CO2MPAS Documentation

Release 4.3.5

Vincenzo Arcidiacono

Nov 15, 2023

TABLE OF CONTENTS

release
4.3.5

rel_date
2023-11-13 15:00:00

home
http://co2mpas.readthedocs.io/

repository
https://github.com/JRCSTU/CO2MPAS-TA

pypi-repo
https://pypi.org/project/co2mpas/

keywords
CO_2, fuel-consumption, WLTP, NEDC, vehicle, automotive, EU, JRC, IET, STU, correlation, back-translation, policy, monitoring, M1, N1, simulator, engineering, scientific

mail box
<JRC-CO2DICE@ec.europa.eu>

team

Model:

- Vincenzo Arcidiacono <vincenzo.arcidiacono@ext.ec.europa.eu>
- Georgios Fontaras <georgios.fontaras@ec.europa.eu>
- Dimitrios Komnos <dimitrios.komnos@ext.ec.europa.eu>
- Jelica Pavlovic <jelica.pavlovic@ext.ec.europa.eu>
- Alessandro Tansini <alessandro.tansini@ec.europa.eu>

Documentation:

- Vincenzo Arcidiacono <vincenzo.arcidiacono@ext.ec.europa.eu>
- Dimitrios Komnos <dimitrios.komnos@ext.ec.europa.eu>
- Francesca Saporiti <francesca.saporiti@ext.ec.europa.eu>
- Alessandro Tansini <alessandro.tansini@ec.europa.eu>

Past contributors:

- Kostis Anagnostopoulos <kostis.anagnostopoulos@ext.ec.europa.eu>
- Biagio Ciuffo <biagio.ciuffo@ec.europa.eu>
- Stefano Corsi <stefano.corsi@ext.ec.europa.eu>
- Spyridon Konstantinidis <Spyridon.konstantinidis@ext.ec.europa.eu>

- Ashenafi Makie <ashenafi.makie@ext.ec.europa.eu>
- Iker Prado Rujas <iker.prado-rujas@ext.ec.europa.eu>
- Stefanos Tsiamakis <stsiakma@gmail.com>
- Victor Valverde <victor.valverde-morales@ec.europa.eu>

CHAPTER

ONE

WHAT IS CO_2MPAS?

CO_2MPAS is backward-looking longitudinal-dynamics CO_2 and fuel-consumption simulator for light-duty M1 & N1 vehicles (cars and vans), specially crafted to *estimate the CO2 emissions of vehicles undergoing NEDC* testing based on the emissions produced *WLTP testing* during *type-approval*, according to the *EU legislation*s *1152/EUR/2017 and 1153/EUR/2017* (see *History* section, below).

It is an open-source project (EUPL 1.1+) developed for Python-3.6+. It runs either as a *console command* or as a *desktop GUI application*, and it uses Excel-files or pure python structures (dictionary and lists) for its input & output data.

1.1 History

The *European Commission* has introduced the *WLTP* as the test procedure for the type I test of the European type-approval of Light-duty vehicles as of September 2017. Its introduction has required the adaptation of CO_2 certification and monitoring procedures set by European regulations (443/2009, 510/2011, 1152/EUR/2017 and 1153/EUR/2017). European Commission's *Joint Research Centre* (JRC) has been assigned the development of this vehicle simulator to facilitate this adaptation.

The European Regulation setting the conditions for using CO_2MPAS can be found in the Comitology Register after its adoption by the *Climate Change Committee* which took place on June 23, 2016, and its 2nd vote for modifications, in April 27, 2017.

CHAPTER

TWO

INSTALLATION

To install CO_2MPAS use (with root privileges):

```
$ pip install co2mpas
```

Or download the latest git version and use (with root privileges):

```
$ python setup.py install
```

2.1 Install extras

Some additional functionality is enabled installing the following extras:

- `cli`: enables the command line interface.
- `sync`: enables the time series synchronization tool (i.e., syncing previously named `datasync`).
- `gui`: enables the graphical user interface.
- `plot`: enables to plot the CO_2MPAS model and the workflow of each run.
- `io`: enables to read/write excel files.
- `driver`: enables the driver model (currently is not available).

To install co2mpas and all extras, do:

```
$ pip install 'co2mpas[all]'
```

CHAPTER

THREE

QUICK START

The following steps are basic commands to get familiar with CO_2MPAS procedural workflow using the command line interface:

- *Run*
- *Input file*
- *Data synchronization*

3.1 Run

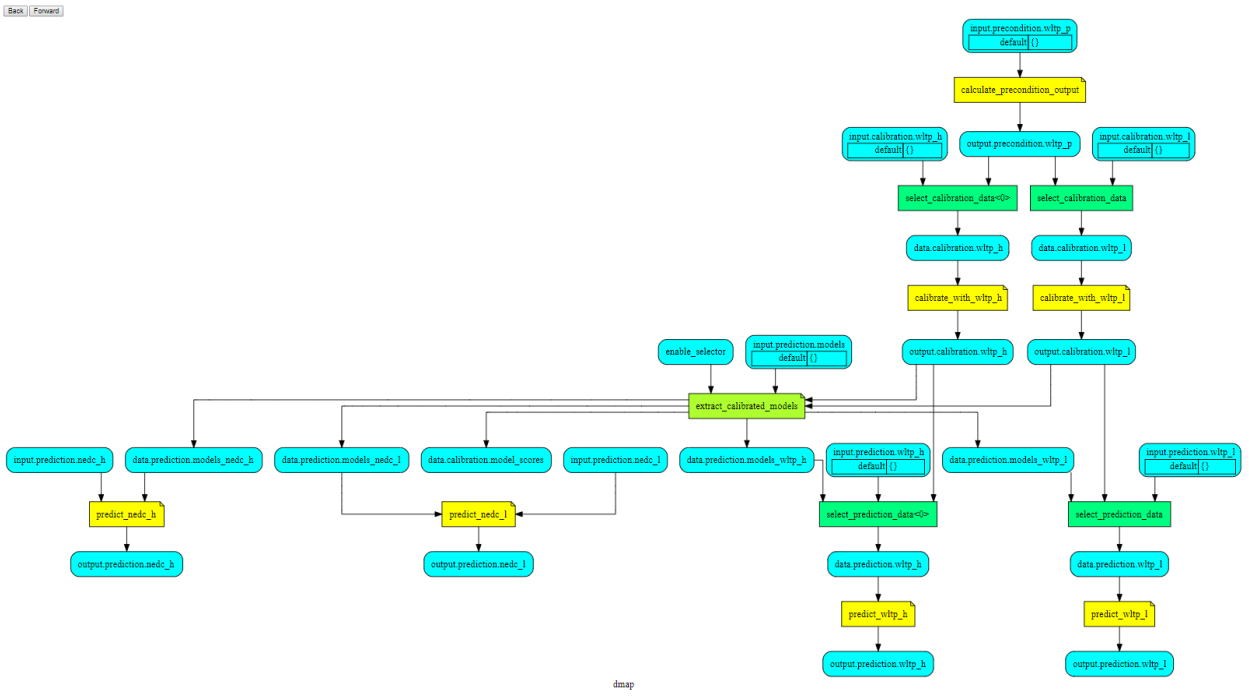
To run CO_2MPAS with some sample data, you have to:

1. Generate some demo files inside the `./input` folder, to get familiar with the input data (for more info check the link):

```
## Generate the demo files and open a demo file.
$ co2mpas demo ./input
$ start ./input/co2mpas_conventional.xlsx
```

2. Run CO_2MPAS and inspect the results in the `./output` folder. The workflow is plotted on the browser (for more info check the link):

```
## Run co2mpas and open the output folder.
$ co2mpas run ./input/co2mpas_conventional.xlsx -O ./output -PL
$ start ./output
```



3.2 Input file

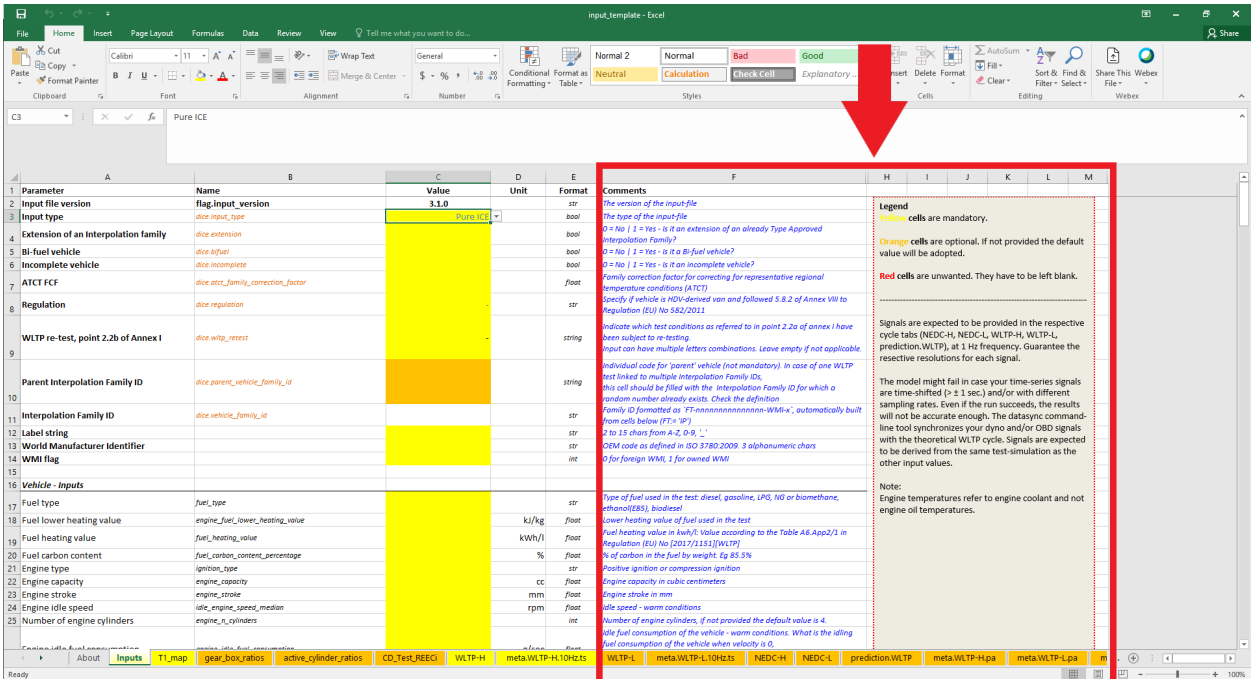
To create an input file with your data, you have to:

1. Generate an empty input template file (i.e., `vehicle.xlsx`) inside the `./input` folder:

```
## Generate template file.
$ co2mpas template ./input/vehicle.xlsx -TT input
```

2. Follow the instructions provided in the excel file to fill the required inputs:

```
## Open the input template.
$ start ./input/vehicle.xlsx
```



3.3 Data synchronization

To synchronize the *dyno* and *OBD* data with the theoretical cycle, you have to:

1. Generate a *synchronization template* file `wltp.xlsx`:

```
## Generate template file.
$ co2mpas syncing template ./to_sync/wltp.xlsx -CT wltp -WC class3b -GB automatic
```

> **Note:** With the command above, the file contains the theoretical `WLTP` velocity profile for an `automatic` vehicle of `class3b`. For more info type `co2mpas syncing template -h` or click the link

2. Fill the `dyno` and `obd` sheets with the relative data collected in the laboratory:

```
## Open the input template.
$ start ./to_sync/wltp.xlsx
```

3. Synchronize the data with the theoretical velocity profile:

```
$ co2mpas syncing sync ./to_sync/wltp.xlsx ./sync/wltp.sync.xlsx
```

4. Copy/Paste the synchronized data (`wltp.sync.xlsx`) contained in the `synced` sheet into the relative sheet of the input template:

```
## Open the synchronized data.
$ start ./sync/wltp.sync.xlsx
```

CHAPTER

FOUR

TUTORIALS

This section explains the functionalities of CO_2MPAS GUI through some video tutorials:

4.1 Inputs

This section shows the utilities to generate and populate the CO_2MPAS input file.

4.1.1 Get input template

Check the video to see how to download an empty input excel-file. The generated file contains the instructions on how to fill the required inputs. For more information use the command `co2mpas template -h` or check the link.

4.1.2 Synchronize time-series

If you have time-series not well synchronized and/or with different sampling rates, the model might fail. As an aid tool, you may use the `syncing` tool to "synchronize" and "re-sample" your data. To use the tool you should execute the following steps:

- Generate and download an *empty* input excel-file (see the video). For more information use the command `co2mpas syncing template -h` or check the link.

  Note: All sheets must contain values for columns `times` and `velocities` because they are the reference signals used to synchronize the data with the theoretical velocity profile.

- Run data synchronization, see the video. For more information use the command `co2mpas syncing sync -h` or check the link.

Note: The synchronized signals are saved into the `synced` sheet.

4.1.3 Download demo files

CO_2MPAS contains 4 demo-files that can be used as a starting point to try out:

1. *co2mpas_conventional.xlsx*: conventional vehicle,
2. *co2mpas_simplan.xlsx*: sample simulation plan,
3. *co2mpas_hybrid.xlsx*: hybrid parallel vehicle,
4. *co2mpas_plugin.xlsx*: hybrid plugin vehicle.

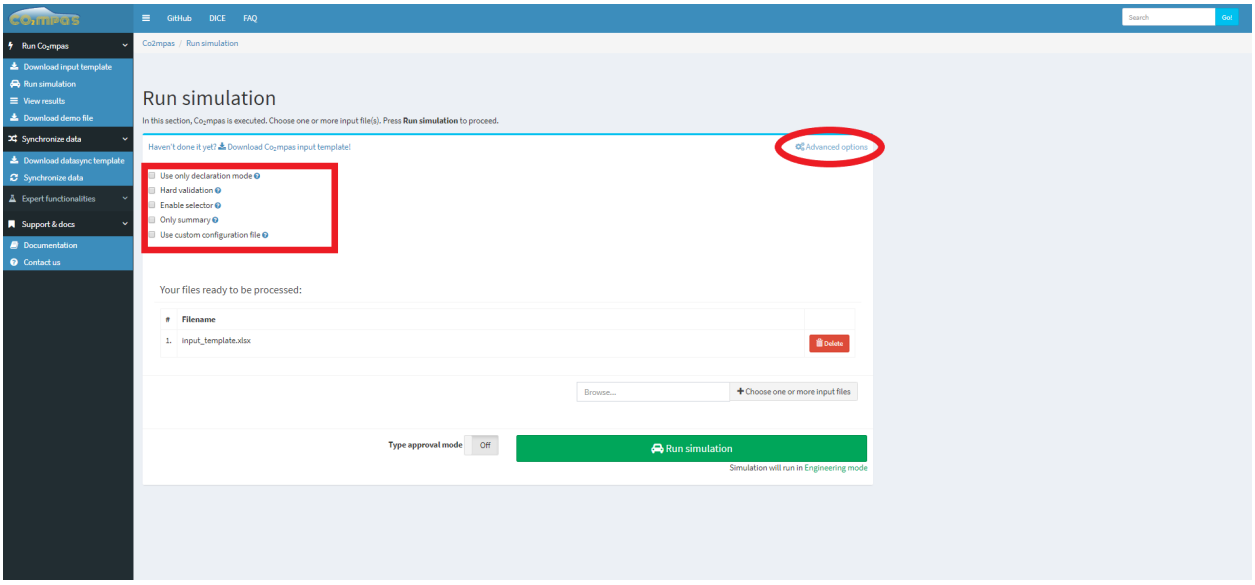
Check the video to see how to download them. For more information use the command `co2mpas demo -h` or check the link.

4.2 Run

This section explains how to run CO_2MPAS:

1. Upload excel file/s (see previous video),
2. click run:

Note: 5 advanced options are available: **use only declaration mode**, **hard validation**, **enable selector**, **only summary**, and **use custom configuration file**. Flag the box to activate them.



3. Get the results (see the previous video).

Output files.

- A CO_2MPAS output per file, named as `<timestamp>-<file-name>.xlsx`.

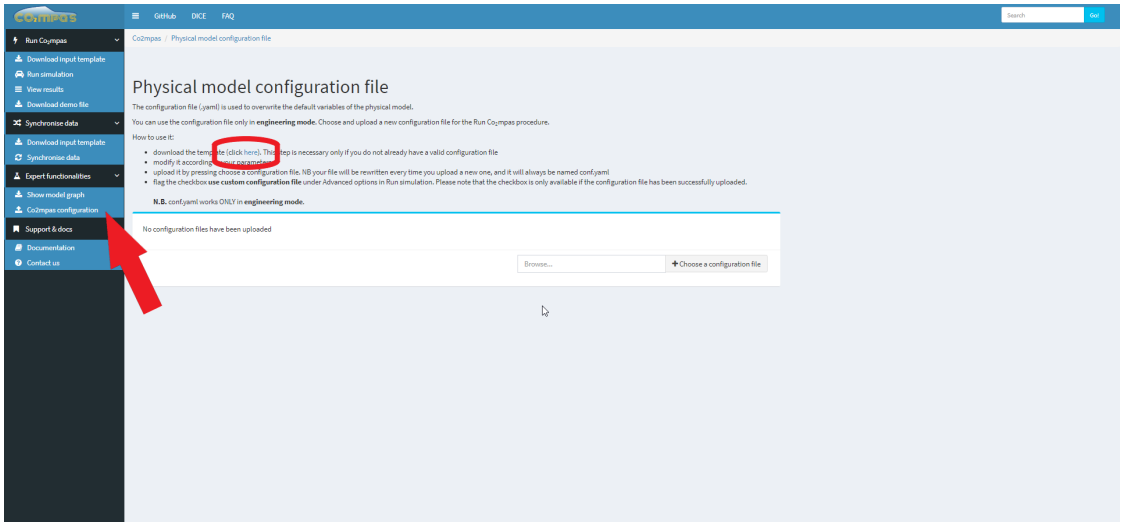
- A summary file like above.

4.2.1 Physical model configuration file

The configuration file (.yaml) is used to overwrite the default variables of the physical model.

Input file:

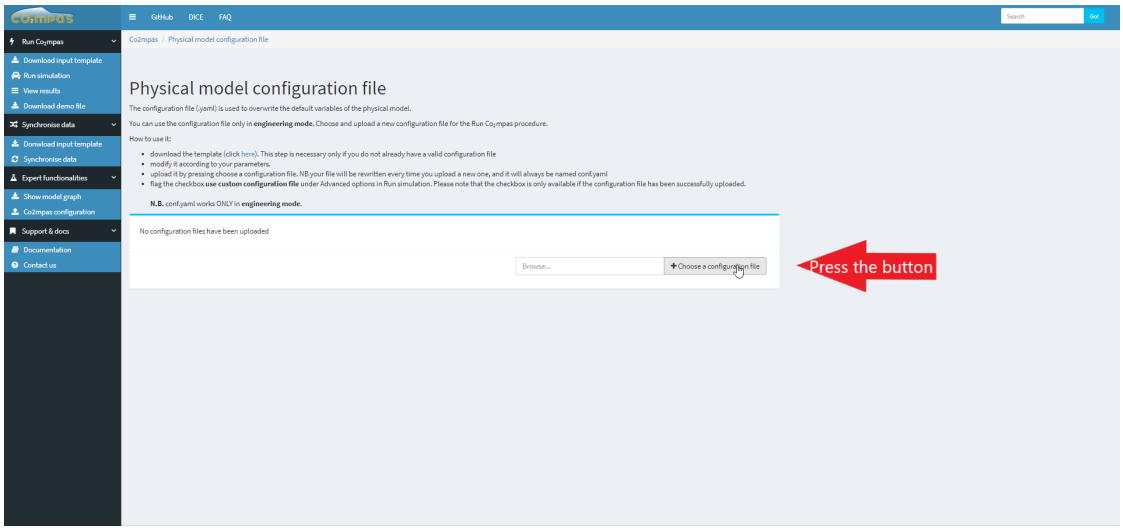
- download the conf.yaml template from the GUI as shown in the image below.



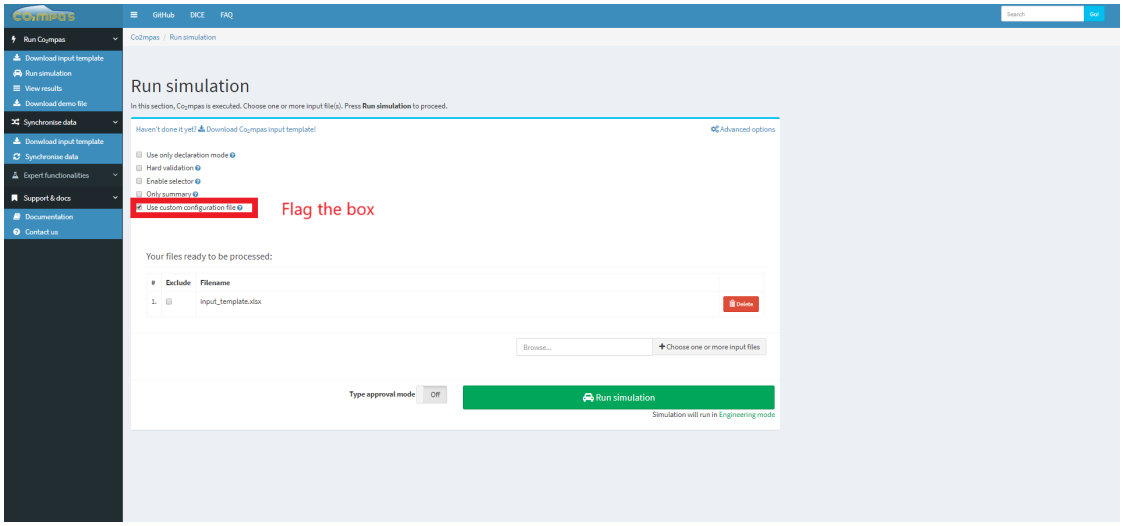
- modify it according to your parameters.

