odysseus

...

Mar 23, 2021

API REFERENCE:

CHAPTER

ONE

CITY DATA MANAGER

1.1 City Data Source

1.1.1 Geo Data Source

1.1.2 Trips Data Gatherer

class DataGatherer(*output_path*, *structured_dataset_name*)

Class for automatically downloading data relating to the New York Citi bike sharing operator from a remote database.

Parameters

- **output_path** (*str*) – path in which to store the file
- **structured_dataset_name** –

bulk_download(*standardize=False*)

download all the datasets available at official citi bike website :param standardize: :type standardize: bool, optional :return:

download_data(*year*, *month*)

Download data for a specific month and year.

Parameters

- **year** (*int*) – year expressed as a four-digit number (e.g. 1999)
- **month** (*int*) – month expressed as a number (e.g. for November the method expects to receive 11)

Returns nothing

structured_dataset_name

get from official website the lists of all downloadable csvs dataset_names[yyyymm] = dataset_name_to_attach_to_root_url

1.1.3 Trips Data Source

All the classes of this module are implementations of the abstract class **TripsDataSource** that we report below.

TripsDataSource class

class TripsDataSource(*city_name*, *data_source_id*, *vehicles_type_id*)

TripsDataSource is an abstract class that contains the information needed to describe a trip. This class is implemented by the other classes of this module. The constructor method takes as parameters:

Parameters

- **city_name** (*str*) – City name. The name also serves to determine the timezone to which the city belongs
- **data_source_id** (*str*) – Data source from which the information is taken. This allows us to have multiple data sources associated with the same city (for example from different operators)
- **vehicles_type_id** (*str*) – Type of service represented by the data source (e.g. car sharing or e-scooter)

load_norm(*year*, *month*)

Load a previously created normalized file from memory. It requests month and year as parameters, and checks if the file for that period exists in memory (looking for it with the same format as *save_norm* in the city folder). If it exists, it returns a pandas.DataFrame containing the data read, otherwise it returns an empty DataFrame

Parameters

- **year** (*int*) – year expressed as a four-digit number (e.g. 1999)
- **month** (*int*) – month expressed as a number (e.g. for November the method expects to receive 11)

Returns If the file exists, it returns a pandas.DataFrame containing the data read, otherwise it returns an empty DataFrame

load_raw()

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise()

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

save_norm()

It stores normalized data both in a csv file and in a pickle file. The files produced are of the format *<year>_<month number>.csv* (or .pickle). For example *2017_11.csv*.

Returns nothing

The City Data Source module is divided into three submodules that deal with adapting the data format. Data from different sources have different formats: for example, geographic positions can be indicated as GPS coordinates or the

city can be divided into a grid and the cell in which you are located can be indicated. Geo Data Source takes care of standardizing geographic information.

1.2 City Geo Trips

1.2.1 CityGeoTrips class

The City Data Manager module takes care of data preprocessing. The simulator supports heterogeneous data sources thanks to this module which, starting from a generic input data format, transforms them following the same format adopted by the other simulator modules. The module is divided into two submodules, City Geo Trips and City Data Source.

CHAPTER

TWO

DEMAND MODELLING

CHAPTER

THREE

SUPPLY MODELLING

CHAPTER

FOUR

E3F2S

4.1 city_data_manager_dashboard module

4.2 e3f2s package

4.2.1 Subpackages

e3f2s.city_data_manager package

Subpackages

e3f2s.city_data_manager.city_data_source package

Subpackages

e3f2s.city_data_manager.city_data_source.geo_data_source package

Submodules

e3f2s.city_data_manager.city_data_source.geo_data_source.austin_census_tracts module

e3f2s.city_data_manager.city_data_source.geo_data_source.calgary_hexagonal_grid module

e3f2s.city_data_manager.city_data_source.geo_data_source.chicago_census_tracts module

e3f2s.city_data_manager.city_data_source.geo_data_source.chicago_community_areas module

e3f2s.city_data_manager.city_data_source.geo_data_source.geo_data_source module

e3f2s.city_data_manager.city_data_source.geo_data_source.minneapolis_centerlines module

e3f2s.city_data_manager.city_data_source.geo_data_source.minneapolis_trails_bikes module

e3f2s.city_data_manager.city_data_source.geo_data_source.norfolk_census_tracts module

