobutanalde-

- hyde”, “propionaldehyde”, “hexanal”, “i_valeraldehyde”, “valeraldehyde”, “o_tolualdehyde”, “m_tolualdehyde”, “p_tolualdehyde”.
- KETONES: “acetone”, “methylethylketone”.
 - AROMATICS: “toluene”, “ethylbenzene”, “m_p_xylene”, “o_xylene”, “ONE_2_3_trimethylbenzene”, “ONE_2_4_trimethylbenzene”, “ONE_3_5_trimethylbenzene”, “styrene”, “benzene”, “C9”, “C10”, “C13”.

Also, some Brazilian emission factors and speciations for WRF-Chem, mechanisms: “e_eth”, “e_hc3”, “e_hc5”, “e_hc8”, “e_ol2”, “e_olt”, “e_oli”, “e_iso”, “e_tol”, “e_xyl”, “e_c2h5oh”, “e_ald”, “e_hcho”, “e_ch3oh”, “e_ket”, “E_SO4i”, “E_SO4j”, “E_NO3i”, “E_NO3j”, “E_MP2.5i”, “E_MP2.5j”, “E_ORGi”, “E_ORGj”, “E_ECi”, “E_ECj”, “H₂O”.

By the end of this book, more emission factors will be added.

Because VEIN has the capabilities to produce species from this pollutants including volatile organic compounds (VOC) and particulate matter compounds.

The speciation of emissions in VEIN can be done with three ways, by selecting the pollutants in ef_speed* functions, other way explicit and the other implicit. The explicit way is by usint the function for the specific available speciation. The implicit way is by adding a percentage value in the argument in any of the functions of emission factors. The implicit way requires that the user must know the percentage of the species of pollutant.

8.1 Speed functions of VOC species

As mentioned before, the speed function emission factors, currently covers several pollutants. We firstly focus on som PAH and POP. They way to select them is just:

```
library  
c  
  
lapply:length function  
EmissionFactorsef_ldv_speed  
  
names
```

These pollutants comes from Ntziachristos and Samaras (2016) and in this case, are not dependent on speed. I used lapply to iterate in each pollutant at 10 km/h. Now I will show the same for dioxins and furans.

```
library  
c  
lapply:length function
```

```
EmissionFactorsef_ldv_speed
```

```
names
```

In the case of nmhc, the estimation is:

```
library  
c  
lapply:length function  
EmissionFactorsef_ldv_speed  
c  
  
names
```

Now the same example for one pollutant and several euro standards.

```
library  
c  
lapply:length function  
EmissionFactorsef_ldv_speed  
c  
  
names
```

8.2 Speciate function

The arguments of the function are:

args::

Where,

- : Emissions estimation
- : speciation: The speciations are: “bcom”, “tyre”, “break”, “road”, “iag”, “nox” and “nmhc”. ‘iag’ now includes a speciation for use of industrial and building paintings. “bcom” stands for black carbon and organic matter.
- : Type of vehicle: When spec is “bcom” or “nox” veh can be “PC”, “LCV”, “HDV” or “Motorcycle”. When spec is “iag” veh can take two values depending: when the speciation is for vehicles veh accepts “veh”, eu “Evaporative”, “Liquid” or “Exhaust” and fuel “G”, “E” or “D”, when the speciation is for painting, veh is “paint” fuel or eu can be “industrial” or “building” when spec is “nmhc”, veh can be “LDV” with fuel “G” or “D” and eu “PRE”, “I”, “II”, “III”, “IV”, “V”, or “VI”. when spec is “nmhc”, veh can be “HDV” with fuel “D” and eu “PRE”, “I”, “II”, “III”, “IV”, “V”, or “VI”. when spec is “nmhc” and fuel is “LPG”, veh and eu must be “ALL”
- : 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”. When spec is “nmhc” fuel can be G, D or LPG.

- : sEuro 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"
- : when TRUE shows row of table with respective speciation
- : when TRUE returns a list with number of elements of the list as the number species of pollutants

As shown before, there are currently 8 type of speciations. Now, i will show examples for most of them.

8.3 Black carbon and organic matter

Let's use the data inside the vein and estimate emissions. The vehicles considered are Light Trucks, assuming all HGV. We are using the 4 stage estimation with the first part arranging traffic data

```
library  
  
data  
data  
data  
age_hdv$
```

```
temp_fact$ $ +  
temp_fact$ $  
netspeed $ $ $ $
```

```
c rep  
    rep rep rep rep  
lapply: function  
ef_hdv_speed
```

```
emis      $  
          $
```

```
apply      c
```

```
rbindspeciate  
speciaterowSums :  
speciaterowSums :  
speciaterowSums :  
speciaterowSums :  
speciaterowSums :  
$ :  
aggregate$  list$  
aggregate$  list$  
$  
$
```

Now we have spatial objects of Back Carbon

```
##spplot  list  
rev:::cpt
```

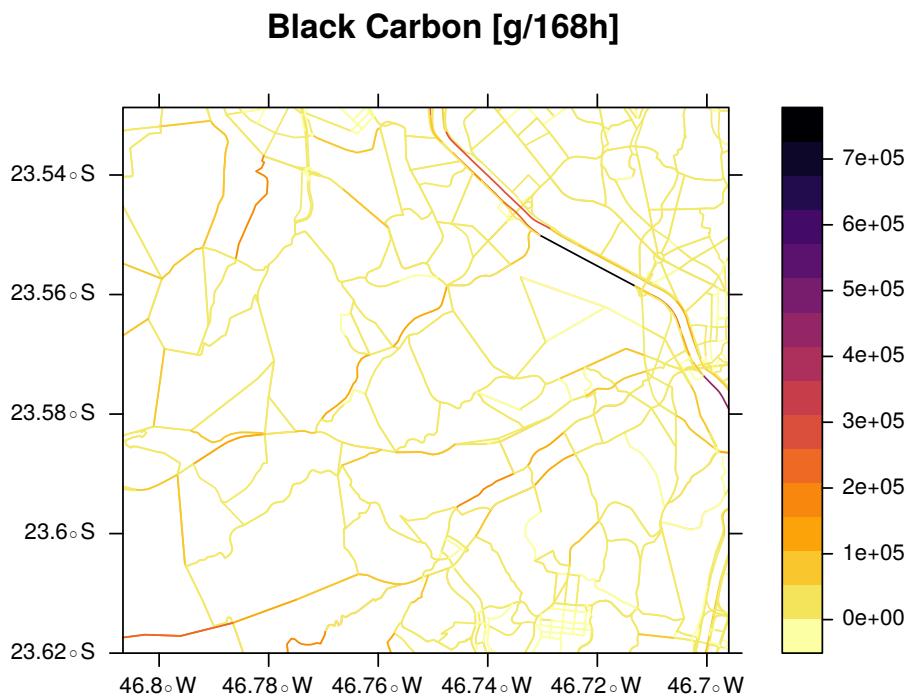


FIGURE 8.1: Black Carbon [g/168h]

and Organic Matter

```
##spplot  list  
rev:::cpt
```

If you are interested in speciate total emissions by age of use not spatial emissions, you could use with and then aggregate the emissions by standard. Then speciate PM by standard:

```
emis_post
```

```
$
```

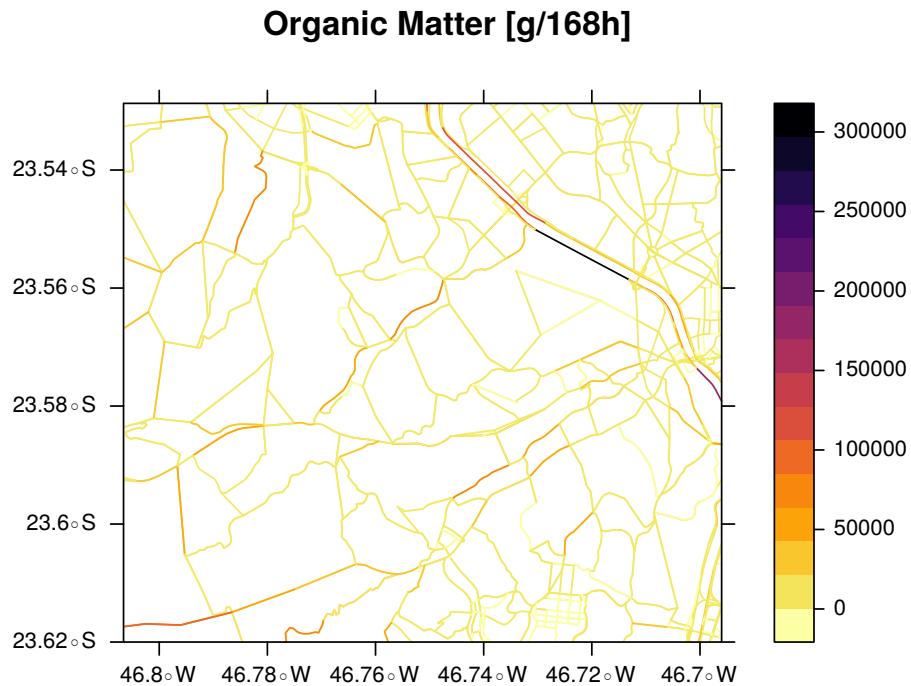


FIGURE 8.2: Organic Matter [g/168h]

TABLE 8.1: Black Carbon and Organic Matter [g/168h]

	I	II	III	IV	PRE
BC	6826527.37503195	5313757.92745243	3321183.01325297	155078.69338865	6700527.80930883
OM	2730610.95001278	1222164.32331406	498177.451987945	20160.2301405245	4690369.46651618

```

aggregate$  list$
names  c
as.data.framesapply: function
speciate$   $

names  $

```

And the resulting speciation is:

8.4 Tyre wear, breaks wear and road abrasion

Tyre, break and road abrasions comes from Ntziachristos and Boulter (2009). Tyres consist in a complex mixture of rubber, which after the use starts to degradate. In deed, when the driving cycle is more aggressive, higher is tyre mass emitted from tyre, with tyre wear emission factors higher in HDV than LDV. Break wear emits mass and with more aggressive driving, more emissions.