How to use it:

- Upload a new configuration file as shown in the picture.



- Flag the checkbox *use custom configuration file* under **Advanced options**.



conf.yaml

- the file will be rewritten every time you upload a new one, and it will always be named conf.yaml

4.2.2 Simulation plan

The simulation plan is an input file containing some extra parameters/sheets with a **scope** `plan.` (see *data naming convention*). It defines the list of variations (i.e., inputs to be overwritten) that have to be applied to a base dataset (i.e., a normal input file of CO_2MPAS).

The simulation plan can save you time! It is able to calibrate the models just once and re-use them for other subsequent predictions, where only some inputs are different (e.g., `times`, `velocities`, `vehicle_mass`, etc.).

To run the simulation plan upload it as an input file, and run it as described in the previous section.

Output files

- A CO_2MPAS output per **file** like *above*.
- A CO_2MPAS output per **variation**, named as `<timestamp>-<variation-id>-<file-name>.xlsx`.
- A summary file like above.

4.3 Model plot

This section shows the utility to investigate the CO_2MPAS model. For more information check *CO2MPAS Model* and *API Reference*.

CHAPTER

FIVE

CO_2MPAS MODEL

CO2MPAS model is plotted here below: you can explore the diagram nests by clicking on them.

The execution of CO_2MPAS model for a single vehicle is a procedure in three sequential stages:

- **Calibration stage**: identifies, calibrates, and selects the best physical models (see next section *Model selection*) from WLTP input data (i.e., `input.calibration.<cycle>`).
- **Model selection stage**: selects the best calibrated models (i.e., `data.prediction.models_<cycle>`) to be used in the prediction stage.
- **Prediction stage**: forecasts the CO_2 emissions using the user's inputs (i.e., `input.prediction.<cycle>`) and the calibrated models. If some/all WLTP inputs are not provided, the function `select_prediction_data` chooses those required to predict CO_2 emissions from `output.calibration.<cycle>`.

The `physical model` is used in both stages: calibration (i.e., `calibrate_with_<cycle>`) and prediction (i.e., `predict_<cycle>`). The identified/calibrated parameters from WLTP data (i.e., `data.prediction.models_<cycle>`) can be grouped by functionality in eight macro-models:

1. `A/T`: gear shifting strategy for automatic transmission,
2. `electrics`: vehicle electric components (i.e., alternator, service battery, drive battery, and DC/DC converter),
3. `clutch-torque-converter`: speed model for clutch or torque converter,
4. `co2_params`: extended willans lines parameters,
5. `after-treatment`: warm up strategy of after treatment,
6. `engine-coolant-temperature`: warm up and cooling models of the engine,
7. `engine-speed`: correlation from velocity to engine speed,
8. `control`: start/stop strategy or ECMS.

5.1 Model selection

The default model selection criteria (i.e., when `enable_selector == False`) are to use the calibrated models from *WLTP-H* data to predict *WLTP-H* and *NEDC-H* and from *WLTP_L* data to predict *WLTP-L* and *NEDC-L*.

On the contrary, if the selector is enabled, the function `extract_calibrated_model` detects/selects the best macro-model for prediction (from *WLTP-H* or *WLTP-L*). The selection is performed according to the model's score, which is the model capability to reproduce the input data, i.e. a weighted average of all computed metrics.

In other words, the calibrated models are used to recalculate (**predict**) the **inputs** of the *WLTP-H* and *WLTP-L* cycles, while the scores are derived from various metrics comparing **inputs** against **predictions**.

Note: A success flag is defined according to upper or lower limits of scores which have been defined empirically by the JRC. If a score is outside the model fails the calibration and a warning is logged.

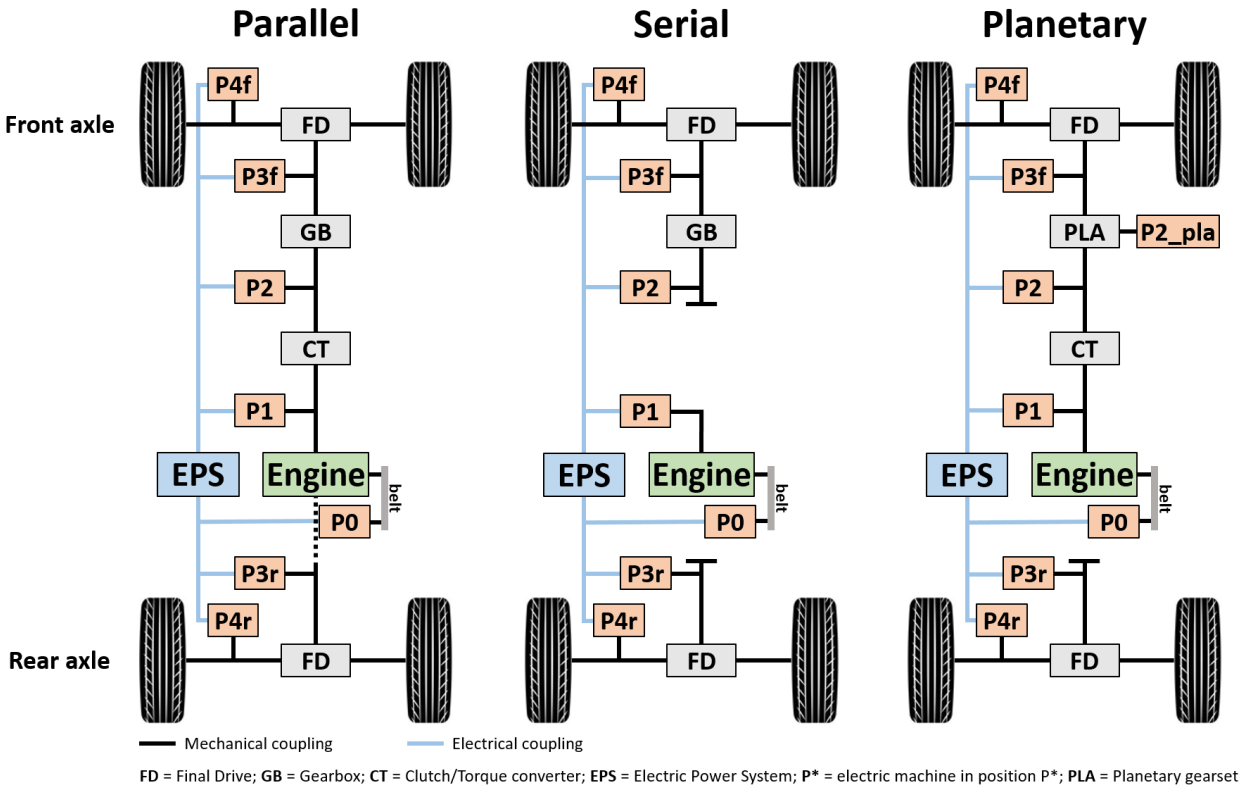
5.2 Hybrid electric vehicles model

Hybrid Electric Vehicles (HEVs) have more complex driveline layouts and control strategies compared to conventional vehicles. Their driveline integrates one or more electric machines for supporting propulsion and/or charging the drive battery, to regenerate braking power, using the engine at a more optimal operating point or even switching it off when appropriate. Therefore, additional components are needed for the simulation of their operation: electric machines, drive battery and DC/DC converter. The vehicle powertrain model of CO_2MPAS is developed in a way to capture and replicate the operation of as many different vehicle architectures as possible through a unique driveline virtual architecture.

5.2.1 Hybrid electric architectures

Three main architectures can be identified for HEVs driveline:

1. **Parallel** hybrids are similar to conventional vehicles as the engine produces mechanical power that is directly used for propulsion; also, engine rotational speed is a function of the rotational speed of the wheels according to the reduction/multiplication effect applied by final drive and gearbox (when present). The electrical machines are used to regenerate braking energy and optimise the load of the engine, but they cannot adjust the rotational speed of the engine when a gear is selected.
2. **Serial** hybrids have an additional degree of freedom for the optimisation, as both the load and the rotational speed of the engine can be selected by the controller. This is possible due to the lack of a mechanical coupling between the engine and the wheels. Therefore, a serial hybrid is always propelled by the electric machines and the engine is used to generate electrical energy that is used for propulsion or battery charging.
3. **Planetary** architecture is instead a driveline configuration that can accomplish, to some extent, the operation of the two architectures previously mentioned. The engine can provide mechanical power that is directly used for propulsion, but at the same time, its rotational speed can be adjusted by the controller becoming independent from wheel speed. This system normally replaces the conventional transmissions (gearboxes and CVTs) as it can adjust the reduction gear ratios to any wanted value, and it integrates two electric machines (one generating and the other one consuming electrical energy).

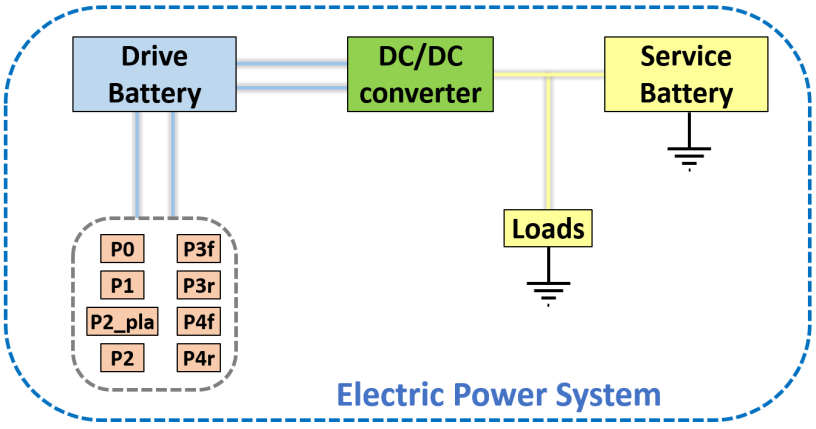


5.2.2 Electric power system

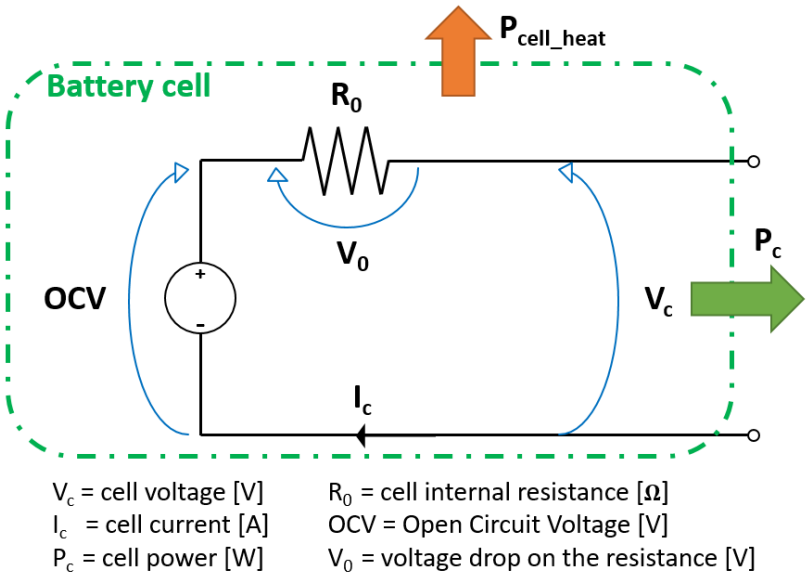
The Electric Power System (EPS) of HEVs is composed by three main components:

- Electric machines (P0, P1, P2, P2_pla, P3f, P3r, P4f, and P4r),
- Batteries (Drive and Service, i.e. high and low voltage batteries), and
- DC/DC converter

The electric machines convert electrical energy into mechanical energy when they need to propel the vehicle and mechanical energy into electrical during regenerative braking or battery charging. This electrical energy, consumed or generated, is exchanged with the drive battery. The DC/DC converter is the component in charge of allowing the energy exchange between the drive battery and the low-voltage electric system of the vehicle, to supply the electrical consumers and charge the service battery when needed.



The efficiency of the **drive battery** is modeled using the **equivalent-circuit cell model** (see image below). The drive battery is seen as a set of battery cells with equal characteristics and size, with a certain combination of cells in series and circuits in parallel. Each cell of the battery suffers of a power loss that is proportional to the cell internal resistance *R0* and the current flowing through it, that is transformed to heat. The performance obtained by the battery is then calculated by considering how many cells in series and parallel are constituting the battery.



5.2.3 Control strategy

The control unit of an HEV runs an optimisation strategy to control the hybrid powertrain and assign the target power to each component (engine and electric machines). The adopted strategy adopted in CO_2MPAS is the **Equivalent Consumption Minimisation Strategy** (**ECMS**), which assigns an equivalent cost - in terms of fuel - to electrical energy use. The strategy evaluates the combination of engine and drive battery power that minimises the overall equivalent energy consumption.

CHAPTER

SIX

DATA NAMING CONVENTIONS

This section describes the data naming convention used in CO_2MPAS input template to construct the *data-values tree*, i.e. the input of the simulation model. There are two naming conventions in the excel file:

- the **sheet-name** name is composed of 5 parts, all optional, but at least one of the first 4 must be present:

```
            base.input.precondition.WLTP-H.ts
            └─ └──┘ └──────────┘ └───┘ └
scope─────────     │          │          │      │
usage─────────────────         │          │      │
stage──────────────────────────           │      │
cycle──────────────────────────────────────      │
sheet_type(optional)─────────────────────────────
```

- the **data-name** consists of 5 parts, the `parameter` part is mandatory, and the last 2 parts can be reversed:

```
               plan.target.prediction.vehicle_mass.WLTP-H
               └─ └──┘ └─────────┘ └──────────┘ └───┘
scope(optional)─     │          │            │           │
usage(optional)────────          │            │           │
stage(optional)──────────────────            │           │
parameter──────────────────────────────────────           │
cycle(optional)──────────────────────────────────────────
```

Note:

- The dot(`.`) may be replaced by space.
- The **usage** and **stage** parts may end with an `s`, denoting plural, and are not case-insensitive, e.g. `Inputs` sheet.

6.1 Description of the name-parts

The options of each name part are described in the following sections.

6.1.1 scope

- `base` [default]: input to CO_2MPAS simulation model.
- `dice`: input for dice.
- `meta`: meta data of the vehicle test.
- `plan`: data used to run a simulation plan and overwrite the inputs of the simulation model.

6.1.2 usage

- `input` [default]: input to calibration and/or to prediction models.
- `data`: intermediate data between calibration and prediction models (see :ref: *model*).
- `output`: CO_2MPAS precondition, calibration, and prediction results.
- `target`: reference-values (**NOT USED NEITHER IN CALIBRATION NOR IN PREDICTION**) to be compared with the CO_2MPAS results.

6.1.3 stage

- `calibration` [default]: calibration data.
- `prediction` [default, if **cycle** in `nedc`]: prediction data.

6.1.4 cycle

- `nedc-h`: *NEDC High* cycle data.
- `nedc-l`: *NEDC Low* cycle data.
- `wltp-h`: *WLTP High* cycle data.
- `wltp-l`: *WLTP Low* cycle data.
- `nedc` [default]: shortcut to set values for `nedc-h` & `nedc-l`.
- `wltp` [default]: shortcut to set values for `wltp-h` & `wltp-l`.
- `all-h`: shortcut to set values for `nedc-h` and `wltp-h`.
- `all-l`: shortcut to set values for `nedc-l` and `wltp-l`.
- `all`: shortcut to set values for `nedc` and `wltp`.

6.1.5 param

Any data-name (e.g. `vehicle_mass`) used in the physical model.

6.1.6 sheet_type

There are three sheet types, which are parsed according to their contained data:

- **pl** [parsed range is `#A1:__`]: table of scalar and time-depended values used into the simulation plan as a variation from the base model.
- **pa** [parsed range is `#B2:C_`]: scalar or not time-depended values (e.g. `r_dynamic`, `gear_box_ratios`, `full_load_speeds`).
- **ts** [parsed range is `#A2:__`]: time-depended values (e.g. `times`, `velocities`, `gears`). Columns without values are skipped. **COLUMNS MUST HAVE THE SAME LENGTH!**

Note: If it is not defined, the default value follows these rules: When **scope** is `plan`, the sheet is parsed as **pl**. If **scope** is `base` and **cycle** is missing in the **sheet-name**, the sheet is parsed as **pa**, otherwise, it is parsed as **ts**.

6.2 Simulation plan

Variations to the base model must be inserted in the provided additional sheets, characterized by the plan. prefix. These sheets consist of tables composed by rows, each row is a single simulation, and columns, the parameters that the user wishes to vary; the columns can contain the following special names:

- **id**: Identifies the variation id.
- **base**: It is the file path of a CO2MPAS excel input. The data are used as new base vehicle.
- **run_base**: If TRUE [default] the base model results are computed and stored, otherwise the data are just loaded.

CHAPTER

SEVEN

FAQ

This page contains the most Frequently Asked Questions regarding CO_2MPAS model, regulation and inputs.

For more questions, please visit: https://github.com/JRCSTU/CO2MPAS-TA/wiki/FAQ

7.1 General

7.1.1 Where can the user download the latest version of the CO_2MPAS?

To download the CO_2MPAS latest version click here. To be notified of every CO_2MPAS release, **watch** our GitHub repository.

7.1.2 Is CO_2MPAS free and will it be in the future?

CO_2MPAS is and will be free. To maintain it under the current EUPL license, any modifications made to the program, or a copy of it, must be licensed in the same way: EUPL.

7.1.3 What is CO_2MPAS physical background and which formulas are applied?

CO_2MPAS is a backward-looking longitudinal-dynamics CO_2 and fuel-consumption simulator for light-duty M1 & N1 vehicles (cars and vans). To check the formulas the user can visit the functions' description pages.

7.1.4 Where can the user find information on the status of the validation?

Detailed validation reports are provided together with every major release of CO_2MPAS in the validation chapter of the wiki.

Validation is performed as well by independent contractors (LAT) of DG CLIMA. Moreover, some stakeholders have conducted independent validation tests on real cars in collaboration with the JRC. The results of these tests have been included in the above-mentioned reports as "real cars".

7.1.5 Where can the user find CO_2MPAS workshop material?

Workshop material is always uploaded in the presentation chapter.

7.2 Model

7.2.1 What is the Data synchronization tool and how does it work?

Synchronization of data from different sources is essential for robust results. CO_2MPAS *syncing* tool uses a common signal as a reference. To avoid time-aligned signals, we advise using the velocity present on the dyno and the obd at the same time. *syncing* tool will shift and re-sample the other signals according to the reference signal provided. The user is allowed to apply different ways of re-sampling the original signals. For more information, please see the instructions.

7.2.2 What is the model selector?

CO_2MPAS consists of several models. If the user provides both WLTP-H and WLTP-L data, the same models will be calibrated twice, according to the data provided by each configuration. If the option *model selector* is switched on, CO_2MPAS will use the model that provides the best scores, no matter if the model was calibrated with another cycle. For example, if the alternator model of the High configuration is better, the same model will be used to predict the Low configuration as well.

7.2.3 Is it possible to simulate other cycles than NEDC or WLTP? How about real on-road tests?

Yes, CO_2MPAS can simulate other cycles, as well as on-road tests. The user can simulate using several extra parameters beyond the official laboratory-measured ones. The user can input the velocity profile followed, road grade, extra auxiliaries losses, extra passengers, different road loads, temperatures, etc. The user will find an example file when downloading the demo files.

7.2.4 Is the usage of internal / development signals allowed (if equivalence is shown)?

OBD signals are regulated and the only one to be used.

7.2.5 What is the start-stop (S/S) activation time?

S/S is the time elapsed from the beginning of the NEDC test to the first time the Start-Stop system is enabled, expressed in seconds [s].

7.2.6 How to insert a new *drive_battery_technology* for hybrid electric vehicles?

The parameter already contains a preselection of drive batteries technologies as a drop-down menu. If you need to insert a different technology, you should remove the "data validation rule" of the excel input file, insert the new data and proceed with the co2mpas run.

CHAPTER

EIGHT

GLOSSARY

this page lists and explains the terms used in the input file.