Module contents

e3f2s.city_data_manager.city_data_source.trips_data_source package

Submodules

e3f2s.city_data_manager.city_data_source.trips_data_source.austin_scooter_trips module

class AustinScooterTrips

Bases: *e3f2s.city_data_manager.city_data_source.trips_data_source.trips_data_source.TripsDataSource*

load_raw()

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise(*year*, *month*)

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

e3f2s.city_data_manager.city_data_source.trips_data_source.big_data_db_trips module

class BigDataDBTrips(*city_name*)

Bases: *e3f2s.city_data_manager.city_data_source.trips_data_source.trips_data_source.TripsDataSource*

load_raw()

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise(*year*, *month*)

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

save_norm(*year*, *month*)

It stores normalized data both in a csv file and in a pickle file. The files produced are of the format *<year>_<month number>.csv* (or .pickle). For example *2017_11.csv*.

Returns nothing

e3f2s.city_data_manager.city_data_source.trips_data_source.calgary_scooter_trips module

class CalgaryScooterTrips

Bases: *e3f2s.city_data_manager.city_data_source.trips_data_source.trips_data_source.TripsDataSource*

load_raw()

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise(*year*, *month*)

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

e3f2s.city_data_manager.city_data_source.trips_data_source.chicago_scooter_trips module

class ChicagoScooterTrips

Bases: *e3f2s.city_data_manager.city_data_source.trips_data_source.trips_data_source.TripsDataSource*

load_raw()

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise(*year*, *month*)

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

e3f2s.city_data_manager.city_data_source.trips_data_source.kansas_city_scooter_trips module

e3f2s.city_data_manager.city_data_source.trips_data_source.louisville_scooter_trips module

class LouisvilleScooterTrips

Bases: *e3f2s.city_data_manager.city_data_source.trips_data_source.trips_data_source.TripsDataSource*

load_raw()

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise(*year*, *month*)

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

e3f2s.city_data_manager.city_data_source.trips_data_source.minneapolis_scooter_trips module

class MinneapolisScooterTrips

Bases: *e3f2s.city_data_manager.city_data_source.trips_data_source.trips_data_source.TripsDataSource*

load_raw (*year*, *month*)

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise (*year*, *month*)

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

e3f2s.city_data_manager.city_data_source.trips_data_source.new_york_city_bikes_trips module

e3f2s.city_data_manager.city_data_source.trips_data_source.new_york_city_taxi_trips module

e3f2s.city_data_manager.city_data_source.trips_data_source.norfolk_scooter_trips module

city_data_source.trips_data_source.trips_data_source module

class TripsDataSource (*city_name*, *data_source_id*, *vehicles_type_id*)

Bases: `object`

TripsDataSource is an abstract class that contains the information needed to describe a trip. This class is implemented by the other classes of this module. The constructor method takes as parameters:

Parameters

- **city_name** (*str*) – City name. The name also serves to determine the timezone to which the city belongs
- **data_source_id** (*str*) – Data source from which the information is taken. This allows us to have multiple data sources associated with the same city (for example from different operators)
- **vehicles_type_id** (*str*) – Type of service represented by the data source (e.g. car sharing or e-scooter)

load_norm (*year*, *month*)

Load a previously created normalized file from memory. It requests month and year as parameters, and checks if the file for that period exists in memory (looking for it with the same format as *save_norm* in the city folder). If it exists, it returns a pandas.DataFrame containing the data read, otherwise it returns an empty DataFrame

Parameters

- **year** (*int*) – year expressed as a four-digit number (e.g. 1999)

- **month** (*int*) – month expressed as a number (e.g. for November the method expects to receive 11)

Returns If the file exists, it returns a pandas.DataFrame containing the data read, otherwise it returns an empty DataFrame

load_raw()

Method for loading the data to be preprocessed. Since the data format differs in the various datasets, the method is left abstract. Each city has its own implementation. All implementations will read the data through the pandas readcsv method

Returns nothing

normalise()

This method is used to standardize the data format. Again the implementation is highly dependent on the data source and almost all modules override the method.