The input are wear emissions in Total Suspended Particles (TSP) ,as explained in section @ref(#ew). The speciation consists in 60% PM₁₀, 42% PM_{2.5}, 6% PM₁ and 4.8% PM_{0.1}. Now, a simple example for TP adspeciation:

```
library  
data  
data  
$  
temp_fact$+$  
netspeed $ $ $ $  
ef_wear  
emis_wear age_ldv$  
$
```

```
emis_wear    age_ldv$  
$
```

```
emis_post
```

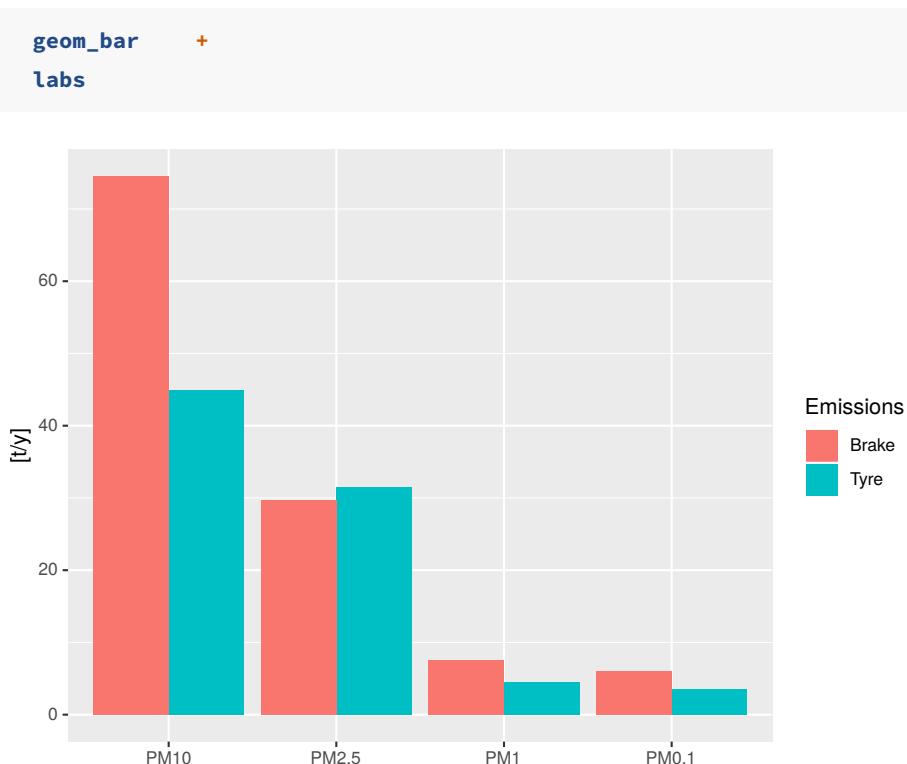
```
emis_post
```

The estimation covered 24 hours and for simplicity, we are assuming a year with 365 similar days and dividing by 1 million to have t/y. Hence, the speciation is:

```
library  
speciate unclass$  
speciate unclass$  
data.frame csapply sapply */  
$ factor names names  
$ crep rep  
ggplot aes +
```

TABLE 8.2: Ratio between break and tyre wear emissions

	x
PM10	1.6593167
PM2.5	0.9433433
PM1	1.6931803
PM0.1	1.6931803

**FIGURE 8.3:** Break and Tyre emissions (t/y)

It is interesting to see that break emissions are higher than tyre wear emissions, and the smaller the diameter, higher the difference, as shows:

The procedure can be used with `spatial` to obtain the spatial speciation.

8.5 NO and NO₂

The speciation of NO_X into NO and NO_2 depends on the type of vehicle, fuel and euro standard. The following example will consists in comparing a Gasoline passenger car and a Diesel truck.

```
library  
data  
data  
$  
$  
temp_fact$+$  
netspeed $ $ $ $  
c rep  
rep rep rep rep  
lapply: function  
ef_ldv_speed  
  
lapply: function  
ef_hdv_speed  
  
emis age_ldv$ $  
$ */
```

```
emis age_ldv$ $  
$ */
```

TABLE 8.3: NO₂ and NO speciation (t/y)

	NO ₂	NO
PC	44.8689	1450.761
HGV	179.4756	1316.154

Then, once the emissions are estimated, we will speciate the emissions by euro standard. The estimation covered 24 hours and for simplicity, we are assuming a year with 365 similar days and dividing by 1 million to have t/y.

```
rowSumsapply c
rbindspeciate
speciate:
speciate:
speciate:
speciate:
speciate:
speciate:
rowSumsapply c
rbindspeciate
speciate:
speciate:
speciate:
speciate:
speciate:
speciate:
data.frame rbindsapply sapply
row.names c
::kable
```

8.6 Volatile organic compounds: nmhc

nmhc were shown earlier in this chapter, but they can be also used to speciate objects, resulting in all species at once. The speciation of NMHC must be applied to NMHC. However, the emission guidelines of Ntziachristos and Samaras (2016) does not show explicit emission factor of NMHC and the suggested procedure is subtract the CH_4 from the HC . This was done on VEIN and now, ef_speed* functions returns NMHC directly. This makes easier to produce speciation.

```
library  
data  
data  
$  
temp_fact$+$  
netspeed $ $ $ $  
c rep  
rep rep rep rep  
lapply: function  
ef_ldv_speed
```

```
c rep  
rep rep rep rep  
lapply:length function  
ef_ldv_speed  
  
emis age_ldv$ $  
$
```

```
emis_post  
  
speciate as.numeric$  
  
names
```

8.7 The speciation iag

The Carbon Bond Mechanism Z (1999) is an lumped-structure mechanism.

Citing:

***“it is currently unfeasible to treat the organic species individually in a regional or global chemistry model for three major reasons:**

- (1) limited computational resources, - (2) lack of detailed speciated emissions inventories, and - (3) lack of kinetic and mechanistic information for all the species and their reaction products.

Thus there exists a need for a condensed mechanism that is capable of describing the tropospheric hydrocarbon chemistry with reasonable accuracy, sensitivity, and speed at regional to global scales.

The research in this aspect is intense and out of the scope of this book. However, it is important that the user who intends to performs atmospheric simulation to understand this why it is important to speciate the emissions and what represent each group.

A popular air quality model nowedays is the Weather Research and Forecasting model coupled to Chemistry (WRF-Chem) (Grell et al., 2005) the **Emissions Guide** () .

The speciation splits the NMHC into the lumped groups for the mechanism CMB-Z. The name comes from the Institute of Astronomy, Geophysics and Atmospheric Sciences (IAG,) from the University of São Paulo (USP). The Departament of Atmospheric Sciences (DAC) of IAG has measure and model air quality models for several years. There are several scientific production such as: Nogueira et al. (2015), Hoshyaripour et al. (2016), Andrade et al. (2017), Freitas et al. (2005), Martins et al. (2006), Pérez-Martinez et al. (2014), Ulke and Andrade (2001), Vivanco and de Fátima Andrade (2006), Boian and Andrade (2012), de Fatima Andrade et al.

(2012), Andrade et al. (2015), Vara-Vela et al. (2016), Rafee (2015), Ibarra-Espinosa et al. (2017c). The following figure, shows an air quality simulation over South East Brazil from DAC/IAG/USP.

Some of these papers show measurements in tunnels, fuel, etc. The speciation it is based on these experiments, and the last versions, covers an update during 2015 so that the exhaust emissions speciation is more representative of brazilian gasoline. This means that:

- ** THE SPECIATION IS BASED ON BRAZILIAN MEASUREMENTS**.

and the user who wants to apply it outside Brazil, should be ensure, that the fleet and fuel are compatible. As that will be hard to accomplish, I can say that this speciation is for and to be used in Brazil, even more, in São Paulo.

The currently speciation is:

The arguments of the function take the following form:

- : Emissions estimation, it is recommended that x are gridded emissions from NMHC.
- : . Alternatively, you can put any name of the following: “e_eth”, “e_hc3”, “e_hc5”, “e_hc8”, “e_ol2”, “e_olt”, “e_oli”, “e_iso”, “e_tol”, “e_xyl”, “e_c2h5oh”, “e_hcho”, “e_ch3oh”, “e_ket”
- : When spec is “iag” veh can take two values depending; when the speciation is for vehicles veh accepts “veh”, eu “Evaporative”, “Liquid” or “Exhaust”
- : “G”, “E” or “D”.
- : “Exhaust” “Evaporative” or “Liquid”.
- : Let this FALSE.
- : Let this TRUE so that the result is a list, and each element of the list a lumped species.