8.1 GENERAL

extension
Expansion of the interpolation line (i.e. extension of the CO_2 values). It cannot be performed for any other purposes (EVAP, etc.). It is defined in section 3 of Annex I of Regulation (EC) No 692/2008.

bifuel
A vehicle with multi-fuel engine, capable of running on two fuels.

incomplete
A vehicle that must undergo at least one further stage of completion before it can be considered complete.

atct_family_correction_factor
Family correction factor used to correct for representative regional temperature conditions (ATCT).

wltp_retest
It indicates which test conditions have been subject to retesting (see point 2.2a of Annex I of Regulation (EU) 2018/2043). Input can have multiple letters combination, leave it empty if not applicable.

parent_vehicle_family_id
The family identifier code of the parent.

regulation
It indicates if WLTP test has been performed in accordance with point 5.1.2 of Annex VIII to Regulation (EU) No 582/2011.

vehicle_family_id
The family identifier code shall consist of one unique string of n-characters and one unique WMI code followed by '1'.

input_type
It indicates the data input type, i.e. *Pure ICE*, *NOVC-HEV*, or *OVC-HEV.*

comments
A field to add comments regarding the DICE procedure. In case of extension, or resubmission, kindly provide a detailed description.

8.2 Model Inputs

fuel_type
It refers to the type of fuel used during the vehicle test. The user must choose the correct one among the following:

- diesel,
- gasoline,
- LPG,
- NG,
- ethanol,
- methanol,
- biodiesel, and
- propane.

engine_fuel_lower_heating_value
Lower heating value of the fuel used in the test, expressed in [kJ/kg] of fuel.

fuel_heating_value
Fuel heating value in kwh/l: Value according to Table A6.App2/1 in Regulation (EU) No [2017/1151][WLTP].

fuel_carbon_content_percentage
The amount of carbon present in the fuel by weight, expressed in [%].

ignition_type
It indicates whether the engine of the vehicle is a *spark ignition* (= *positive ignition*) or a *compression ignition* one.

engine_capacity
The total volume of all the cylinders of the engine expressed in cubic centimeters [cc].

engine_stroke
It is the full travel of the piston along the cylinder, in both directions. It is expressed in [mm].

idle_engine_speed_median
It represents the engine speed in warm conditions during idling, expressed in revolutions per minute [rpm]. It can be measured at the end of a WLTP test.

engine_n_cylinders
It specifies the maximum number of the engine cylinders. The default is *4*.

engine_idle_fuel_consumption
It measures the fuel consumption of the vehicle in warm conditions during idling. The idling fuel consumption of the vehicle, expressed in grams of fuel per second [gFuel/sec] should be measured when:

- the velocity of the vehicle is 0,
- the start-stop system is disengaged,
- the battery state of charge is at balance conditions.

For CO_2MPAS purposes, the engine idle fuel consumption can be measured as follows: just after a WLTP physical test, when the engine is still warm, leave the vehicle to idle for 3 minutes so that it stabilizes. Then make a constant measurement of fuel consumption for 2 minutes. Disregard the first minute, then calculate idle fuel consumption as the average fuel consumption of the vehicle during the subsequent 1 minute.

final_drive_ratio
It is the ratio of gearbox output shaft to driven wheel revolutions. If the vehicle has more than one final drive ratio, it has to be left blank and use the `final_drive_ratios`.

final_drive_ratios
It specifies the final-drive ratios for each gear.

tyre_code
The code of the tyres used in the WLTP/NEDC test (e.g., P195/55R16 85H). CO_2MPAS does not require the full tyre code to work, however at least provide the following information (e.g., 195/55R16):

- nominal width of the tyre, in [mm];
- the ratio of height to width [%]; and
- the load index.

In case that the front and rear wheels are equipped with tyres of different radius (tyres of different width do not affect CO_2MPAS), then the size of the tyres fitted in the powered axle should be declared as input to CO_2MPAS. For vehicles with different front and rear wheels tyres tested in 4x4 mode, then the size of the tyres from the wheels where the OBD/CAN vehicle speed signal is measured should be declared as input to CO_2MPAS.

gear_box_type
The type of gearbox among automatic transmission, manual transmission, continuously variable transmission (CVT) or planetary (exclusively for hybrid vehicles fitted with a planetary gearset).

start_stop_activation_time
It is the time elapsed from the beginning of the NEDC test to the first time the Start-Stop system is enabled, expressed in seconds [s].

alternator_nominal_voltage
Alternator nominal voltage [V], i.e. the nominal voltage of the service battery.

alternator_nominal_power
Alternator maximum power [kW], i.e. the rated power of the electric machine.

service_battery_capacity
Capacity [Ah] of the service battery, e.g. the low voltage battery.

service_battery_nominal_voltage
Service battery nominal voltage [V] as described in Appendix 2 to Sub-Annex 6 to Annex XXI to Regulation (EU) No [2017/1151][WLTP].

initial_temperature
The initial temperature of the test cell during the test. It is used to calibrate the thermal model. The default value is *23* °C for WLTP and *25* °C for NEDC.

alternator_efficiency
It is the ratio of electrical power out of the alternator to the mechanical power put into it. If not expressed by the manufacturer, then it is by default *0.67*.

gear_box_ratios
It defines the ratios of engine to gearbox output shaft revolutions.

full_load_speeds
They are rotational speed setpoints defining the engine full load curve expressed in *RPM*.

full_load_powers
They are power values defining the engine full load curve expressed in *kW*.

vehicle_mass

- For the WLTP: it is the simulated inertia applied during the test on the dyno [kg]. It should reflect correction for rotational mass m_r as foreseen by WLTP regulation for 1-axle chassis dyno testing. (Regulation 2017/1151; Sub-Annex 4; paragraph 2.5.3).
- For the NEDC: it is the inertia [kg] class of NEDC without the correction for rotating parts [kg].

f0

It corresponds to the rolling resistance force [N] when the angle slope is 0 applied to the Dyno during the test cycle. This is defined by Dyno procedure.

f1

It corresponds to the resistance $[\frac{N}{kmh}]$ function of the velocity applied to the Dyno during the test cycle. This is defined by Dyno procedure.

f2

It corresponds to the aerodynamic resistance $[\frac{N}{kmh^2}]$ applied to the Dyno during the test cycle. This is defined by Dyno procedure.

co2_emission_low

It is the CO_2 emissions bag value [g|CO2|/km] of WLTP low phase, not corrected for RCB and not rounded.

co2_emission_medium

It is the CO_2 emissions bag value [g|CO2|/km] of WLTP medium phase not corrected for RCB and not rounded.

co2_emission_high

It is the CO_2 emissions bag value [g|CO2|/km] of WLTP high phase not corrected for RCB and not rounded.

co2_emission_extra_high

It is the CO_2 emissions bag value [g|CO2|/km] of WLTP extra high phase not corrected for RCB and not rounded.

depleting_co2_emission_value

It is the combined CO_2 emissions value [g|CO2|/km] of the charge depleting tests.

fuel_consumption_value

It is the combined fuel consumption [l/100km] of the test not corrected.

sustaining_fuel_consumption_value

It is the combined fuel consumption [l/100km] of the charge sustaining test not corrected.

rcb_correction

It says if the RCB correction has to be (or has been) performed.

speed_distance_correction

It says if the speed distance correction has to be (or has been) performed.

corrected_co2_emission_value

It is the combined CO_2 emissions value [g|CO2|/km] corrected for RCB (if applicable), speed & distance (if applicable), Ki factor (if applicable), and ATCT (MCO2, C, 5 values from appendix 4 to Annex I to Regulation (EU) 2017/1151).

corrected_sustaining_co2_emission_value

It is the combined CO_2 emissions value [g|CO2|/km] of the charge sustaining test corrected for RCB (if applicable), speed & distance (if applicable), Ki factor (if applicable), and ATCT (MCO2, C, 5 values from appendix 4 to Annex I to Regulation (EU) 2017/1151).

declared_co2_emission_value

It is the declared CO_2 emissions value [g|CO2|/km]. Value should be corrected for RCB (if applicable), speed & distance (if applicable), Ki factor (if applicable), and ATCT (MCO2, C, 5 values from appendix 4 to Annex I to Regulation (EU) 2017/1151).

declared_sustaining_co2_emission_value

It is the declared CO_2 emissions value [g|CO2|/km] of the charge sustaining test. Value should be corrected for

RCB (if applicable), speed & distance (if applicable), Ki factor (if applicable), and ATCT (MCO2, C, 5 values from appendix 4 to Annex I to Regulation (EU) 2017/1151).

declared_depleting_co2_emission_value
It is the declared CO_2 emissions value [g|CO2|/km] of the charge depleting tests. Value should be corrected for RCB (if applicable), speed & distance (if applicable), Ki factor (if applicable), and ATCT (MCO2, C, 5 values from appendix 4 to Annex I to Regulation (EU) 2017/1151).

transition_cycle_index
Index of the transition cycle according to entry 2.1.1.4.1.4 of Appendix 8a to Annex I to Regulation (EU) 2017/1151. The transition cycle is the cycle before the confirmation cycle (where the break-off criterion is satisfied) in the charge-depleting sequence. In the transition cycle the operation of the vehicle can be partly charge-depleting and partly charge-sustaining.

relative_electric_energy_change
The Relative Electric Energy Change (REEC) is a measure of the discharge of the vehicle traction REESS during the Charge Depleting test. It is calculated as the energy battery balance over the cycle divided by cycle energy, according to paraghraph 3.2.4.5.2 of Sub-Annex 8 to Annex XXI to Regulation (EU) 2017/1151.

wltp_electric_range
The cycle-specific equivalent all-electric range (EAER) is an indication of the distance that the vehicle can drive using electric energy, according to paraghraph 4.4.4 of Sub-Annex 8 to Annex XXI to Regulation (EU) 2017/1151.

nedc_electric_range
The NEDC electric range, calculated according to paragraph 4.2.2.1 of Annex 9 to UN Regulation 101, is an indication of the distance that the vehicle can drive using electric energy.

n_wheel_drive
It specifies whether the test is conducted on 2-wheel driving or 4-wheel driving.

engine_is_turbo
It specifies if the air intake of the engine is equipped with any kind of forced induction system set like a turbocharger or supercharger.

has_start_stop
It specifies if the start-stop system shuts down the engine of the vehicle during idling to reduce fuel consumption and it restarts it again when the footbrake/clutch is pressed.

has_energy_recuperation
It specifies if the vehicle is equipped with any kind of brake energy recuperation technology or regenerative breaking.

has_torque_converter
It specifies if the vehicle is equipped with a torque converter.

fuel_saving_at_strategy
It allows CO_2MPAS to use gear at constant speed driving higher than when in transient conditions, resulting in a reduction of the fuel consumption.

has_periodically_regenerating_systems
It specifies if the vehicle is equipped with periodically regenerating systems (anti-pollution devices such as catalytic converter or particulate trap). During cycles where regeneration occurs, emission standards need not apply. If a periodic regeneration occurs at least once per Type 1 test and has already occurred at least once during vehicle preparation or the distance between two successive periodic regenerations is more than 4000 km of driving repeated Type 1 tests, it does not require a special test procedure. In this case, Ki factor should be set to 1.0 (`ki_multiplicative`), or 0.0 (`ki_additive`).

engine_has_variable_valve_actuation
It specifies if the engine is equipped with technologies that are used to enable variable valve event timing, duration

and/or lift. For example, Valve Timing Control (VTC) — also referred to as Variable Valve Timing (VVT) systems - and Variable Valve Lift (VVL) or a combination of these systems (phasing, timing and lift variation).

has_engine_idle_coasting
It specifies if the engine is allowed to idle during vehicle coasting in order to save fuel.

has_engine_off_coasting
It specifies if the engine is allowed to turn off during vehicle coasting in order to save fuel.

engine_has_cylinder_deactivation
It specifies if the engine has a cylinder deactivation system. If yes provide the active cylinder ratios in the tab *active_cylinder_ratios*.

active_cylinder_ratios
They are the plausible deactivation ratios. For example, in the case of an 8-cylinder engine, a 50% deactivation (4 cylinders off) or a 25% deactivation ratio (2 cylinders off) are plausible.

Note that the *active_cylinder_ratios* always start with 1 (all cylinders are active) and then the user can set the corresponding plausible ratios.

has_lean_burn
It specifies if the vehicle has lean-burn (LB) technology. This technology refers to the burning of fuel with an excess of air in an internal combustion engine.

has_gear_box_thermal_management
It specifies if the temperature of the gearbox is regulated from the vehicle's cooling circuit using a heat-exchanger, heating storage system or other methods for directing engine waste-heat to the gearbox. Gearbox mounting and other passive systems (encapsulation) should not be considered.

has_exhausted_gas_recirculation
It specifies if a portion of an engine's exhaust gas back to the engine cylinders to reduce NO_x emissions. The technology does not concern internal (in-cylinder) EGR.

has_selective_catalytic_reduction
It specifies if the vehicle has the Selective Catalytic Reduction (SCR) system active (Urea), or passive (Ammonia) to reduce NO_x emissions.

n_dyno_axes
It defines the Dyno rotating axis used during the test.

kco2_wltp_correction_factor
CO_2-emission correction coefficient (KCO2) for charge sustaining battery energy balance correction. Paragraph 2.3.2 of Appendix 2 of Sub-Annex 8 to Annex XXI to Regulation (EU) 2017/1151.

kco2_nedc_correction_factor
CO_2-emission correction coefficient (KCO2) for charge sustaining battery energy balance correction. Paragraph 5.3.5 of Annex 8 of UNECE Regulation No. 101 Rev.3.

planetary_ratio
It is the ratio existing between the planetary speed and the final drive speed during electric drive (engine speed =0). The planetary speed is the rotational speed of the planetary gearset side that is not the engine nor the final drive side (the branch that goes to the motor P2 planetary, referred to as the planetary side in this documentation).

initial_drive_battery_state_of_charge
It is the initial state of charge of the drive battery at the beginning of the test.

drive_battery_n_cells
It is the number of cells of the drive battery.

drive_battery_technology
If is the technology of the drive battery. The technologies included in CO_2MPAS are:

- NiMH: Nickel-metal hydride
- Li-NCA (Li-Ni-Co-Al): Lithium Nickel Cobalt Aluminum Oxide
- Li-NCM (Li-Ni-Mn-Co): Lithium Nickel Manganese Cobalt Oxide
- Li-MO (Li-Mn): Lithium Manganese Oxide
- Li-FP (Li-Fe-P): Lithium Iron Phosphate
- Li-TO (Li-Ti): Lithium Titanate Oxide

drive_battery_capacity
Capacity [Ah] of the drive battery, e.g. the high voltage battery.

drive_battery_nominal_voltage
Drive battery nominal voltage [V], e.g. the nominal voltage of the high voltage battery.

motor_p0_maximum_power
Maximum power (i.e., the rated power) output of motor P0 [kW].

motor_p0_maximum_torque
Maximum torque output of motor P0 [Nm].

motor_p0_speed_ratio
The ratio between motor P0 speed and engine speed [-] (e.g. motor P0 connected to the engine belt with ratio equal to 3 is spinning three times faster than the engine).

motor_p1_maximum_power
Maximum power (i.e., the rated power) output of motor P1 [kW].

motor_p1_maximum_torque
Maximum torque output of motor P1 [Nm].

motor_p1_speed_ratio
The ratio between motor P1 speed and engine speed [-] (e.g. motor P1 connected to the engine crankshaft with ratio equal to 3 is spinning three times faster than the engine).

motor_p2_maximum_power
Maximum power (i.e., the rated power) output of motor P2 [kW].

motor_p2_maximum_torque
Maximum torque output of motor P2 [Nm].

motor_p2_speed_ratio
The ratio between motor P2 speed and transmission input speed [-] (motor P2 speed is proportional to wheels rotational speed multiplied by the final drive ratio and the transmission gear ratio).

motor_p2_planetary_maximum_power
Maximum power (i.e., the rated power) output of motor P2 planetary [kW].

motor_p2_planetary_maximum_torque
Maximum torque output of motor P2 planetary [Nm].

motor_p2_planetary_speed_ratio
The ratio between planetary motor P2 speed and planetary side (branch that goes to planetary motor P2) speed.

motor_p3_front_maximum_power
Maximum power (i.e., the rated power) output of motor P3 front [kW].

motor_p3_front_maximum_torque
Maximum torque output of motor P3 front [Nm].

motor_p3_front_speed_ratio
The ratio between motor P3 front speed and final drive input speed [-] (motor P3 front speed is equal to wheels

rotational speed multiplied by the final drive ratio and), where final drive input speed is the rotational speed of the shaft downstream the gearbox (therefore it's part of the engine driveline).

motor_p3_rear_maximum_power
Maximum power (i.e., the rated power) output of motor P3 rear [kW].

motor_p3_rear_maximum_torque
Maximum torque output of motor P3 rear [Nm].

motor_p3_rear_speed_ratio
The ratio between motor P3 rear speed and final drive input speed [-] (motor P3 rear speed is proportional to wheels rotational speed multiplied by the final drive ratio), where final drive input speed is the rotational speed of the shaft downstream the gearbox (therefore it's part of the engine driveline).

motor_p4_front_maximum_power
Maximum power (i.e., the rated power) output of motor P4 front [kW]. When two P4 motors are present on the same axle, their specifications have to be combined to obtain an equivalent single motor in P4 position.

motor_p4_front_maximum_torque
Maximum torque output of motor P4 front [Nm]. When two P4 motors are present on the same axle, their specifications have to be combined to obtain an equivalent single motor in P4 position.

motor_p4_front_speed_ratio
The ratio between motor P4 front speed and wheels speed [-] (motor P4 front speed is proportional to wheels rotational speed).

motor_p4_rear_maximum_power
Maximum power (i.e., the rated power) output of motor P4 rear [kW]. When two P4 motors are present on the same axle, their specifications have to be combined to obtain an equivalent single motor in P4 position.

motor_p4_rear_maximum_torque
Maximum torque output of motor P4 rear [Nm]. When two P4 motors are present on the same axle, their specifications have to be combined to obtain an equivalent single motor in P4 position.

motor_p4_rear_speed_ratio
The ratio between motor P4 rear speed and wheels speed [-] (motor P4 rear speed is proportional to wheels rotational speed).

8.3 Time Series

times
It is the time vector [s].

velocities
It is the actual vehicle speed vector [km/h] from the dynamometer.

obd_velocities
It is the actual vehicle speed vector [km/h] from the OBD.

gears
It is the actual gear vector [-]. If the name of the parameter is *target.calibration.gears* it refers to the theoretical gears calculated according to Heinz Steven tool [-].

bag_phases
It is the array to associate time values with different bag phases (this can be used to modify the duration of the phases from the default values).

engine_speeds_out
It is the actual engine rotational speed vector [rpm] from the OBD.

engine_coolant_temperatures
It is the actual engine coolant temperature vector [°C] from the OBD.

co2_normalization_references
It is the normalization reference for CO_2 emissions (e.g. engine load, engine power output).

alternator_currents
It is the current vector produced by the alternator [A] (current is negative when the alternator is supplying power to the low-voltage electrical system).

service_battery_currents
It is the current vector flowing through the service battery [A] (current is positive when the battery is being charged, negative when discharged).

drive_battery_voltages
It is the voltage vector of the drive battery [V].

drive_battery_currents
It is the current flowing through the drive battery [A] (current is positive when the battery is being charged, negative when discharged).

dcdc_converter_currents
It is the current flowing through the DCDC converter measured on the low-voltage side [A] (current is negative when the DCDC converter is supplying power to the low-voltage electrical system).

8.4 General Terms

type-approval
It is the authority that grants that a vehicle conforms to the EU Regulation.

EU legislation
COMMISSION IMPLEMENTING REGULATION (EU) 2017/1152: sets out a methodology for determining the correlation parameters necessary for reflecting the change in the regulatory test procedure with regard to light commercial vehicles. COMMISSION IMPLEMENTING REGULATION (EU) 2017/1153: sets out a methodology for determining the correlation parameters necessary for reflecting the change in the regulatory test procedure and amending Regulation (EU) No 1014/2010.

CHAPTER

NINE

CONTRIBUTING TO CO2MPAS

If you want to contribute to CO_2MPAS and improve it, your help is very welcome. Your contribution should be sent by a *pull request*. Next sections explains how to implement and submit a new functionality:

- clone the repository
- implement a new functionality
- open a pull request

9.1 Clone the repository

The first step to contribute to CO_2MPAS is to clone the repository:

- Create a personal fork of the co2mpas repository on Github.
- Clone the fork on your local machine. Your remote repo on Github is called `origin`.
- Add the original repository as a remote called `upstream`, to maintain updated your fork.
- If you created your fork a while ago be sure to pull `upstream` changes into your local repository.
- Create a new branch to work on! Branch from `dev`.

9.2 How to implement a new functionality

Test cases are very important. This library uses a data-driven testing approach. To implement a new function I recommend the test-driven development cycle. Hence, when you think that the code is ready, add new test in `test` folder.

When all test cases are ok (`python setup.py test`), open a pull request.

Note: A pull request without new test case will not be taken into consideration.

9.3 How to open a pull request

Well done! Your contribution is ready to be submitted:

- Squash your commits into a single commit with git's interactive rebase. Create a new branch if necessary. Always write your commit messages in the present tense. Your commit message should describe what the commit, when applied, does to the code – not what you did to the code.
- Push your branch to your fork on Github (i.e., `git push origin dev`).
- From your fork open a *pull request* in the correct branch. Target the project's `dev` branch!
- Once the *pull request* is approved and merged you can pull the changes from `upstream` to your local repo and delete your extra branch(es).