Returns A normalized pandas.DataFrame

save_norm()

It stores normalized data both in a csv file and in a pickle file. The files produced are of the format *<year>_<month number>.csv* (or .pickle). For example *2017_11.csv*.

Returns nothing

Module contents

Module contents

e3f2s.city_data_manager.city_geo_trips package

Submodules

e3f2s.city_data_manager.city_geo_trips.austin_geo_trips module

e3f2s.city_data_manager.city_geo_trips.big_data_db_geo_trips module

e3f2s.city_data_manager.city_geo_trips.calgary_geo_trips module

e3f2s.city_data_manager.city_geo_trips.chicago_geo_trips module

e3f2s.city_data_manager.city_geo_trips.city_geo_trips module

e3f2s.city_data_manager.city_geo_trips.kansas_city_geo_trips module

e3f2s.city_data_manager.city_geo_trips.louisville_geo_trips module

e3f2s.city_data_manager.city_geo_trips.minneapolis_geo_trips module

e3f2s.city_data_manager.city_geo_trips.norfolk_geo_trips module

e3f2s.city_data_manager.city_geo_trips.nyc_citi_bike_geo_trips module

Module contents

e3f2s.city_data_manager.config package

Submodules

e3f2s.city_data_manager.config.config module

Module contents

Module contents

e3f2s.demand_modelling package

Subpackages

e3f2s.demand_modelling.demand_model_configs package

Submodules

e3f2s.demand_modelling.demand_model_configs.default_config module

Module contents

Submodules

e3f2s.demand_modelling.demand_model module

e3f2s.demand_modelling.loader module

Module contents

e3f2s.simulator package

Subpackages

e3f2s.simulator.demand_model_validation package

Submodules

e3f2s.simulator.demand_model_validation.model_validation module

e3f2s.simulator.demand_model_validation.model_validation_plot module

e3f2s.simulator.demand_model_validation.model_validation_utils module

get_day_moments (*sim_reqs_eventG*, *sim_reqs_traceB*)

get_double_grouped_zones_errs (*sim_reqs_eventG*, *sim_reqs_traceB*, *group_cols*)

get_grouped_reqs_count (*group_col*, *sim_reqs_eventG*, *sim_reqs_traceB*)

get_grouped_zones_errs (*sim_reqs_eventG*, *sim_reqs_traceB*, *group_col*)

get_od_err (*grid*, *sim_reqs_eventG*, *sim_reqs_traceB*)

get_od_err_daymoments (*grid*, *sim_reqs_eventG*, *sim_reqs_traceB*)

get_plot_samples (*ia_threshold*, *sim_reqs_eventG*, *trace_timeouts*)

get_tot_zones_errs (*sim_reqs_eventG*, *sim_reqs_traceB*)

Module contents

e3f2s.simulator.multiple_runs package

Submodules

e3f2s.simulator.multiple_runs.multiple_runs module

e3f2s.simulator.multiple_runs.spark_multiple_runs module

Module contents

e3f2s.simulator.simulation package

Submodules

e3f2s.simulator.simulation.charging_primitives module

e3f2s.simulator.simulation.charging_strategies module

e3f2s.simulator.simulation.model_driven_simulator module

e3f2s.simulator.simulation.relocation_primitives module

e3f2s.simulator.simulation.relocation_strategies module

e3f2s.simulator.simulation.scooter_relocation_primitives module

e3f2s.simulator.simulation.scooter_relocation_strategies module

e3f2s.simulator.simulation.sim_metrics module

class SimMetrics (*metrics_dict*)
Bases: `object`

update_metrics (*metrics*, *value*)