Example:

```
data.frame rnorm  
speciate  
head
```

OZONE CONCENTRATION and SURFACE WIND

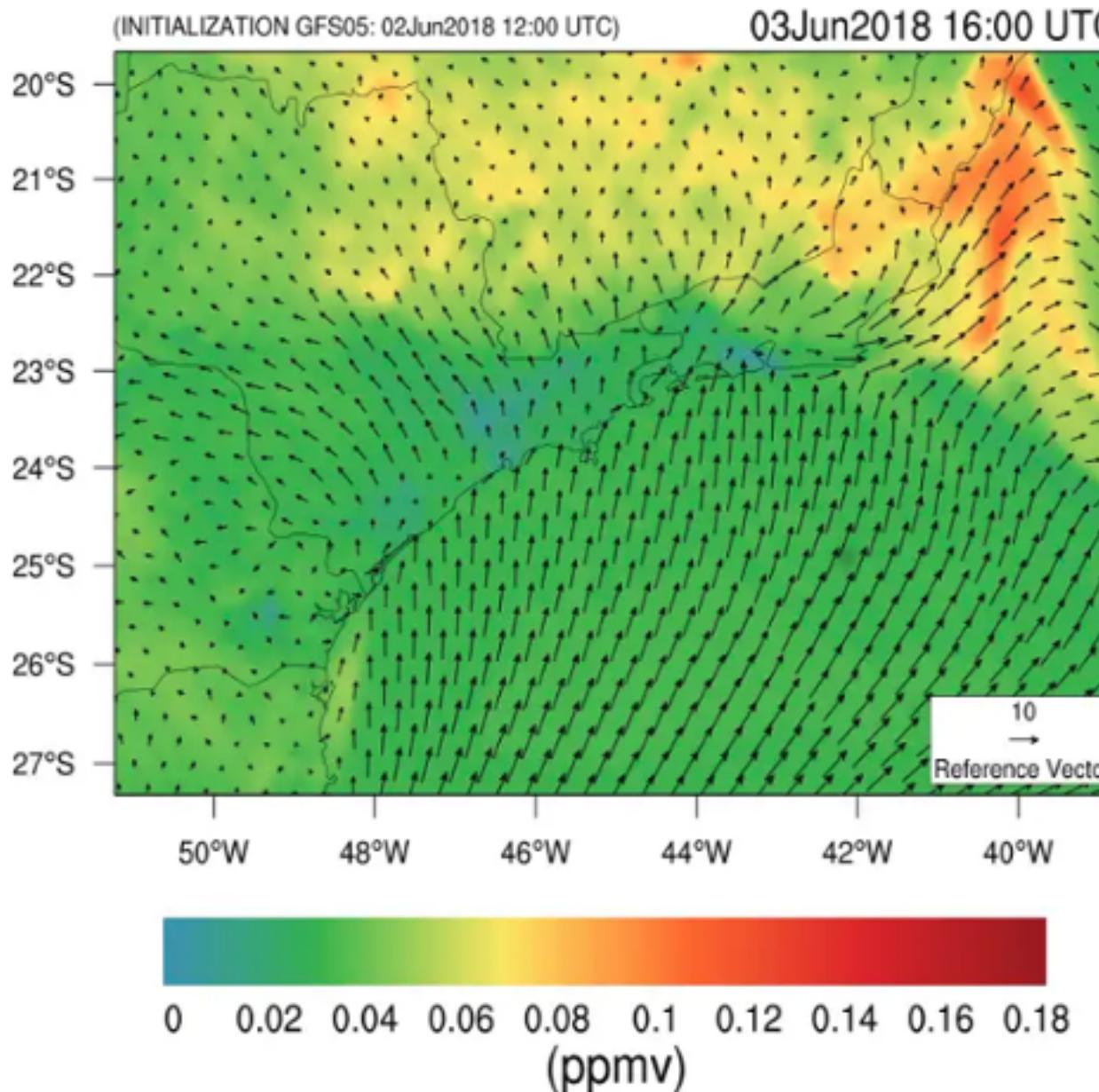


FIGURE 8.4: <http://www.lapat.iag.usp.br/aerossol/wrf9/index.php>

NMHC	MIR	NMHC with E25			NMHC
		Exhaust	Evap	Liq	Exhaust
<i>ETH</i>	0.28	0.2826	0.2317	0.0250	0
<i>HC₃</i>	1.33	0.4352	0.3567	0.2400	0
<i>HC₅</i>	1.58	0.1586	0.1300	0.4500	0
<i>HC₈</i>	1.01	0.0765	0.0627	0	0.0490
<i>OL₂</i>	9.07	0.5807	0.2800	0.0382	0.1260
<i>OLT</i>	8.91	0.2434	0.1174	0.200	0.0018
<i>OLI</i>	1.94	0.1614	0.1323	0.4600	0.0013
<i>ISO</i>	10.68	0.0077	0.0037	0	0
<i>TOL</i>	2.5	0.1405	0.1152	0.0850	0.0050
<i>XYL</i>	8.15	0.2677	0.1291	0	0.0149
<i>C₂H₅OH</i>	1.69	0.3082	0.3082	0.3500	1.5226
<i>HCHO</i>	6.98	0.1127	0.0508	0	0.3605
<i>ALD</i>	8.96	0.0864	0.0663	0	0.1317
<i>CH₃OH</i>	0.65	0	0	0	0
<i>KET</i>	1.65	0	0	0	0

FIGURE 8.5: Speciation for NMHC by type of fuel and process mol/100g
(Ibarra-Espinosa, S., 2017)

```
speciate
```

```
names
```

```
for in :length
print:
```

TABLE 8.4: Speciation of PM_{2.5}

E_SO4i	E_SO4j	E_NO3i	E_NO3j	E_MP2.5i	E_MP2.5j	E_ORGi	E_ORGj	E_ECi	E_ECj	C
0.0077	0.0623	0.00247	0.01053	0.1	0.3	0.0304	0.1296	0.056	0.024	C

8.8 The speciation pmiag

The speciation is also based on IAG studies and applied only for PM_{2.5} emissions. The speciation splits the PM_{2.5} in percentages as follow:

When running this speciation, the only two required arguments are `and .`. Also, There is a message that this speciation applies **only in gridded emissions, because the unit must be in flux g/km²/h**, and internally are transformed to the required unit \$ μg/m²/s\$ as:

$$\frac{g}{(km)^2 * h} * \frac{10^6 \mu g}{g} * \left(\frac{km}{1000m}\right)^2 * \frac{h}{3600s} \frac{1}{dx^2}$$

`dx` is the length of the grid cell. Example:

```
library
data
  data.frame rnorm length
@ 
make_grid //
```

emis_grid

speciate

names

head

9

Inputs for atmospheric models

This last chapter of this book is about generating inputs for atmospheric models, specifically, **WRF-Chem** (Grell et al., 2005). As the Pacific Northwest National Laboratory (PNNL,) defines:

The Weather Research and Forecasting (WRF) model is a next generation meteorological model being developed collaboratively among several agencies (NOAA/NCEP, NOAA/ESRL, NCAR). WRF-Chem is a version of WRF that also simultaneously simulates the emission, turbulent mixing, transport, transformation, and fate of trace gases and aerosols.

Initially, VEIN counted with the function which created a data.frame object of hourly gridded emissions with each pollutant as a column, from the first cell to the last. The data-frame extendend in long format to each hour. This data.frame was designed to be used with script Another Assimilation System for WRF (AS4WRF) (Vara-Vela et al., 2016) written in NCL (Boulder, 2017).

This approach was effective and already tested for some Latinamerican cities (Ibarra et al., 2015a), (Ibarra et al., 2015b). However, this function depends on external software which may not be available. Therefore, me andsome collegues developed the package (Ibarra-Espinosa et al., 2018b), (Ibarra-Espinosa et al., 2017b), which is an R package to read and export emissions to atmospheric models. currently covers the models WRF-Chem (Grell et al., 2005), SPM-BRAMS (Freitas et al., 2005) and R-LINE (Snyder et al., 2013), but only the WRF-Chem interface has been fully implemented. Therefore, this chapter covers only this model.

The concept of connecting VEIN and Wrf-Chem via eixport is quite simple with two approaches:

1. Generating for the wrfinput grid and inputting this emissions into the wrfchemi input file..

2. Generating gridded emissions data-frame for AS4WRF.
-

9.1 WRFChem input with wrfout and GriddedEmissionsArray

The process for generation WRF-Chem input files with is shown on Fig. 9.1. Pink boxes are classes, grey boxes are functions, green functions are functions and white boxes, external objects. Here, the external input file is the `wrfinputd_ox` file, where the `x` is for the domain. This file is inputted into the function which creates a polygon grid needed by . The main characteristic of this grid, is that it has the resolution of the wrfout file.

The `wrfinputd_dox` file is also used by the function to create a `wrfchemi` input file with zeroes.

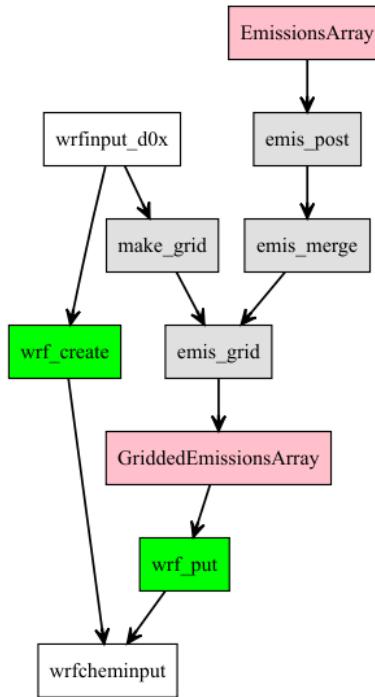
The objects with class for each type of vehicle and pollutant are processed by the function which create street emissions. Then, the function merges all the street emissions files from `emis_grid sf GriddedEmissionsArray` to create an object of that class, and the dimensions of the wrfout file, that is, the sanem number of spatial rows and columns.

Finally, the objects are inputted into the `wrfchemi` file with the function .

Example

9.1.1 Creating a WRF-Chem input file

The emission file will be generated with traffic simulation from the Traffic Engineering Company (CET) for 2012, which consists in traffic flow for morning rush hour of Light Duty Vehicles (LDV). The emission factors will be the data and the wrfout comes from the package. Let's go:

**FIGURE 9.1:** Generation of wrfchem inputs using GriddedEmissionsArray**9.1.1.1 o) Network**

The object net has a class , we transform it into a object. The length of the road is the field 'lkm' in the object net, which are in km but has no units. We must add the right units in order to use

```

require
require
require
readRDS
class
  ::st_as_sf

$ ::set_unitsst_length
  
```

9.1.1.2 1) Vehicular composition

In this case, the vehicular composition consists in only two vehicles, Passenger Cars using Gasoline with 25% of Ethanol and Light Trucks consuming Diesel with 5% of biodiesel. Also, the emission factors cover CO . The temporal distribution will cover only one hour.

```
age_ldv$
```

9.1.1.3 2) Emission factors

The emission factors used comes from CETESB (2015), which are constant by age of use. In practice, this data.frame is like an Excel spreadsheet. Then, the constructor function converts our numeric vector, which is the column of our data.frame, into required type of object of the function. Parenthesis were added in order to print the objects in one line.

```
data
EmissionFactorsList$ ==
```

9.1.1.4 3) Estimation of emissions

The estimation of emissions is shown in the following code chunk in a simplified way. The emissions array are aggregated.

```
data
emis $
```

9.1.1.5 4) Post-estimations

The resulting emissions array is now processed with .