CHAPTER

TEN

API REFERENCE

The core of the library is composed from the following modules: Defines the file processing chain model *dsp*.

cli	Define CO2MPAS command line interface.
utils	It contains classes and functions of general utility.
defaults	Constants for the CO2MPAS physical model.

10.1 cli

Define CO2MPAS command line interface.

10.1.1 co2mpas

CO2MPAS command line tool.

```
co2mpas [OPTIONS] COMMAND [ARGS]...
```

Options

--version

Show the version and exit.

conf

Writes a CO2MPAS model-configuration file into OUTPUT_FILE.

OUTPUT_FILE: File path *.yaml*. [default: ./conf.yaml]

```
co2mpas conf [OPTIONS] [OUTPUT_FILE]
```

Options

-MC, --model-conf <model_conf>
Model-configuration file path *.yaml*.

-v, --verbosity <LVL>
Either CRITICAL, ERROR, WARNING, INFO or DEBUG

Arguments

OUTPUT_FILE
Optional argument

demo

Writes a CO2MPAS demo files into OUTPUT_FOLDER.

OUTPUT_FOLDER: Folder path. [default: ./inputs]

```
co2mpas demo [OPTIONS] [OUTPUT_FOLDER]
```

Options

-v, --verbosity <LVL>
Either CRITICAL, ERROR, WARNING, INFO or DEBUG

Arguments

OUTPUT_FOLDER
Optional argument

plot

Plots the full CO2MPAS model into CACHE_FOLDER.

```
co2mpas plot [OPTIONS]
```

Options

-C, --cache-folder <cache_folder>
Folder to save temporary html files.

Default
`./cache_plot`

-H, --host <host>
Hostname to listen on.

Default
`127.0.0.1`

-P, --port <port>
Port of the webserver.

Default
5000

-v, --verbosity <LVL>
Either CRITICAL, ERROR, WARNING, INFO or DEBUG

run

Run CO2MPAS for all files into INPUT_FILES.

INPUT_FILES: List of input files and/or folders
(format: .xlsx, .dill, .co2mpas.ta, .co2mpas).

```
co2mpas run [OPTIONS] [INPUT_FILES]...
```

Options

-O, --output-folder <output_folder>
Output folder.

Default
./outputs

-EK, --encryption-keys <encryption_keys>
Encryption keys for TA mode.

Default
./DICE_KEYS/dice.co2mpas.keys

-SK, --sign-key <sign_key>
User signature key for TA mode.

Default
./DICE_KEYS/sign.co2mpas.key

-C, --cache-folder <cache_folder>
Folder to save temporary html files.

Default
./cache_plot

-H, --host <host>
Hostname to listen on.

Default
127.0.0.1

-P, --port <port>
Port of the webserver.

Default
5000

-OT, --output-template <output_template>
Template output.

-MC, --model-conf <model_conf>
Model-configuration file path *.yaml*.

-OS, --only-summary
Do not save vehicle outputs, just the summary.

-AS, --augmented-summary
More outputs to the summary.

-HV, --hard-validation
Add extra data validations.

-DM, --declaration-mode
Use only the declaration data.

-ES, --enable-selector
Enable the selection of the best model to predict both H/L cycles.

-TA, --type-approval-mode
Is launched for TA?

-PL, --plot-workflow
Open workflow-plot in browser, after run finished.

-KP, --encryption-keys-passwords <encryption_keys_passwords>
Encryption keys passwords file for reading TA files.

Default
`./DICE_KEYS/secret.passwords`

-v, --verbosity <LVL>
Either CRITICAL, ERROR, WARNING, INFO or DEBUG

Arguments

INPUT_FILES
Optional argument(s)

syncing

SYNCING command line tool.

```
co2mpas syncing [OPTIONS] COMMAND [ARGS]...
```

Options

--version
Show the version and exit.

sync

Synchronise and re-sample data-sets defined in INPUT_FILE and writes shifts and synchronised data into the OUTPUT_FILE.

INPUT_FILE: Data-sets input file (format: .xlsx, .json).

OUTPUT_FILE: output file (format: .xlsx, .json).

DATA_NAMES: to filter out the data sets to synchronise.

```
co2mpas syncing sync [OPTIONS] INPUT_FILE OUTPUT_FILE [DATA_NAMES]...
```

Options

-R, --reference-name <reference_name>
Reference data-set name.

Default
theoretical

-x, --x-label <x_label>
Default *var-name* of the common x-axis to synchronise and re-sampled the data-sets.

Default
times

-y, --y-label <y_label>
Default *var-name* of the common y-axis to synchronise the data-sets.

Default
velocities

-I, --interpolation-method <interpolation_method>
Interpolation method used to re-sample all data-sets.

Default
linear

Options
linear | nearest | zero | slinear | quadratic | cubic | pchip | akima | integral | polynomial0 | polynomial1 | polynomial2 | polynomial3 | polynomial4 | spline5 | spline7 | spline9

-S, --sets-mapping-fpath <sets_mapping_fpath>
File path (*.json*) of data-sets mapping definition. It is like`{"<set-name>": {"<new-name>": "<old-name>", ...}, ...}`.

-L, --labels-fpath <labels_fpath>
File path (*.json*) of *reference-labels* (i.e., "x", "y"). It is like *{"<set-name>": {"x": "<x-label>", "y": "<y-label>"}, ...}*.

-M, --methods-fpath <methods_fpath>
File path (*.json*) of interpolation methods. It is like *{"<set-name>": {"<var-name>": "<interp>", ...}, ...}*.

-NS, --no-sync
Executes only the re-sampling without data synchronisation.

-H, --header <header>
Row (0-indexed) to use for the column labels.

-v, --verbosity <LVL>
Either CRITICAL, ERROR, WARNING, INFO or DEBUG

Arguments

INPUT_FILE
Required argument

OUTPUT_FILE
Required argument

DATA_NAMES
Optional argument(s)

template

Writes a sample template OUTPUT_FILE.

OUTPUT_FILE: SYNCING input template file (.xlsx). [default: ./datasync.xlsx]

```
co2mpas syncing template [OPTIONS] [OUTPUT_FILE]
```

Options

-CT, --cycle-type <cycle_type>
Cycle type.

Default
`wltp`

Options
nedc | wltp

-WC, --wltp-class <wltp_class>
WLTP vehicle class.

Default
`class3b`

Options
class1 | class2 | class3a | class3b

-GB, --gear-box-type <gear_box_type>
Gear box type.

Default
`automatic`

Options
manual | automatic

-v, --verbosity <LVL>
Either CRITICAL, ERROR, WARNING, INFO or DEBUG

Arguments

OUTPUT_FILE
Optional argument

template

Writes a CO2MPAS input/output template into OUTPUT_FILE.

OUTPUT_FILE: File path *.xlsx*. [default: ./template.xlsx]

```
co2mpas template [OPTIONS] [OUTPUT_FILE]
```

Options

-TT, --template-type <template_type>
Template file type.

Default
`input`

Options
input | output

-v, --verbosity <LVL>
Either CRITICAL, ERROR, WARNING, INFO or DEBUG

Arguments

OUTPUT_FILE
Optional argument

10.2 utils

It contains classes and functions of general utility.

Functions

argmax	Returns the indices of the maximum values along an axis.
check_first_arg	Check first arg is true.
check_first_arg_false	Check first arg is false.
clear_fluctuations	Clears the gear identification fluctuations.
get_inliers	Returns the inliers data.
index_phases	Return the indices of the phases when is true.
mad	Median Absolute Deviation.
mae	Mean absolute error.
median_filter	Calculates the moving median-high of y values over a constant dx.
numpy_random_seed	Set temporary the numpy random state.
pairwise	A sequence of overlapping sub-sequences.
reject_outliers	Calculates the median and standard deviation of the sample rejecting the outliers.
sliding_window	Returns a sliding window (of width dx) over data from the iterable.

10.2.1 argmax

argmax(*values*, ***kws*)

Returns the indices of the maximum values along an axis.

Parameters
values (*numpy.array | list*) – Input array.

Returns
Indices of the maximum values

Return type
numpy.ndarray

10.2.2 check_first_arg

check_first_arg(*first*, **args*)

Check first arg is true.

Parameters
first (*T*) – First arg.

Returns
If first arg is true.

Return type
bool

10.2.3 check_first_arg_false

check_first_arg_false(*first*, **args*)

Check first arg is false.

Parameters
first (*T*) – First arg.

Returns
If first arg is false.

Return type
bool

10.2.4 clear_fluctuations

clear_fluctuations(*times*, *gears*, *dt_window*)

Clears the gear identification fluctuations.

Parameters

- **times** (*numpy.array*) – Time vector.
- **gears** (*numpy.array*) – Gear vector.
- **dt_window** (*float*) – Time window.

Returns
Gear vector corrected from fluctuations.

Return type
numpy.array

10.2.5 get_inliers

get_inliers(*x*, *n=1*, *med=<function median>*, *std=<function std>*)

Returns the inliers data.

Parameters

- **x** (*Iterable*) – Input data.
- **n** (*int*) – Number of standard deviations.
- **med** (*callable, optional*) – Median function.
- **std** (*callable, optional*) – Standard deviation function.

Returns
Inliers mask, median and standard deviation of inliers.

Return type
(numpy.array, float, float)

10.2.6 index_phases

index_phases(*phases*)

Return the indices of the phases when is true.

Parameters
phases (*numpy.array*) – Phases vector.

Returns
Indices of the phases when is true.

Return type
numpy.array

10.2.7 mad

mad(*x*, *med=None*)

Median Absolute Deviation.

10.2.8 mae

mae(*x*, *y*, *w=None*)

Mean absolute error.

Parameters

- **x** (*numpy.array*) – Reference values.
- **y** (*numpy.array*) – Output values.
- **w** (*numpy.array*) – Weights.

Returns
Mean absolute error.

Return type
float

10.2.9 median_filter

median_filter(*x*, *y*, *dx_window*, *filter=<function median_high>*)

Calculates the moving median-high of y values over a constant dx.

Parameters

- **x** (*Iterable*) – x data.
- **y** (*Iterable*) – y data.
- **dx_window** (*float*) – dx window.
- **filter** (*callable*) – Filter function.

Returns
Moving median-high of y values over a constant dx.

Return type
numpy.array

10.2.10 numpy_random_seed

numpy_random_seed(*seed*)

Set temporary the numpy random state.

Parameters
seed (*int*) – Seed for *RandomState*.

10.2.11 pairwise

pairwise(*iterable*)

A sequence of overlapping sub-sequences.

Parameters
iterable (*iterable*) – An iterable object.

Returns
A zip object.

Return type
zip

Example:

```
>>> list(pairwise([1, 2, 3, 4, 5]))
[(1, 2), (2, 3), (3, 4), (4, 5)]
```

10.2.12 reject_outliers

reject_outliers(*x*, *n=1*, *med=<function median>*, *std=<function std>*)

Calculates the median and standard deviation of the sample rejecting the outliers.

Parameters

- **x** (*Iterable*) – Input data.
- **n** (*int*) – Number of standard deviations.
- **med** (*callable, optional*) – Median function.
- **std** (*callable, optional*) – Standard deviation function.

Returns
Median and standard deviation.

Return type
(float, float)

10.2.13 sliding_window

sliding_window(*xy*, *dx_window*)

Returns a sliding window (of width dx) over data from the iterable.

Parameters

- **xy** (*list[(float, float) | list[float]]*) – X and Y values.
- **dx_window** (*float*) – dX window.

Returns
Data (x & y) inside the time window.

Return type
generator

Classes

Constants
List

10.2.14 Constants

class Constants

Methods

__init__	
clear	
copy	
dump	
from_dict	
fromkeys	Create a new dictionary with keys from iterable and values set to value.
get	Return the value for key if key is in the dictionary, else default.
items	
keys	
load	
pop	If the key is not found, return the default if given; otherwise, raise a KeyError.
popitem	Remove and return a (key, value) pair as a 2-tuple.
setdefault	Insert key with a value of default if key is not in the dictionary.
to_dict	
update	If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]
values	

__init__

Constants.**__init__**(**args*, ***kwargs*)

clear

Constants.**clear**() → None. Remove all items from D.

copy

Constants.**copy**() → a shallow copy of D

dump

Constants.**dump**(*file*, *default_flow_style=False*, ***kw*)

from_dict

Constants.**from_dict**(*d*)

fromkeys

Constants.**fromkeys**(*value=None*, */*)

Create a new dictionary with keys from iterable and values set to value.

get

Constants.**get**(*key*, *default=None*, */*)

Return the value for key if key is in the dictionary, else default.

items

Constants.**items**() → a set-like object providing a view on D's items

keys

Constants.**keys**() → a set-like object providing a view on D's keys

load

Constants.**load**(*file*, ***kw*)

pop

Constants.**pop**(*k*[, *d*]) → v, remove specified key and return the corresponding value.

If the key is not found, return the default if given; otherwise, raise a KeyError.

popitem

Constants.**popitem**()

Remove and return a (key, value) pair as a 2-tuple.

Pairs are returned in LIFO (last-in, first-out) order. Raises KeyError if the dict is empty.

setdefault

Constants.**setdefault**(*key*, *default=None*, */*)

Insert key with a value of default if key is not in the dictionary.

Return the value for key if key is in the dictionary, else default.

to_dict

Constants.**to_dict**()

update

Constants.**update**([*E*,]***F*) → None. Update D from dict/iterable E and F.

If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]

values

Constants.**values**() → an object providing a view on D's values

__init__(**args*, ***kwargs*)

10.2.15 List

class List(**args*, *dtype=<class 'float'>*, ***kwargs*)

Methods

__init__	
append	Append object to the end of the list.
clear	Remove all items from list.
copy	Return a shallow copy of the list.
count	Return number of occurrences of value.
extend	Extend list by appending elements from the iterable.
index	Return first index of value.
insert	Insert object before index.
pop	Remove and return item at index (default last).
remove	Remove first occurrence of value.
reverse	Reverse *IN PLACE*.
sort	Sort the list in ascending order and return None.
toarray	

__init__

List.**__init__**(**args*, ***kwargs*)

append

List.**append**(*object*, */*)

Append object to the end of the list.

clear

List.**clear**()

Remove all items from list.

copy

List.**copy**()

Return a shallow copy of the list.

count

List.**count**(*value*, */*)

Return number of occurrences of value.

extend

List.**extend**(*iterable*, /)

Extend list by appending elements from the iterable.

index

List.**index**(*value*, *start=0*, *stop=9223372036854775807*, /)

Return first index of value.

Raises ValueError if the value is not present.

insert

List.**insert**(*index*, *object*, /)

Insert object before index.

pop

List.**pop**(*index=-1*, /)

Remove and return item at index (default last).

Raises IndexError if list is empty or index is out of range.

remove

List.**remove**(*value*, /)

Remove first occurrence of value.

Raises ValueError if the value is not present.

reverse

List.**reverse**()

Reverse *IN PLACE*.

sort

List.**sort**(*, *key=None*, *reverse=False*)

Sort the list in ascending order and return None.

The sort is in-place (i.e. the list itself is modified) and stable (i.e. the order of two equal elements is maintained).

If a key function is given, apply it once to each list item and sort them, ascending or descending, according to their function values.

The reverse flag can be set to sort in descending order.

toarray

List.**toarray**(*dtype=None*, **args*, ***kwargs*)

__init__(**args*, ***kwargs*)

Attributes

dtype
empty

dtype

List.**dtype = None**

empty

List.**empty = empty**

10.3 defaults

Constants for the CO2MPAS physical model.

Classes

Defaults
Functions
Values

10.3.1 Defaults

class Defaults

Methods

__init__	
clear	
copy	
dump	
from_dict	
fromkeys	Create a new dictionary with keys from iterable and values set to value.
get	Return the value for key if key is in the dictionary, else default.
items	
keys	
load	
pop	If the key is not found, return the default if given; otherwise, raise a KeyError.
popitem	Remove and return a (key, value) pair as a 2-tuple.
setdefault	Insert key with a value of default if key is not in the dictionary.
to_dict	
update	If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]

__init__

Defaults.**__init__**(**args*, ***kwargs*)

clear

Defaults.**clear**() → None. Remove all items from D.

copy

Defaults.**copy**() → a shallow copy of D

dump

Defaults.**dump**(*file*, *default_flow_style=False*, ***kw*)

from_dict

Defaults.**from_dict**(*d*)

fromkeys

Defaults.**fromkeys**(*value=None*, */*)

Create a new dictionary with keys from iterable and values set to value.

get

Defaults.**get**(*key*, *default=None*, */*)

Return the value for key if key is in the dictionary, else default.

items

Defaults.**items**() → a set-like object providing a view on D's items

keys

Defaults.**keys**() → a set-like object providing a view on D's keys

load

Defaults.**load**(*file*, ***kw*)

pop

Defaults.**pop**(*k*[, *d*]) → v, remove specified key and return the corresponding value.

If the key is not found, return the default if given; otherwise, raise a KeyError.

popitem

Defaults.**popitem**()

Remove and return a (key, value) pair as a 2-tuple.

Pairs are returned in LIFO (last-in, first-out) order. Raises KeyError if the dict is empty.

setdefault

Defaults.**setdefault**(*key*, *default=None*, */*)

Insert key with a value of default if key is not in the dictionary.

Return the value for key if key is in the dictionary, else default.

to_dict

Defaults.**to_dict**()

update

Defaults.**update**([*E*,]***F*) → None. Update D from dict/iterable E and F.

If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]

__init__(**args*, ***kwargs*)

Attributes

EPS	Machine error.
INF	Infinite value.
functions	
values	

EPS

Defaults.**EPS = 1.1920929e-07**

Machine error.

INF

Defaults.**INF = 10000.0**

Infinite value.

functions

Defaults.**functions = {}**

values

Defaults.**values = {}**

values = {}

EPS = 1.1920929e-07

Machine error.

INF = 10000.0

Infinite value.

10.3.2 Functions

class Functions

Methods

__init__	
clear	
copy	
dump	
from_dict	
fromkeys	Create a new dictionary with keys from iterable and values set to value.
get	Return the value for key if key is in the dictionary, else default.
items	
keys	
load	
pop	If the key is not found, return the default if given; otherwise, raise a KeyError.
popitem	Remove and return a (key, value) pair as a 2-tuple.
setdefault	Insert key with a value of default if key is not in the dictionary.
to_dict	
update	If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]
values	

__init__

Functions.**__init__**(**args*, ***kwargs*)

clear

Functions.**clear**() → None. Remove all items from D.

copy

Functions.**copy**() → a shallow copy of D

dump

Functions.**dump**(*file*, *default_flow_style=False*, ***kw*)

from_dict

Functions.**from_dict**(*d*)

fromkeys

Functions.**fromkeys**(*value=None*, */*)

Create a new dictionary with keys from iterable and values set to value.

get

Functions.**get**(*key*, *default=None*, */*)

Return the value for key if key is in the dictionary, else default.

items

Functions.**items**() → a set-like object providing a view on D's items

keys

Functions.**keys**() → a set-like object providing a view on D's keys

load

Functions.**load**(*file*, ***kw*)

pop

Functions.**pop**(*k*[, *d*]) → v, remove specified key and return the corresponding value.

If the key is not found, return the default if given; otherwise, raise a KeyError.

popitem

Functions.**popitem**()

Remove and return a (key, value) pair as a 2-tuple.

Pairs are returned in LIFO (last-in, first-out) order. Raises KeyError if the dict is empty.

setdefault

Functions.**setdefault**(*key*, *default=None*, */*)

Insert key with a value of default if key is not in the dictionary.

Return the value for key if key is in the dictionary, else default.

to_dict

Functions.**to_dict**()

update

Functions.**update**([*E*,]***F*) → None. Update D from dict/iterable E and F.