e3f2s.simulator.simulation.simulator module

e3f2s.simulator.simulation.trace_driven_simulator module

e3f2s.simulator.simulation.vehicle_relocation_primitives module

e3f2s.simulator.simulation.vehicle_relocation_strategies module

Module contents

e3f2s.simulator.simulation_data_structures package

Submodules

e3f2s.simulator.simulation_data_structures.charging_station module

class ChargingStation (*env*, *num_poles*, *zone_id*, *station_conf*, *sim_scenario_conf*, *sim_start_time*)
Bases: `e3f2s.supply_modelling.charging_station.Pole`

charge (*vehicle*, *start_time*, *soc_delta_charging_trip*, *duration*)

monitor (*data*, *resource*)

e3f2s.simulator.simulation_data_structures.vehicle module

class Vehicle (*env*, *plate*, *start_zone*, *start_soc*, *vehicle_config*, *energymix_conf*, *sim_scenario_conf*, *sim_start_time*)
Bases: `e3f2s.supply_modelling.vehicle.Vehicle`

booking (*booking_request*)

charge (*percentage*)

e3f2s.simulator.simulation_data_structures.zone module

class Zone (*env*, *zone_id*, *sim_start_time*, *vehicles*)
Bases: `object`

add_vehicle (*t*)

remove_vehicle (*t*)

update_status (*t*)

Module contents

e3f2s.simulator.simulation_input package

Subpackages

e3f2s.simulator.simulation_input.sim_configs_versioned package

Subpackages

e3f2s.simulator.simulation_input.sim_configs_versioned.generalisation package

Subpackages

e3f2s.simulator.simulation_input.sim_configs_versioned.generalisation.fleet_size package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.generalisation.fleet_size.multiple_runs_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.generalisation.fleet_size.sim_general_conf module

Module contents

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2 package

Subpackages

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1a package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1a.multiple_runs_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1a.sim_general_conf module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1b package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1b.multiple_runs_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1b.sim_general_conf module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1c package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1c.multiple_runs_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set1c.sim_general_conf module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set2 package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set2.multiple_runs_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.isc2.isc2_set2.sim_general_conf module

Module contents

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo package

Subpackages

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_magic_relocat package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_magic_relocat module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_magic_relocat
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_no_relocation
package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_no_relocation.
module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_no_relocation.
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_relocation_str
package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_relocation_str
module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_relocation_str
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_test
package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_test.multiple_r
module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_eventG_test.sim_gene
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_traceB_test
package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_traceB_test.multiple_r
module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.louisville_multiple_runs_traceB_test.sim_gener
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_magic_relo
package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_magic_relo
module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_magic_relo
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_no_relocati
package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_no_relocati
module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_no_relocati
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_relocation_
package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_relocation_
module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_relocation_
module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_test package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_test.multipl module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_eventG_test.sim_ge module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_traceB_test package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_traceB_test.multipl module

e3f2s.simulator.simulation_input.sim_configs_versioned.leonardo.minneapolis_multiple_runs_traceB_test.sim_ge module

Module contents

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.test package

Subpackages

e3f2s.simulator.simulation_input.sim_configs_versioned.test.big_data_db_test package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.test.big_data_db_test.multiple_runs_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.test.big_data_db_test.sim_general_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.test.big_data_db_test.single_run_conf module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.test.city_single_run_test package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.test.city_single_run_test.sim_general_conf module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.test.mito_mobility_test package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.test.mito_mobility_test.sim_general_conf module

e3f2s.simulator.simulation_input.sim_configs_versioned.test.mito_mobility_test.single_run_conf module

Module contents

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.xian package

Subpackages

e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strategies_test package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strategies_test.multiple_runs_c module

e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strategies_test.sim_general_co module

Module contents

e3f2s.simulator.simulation_input.sim_configs_versioned.xian.turin_test package

Submodules

e3f2s.simulator.simulation_input.sim_configs_versioned.xian.turin_test.sim_run_conf module

Module contents

Module contents

Module contents

Submodules

e3f2s.simulator.simulation_input.costs_conf module

e3f2s.simulator.simulation_input.sim_config_grid module

class EFFCS_SimConfGrid(*conf_grid*)