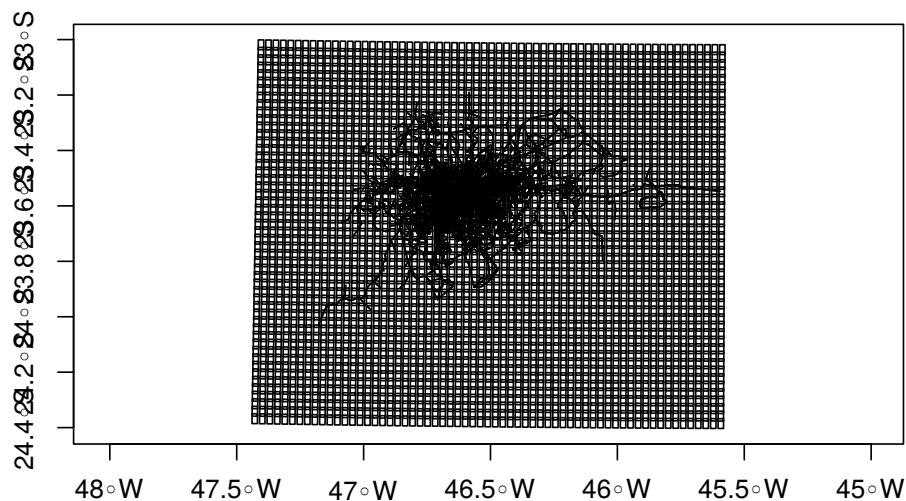
```
emis_post
```

9.1.1.5.1 4a WRFChem input file with

Now it is time generate the emissions grid. In order to create a wrfchem input file, we need a wrfout file. In this case, the wrfout covers a wider area of the emissions, located in São Paulo, Brazil. The wrfout file is available from the R package .

```
pasteSystem.file  
make_grid
```

```
plotst_geometry  
plotst_geometry
```



Notice that when we create grid from the path to wrfinput, there is a message with *Number of lat points 51* and *Number of lon points 63*. This information is critical for later conversions. Now that we have the street emissions and the grid, we can obtain our gridded emissions:

```
emis_grid
```

```
names
```

```
$
```

This function requires the package `raster` when the data is in latitude and longitude degrees. There are some messages about the total emissions of street and grid. This was made in order to ensure that `emis_grid` is conservative. The resulting object is an object of `POLYGON`. In order to plot the emissions, I selected the emissions above 1, because it is easier to see the plot. If we plot the gridded emissions, we will see:

```
$ $* + ^-
plot$ >
  ::cpt
```

The emissions are mostly distributed into main roads, as expected. The

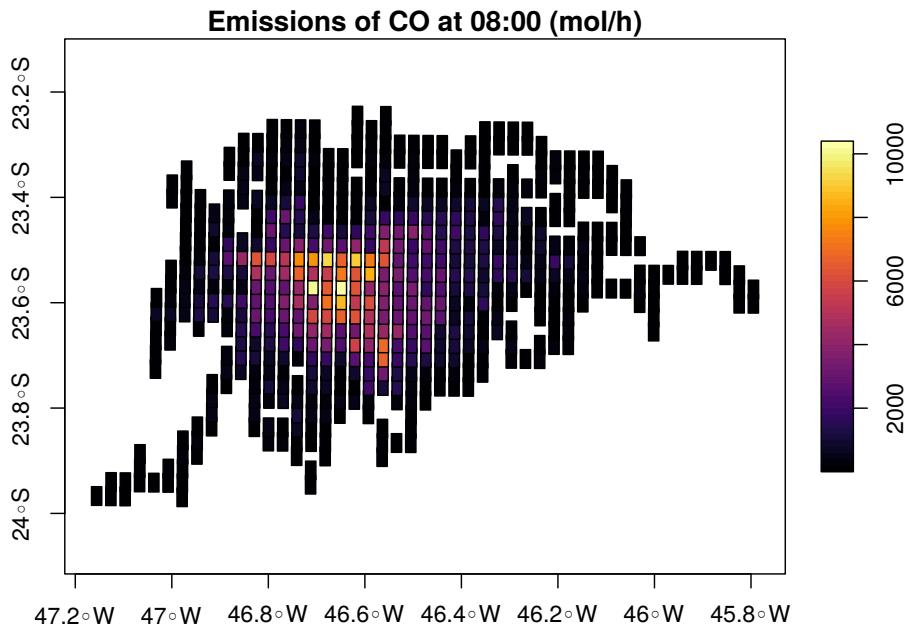


FIGURE 9.2: Gridded emissions (mol/h)

next step is create the `GriddedEmissionsArray` that will be inputted into the `wrfchem` functions. The `GriddedEmissionsArray` is a constructor function that reads class, `GriddedEmissions`, of POLYGONS, a `NetCDF` or a `GRIB` to create an Array of the gridded emissions. The dimensions are lat, lon, mass and time. Therefore, it is important to know the number of lat and lon points!

```
GriddedEmissionsArray
```

```
plot ::cpt
```

The image is similar to the plot of the gridded emissions, which means that it is correct. Now we can input the our new created object `GriddedEmissionsArray` into the `wrfchem` input file, but first, we need to create it. This is done using the `GriddedEmissionsArray` functions.

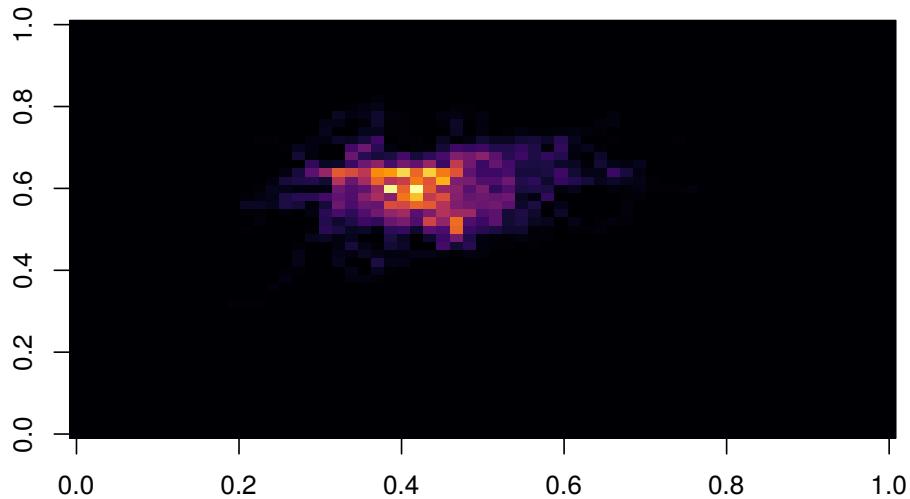


FIGURE 9.3: GriddedEmissionsArray (mol/h)

We first load the library . Then load the data which has the emissions options for WRFchem. In this case, we are creating an wrfchem input file with 40 pollutants, with o.

```
library  
data  
  
wrf_create_system.file  
tempdir
```

The wrfchemi input file was created on a temporal directory with the function . Now we can input our into our new wrfchem input file.

```
list.files tempdir  
  
wrf_put
```

Then, we just check our resulting NetCDF file plotting the emissions we just put.

wrf_plot

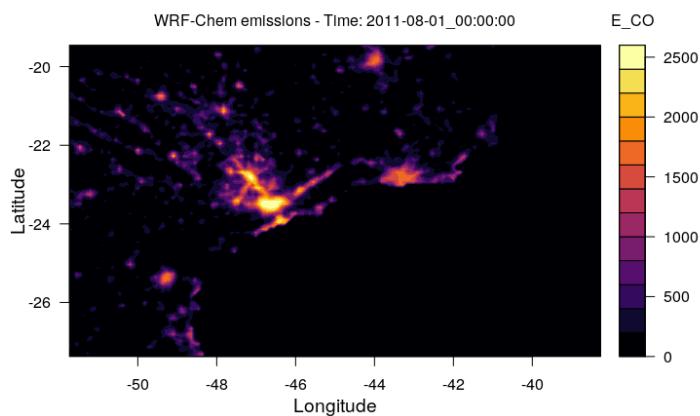


FIGURE 9.4: Wrfchem input file (NetCDF) (mol/h)

9.1.1.5.2 4b WRFChem input file with

creates a with columns lat, long, id, pollutants, local time and GMT time. This dataframe has the proper format to be used with WRF assimilation system: “ASimulation System 4 WRF (AS4WRF, Vara-Vela (2013))

args

where:

- : Gridded emissions, which can be a SpatialPolygonsDataFrame, or a list of SpatialPolygonsDataFrame, or a sf object of “POLYGON”. The user must enter a list with 36 SpatialPolygonsDataFrame with emissions for the mechanism CBM-Z.

- : Number of repetitions of the emissions period
- : 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
- : Time zone as required in for function as.POSIXct
- : Coordinate reference system, e.g. “+init=epsg:4326”. Used to transform the coordinates of the output
- : logical value to indicate if sdf is a list or not
- : ignored.

In this case, we can use our gridded emissions directly here:

```
$  
emis_wrf
```

```
head
```

Then you must export this into a .txt file. You must include the number of columns to match your luded species and also take out the columns time_lt, time_utc and dutch. These three columns are just informative.

```
:  
write.table tempfile
```

Finally, in order to generate the wrfchem input file, you must have the resulting .txt file, a wrfinput file with the same grid spacing, the NCL script AS4WRF and a namelist. AS4WRF and the namelist can be obtained contacting¹ and².

¹²

10

Quality check and errors

This last chapter of this book presents aspects that must be considered for any practitioner interested in developing emissions inventories. The first part covers a summary of the EMEP/EEA air pollutant emission inventory guidebook 2016, Technical guidance to prepare national emission inventories (McGlade and Vidic, 2009), based on the Intergovernmental Panel on Climate Change (IPCC) 2006 Good Practice Guidance [Change (2006). The second part consists in advices and specific considerations for avoiding errors when running VEIN in each part of the inventory.

10.1 Guidance from EMEP/EEA air pollutant emission inventory guidebook

I believe that this part of this chapter delivers knowledge that you really will gain with experience. But even with experience you can get lost, and returning to this part would save you. This part is based on the EMEP/EEA air pollutant emission inventory guidebook (McGlade and Vidic, 2009).

10.1.1 Key categories

Key categories are the most important **source** categories. This section is based on the chapter Key category analysis and methodological choice (J, 2013) of the EMEP/EEA air pollutant emission inventory guidebook 2013. They are important because they are responsible for most of pollution in a determined region. For instance, it has been described that vehicles are the most important source of pollution in mega-cities (Molina and Molina, 2004). Evenmore, in the case of São Paulo, the official emissions inventory

informs that vehicles are responsible for more than 97 % of CO , 79 % of HC , 68 % of NO_X , and only 29 % of PM and 22 % of SO_X (CETESB, 2015). This shows another characteristics of key categories: they depend of specific pollutants.

For another example, consider a city which suffers cold weather with use bio-mass burning for cooking and heating. The main source of $PM_{2.5}$ will be certainly bio-mass burning.

Now consider an industrial region with industrial process that emit too much SO_X . The key-categories might be industrial sources.

And one more example, consider a huge mega-city with electric mobility. However, the main source of electricity are thermoelectric based on carbon (the key).

This implies that it should be a nomenclature for categorizing and naming sources. IPCC, has a nomenclature for instance.