If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]

values

Functions.**values**() → an object providing a view on D's values

__init__(**args*, ***kwargs*)

Attributes

ENABLE_ALL_FUNCTIONS

ENABLE_ALL_FUNCTIONS

Functions.**ENABLE_ALL_FUNCTIONS = False**

class default_after_treatment_warm_up_duration

duration = 60

After treatment warm up duration for conventional vehicles [s].

class default_after_treatment_cooling_duration

duration = inf

After treatment cooling duration for conventional vehicles [s].

class parse_solution

CALIBRATION_AS_TARGETS = False

Use all calibration outputs as relative prediction targets.

class select_best_model

MAP = {None: {'wltp_h': ('nedc_h', 'wltp_h'), 'wltp_l': ('nedc_l', 'wltp_l'), 'wltp_m': ('wltp_m',)}}

Model selector mapping.

class calculate_cylinder_deactivation_valid_phases

LIMIT = 0.0001

Engine inertia [kW].

class calculate_drive_battery_n_parallel_cells_v1

reference_volt = {'Li-FP (Li-Fe-P)': 2.4, 'Li-MO (Li-Mn)': 2.9, 'Li-NCA (Li-Ni-Co-Al)': 2.9, 'Li-NCM (Li-Ni-Mn-Co)': 2.9, 'Li-TO (Li-Ti)': 1.7, 'NiMH': 1.1, 'unknown': 2.9}

Voltage for calculating the number of parallel battery cells [V].

class default_planetary_mean_efficiency

efficiency = 0.97

Planetary mean efficiency [-].

class default_planetary_ratio

ratio = 2.6

Fundamental planetary speed ratio [-].

class identify_after_treatment_warm_up_phases

cooling_time = 400

After treatment cooling time [s].

class define_service_battery_electric_powers_supply_threshold

min_soc = 0.1

Minimum SOC variation to define service battery charging status [%].

min_current = -1.0

Maximum allowed negative current for the service battery being considered not charging [A].

class default_final_drive_efficiency

formula = '1 - (n_wheel_drive - 2) / 100'

Formula for the default final drive efficiency [function].

class predict_clutch_tc_speeds_delta

ENABLE = False

Enable prediction of clutch or torque converter speeds delta?

class default_tc_normalized_m1000_curve

```
curve = {'x':  [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.65, 0.7, 0.75, 0.8, 0.87,
0.9, 0.95, 1, 1.1, 1.222, 1.375, 1.571, 1.833, 2.2, 2.5, 3, 3.5, 4, 4.5, 5],
'y':  [1, 0.97, 0.93, 0.9, 0.87, 0.83, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5,
0.25, 0, -0.099, -0.198, -0.336, -0.535, -0.828, -1.306, -1.781, -2.772, -4.071,
-5.746, -7.861, -10.48]}
```

Normalized m1000 curve [-].

class default_n_wheel

```
n_wheel = 4
```

Total number of wheels [-].

class default_static_friction

```
coeff = {'new':  {'dry':  0.85, 'ice':  0.1, 'puddles':  0.5, 'rainfall':  0.55,
'wet':  0.65}, 'worm':  {'dry':  1, 'ice':  0.1, 'puddles':  0.25, 'rainfall':
0.4, 'wet':  0.5}}
```

Static friction coefficient [-].

class calculate_velocities

```
dt_window = 5
```

Time window for the moving average of obd velocities [s].

class calculate_engine_temperature_derivatives

```
dx = 4
```

Derivative spacing [s].

```
order = 7
```

Degree of the smoothing spline [-].

```
tw = 20
```

Time window for smoother [s].

class default_clutch_window

```
clutch_window = (0, 0.95384615)
```

Clutching time window [s].

class define_fuel_type_and_is_hybrid

```
is_hybrid = {1:  False, 2:  False, 3:  False, 4:  False, 5:  False, 6:  False,
7:  False, 8:  True, 9:  False, 10:  False, 11:  False, 12:  False, 13:  False,
14:  False, 15:  True, 16:  True, 17:  True, 18:  True, 19:  True, 20:  True,
21:  True, 22:  True, 23:  False}
```

Is the vehicle hybrid?

```
fuel_type = {1:  'gasoline', 2:  'methanol', 3:  'ethanol', 4:  'diesel', 5:
'LPG', 6:  'NG', 7:  'propane', 8:  None, 9:  'gasoline', 10:  'methanol', 11:
'ethanol', 12:  'LPG', 13:  'NG', 14:  'propane', 15:  None, 16:  'gasoline',
17:  'gasoline', 18:  'ethanol', 19:  'diesel', 20:  None, 21:  'gasoline', 22:
None, 23:  'diesel'}
```

The vehicle fuel type.

class CMV

ENABLE_OPT_LOOP = False

Enable optimization loop?

class default_start_stop_activation_time

ENABLE = False

Enable function?

threshold = 30

Start-stop activation time threshold [s].

class calculate_last_gear_box_ratio_v1

MAX_RATIO = 2

Maximum admissible ratio for the last gear [-].

MIN_RATIO = 0.2

Minimum admissible ratio for the last gear [-].

DELTA_RATIO = 0.1

Delta ratio for calculating the last gear [-].

class calculate_maximum_velocity

MAX_VEL = 1000

Maximum admissible velocity for the vehicle maximum velocity [km/h].

MIN_VEL = 1

Minimum admissible velocity for the vehicle maximum velocity [km/h].

DELTA_VEL = 1

Delta ratio for calculating the vehicle maximum velocity [km/h].

PREC_FLC = 0.9

Full load curve percentage fro calculating the available power [-].

class MGS

MAX_VEL = 1000

Maximum admissible velocity for the vehicle maximum velocity [km/h].

MIN_VEL = 1

Minimum admissible velocity for the vehicle maximum velocity [km/h].

DELTA_VEL = 0.1

Delta ratio for calculating the vehicle maximum velocity [km/h].

PREC_FLC = 0.9

Full load curve percentage fro calculating the available power [-].

class calculate_maximum_velocity_v2

MAX_VEL = 1000

Maximum admissible velocity for the vehicle maximum velocity [km/h].

MIN_VEL = 1

Minimum admissible velocity for the vehicle maximum velocity [km/h].

DELTA_VEL = 1

Delta ratio for calculating the vehicle maximum velocity [km/h].

class calculate_first_gear_box_ratio

STARTING_SLOPE = 0.5

Starting slope [-].

MAX_TORQUE_PERCENTAGE = 0.95

Percentage of maximum engine torque to calculate the first gear [-].

class design_gear_box_ratios

f_two = [1, 1.01, 1.02, 1.03, 1.04, 1.05, 1.06, 1.07, 1.08, 1.09, 1.1, 1.11, 1.12, 1.13, 1.14, 1.15, 1.16, 1.17, 1.18, 1.19, 1.2]

Two factor to design the gear box ratios [-].

f_tuning = [1, 1.01, 1.02, 1.03, 1.04, 1.05, 1.06, 1.07, 1.08, 1.09, 1.1]

Tuning factor to design the gear box ratios [-].

class calculate_aerodynamic_drag_coefficient_v1

cw = {'bestelwagen': 0.35, 'bus': 0.35, 'cabriolet': 0.28, 'coupé': 0.27, 'hatchback': 0.3, 'mpv': 0.3, 'pick-up': 0.4, 'sedan': 0.27, 'stationwagon': 0.28, 'suv/crossover': 0.35}

Aerodynamic drag coefficients function of vehicle body [-].

class calculate_aerodynamic_drag_coefficient

cw = {'A': 0.34, 'B': 0.31, 'C': 0.29, 'D': 0.3, 'E': 0.3, 'F': 0.28, 'G': 0.29, 'H': 0.29, 'I': 0.32, 'J': 0.38, 'K': 0.38, 'L': 0.32, 'M': 0.32, 'N': 0.38, 'S': 0.29}

Aerodynamic drag coefficients function of vehicle category [-].

class calculate_f2

roof_box = 1.2

Deteriorating coefficient of the aerodynamic drag and frontal area due to the roof box [-].

class calculate_rolling_resistance_coeff

coeff = {'C1': {'A': 0.005900000000000001, 'B': 0.0070999999999999995, 'C': 0.008400000000000001, 'D': 0.009800000000000001, 'E': 0.009800000000000001, 'F': 0.011300000000000001, 'G': 0.0129}, 'C2': {'A': 0.004900000000000001, 'B': 0.0060999999999999995, 'C': 0.0074, 'D': 0.0086, 'E': 0.0086, 'F': 0.0099, 'G': 0.0112}, 'C3': {'A': 0.0035, 'B': 0.0045, 'C': 0.0055, 'D': 0.0065, 'E': 0.0075, 'F': 0.0085, 'G': 0.0085}}

Rolling resistance coeff, function of tyre class and category [-]. http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX: 02009R1222-20120530&from=EN (table A4/1 of eu legislation not world wide)

class calculate_f1

qm = (1.38045, -35.8675)

Linear model coefficients.

class calculate_raw_frontal_area_v1

```
    formulas = {'A': '0.4041 * np.log(vehicle_mass) - 0.338', 'B': '0.4041 *
    np.log(vehicle_mass) - 0.338', 'C': '0.4041 * np.log(vehicle_mass) - 0.338',
    'D': '0.4041 * np.log(vehicle_mass) - 0.338', 'E': '0.4041 *
    np.log(vehicle_mass) - 0.338', 'F': '0.4041 * np.log(vehicle_mass) - 0.338',
    'J': '0.0007 * vehicle_mass + 1.8721', 'M': '0.0007 * vehicle_mass + 1.8721',
    'S': '0.4041 * np.log(vehicle_mass) - 0.338'}
```

Frontal area formulas function of vehicle_mass [function].

```
class calculate_frontal_area
```

```
    projection_factor = 0.84
```

Projection factor from the row frontal area (h * w) [-].

```
class select_prediction_data
```

```
    prediction_data = [['calibration', ['angle_slope', 'alternator_nominal_voltage',
    'alternator_efficiency', 'battery_capacity', 'cycle_type', 'cycle_name',
    'engine_capacity', 'engine_stroke', 'final_drive_efficiency',
    'final_drive_ratios', 'frontal_area', 'final_drive_ratio',
    'engine_thermostat_temperature', 'aerodynamic_drag_coefficient', 'fuel_type',
    'ignition_type', 'gear_box_type', 'engine_max_power',
    'engine_speed_at_max_power', 'rolling_resistance_coeff',
    'time_cold_hot_transition', 'engine_idle_fuel_consumption', 'engine_type',
    'has_start_stop', 'engine_is_turbo', 'engine_fuel_lower_heating_value', 'f0',
    'has_energy_recuperation', 'fuel_carbon_content_percentage', 'f1', 'f2',
    'full_load_speeds', 'plateau_acceleration', 'full_load_powers',
    'fuel_saving_at_strategy', 'stand_still_torque_ratio', 'lockup_speed_ratio',
    'change_gear_window_width', 'alternator_start_window_width', 'stop_velocity',
    'min_time_engine_on_after_start', 'min_engine_on_speed',
    'max_velocity_full_load_correction', 'is_hybrid', 'tyre_code',
    'engine_has_cylinder_deactivation', 'active_cylinder_ratios',
    'engine_has_variable_valve_actuation', 'has_torque_converter',
    'has_gear_box_thermal_management', 'has_lean_burn', 'ki_multiplicative',
    'n_wheel_drive', 'ki_additive', 'has_periodically_regenerating_systems',
    'n_dyno_axes', 'has_selective_catalytic_reduction',
    'start_stop_activation_time', 'has_exhausted_gas_recirculation',
    'engine_n_cylinders', 'initial_drive_battery_state_of_charge',
    'motor_p0_speed_ratio', 'motor_p1_speed_ratio', 'motor_p2_speed_ratio',
    'motor_p2_planetary_speed_ratio', 'motor_p3_front_speed_ratio',
    'motor_p3_rear_speed_ratio', 'motor_p4_front_speed_ratio',
    'motor_p4_rear_speed_ratio', 'rcb_correction', 'vehicle_mass',
    'speed_distance_correction', 'atct_family_correction_factor', 'is_plugin']],
    ['models', 'all'], ['user', 'all']]
```

WLTP calibration data used to predict.

```
class FMEP_egr
```

```
    egr_fact_map = {('compression', False):  3, ('compression', True):  4,
    ('positive natural aspiration', False):  2, ('positive turbo', False):  1}
```

Exhausted gas recirculation multiplication factors ids [-].

```
class identify_co2_emissions
```

```
    n_perturbations = 100
```

Number of perturbations to identify the co2_emissions [-].

enable_first_step = True

Enable first step in the co2_params calibration? [-]

enable_second_step = True

Enable second step in the co2_params calibration? [-]

enable_third_step = False

Enable third step co2_params calibration in perturbation loop? [-]

xatol = 0.0001

Absolute error in k_refactor between iterations that is acceptable for convergence in perturbation loop [-].

class calibrate_co2_params

enable_first_step = False

Enable first step in the co2_params calibration? [-]

enable_second_step = False

Enable second step in the co2_params calibration? [-]

enable_third_step = True

Enable third step in the co2_params calibration? [-]

class BatteryStatusModel

min_delta_time_boundaries = 5

Minimum delta time to consider valid a charging state to fit charges boundaries [s].

min_percentile_bers = 90

Minimum acceptance percentile to fit the bers threshold [%].

min_delta_soc = 8

Minimum delta soc to set the charging boundaries [%].

class default_ki_multiplicative

ki_multiplicative = {False: 1.0, True: 1.05}

Multiplicative correction for vehicles with periodically regenerating systems [-].

class define_fmep_model

acr_full_bmep_curve_percentage = 0.72

Percentage of max full bmep curve used as limit in cylinder deactivation strategy [-].

acr_max_mean_piston_speeds_percentage = 0.45

Percentage of max mean piston speeds used as upper limit in cylinder deactivation strategy [-].

acr_min_mean_piston_speeds_percentage = 1.5

Percentage of min mean piston speeds used as lower limit in cylinder deactivation strategy [-].

lb_full_bmep_curve_percentage = 0.4

Percentage of max full bmep curve used as limit in lean burn strategy [-].

lb_max_mean_piston_speeds_percentage = 0.6

Percentage of max mean piston speeds used as limit in lean burn strategy [-].

egr_full_bmep_curve_percentage = 0.5

Percentage of max full bmep curve used as limit in exhausted gas recirculation strategy [-].

egr_max_mean_piston_speeds_percentage = 0.5

Percentage of max mean piston speeds used as limit in exhausted gas recirculation strategy [-].

class define_idle_model_detector

EPS = 100.0

eps parameter of DBSCAN [RPM].

class identify_idle_engine_speed_std

MIN_STD = 100.0

Min standard deviation value [RPM].

MAX_STD_PERC = 0.3

Max standard deviation percentage of median value [-].

class StartStopModel

stop_velocity = 2.0

Maximum allowed velocity to stop the engine [km/h].

plateau_acceleration = 0.2777777777777778

Minimum acceleration to switch on the engine [m/s2].

class correct_constant_velocity

CON_VEL_UP_SHIFT = (15.0, 32.0, 50.0, 70.0)

Constant velocities to correct the upper limits for NEDC [km/h].

CON_VEL_DN_SHIFT = (35.0, 50.0)

Constant velocities to correct the bottom limits for NEDC[km/h].

class define_initial_co2_emission_model_params_guess

CO2_PARAMS = {'compression': [('a', {'min': 0.0, 'value': 0.391197}), ('b', {'value': 0.028604}), ('c', {'value': -0.00196}), ('a2', {'max': 0.0, 'value': -0.0012}), ('b2', {'max': 1, 'min': -1, 'value': 0, 'vary': False}), ('l', {'max': 0.0, 'value': -1.55291}), ('l2', {'max': 0.0, 'value': -0.0076}), ('t1', {'max': 8.0, 'min': 0.0, 'value': 3.5}), ('dt', {'min': 0.0, 'value': 1.0}), ('t0', {'expr': 't1 + dt', 'max': 8.0, 'value': 4.5})], 'positive natural aspiration': [('a', {'min': 0.0, 'value': 0.4851}), ('b', {'value': 0.01193}), ('c', {'value': -0.00065}), ('a2', {'max': 0.0, 'value': -0.00385}), ('b2', {'max': 1, 'min': -1, 'value': 0, 'vary': False}), ('l', {'max': 0.0, 'value': -2.39882}), ('l2', {'max': 0.0, 'value': -0.00286}), ('t1', {'max': 8.0, 'min': 0.0, 'value': 3.5}), ('dt', {'min': 0.0, 'value': 1.0}), ('t0', {'expr': 't1 + dt', 'max': 8.0, 'value': 4.5})], 'positive turbo': [('a', {'min': 0.0, 'value': 0.468678}), ('b', {'value': 0.011859}), ('c', {'value': -0.00069}), ('a2', {'max': 0.0, 'value': -0.00266}), ('b2', {'max': 1, 'min': -1, 'value': 0, 'vary': False}), ('l', {'max': 0.0, 'value': -2.14063}), ('l2', {'max': 0.0, 'value': -0.0025}), ('t1', {'max': 8.0, 'min': 0.0, 'value': 3.5}), ('dt', {'min': 0.0, 'value': 1.0}), ('t0', {'expr': 't1 + dt', 'max': 8.0, 'value': 4.5})]}

Initial guess CO2 emission model params.

class default_specific_gear_shifting

SPECIFIC_GEAR_SHIFTING = 'ALL'

Specific gear shifting model.

class default_clutch_k_factor_curve

STAND_STILL_TORQUE_RATIO = 1.0

Torque ratio when speed ratio==0 for clutch model.

LOCKUP_SPEED_RATIO = 0.0

Minimum speed ratio where torque ratio==1 for clutch model.

class default_tc_k_factor_curve

STAND_STILL_TORQUE_RATIO = 1.9

Torque ratio when speed ratio==0 for torque converter model.

LOCKUP_SPEED_RATIO = 0.87

Minimum speed ratio where torque ratio==1 for torque converter model.

class select_default_n_dyno_axes

DYNO_AXES = {'NEDC': {2: 1, 4: 1}, 'WLTP': {2: 1, 4: 2}}

Number of dyno axes [-].

class select_phases_integration_times

INTEGRATION_TIMES = {'NEDC': (0.0, 780.0, 1180.0), 'WLTP': (0.0, 590.0, 1023.0, 1478.0, 1800.0)}

Cycle phases integration times [s].

class get_gear_box_efficiency_constants

PARAMS = {False: {'gbp00': {'m': 0.0043233434399999994, 'q': {'cold': 0.695884329, 'hot': 0.29823614099999995}}, 'gbp01': {'q': {'cold': 0.9793688500000001, 'hot': 0.98951775000000001}}, 'gbp10': {'m': 2.4525999999999996e-06, 'q': {'cold': 0.00012958919999999998, 'hot': 5.15957e-05}}}, True: {'gbp00': {'m': 0.0043233434399999994, 'q': {'cold': 0.695884329, 'hot': 0.29823614099999995}}, 'gbp01': {'q': {'cold': 0.96921995, 'hot': 0.97936885000000001}}, 'gbp10': {'m': 2.4525999999999996e-06, 'q': {'cold': 0.0005171569, 'hot': 0.0001547871}}}}

Vehicle gear box efficiency constants (gbp00, gbp10, and gbp01).

class calculate_engine_mass

PARAMS = {'mass_coeff': {'compression': 1.1, 'positive': 1.0}, 'mass_reg_coeff': (0.4208, 60)}

Equivalent gear box heat capacity parameters.

class calculate_engine_heat_capacity

PARAMS = {'heat_capacity': {'body': 460.0, 'coolant': 3558.1, 'crankcase': 460.0, 'crankshaft': 490.0, 'cyl_head': 910.0, 'oil': 2090.0, 'pistons': 910.0}, 'heated_mass_percentage': {'body': 0.1, 'coolant': 0.04, 'crankcase': 0.18, 'crankshaft': 0.08, 'cyl_head': 0.09, 'oil': 0.055, 'pistons': 0.025}}

Equivalent gear box heat capacity parameters.

class calculate_equivalent_gear_box_heat_capacity

```
PARAMS = {'gear_box_mass_engine_ratio': 0.25, 'heat_capacity': {'oil':
2090.0}, 'thermal_management_factor': 0.5}
```

Equivalent gear box heat capacity parameters.

class default_full_load_speeds_and_powers

```
FULL_LOAD = {'compression': ([0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,
1.0, 1.1, 1.2], [0.1, 0.278071182, 0.427366185, 0.572340499, 0.683251935,
0.772776746, 0.846217049, 0.906754984, 0.94977083, 0.981937981, 1, 0.937598144,
0.85]), 'positive': ([0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0,
1.1, 1.2], [0.1, 0.198238659, 0.30313392, 0.410104642, 0.516920841, 0.621300767,
0.723313491, 0.820780368, 0.901750158, 0.962968496, 0.995867804, 0.953356174,
0.85])}
```

Vehicle normalized full load curve.

class calculate_engine_max_torque

```
PARAMS = {'compression': 1.1, 'positive': 1.25}
```

Engine nominal torque params.

class calculate_engine_moment_inertia

```
PARAMS = {'compression': 2, 'positive': 1}
```

Engine moment of inertia params.

class calculate_co2_emissions

class default_initial_drive_battery_state_of_charge

class default_initial_service_battery_state_of_charge

class default_fuel_density

```
FUEL_DENSITY = {'LPG': 699.6521739130435, 'NG': 713.6141906873614, 'biodiesel':
890.0, 'diesel': 832.0, 'ethanol': 794.0, 'gasoline': 745.0, 'methanol':
791.0, 'propane': 510.0}
```

Fuel density [g/l].

class default_fuel_lower_heating_value

```
LHV = {'LPG': 46000.0, 'NG': 45100.0, 'biodiesel': 37900.0, 'diesel': 43100.0,
'ethanol': 26800.0, 'gasoline': 43200.0, 'methanol': 19800.0, 'propane':
49680.0}
```

Fuel lower heating value [kJ/kg].

class default_fuel_carbon_content

```
CARBON_CONTENT = {'LPG': 1.35, 'NG': 3.21, 'biodiesel': 2.81, 'diesel': 3.16,
'ethanol': 1.91, 'gasoline': 3.17, 'methanol': 1.37, 'propane': 2.99}
```

Fuel carbon content [CO2g/g].