Bases: `object`

e3f2s.simulator.simulation_input.sim_input module

e3f2s.simulator.simulation_input.sim_input_paths module

e3f2s.simulator.simulation_input.station_conf module

e3f2s.simulator.simulation_input.vehicle_conf module

Module contents

e3f2s.simulator.simulation_output package

Submodules

e3f2s.simulator.simulation_output.multiple_runs_plotter module

class EFFCS_MultipleRunsPlotter(*city*, *sim_scenario_name*, *sim_general_conf*, *sim_scenario_conf_grid*, *x_col*, *y_col*, *param_col*)

Bases: `object`

plot_x_y_param()

e3f2s.simulator.simulation_output.plot_multiple_runs module

plot_events_percentage(*sim_stats_df*, *x_col*, *title_add*, *figpath*, *figname*)

plot_param_cross_section(*results_df*, *x_col*, *y_col*, *param_col*, *figpath*, *figname*, *fixed_params_dict*)

e3f2s.simulator.simulation_output.sim_output module

class SimOutput(*sim*)

Bases: `object`

e3f2s.simulator.simulation_output.sim_output_plotter module

class EFFCS_SimOutputPlotter(*sim_output*, *city*, *sim_scenario_name*)

Bases: `object`

plot_charging_infrastructure()

plot_choropleth(*col*)

plot_city_zones()

plot_events_hourly_count_boxplot(*which_df*, *start_or_end*)

plot_events_profile_barh()

plot_events_t()

plot_fleet_status_t()

plot_hourly_relocost_boxplot()

plot_n_vehicles_charging_hourly_mean_boxplot()

e3f2s.simulator.simulation_output.sim_stats module

class SimStats

Bases: `object`

get_stats_from_sim(*sim*)

Module contents

e3f2s.simulator.single_run package

Submodules

e3f2s.simulator.single_run.get_eventG_input module

e3f2s.simulator.single_run.get_traceB_input module

e3f2s.simulator.single_run.run_eventG_sim module

e3f2s.simulator.single_run.run_traceB_sim module

e3f2s.simulator.single_run.single_run module

Module contents

Module contents

e3f2s.supply_modelling package

Subpackages

e3f2s.supply_modelling.supply_model_configs package

Submodules

e3f2s.supply_modelling.supply_model_configs.default_config module

Module contents

Submodules

e3f2s.supply_modelling.charging_station module

class Pole(*station_config*)

Bases: `object`

get_charging_time_from_energy(*energy_mj*)

get_energy_from_charging_time(*charging_time*)

get_fuelcost_from_energy(*energy_mj*)

e3f2s.supply_modelling.energymix_loader module

class EnergyMix(*city*, *year*)

Bases: `object`

evaluate_emissions()

evaluate_energy()

open_database()

e3f2s.supply_modelling.supply_model module

e3f2s.supply_modelling.vehicle module

class Vehicle(*vehicle_config*, *energy_mix_conf*)

Bases: `object`

consumption_to_percentage(*consumption*)

distance_to_consumption(*distance*)

distance_to_tanktowheel_emission(*distance*)

distance_to_welltotank_emission(*distance*)

from_kml_to_energyperkm()

from_kml_to_lkm()

get_charging_time_from_perc(*actual_level_perc*, *flow_amount*, *profile*, *beta=100*)

get_percentage_from_charging_time(*charging_time*, *flow_amount*, *profile*)

percentage_to_consumption(*percentage*)

tanktowheel_energy_from_perc(*percentage*)

welltotank_energy_from_perc(*percentage*)

Module contents

e3f2s.utils package

Submodules

e3f2s.utils.bookings_utils module

update_req_time_info(*booking_request*)

e3f2s.utils.cost_utils module

charging_station_lord_cost(*costs*)

get_fuelcost_from_energy(*fuel_type*, *fuel_costs*, *energy_mj*)

insert_scenario_costs(*df*, *sim_scenario_conf*, *vehicles_cost_conf*, *poles_cost_conf*)

insert_sim_costs(*df*, *sim_scenario_conf*, *fuel_costs*, *administrative_cost_conf*, *vehicles_cost_conf*)

e3f2s.utils.geospatial_utils module

e3f2s.utils.path_utils module

check_create_path(*path*)