The idea behind key categories, is that most efforts must be applied in this categories, obtaining the highest level of detail with less uncertainty. Hence, the emissions guidelines (McGlade and Vidic, 2009) proposes three levels of complexity estimation methods, known as Tier Methods. The complexity increases from level 1, 2 and 3. The VEIN tier is 3, which is that the most complex function of vehicular emissions estimations are incorporated, with the exception of evaporative. I need to improve that part.

- Tier 1: a simpler method which include available activity and emissions factors.
- Tier 2: Similar to Tier 1, but includes more specific emission factors.
- Tier 3: Most complex methodologies with higher level of detail, temporal and spatial.

10.1.2 Uncertainty

This is a very important part in any emissions inventory and a dedicated chapter or section should be made in any report/paper. This section is based on the chapter Uncertainties (Pulles T, 2013) of the EMEP/EEA air pollutant emission inventory guidebook 2013. The 2006 IPCC Guidelines

chapter states that estimating the uncertainty of an inventory is needed (Change, 2006).

It is recommended to use 95% confidence interval. This means that:

- there is a 95% of probability that the actual value of the quantity estimates is within the interval defined by the confidence interval.
- The probability of the actual value will be outside the range it is 5%.

The general form for estimating emissions was shown on Eq. (1.1) (Pulles T, 2013). This equation will be the base for calculating the uncertainty. This section is applied when measurements are made, for the case of activity or emission factors. In those case, it is possible calculate the required confidence intervals.

10.1.2.1 Default uncertainty ranges

Activity data

The following table is taking directly from McGlade and Vidic (2009) and propose indicative ranges that could be applied in cases where no independent data are available.

- National official statistics: - .
- Update of last year's statistics using Gross Economic Growth factors: 0 - 2%.
- International Energy Agency (IEA) statistics: OECD: 2 - 3%, non-OECD 5-10%.
- United Nation data bases: 5 - 10%.
- Default values, other sources: 30 - 100%.

Emission factors

The following table is taking directly from McGlade and Vidic (2009) and propose rating definitions for emission factors(McGlade and Vidic, 2009).

- A: An estimate based on a large number of measurements made at a large number of facilities that fully represent the sector. Error: 10 o 30%.
- B: An estimate based on a large number of measurements made at a large number of facilities that represent a large part of the sector. Error: 20 to 60%

TABLE 10.1: Precision indicators (Ntziachristos and Samaras 2016)

Category	NOx	CO	NMHC	CH4	PM	N ₂ O	NH ₃	CO ₂
PC G w/o Catalyst	A	A	A	A	-	C	C	A
PC G w Catalyst	A	A	A	A	-	A	A	A
PC D	A	A	A	A	A	B	B	A
PC LPG	A	A	A	-	-	-	-	A
PC LPG w/o Catalyst	A	A	A	A	D	C	C	A
PC LPG w Catalyst	D	D	D	D	D	D	D	A
PC 2 strokes	B	B	B	D	-	D	D	B
LCV G	B	B	B	C	-	B	B	A
LCV D	B	B	B	C	A	B	B	A
HDV G	D	D	D	C	-	D	D	D
HDV D	A	A	A	D	A	B	B	A
MC cc <50	A	A	A	B	-	B	B	A
MC cc > 50 2 strokes	A	A	A	B	-	B	B	A
MC cc > 50 4 strokes	A	A	A	B	-	B	B	A
Cold-start PC G Pre Euro	B	B	B	-	-	-	-	B
Cold-start PC G Euros	B	B	B	A	-	-	-	A
Cold-start PC D Pre Euro	C	C	C	-	C	-	-	B
Cold-start PC D Euros	A	A	A	A	A	-	-	A
Cold-start PC LPG	C	C	C	-	-	-	-	B
Cold-start LCV G	D	D	D	-	-	-	-	D
Cold-start LCV F	D	D	D	-	D	-	-	D

- C: An estimate based on a number of measurements made at a small number of representative facilities, or an engineering judgement based on a number of relevant facts. Error: 50 to 200%.
- D: An estimate based on single measurements, or an engineering calculation derived from a number of relevant facts. Error: 100 to 300%.
- E: An estimate based on an engineering calculation derived from assumptions only. Error: Order of magnitude.

In McGlade and Vidic (2009) appears ratings for road transport emission factors that differ from the ratings that appear in Ntziachristos and Samaras (2016). To preserve consistency in this book, here is presented the precision indicators, Table 4-1 of Ntziachristos and Samaras (2016) report.

Uncertainties can be aggregated with two approaches

- *Rule A:* uncertainties are combined by *addition*, as shown in Eq. (10.1):

$$U_{total} = \sqrt{\frac{\sum_{i=1}^n (U_i \cdot x_i)^2}{\sum_{i=1}^n}} \quad (10.1)$$

Where, x are the quantities, U_i are the uncertain quantities and the percentage uncertainties (half the 95% confidence interval) associated with them, respectively. U_{total} is the percentage uncertainty in the sum of the quantities (half the 95% confidence interval divided by the total (i.e. mean) and expressed as percentage).

- *Rule B:* uncertainties are combined by *multiplication*, as shown on Eq. (10.2):

$$U_{total} = \sqrt{\sum_{i=1}^n (U_i)^2} \quad (10.2)$$

Where, U_i are the percentage quantities (half the 95% confidence interval) associated with each of the quantities. U_{total} is the percentage in the product of the quantities (half the 95% confidence interval divided by the total and expressed as percentage).

Alternatively, a Monte Carlo simulation can be done.

10.1.3 Quality Assurance and Quality Check

This section is mostly based on the chapter Inventory management, improvement and QA/QC (Goodwin et al., 2013) of the EMEP/EEA air pollutant emission inventory guidebook 2013.

According to Wikipedia:

- **Quality assurance (QA)** is a way of preventing mistakes and defects in manufactured products and avoiding problems when delivering solutions or services to customers (Wikipedia contributors, 2018b).
- **Quality control (QC)** is a process by which entities review the quality of all factors involved in production (Wikipedia contributors, 2018c).

The main reference in QA/QC are the International Organization for Standardization ISO 9000 (Wikipedia contributors, 2018a).

The idea is to avoid errors in the development of the inventory. Hence, it is very important that the objective of the inventory, frequency and spatial and temporal resolution must be very clear. As mentioned before, the inventory can have a purpose of policy application or scientific. In any case, the QAQC procedures should be explicitly stated covering five data quality objectives **transparency, consistency, comparability, completeness and accuracy**.

This means that another researcher/practitioner should be able to reproduce the results. However, this is not always the case. Evenmore, it has been shown that there are currently a crisis of reproducibility in science, where ‘more than 70% of researchers have tried and failed to reproduce another scientist’s experiments, and more than half have failed to reproduce their own experiments’ (Baker, 2016). I believe that emissions inventories should guarantee the reproducibility of results, in scientific and policy application inventories.

10.1.4 Inventory management cycle

Another important aspect is the inventory management cycle. The inventory manager is responsible for institutional arrangements, meet deadlines and for the inventory management cycle. The inventory compiler gets the data and estimate the emissions with the respective Tier.

The cycle is:

1. Priorization of improvements: As the resources are limited, prioritization must be given to key categories.
2. Data-collection: It is important to establish formal agreements needs with data providers using protocols. The protocols must clearly show the data needed, its format, content and dates of delivery.
3. Inventory compilation. The inventory compiler estimate the emissions.
4. Consolidation. The inventory manager consolidates all the emis-

sions ensuring quality in data and methods for estimating emissions.

5. Data Quality Review.
 6. Reporting.
 7. Lessons learned and improvement review.
 8. Inventory Management Report.
 9. Quality Assurance and Quality Control Plan.
-

10.2 Avoiding errors with VEIN

The first part of this chapter covered some fundamental aspects in emissions inventory management for quality assurance and quality control. This part covers typical errors that I've faced using VEIN and I hope it helps users how to avoid them.

Currently, VEIN does not have graphical user interface (GUI) and runs on R (R Core Team, 2017), which can be intimidating for new users. Also, even experienced users can do mistakes. If you find a mistake, write it down to avoid it in future. Be organized. I recommend to code your inventory keeping in mind that you will have to check your scripts in the future, so it would be nice if in the first try your code looks nice. And also, don't panic.

10.2.1 Traffic flow

One of the first inputs of it is the traffic flow data. You must have an agreement with data providers to get the data. Also, the compiler must understand the format of the data. In my experience, data providers come from government agencies with limited resources and few time. This means that, if there is not a formal agreement, they won't spend too much time for processing their data to meet your needs (because it is not part of their jobs). And even, if there was a formal agreement, the data processing must be done by the data compiler.

The data provided in this book covers a travel demand output for Metropoli-

tan Are of São Paulo (MASP) made by the Traffic Company Engineering of São Paulo, for the year 2012. The has the same content inside VEIN package, with the exception that covers the whole MASP. The data originally is in format MAPINFO. Therefore, the package `sf` must be used to read the data.

```
library
st_read
```

The section @ref(#traffic) shows details about this data.

- Have a meeting with data delivers to check and solve any doubt regarding the data.
- Check if your data is projected or not. The data probably will be projected with a UTM zone, for instance code 31983 . VEIN imports functions from and which depend on GEOS libraries. This means that VEIN can work with data projected or not. Nevertheless, I suggest to work with projected data for your location.
- Ensure that the data is correctly read and that there are no objects of class in the columns.
- Make sure that of what are the units of the traffic flow and remember that most of VEIN functions are designed to work with hourly traffic flow.
- Plot the data. Check if the data looks fine or have some mistakes.
- Calculate the length of each road. Make proper transformations and **make sure that resulting length of road as units km**. I suggest to use the name `lkm`. This can be easily done with in R with packages and installed. Let's say that your data is named `net` and your data is already projected. Also, that your traffic flow is named `ldv` for light duty vehicles and `hdv` for heavy duty vehicles.

```
plot
plot
$ ::st_length
$$ ::set_units$
```

Despite that this transformation can be done by dividing `lkm` by 1000, this

would be wrong because the wouldn't change and they must be . Hence, using the functions of the package is the recommended way.

10.2.2 Vehicular composition

The vehicular composition is a critical part in the emissions inventories for vehicles. It consists in the percentage of each type of vehicle and technology. Despite that it seems pretty straightforward, making a mistake in this part of the emissions inventories would cost lots of time. **Developing an emissions inventory is a complex task, you must take great care in simple calculations, because if the results are not consistent, it can take LOTS of time, to find the error, usually when the deadlines are dead.**

The function needs the vehicular composition to create the respective directories to run VEIN. I have the feeling that most of model users like structured models with clear input and outputs. VEIN is not like that, and this is in part due to the nature of the emissions inventories. I could design a model that works perfectly with one type of input, but in real life, a desired input is not always easy to get. For instance, the traffic simulation used inside the model is not easy to get, even in cities of developed countries. Hence, the inventory compiler must struggle to do their job. Therefore, VEIN was designed with flexibility and versatility on mind.