10.3.3 Values

class Values

Methods

__init__	
clear	
copy	
dump	
from_dict	
fromkeys	Create a new dictionary with keys from iterable and values set to value.
get	Return the value for key if key is in the dictionary, else default.
items	
keys	
load	
pop	If the key is not found, return the default if given; otherwise, raise a KeyError.
popitem	Remove and return a (key, value) pair as a 2-tuple.
setdefault	Insert key with a value of default if key is not in the dictionary.
to_dict	
update	If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]
values	

__init__

Values.**__init__**(**args*, ***kwargs*)

clear

Values.**clear**() → None. Remove all items from D.

copy

Values.**copy**() → a shallow copy of D

dump

Values.**dump**(*file*, *default_flow_style=False*, ***kw*)

from_dict

Values.**from_dict**(*d*)

fromkeys

Values.**fromkeys**(*value=None*, */*)

Create a new dictionary with keys from iterable and values set to value.

get

Values.**get**(*key*, *default=None*, */*)

Return the value for key if key is in the dictionary, else default.

items

Values.**items**() → a set-like object providing a view on D's items

keys

Values.**keys**() → a set-like object providing a view on D's keys

load

Values.**load**(*file*, ***kw*)

pop

Values.**pop**(*k*[, *d*]) → v, remove specified key and return the corresponding value.

If the key is not found, return the default if given; otherwise, raise a KeyError.

popitem

Values.**popitem**()

Remove and return a (key, value) pair as a 2-tuple.

Pairs are returned in LIFO (last-in, first-out) order. Raises KeyError if the dict is empty.

setdefault

Values.**setdefault**(*key*, *default=None*, */*)

Insert key with a value of default if key is not in the dictionary.

Return the value for key if key is in the dictionary, else default.

to_dict

Values.**to_dict**()

update

Values.**update**([*E*,]***F*) → None. Update D from dict/iterable E and F.

If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]

values

Values.**values**() → an object providing a view on D's values

__init__(**args*, ***kwargs*)

Attributes

active_cylinder_ratios	Possible percentages of active cylinders [-].
air_temperature	Air temperature [°C].
alternator_efficiency	Alternator efficiency [-].
angle_slope	Angle slope [rad].
atct_family_correction_factor	Family correction factor for representative regional temperatures [-].
atmospheric_pressure	Atmospheric pressure [kPa].
auxiliaries_power_loss	Constant power loss due to engine auxiliaries [kW].
auxiliaries_torque_loss_factors	Constant torque loss factors due to engine auxiliaries [N/cc, N*m].

continues on next page

Table 1 – continued from previous page

belt_efficiency	Belt efficiency [-].
cargo_mass	Cargo mass [kg].
change_gear_window_width	Time window used to apply gear change filters [s].
co2_params	CO2 emission model params.
correct_f0	A different preconditioning cycle was used for WLTP and NEDC?
delta_time_engine_starter	Time elapsed to turn on the engine with electric starter [s].
drive_battery_technology	Drive battery technology type.
enable_manual_wltp_gearshift	Enable the manual gearshift from WLTP regulation?
enable_phases_willans	Enable the calculation of Willans coefficients for all phases?
enable_willans	Enable the calculation of Willans coefficients for the cycle?
engine_has_cylinder_deactivation	Does the engine have cylinder deactivation technology?
engine_has_variable_valve_actuation	Does the engine feature variable valve actuation? [-].
engine_is_turbo	If the engine is equipped with any kind of charging.
engine_n_cylinders	Number of engine cylinders [-].
final_drive_ratio	Final drive ratio [-].
fuel_mass	Fuel mass [kg].
fuel_saving_at_strategy	Apply the eco-mode gear shifting?
gear_box_temperature_references	Cold and hot gear box reference temperatures [°C].
has_energy_recuperation	Does the vehicle have energy recuperation features?
has_gear_box_thermal_management	Does the gear box have some additional technology to heat up faster?
has_lean_burn	Does the engine have lean burn technology?
has_periodically_regenerating_systems	Does the vehicle has periodically regenerating systems? [-].
has_roof_box	Has the vehicle a roof box? [-].
has_selective_catalytic_reduction	Does the engine have selective catalytic reduction technology?
has_start_stop	Does the vehicle have start/stop system?
idle_engine_speed_std	Standard deviation of idle engine speed [RPM].
initial_temperature_NEDC	Initial temperature of the test cell of NEDC [°C].
initial_temperature_WLTP	Initial temperature of the test cell of WLTP [°C].
is_cycle_hot	Is an hot cycle?
is_plugin	Is the vehicle plugin hybrid?
is_serial	Is the vehicle serial hybrid?
k1	K1 NEDC parameter (first or second gear) [-].
k2	K2 NEDC parameter (first or second gear) [-].
k5	K5 NEDC parameter (first or second gear) [-].
ki_additive	Default additive correction for vehicles with periodically regenerating systems [CO2g/km].
max_time_NEDC	NEDC cycle time [s].
max_time_WLTP	WLTP cycle time [s].
max_velocity_full_load_correction	Threshold vehicle velocity for gear correction due to full load curve [km/h].
min_engine_on_speed	Minimum vehicle engine speed [RPM].
min_time_engine_on_after_start	Minimum time of engine on after a start [s].
minimum_elevation_distance	Minimum distance between points for the elevation interpolation [m].

continues on next page

Table 1 – continued from previous page

n_passengers	Number of passengers including driver [-].
passenger_mass	Average passenger mass [kg].
plateau_acceleration	Maximum acceleration to be at constant velocity [m/s2].
rcb_correction	Apply RCB correction?
road_state	Road state (i.e., dry, wet, rainfall, puddles, ice).
service_battery_start_window_width	Service battery start window width [s].
start_stop_activation_time	Start-stop activation time threshold [s].
starter_efficiency	Starter efficiency [-].
stop_velocity	Maximum velocity to consider the vehicle stopped [km/h].
time_cold_hot_transition	A/T Time at cold hot transition phase [s].
time_sample_frequency	Time frequency [1/s].
tyre_class	Tyre class (C1, C2, and C3).
tyre_dynamic_rolling_coefficient	Empirical value in case of CVT [-].
tyre_state	Tyre state (i.e., new or worm).
use_dt_gear_shifting	If to use decision tree classifiers to predict gears.
wheel_drive	Wheel drive (i.e., front, rear, front+rear).
wltp_base_model	WLTP base model params.

active_cylinder_ratios

`Values.`**`active_cylinder_ratios = (1.0,)`**

Possible percentages of active cylinders [-].

air_temperature

`Values.`**`air_temperature = 20`**

Air temperature [°C].

alternator_efficiency

`Values.`**`alternator_efficiency = 0.67`**

Alternator efficiency [-].

angle_slope

`Values.`**`angle_slope = 0.0`**

Angle slope [rad].

atct_family_correction_factor

Values.**atct_family_correction_factor = 1.0**

Family correction factor for representative regional temperatures [-].

atmospheric_pressure

Values.**atmospheric_pressure = 101.325**

Atmospheric pressure [kPa].

auxiliaries_power_loss

Values.**auxiliaries_power_loss = 0.0213**

Constant power loss due to engine auxiliaries [kW].

auxiliaries_torque_loss_factors

Values.**auxiliaries_torque_loss_factors = (0.175, 0.2021)**

Constant torque loss factors due to engine auxiliaries [N/cc, N*m].

belt_efficiency

Values.**belt_efficiency = 0.8**

Belt efficiency [-].

cargo_mass

Values.**cargo_mass = 0**

Cargo mass [kg].

change_gear_window_width

Values.**change_gear_window_width = 4.0**

Time window used to apply gear change filters [s].

co2_params

Values.**co2_params = {}**

CO2 emission model params.

correct_f0

Values.**correct_f0 = False**
A different preconditioning cycle was used for WLTP and NEDC?

delta_time_engine_starter

Values.**delta_time_engine_starter = 0.5**
Time elapsed to turn on the engine with electric starter [s].

drive_battery_technology

Values.**drive_battery_technology = 'unknown'**
Drive battery technology type.

enable_manual_wltp_gearshift

Values.**enable_manual_wltp_gearshift = True**
Enable the manual gearshift from WLTP regulation?

enable_phases_willans

Values.**enable_phases_willans = False**
Enable the calculation of Willans coefficients for all phases?

enable_willans

Values.**enable_willans = False**
Enable the calculation of Willans coefficients for the cycle?

engine_has_cylinder_deactivation

Values.**engine_has_cylinder_deactivation = False**
Does the engine have cylinder deactivation technology?

engine_has_variable_valve_actuation

Values.**engine_has_variable_valve_actuation = False**
Does the engine feature variable valve actuation? [-].

engine_is_turbo

Values.**engine_is_turbo = True**
If the engine is equipped with any kind of charging.

engine_n_cylinders

Values.**engine_n_cylinders = 4**
Number of engine cylinders [-].

final_drive_ratio

Values.**final_drive_ratio = 1.0**
Final drive ratio [-].

fuel_mass

Values.**fuel_mass = 25**
Fuel mass [kg].

fuel_saving_at_strategy

Values.**fuel_saving_at_strategy = True**
Apply the eco-mode gear shifting?

gear_box_temperature_references

Values.**gear_box_temperature_references = (40.0, 80.0)**
Cold and hot gear box reference temperatures [°C].

has_energy_recuperation

Values.**has_energy_recuperation = True**
Does the vehicle have energy recuperation features?

has_gear_box_thermal_management

Values.**has_gear_box_thermal_management = False**
Does the gear box have some additional technology to heat up faster?

has_lean_burn

Values.**has_lean_burn = False**

Does the engine have lean burn technology?

has_periodically_regenerating_systems

Values.**has_periodically_regenerating_systems = False**

Does the vehicle has periodically regenerating systems? [-].

has_roof_box

Values.**has_roof_box = False**

Has the vehicle a roof box? [-].

has_selective_catalytic_reduction

Values.**has_selective_catalytic_reduction = False**

Does the engine have selective catalytic reduction technology?

has_start_stop

Values.**has_start_stop = True**

Does the vehicle have start/stop system?

idle_engine_speed_std

Values.**idle_engine_speed_std = 100.0**

Standard deviation of idle engine speed [RPM].

initial_temperature_NEDC

Values.**initial_temperature_NEDC = 25.0**

Initial temperature of the test cell of NEDC [°C].

initial_temperature_WLTP

Values.**initial_temperature_WLTP = 23.0**

Initial temperature of the test cell of WLTP [°C].

is_cycle_hot

Values.**is_cycle_hot = False**

Is an hot cycle?

is_plugin

Values.**is_plugin = False**

Is the vehicle plugin hybrid?

is_serial

Values.**is_serial = False**

Is the vehicle serial hybrid?

k1

Values.**k1 = 1**

K1 NEDC parameter (first or second gear) [-].

k2

Values.**k2 = 2**

K2 NEDC parameter (first or second gear) [-].

k5

Values.**k5 = 2**

K5 NEDC parameter (first or second gear) [-].

ki_additive

Values.**ki_additive = 0**

Default additive correction for vehicles with periodically regenerating systems [CO2g/km].

max_time_NEDC

Values.**max_time_NEDC = 1180.0**

NEDC cycle time [s].

max_time_WLTP

Values.**max_time_WLTP = 1800.0**

WLTP cycle time [s].

max_velocity_full_load_correction

Values.**max_velocity_full_load_correction = 100.0**

Threshold vehicle velocity for gear correction due to full load curve [km/h].

min_engine_on_speed

Values.**min_engine_on_speed = 10.0**

Minimum vehicle engine speed [RPM].

min_time_engine_on_after_start

Values.**min_time_engine_on_after_start = 4.0**

Minimum time of engine on after a start [s].

minimum_elevation_distance

Values.**minimum_elevation_distance = 30**

Minimum distance between points for the elevation interpolation [m].

n_passengers

Values.**n_passengers = 1**

Number of passengers including driver [-].

passenger_mass

Values.**passenger_mass = 75**

Average passenger mass [kg].

plateau_acceleration

Values.**plateau_acceleration = 0.10000011920929**

Maximum acceleration to be at constant velocity [m/s2].

rcb_correction

Values.**rcb_correction = True**

Apply RCB correction?

road_state

Values.**road_state = 'dry'**

Road state (i.e., dry, wet, rainfall, puddles, ice).

service_battery_start_window_width

Values.**service_battery_start_window_width = 4.0**

Service battery start window width [s].

start_stop_activation_time

Values.**start_stop_activation_time = 30**

Start-stop activation time threshold [s].

starter_efficiency

Values.**starter_efficiency = 0.7**

Starter efficiency [-].

stop_velocity

Values.**stop_velocity = 1.00000011920929**

Maximum velocity to consider the vehicle stopped [km/h].

time_cold_hot_transition

Values.**time_cold_hot_transition = 300.0**

A/T Time at cold hot transition phase [s].

time_sample_frequency

Values.**time_sample_frequency = 1.0**

Time frequency [1/s].

tyre_class

Values.**tyre_class = 'C1'**
Tyre class (C1, C2, and C3).

tyre_dynamic_rolling_coefficient

Values.**tyre_dynamic_rolling_coefficient = 0.9713375796178343**
Empirical value in case of CVT [-].

tyre_state

Values.**tyre_state = 'new'**
Tyre state (i.e., new or worm).

use_dt_gear_shifting

Values.**use_dt_gear_shifting = False**
If to use decision tree classifiers to predict gears.

wheel_drive

Values.**wheel_drive = 'front'**
Wheel drive (i.e., front, rear, front+rear).

wltp_base_model

Values.**wltp_base_model = {}**
WLTP base model params.

minimum_elevation_distance = 30
Minimum distance between points for the elevation interpolation [m].

is_plugin = False
Is the vehicle plugin hybrid?

is_serial = False
Is the vehicle serial hybrid?

rcb_correction = True
Apply RCB correction?

atct_family_correction_factor = 1.0
Family correction factor for representative regional temperatures [-].

drive_battery_technology = 'unknown'
Drive battery technology type.

belt_efficiency = 0.8
Belt efficiency [-].

starter_efficiency = 0.7

Starter efficiency [-].

tyre_state = 'new'

Tyre state (i.e., new or worm).

road_state = 'dry'

Road state (i.e., dry, wet, rainfall, puddles, ice).

engine_n_cylinders = 4

Number of engine cylinders [-].

ki_additive = 0

Default additive correction for vehicles with periodically regenerating systems [CO2g/km].

n_passengers = 1

Number of passengers including driver [-].

passenger_mass = 75

Average passenger mass [kg].

cargo_mass = 0

Cargo mass [kg].

fuel_mass = 25

Fuel mass [kg].

has_selective_catalytic_reduction = False

Does the engine have selective catalytic reduction technology?

has_lean_burn = False

Does the engine have lean burn technology?

has_gear_box_thermal_management = False

Does the gear box have some additional technology to heat up faster?

has_periodically_regenerating_systems = False

Does the vehicle has periodically regenerating systems? [-].

active_cylinder_ratios = (1.0,)

Possible percentages of active cylinders [-].

engine_has_variable_valve_actuation = False

Does the engine feature variable valve actuation? [-].

max_time_NEDC = 1180.0

NEDC cycle time [s].

max_time_WLTP = 1800.0

WLTP cycle time [s].

stop_velocity = 1.00000011920929

Maximum velocity to consider the vehicle stopped [km/h].

plateau_acceleration = 0.10000011920929

Maximum acceleration to be at constant velocity [m/s2].

has_start_stop = True

Does the vehicle have start/stop system?

engine_has_cylinder_deactivation = False

Does the engine have cylinder deactivation technology?

min_engine_on_speed = 10.0

Minimum vehicle engine speed [RPM].

min_time_engine_on_after_start = 4.0

Minimum time of engine on after a start [s].

change_gear_window_width = 4.0

Time window used to apply gear change filters [s].

service_battery_start_window_width = 4.0

Service battery start window width [s].

max_velocity_full_load_correction = 100.0

Threshold vehicle velocity for gear correction due to full load curve [km/h].

air_temperature = 20

Air temperature [°C].

atmospheric_pressure = 101.325

Atmospheric pressure [kPa].

has_roof_box = False

Has the vehicle a roof box? [-].

tyre_class = 'C1'

Tyre class (C1, C2, and C3).

angle_slope = 0.0

Angle slope [rad].

correct_f0 = False

A different preconditioning cycle was used for WLTP and NEDC?

final_drive_ratio = 1.0

Final drive ratio [-].

wheel_drive = 'front'

Wheel drive (i.e., front, rear, front+rear).

fuel_saving_at_strategy = True

Apply the eco-mode gear shifting?

gear_box_temperature_references = (40.0, 80.0)

Cold and hot gear box reference temperatures [°C].

auxiliaries_torque_loss_factors = (0.175, 0.2021)

Constant torque loss factors due to engine auxiliaries [N/cc, N*m].

auxiliaries_power_loss = 0.0213

Constant power loss due to engine auxiliaries [kW].

engine_is_turbo = True

If the engine is equipped with any kind of charging.

start_stop_activation_time = 30

Start-stop activation time threshold [s].

idle_engine_speed_std = 100.0

Standard deviation of idle engine speed [RPM].

is_cycle_hot = False

Is an hot cycle?

co2_params = {}

CO2 emission model params.

enable_phases_willans = False

Enable the calculation of Willans coefficients for all phases?

enable_willans = False

Enable the calculation of Willans coefficients for the cycle?

enable_manual_wltp_gearshift = True

Enable the manual gearshift from WLTP regulation?

alternator_efficiency = 0.67

Alternator efficiency [-].

delta_time_engine_starter = 0.5

Time elapsed to turn on the engine with electric starter [s].

use_dt_gear_shifting = False

If to use decision tree classifiers to predict gears.

has_energy_recuperation = True

Does the vehicle have energy recuperation features?

time_cold_hot_transition = 300.0

A/T Time at cold hot transition phase [s].

time_sample_frequency = 1.0

Time frequency [1/s].

initial_temperature_NEDC = 25.0

Initial temperature of the test cell of NEDC [°C].

initial_temperature_WLTP = 23.0

Initial temperature of the test cell of WLTP [°C].

k1 = 1

K1 NEDC parameter (first or second gear) [-].

k2 = 2

K2 NEDC parameter (first or second gear) [-].

k5 = 2

K5 NEDC parameter (first or second gear) [-].

`wltp_base_model = {}`

WLTP base model params.

`tyre_dynamic_rolling_coefficient = 0.9713375796178343`

Empirical value in case of CVT [-].

CHAPTER

ELEVEN

CHANGELOG

11.1 v4.3.5 (2023-11-13)

11.1.1 Feat

- (defaults): Add missing *aerodynamic_drag_coefficients*.
- (wheels): Add `Z`+ carcass.
- (cycle): Integrate *gearshift* library.
- (core): Update schedula version.

11.1.2 Fix

- (excel): Correct excel chart definition.
- (core): Correct type errors.
- (core): Correct warning and bugs related to *sklearn* or *pandas*.
- (doc): Fixed authors and FMB.

11.2 v4.3.4 (2022-11-09)

11.2.1 Fix

- (write): Correct dataframe concatenation.
- (gear_box): Correct *CMV* model conversion.
- (alternator): Correct *default_model* of *AlternatorCurrentModel*.
- (core): Correct missing requirements and defaults.

11.3 v4.3.3 (2022-11-07)

11.3.1 Feat

- (core): Add new option *none* for *gear_box_type*.
- (vehicle): Add *aerodynamic_drag_area* data node and calculation.

11.3.2 Fix

- (cycle): Add missing default.

11.4 v4.3.2 (2022-02-07)

11.4.1 Fix

- (schema): Correct f0 schema according to JET.

11.5 v4.3.1 (2022-02-03)

11.5.1 Fix

- (schema): Correct maximum velocity range schema.

11.6 v4.3.0 (2022-01-13)

11.6.1 Feat

- (bin): Add bin folder to publish the repo.
- (doc): Update copyright.
- (doc): Update documentation.
- (input): Update input template.
- (core): Improve speed performances.
- (load): Calculate default *is_plugin* from *input_type*.
- (core): Improve speed performances.
- (physical): Enable, simplify, and refactor some function.
- (physical): Add *fuel_consumptions_liters* and *fuel_consumptions_liters_value* calculations.
- (physical): Add useful outputs.
- (wheels): Add *euro* tyre code.
- (physical): Remove minor warnings.
- (alternator): Add calibration for *alternator_charging_currents*.

- (schema): Make ravel optional.
- (dice): Add flag as inputs.

11.6.2 Fix

- (core): Correct some deprecation warning.
- (core): Correct some deprecation warning.
- (load): Correct error message formatting error.
- (doc): deleted all reference to TAA.
- (faq): Glossary corrected.

11.7 v4.2.0 (2021-10-01)

11.7.1 Feat

- (template): Update input template according to JET.
- (core): Update copyright.
- (core): Update copyright.
- (excel): Enable input dictionary merging.
- (schema): Add *has_capped_velocity* and *maximum_velocity_range* parameters.
- (models): Speedup model creation.
- (core): Update model according to cars-data database.
- (excel): Drop precondition stage.
- (excel): Add *WLTP-M* cycle.
- (schema): Add *maximum_velocity*, *capped_velocity*, and *vehicle_mass_running_order* params.
- (excel): Add parser for *matrix* sheets.
- (core): Modify input file according to new DICE4.
- (engine): Implement temperature time-shift.
- (thermal): Add *after_treatment_warm_up_phases* to calibrate the thermal model.
- (fc): Replace tau function with *after_treatment_warm_up_phases*.
- (load): Use *xlref* instead of *pandalone*.
- (load): Improve schema parser performances.
- (hybrid): Add feature to plot the *FuelMapModel* and replace *matplotlib* with *plotly*.
- (vehicle): Filter out close elevation points.
- (vehicle): Add formula to identify vehicle road loads and vehicle mass.
- (clutch_tc): Add formulas to calculate *wheel_powers* from *gear_box_powers_out*.
- (clutch_tc): Add formulas to calculate *gear_box_powers_in* from *clutch_tc_powers*.
- (engine): Add formulas to calculate *clutch_tc_powers* from *engine_powers_out*.