e3f2s.utils.time_utils module

get_grouped_aggfunc(*df*, *group_cols*, *stats_col*, *aggfuncs*)

get_grouped_resampled_aggfunc(*df*, *group_cols*, *freq*, *stats_col*, *aggfuncs*)

get_grouped_resampled_count(*df*, *group_cols*, *freq*)

get_grouped_resampled_count_aggfunc(*df*, *group_cols*, *freq*, *aggfuncs*)

get_hourly_count_with_time_cols(*trips_df_norm*, *start_or_end*)

get_hourly_mean_with_time_cols(*df_norm*, *start_or_end*, *mean_col*)

get_resampled_aggfunc(*df*, *freq*, *stats_col*, *aggfuncs*)

get_resampled_grouped_aggfunc(*trips_df_norm*, *start_or_end*, *stats_col*, *time_categorical_col*, *freq*, *aggfunc*)

get_resampled_grouped_count_aggfunc(*trips_df_norm*, *start_or_end*, *time_group_col*, *freq*, *aggfunc*)

get_time_group_columns(*trips_df_norm*)

get_time_grouped_hourly_count(*df_norm*, *start_or_end*, *which_df*)

get_time_grouped_hourly_mean(*df_norm*, *start_or_end*, *which_df*, *mean_col*)

get_weekday_int_from_string(*s*)

get_weekday_string_from_int(*i*)

reshape_time_grouped_signature(*time_grouped_signatures*)

update_req_time_info(*booking_request*)

Module contents

4.2.2 Module contents

CHAPTER

FIVE

INTRODUCTION

e3f2s is a software for simulation of shared electric fleets in urban environments. It is still a prototype in active development. However, it is already stable enough to be used for research activities. In order to understand what e3f2s is capable of, please first read the three papers available in the folder */home/det_tesi/a.ciociola/input_simulator/papers*.

e3f2s is composed by three main modules:

- **city_data_manager**: receives in input data from different sources and output a simulation-ready version of the same data.
- **simulator**: contains data structures and algorithms for actual simulations.
- **utils**: contains some utility functions used across the other modules.

In this tutorial, we focus on running our first simulation with e3f2s.

CHAPTER

SIX

SETUP REPOSITORY, ENVIRONMENT AND DATA

First, let's clone the public git repository and move data into the right folder. For now, we skip explanations about *city_data_manager* functionalities.

```
git clone https://github.com/AleCioc/e3f2s my-e3f2s-folder
cp -r /home/det_tesi/a.ciociola/input_simulator/data my-e3f2s-folder/e3f2s/
↪city_data_manager
```

Then, let's install all the necessary libraries.

```
pip install --user -r my-e3f2s-folder/requirements.txt
```

CHAPTER

SEVEN

CONFIGURING SIMULATION INPUT

The folder *e3f2s/simulator/simulation_input* contains configuration files for simulation.

In particular:

- **sim_configs_target.json**: contains the name of the configuration to run
- **sim_configs_versioned**: contains one folder for each saved configuration e.g. *sim_configs_versioned/turin_iscc_set1* contains the configuration for the first set of simulation used in ISCC paper.

Each configuration folder must contain the following Python files:

- **sim_run_conf.py**: specifies used data source, run mode (single_run or multiple_runs), number of cores to use in case of multiple runs, simulation type (trace-driven or model-driven) and output folder name
- **sim_general_conf.py**: specifies macroscopic parameters about spatial and temporal configuration, as well as fleet load factor policy.
- **single_run_conf.py**: specifies scenario configuration for single run
- **model_validation_conf.py**: special case of single run
- **multiple_runs_conf.py**: specifies scenario configuration for multiple runs
- **vehicle_config.py**: specifies vehicles characteristics
- **cost_conf.py**: specifies cost/revenue configuration

Let's create a new folder for a new configuration:

```
cp -r /home/det_tesi/a.ciociola/input_simulator/ my-e3f2s-folder/e3f2s/
↪simulator/simulation_input/sim_configs_versioned/
```

Modify configurations as you desire, then run the simulator:

```
cd my-e3f2s-folder/
python -m e3f2s.simulator
```

Let's wait for simulation to finish and let's check the results folder and the figures folder (figures are created automatically only in single run mode)

```
ls my-e3f2s-folder/simulator/results/Torino/single_run/test
ls my-e3f2s-folder/simulator/figures/Torino/single_run/test
```

Done! Now we can explore our results and eventually produce other analysis and plots.