Anyways, the vehicular composition are the number of each type the following vehicles: PC, LCV, HGB, BUS, MC. Then, each sub category will be divided by type of age of use. For instance, the vehicular emissions inventory for São Paulo State considers 4 type of vehicles:

- Particulate Cars using gasohol (Gasoline with 25% of ethanol, PC_25).
- Particulate Cars Flex using gasohol (Gasoline with 25% of ethanol, PC_F25).
- Particulate Cars Flex using ethanol (Gasoline with 25% of ethanol, PC_FE100).
- Particulate Cars using ethanol (Gasoline with 25% of ethanol, PC_E100).

This means that the number of PC is 4.

Then each of this vehicles is divided by age of use. As consequence, we must know when each of these vehicle entered and went out of the market.

Again, for São Paulo conditions, Flex engines entered into market in 2003 and nowadays most of new cars are flex. Vehicles with engines designed for burning ethanol had a peak in early 80s, but they went off the market in 2006. Gasoline vehicles have been in circulation from the beginning and CETESB inventory considers a life span of 40 years of use.

The same analyses must be done with each of the vehicle categories.

The distribution by age of use indirectly shows the technology associated with emission standards by age of use.

10.2.3 Units

Currently, VEIN checks that the variable lkm (length) must be in km. This is to avoid that the user wrongly enter length in meters or without units. As a result, the emissions would be huge and wrong.

In the case of vehicles, currently there are not designed a unit of "vehicles" in VEIN. In transportation studies, it is used the unit "equivalent vehicle" which standardize each type of vehicle by its size. For instance, LCV are sometimes equivalent of 1.2 vehicles. Buses or HGV can be equivalent to 2 or more vehicles. However, there are no such type of this things in VEIN and units management is designed to control emission factors and length.

The temporal dimension is also an aspect difficult to handle regarding the units. This is because, sometimes mileage are correctly in km, but the timespan is in one year. Therefore, the units management is designed to avoid errors with emission factors and calculation of emissions. This means that a future version of VEIN will eradicate all temporal dimensions to ensure right calculation of mass only. This means that VEIN will not check if the data is hourly or annual.

10.2.4 Emission factors

When I started this book, VEIN contained few emission factors. Now, the version 0.5.4 has all the 2016 emission factors of CETESB, almost 100 emission factors from the European Emission guidelines and all the

BASE emission factors from the International Vehicle Model (IVE). The functions to access this data are:

- ef_cetesb.
- ef_ldv_speed.
- ef_hdv_speed.
- ef_ive.

I will be working to import emission factors from HBEFA model which are based traffic situation (very cool).

10.2.5 Deterioration

VEIN currently cover the deterioration emission factors from the European emission guidelines. This values results in a simply numeric vector depending on standard, mileage, type of vehicle and pollutant. However, it would be possible to use any other deterioration factors.

Ensure to use correctly the data and cite respective literature.

10.2.6 Fuel evaluation

It is a good practice to ensure that the mass of fuel consumed of vehicle in your study area, matches the fuel sale son your area. If not, calibrate your emissions by number of vehicles (if bottom-up) or a combination of vehicles and mileage (if top-down) to match fuel sales.

10.2.7 Emissions estimation

Most of emission functions returns an array of emissions, which means a matrix of matrices. The idea was that each dimension has a meaning but I've been thinking that it is not necessary and each dimension should have different nature, for instance, it is not necessary two dimensions for time, despite that one indicates hour and the other days. So I might change that in future. Don't panic, I'm always trying to change the functions without breaking older code.

I hope you liked this book.

Appendix A: Fast Example of VEIN

```
library  
inventory file.pathtempdir  
  
sourcepaste0file.pathtempdir
```



Appendix B: Detailed Example of VEIN

.1 Network

```
inventoryfile.pathtempdir
setwd
data
data
data
temp_fact$+$
netspeed $ $ $ $
saveRDS
saveRDS
```

.2 Vehicular composition (CETESB, 2015)

The vehicular composition has 5 types of vehicles: Passenger Cars (PC), Light Commercial Vehicles (LCV), Heavy Good Vehicles or trucks (HGV), Buses (BUS), and Motorcycles (MC). The default value of this argument is:

, which means that there are 4 types of PC, 5 of LCV, 5 of trucks, 3 of buses and 9 types of motorcycles. This composition comes from the vehicular emissions inventory of the Environmental Agency of São Paulo, Brazil (CETESB, 2015). In Brazil, the fuel used in vehicles is blended with ethanol with and biodiesel. The user can use **any** vehicular composition that represents its fleet with up-to 99 type of vehicles per category. The default composition according Brazilian emissions inventory is:

Passenger Cars (PC)

1. Passenger Cars using Gasoline blended with 25% of ethanol (E25).
2. Passenger Cars with Flex engine using Gasoline blended with 25% of ethanol (FE25).
3. Passenger Cars with Flex engine using 100% of ethanol (FE100).
4. Passenger Cars with engines that uses only ethanol (E100).

Light Commercial Vehicles (LCV)

1. Light Commercial Vehicles with gross weight (GW) less than 3.8 t using Gasoline blended with 25% of ethanol (E25).
2. Light Commercial Vehicles with GW less than 3.8 t with Flex engine using Gasoline blended with 25% of ethanol (FE25).
3. Light Commercial Vehicles with GW less than 3.8 t with Flex engine using 100% of ethanol (FE100).
4. Light Commercial Vehicles with GW less than 3.8 t with engines that uses only ethanol (E100).
5. Light Commercial Vehicles with GW less than 3.8 t with engines that uses only ethanol (E100).

Heavy Good Vehicles (HGV) or Trucks

1. Semi Light Trucks (SLT) with $3.8 \text{ t} < \text{GW} < 6 \text{ t}$ using Diesel blended with 5% of biodiesel (B5).
2. Light Trucks (LT) with $6 \text{ t} < \text{GW} < 10 \text{ t}$ using Diesel blended with 5% of biodiesel (B5).
3. Medium Trucks (MT) with $10 \text{ t} < \text{GW} < 15 \text{ t}$ using Diesel blended with 5% of biodiesel (B5).
4. Semi Heavy Trucks (SHT) with $15 \text{ t} < \text{GW}$ on a Rigid Truck (RT) and $\text{GW} < 40$ on an Articulated Truck (AT) using Diesel blended with 5% of biodiesel (B5).
5. Heavy Trucks (HT) with $15 \text{ t} < \text{GW}$ on a RT and $\text{GW} \geq 40$ on an AT using Diesel blended with 5% of biodiesel (B5).

Buses

1. Urban Bus (UB) using Diesel blended with 5% of biodiesel (B5).
2. Small Urban Bus (SUB) using Diesel blended with 5% of biodiesel (B5).
3. Motorway Bus (MB) or Coach using Diesel blended with 5% of biodiesel (B5).

Motorcycles (MC)

1. Motorcycles with engine cc < 150 using Gasoline blended with 25% of ethanol (E25).
2. Motorcycles with engine 150 < cc < 500 using Gasoline blended with 25% of ethanol (E25).
3. Motorcycles with engine cc > 500 using Gasoline blended with 25% of ethanol (E25).
4. Motorcycles with engine cc < 150 with Flex engine using Gasoline blended with 25% of ethanol (FE25).
5. Motorcycles with engine 150 < cc < 500 with Flex engine using Gasoline blended with 25% of ethanol (FE25).
6. Motorcycles with engine cc > 500 with Flex engine using Gasoline blended with 25% of ethanol (FE25).
7. Motorcycles with engine cc < 150 with Flex engine using 100% of ethanol (FE100).
8. Motorcycles with engine 150 < cc < 500 with Flex engine using 100% of ethanol (FE100).
9. Motorcycles with engine cc > 500 with Flex engine using 100% of ethanol (FE100).

.3 Traffic data

The vehicular composition will split the traffic simulation as shown on next table. The simulation has traffic for Light Duty Vehicles and Heavy Good Vehicles, therefore, the vehicular composition must split this categories and considerations:

TABLE .2: Vehicular composition of PC for applied in this book

Vehicles	Composition
PC_E25	37.25%
PC_FE25	22.26%
PC_FE100	37.97%
PC_E100	2.44%

- Passenger Cars (PC) = 75% of Light Duty Vehicles (LDV) of traffic simulation.
- Light Commercial Vehicles (LCV) = 10% of LDV of traffic simulation.
- Motorcycles (MC) = 15% of LDV on traffic simulation.
- Heavy Good Vehicles (HGV) = 75% of Heavy Duty Vehicles (HDV).
- Buses == 25% of Heavy Duty Vehicles (HDV).
- LCV and PC have vehicles with flex engines capable to run with any mixture of gasoline and ethanol (Giroldo et al., 2005).
- Type of fuel consumed consists in gasoline blended with 25% of ethanol (E25).
- The vehicular composition consists in type of vehicles and type of fuel. Optionally, the user could have a wider vehicular composition considering more sizes, gross weight, etc.
- The percentages in composition apply for each type of vehicle PC, LCV, HGV, BUS and MC.