- (docker): Add script to extract the exe distribution.
- (gear_box): Add alternative *engine_thermostat_temperature* identification when missing *gear_box_powers_out*.
- (gear_box): Add alternative *gears* identification when missing *motive_powers*.
- (utils): Add *pairwise* function.
- (report): Add all vehicle summary info to augmented summary.
- (co2): Add scalar values to outputs for phases_co2_emissions.
- (report): Remove *numpy* warnings when comparing outputs vs targets.
- (summary): Add all comparisons in the augmented summary.
- (model): Add default to configure WLTP selection data.
- (selector): Add default to use all wltp input data as target in calibration.
- (selector): Add default to modify selector strategy.

11.7.2 Fix

- (excel): Update dice parameter parser.
- (schema): Remove unused functions.
- (excel): Add function to calculate vehicle mass from test mass.
- (excel): Remove *xgboost* warning.
- (schema): Correct schema error on error formatting.
- (excel): Remove *openpyxl* and *xgboost* warning.
- (excel): Ensure correct excel parsing of matrix tables.
- (load): Ensure excel reading specifying the engine.
- (cli): Correct typo.
- (excel): Rename *.wet.ta* file extension to *.jet.ta*.
- (excel): Ensure remote reading using xlref.
- (rtd): Correct documentation rendering in *rtd*.
- (excel): Update parser according to new *xlref*.
- (schema): Add pax into validation schema.
- (load): Add new file extension *.wet.ta*.
- (load): Correct parser for gear box ratios.
- (engine): Correct wrong link to calculate *full_load_speeds*.
- (engine): Disable *idle_model_detector* in case of hybrids.
- (hybrid): Add simple fc calibration model for hybrids to bypass *after_treatment_warm_up_phases*.
- (excel): Correct Ref class.
- (demos): Correct missing data in simplan.
- (co2): Correct wrong function name.
- (gear_box): Correct missing formula.

- (physical): Remove warnings.
- (final_drive) #571: Correct calculation of final drive powers.
- (batteries) #570: Implement constant model (i.e. r0 = 0).
- (templates): Remove unused hidden dice report.
- (cli): Correct logging level.
- (hybrid): Correct calculation of engine power losses when speed is zero.
- (gear_box): Correct gears identification for hybrid.
- (templates) #567: Correct typo in NEDC-L cell.
- (gear_box): Correct bug when *motive_powers is None*.
- (ta): Correct early closure of input file.
- (gear_box): Restructure loss model and correct thermal calculation.
- (gear_box): Improve performance of *calculate_gear_box_efficiencies_torques_temperatures* function.
- (docker): Updater pyinstaller version to 3.6.
- (docker): Correct requirements.
- (plot): Remove page caching from dsp plot.
- (fc): Correct calculation order for extended phases.
- (docker): Correct build script.
- (setup): Remove wtlp limitation dependency that brakes the setup.
- (physical): Remove syntax error warning.
- (write): Use *get_node* instead *search_node_description*.
- (cli) #564: Correct bug of *co2mpas sync template* cli.
- (core): Avoid numpy when import just co2mpas.
- (schema): Improve float parser.
- (doc): Add glossary links for *Time Series* and *General Terms*.
- (doc): Add iframe with interactive model graph.
- (doc): Add missing sub-model doc.
- (doc): Correct *extract_calibrated_model* link.
- (faq): FAQ link corrected.
- (faq): Updated where to download.
- (doc): Description of *has_periodically_regenerating_systems* according to 2017/1151.
- (doc) #563: Change to name, surname of the team members.
- (docs): executable name.

11.7.3 Other

- Update copyright.

11.8 v4.1.10 (2019-11-07)

11.8.1 Fix

- (optimization) #561: Use float32 for fmin error function.

11.9 v4.1.9 (2019-11-04)

11.9.1 Fix

- (excel): Correct parser for all-l.
- (schema): Correct error message for input file version.
- (template): Correct wrong cell reference.
- (setup): Fixed link setup.

11.10 v4.1.8 (2019-10-24): Wine Release

CO_2MPAS project has been split into multiple repositories (#506). The current repository contains just CO_2MPAS model. The other functionalities are installed as extra (i.e., DICE, GUI, sync).

11.10.1 Important changes:

The main changes made in this release regards:

GUI

A new graphical user interface (GUI) has replaced the previous one.

Documentation

All documentation has been reviewed and updated (#533, #540). There are two new sections: FAQ, and Contributing to CO_2MPAS. The documentation is now stored in Read the Docs (see the site).

I/O Data & Demo

The input excel file has been updated to version 3.1.0. (#544), as per the 2019 amendments to Regulations (EU) 2017/1152 and 2017/1153.

The demo files have been reviewed and now four files are available (#544, #538):

1. *co2mpas_conventional.xlsx*: conventional vehicle,
2. *co2mpas_simplan.xlsx*: sample simulation plan,
3. *co2mpas_hybrid.xlsx*: hybrid parallel vehicle.
4. *co2mpas_plugin.xlsx*: hybrid plugin vehicle.

Model

- Implemented Hybrids Electric Model for parallel, planetary, and serial architectures (#516, #536, #540, #541). It consists of nine electric motors (i.e., P0, P1, P2 planetary, P2, P3 front, P3 rear, P4 front, P4 rear, and starter), one DC/DC converter, and two batteries (i.e., service and drive batteries).
- Improved the stability of the thermal model (#458, #498, #516), the gearbox identification (#551) and the alternator model.
- Corrected the calibration of the Start/Stop model (#512).
- Updated the torque converter model according to VDI253 standard (#515).
- Refined the cylinder deactivation model (#517).
- Implemented parser for PAX tyre code (#507).
- Added formulas to calculate the corrected CO_2 emissions according to WLTP and NEDC regulations (#539).

11.10.2 Known Limitations

1. Certain programs (for example Skype) could be pre-empting (or reserving) some tcp/ip ports and therefore could conflict with CO_2MPAS graphical interface that tries to launch a web server on a port in the higher range (> 10000).
2. Certain antivirus (for example Avast) could include python in the list of malicious software; however, this is not to be considered harmful. If this happens the antivirus should be disabled when running CO_2MPAS, or a special exclusion should be granted to the CO_2MPAS executable.
3. If CO_2MPAS is installed in Windows 7 without ServicePack-1, you will get an error like the following:

```
Error loading Python DLL 'C:\Users\admin\AppData\Local\Temp\_MEI60402\python36.dll'.
LoadLibrary: The specified procedure could not be found.
Error loading Python DLL 'C:\Users\admin\AppData\Local\Temp\_MEI59722\python36.dll'.
LoadLibrary: The specified procedure could not be found.
Delete file: C:\apps\co2mpas\pkgs\env.txt
Output folder: C:\apps\co2mpas\conda-meta
Extract: history
Creating CO2MPAS menus...
Error loading Python DLL 'C:\Users\admin\AppData\Local\Temp\_MEI51722\python36.dll'.
LoadLibrary: The specified procedure could not be found.
Execute: "C:\apps\co2mpas\pythonw.exe" -E -s "C:\apps\co2mpas\Lib\_nsis.py" mkdirs
Running post install...
```

(continues on next page)

(continued from previous page)

```
Execute: "C:\apps\co2mpas\pythonw.exe" -E -s "C:\apps\co2mpas\Lib\_nsis.py" post_
↪install
Created uninstaller: C:\apps\co2mpas\Uninstall-CO2MPAS.exe
Completed
```

4. If you use Internet Explorer version 9 or earlier, you might experience some problems (i.e., impossible to choose the input file for the synchronisation, etc..).

11.11 v3.0.0 (2019-01-29): "VOLO" Release

CO_2MPAS 3.0.X becomes official on February 1st, 2019.

- There will be an overlapping period with the previous official CO_2MPAS version **2.0.0** of 2 weeks (until February 15th).
- This release incorporates the amendments of the Regulation (EU) 2017/1153, 2018/2043 of 18 December 2018 to the Type Approval procedure along with few fixes on the software.
- The engineering-model is 100% the same with the 2.1.0, 30-Nov-2018: "DADO" Release and the version-bump (2.X.X –> 3.X.X) is just facilitation for the users, to recognize which release is suitable for the amended Correlation Regulations.
- The Type Approval mode (_TA_) of this release is **incompatible** with all previous Input File versions. The _Batch_ mode, for engineering purposes, remains compatible.
- the _TA_ mode of this release generates a single "_.zip_" output that contains all files used and generated by CO_2MPAS.
- This release is comprised of 4 python packages: co2sim, co2dice, co2gui, and co2mpas.

11.11.1 Installation

This release will not be distributed as an **AllInOne** (AIO) package. It is based on the 2.0.0, 31-Aug-2018: "Unleash" Release, launched on 1 September 2018. There are two options for installation:

1. Install it in your current working AIO-v2.0.0.
2. **Preferably** in a clean AIO-v2.0.0, to have the possibility to use the old CO_2MPAS-v2.0.0 + DICE2 for the two-week overlapping period;

- **To install:**

  `console pip uninstall co2sim co2dice co2gui co2mpas -y pip install co2mpas `

Note: If you want to install this specific version at a later date, after more releases have happened, use this command: `console pip install co2mpas==3.0.0 `

11.11.2 Important Changes

Model

No model changes.

IO Data

- Input-file version from 3.0.0 –> **3.0.1**. - It hosts a few modifications after interactions with users. - The input file contained in this release cannot run in older CO_2MPAS releases in the _TA_ mode.

DICE

- The old DICE2 is deprecated, and must not be used after the 15th of February,
- it is replaced by the centralized DICE3 server. There will be a new procedure to configure the keys to _sign_ and _encrypt_ the data.

Demo Files

- The input-file changed, and we have prepared new demo files to help the users adjust. Since we do not distribute an **AllInOne** package, you may download the new files:
  - **from the console:**
    ```
    `console co2mpas demo --download `
    ```
  - From this link

11.12 v2.0.0 (2018-08-31): "Unleash" Release

Changes since 1.7.4.post0:

11.12.1 Breaking:

1. The `pip` utility contained in the old AIO is outdated (9.0.1) and cannot correctly install the transitive dependencies of new `|co2mpas|`, even for development purposes. Please upgrade your `pip` before following the installation or upgrade instructions for developers.
2. The `vehicle_family_id` format has changed (but old format is still supported):

   ```
   OLD: FT-TA-WMI-yyyy-nnnn
   NEW: FT-nnnnnnnnnnnnnnn-WMI-x
   ```

3. The CO_2MPAS python package has been splitted (see #408), and is now served by 4 python packages listed below. In practice this means that you can still receive bug-fixes and new features for the DICE or the GUI, while keeping the simulation-model intact.
   1. `co2sim`: the simulator, for standalone/engineering work. Now all IO-libraries and graph-drawing are optional, specified the `io` & `plot` "extras". If you need just the simulator to experiment, you need this command to install/upgrade it with:

```
pip install co2sim[io,plot] -U
```

2. `co2dice`: the backend & commands for DICE (Distributed Impromptu Co2mpas Evaluation).
3. `co2gui`: the GUI.
4. `co2mpas`: installs all of the above, and `[io,plot]` extras.

The relationships between the sub-projects are depicted below:

```
co2sim[io,plot]
  |    |
  |  co2dice
  |  /  \
 co2gui  WebStamper
   |
co2mpas
```

Note: `co2sim` on startup checks if the old `co2mpas-v1.x` is still installed, and aborts In that case, uninstall all projects and re-install them, to be on the safe side, with this commands:

```
pip uninstall -y co2sim co2dice co2gui co2mpas
pip install co2sim co2dice co2gui co2mpas -U
```

Model

- feat(co2_emissions): Add `engine_n_cylinders` as input value and a TA parameter.
- feat(ta): New TA output file.

  Running CO2MPAS in TA mode, will produce an extra file containing the DICE report. This file will be used in the feature version of DICE.
- feat(core): Improve calibration performances 60%.
- feat(manual): Add a manual prediction model according GTR.
- feat(gearbox): Add utility to design gearbox ratios if they cannot be identified based on `maximum_velocity` and `maximum_vehicle_laden_mass`.

  This is not affecting the TA mode.
- fix(co2mpas_template.xlsx): The parameter "Vehicle Family ID" changes to "Interpolation Family ID".
- fix(co2mpas_template.xlsx): Meta data.

  Add additional sheets for meta data. As for September 2018, the user can voluntarily add data related to the all WLTP tests held for a specific Interpolation Family ID. Since this addition is optional, the cells are colored orange.
- fix(vehicle): Default `n_dyno_axes` as function of `n_wheel_drive` for wltp (4wd–>2d, 2wd–>1d).

  If nothing is specified, default values now are: `n_dyno_axes = 1` `n_wheel_drive = 2`

  If only `n_wheel_drive` is selected, then the default for `n_dyno_axes` is calculated as function of `n_wheel_drive` for wltp (4wd–>2d, 2wd–>1d)

  If only n_dyno_axes is selected, then the default for `n_wheel_drive` is always 2.

- fix(vva): Remove `_check_vva`. `engine_has_variable_valve_actuation = True` and `ignition_type = 'compression'` is permitted.
- fix(ki_factor): Rename `ki_factor` to `ki_multiplicative` and add `ki_additive value`.
- fix(start_stop): Disable `start_stop_activation_time` when `has_start_stop == True`.
- fix(co2_emission): Disable `define_idle_fuel_consumption_model` when *idle_fuel_consumption* is not given.
- fix(ta): Disable function *define_idle_fuel_consumption_model* and *default_start_stop_activation_time*.
- fix(electrics): Improve calculation of state of charges.
- fix(at): Correct `correct_gear_full_load` method using the best gear instead the minimum when there is not sufficient power.

IO Data

- BREAK: Bumped input-file version from `2.2.8 --> 2.3.0`. And improved file-version comparison
- CHANGE: Changed *vehicle_family_id* format, but old format is still supported (#473):

```
OLD: FT-TA-WMI-yyyy-nnnn
NEW: FT-nnnnnnnnnnnnnnn-WMI-x
```

- feat: Input-template provide separate H/L fields for both *ki multiplicative* and *Ki additive* parameters.
- drop: remove deprecated `co2mpas gui` sub-command - `co2gui` top-level command is the norm since January 2017.

Dice

- FEAT: Added a new **"Stamp" button** on the GUI, stamping with *WebStamper* in the background in one step; internally it invokes the new `dicer` command (see below)(#378).
- FEAT: Added the simplified top-level sub-command `co2dice dicer` which executes *a sequencer of commands* to dice new **or existing** project through *WebStamper*, in a single step.:

```
co2dice dicer -i co2mpas_demo-1.xlsx -o O/20180812_213917-co2mpas_demo-1.xlsx
```

  Specifically when the project exists, e.g. when clicking again the *GUI-button*, it compares the given files *bit-by-bit* with the ones present already in the project, and proceeds *only when there are no differences*. Otherwise (or on network error), falling back to cli commands is needed, similar to what is done with abnormal cases such as `--recertify`, over-writing files, etc.
- All dice-commands and *WebStamper* now also work with files, since *Dices* can potentially be MBs in size; **Copy + Paste** becomes problematic in these cases.
- Added low-level `co2dice tstamp wstamp` cli-command that Stamps a pre-generated Dice through *WebStamper*.
- FEAT: The commands `co2dice dicer|init|append|report|recv|parse` and `co2dice tstamp wstamp`, support one or more `--write-file <path>/-W` options, to and every time they run, they can *append* or *overwrite* into all given `<path>` these 3 items as they are generated/received:
  1. Dice report;
  2. Stamp (or any errors received from WebStamper);

3. Decision.

By default, one <path> is always `~/.co2dice/reports.txt`, so this becomes the de-facto "keeper" of all reports exchanged (to mitigate a *known limitation* about not being able to retrieve old *stamps*). The location of the *reports.txt* file is configurable with

- `c.ReportsKeeper.default_reports_fpath` configuration property, and/or
- `CO2DICE_REPORTS_FPATH` (the env-var takes precedence).

- feat: command `co2dice project report <report-index>` can retrieve older reports (not just the latest one). Negative indexes count from the end, and need a trick to use them:

```
co2dice project report -- -2
```

  There is still no higher-level command to retrieveing *Stamps* (an old *known limitation*); internal git commands can do this.

- drop: deprecate all email-stamper commands; few new enhancements were applied on them.
- feat(#466, #467, io, dice): Add `--with-inputs` on `co2dice project init|append|report|dicer` commands that override flag in user-data *.xlsx* file, and attached all inputs encrypted in dice.
- feat: add 2 sub-commands in *report* standalone command:

```
co2dice report extract  # that's the old `co2dice report`
co2dice report unlock   # unlocks encrypted inputs in dice/stamps
```

- feat(dice): all dice commands accept `--quiet/-q` option that along with `--verbose/-v` they control the eventual logging-level.

  It is actually possible to give multiple *-q* / *-v* in the command line, and the verbose level is an algebraic additions of all of them, starting from *INFO* level.

  BUT if any -v is given, the *Spec.verbosed* trait-parameter is set to true. (see #476, #479).

- doc: small fixes on help-text of project commands.
- feat(dice): prepare the new-dice functionality of `tar`ing everything (see #480).

  The new `flag.encrypt_inputs` in input-xlsx file, configured by `ENCRYPTION_KEYS_PATH`, works for dice-2 but not yet respected by the old-dice commands; must revive #4de77ea1e.

- refact: renamed various internal classes and modules for clarity.

Various

- FIX: Support *pip >= 10+* (see #26).
- break: changed cmd-line scripts entry-points; if you install from sources, remember to run first: `pip install -e {co2mpas-dir}`
- Pinned versions of dependencies affecting the accuracy of the calculations, to achieve stronger reproducibility; these dependent libraries are shiped with AIO (see #427).
- Accurate versioning of project with polyvers.
- feat: provide a *docker* script, ensuring correct *numpy-base+MKL* installed in *conda* requirements.
- WebStamp: split-off *v1.9.0a1* as separate sub-project in sources.

11.12.2 Known Limitations

- Reproducibility of results has been greatly enhanced, with quasi-identical results in different platforms (*linux/Windows*).
- DICE:
  - Fixed known limitation of *1.7.3* (#448) of importing stamps from an older duplicate dice.
  - It is not possible to `-recertify` from `nedc` state (when mored files have been appended after stamping).
  - There is still no high level command to view Stamps (see low-level command in the old known limitation item). But stamps received are now save in `~/.co2dice/reports.txt` (along with dices and decisions).
  - The decision-number generated still never includes the numbers 10, 20, ...90.
  - All previous known limitations regarding mail-stamper still apply. But these commands are now *deprecated*.

11.13 Intermediate releases for `2.0.x`:

Note:

- Releases with `r` prefix signify version published in *PyPi*.
- Releases with `v` prefix signify internal milestones.

11.13.1 `|co2mpas|-r2.0.0.post0`, 1 Sep 2018

doc: Just to fix site and *PyPi* landing page.

11.13.2 `r2.0.0`, 31 Aug 2018

- fix: hide excess warnings.

11.13.3 `co2sim/co2gui: v2.0.0rc3, co2dice/webstamper: v2.0.0rc1`, 30 Aug 2018

- FIX: Print remote-errors when WebStamper rejects a Dice.
- fix: WebStamper had regressed and were reacting violently with http-error=500 ("server-failure") even on client mistakes; now they became http-error=400.
- fix: eliminate minor deprecation warning about XGBoost(seed=) keyword.

11.13.4 v2.0.0rc2 for co2sim & co2gui, 28 Aug 2018

- FIX: add data (xlsx-files & icons) to *co2sim* & *co2gui* wheels.
- `v2.0.0rc1` tried but didn't deliver due to missing package-data folders.

11.13.5 v2.0.0rc0, 24 Aug 2018

- DROP: make `co2deps` pinning-versions project disappear into the void, from where it came from, last milestone. Adding a moribund co2-project into PyPi (until pip bug pypa/pip#3878 gets fixed) is a waste of effort.
- ENH: extracted `plot` extras from `co2sim` dependencies. Significant work on all project dependencies (#408, #427 & #463). Coupled with the new `wltp-0.1.0a3` & `pandalone-0.2.4.post1` releases, now it is possible to use CO_2MPAS-simulator with narrowed-down dependencies (see docker-image size reduction, above).
- REFACT: separated DICE from SIM subprojects until really necessary (e.g. when extracting data from appended files). Some code-repetition needed, started moving utilities from `__main__.py` into own util-modules, at least for *co2dice*.
- ENH: update alpine-GCC in *docker* with recent instructions,and eventually used the debian image, which ends up the same size with less fuzz. Docker-image *co2sim* wheel is now created *outside of docker* with its proper version-id of visible; paths updated, scripts enhanced, files documented.
- ENH: *setup.py* does not prevent from running in old Python versions (e.g to build *wheels* in Py-2, also in #408).
- feat: dice-report encryption supports multiple recipients.
- feat: gui re-reads configurations on each DICE-button click.
- chore: add *GNU Makefiles* for rudimentary support to clean, build and maintain the new sub-projectrs.