PYTHON MODULE INDEX

e

INDEX

module, 25
e3f2s.simulator.demand_model_validation
module, 15
e3f2s.simulator.demand_model_validation.model_validation_utils
module, 15
e3f2s.simulator.multiple_runs
module, 15
e3f2s.simulator.simulation
module, 16
e3f2s.simulator.simulation.sim_metrics
module, 16
e3f2s.simulator.simulation_data_structures
module, 17
e3f2s.simulator.simulation_data_structures.charging_station
module, 16
e3f2s.simulator.simulation_data_structures.vehicle
module, 16
e3f2s.simulator.simulation_data_structures.zone
module, 16
e3f2s.simulator.simulation_input
module, 23
e3f2s.simulator.simulation_input.costs_co
module, 23
e3f2s.simulator.simulation_input.sim_config_grid
module, 23
e3f2s.simulator.simulation_input.sim_con
module, 23
e3f2s.simulator.simulation_input.sim_con
module, 17
e3f2s.simulator.simulation_input.sim_con
module, 17
e3f2s.simulator.simulation_input.sim_con
module, 17
e3f2s.simulator.simulation_input.sim_con
module, 17
e3f2s.simulator.simulation_input.sim_con
module, 18
e3f2s.simulator.simulation_input.sim_con
module, 17
e3f2s.simulator.simulation_input.sim_con
module, 17
e3f2s.simulator.simulation_input.sim_con
module, 17
e3f2s.simulator.simulation_input.sim_con
module, 18
e3f2s.simulator.simulation_input.sim_con
module, 18
e3f2s.simulator.simulation_input.sim_con
module, 18
e3f2s.simulator.simulation_input.sim_con
module, 18
e3f2s.simulator.simulation_input.sim_con
module, 18
e3f2s.simulator.simulation_input.sim_con

module, 18
e3f2s.simulator.simulation_input.sim_configs_versio
module, 18
e3f2s.simulator.simulation_input.sim_configs_versio
module, 18
e3f2s.simulator.simulation_input.sim_configs_versio
module, 18
e3f2s.simulator.simulation_input.sim_configs_versio
module, 21
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 18
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 19
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 20
e3f2s.simulator.simulation_input.sim_configs_versio
module, 21
e3f2s.simulator.simulation_input.sim_configs_versio

module, 20
e3f2s.simulator.simulation_input.sim_conf
module, 20
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 22
e3f2s.simulator.simulation_input.sim_conf
module, 22
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 21
e3f2s.simulator.simulation_input.sim_conf
module, 22
e3f2s.simulator.simulation_input.sim_conf
module, 22
e3f2s.simulator.simulation_input.sim_conf
module, 22
e3f2s.simulator.simulation_input.sim_conf
module, 22
e3f2s.simulator.simulation_input.sim_conf
module, 22
e3f2s.simulator.simulation_input.sim_conf
module, 23
e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strategies_
module, 22
e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strategies_
module, 22
e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strategies_
module, 22
e3f2s.simulator.simulation_input.sim_configs_versioned.xian.turin_test
module, 23
e3f2s.simulator.simulation_input.sim_configs_versioned.xian.turin_test_conf
module, 23
e3f2s.simulator.simulation_input.sim_input
module, 23
e3f2s.simulator.simulation_input.station_conf
module, 23
e3f2s.simulator.simulation_input.vehicle_conf
module, 23
e3f2s.simulator.simulation_output
module, 24
e3f2s.simulator.simulation_output.multiple_runs_plotter
module, 23
e3f2s.simulator.simulation_output.plot_multiple_runs
module, 24
e3f2s.simulator.simulation_output.sim_output
module, 24
e3f2s.simulator.simulation_output.sim_output_plotter
module, 24
e3f2s.simulator.simulation_output.sim_stats
module, 24
e3f2s.simulator.single_run
module, 25
e3f2s.supply_modelling
module, 26
e3f2s.supply_modelling.charging_station
module, 25
e3f2s.supply_modelling.energymix_loader
module, 25
e3f2s.supply_modelling.supply_model
module, 25
e3f2s.supply_modelling.supply_model_conf
module, 25
e3f2s.supply_modelling.vehicle
module, 26
e3f2s.utils
module, 27
e3f2s.utils.bookings_utils
module, 26
e3f2s.utils.cdst_utils
module, 26
e3f2s.utils.path_utils
module, 27
e3f2s.utils.time_utils
module, 27
EFFCS_MultipleRunsPlotter (*class in e3f2s.simulator.simulation_output.multiple_runs_plotter*), 23
EFFCS_SimConfGrid (*class in e3f2s.simulator.simulation_input.sim_config_grid*), 23
EFFCS_SimOutputPlotter (*class in e3f2s.simulator.simulation_output.sim_output_plotter*), 24
EnergyMix (*class in e3f2s.supply_modelling.energymix_loader*), 25
evaluate_emissions() (*EnergyMix method*), 25
evaluate_energy() (*EnergyMix method*), 25