Passenger Cars

The vehicular composition consisted in PC using E25, PC with Flex engines using E25 and E100, and OC with engines for consuming only E100. The PC with Flex engines entered into Brazilian market in 2003, therefore, at year 2015 flex vehicles has 13 years of use. On the other case, PC with engines for E100 went out of the market in 2007, therefore, the newest PC with E100 engine has 9 years of use.

```
age_ldv $  
/*  
age_ldv $  
/*
```

TABLE .3: Vehicular composition of LCV for applied in this book

Vehicles	Composition
LCV_E25	39.13%
LCV_FE25	15.21%
LCV_FE100	25.90%
LCV_E100	1.18%
LCV_B5	18.58%

```
age_ldv $ /*  
age_ldv $ /*  
saveRDS  
saveRDS  
saveRDS  
saveRDS
```

Light Commercial Vehicles

The engine/fuel type affects in the same way to LCV with PC vehicles. However, this category has a fraction of vehicles being driven with diesel. In Brazil, all diesle is blended with approximatly 5% of biodiesel, then it is named as B5. The categorization of the names in LCV is similar with PC.

```
age_ldv $ /*  
age_ldv $ /*  
age_ldv $ /*  
age_ldv $ /*  
age_ldv $ /*
```

TABLE .4: Vehicular composition of HGV for applied in this book

Vehicles	Composition
SLT	8.38%
LT	25.50%
MT	15.28%
SHT	24.98%
HT	25.85%

```
saveRDS
saveRDS
saveRDS
saveRDS
saveRDS
```

Heavy Good Vehicles

HGV uses diesel which is blended with approximatly 5% of biodiesel from sugar- cane.

```
age_hdv $ /*
saveRDS
saveRDS
saveRDS
saveRDS
saveRDS
```

TABLE .5: Vehicular composition of BUS for applied in this book

Vehicles	Composition
UB	77.43%
SUB	9.07%
MB	13.5%

Buses

In Brazil there are many type of buses, but in this book we are focussing in the most abundant: Urban Buses, Small Urban Buses and Motoreway Buses or Coach. According to the Secretary of Urban Mobility of São Paulo (Sptrans,), the fleet has an average age of use of 5 years and 5 months. To achieve this average age, the agemax of this vehicles is 10 years of use.

```
age_hdv $ /*  
age_hdv $ /*  
age_hdv $ /*  
saveRDS  
saveRDS  
saveRDS
```

Motorcycles

The vehicular composition consisted in Motorcycles using E25, and recently, in year 2010, entered into the market flex motorcycles, which can use gasoline E25 or ethano E100. Therefore, the oldest flex MC is 6 years old.

```
age_hdv $ /*  
age_hdv $ /*
```

TABLE .6: Vehicular composition of MC for applied in this book

Vehicles	Composition
MC_150_E25	72.97%
MC_150_500_E25	11.28%
MC_500_E25	3.15%
MC_150_FE25	3.93%
MC_150_500_FE25	0.57%
MC_500_FE25	0.16%
MC_150_FE100	6.69%
MC_150_500_FE100	0.98%
MC_500_FE100	0.27%

```
age_hdv $
```

```
/*
```

```
saveRDS
```

```
saveRDS
```

```
saveRDS
```

```
saveRDS
```

saveRDS

saveRDS

saveRDS

saveRDS

saveRDS



Appendix B



Bibliography

- Amato, F., Querol, X., Johansson, C., Nagl, C., and Alastuey, A. (2010). A review on the effectiveness of street sweeping, washing and dust suppressants as urban pm control methods. *Science of The Total Environment*, 408(16):3070 – 3084.
- Andrade, M. d. F., Kumar, P., de Freitas, E. D., Ynoue, R. Y., Martins, J., Martins, L. D., Nogueira, T., Perez-Martinez, P., de Miranda, R. M., Albuquerque, T., et al. (2017). Air quality in the megacity of São Paulo: Evolution over the last 30 years and future perspectives. *Atmospheric Environment*, 159:66–82.
- Andrade, M. d. F., Ynoue, R. Y., Freitas, E. D., Todesco, E., Vara Vela, A., Ibarra, S., Martins, L. D., Martins, J. A., and Carvalho, V. S. B. (2015). Air quality forecasting system for southeastern Brazil. *Frontiers in Environmental Science*, 3:1–12.
- Anselin, L. (1995). Local indicators of spatial association—lisa. *Geographical analysis*, 27(2):93–115.
- Arbor, A. (2003). User's guide to mobile 6.1 and mobile 6.2. *Assessment and Standard Division, Office of Transportation and Air Quality*, USEPA, Mich.
- Baker, M. (2016). Is there a reproducibility crisis? a nature survey lifts the lid on how researchers view the crisis rocking science and what they think will help. *Nature*, 533(7604):452–455.
- Barlow (2009). A reference book of driving cycles for use in the measurement of road vehicle emissions. Technical report, Transport Research Laboratory.
- Boian, C. and Andrade, M. d. F. (2012). Characterization of ozone transport among metropolitan regions. *Revista Brasileira de Meteorologia*, 27(2):229–242.

- Boulder, C. U. (2017). The ncar command language (ncl)(version 6.4. 0).
- Boulter and Barlow (2009). Artemis: Assessment and reliability of transport emission models and inventory systems final report. Technical report, Transport Research Lalboratory.
- Bruni, A. D. C. and Bales, M. P. (2013). Curvas de intensidade de uso por tipo de veículo automotor da frota da cidade de são paulo. Technical report, CETESB. <http://veicular.cetesb.sp.gov.br/wp-content/uploads/sites/35/2013/12/curvas-intensidade-uso-veiculos-automotores-cidade-sao-paulo.pdf>.
- Bureau of Public Roads (1964). Traffic and assignment mannal. Technical report, Dept. of Commerce, Urban Planning Division, Washington D.C.
- CETESB (2015). Emissões veiculares no estado de são paulo 2014. <http://veicular.cetesb.sp.gov.br/relatorios-e-publicacoes/>.
- Change, I. (2006). 2006 ipcc guidelines for national greenhouse gas inventories.
- Chen, R.-H., Chiang, L.-B., Chen, C.-N., and Lin, T.-H. (2011). Cold-start emissions of an si engine using ethanol–gasoline blended fuel. *Applied Thermal Engineering*, 31(8):1463–1467.
- contributors, O. (2017). Planet dump retrieved from <https://planet.osm.org>. Accessed on 18-07-2017.
- Cooley, D. (2018). *googleway: Accesses Google Maps APIs to Retrieve Data and Plot Maps*. R package version 2.5.0.
- Corvalán, R. M., Osses, M., and Urrutia, C. M. (2002). Hot emission model for mobile sources: application to the metropolitan region of the city of santiago, chile. *Journal of the Air & Waste Management Association*, 52(2):167–174.
- Corvalán, R. M. and Vargas, D. (2003). Experimental analysis of emission deterioration factors for light duty catalytic vehicles case study: Santiago, chile. *Transportation Research Part D: Transport and Environment*, 8(4):315–322.

- Davis, N., Lents, J., Osses, M., Nikkila, N., and Barth, M. (2005). Part 3: Developing countries: development and application of an international vehicle emissions model. *Transportation Research Record: Journal of the Transportation Research Board*, 8(1939):155–165.
- de Fatima Andrade, M., de Miranda, R. M., Fornaro, A., Kerr, A., Oyama, B., de Andre, P. A., and Saldiva, P. (2012). Vehicle emissions and pm_{2.5} mass concentrations in six brazilian cities. *Air Quality, Atmosphere & Health*, 5(1):79–88.
- Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271.
- Dowle, M. and Srinivasan, A. (2017). *data.table: Extension of 'data.frame'*. R package version 1.10.4-3.
- EJ-JRC/PBL (2016). Emission database for global atmospheric research (edgar), release edgar v4.3.1_v2 (1970 - 2010).
- Eom, J., Park, M., Heo, T.-Y., and Huntsinger, L. (2006). Improving the prediction of annual average daily traffic for nonfreeway facilities by applying a spatial statistical method. *Transportation Research Record: Journal of the Transportation Research Board*, 1(1968):20–29.
- Freitas, E., Martins, L., Silva Dias, P., and Andrade, M. d. F. (2005). A simple photochemical module implemented in rams for tropospheric ozone concentration forecast in the metropolitan area of são paulo, brazil: Coupling and validation. *Atmospheric Environment*, 1(39):6352–6361.
- Giroldo, M. B., Werninghaus, E., Coelho, E., and Makant, W. (2005). Development of 1.6l flex fuel engine for brazilian market. In *SAE Technical Paper*. SAE International.
- González, C., Gómez, C., Rojas, N., Acevedo, H., and Aristizábal, B. (2017). Relative impact of on-road vehicular and point-source industrial emissions of air pollutants in a medium-sized andean city. *Atmospheric Environment*, 152:279–289.
- Goodwin, T. P., J. A., L. T., and K. R. (2013). Emep/eea air pollutant emission inventory guidebook 2013: Inventory management, improvement

- and qa/qc. Technical report, Technical report 2013, EEA, Copenhagen, Denmark.
- Grell, G. A., Peckham, S. E., Schmitz, R., McKeen, S. A., Frost, G., Skamarock, W. C., and Eder, B. (2005). Fully coupled “online” chemistry within the wrf model. *Atmospheric Environment*, 39(37):6957–6975.
- Hoshyaripour, G., Brasseur, G., Andrade, M., Gavidia-Calderón, M., Bouarar, I., and Ynoue, R. (2016). Prediction of ground-level ozone concentration in são paulo, brazil: Deterministic versus statistic models. *Atmospheric Environment*, 145:365 – 375.
- Ibarra, S., Rehein, A., Vara-Vela, A., and Ynoue, R., editors (2015a). *Simulation For The Metropolitan Region Of Porto Alegre*.
- Ibarra, S., Ynoue, R., and Vara-Vela, A., editors (2015b). *Emissions inventory and atmospheric simulation of 58 urban centers of South America*.
- Ibarra-Espinosa, S. (2017). *Air pollution modeling in São Paulo using bottom-up vehicular emissions inventories*. PhD thesis, University of São Paulo.
- Ibarra-Espinosa, S., Freitas, E., Ynoue, R., de Fátima Andrade, M., and Schuch, D. (2018a). Towards a vectorial global vehicular emissions inventory. In *2018 Joint 14th iCACGP Quadrennial Symposium/15th Igac Science Conference*.
- Ibarra-Espinosa, S., O’Sullivan, S., Osses, M., and Ynoue, R. (2017a). Negative binomial and quasi-poisson regressions of hourly traffic data with openstreetmap. *Journal of Computers, Environment and Urban Systems*. Under Review.
- Ibarra-Espinosa, S., Schuch, D., and de Freitas, E. D. (2017b). *eixport: An R package to export emissions to atmospheric models*. R package version 0.3.4.
- Ibarra-Espinosa, S., Schuch, D., and Dias de Freitas, E. (2018b). eixport: An r package to export emissions to atmospheric models. *The Journal of Open Source Software*.
- Ibarra-Espinosa, S., Ynoue, R., O’Sullivan, S., Pebesma, E., Andrade, M.