11.13.6 v2.0.0b0, 20 Aug 2018

- BREAK: SPLIT CO2MPAS(#408) and moved packages in `.sub-dir/src/`:
  1. `co2sim[io]`: *root*`/pCO2SIM`
  2. `co2dice`: *root*`/pCO2DICE`
  3. `co2gui`: *root*`/pCO2GUI`
  4. `co2deps`: *root*`/pCO2DEPS`
  5. `co2mpas[pindeps]`: *root*
  6. `WebStamper`: *root*`/pWebStamper`
  - Also extracted `io` extras from `co2sim` dependencies.
- enh: use *GNU Makefile* for developers to manage sub-projects.
- enh: Dice-button reloads configurations when clicked (e.g. to read `WstampSpec.recpients` parameter if modified by the user-on-the-spot).
- enh: dice log-messages denote reports with line-numberss (not char-nums).

11.14 Intermediate releases for `1.9.x`:

11.14.1 `v1.9.2rc1`, 16 Aug 2018

- FIX: GUI mechanincs for logs and jobs.
- fix: finalized behavior for button-states.
- enh: possible to mute email-stamper deprecations with `EmailStamperWarning.mute`.
- enh: RELAX I/O file-pairing rule for `dicer` cmd, any 2 io-files is now ok.

11.14.2 `v1.9.2rc0`, 14 Aug 2018 (BROKEN GUI)

- ENH: Add logging-timestamps in `~/.co2dice/reports.txt` maintained by the `ReportsKeeper`(renamed from ``FileWritingMixin`) which now supports writing to multiple files through the tested *logging* library.
- enh: make location of the *reports.txt* file configurable with:
  - `c.ReportsKeeper.default_reports_fpath` property and
  - `CO2DICE_REPORTS_FPATH` (env-var takes precedence).
- REFACT: move DicerCMD (& DicerSpec) in their own files and render them top-level sub-commands. Also renamed modules:
  - `baseapp --> cmdlets` not to confuse with `base` module.
  - `dice --> cli` not to confuse with `dicer` module and the too-overloaded :term;`dice`.
- enh: replace old output-clipping machinery in `tstamp recv` with shrink-slice.
- enh: teach GUI to also use HTTP-sessions (like `dicer` command does).
- GUI-state behavior was still not mature.

11.14.3 `r1.9.1b1`, 13 Aug 2018

- FIX: `project dicer` command and GUI new *Dice-button* were failing to compare correctly existing files in project with new ones.

  Enhanced error-reporting of the button.
- doc: Update DICE-changes since previous major release.
- doc: Add glossary terms for links from new data in the excel input-file .
- doc: updated the dice changes for the forthcoming major-release, above
- dev: add "scafolding" to facilitate developing dice-button.

11.14.4 v1.9.1b0, 10 Aug 2018

- FEAT: Finished implementing the GUI "Stamp" button (it appends also new-dice *tar*, see #378).
- Retrofitted *project dice* command into a new "DICER" class, working as *a sequencer of commands* to dice new **or existing** projects through *WebStamper* only. Specifically now it compares the given files with the ones already in the project. Manual intervention is still needed in abnormal cases (`--recertify`, over-writing files, etc).
- **Added WebAPI + *co2dice tstamp wstamp* cli-commands to check stamps and** connectivity to WebStamper.
- **Renamed cmd `project dice --> dicer` not to overload the *dice* word; it is** a *sequencer* after all.
- feat: rename `-W=~/co2dice.reports.txt --> ~/.co2dice/reports.txt` to reuse dice folder.
- drop: removed *co2dice project tstamp* command, deprecated since 5-may-2017.
- enh: *project dicer* cmd uses HTTP-sessions when talking to WebStamper, but not the GUI button yet.
- fix: `-W--write-fpath` works more reliably, and by defaults it writes into renamed `~/.co2dice/reports.txt`.

11.14.5 v1.9.1a2, 10 Aug 2018

Fixes and features for the GUI *Stamp-button* and supporting `project dice` command.

- FEAT: `co2dice project dicer|init|append|report|recv|parse` and the `co2dice tstamp wstamp` commands, they have by default `--write-file=~/.co2dice/reports.txt` file, so every time they run, they *APPENDED* into this file these 3 items:
  1. Dice report;
  2. Stamp (or any errors received from the WebStamper);
  3. Decision.
- doc: deprecate all email-stamper commands; few new enhancements were applied on them.
- drop: remove deprecated `co2mpas gui` cmd - *co2gui* is the norm since Jan 2017.
- doc: small fixes on help-text of project commands.
- refact: extract dice-cmd functionality into its own Spec class.
- sources: move `tkui.py` into it's own package. (needs re-install from sources).
- WIP: Add GUI "Stamp" button that appends also new-dice *tar* (see #378).

11.14.6 v1.9.1a1, 10 Aug 2018

Implement the new `project dice` command.

- Work started since *v1.9.1a0: 8 Aug 2018*.
- FEAT: NEW WEB-API CMDS: - `co2dice project dicer`: Dice a new project in one action through WebStamper. - `tstamp wstamp`: Stamp pre-generated Dice through WebStamper.
- feat: `co2dice project report` command can retrieve older reports. (not just the latest). For *Stamps*, internal git commands are still needed.
- WIP: Add GUI "Stamp" button.

11.14.7 r1.9.0b2, 7 Aug 2018

Version in *PyPi* deemed OK for release. Mostly doc-changes since *b1*.

11.14.8 v1.9.0b1, 2 Aug 2018

More changes at input-data, new-dice code and small model changes. Not released in *PyPi*.

- feat(dice): teach the options `--write-fpath/-W` and `--shrink` to the commands:

  ```
  co2dice project (init|append|report|parse|trecv)
  ```

  so they can write directly results (i.e. report) in local files, and avoid printing big output to the console (see #466). *WebStamper* also works now with files, since files can potentially be Mbs in size.
- feat(dice): teach dice commands `--quiet/-q` option that along with `--verbose/-v` they control logging-level. It is actually possible to give multiple *-q* / *-v* in the command line, and the verbose level is an algebraic additions of all of them, starting from *INFO* level. BUT if any -v is given, the *Spec.verbosed* trait-parameter is set to true. (see #476, #479).
- feat(dice): prepare the new-dice functionality of taring everything (see #480). Add `flag.encrypt_inputs` in input-xlsx file, configured by `ENCRYPTION_KEYS_PATH`, but not yet respected by the dice commands; must revive #4de77ea1e.
- feat(WebStamper): Support Upload dice-reports from local-files & Download Stamp to local-files.
- fix(dice): fix redirection/piping of commands.
- fix(site): Update to latest *schedula-2.3.x* to fix site-generation (see #476, #e534168b).
- enh(doc): Update all copyright notices to "2018".
- refact(sources): start using `__main__.py` also for dice, but without putting too much code in it, just for **PEP 366** relative-imports to work.

11.14.9 r1.9.0b0, 31 Jul 2018

1st release with new-dice functionality.

11.14.10 v1.9.0a2, 11 Jul 2018

- WebStamp: split-off *v1.9.0a1* as separate sub-project in sources.

IO Data

- IO: Input-template provide separate H/L fields for both *ki multiplicative* and *Ki additive* parameters.

11.14.11 v1.9.0a1, 5 Jul 2018

Bumped *minor* number to signify that the VF_ID and input-file version have changed forward-incompatibly. Very roughly tested (see #472). (*v1.9.0a0* was a checkpoint after *VF_ID* preliminary changes).

- CHANGE: Changed *vehicle_family_id* format, but old format is still supported (#473):

```
OLD: FT-TA-WMI-yyyy-nnnn
NEW: FT-nnnnnnnnnnnnnnn-WMI-x
```

- BREAK: Bumped input-file version from `2.2.8 --> 2.3.0`. And improved file-version comparison (Semantic Versioning)
- fix: completed transition to *polyversion* monorepo scheme.
- docker: ensure correct *numpy-base+MKL* installed in *conda* requirements.

Model

- FIX: Gear-model does not dance (#427).
- fix: remove some pandas warnings

11.15 Intermediate releases for 1.8.x:

11.15.1 v1.8.1a2, 12 Jun 2018

Tagged as `co2mpas_v1.8.1a0` just to switch *polyversion* repo-scheme, from *mono-project –> monorepo* (switch will complete in next tag).

- feat(#466, #467, io, dice): Add `--with-inputs` on `report` commands that override flag in user-data *.xlsx* file, and attached all inputs encrypted in dice.
- Add 2 sub-commands in *report* standalone command:

```
co2dice report extract  # that's the old `co2dice report`
co2dice report unlock   # unlocks encrypted inputs in dice/stamps
```

- testing #375: - dice: need *pytest* to run its TCs. - dice: cannot run all tests together, only one module by one. All pass

11.15.2 v1.8.0a1, 7 Jun 2018

- FIX dice, did not start due to *polyversion* not being engraved.
- The `CO2MPARE_ENABLED` fails with:

```
ERROR:co2mpas_main:Invalid value '1' for env-var[CO2MPARE_ENABLED]!
  Should be one of (0 f false n no off 1 t true y yes on).
```

11.15.3 v1.8.0a0, 6 Jun 2018

PINNED REQUIRED VERSIONS, served with AIO-1.8.1a1

11.15.4 v1.8.0.dev1, 29 May 2018

- chore:(build, #408, #0761ba9d6): Start versioning project with *polyvers* tool, as *mono-project*.
- feat(data, #???): Implemented *co2mparable* generation for ex-post reproducibility studies.

11.15.5 v1.8.0.dev0, 22 May 2018

Included in 1st AIO-UpgradePack (see #463).

- chore(build, #e90680fae): removed *setup_requires*; must have these packages installed before attempting to install in "develop mode":

```
pip, setuptools setuptools-git >= 0.3, wheel, polyvers
```

- feat(deps): Add *xgboost* native-lib dependency, for speed.

11.15.6 Pre-v1.8.0.dev0, 15 Nov 2017

- feat(model): Add utility to design gearbox ratios if they cannot be identified based on *maximum_velocity* and *maximum_vehicle_laden_mass*. This is not affecting the TA mode.
- feat(model): Add function to calculate the *vehicle_mass* from *curb mass*, *cargo_mass*, *curb_mass*, *fuel_mass*, *passenger_mass*, and *n_passengers*. This is not affecting the TA mode.
- Dice & WebStamper updates…

11.16 Intermediate releases for 1.7.x:

11.16.1 v1.7.4.post3, 10 Aug 2018

Settled dependencies for **pip** and **conda** environments.

11.16.2 v1.7.4.post2, 8 Aug 2018

- Fixed regression by "piping to stdout" of previous broken release *1.7.1.post1*.
- Pinned dependencies needed for downgrading from *v1.9.x*.

  Transitive dependencies are now served from 2 places:

  - `setup.py`: contains bounded dependency versions to ensure proper functioning, but not reproducibility.

    These bounded versions apply when installing from *PyPi* with command `pip instal co2mpas==1.7.4.post2`; then **pip** will install dependencies with as few as possible transitive re-installations.

  - `requirements/exe.pip` & `requirements/install_conda_reqs.sh`: contain the *pinned* versions of all calculation-important dependent libraries (see #463).

    You need to get the sources (e.g. git-clone the repo) to access this file, and then run the command `pip install -r <git-repo>/requirements/exe.pip`.

11.16.3 `v1.7.4.post1`, 3 Aug 2018 (BROKEN!)

Backport fixes to facilitate comparisons with forthcoming release 1.9+.

- Support *pip* >= *10*+ (fixes #26).
- Fix conflicting *dill* requirement.
- Fix piping dice-commands to stdout.

11.16.4 v1.7.4.post0, 11 Dec 2017

Never released in *PyPi*, just for fixes for WebStamper and the site for "Toketos".

- feat(wstamp): cache last sender+recipient in cookies.

11.16.5 v1.7.4, 15 Nov 2017: "Toketos"

- feat(dice, #447): Allow skipping `tsend -n` command to facilitate WebStamper, and transition from `tagged` –> `sample` / `nosample`.
- fix(co2p, #448): *tparse* checks stamp is on last-tag (unless forced). Was a "Known limitation" of previous versions.

11.16.6 v1.7.3.post0, 16 Oct 2017

- feat(co2p): The new option `--recertify` to `co2dice project append` allows to extend certification files for some vehile-family with new ones

  ---

  Note: The old declaration-files are ALWAYS retained in the history of "re-certified" projects. You may control whether they old files will be also visible in the new Dice-report or not.

  For the new dice-report to contain ALL files (and in in alphabetical-order), use *different* file names - otherwise, the old-files will be overwritten. In the later case, the old files will be visible only to those having access to the whole project, such as the TAAs after receiving the project's exported archive.

- fix(co2p): `co2dice project` commands were raising NPE exception when iterating existing dice tags, e.g. `co2dice project export .` to export only the current project raised:

```
AttributeError: 'NoneType' object has no attribute 'startswith'
```

- fix(tstamp): `co2dice tstamp` were raising NPE exceptions when `-force` used on invalid signatures.

Known Limitations

co2dice(#448): if more than one dice-report is generated for a project, it is still possible to parse anyone tstamp on the project - no check against the hash-1 performed. So practically in this case, the history of the project is corrupted.

11.17 v1.7.3, 16 August 2017: "T-REA" Release

- Dice & model fine-tuning.
- Includes changes also from **RETRACTED** `v1.6.1.post0`, 13 July 2017, "T-bone" release.

11.17.1 DICE

- feat(config): stop accepting test-key (`'CBBB52FF'`); you would receive this error message:

```
After July 27 2017 you cannot use test-key for official runs!

Generate a new key, and remember to re-encrypt your passwords with it.
If you still want to run an experiment, add `--GpgSpec.allow_test_key=True`
command-line option.
```

  You have to modify your configurations and set `GpgSpec.master_key` to your newly-generated key, and **re-encrypt your passowords in persist file.**

- feat(config): dice commands would complain if config-file(s) missing; remember to transfer your configurations from your old AIO (with all changes needed).
- feat(AIO): prepare for installing AIO in *multi-user/shared* environments; the important environment variable is `HOME` (read `[AIO]/.co2mpad_env.bat` file and run `co2dice config paths` command). Enhanced `Cmd.config_paths` parameter to properly work with *persistent* JSON file even if a list of "overlayed" files/folders is given.
- feat(config): enhance `co2dice config (desc | show | paths)` commands to provide help-text and configured values for specific classes & params and all interesting variables affecting configurations. (alternatives to the much coarser `--help` and `--help-all` options).
- Tstamping & networking:
  - feat(#382): enhance handling of email encodings on send/recv:
    - add configurations choices for *Content-Transfer-Enconding* when sending non-ASCII emails or working with Outlook (usually *'=0A=0D=0E'* chars scattered in the email); read help on those parameters, with this command:

      ```
      co2dice config desc transfer_enc  quote_printable
      ```

    - add `TstampSender.scramble_tag` & `TstampReceiver.un_quote_printable` options for dealing with non-ASCII dice-reports.
  - `(t)recv` cmds: add `--subject`, `--on` and `--wait-criteria` options for search criteria on the `tstamp recv` and `project trecv` subcmds;
  - `(t)recv` cmds: renamed `email_criteria-->rfc-criteria`, enhancing their syntax help;
  - `(t)parse` can guess if a "naked" dice-reports tags is given (specify `--tag` to be explicit).
  - `(t)recv` cmd: added `--page` option to download a "slice" of from the server.

  - improve `(t)parse` command's `dice` printout to include project/issuer/dates.
  - `(t)recv`: BCC-addresses were treated as CCs; `--raw` STDOUT was corrupted; emails received
  - feat(report): print out the key used to sign dice-report.
- Projects:
  - feat(project): store tstamp-email verbatim, and sign 2nd HASH report.
  - refact(git): compatible-bump of dice-report format-version: `1.0.0-->1.0.1`.
  - feat(log): possible to modify selectively logging output with `~/logconf.yaml` file; generally improve error handling and logging of commands.
  - `co2dice project export`:
    * fix(#18): fix command not to include dices from all projects.
    * feat(#423, #435): add `--out` option to set the out-fpath of the archive, and the `--erase-afterwards` to facilitate starting a project.

      Note: Do not (ab)use `project export --erase-afterwards` on diced projects.

  - `co2dice project open`: auto-deduce project to open if only one exists.
  - `co2dice project backup`: add `--erase-afterwards` option.

Known Limitations

- Microsoft Outlook Servers are known to corrupt the dice-emails; depending on the version and the configurations, most of the times they can be fixed. If not, as a last resort, another email-account may be used. A permanent solution to the problem is will be provided when the the *Exchange Web Services (EWS)* protocol is implemented in |*co2mpas*|.
- On *Yahoo* servers, the `TstampReceiver.subject_prefix` param must not contain any brackets (`[]`). The are included by default, so you have to modify that in your configs.
- Using GMail accounts to send Dice may not(!) receive the reply-back "Proof of Posting" reply (or it may delay up to days). Please perform tests to discover that, and use another email-provided if that's the case. Additionally, Google's security provisions for some countries may be too strict to allow SMTP/IMAP access. In all cases, you need to enable allow less secure apps to access your account.
- Some combinations of outbound & inbound accounts for dice reports and timsestamps may not work due to DMARC restrictions. JRC will offer more alternative "paths" for running Dices. All major providers (Google, Yahoo, Microsoft) will not allow your dice-report to be stamped and forwarded to `TstampSender.stamp_recipients` other than the Comission; you may (or may not) receive "bounce" emails explaining that.
- There is no high level command to view the stamp for some project; Assuming your project is in `sample` or `nosample` state, use this cmd:

```
cat %HOME%/.co2dice/repo/tstamp.txt
```

- The decision-number generated never includes the numbers 10, 20, ...90. This does not change the odds for `SAMPLE`/`NOSAMPLE` but it does affect the odds for double-testing *Low* vs *High* vehicles (4 vs 5).

11.17.2 Datasync

- #390: Datasync was producing 0 values in the first and/or in the last cells. This has been fixed extending the given signal with the first and last values.
- #424: remove buggy interpolation methods.

11.17.3 Model-changes

- #d21b665, #5f8f58b, #33538be: Speedup the model avoiding useless identifications during the prediction phase.

Vehicle model

- #d90c697: Add road loads calculation from vehicle and tyre category.
- #952f16b: Update the *rolling_resistance_coeff* according to table A4/1 of EU legislation not world wide.
- #952f16b: Add function to calculate *aerodynamic_drag_coefficient* from vehicle_body.

Thermal model

- #169: Add a filter to remove invalid temperature derivatives (i.e., *abs(DT) >= 0.7*) during the cold phase.

Clutch model

- #330: Some extra RPM (peaks) has been verified before the engine's stops. This problem has been resolved filtering out *clutch_delta > 0* when *acc < 0* and adding a *features selection* in the calibration of the model.

Engine model

- #4c07751: The *auxiliaries_torque_losses* are function of *engine_capacity*.

CO2 model

- #350: Complete fuel default characteristics (LHV, Carbon Content, and Density).
- #2e890f0: Fix of the bug in *tau_function* when a hot cycle is given.
- #399: Implement a fuzzy rescaling function to improve the stability of the model when rounding the WLTP bag values.
- #401: Set co2_params limits to avoid unfeasible results.
- #402: Rewrite of *calibrate_co2_params* function.
- #391, #403: Use the *identified_co2_params* as initial guess of the *calibrate_co2_params*. Update co2 optimizer enabling all steps in the identification and disabling the first two steps in the calibration. Optimize the parameters that define the gearbox, torque, and power losses.

11.17.4 IO & Data:

- fix(xlsx, #426): excel validation formulas on input-template & demos did not accept *vehicle-family-id* with single-digit TA-ids.
- #314, gh:*410*: MOVED MOST DEMO-FILES to AIO archive - 2 files are left. Updated `|co2mpas| demo` command to use them if found; add `--download` option to get the very latest from Internet.
- main: rename logging option `--quite` –> `--quiet`.
- #380: Add cycle scores to output template.
- #391: Add model scores to summary file.
- #399: Report *co2_rescaling_scores* to output and summary files.
- #407: Disable input-file caching by default (renamed option `--override-cache --> use-cache`.

Known Limitations

- The `co2mpas modelgraph` command cannot plot flow-diagrams if Internet Explorer (IE) is the default browser.

11.17.5 GUI

- feat: `co2gui` command does not block, and stores logs in temporary-file. It launches this file in a text-editor in case of failures.
- feat: remember position and size between launches (stored in *persistent* JSON file).

11.17.6 AIO

- Detect 32bit Windows early, and notify user with an error-popup.
- Possible to extract archive into path with SPACES (not recommended though).
- Switched from Cygwin–>MSYS2 for the POSIX layer, for better support in Windows paths, and *pacman* update manager. Size increased from ~350MB –> ~530MB.
  - feat(install): reimplement cygwin's *mkshortcut.exe* in VBScript.
  - fix(git): use *cygpath.exe* to convert Windows paths and respect mount-points (see GitPython#639).
- Use `[AIO]` to signify the ALLINONE base-folder in the documentation; use it in CO_2MPAS to suppress excessive development warnings.

CHAPTER

TWELVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

C

INDEX

Symbols

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X