F

from_kml_to_energyperkm() (*Vehicle method*), 26
from_kml_to_lkm() (*Vehicle method*), 26

G

I

L

M

e3f2s.simulator.simulation_input.sim_con[illegible] 20
e3f2s.simulator.simulation_input.sim_con[illegible] 20
e3f2s.simulator.simulation_input.sim_con[illegible] 20
e3f2s.simulator.simulation_input.sim_con[illegible] 20
e3f2s.simulator.simulation_input.sim_con[illegible] 20
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 20
e3f2s.simulator.simulation_input.sim_con[illegible] 20
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 22
e3f2s.simulator.simulation_input.sim_con[illegible] 22
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 21
e3f2s.simulator.simulation_input.sim_con[illegible] 22
e3f2s.simulator.simulation_input.sim_[illegible] 22
e3f2s.simulator.simulation_input.sim_configs_versioned.test.mito_mobility_test, 22
e3f2s.simulator.simulation_input.sim_configs_versioned.test.mito_mobility_test.sim_gene 22
e3f2s.simulator.simulation_input.sim_configs_versioned.test.mito_mobility_test.single_r 22
e3f2s.simulator.simulation_input.sim_configs_versioned.xian, 23
e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strateg 22
e3f2s.simulator.simulation_input.sim_configs_versioned.xian.charging_relocation_strateg 22

e3f2s.[illegible] 22
e3f2s.[illegible] 23
e3f2s.[illegible] 23
e3f2s.[illegible] 23
e3f2s.[illegible] 23
e3f2s.[illegible] 23
e3f2s.[illegible] 24
e3f2s.[illegible] 23
e3f2s.[illegible] 24
e3f2s.[illegible] 24
e3f2s.[illegible] 24
e3f2s.[illegible] 24
e3f2s.[illegible], 25
e3f2s.supply_modelling, 26
e3f2s.[illegible] 25
e3f2s.[illegible]ing.energymix_loader, 25
e3f2s.[illegible] 25
e3f2s.[illegible] 25
e3f2s.[illegible], 26
e3f2s.utils, 27
e3f2s.[illegible], 26
e3f2s.utils.cost_utils, 26
e3f2s.[illegible], 27
e3f2s.utils.time_utils, 27
monitor() (*ChargingStation method*), 16

N

normalise() (*AustinScooterTrips method*), 10
normalise() (*BigDataDBTrips method*), 10
normalise() (*CalgaryScooterTrips method*), 11
normalise() (*ChicagoScooterTrips method*), 11
normalise() (*LouisvilleScooterTrips method*), 11
normalise() (*MinneapolisScooterTrips method*), 12
normalise() (*TripsDataSource method*), 2, 13

O

open_database() (*in module e3f2s.supply_modelling.energymix_loader*), 25

P

R

S

T

U

V

W

Z