- D. F., and Osses, M. (2017c). Vein vo.2.2: an r package for bottom-up vehicular emissions inventories. *Geoscientific Model Development Discussions*, 2017:1–29.
- Ibarra-Espinosa, S., Ynoue, R., O’Sullivan, S., Pebesma, E., Andrade, M. D. F., and Osses, M. (2018c). Vein vo.2.2: an r package for bottom-up vehicular emissions inventories. *Geoscientific Model Development*, 11(6):2209–2229.
- Ihaka, R. and Gentleman, R. (1996). R: a language for data analysis and graphics. *Journal of computational and graphical statistics*, 5(3):299–314.
- J, G. (2013). Emep/eea air pollutant emission inventory guidebook 2013: Key category analysis and methodological choice. Technical report, Technical report 2013, EEA, Copenhagen, Denmark.
- Jimenez, J. L., McClintock, P., McRae, G., Nelson, D. D., and Zahniser, M. S. (1999). Vehicle specific power: A useful parameter for remote sensing and emission studies. In *Ninth CRC On-Road Vehicle Emissions Workshop, San Diego, CA*.
- Koupal, J., Cumberworth, M., Michaels, H., Beardsley, M., and Brzezinski, D. (2003). Design and implementation of moves: Epa’s new generation mobile source emission model. *Ann Arbor*, 1001:48105.
- Lam, W. H. and Xu, J. (2000). Estimation of aadt from short period counts in hong kong—a comparison between neural network method and regression analysis. *Journal of Advanced Transportation*, 34(2):249–268.
- Lents, J., Davis, N., Nikkila, N., and Osses, M. (2004). São paulo vehicle activity study. Technical report, International Sustainable System Research Center, 605 South Palm Street, Suite C, La Habra, CA 90631, USA.
- Lowry, M. (2014). Spatial interpolation of traffic counts based on origin–destination centrality. *Journal of Transport Geography*, 36:98–105.
- Lükewille, A., Tista, M., and Hoenders, R. (2017). European union emission inventory report 1990–2010 under the unece convention on long-range transboundary air pollution (lrtap). Technical report, EEA technical report.

- Martins, L. D., Andrade, M. F., Freitas, E. D., Pretto, A., Gatti, L. V., Albuquerque, É. L., Tomaz, E., Guardani, M. L., Martins, M. H., and Junior, O. M. (2006). Emission factors for gas-powered vehicles traveling through road tunnels in São Paulo, Brazil. *Environ. Sci. Technol.*, 40(21):6722–6729.
- Matloff, N. (2011). *The art of R programming: A tour of statistical software design*. No Starch Press.
- McGlade, J. and Vidic, S. (2009). Emeep/eea air pollutant emission inventory guidebook 2009: Technical guidance to prepare national emission inventories. Technical report, Technical report 9/2009, EEA, Copenhagen, Denmark.
- Mellios, G. and Ntziachristos, L. (2016). Emeep/eea emission inventory guidebook; gasoline evaporation from vehicles. Technical report, European Environment Agency, Copenhagen.
- Metro (2017). Pesquisa origem e destino. Accessed on 10-08-2017.
- Ministério do Meio Ambiente (2011). Primeiro inventário nacional de emissões atmosféricas por veículos automotores rodoviários.
- Molina, M. J. and Molina, L. T. (2004). Megacities and atmospheric pollution. *Journal of the Air & Waste Management Association*, 54(6):644–680.
- Muenchow, J., Schratz, P., and Brenning, A. (2018). Rqgis: Integrating r with qgis for statistical geocomputing. *R Journal*. Accepted for publication on 2017-12-04.
- Nogueira, T., de Souza, K. F., Fornaro, A., de Fatima Andrade, M., and de Carvalho, L. R. F. (2015). On-road emissions of carbonyls from vehicles powered by biofuel blends in traffic tunnels in the metropolitan area of São Paulo, Brazil. *Atmospheric Environment*, 108:88–97.
- Ntziachristos, L. and Boulter, P. (2009). Emeep/eea emission inventory guidebook; road transport: Automobile tyre and break wear and road abrasion. *European Environment Agency, Copenhagen*.

- Ntziachristos, L. and Samaras, Z. (2000). Speed-dependent representative emission factors for catalyst passenger cars and influencing parameters. *Atmospheric Environment*, 34(27):4611–4619.
- Ntziachristos, L. and Samaras, Z. (2016). Emeep/eea emission inventory guidebook; road transport: Passenger cars, light commercial trucks, heavy-duty vehicles including buses and motorcycles. *European Environment Agency, Copenhagen*.
- Ortuzar, J. d. D. and Willumsen, L. G. (2002). *Modelling transport*, volume 3. Wiley.
- P, B., T, B., S, L., and I, M. (2009). Emission factors 2009 report 1. a review of methods for determining hot exhaust emission factors for road vehicles. Technical report, Transport Research Lalboratory.
- Padgham, M. and Peutschchnig, A. (2018). *dodgr Distances on Directed Graphs*. R package version 0.1.0.
- Patrushev, A. (2007). Shortest path search in real road networks with pgRouting. *Free and Open Source Software For Geospatial FOSS*.
- Pebesma, E. (2017). *sf: Simple Features for R*. R package version 0.5-5.
- Pebesma, E., Mailund, T., and Hiebert, J. (2016). Measurement units in R. *The R Journal*, 8(2):486–494.
- Pebesma, E. J. and Bivand, R. S. (2005). Classes and methods for spatial data in R. *R News*, 5(2):9–13.
- Peng, R. D. (2015). *R programming for data science*. Leanpub.com.
- Pérez-Martinez, P., Miranda, R., Nogueira, T., Guardani, M., Fornaro, A., Ynoue, R., and Andrade, M. (2014). Emission factors of air pollutants from vehicles measured inside road tunnels in s ao paulo: case study comparison. *Int. J. Environ. Sci. Te.*, 11(8):2155–2168.
- Pierce, D. (2017). *ncdf4: Interface to Unidata netCDF (Version 4 or Earlier) Format Data Files*. R package version 1.16.
- Pulles, T. and Heslinga, D. (2010). The art of emission inventorying. *TNO, Utrecht*.

- Pulles T, K. J. (2013). Emep/eea air pollutant emission inventory guide-book 2013: Uncertainties. Technical report, Technical report 2013, EEA, Copenhagen, Denmark.
- QGIS Development Team (2017). *QGIS Geographic Information System*. Open Source Geospatial Foundation.
- R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.
- Rafee, S. A. A. (2015). *Estudo numerico do impacto das emissões veiculares e fixas da cidade de Manaus nas concentrações de poluentes atmosféricos da região amazônica*. PhD thesis, Londrina: Universidade Tecnologica Federal do Parana.
- Réquia Jr, W. J., Koutrakis, P., and Roig, H. L. (2015). Spatial distribution of vehicle emission inventories in the federal district, brazil. *Atmospheric Environment*, 112:32–39.
- Sapp, W. (1938). Runctions of a highway-planning survey department. *Annual Road School*.
- Selby, B. and Kockelman, K. M. (2013). Spatial prediction of traffic levels in unmeasured locations: applications of universal kriging and geographically weighted regression. *Journal of Transport Geography*, 29:24–32.
- Snyder, M. G., Venkatram, A., Heist, D. K., Perry, S. G., Petersen, W. B., and Isakov, V. (2013). Rline: A line source dispersion model for near-surface releases. *Atmospheric Environment*, 77:748 – 756.
- Ulke, A. G. and Andrade, M. F. (2001). Modeling urban air pollution in sao paulo, brazil: sensitivity of model predicted concentrations to different turbulence parameterizations. *Atmospheric Environment*, 35(10):1747–1763.
- UNFCCC (2017). *Greenhouse Gas Inventory Data*.
- USA-EPA (2016). Ap42 compilation of emission factors; resuspension emissions from paved roads. *Environment Protection Agency, USA*.

- Vara-Vela, A. (2013). *Avaliação do impacto da mudança dos fatores de emissão veicular na formação de ozônio troposférico na Região Metropolitana de São Paulo(RMSP)*. PhD thesis, Instituto de Astronomia, Geofísica e Ciências Atmosféricas, Universidade de São Paulo, São Paulo, Brazil.
- Vara-Vela, A., Andrade, M. F., Kumar, P., Ynoue, R. Y., and Muñoz, A. G. (2016). Impact of vehicular emissions on the formation of fine particles in the São Paulo metropolitan area: a numerical study with the wrf-chem model. *Atmos. Chem. and Phys.*, 16:777–797.
- Vivanco, M. G. and de Fátima Andrade, M. (2006). Validation of the emission inventory in the São Paulo metropolitan area of Brazil, based on ambient concentrations ratios of CO, NMOC and NO_x and on a photochemical model. *Atmospheric Environment*, 40(7):1189–1198.
- Wang, X. and Kockelman, K. (2009). Forecasting network data: Spatial interpolation of traffic counts from Texas data. *Transportation Research Record: Journal of the Transportation Research Board*, 2105:100–108.
- Weilenmann, M., Favez, J.-Y., and Alvarez, R. (2009). Cold-start emissions of modern passenger cars at different low ambient temperatures and their evolution over vehicle legislation categories. *Atmospheric environment*, 43(15):2419–2429.
- Wickham, H. (2009). *ggplot2: elegant graphics for data analysis*. Springer New York.
- Wickham, H. (2015). *R packages: organize, test, document, and share your code*. ”O'Reilly Media, Inc.”.
- Wickham, H. and Bryan, J. (2017). *readxl: Read Excel Files*. R package version 1.0.0.
- Wickham, H. and Chang, W. (2017). *devtools: Tools to Make Developing R Packages Easier*. R package version 1.13.4.
- Wickham, H. and Henry, L. (2018). *tidyverse: Easily Tidy Data with 'spread()' and 'gather()' Functions*. R package version 0.8.0.
- Wikipedia contributors (2018a). Iso 9000 — Wikipedia, the free encyclopedia. [Online; accessed 16-July-2018].

- Wikipedia contributors (2018b). Quality assurance — Wikipedia, the free encyclopedia. [Online; accessed 16-July-2018].
- Wikipedia contributors (2018c). Quality control — Wikipedia, the free encyclopedia. [Online; accessed 16-July-2018].
- Zaveri, R. A. and Peters, L. K. (1999). A new lumped structure photochemical mechanism for large-scale applications. *Journal of Geophysical Research: Atmospheres*, 104(D23):30387–30415.
- Zeileis, A., Kleiber, C., and Jackman, S. (2008). Regression models for count data in r. *Journal of statistical software*, 27(8):1–25.
- Zhao, F. and Chung, S. (2001). Contributing factors of annual average daily traffic in a florida county: exploration with geographic information system and regression models. *Transportation Research Record: Journal of the Transportation Research Board*, 1(1769):113–122.